Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

An algorithm for integrating peer-to-peer ridesharing and schedule-based transit system for first mile/last mile access

Pramesh Kumar, Alireza Khani [*]

Department of Civil, Environmental and Geo-Engineering, University of Minnesota, Twin Cities, MN, United States

A B S T R A C T

Due to limited transit network coverage and infrequent service, suburban commuters often face the transit first mile/last mile (FMLM) problem. To deal with this, they either drive to a park-and-ride location to take transit, use carpooling, or drive directly to their destination to avoid inconvenience. Ridesharing, an emerging mode of transportation, can solve the transit first mile/last mile problem. In this setup, a driver can drive a ride-seeker to a transit station, from where the rider can take transit to her respective destination. The problem requires solving a ridesharing matching problem with the routing of riders in a multimodal transportation network. We develop a transit-based ridesharing matching algorithm to solve this problem. The method leverages the schedule-based transit shortest path to generate feasible matches and then solves a matching optimization program to find an optimal match between riders and drivers. The proposed method not only assigns an optimal driver to the rider but also assigns an optimal transit stop and a transit vehicle trip departing from that stop for the rest of the rider's itinerary. We also introduce the application of space-time prism (STP) (the geographical area which can be reached by a traveler given the time constraints) in the context of ridesharing to reduce the computational time by reducing the network search. An algorithm to solve this problem dynamically using a rolling horizon approach is also presented. We use simulated data obtained from the activity-based travel demand model of Twin Cities, MN to show that the transit-based ridesharing can solve the FMLM problem and save a significant number of vehicle-hours spent in the system.

1. Introduction

Due to a rise in the number of motor vehicles resulting from increasing travel demand, urban highways are facing an inescapable condition of traffic congestion. The congestion on roads can be primarily ascribed to an increase in the use of a personal vehicle for traveling. Besides, we have observed that the vehicle capacity is often under-utilized, e.g., National Household Travel Survey 2009 shows that on an average, only 1.7 out of 4 available seats in cars were utilized (Santos et al., 2009; Masoud and Jayakrishnan, 2017a). As the aim is to transport people, not cars, an alternative way to provide mobility is to make efficient use of existing transportation infrastructure. Public transportation, which can carry multiple passengers, is widely considered as a practical solution to the congestion problem by reducing vehicle-miles traveled (VMT) on roads (Aftabuzzaman et al., 2015).

To use public transit for travel, one has to walk from origin to a boarding stop to access the service and then walk again from

---

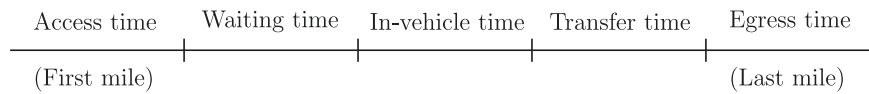| Access time	Waiting time	In-vehicle time	Transfer time	Egress time
(First mile)				(Last mile)

Fig. 1. Components of a tansit trip.

alighting stop to the respective destination. Formally, these walking components of a transit trip are known as "first mile" and "last mile," which describe the beginning and end of an individual transit trip (Fig. 1) respectively. Due to limited transit coverage, it is sometimes difficult or impossible to walk to/from transit stop on either end of a transit trip. This inaccessibility problem is also known as the *first mile/last mile (FMLM) problem for transit*. The problem is common among people commuting from low-density areas where transit service is not available or less frequent. The limited coverage in these areas is because of the economic in-viability of providing transit service. To encourage people living in suburban areas to use transit to commute, transportation agencies need to provide an effective transportation service that can help them to cover their first/last mile. Various possible solutions to this problem are driving to a park-and-ride facility, use para-transit services, taxi/carpooling, or bike to a transit station. Although these solutions to the FMLM problem are feasible, they are either not attractive among commuters or too expensive for daily travel, which drives travelers to use personal vehicle for their daily commute. A cheaper solution is to use the existing trips as a feeder service to transit stations. Ridesharing, which has attracted attention during recent years, can be a potential solution to the transit FMLM problem. In this program, both riders and drivers can submit a request in real-time, and an automated system will match them for sharing a ride while satisfying the spatial and temporal constraints for both

The ridesharing programs are becoming popular in cities (Agatz et al., 2011; Masoud et al., 2017). However, a few studies such as Masoud et al. (2017) have raised a concern that this increase in the participation in ridesharing programs is shifted from the transit ridership. Besides, Wang et al. (2018), using a ridesharing equilibrium model, showed that when driving is faster and more expensive than transit, then no cost-sharing strategy can sustain a ridesharing program without shifting the demand from transit mode. In that case, ridesharing, which was aimed to reduce congestion on roads, may not be as effective as anticipated, and it will become a competitor of transit mode. However, the equilibrium conditions proved in that study is a function of the matching probability and does not consider the integration of ridesharing and transit mode. A ridesharing program integrating transit with ridesharing can help in shifting the share from single-occupancy vehicles and hence increase transit ridership. In this program, ridesharing can provide a fast and reliable mobility service in low-density areas, and transit can provide mobility in high-density areas.

In this article, we develop a framework to integrate ridesharing with transit to solve the FMLM problem. Riders who are looking for an affordable ride to a transit station and drivers who want to join this program to alleviate the negative impacts of car travel and to receive a small compensation can submit their request to an automated system. The framework consists of the development of an algorithm for transit passenger routing in a schedule-based transit network and a matching optimization program to match riders and drivers up to the first mile of the transit trip. Overall, this article makes the following contributions:

1. Propose the operation of a large scale transit-based ridesharing program,
2. Development of a schedule-based transit network algorithm for finding feasible matches,
3. Incorporating the state-of-the-art network search reduction techniques to improve the computational time without compromising the solution quality,
4. Development of a *rolling horizon* algorithm to solve the dynamic transit ridesharing problem,
5. Use simulation to assess the efficiency of the proposed method on a large scale transit network of Twin Cities, MN

The rest of the article is structured as follows. Section 2 reviews previous work on transit FMLM problem and ridesharing matching algorithms, which is followed by the motivation for this research. Section 3 introduces the terms and definitions used in this article and describes the operation of the transit-based ridesharing program. Then, Section 4 formally defines the transit-based ridesharing matching problem. After that, an algorithm to find potential matches and a matching optimization model is developed in Section 5, which is followed by the development of a rolling horizon algorithm. Then, Section 6 presents the results of simulation experiments and Section 7 describes ways to relax various assumptions in this study. Finally, conclusions and recommendations for future research are given in Section 8.

2. Related work and motivation for this study

Previous studies have found that access to transit is a significant factor affecting its modal share (Moniruzzaman and Páez, 2012; Brons et al., 2009; Kalaanidhi and Gunasekaran, 2013). Therefore, in areas with limited network coverage, it is crucial to provide a feeder service to attract riders to use transit service. Various modes of transport have been studied and implemented to improve access to transit mode. This includes integrating bikesharing and transit (Martin and Shaheen, 2014; Rietveld, 2000; Martens, 2004), designing a demand responsive transit feeder service to solve the FMLM problem (Wang, 2017; Maheo et al., 2017; Cayford and Yim, 2004; Koffman, 2004; Lee and Savelsbergh, 2017; Quadrifoglio et al., 2008; Shen and Quadrifoglio, 2012; Li and Quadrifoglio, 2009), use of park-and-ride facilities (Nassir et al., 2012; Khani et al., 2012; Webb and Khani, 2020), and integrating ridesharing and transit (Masoud et al., 2017; Stiglic et al., 2018; Bian and Liu, 2019a; Ma et al., 2019; Chen et al., 2020). We summarize the literature related to transit-based ridesharing into three main categories.

1. *Designing a feeder service to the transit mode*: The transit-based ridesharing problem resembles similarities with the literature focused on designing a feeder service to the transit mode such as designing a demand responsive connector (DRC) (Cayford and Yim, 2004; Koffman, 2004; Lee and Savelsbergh, 2017; Quadrifoglio et al., 2008; Shen and Quadrifoglio, 2012; Li and Quadrifoglio, 2009) or dial-a-ride service with transfers (DART) (Masson et al., 2014; Deleplanque and Quilliot, 2013; Jafari et al., 2016; Häll et al., 2009). Cayford and Yim (2004) designed a demand-responsive dial-a-ride service in California. The service was designed to be fixed-route during high demand and flexible-route during low demand period. A transit cooperative research program (TCRP) report also presents experiences from flexible transit services in various cities in the United States (Koffman, 2004). While designing vehicle-scheduling for a demand responsive system, many studies consider zoning in which a vehicle would operate within the zonal boundary to pick up the passengers and drop them off at a predetermined transit station (Quadrifoglio et al., 2008; Shen and Quadrifoglio, 2012; Li and Quadrifoglio, 2009). Mahéo et al. (2019) proposed the design of a hub and shuttle public transit system in Canberra. Wang (2017) proposed a mixed-integer linear program and heuristic techniques to design routing and scheduling for a last-mile transportation service. However, the transit-based ridesharing problem we consider in this article, is different from the ones discussed above in the following aspects. First, in ridesharing, the drivers are not hired to work for a private entity, so the departure and arrival time constraints of both drivers and riders need to be incorporated in the model. Second, we cannot assume an unlimited supply of vehicles to serve rider trips (Lee and Savelsbergh, 2017). Third, ridesharing problem is highly dynamic in nature, where drivers and riders enter the system on short notice, it is difficult to solve a possible discrete optimization model proposed in the studies cited above in real-time.

2. *Ridesharing matching algorithms*: The current study proposes a matching algorithm for peer-to-peer (P2P) ridesharing as an alternative mode of transportation to access transit. The ridesharing matching problem has recently got the attention of many researchers. Agatz et al. (2012) and Furuhata et al. (2013) reviewed various forms of ridesharing and outlined the difficulties that arise when developing a mechanism for such a system. Agatz et al. (2011) developed a simple optimization model for matching single driver and rider while satisfying their time budget constraints and keeping the driver or rider role flexible. Stiglic et al. (2015) pointed out the benefits of meeting points in ridesharing. Santi et al. (2014) proposed the idea of a shareability network to reduce the size of the matching problem. As the method was only able to match two riders (optimally) and three riders (heuristically), Alonso-mora et al. (2017) improved this idea by creating a request-to-vehicle (RV) and a request-to-trip-to-vehicle (RTV) graph to solve a high capacity ridesharing problem. Masoud and Jayakrishnan (2017a,b) presented a solution to the general peer-to-peer multiple matching problem with transfers in a time-dependent network. They proposed a decomposition algorithm and a dynamic programming approach to solve the problem and showed an increase in the total number of matching by using their algorithm. Ordóñez et al. (2017),Sutherland and Jarrahi (2018), and Tafreshian et al. (2020) present recent reviews on matching algorithms, cost-sharing mechanisms, and traffic equilibrium related to ridesharing.

3. *Integrating ridesharing with transit*: To integrate ridesharing with transit, Masoud et al. (2017) used the same dynamic programming framework as in Masoud and Jayakrishnan (2017b) to solve the current problem by introducing the transit stops and transfers as go-to-points and transit route as an inflexible driver in their algorithm. They considered one transit route and limited the number of transfer points to 40 to reduce the computational time. The major obstacle to applying their algorithm to a large-scale transit system would be high computational time. Ma et al. (2019) proposed various queueing-theoretic vehicle dispatch and idle vehicle relocation algorithms for ridesharing with a possible drop-off of a passenger at a transit stop. Bian and Liu (2019a,b) developed a matching optimization program and a Vickrey-Clarke-Groove (VCG) mechanism to determine the optimal vehicle-passenger matching, vehicle routing plan, and a customized pricing scheme respectively. The proposed mechanism is proved to be individual rational, incentive compatible, and price non-negative. Chen et al. (2020) developed a mixed-integer linear programming (MILP) to solve the FMLM with autonomous vehicles. A pioneering effort in developing a matching algorithm for this multimodal system is described by Stiglic et al. (2018). They presented a matching framework to match riders and drivers for transit, park-and-ride, or ridesharing mode. However, they considered a cyclic (frequency-based) time table for transit service, assumed the closest stop to the destination as the alighting stop, and selected a transit trip with the least driving distance to make the complex problem easier to solve.

2.1. Motivation

The following points motivate us to pursue emerging research on transit-based ridesharing problem:

1. *Complexity of a schedule-based (SB) transit service*: Previous studies such as Stiglic et al. (2018), Masoud and Jayakrishnan (2017b) have considered a cyclic (or frequency-based) transit service while designing a method for transit-based ridesharing. In the case of a cyclic timetable with identical trains, the travel time is not dependent on the departure time of the train from a transit station. However, large-scale transit systems incorporate traffic congestion and transit demand (to evaluate dwell time) while designing the schedule that usually varies during the day. Incorporating schedule becomes even more important when the transit service is not frequent, e.g., at park-and-ride stations, where buses arrive almost every hour. In that case, a frequency-based approach will result in an inaccurate estimation of wait time. The correct estimation of wait time, walking time, and transfer time are important to assess the feasibility of matches. We address this issue by proposing a systematic schedule-based ridesharing algorithm to find feasible matches. Furthermore, this would help in developing a transit-based ridesharing app that can provide directions in a multimodal transportation network.
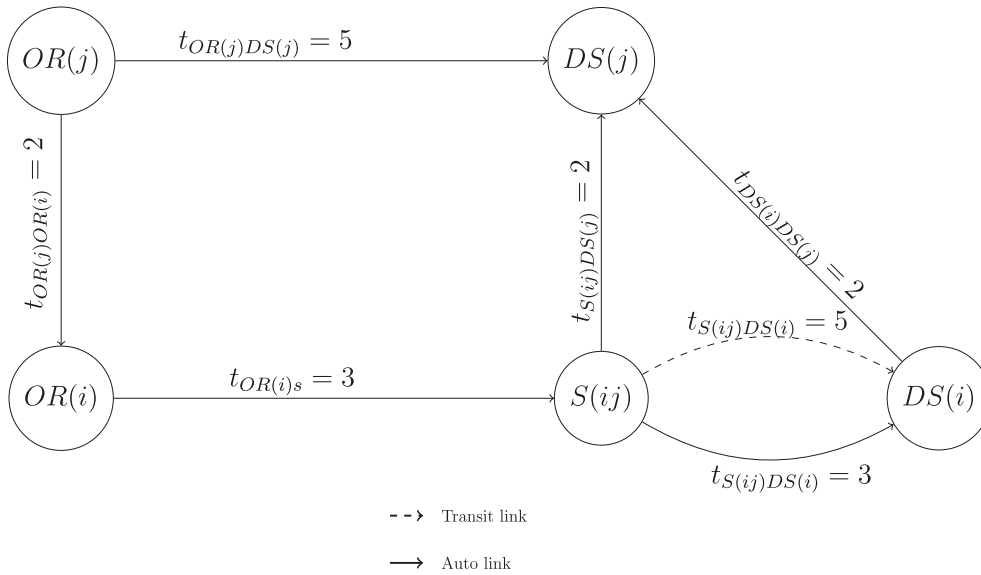
Fig. 2. An example of transit-based ridesharing.

2. *Drop-off station*: Previous studies have considered the closest stop from origin and destination for the drop-off and alighting (respectively) of a passenger. However, the closest stop may not give access to a faster transit route. Our proposed algorithm considers all the possible boarding and alighting stops. This will improve the possibility of assigning a faster service to the passenger.

3. *Storage complexity*: Previous studies have used stored shortest transit travel itineraries. However, it is difficult to do so for an SB transit network due to huge storage cost. For example, the Twin Cities transit system has 13673 stops in the network and for 1440 possible departure times in a day, we need to store $2.1e^{12}$ possible travel times and itineraries. Furthermore, the change in the schedule for various days (weekdays, weekends, or holidays) makes it cumbersome to store such information. One of the contributions of this study is to avoid such storage with transit direction computation for different transit schedules using the proposed algorithm.

All the above-cited studies have made important contributions to this challenging problem, however, none of the studies has considered the complexities of the transit part of the trip and showed application on a large-scale transit network. The current study leverages the schedule-based transit shortest path (SBTSP) algorithm to find potential matches between riders and drivers. The proposed algorithm uses the concept of space-time prism of both rider and driver to reduce the size of the network search. To solve this problem in real-time, a practical algorithm using a rolling horizon approach is proposed in this paper. The proposed method not only matches riders and drivers for the first/last mile of the rider's trip but also assigns an optimal transit station and a transit vehicle trip from the schedule for the rest of the rider's itinerary.

3. Preliminaries

In this section, we discuss operation of a transit-based ridesharing program and describe notations and concepts useful to understand the matching problem. All the notations used in this article are also summarized in Table 6. The ridesharing program receives a set of requests ($P = R \cup D$) that can be partitioned into a set of riders R, who are looking for a ride and have first mile/last mile problem, and a set of drivers D, who are willing to give a ride to the ride seekers to a transit station. Each request $p \in P$ is defined by a tuple $\{OR(p), DS(p), \tau^{at}(p), \tau^{pd}(p), \tau^{pa}(p), \delta(p), \Delta(p)\}$, indicating its origin $OR(p)$, destination $DS(p)$, announcement time $\tau^{at}(p)$, preferred departure time from origin $\tau^{pd}(p)$, preferred arrival time at destination $\tau^{pa}(p)$, maximum acceptable schedule deviation $\delta(p)$, and total travel delay $\Delta(p)$ allowed in the trip. Let t_{ij} be the travel time from location i to location j. Using above information, the earliest ($\tau^{ed}(p)$) and latest ($\tau^{ld}(p)$) departure time from origin, earliest ($\tau^{ea}(p)$) and latest ($\tau^{la}(p)$) arrival time at destination, and maximum ride time ($t_{max}(p)$) can be calculated as:

$$\tau^{ed}(p) = \tau^{pd}(p) - \delta(p) \tag{1a}$$

$$\tau^{ld}(p) = \tau^{pd}(p) + \delta(p) \tag{1b}$$

$$\tau^{ea}(p) = \tau^{pa}(p) - \delta(p) \tag{1c}$$

$$\tau^{la}(p) = \tau^{pa}(p) + \delta(p) \tag{1d}$$

$$t_{max} = t_{OR(p),DS(p)} + \Delta \tag{1e}$$

In this ridesharing program, there is an online platform where riders and drivers can submit their requests, and the system would match these riders and drivers to travel together. A rider is matched with some driver who can drop off the rider at a transit station, from where the rider can take a bus or train to their destination. To get an insight into the problem, let us consider the following instance (see Fig. 2):

A driver $j \in D$ is going from $OR(j)$ to $DS(j)$, which takes about 5 min by driving. A rider $i \in R$, going from $OR(i)$ to $DS(i)$ is facing the first-mile access problem and cannot take transit, so i must drive to the destination, which takes 6 min of driving time. However, if j agrees to give a ride to i, then j can drop i off at a station $S(ij)$, from where i can take transit to her destination and j can drive to her destination. The overall auto vehicle-minutes spent on the combined trip with transit-based ridesharing is 7 min in comparison to $5 + 6 = 11$ min without ridesharing and 10 min with only ridesharing (i.e., driver j going to rider's destination $DS(i)$ and then going to her destination $DS(j)$). Due to time constraints, it might not be possible for a driver to give a ride all the way to the rider's destination but can only give a ride to a transit station. This example shows that transit-based ridesharing can save a lot more vehicle-min than any other mode considered here and possibly more matching rate than a stand-alone ridesharing program.

3.1. Network topology

In this section, we describe the notations related to graphs characterizing the road and transit network as these notations will be used throughout the text.

3.1.1. Road network
Let $G_R(N_R, A_R)$ be a digraph representing road network, where N_R is the set of nodes and A_R is the set of links connecting these nodes. We consider a static or non-timed road network for this study. Let $t : N_R \times N_R \to \Re_+$ be a function that returns the shortest travel time between two nodes in the road network. The efficient shortest path algorithms (such as Dijkstra's algorithm using binary heap data structure) take a fraction of a second to compute the shortest path on a network with thousands of nodes and links. For this network, we store the shortest-path trees in the memory (storage cost $\mathcal{O}(N_R^2)$) and call it whenever required by the matching algorithm. Let us denote the set of origins and the destinations of the participants as $O = \cup_{p \in P} OR(p)$ and $D = \cup_{p \in P} DS(p)$ respectively. Note that $O \cup D \subseteq N_R$.

3.1.2. Transit network
A schedule-based (SB) transit network can capture the complexities of the movement of a passenger such as precise waiting time, in-vehicle time, and transfer time to other routes. This network $G_T(N_T, A_T)$ is a digraph that is created using service schedule data obtained from transit agencies. The General Transit Feed Specification (GTFS) is a standard format of transit schedule data released publicly by various transit agencies throughout the world. We use trip-based network representation for modeling the transit network (Khani et al., 2014). In this procedure, the main consideration is given to the fact that every deviation from a transit route could not be considered as a transfer. Depending upon the acceptable walking time, waiting time, and direction of movement, transfers are created between two transit routes. The details about the creation of this network are described in the next paragraph.

In the schedule data, let us denote the set of transit stops[1] as B, set of transit routes/lines[2] as L, and set of trips[3] as K. Each trip $k \in K$ is characterized by a route $l_k \in L$, set of stops (or nodes) $B_k \subset B \times K$, scheduled arrival/departure time $\tau : B_k \mapsto \Re_+$ at these stops, and a sequence $\zeta : B_k \mapsto \Re$ in which these stops are visited. The set of nodes in the transit network are defined as $N_T = \cup_{k \in K} B_k$. The set of links in the transit network, $A_T = A_{T_a} \cup A_{T_v} \cup A_{T_w}$ consists of three types of links, namely, acess/egress links A_{T_a}, in-vehicle links A_{T_v}, and walking/waiting transfer links A_{T_w}. Let $w : (N_T \cup N_R) \times (N_T \cup N_R) \mapsto \Re_+$ be the walking distance between two nodes. The access/egress arcs are walking links created between origins and transit stops or between transit stops and destinations if the distance between a pair is less than an acceptable walking distance w_0 (say 0.75 mi), i.e., $A_{T_a} = \{(i,j) : w(i,j) \leqslant w_0 \text{ for some } i \in O, j \in N_T\} \cup \{(i,j) : w(i,j) \leqslant w_0 \text{ for some } i \in N_T, j \in D\}$. The in-vehicle links $A_{T_v} = \{(i,j) : i,j \in B_k \text{ for some } k \in K\}$ are created using itinerary of any transit trip. Finally, the waiting/walking transfer links are defined as $A_{T_w} = \{(i,j) \mid i \in B_k, j \in B_{k'} \text{ for some } k, k' \in K, l_i \neq l_j, \zeta(i) \neq 1, \zeta(j) \neq \max_{m \in B_{k'}} \zeta(m), w(i,j) \leqslant w_1, |\tau(j) - \tau(i) - w(i,j)| \leqslant \rho\}$, where ρ is the threshold by which a bus trip can be early or late, and w_1 is the acceptable walking distance (say 0.25 mi) for transfers.

Finally, let $\mathcal{Z} : N_T \mapsto N_R$ be a function which outputs closest road node for any transit node, i.e., $\mathcal{Z}(s) = \text{argmin}\{w(n,s) : n \in N_R\}$. Let $A_m = \{(i,j) : i = \mathcal{Z}(j) \text{ for some } j \in N_T\}$ be the mode transfer links which are used to access the transit network from a road node and vice versa. The overall multimodal network is denoted by $G(N, A)$, where $N = N_R \cup N_T$, and $A = A_R \cup A_T \cup A_m$.

---

[1] A transit stop is a geographic location from where a traveler can board/alight the bus.
[2] A transit line is defined by a set of stops with a starting and an end point between which buses run back and forth.
[3] A trip is a travel itinerary of a bus with arrival and departure time specified at different stops.
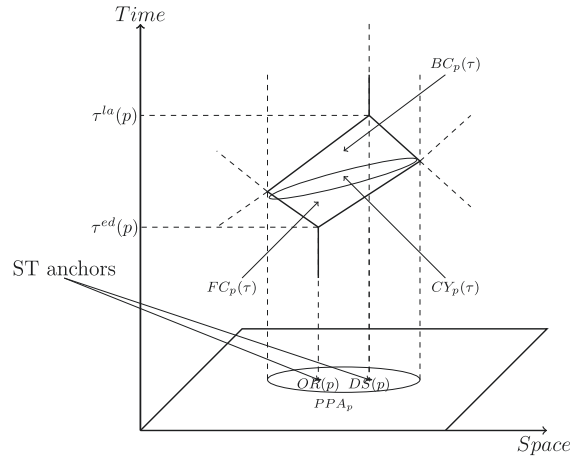
Fig. 3. Space-time prism ($STP(p)$) for participant p.

3.2. Space-time prism

Before delving into the matching problem, let us introduce a few more definitions which will help create a network search reduction technique later in the Section 5. The following definitions are given in the context of ridesharing, however, more general definitions can be found in Miller (2017).

Because of spatial and temporal limitations, a traveler can only be present in one place at a time due to which the activities performed by a traveler in space are restricted by the available time budget. Space-time prism provides us a framework that recognizes underlying constraints on human activities in space-time and also provides us an effective way of keeping track of these conditions.

Definition 1. (*Anchor*)   An anchor is a node in the network at which a traveler begins/ends her journey. For a participant $p \in P$ in the ridesharing program, $OR(p)$ and $DS(p)$ are possible anchors.

Definition 2. (*Time-forward and time-backward cone*)   The time-forward cone, $FC_p(\tau)$, for a participant $p \in P$ pointed at the anchor $OR(p)$ is the set of nodes that can be reached at time τ within a given time budget. Conversely, the time-backward cone, $BC_p(\tau)$, for a participant $p \in P$ pointed at the anchor $DS(p)$ is the set of nodes from where $DS(p)$ can be reached within the remaining time budget $\tau^{la}(p) - \tau$. Mathematically,

$$FC_p(\tau) = \{n : \ \tau \geqslant \tau^{ed}(p) + t_{OR(p)n}, \tau \leqslant \tau^{la}(p)\} \tag{2a}$$

$$BC_p(\tau) = \{n : \ \tau \leqslant \tau^{la}(p) - t_{nDS(p)}, \tau \geqslant \tau^{ed}(p)\} \tag{2b}$$

Definition 3. (*Space-time prism*)   The space time prism, $STP(p)$, for a participant $p \in P$ is defined as

$$STP_p(\tau) = FC_p(\tau) \cap BC_p(\tau) \tag{3}$$

Definition 4. (*Potential path area*)   The space that is accessible to a participant within a given time budget is known as potential path area (PPA).

$$PPA_p = \{n : t_{OR(p)n} + \ t_{nDS(p)} \leqslant \tau^{la}(p) - \tau^{ed}(p)\} \tag{4}$$

The anchors, time-forward cone, time-backward cone, potential path area, and space-time prism are visualized in Fig. 3. The constraints defining the space-time prism can be used to reduce the network search for an optimal itinerary.

Proposition 1. (*Potential Path Area*)   *The projection of space-time prism onto space is the potential path area PPA_p for participant p.*

Proof.   Using (2a) and (2b), we have

$$-\tau \leqslant -\tau^{ed}(p) - t_{OR(p)n} \tag{5}$$
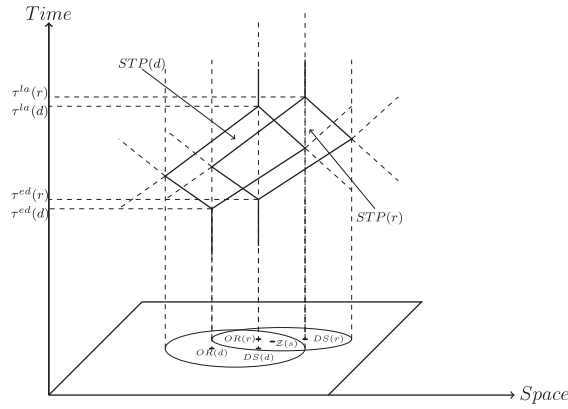
$$\tau \leqslant \tau^{la}(p) - t_{nDS(p)} \tag{6}$$

Fig. 4. Feasible region where r and d can meet.

Adding (5) and (6) yields,

$$\tau^{la}(p) - \tau^{ed}(p) - t_{nDS(p)} - t_{OR(p)n} \geq 0$$
$$\Rightarrow \quad \tau^{la}(p) - \tau^{ed}(p) \geq t_{OR(p)n} + t_{nDS(p)}. \tag{7}$$

which is same as (4). $\square$

4. Transit-based ridesharing matching problem

In this section, we define the transit-based ridesharing problem. We make several assumptions that help in defining our problem.

4.1. Assumptions

1. All buses and trains are assumed to have sufficient capacity. Passengers do not face denied boarding due to capacity constraints.
2. Transit service is reliable, which means that all buses and trains arrive on time according to their schedule.
3. Time spent in boarding and alighting a transit vehicle or getting on or off a car is negligible and can be incorporated as a service time (t^{ser}). The service time can also be used to account for delays in the schedule.
4. Transfer of a passenger between several drivers (multi-hop) is not allowed as this can be inconvenient for a rider and reduce the attractiveness of the program.

4.2. Problem statement

The transit-based ridesharing matching problem is defined as follows. Given a set of requests P, a road network G_R, and a transit network G_T, find the following:

1. An optimal match between drivers and riders for ridesharing so that a given objective function is optimized, and
2. Optimal drop off transit stops location $s \in B$ for the riders, and
3. Optimal transit itinerary $\mathcal{I}$ for the riders.

Definition 5. (*Feasible match*)   A match $(r,d,s,\mathcal{I})$ between a rider r and driver d, drop off stop location s, and transit trip itinerary $\mathcal{I}$ is feasible if and only if it satisfies the following constraints:

1. d should depart from $OR(d)$ after $\tau^{ed}(d)$
2. d should reach at $OR(r)$ after $\tau^{ed}(r)$
3. r should reach at $DS(r)$ before $\tau^{la}(r)$
4. d should reach at $DS(d)$ before $\tau^{la}(d)$
5. $\mathcal{I}$ should be optimal for r i.e., best possible transit path from stop s consisting of different weights to transit trip components

A feasible match between a driver and a rider must satisfy time constraints associated with the earliest departure and latest arrival time. They restrict passenger access to each node in the network at different times. The space-time prism in (3) captures above four conditions. By making use of it, we make the following proposition.

Proposition 2. *A driver $d \in D$ and a rider $r \in R$ can share a ride together if and only if $OR(r) \in STP_r(\tau) \cap STP_d(\tau)$ as well as $\mathcal{Z}_{(s_k)} \in STP_r(\tau) \cap STP_d(\tau)$, where $s_k \in N_T$ is the drop-off node of r.*

Proof. ($\Rightarrow$) We prove this by contradiction. Let $J = \{OR(r), \mathcal{Z}(s_k)\}$. Let us assume that $J \subsetneq STP_d(\tau)$, then for some $n \in J, n \notin PPA_d$, i.e., $\tau^{ed}(d) + t_{OR(d)n} + t_{nDS(d)} > \tau^{la}(d)$, which means d cannot reach to her destination in time, thus violating the time constraints which is a contradiction according to (2a)–(4). Furthermore, $OR(r)$ is clearly in $STP(r)$ as this node is the anchor of time-forward cone. However, if $\mathcal{Z}(s_k) \notin STP_r(\tau)$, then, $\mathcal{Z}(s_k) \notin PPA_r$, i.e., $\tau^{ed}(r) + t_{OR(r)\mathcal{Z}(s_k)} + t_{s_kDS(r)} > \tau^{la}(r)$, resulting in the violation of time constraints for a feasible match.($\Leftarrow$) If $J \subseteq STP_r(\tau) \cap STP_d(\tau)$, then both nodes present in J are accessible by r and d without violating the time budget constraints for any of them, which means they can be present at these nodes at the same time, making it possible to share a ride. Therefore, the proposition holds. $\square$

Following Proposition (2), $STP_r(\tau) \cap STP_d(\tau)$ provides us the set of potential nodes in the network where r and d can be present at the same time τ. These nodes can be used for potential rideshare between r and d as shown in Fig. 4. This approach will greatly reduce the size of the network to be considered for ridesharing and thus reduce the computational time for finding a feasible match between a rider and driver.

5. Solving transit-based ridesharing matching problem

For simplicity, we present a methodology for riders facing the first-mile access problem, i.e., no transit access at the origin for inbound trips due to limited transit network coverage or infrequent service in suburbs but sufficient transit access to destinations in the city center. The procedure can be reversed for the last mile access problem (for outbound trips), a discussion on which is given in Section 5.1.1. The process of finding optimal matches for transit-based ridesharing follows a two-step procedure. The first step finds a set of feasible matches between drivers and riders, and in the second step, we use a matching optimization program to assign riders to drivers. They are described in detail below:

5.1. An algorithm to find feasible matches

The proposed method leverages *schedule-based transit shortest path* (SBTSP) in developing an algorithm for finding feasible matches. The quickest path in an SB transit network may not be optimal for a passenger. For example, the quickest path may have several transfers that are less attractive to a passenger. By assigning different weights to the cost of different links, SBTSP finds an optimal path according to user route choice behavior. For example, waiting and transfers are considered onerous components of a transit trip and hence can be assigned a higher weight. The developed algorithm returns a set of feasible matches for the riders and drivers participating in this program. We assume that the reader is familiar with shortest path labeling algorithms for the discussion to follow in the paragraph below:

Given that r departs from node $n \in N_T$, we find the labels γ_n that specify the latest time r should depart from n in order to reach $DS(r)$ before $\tau^{la}(r)$. Let SEL be a scan eligible list, ξ_i be the predecessor of node i in shortest path, $\Gamma^{-1}(i) = \{j \mid (j, i) \in A_T\}$ be the backward star of node i (i.e., collection of all adjacent nodes from where node i can be reached). We now define weights associated with different types of links in G_T for calculating a generalized cost. Let $\eta_a, \eta_v,$ and η_w be the weights associated with links in set $A_{Ta}, A_{Tv},$ and A_{Tw} respectively. Using weighted sum of the cost of traversing different types of link in the transit network, a generalized cost label γ_n^{gc} is maintained for each node $n \in N_T$.

Bellman's principle of optimality: For any node $i \in N_T, \gamma_i^{gc}$ should satisfy the following condition:

$$\gamma_j^{gc} = \min_{j \in \Gamma(i)}\{\gamma_i^{gc} + t_{ij}\} \tag{8}$$

Pseudocode for finding feasible matches is given in Algorithm 1. The algorithm starts with the initialization of a set of potential matches as $\mathcal{M} = \phi$. For every rider $r \in R$, we run a backward shortest path from $DS(r)$. We initialize a scan eligible list (SEL) and maintain two different types of labels–time labels γ and generalized cost labels γ^{gc}. The time labels are used to check if the time constraints are satisfied for both riders and drivers while the generalized cost labels γ^{gc} are used to maintain a minimum generalized cost for a rider which is calculated as the weighted sum of the cost of traversing different types of link in the transit network (line 11–16). Line 18 checks the Bellman's principle of optimality (8) and restrict the network search by using driving time as a lower bound on transit time from $OR(r)$ to node j. While updating labels of each node $j \in N_T$, each driver $d \in D$ is checked for her compatibility with given rider r. Line 23 checks if d can reach $OR(r)$ by $\tau^{ed}(r)$, then we further check if d can reach $DS(d)$ before $\tau^{la}(d)$. In other words, these two *if* statements check if $j \in STP_d(\gamma) \cap STP_r(\gamma)$. If all these conditions are satisfied, then we add (r, d, j, ξ) to $\mathcal{M}$ to $\mathcal{M}$. The node j and predecessor set ξ can be used to retrieve the drop off station location s and optimal itinerary $\mathcal{I}$ for r. The procedure will be reversed for the last mile problem, in which case, we run the algorithm in the forward direction.

Algorithm 1.   Pseudocode for finding rider driver feasible matches

---

1: **procedure** RIDERDRIVERPOTENTIALMATCH(R, D, G, η)
2:   $\mathcal{M} \leftarrow \{\}$ ▷ intializing set of potential matches
3:   **for** r in R **do**
4:     $i \leftarrow DS(r), \gamma_i \leftarrow \tau^{la}(r), \gamma_i^{gc} \leftarrow 0, \xi_i = $ NULL
5:     $\gamma_j \leftarrow \infty, \gamma_j^{gc} \leftarrow \infty, \xi_j \leftarrow \phi \quad \forall j \neq i$ ▷ intializing node labels and SEL
6:     SEL $\leftarrow \{i\}$
7:     **while** SEL $\neq \phi$ **do**
8:       $i \leftarrow \underset{k}{\operatorname{argmin}}\{\gamma_k^{gc} \mid k \in $ SEL$\}$
9:       SEL $\leftarrow$ SEL$\setminus\{i\}$
10:       **for** each $j \in \Gamma^{-1}(i)$ **do**
11:         **if** $(j, i) \in A_{T_a}$ **then**
12:           $\gamma_{new}^{gc} \leftarrow \gamma_i^{gc} + \eta_a * t_{ji}$
13:         **else if** $(j, i) \in A_{T_v}$ **then**
14:           $\gamma_{new}^{gc} \leftarrow \gamma_i^{gc} + \eta_v * t_{ji}$
15:         **else**
16:           $\gamma_{new}^{gc} \leftarrow \gamma_i^{gc} + \eta_w * t_{ji}$
17:         $\gamma_{new} \leftarrow \gamma_i - t_{ji}$
18:         **if** $\gamma_{new}^{gc} < \gamma_j^{gc}$ and $\tau^{ed}(r) + t_{OR(r)\mathcal{Z}(j)} + t^{ser} < \gamma_{new}$ **then**
19:           $\gamma_j^{gc} \leftarrow \gamma_{new}^{gc}, \gamma_j \leftarrow \gamma_{new}, \xi_j \leftarrow i$
20:           SEL $\leftarrow$ SEL $\cup\{j\}$ ▷ Updating labels
21:           **for** d in D **do**
22:             $\tau_{OR(r)} \leftarrow \tau^{ed}(d) + t_{OR(d)OR(r)}$
23:             **if** $\tau^{ed}(r) \leq \tau_{OR(r)} \leq \tau^{ld}(r)$ **then**
24:               **if** $\tau_{OR(r)} + t_{OR(r)\mathcal{Z}(j)} + t^{ser} \leq \gamma_j$ ▷ checking time constraints
25:                 and $\tau_{OR(r)} + t_{OR(r),\mathcal{Z}(j)} + t^{ser} + t_{\mathcal{Z}(j),DS(d)} \leq \tau^{la}(d)$ **then**
26:                 $\mathcal{M} \leftarrow \mathcal{M} \cup \{(r, d, j, \xi)\}$
    return $\mathcal{M}$

To illustrate the use of above algorithm, consider an example network given in Fig. 5. In this graph,

$$N_R = \{OR(r), OR(d), n_1, n_2, DS(r), DS(d)\}$$
$$B = \{s_1, s_2, s_3, s_4, s_5\}$$
$$N_T = \{u_1, u_2, u_3, u_4, u_5, u_6, u_1', u_2', u_3', u_4', u_5', u_6'\}$$

Assume $\tau^{ed}(d) = 0$ min, $\tau^{la}(d) = 20$ min, $\tau^{la}(r) = 40$ min, and $\tau^{ed}(r) = 10$ min. Suppose the driver d starts at $\tau^{ed}(d) = 0$ min and reach $OR(r)$ at 10 min, from where she has option to go to n_1 or n_2. However, going to n_2 and then $DS(d)$ will make the driver late as $\tau^{la}(d) = 20$ min. So d can drop-off r at n_1, from where rider can walk to station s_1. The service time in getting off and walking to the transit station is included in the access links. We assume weights to different types of transit links as $\eta_a = 1, \eta_w = 2, \eta_v = 1$. Now there are three possible paths to go from s_1 to $DS(r)$, namely

$$\pi_1 = \{(s_1, u_1), (u_1, u_1'), (u_1', s_3), (s_3, u_6), (u_6, u_6'), (u_6', s_4), (s_4, DS(r))\}$$
$$\pi_2 = \{(s_1, u_2), (u_2, u_2'), (u_2', s_4), (s_4, DS(r))\}$$
$$\pi_3 = \{(s_1, u_3,), (u_3, u_3'), (u_3', s_4), (s_4, DS(r))\}$$

The time of arrival at $DS(r)$ using π_1, π_2, and π_3 are 36 min, 38 min, and 44 min. Therefore, π_3 is not feasible as time of arrival at $DS(r)$ is greater than $\tau^{la}(r) = 40$ min. We can see that π_1 is the quickest path to $DS(r)$. However, the generalized cost to reach $DS(r)$ using π_1 and π_2 are 43 min and 40 min respectively. The higher generalized cost of π_1 can be attributed to transfers and the waiting involved. In real transit networks, sometimes this can greatly affect user perception and should be incorporated in shortest path calculations. Moreover, a path involving more transfers may not be reliable and can result in more waiting time than scheduled. Therefore, the rider would prefer to take π_2 to reach her destination.

Proposition 3. Algorithm 1 *terminates after a finite number of iterations and produces optimal transit path for riders.*
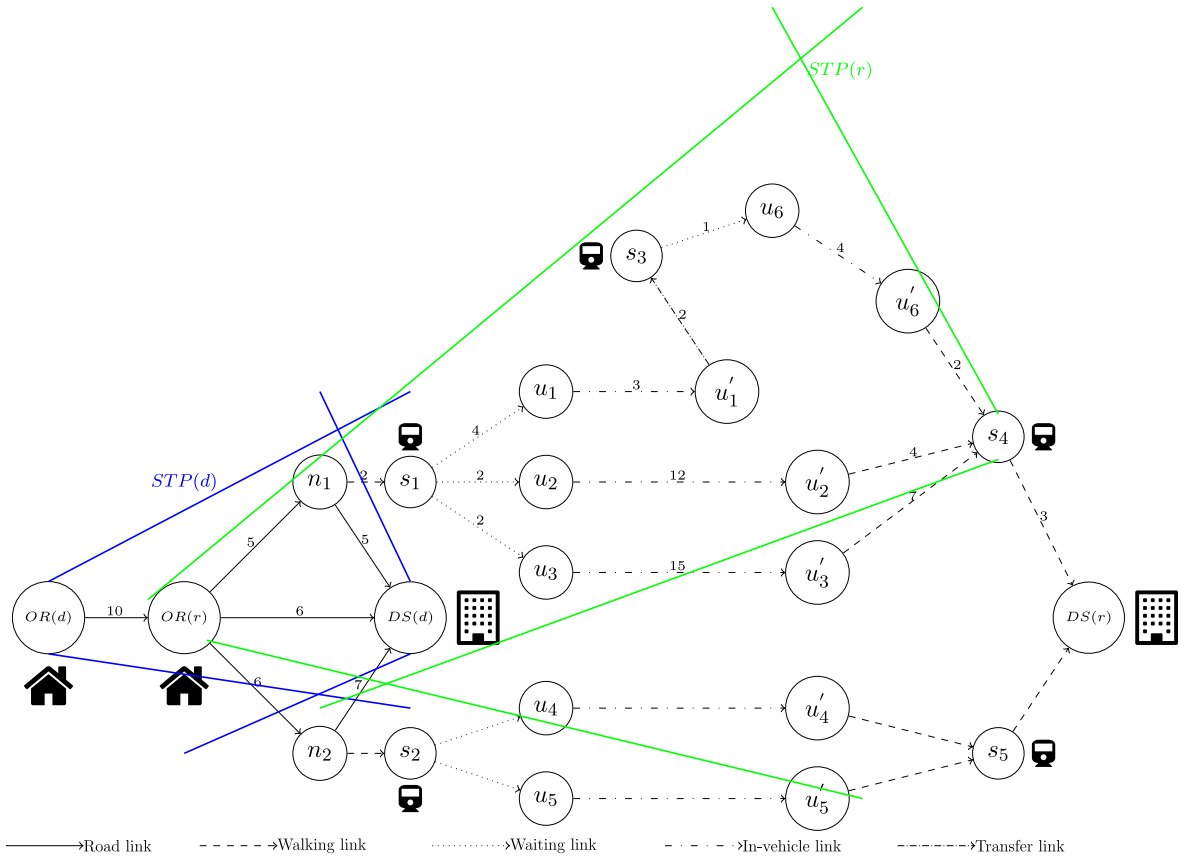
Fig. 5. An example network to illustrate the processing of Algorithm 1.

Proof. As the number of riders $|R|$ and drivers $|D|$ are finite in the program, the number of iterations by two *for* loops (Line 3 and 21) should be finite ($|R||D|$). While finding an optimal itinerary for a rider up to current node j, the labels cannot be unbounded as link costs are non-negative and any node enters SEL only once. Therefore, the algorithm terminates in the finite number of iterations. Also, the algorithm satisfies (8), hence the labels should be optimal. □

Remark 1. Algorithm 1 will not cut off any feasible solution if and only if $\eta_a = \eta_w = \eta_v = 1$

Consideration of varying weights to different links may cut off feasible solutions that are not attractive to the passenger. But if we want to consider all the feasible solutions, the coefficients should be $\eta_a = \eta_w = \eta_v = \eta_t = 1$. For practical purposes, this input should be user-specified.

Proposition 4. *The worst case computational complexity of Algorithm 1 is $\mathcal{O}(|R|(|D||A_T|\log|N_T| + |N_T|\log|N_T|))$.*

Proof. Lines 2, 4–6 can be done in $\mathcal{O}(1)$ time. Assuming that the shortest path algorithm is implemented using Binary heap data structure, Line 7–20 consists of two major steps, finding the node i with minimum label γ^{gc} from SEL (Line 8), which can be done in $\mathcal{O}(|N_T|\log(|N_T|))$ and updating labels of new nodes which can be done in $\mathcal{O}(|A_T|\log(|N_T|))$ time. While updating labels of each node j, they are checked for possible drop-off location for rider. The space-time prism constraints are incorporated in Line 18 and 22–25, can be done in $\mathcal{O}(1)$ time and are repeated for each driver $d \in D$. So, updating of labels should be multiplied with the cardinality of set D. Also, the shortest path algorithm is run for each $r \in R$, the overall complexity should be multiplied by the cardinality of set R. Hence, the worst case computational complexity of Algorithm 1 is $\mathcal{O}(|R|(|D||A_T|\log|N_T| + |N_T|\log|N_T|))$.

Remark 2. If $|D||A_T| = \Omega(|N_T|)$, then the worst case computational complexity of Algorithm 1 can be given as $\mathcal{O}(|R||N_T|\log|N_T|)$. However, the actual cost will be much lower than the worst case. For example, a small number of nodes will be added to SEL due to the strict constraint in line 18.

5.1.1. Last mile access problem

In this case, a passenger can access transit for the first mile but cannot reach her destination due to the unavailability of transit service to complete the last mile. This is a common problem faced by commuters going back to home from a high-density area to a low-density area. To solve this problem, a driver also going in the same direction can pick up the passenger from alighting stop and drop her off at the respective destination. Algorithm 1 can be modified by running the forward shortest path from $OR(r)$ and checking if driver
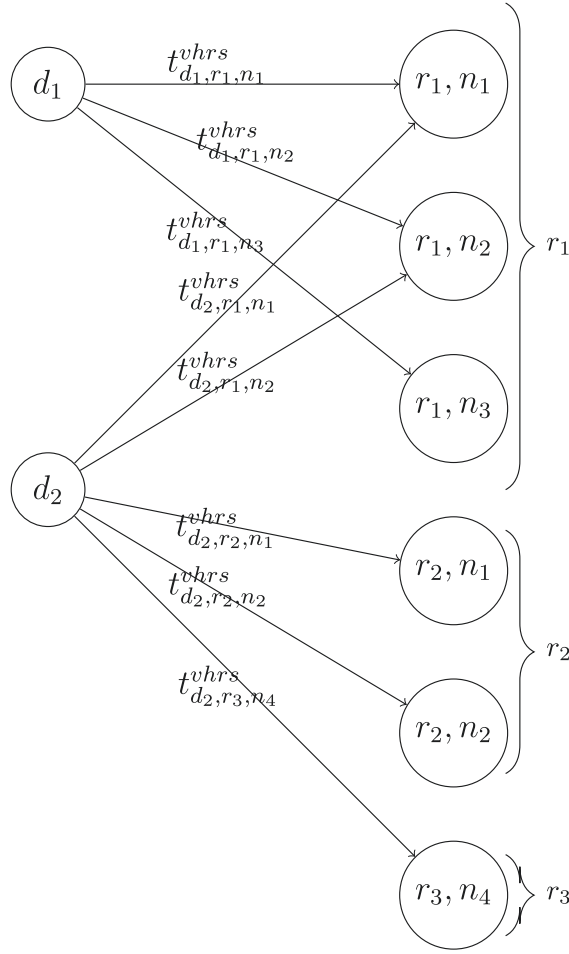
Fig. 6. An illustration of bipartite graph in case of transit-based ridematching.

d driving from $OR(d)$ to a transit node j can reach before bus/train arrival time at j, drive from that station to $DS(r)$ and then $DS(d)$ without violating the time constraints. Specifically, line 23–25 in Algorithm 1 can be replaced with the following constraints:

$$\tau^{ed}(d) + t_{OR(d)\mathcal{Z}(j)} \leqslant \gamma_{new} \tag{9}$$

$$\gamma_{new} + t_{\mathcal{Z}(j)DS(r)} \leqslant \tau^{la}(r) \tag{10}$$

$$\gamma_{new} + t_{\mathcal{Z}(j)DS(r)} + t_{DS(r)DS(d)} \leqslant \tau^{la}(d) \tag{11}$$

5.2. Optimal assignment of drivers to riders

The second step of the procedure is to find an optimal match between riders and drivers among the feasible matches found by Algorithm 1. This is formalized as an Integer Linear Program (ILP). The goal of this optimization program is to find optimal matches out of feasible matches that optimize a given objective. The optimization problem is described below:

5.2.1. Variables

Let $\mathcal{M}_r = \{m \in \mathcal{M} : r \in m\}$ be the set of all feasible matches of r present in $\mathcal{M}$. Similarly, $\mathcal{M}_d = \{m \in \mathcal{M} : d \in m\}$ be the set of all the feasible matches in $\mathcal{M}$ involving $d \in D$. The set of feasible matches can be seen as edges between riders and drivers in a graph shown in Fig. 6, in which a rider can appear more than once with different drop off location and itinerary. Let us assume a binary variable ϵ_k, which takes the value 1, if edge $k \in \mathcal{M}$ is selected, and 0 otherwise. Let $t_k^{drive}, t_k^{transit}, t_k^{walk}, t_k^{\#trans}, t_k^{wait}, d_k^{vmi}$ and t_k^{vhrs} be the driving time, transit time, walking time, number of transfers, waiting time, vehicle-miles savings and vehicle-hrs savings associated with edge $k \in \mathcal{M}$. These parameters can be used to define different objectives for the matching optimization program. The vehicle-hrs savings t_k^{vhrs} can be computed by subtracting t_k^{drive} from sum of driving time when there is no ridesharing (12).

$$t_k^{vhrs} = t_{OR(r)DS(r)} + t_{OR(d)DS(d)} - t_k^{drive} \text{ for } r, d \in k \quad \forall k \in \mathcal{M} \tag{12}$$

5.2.2. Objective

There can be several objectives for matching riders and drivers. We consider two different objectives for this problem:

1. Maximize total number of matches, $Z_1 = \sum_{k \in \mathcal{M}} \epsilon_k$
2. Maximize veh-hrs savings, $Z_2 = \sum_{k \in \mathcal{M}} t_k^{vhrs} \epsilon_k$

5.2.3. Constraints

There are two types of constraints which are given below:

1. Each rider should be assigned to at most one driver.

$$\sum_{k \in \mathcal{M}_r} \epsilon_k \leqslant 1 \quad \forall r \in R \tag{13}$$

2. Each driver should be assigned to at most one rider.

$$\sum_{k \in \mathcal{M}_d} \epsilon_k \leqslant 1 \quad \forall d \in D \tag{14}$$

The overall optimization problem to optimally assign riders with drivers can ve written as follows:

$$\begin{aligned}
\underset{\epsilon}{\text{maximize}} \quad & Z \\
\text{subject to} \quad & \sum_{k \in \mathcal{M}_r} \epsilon_k \leqslant 1 \quad \forall r \in R \\
& \sum_{k \in \mathcal{M}_d} \epsilon_k \leqslant 1 \quad \forall d \in D \\
& \epsilon_k = \{0, 1\} \quad \forall k \in \mathcal{M}
\end{aligned} \tag{15}$$

Remark 3. The solution obtained by LP relaxation of (15) produces an integral solution.

Proof. (*Informal*)   This is because the problem can be formulated as a minimum cost flow problem with integer capacities and the corresponding polytope defined by the constrains satisfies the totally unimodularity condition. As each edge is unique, each column of the matrix produced by the set of constraints will still have at most two non-zero values. Of course, each entry of the matrix would be either 0 or 1, and rows of the matrix can still be partitioned into two sets with non-zero values in separate sets. □

5.3. A rolling horizon algorithm for dynamic transit-based ridesharing

The ridesharing matching problem is dynamic in nature as requests arrive continuously overtime throughout the day. This requires solving the given problem several times during the day since future requests are unknown and drivers and riders enter and leave the system continuously over time. This type of uncertainty is usually managed using *rolling horizon strategy* (Agatz et al., 2011; Wang et al., 2017). Using this strategy, the matching problem is solved in several instances, and matches are not finalized until necessitated by the deadline. The frequency of optimization can be determined using two approaches-*event driven* or *fixed time step* approach (Najmi et al., 2017). The event-driven approach initiates optimization at each time a new request is received. However, this may lead to synchronization issues if a new request is received before solving the previous problem. For the framework proposed in this study, it is efficient to use a fixed time-step approach with predefined time steps. Pseudocode for using a rolling horizon approach to solve transit-based ridesharing matching problem is proposed in Algorithm 2. The algorithm starts with initializing set $D^{sys}(\tau)$ and $R^{sys}(\tau)$ representing driver and rider requests currently in the system. At each time step κ, we consider all new rider and driver requests (D^{new} and R^{new} respectively) received by the platform during the time interval of length σ. We add these new requests to the set of existing requests $D^{sys}(\tau)$ and $R^{sys}(\tau)$. We then maintain a set of feasible match $\mathcal{M}$ for $d \in D^{sys}(\tau)$ and $r \in R^{sys}(\tau)$. To make it computationally efficient, we run Algorithm 1 for new riders and drivers only (line 12–13). The new feasible matches found using Algorithm 1 are added to the set $\mathcal{M}$. Then, an optimization problem is called to decide a set of optimal matches $\mathcal{M}^*$ for given feasible matches $\mathcal{M}$. After this, a subset of matches $\mathcal{M}^{fin} \subseteq \mathcal{M}^*$ are finalized for the requests which are expiring in next time step. A lead time μ is also subtracted from $\tau^{ld}(p)$ to allow some flexibility to riders and drivers to get ready for the trip. The requests which are finalized and the ones which did not found any optimal matches and expiring in the next step are also removed from the system. Then, the horizon is rolled over to the next time step and the process is repeated. The results of a simulation experiment are given in Section 6.7.

Algorithm 2. Pseudocode for dynamic ridesharing using rolling horizon policy

---

1: **procedure** ROLLINGHORIZON
2: $\quad D^{sys}(\tau) \leftarrow \{\}$ $\qquad\qquad\qquad\qquad\qquad$ ▷ set of drivers in the system at time τ
3: $\quad R^{sys}(\tau) \leftarrow \{\}$ $\qquad\qquad\qquad\qquad\qquad$ ▷ set of riders in the system at time τ
4: $\quad \mathcal{M}(\tau) \leftarrow \{\}$ $\qquad\qquad\qquad\qquad\qquad\quad$ ▷ set of feasible matches at time τ
5: $\quad \mathcal{M}^{\text{fin}}(\tau) \leftarrow \{\}$ $\qquad\qquad\qquad\qquad$ ▷ set of final optimal matches at time τ
6: $\quad \tau \leftarrow 0$
7: $\quad$ **for** $\kappa \in \{1, 2, 3, ..., n\}$ **do** $\qquad\qquad\qquad\qquad\qquad$ ▷ time steps
8: $\qquad R^{new} \leftarrow \{r \in R | \tau \le \tau^{at}(r) \le \tau + \kappa\sigma\}$ $\qquad$ ▷ set of new riders in time step k
9: $\qquad D^{new} \leftarrow \{d \in D | \tau \le \tau^{at}(d) \le \tau + \kappa\sigma\}$ $\qquad$ ▷ set of new drivers in time step k
10: $\qquad D^{sys} \leftarrow D^{sys} \cup D^{new}$
11: $\qquad R^{sys} \leftarrow R^{sys} \cup R^{new}$
12: $\qquad \mathcal{M} \leftarrow \mathcal{M} \cup \text{RIDERDRIVERPOTENTIALMATCH}(R^{new}, D^{sys}, G_R, G_T, \eta)$
13: $\qquad \mathcal{M} \leftarrow \mathcal{M} \cup \text{RIDERDRIVERPOTENTIALMATCH}(R^{sys}, D^{new}, G_R, G_T, \eta)$
14: $\qquad \mathcal{M}^* \leftarrow \text{OPTIMIZE}(\mathcal{M})$ $\qquad\qquad$ ▷ set of current optimal matches
15: $\qquad$ **for** $(r, d, j, \xi) \in \mathcal{M}^*$ **do**
16: $\qquad\quad$ **if** $\tau^{ld}(r) - (\tau + \kappa\sigma) \le \mu$ **or** $\tau^{ld}(d) - (\tau + \kappa\sigma) \le \mu$ **then**
17: $\qquad\qquad \mathcal{M}^{\text{fin}} \leftarrow \mathcal{M}^{\text{fin}} \cup \{(r, d, j, \xi)\}$
18: $\qquad\qquad \mathcal{M} \leftarrow \mathcal{M} \setminus \{m \in \mathcal{M} \mid (r, d, j, \xi) \in \mathcal{M}^* \ \& \ r \in m \ \& \ d \in m\}$
19: $\qquad\qquad D^{sys} \leftarrow D^{sys} \setminus \{d\}, R^{sys} \leftarrow R^{sys} \setminus \{r\}$
20: $\qquad \tau \leftarrow \tau + \kappa\sigma$

---

6. Numerical experiments

In this section, we present results from simulation experiments to get more insights and analyze the benefits of the proposed transit-based ridesharing program. We implemented Algorithms 1 and 2 in Python 2 on a standard PC with Core i5 3.00 GHz processor with 16 GB RAM.

6.1. Simulation setup

The simulation environment is based on the 2010 activity-based travel demand model for Twin Cities, MN developed by Metro-politan Council (Cambridge Systematics, 2015). The Twin Cities metropolitan region is spread over $8,120 \text{ mi}^2$ area with a population of about 3.55 million. There are more than 11 million trips occurring every day. The activity-based model takes input from travel behavior surveys and outputs simulated trips for the entire metropolitan region. The model output used for this study consists of trip origin, destination, activity purpose, the mode used, departure time, and arrival time. For this ridesharing program, we consider only the commuting and school trips during morning peak hours (6:00 AM to 9:00 AM). The departure and arrival time of each trip in this data is available on a 30-min scale, so we added a uniform random number between 0 to 30 min to represent the variation in the departure time of trips. The Twin Cities transportation network has 3030 traffic analysis zones (TAZs), 23,812 nodes, and 56,688 links. Using a static traffic assignment program (Kumar, 2017), we calculated the value of equilibrium travel time on links for an hourly demand obtained from the above model. For each trip, the value of the shortest path travel time is increased by a certain proportion, known as *time flexibility*. The latest arrival time $\{\tau^{la}(p), p \in P\}$ at passengers' destination are obtained by adding this increased travel time to the earliest departure time $\{\tau^{ed}(p), p \in P\}$. Then a uniform random number between 0 and 60 min is further subtracted from $\{\tau^{ed}(p), p \in P\}$ to calculate the value of announcement time $\{\tau^{at}(p), p \in P\}$ of trips. After this, unlike other studies, we explicitly selected trips with the transit first-mile access problem (i.e., no transit stop within a buffer distance of 0.75 mi). This is important to assess the efficiency of a matching algorithm designed to solve the first-mile problem. The driver or rider role to a given trip is also assigned randomly based on the driver-rider ratio. Other specifications are given in Table 1.

Metro Transit is the primary transit agency in the Twin Cities region, offering an integrated network of buses, light rails, and a commuter train. The schedule-based transit network is created using the GTFS data obtained from Minnesota Geospatial Commons (2016) with the help of an R script (Kumar and Khani, 2019). The script produces different types of transit network links A_T based on the set of pre-defined parameters. The time to traverse an access link (i, j) is calculated by dividing the distance between i and j and speed of walking, which we considered as 3 mi/hr. The Metro Transit GTFS feed contains 191 routes, 13,672 stops, and 9,042 vehicle trips. The number of different types of transit network links generated is given in Table 2. The standard implementation of schedule-based transit shortest path using binary heaps is also shared as open-source code (Khani, 2018).

In the following subsections, we first present the results for a static matching problem, in which case the requests are known before the start of the day. Then, a dynamic simulation experiment results using the rolling horizon policy is presented in Section 6.7. The

Table 1
Characteristics of simulation setup.

Trips	Commuting and school trips having first mile access problem
Driver-rider ratio	1
Acceptable walking distance to/from bus stop	0.75 mi
Acceptable transfer distance for walking	0.25 mi
Walking speed of riders	3 mi/h
Maximum wait time at transit stop	10 min
t^{ser}	2 min
η	{1.5, 2.0, 1.0, 2.0}
Vehicle capacity	2 seats

Table 2
Twin Cities schedule-based transit network.

Number of nodes	490,112
Number of in-vehicle links	481,065
Number of waiting transfer links	241,187
Number of walking transfer links	5,082,747
Number of access/egress links	8,496,605
Total number of links	14,301,604
Computational time to generate transit network	8 min

sensitivity analysis of different parameters, such as participation rate, rider, and driver time flexibility, and the driver-rider ratio is also presented. The results are computed for different objective functions given in Section 5.2.
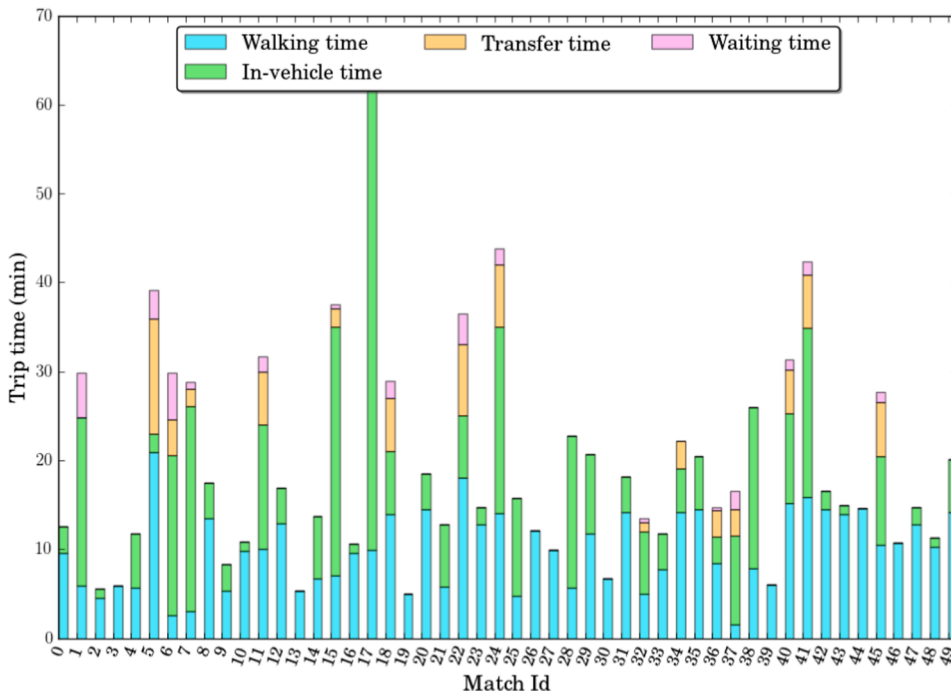
6.2. Static simulation

For this experiment, the driver-rider ratio is equal to 1.0, i.e., an equal number of drivers and riders in the system and a participation rate of 1% (1,087 riders and 1,092 drivers) among travelers facing the transit first-mile access problem. We provide time flexibility of 40% to the drivers and 80% to the riders as transit travel time is likely to be higher than the auto travel time. To avoid any bias, we draw five different samples and calculate the average of the results obtained. The results obtained using different samples with different matching objectives are given in Table 3. We can observe that for 1,088 drivers and 1,097 riders, Algorithm 1 took an average of 7.7 min and produced 12,338 feasible matches. It only took a fraction of a second to solve the matching optimization problem (15). The average number of optimal matches found using objective Z_1 exceeds objective Z_2 by a small number. However, the average veh-hrs savings found using objective Z_2 is significantly more than the value found using Z_1. As Z_2 is about maximizing veh-hrs savings, the average detour time of drivers found using objective Z_1 is more than Z_2 by 2 min. On the other hand, the average shared time between a rider and a driver up to a transit station and average transit time using objective Z_2 is more than Z_1. The average walking time of riders using both objectives is almost similar to without a transfer, the walking component of the transit trip should be between the destination and closest transit station only. The waiting time is negligible which shows only a few optimal trips have a waiting transfer, which is assigned a higher weight in the algorithm.

Table 3
Static simulation results.

Sample number	1		2		3		4		5		Average	
Number of participants	1,120 D, 1,059 R		1,035 D, 1,144 R		1,096 D, 1,083 R		1,092 D, 1,087 R		1,099 D, 1,080 R		1,088 D, 1,097 R	
Run time of Algorithm 1 (min)	7.55		7.46		7.64		7.47		8.4		7.704	
Number of feasible matches found	22,397		10,415		9,936		10,021		8,924		12,338	
Number of riders and drivers in feasible matches	364 R, 604 D		218 R, 344 D		178 R, 299 D		182 R, 334 D		187 R, 346 D		263 R, 386 D	
Objective	Z_1	Z_2	Z_1	Z_2	Z_1	Z_2	Z_1	Z_2	Z_1	Z_2	Z_1	Z_2
Run time of optimization program (sec)	0.1	0.12	0.08	0.08	0.07	0.08	0.07	0.15	0.07	0.09	0.09	0.1
Optimal number of matches found	319	316	192	192	155	154	166	166	168	166	200	198
Total vehicle-hrs savings	107.4	122.47	55.97	67.03	48.81	57.15	48.14	55.97	47.17	58.7	61.5	71.6
Average driver detour time (min)	10.12	7.65	11.58	8.82	11.39	8.48	11.78	9.08	11.95	8.29	11.36	9.2
Shared time between driver and rider (min)	29.54	29.9	27.86	28.1	29.54	29.44	28.36	28.24	28.12	28.52	28.68	34.4
Average transit time (min)	6.06	6.41	6.44	6.59	6.59	7.64	5.43	5.76	5.29	6.32	5.29	6.32
Average walking time (min)	10.75	10.85	10.1	11	11	10.85	10.24	10.01	10.57	10.46	10.532	10.452
Average waiting time (min)	0.5	0.63	0.59	0.61	0.61	0.64	0.58	0.54	0.463	0.4	0.5486	0.574
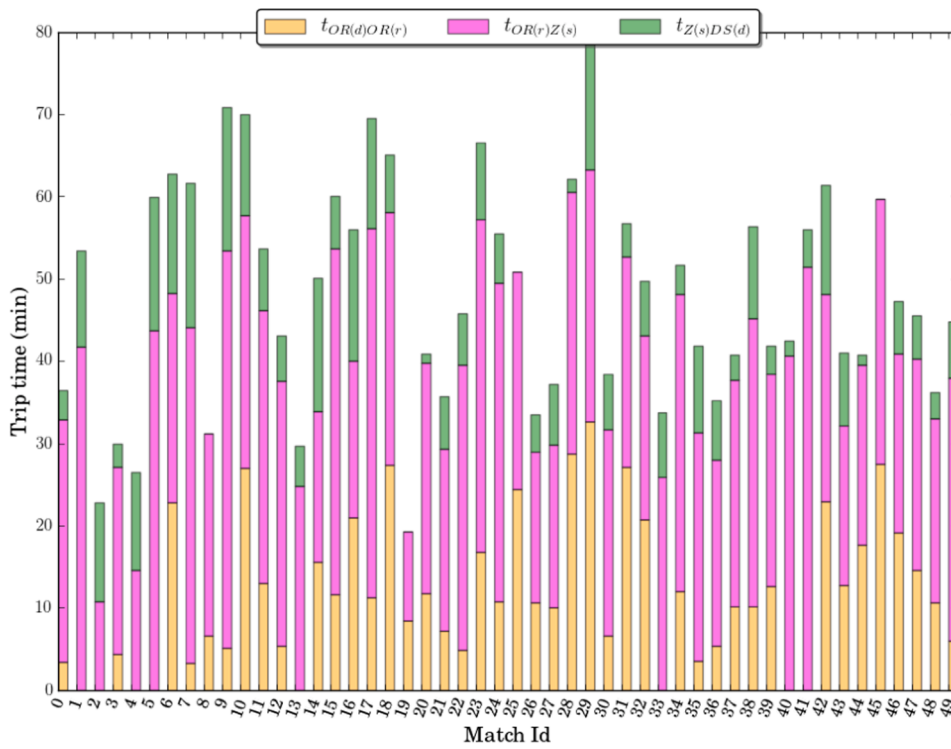
(a) Driver trip components

(b) Riders trip components

Fig. 7. Different components of travel time for riders and drivers.

(a) Run time of Algorithm 1   (b) Number of feasible matches
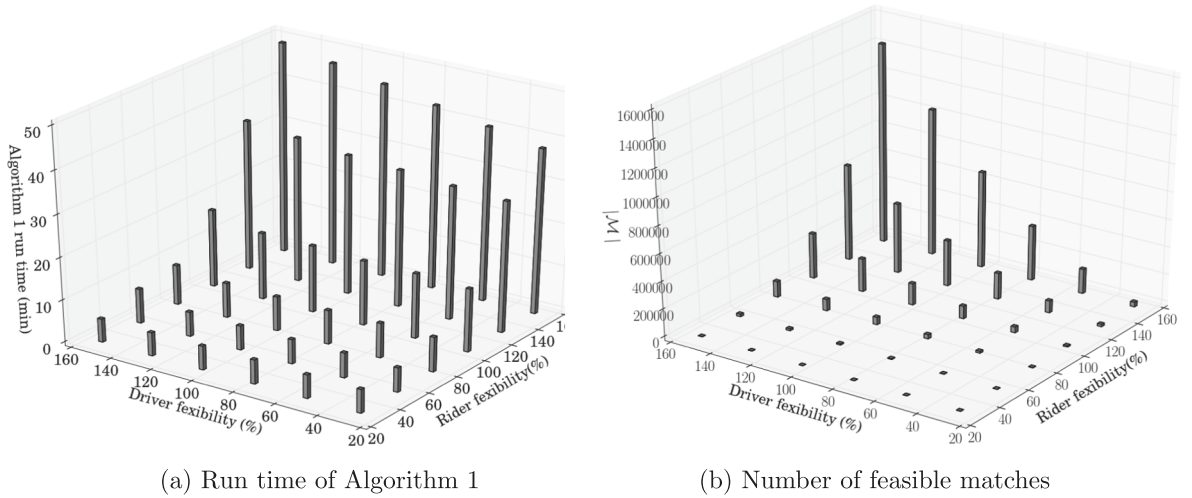
Fig. 8. The effect of time flexibility of riders and drivers on (a) run time of Algorithm 1 and (b) number of feasible matches found.

To closely examine different components of individual driver and rider-trips, we created a stacked bar chart for the average value of different travel time components (Fig. 7). Fig. 7a shows the driver trips, and Fig. 7b shows the rider-trips. A particular ID in both panels represents a match using objective Z_2. This match ID can be compared in both plots to see the relationship between different components of the trip time. As we can see drivers spent most of the time driving a rider to a station. Drivers also spent a significant amount of time driving from their origin to a rider's origin. The driving time from station drop off location to drivers' destination is low as compared to the other two components of the drivers' itinerary. The average driver detour time of 9.2 min (Table 3) shows that while maximizing the vehicle-hours savings, drivers are assigned to riders whose drop-off transit stations are close to drivers' destinations.

Looking at rider itineraries, for most of the riders, the drive time from the origin location to the transit station is a significant component of the trip followed by in-vehicle time and then walking time. Walking time is higher in most of the itineraries because of the unavailability of the actual destination location in the data due to which a walking component from the centroid of TAZ to alighting stop is specified. We can observe that most of the riders are assigned a trip without any walking and waiting transfer to the other routes due to their higher weight in the algorithm. About 31% of the riders were found to have one or two transfers on their trip.

One of the common objectives for the matching optimization program (15) in the literature (Agatz et al., 2011; Agatz et al., 2012) is to maximize the vehicle-miles savings. The overall vehicle-miles savings is calculated as $Z_3 = \sum_{k \in \mathcal{M}} d_k^{vmi}$, where savings due to individual matches d_k^{vmi} can be computed using (16). We conducted an experiment using the objective function Z_3 and found results close to maximizing Z_2. We found that the average number of matches, vehicle-hrs savings, and vehicle-miles savings using (Z_2, Z_3) are equal to (198, 199), (71.6, 64.46) veh-hrs, and (10069, 10940) veh-miles respectively. This is not surprising, as for average speed, the distance remains proportional to time. Furthermore, the objective Z_2 is quite popular in the system-optimal design of transportation systems (Sheffi, 1985). Therefore, for further experiments in this article, we keep Z_1 and Z_2 only.

$$d_k^{vmi} = dist_{OR(r)DS(r)} + dist_{OR(d)DS(d)} - dist_k^{drive} \text{ for } r, d \in k \quad \forall k \in \mathcal{M} \qquad (16)$$

6.3. Sensitivity to time flexibility

To better understand the effect of time flexibility, we performed a sensitivity analysis on the time flexibility of the drivers and riders. We kept the participation rate 1% and a driver-rider ratio equal to 1.0, similar to previous tests. For varying rider and driver time flexibility (25 % to 150%), the results are compiled for computational time, number of feasible matches, number of optimal matches, and vehicle-hrs savings using both objectives. Fig. 8a shows the effect of varying time flexibility on the computational time of Algorithm 1. With more time flexibility in rider and driver travel time, the runtime of Algorithm 1 increases. The increase in computational time with an increase in the drivers' flexibility is due to an increase in the number of feasible rider matches. The increase in rider time flexibility can significantly increase the computational time due to a higher number of nodes in space-time prism of riders resulting in a higher number of feasible matches. The effect of rider time flexibility on computational time is more than driver time flexibility. Fig. 8b shows the number of feasible matches found for an increase in rider and driver time flexibility. In this figure, we can observe an abrupt increase in the number of feasible matches with the increase in rider and driver time flexibility. This is due to the
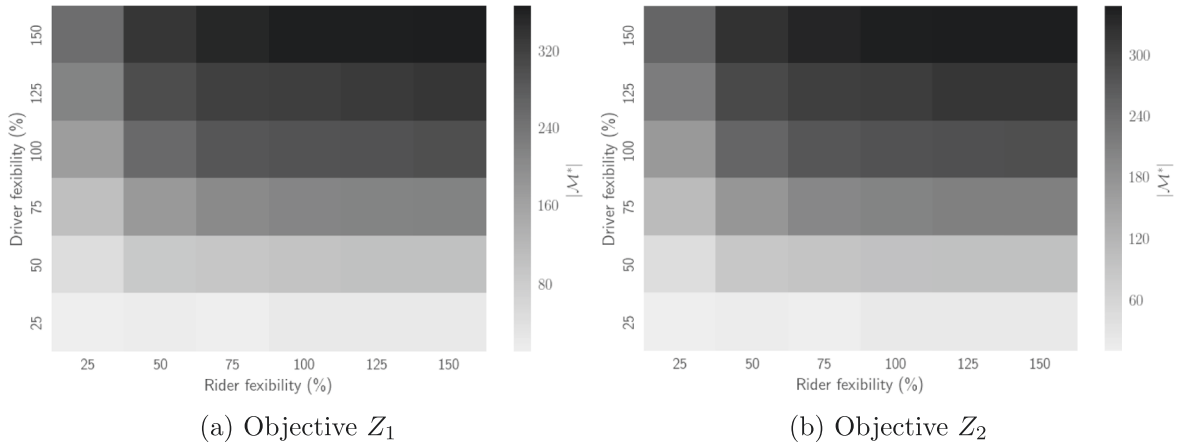
(a) Objective Z_1

(b) Objective Z_2

Fig. 9. The effect of time flexibility of riders and drivers on optimal number of matches.

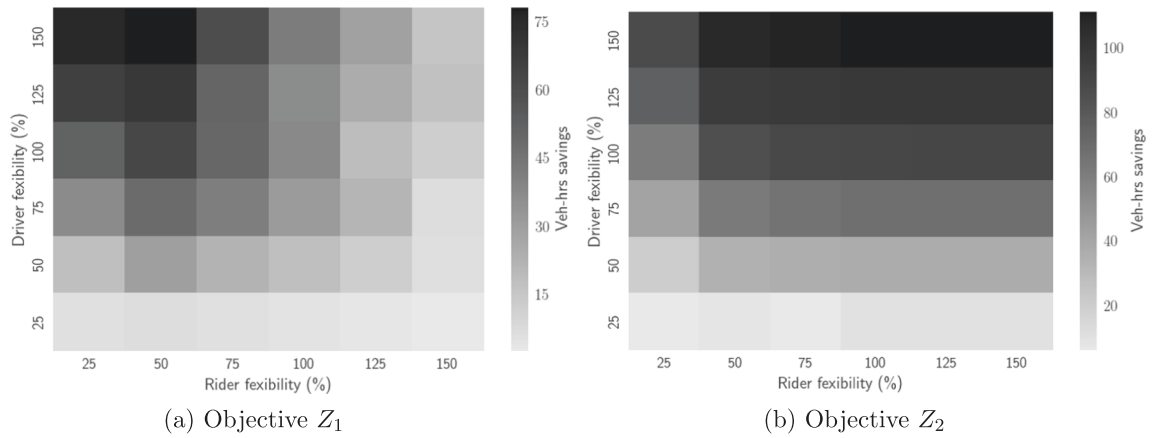(a) Objective Z_1

(b) Objective Z_2

Fig. 10. The effect of time flexibility of riders and drivers on veh-hrs savings.

increase in the number of feasible transit stops and transit trip itineraries.

Figs. 9 and 10 show the effect of varying rider and driver time flexibility on the number of optimal matches found and vehicle-hrs savings using different objectives. In general, the number of optimal matches found increase with an increase in the time flexibility of both riders and drivers. As expected, the number of optimal matches are more in case of objective Z_1 in comparison to Z_2. Similarly, the veh-hrs savings in case of objective Z_2 is higher in comparison to Z_1. Interestingly, the effect of rider flexibility on the number of optimal matches found is more in comparison to driver time flexibility. However, in the case of veh-hrs savings, the driver time flexibility has more effect on objective Z_1, and rider time flexibility has more effect on objective Z_2.

6.4. Sensitivity to participation rate

Using 150% time flexibility for riders, 50% time flexibility for drivers, and the driver-rider ratio of 1.0, results for different participation rates varying from 0.5% to 2% are stated in Table 4. With increasing participation rate, run time of Algorithm 1, the number of feasible matches, the percentage of drivers and riders matched, and veh-hrs savings increase. With a 2% participation rate, the amount of veh-hrs savings can be as high as 230.8 veh-hrs using objective Z_2. This experiment shows that a higher participation rate can result in batter matching results.

Table 4

Sensitivity to participation ratio.

Participation %	$|\mathcal{M}|$	% drivers matched		% riders matched		veh-hrs savings	
		Z_1	Z_2	Z_1	Z_2	Z_1	Z_2
0.5	32,377	22.44	21.91	24.28	23.71	25.53	36.85
1	131,405	28.09	27.45	28.64	27.99	65.68	93.81
2	676,671	30.68	30.21	30.09	28.55	161.06	230.48
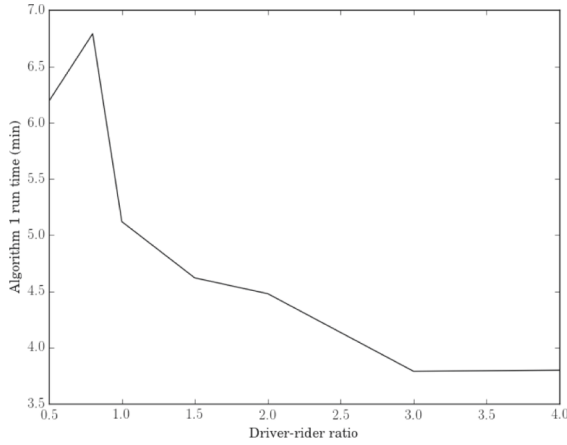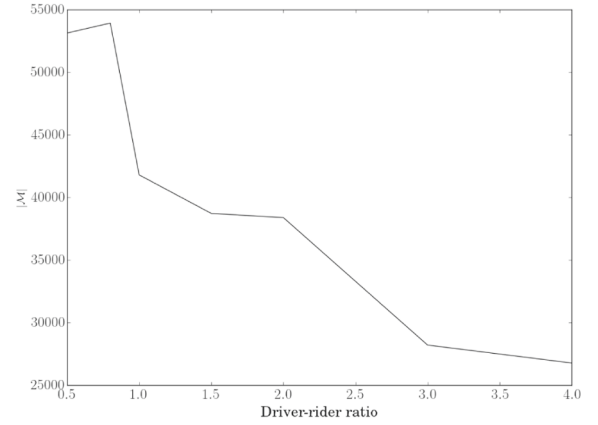
(a) Run time of Algorithm 1   (b) Number of feasible matches

Fig. 11. The effect of driver-rider ratio on (a) run time of Algorithm 1 and (b) number of feasible matches found.

6.5. Sensitivity to driver-rider ratio

To see how the number of riders and drivers in the system affect results, we performed a sensitivity analysis on the driver-rider ratio. The time flexibility was kept 80% for riders and 50% for drivers and the participation rate was kept 1%. The driver-rider role in the data was assigned randomly varying from 0.5 to 4.0. As we can see in Fig. 11 that with more number of riders in the system, the runtime of Algorithm 1 and number of feasible matches, $|\mathcal{M}|$ is higher. The highest value of both quantities was observed when the number of drivers and riders are equal in the system.

A similar trend can be observed in case of the optimal number of matches and veh-hrs savings (Fig. 12 and Fig. 13). With a higher number of riders in the system, more riders can be matched resulting in higher matching rate and higher veh-hrs savings. Fig. 12 shows both objectives result in an almost similar matching rate. However, objective Z_2 results in significantly higher veh-hrs savings in
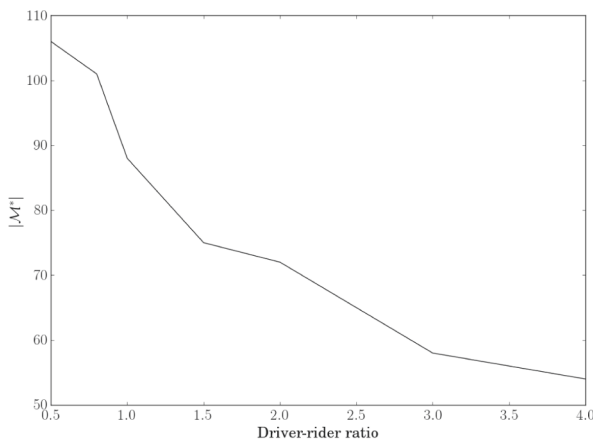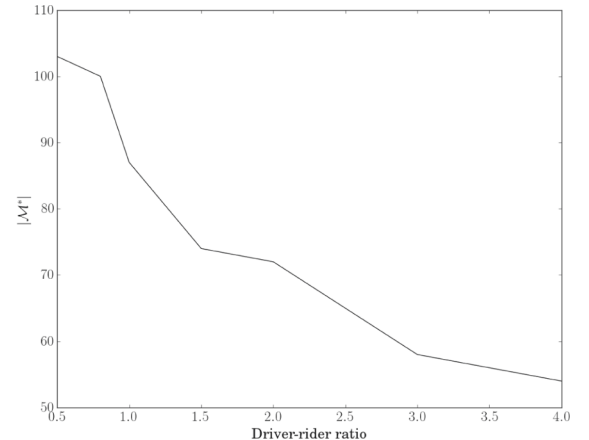
(a) Objective Z_1 (b) Objective Z_2

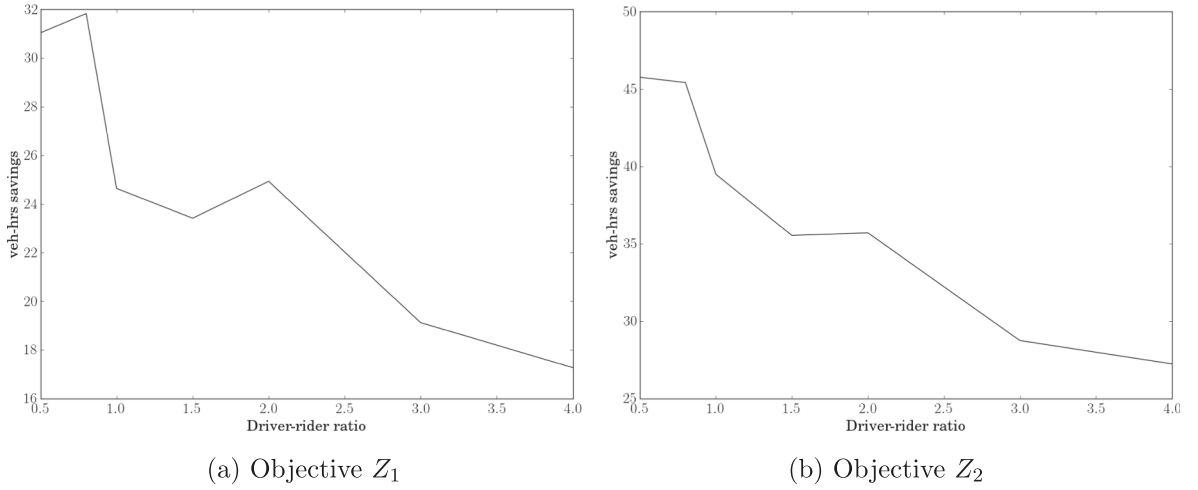**Fig. 12.** The effect of driver-rider ratio on optimal number of matches found.

(a) Objective Z_1 (b) Objective Z_2

Fig. 13. The effect of driver-rider ratio on veh-hrs savings.

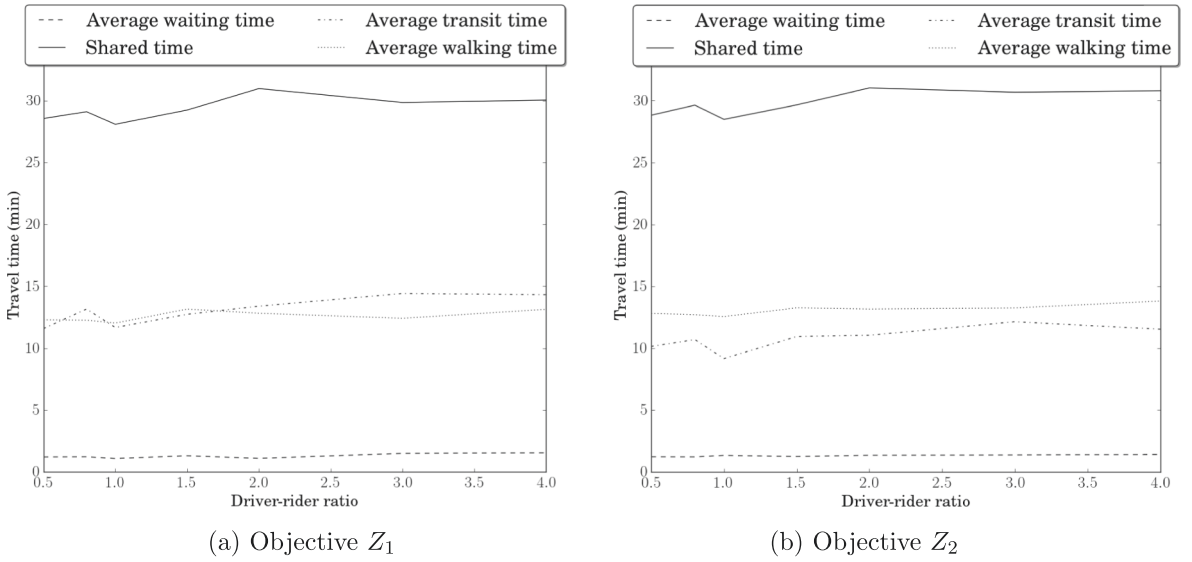(a) Objective Z_1 (b) Objective Z_2

Fig. 14. The effect of driver-rider ratio on different time components of rider itinerary.

comparison to objective Z_1 (Fig. 13).

The effect of driver-rider ratio on the value of transit travel time components of matched participants is negligible after driver-rider ratio equal to 1.0. Overall, the average shared time between riders and drivers has a higher value in comparison to other components (Fig. 14).

6.6. Combining transit-based ridesharing with stand-alone ridesharing

In Section 3, we presented a hypothesis that combining the transit-based ridesharing with stand-alone ridesharing would improve the matching rate and overall savings in veh-hrs. The matching rate is defined as the number of matches divided by the average number of riders and drivers in the system. To get insights into this hypothesis, we run simulations in three different settings:

1. *Stand-alone ridesharing (RS)*: In this setting, we find matches between riders and drivers, in which case, the complete trip of a rider is covered by the driver. We use the method proposed by Agatz et al. (2011) to compute the feasible matches and then solve the optimization program (15) to find optimal matches.
2. *Stand-alone transit-based ridesharing (TRS)*: In this setting, we run Algorithm 1 to find feasible matches and solve the optimization program (15) to find optimal matches.

Table 5
Comparing matching results for three different setting of ridesharing.

		RS	TRS	RS + TRS
	No of participants	1,035 D, 1,144 R	1,035 D, 1,144 R	1,035 D, 1,144 R
	Number of riders and drivers in the feasible matches	925R 716 D	218R 344 D	936R, 720D
Maximize savings	Final matches found	562	192	575 (123 TRS, 452 RS)
	Matching rate	51 %	18 %	53 %
	Vehicle-hrs savings	138	67	149
	Average driver ride time (min)	37.80	45.83	37.63
	Average rider ride time (min)	22.40	47.10	26.97
	Driver detour time (min)	7.66	8.82	7.47
	Shared time(min)	22.40	28.10	22.55
Maximize matching	Final matches found	583	192	593 (104 TRS, 489 RS)
	Matching rate	53 %	18 %	55 %
	Vehicle-hrs savings	109.73	55.34	114
	Average driver ride time (min)	39.52	48.86	39.62
	Average rider ride time (min)	20.43	46.15	23.90
	Driver detour time (min)	9.14	11.85	9.29
	Shared time(min)	20.43	27.92	20.62

3. *Combining stand-alone ridesharing and transit-based ridesharing (RS + TRS)*: In this setting, we allow drivers to cover the rider's complete trip. We find feasible matches using both Algorithm 1s and the method proposed by Agatz et al. (2011), and solve the optimization program (15) to find optimal matches.

For the driver-rider ratio equal to 1, the results are presented for all of the above settings in Table 5. For 1035 drivers, and 1144 riders in the system, the optimal matches found in three different settings are 562, 192, and 575 respectively. Combining the RS with TRS increases the matching rate in the case of both objectives (Z_1 and Z_2). However, the savings in the case of Z_2 is significantly higher in comparison to Z_1. If we look at the results, we can see that the average savings per match by transit-based ridesharing setting are 21 veh-min in comparison to 14 veh-min in case of stand-alone ridesharing. The average savings is calculated by dividing the veh-hrs saving by the number of final matches. This shows that involving transit trips can significantly help in reducing congestion. Moreover, with better results, by combining both types of ridesharing settings, we have more confidence in the gain from adopting the proposed system and promoting transit.

6.7. An experiment on dynamic ridesharing using rolling horizon policy

To deal with the dynamics of a transit-based ridesharing system, we presented a rolling horizon algorithm in Section 5.3. In this section, we prepare a simulation environment to see how the rolling horizon approach works. We assumed a time step, σ = of 5 min for this experiment. Depending on the demand, σ plays an important role in deciding if one iteration of Algorithm 2 can be solved within a time interval of σ. For this experiment, we consider driver-rider ratio equal to 1.0, rider, and driver flexibility to be equal to 150% and 50% respectively, and 1% participation rate during the morning peak. Fig. 15 shows simulation results from 5:00 AM to 9:00 AM. The computational time results are presented in Fig. 15a. The maximum computational time of 4.23 min was observed at 7:20 AM when there were 412 participants in the system. This shows that Algorithm 2 can run successfully for a 1% participation rate with given σ. The optimization time remains a fraction of second for each iteration and most of the time was spent in finding feasible matches using Algorithm 1.

Fig. 15b shows the number of riders and drivers present and added in each interval. We can observe a peak demand between 7:00 AM-8:00 AM. The number of matches finalized in each time step is almost equal to the number of rider and driver requests being expired due to the unavailability of match satisfying given constraints. Overall, about 54% of the participants found a successful match during the simulation. This example shows that the given algorithm can handle uncertainties arising in this dynamic matching problem.

7. Discussion

The proposed research attempts to solve the transit FMLM problem using ridesharing under certain assumptions mentioned in Section 4.1. We discuss the possible implications of these assumptions and possibilities to resolve those issues. First, we assumed that transit vehicles have unlimited capacity. However, the congested transit systems often face the problem where passengers fail to board the train due to limited capacity. This will result in extra waiting time, which was not considered when finding feasible matches. A time-dependent origin-destination flow matrix (Kumar et al., 2018; Nassir et al., 2011; Kumar et al., 2019) can help in evaluating a failure to board probability to incorporate in the model. Second, it was assumed that the travel time on both road and transit network is
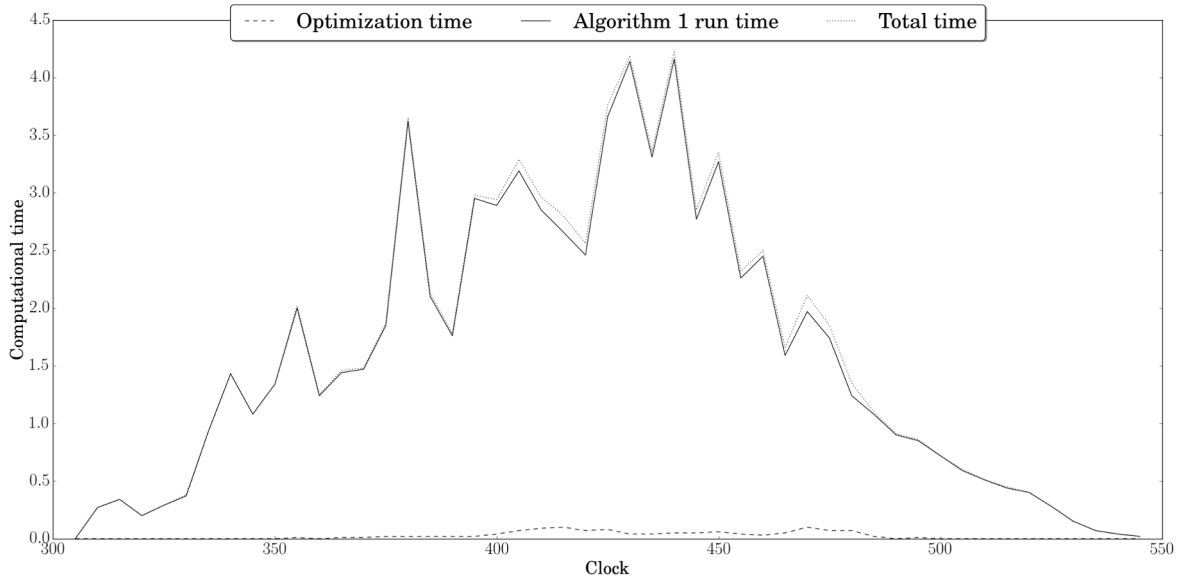
(a) Time spent during the simulation in each time interval
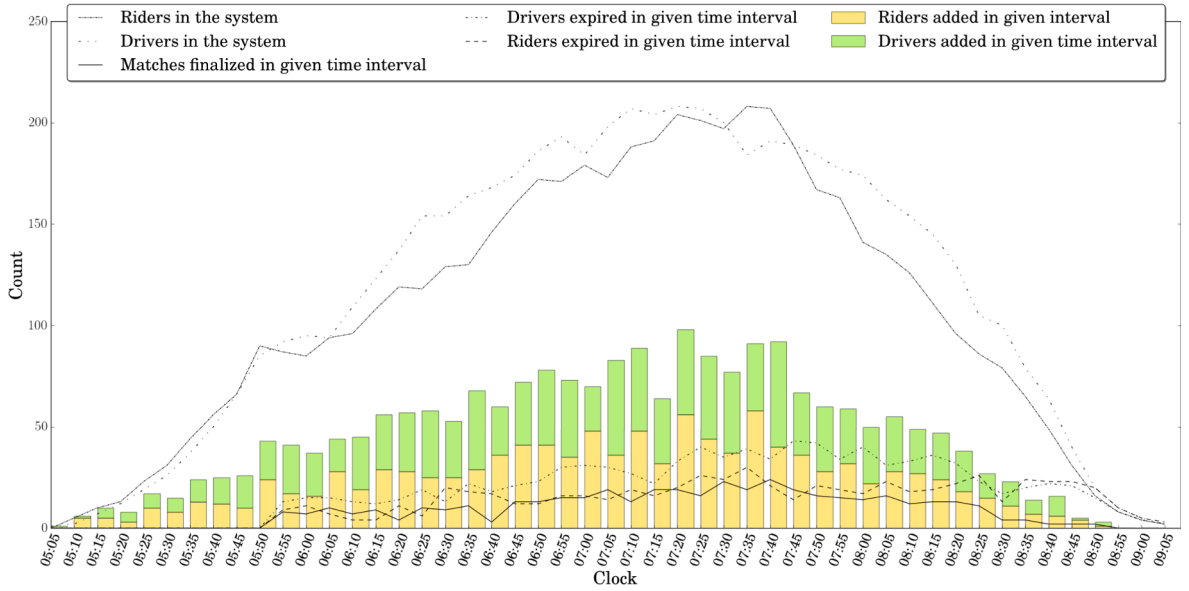
(b) Distribution of riders and drivers in the system

Fig. 15. Algorithm 2 simulation results.

reliable. However, the travel time experienced by passengers is often subject to uncertainty. This uncertainty is more prominent in the case of bus transit service. One way to address this problem is to create matches in the framework which allow drop-off only at stops served by high-frequency routes. The higher the frequency of transit service, lesser is the waiting time. Another way is to formulate this problem as an online shortest path problem, which accounts for the uncertainty in the travel time and provides optimal policies for each possible realization (Waller and Ziliaskopoulos, 2002; Khani, 2019). Future work is required to develop an algorithm that creates feasible matches based on online information. Third, better calibration of service time based on the dwell time models is needed.

Finally, for the success of a transit-ridesharing program, we need to design a mechanism such that all the players participating in this program are well-off. In the initial phase, transit agencies and the Government can provide subsidies to attract more drivers to participate in this program. Riders can pay for the service, and in return, they do not have to worry about the parking and gas cost. Drivers can be encouraged to participate in this program to reduce the negative externalities of congestion and for minimal compensation. Bian and Liu (2019a,b) designed such a mechanism, but further research is needed in this area to develop a more efficient mechanism.

8. Conclusions and future research

This research proposes ridesharing as a promising solution to the transit FMLM problem. To integrate ridesharing with fixed-route transit, we developed a labeling algorithm to find feasible matches between drivers and riders. The algorithm uses a schedule-based transit shortest path to generate optimal itineraries for riders, which captures various complexities of a transit network such as precise waiting time, in-vehicle time, transfer time, and access time. To reduce the size of the network search, the algorithm uses the concept of space-time prism (STP) which provides constraints on rider and driver movement based on the available time budget and location. Using a matching optimization program, the riders and drivers are matched together up to the first mile of the rider trip. Two different objectives are considered for the problem; maximizing the total number of matches and maximizing the total vehicle-hrs savings. Using simulation experiments on real data from a large scale network, we found that the proposed ridesharing program can save a significant amount of veh-hrs spent in the system. The number of matches found using both objectives were found to be close to each other. However, maximizing veh-hrs savings can save a lot more veh-hrs than maximizing the total number of matches. This shows that maximizing the matching rate is not a good objective. We also observed different trip components of drivers and riders and found that driving from riders' origin to drop-off stations is a significant component of both riders and drivers' itinerary. This is due to less time flexibility in riders' schedule as transit travel time is generally higher than the corresponding driving time. We also performed a sensitivity analysis on the participation rate, time flexibility, and driver-rider ratio. With an increase in time flexibility and participation rate, the runtime of the algorithm increases, producing a higher number of feasible matches. An equal number of riders and drivers in the system maximizes the matching rate. By combining the stand-alone ridesharing with the transit-based ridesharing program, we obtained a better matching rate and veh-hrs savings. We also proposed an algorithm to solve the given ridesharing matching problem in real-time using a rolling horizon approach. For a 1% participation rate among travelers facing the FMLM problem, a time step of 5 min is adequate for solving the given problem on a regular desktop computer. Furthermore, computational time can be further reduced by using parallelization techniques. Using simulation experiments, we observed that about 46% of riders and drivers did not find a suitable match due to the lack of a feasible partner or time flexibility. Twin Cities sub-urban region being large and sparse has a lower number of trips generated in those regions. The lower matching rate is due to less number of requests. With a higher number of requests (e.g., in denser urban areas such as Manhattan region), the matching rate can be higher.

The current research can be expanded in multiple directions. To improve the matching rate of the current program, there is a need to develop a multiple-matching algorithm. One possibility is to sequentially grow the number of feasible riders for a given driver by checking different time constraints. It is also challenging to develop such an algorithm for riders facing both first and last mile problems which means there is no transit access at both ends of a transit trip. There is a need to develop an algorithm that can generate several ridesharing modal choices for users.

Appendix A

Table 6.

Table 6
Description of notations used in this article.

P	≜	Set of participants
R	≜	Set of riders
D	≜	Set of drivers
$OD(p), DS(p)$	≜	Origin and destination of participant p
$\tau^{at}(p), \tau^{pd}(p), \tau^{pa}(p)$	≜	Announce time, preferred departure time, preferred arrival time,
$\tau^{ed}(p), \tau^{la}(p)$		earliest departure time, and latest arrival time of participant p
a $\delta(p), \Delta(p)$	≜	Maximum schedule deviation and total delay allowed
		in the trip of participant p
$G_R(N_R, A_R)$	≜	Road network consisting of nodes N_R and directed links A_R
$t(a,b)$	≜	Shortest travel time between $a, b \in N_R$
B	≜	Set of transit stops
K	≜	Set of transit trips
L	≜	Set of transit routes
$G_T(N_T, A_T)$	≜	Transit network consisting of nodes N_T and directed links A_T
$A_{T_a}, A_{T_v}, A_{T_w}$	≜	Set of access/egress, in-vehicle, and walking/waiting transfer links
$w(a,b)$	≜	Walking distance between nodes $a, b \in N_T \cup N_R$
$\mathcal{Z}(s)$	≜	Closest road node to the transit stop s
A_m	≜	Set of mode transfer links
$FC_p(\tau), FC_p(\tau)$	≜	Forward and backward cone of $p \in P$ pointed at $OR(p)$ and $DS(p)$ resp.
$STP_p(\tau)$	≜	Space-time prism of participant p
PPA_p	≜	Potential path area of participant p
γ_n	≜	Latest time the rider should depart from n
γ_n^{gc}	≜	Generalized cost label of node n
ξ_n	≜	Predecessor of node n in the shortest path
η_a, η_v, η_w	≜	Weights associated with links in set $A_{T_a}, A_{T_v},$ and A_{T_w}.
$\Gamma(n)$	≜	Forward star of node n
$\Gamma^{-1}(n)$	≜	Backward star of node n
$\mathcal{M}$	≜	Set of feasible matches
$D^{sys}(\tau), R^{sys}(\tau)$	≜	Drivers and riders in the system at time τ
$\mathcal{M}(\tau), \mathcal{M}^{fin}(\tau)$	≜	Feasible and finalized matches in the system at time τ
$\mathcal{M}^*$	≜	Set of finalized matches

References

Aftabuzzaman, M., Currie, G., Sarvi, M., 2015. Evaluating the congestion relief impacts of public transport in monetary terms. J. Public Transp. 13 (1), 1–24.

Agatz, N.A., Erera, A.L., Savelsbergh, M.W., Wang, X., 2011. Dynamic ride-sharing: A simulation study in metro Atlanta. Transp. Res. Part B: Methodol. 45 (9), 1450–1464. https://doi.org/10.1016/j.trb.2011.05.017.

Agatz, N., Erera, A., Savelsbergh, M., Wang, X., 2012. Optimization for dynamic ride-sharing: A review. Eur. J. Oper. Res. 223 (2), 295–303.

Alonso-mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., Rus, D., 2017. Trip-vehicle assignment. Proc. Natl. Acad. Sci. USA 114 (3), 462–467. http://www.ncbi.nlm.nih.gov/pubmed/28049820 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5255617.

Bian, Z., Liu, X., 2019a. Mechanism design for first-mile ridesharing based on personalized requirements part I: Theoretical analysis in generalized scenarios. Transp. Res. Part B: Methodol. 120, 147–171. https://doi.org/10.1016/j.trb.2018.12.009.

Bian, Z., Liu, X., 2019b. Mechanism design for first-mile ridesharing based on personalized requirements part II: Solution algorithm for large-scale problems. Transp. Res. Part B: Methodol. 120, 172–192. https://doi.org/10.1016/j.trb.2018.12.014.

Brons, M., Givoni, M., Rietveld, P., 2009. Access to railway stations and its potential in increasing rail use. Transp. Res. Part A: Policy Pract. 43 (2), 136–149. https://doi.org/10.1016/j.tra.2008.08.002.

Cambridge Systematics, I., 2015. Model User Guide, 2010 Travel Behavior Inventory, Technical report, Chicago, IL.

Cayford, R. Yim, Y.B.Y., 2004. Personalized Demand-Responsive Transit Service.

Chen, S., Wang, H., Meng, Q., 2020. Solving the first-mile ridesharing problem using autonomous vehicles. Comput.-Aided Civ. Infrastruct. Eng. 35 (1), 45–60.

Deleplanque, S., Quilliot, A., 2013. Dial-a-ride problem with time windows, transshipments, and dynamic transfer points, vol. 46. IFAC. URL: doi: 10.3182/20130619-3-RU-3018.00435.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.E., Wang, X., Koenig, S., 2013. Ridesharing: The state-of-the-art and future directions. Transp. Res. Part B: Methodol. 57, 28–46. URL: doi: 10.1016/j.trb.2013.08.012.

Häll, C.H., Andersson, H., Lundgren, J.T., Värbrand, P., 2009. The Integrated Dial-a-Ride Problem. Public Transp. 1 (1), 39–54.

Jafari, E., Rambha, T., Khani, A., Boyles, S.D., 2016. The For-Profit Dial-a-Ride Problem on Dynamic Networks The For-Profit Dial-a-Ride Problem on Dynamic Networks. Transp. Res. Board 5300 (July).

Kalaanidhi, S., Gunasekaran, K., 2013. Estimation of bus transport ridership accounting accessibility. Procedia Soc. Behav. Sci. 104, 885–893. http://linkinghub.elsevier.com/retrieve/pii/S1877042813045746.

Khani, A., 2018. akhani/Schedule-based-transit-shortest-path: SBTSP akhani/Schedule-based-transit-shortest-path: SBTSP. URL: doi: 10.5281/zenodo.1341312.

Khani, A., 2019. An online shortest path algorithm for reliable routing in schedule-based transit networks considering transfer failure probability. Transp. Res. Part B: Methodol.

Khani, A., Hickman, M., Noh, H., 2014. Trip-based path algorithms using the transit network hierarchy. Netw. Spatial Econ. 15 (3), 635–653.

Khani, A., Lee, S., Hickman, M., Noh, H., Nassir, N., 2012. Intermodal path algorithm for time-dependent auto network and scheduled transit service. Transp. Res. Rec. J. Transp. Res. Board 2284 (2284), 40–46 http://trrjournalonline.trb.org/doi/10.3141/2284-05.

Koffman, D., 2004. Operational Experiences with Flexible Transit Services. National Academies Press.

Kumar, P., 2017. Static Traffic Assignment using User Equilibrium and Stochastic User Equilibrium- Python Code. URL: doi: 10.5281/zenodo.1045573.

Kumar, P., Khani, A., 2019. prameshk/GTFS-to-schedule-based-transit-network: GTFS to schedule-based transit network. URL: doi: 10.5281/zenodo.2531085.

Kumar, P., Khani, A., He, Q., 2018. A robust method for estimating transit passenger trajectories using automated data. Transp. Res. Part C: Emerg. Technol. 95.

Lee, A., Savelsbergh, M., 2017. An extended demand responsive connector. EURO J. Transp. Logist. 6 (1), 25–50. https://doi.org/10.1007/s13676-014-0060-6.

Li, X., Quadrifoglio, L., 2009. Optimal zone design for feeder transit services. Transp. Res. Rec. J. Transp. Res. Board 2111 (1), 100–108.

Ma, T.Y., Rasulkhani, S., Chow, J.Y., Klein, S., 2019. A dynamic ridesharing dispatch and idle vehicle repositioning strategy with integrated transit transfers. Transp. Res. Part E: Logist. Transp. Rev. 128 (July), 417–442. https://doi.org/10.1016/j.tre.2019.07.002.

Maheo, A., Kilby, P., Van Hentenryck, P., 2017. Benders decomposition for the design of a hub and shuttle public transit system. Transp. Sci. 53 (1), 77–88.

Mahéo, A., Kilby, P., Van Hentenryck, P., 2019. Benders decomposition for the design of a hub and shuttle public transit system. Transp. Sci. 53 (1), 77–88.

Martens, K., 2004. The bicycle as a feedering mode: experiences from three European countries. Transp. Res. Part D: Transp. Environ. 9 (4), 281–294.

Martin, E.W., Shaheen, S.A., 2014. Evaluating public transit modal shift dynamics in response to bikesharing: A tale of two U.S. cities. J. Transp. Geogr. 41, 315–324. https://doi.org/10.1016/j.jtrangeo.2014.06.026.

Kumar, P., Khani, A., Davis, G.A., 2019. Transit route origin-destination matrix estimation using compressed sensing. Transp. Res. Rec. 0(0), 0361198119845896. URL: doi: 10.1177/0361198119845896.

Masoud, N., Jayakrishnan, R., 2017a. A decomposition algorithm to solve the multi-hop Peer-to-Peer ride-matching problem. Transp. Res. Part B: Methodol. 99, 1–29.

Masoud, N., Jayakrishnan, R., 2017b. A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system. Transp. Res. Part B: Methodol. 106, 218–236. URL: doi: 10.1016/j.trb.2017.10.006.

Masoud, N., Nam, D., Yu, J., Jayakrishnan, R., 2017. Promoting peer-to-peer ridesharing services as transit system feeders. Transp. Res. Rec. J. Transp. Res. Board 2650, 74–83 http://trrjournalonline.trb.org/doi/10.3141/2650-09.

Masson, R., Lehuédé, F., Péton, O., 2014. The dial-a-ride problem with transfers. Comput. Oper. Res. 41 (1), 12–23. https://doi.org/10.1016/j.cor.2013.07.020.

Miller, H.J., 2017. Time Geography and Space-Time Prism', International Encyclopedia of Geography: People, the Earth, Environment and Technology, pp. 1–19. URL: http://doi.wiley.com/10.1002/9781118786352.wbieg0431.

Minnesota Geospatial Commons, 2016. Metro Transit Schedule Data - General Transit Feed Specification (GTFS). URL: https://gisdata.mn.gov/dataset/us-mn-state-metc-trans-transit-schedule-google-fd.

Moniruzzaman, M., Páez, A., 2012. Accessibility to transit, by transit, and mode share: application of a logistic model with spatial filters. J. Transp. Geogr. 24, 198–205. https://doi.org/10.1016/j.jtrangeo.2012.02.006.

Najmi, A., Rey, D., Rashidi, T.H., 2017. Novel dynamic formulations for real-time ride-sharing systems. Transp. Res. Part E: Logist. Transp. Rev. 108 (October), 122–140. https://doi.org/10.1016/j.tre.2017.10.009.

Nassir, N., Khani, A., Hickman, M., Noh, H., 2012. Algorithm for intermodal optimal multidestination tour with dynamic travel times. Transp. Res. Rec. J. Transp. Res. Board 2283, 57–66 http://trrjournalonline.trb.org/doi/10.3141/2283-06.

Nassir, N., Khani, A., Lee, S., Noh, H., Hickman, M., 2011. Transit stop-level origin-destination estimation through use of transit schedule and automated data collection system. Transp. Res. Rec. J. Transp. Res. Board 2263, 140–150 http://trrjournalonline.trb.org/doi/10.3141/2263-16.

Ordóñez, F., Dessouky, M.M., Ord, F., 2017. INFORMS Tutorials in Operations Research Dynamic Ridesharing Dynamic Ridesharing (October).

Quadrifoglio, L., Dessouky, M.M., Ordóñez, F., 2008. A simulation study of demand responsive transit system design. Transp. Res. Part A: Policy Pract. 42 (4), 718–737.

Rietveld, P., 2000. The accessibility of railway stations: The role of the bicycle in The Netherlands. Transp. Res. Part D: Transp. Environ. 5 (1), 71–75.

Santi, P., Resta, G., Szell, M., Sobolevsky, S., Strogatz, S.H., Ratti, C., 2014. Quantifying the benefits of vehicle pooling with shareability networks. Proc. Nat. Acad. Sci. 111 (37), 13290–13294 http://www.pnas.org/lookup/doi/10.1073/pnas.1403657111.

Santos, A., McGuckin, N., Nakamoto, H.Y., Gray, D., Liss, S., 2009. 2009 National Household Travel Survey. Security (July), Available: http://nhts.ornl.gov/download.shtml. URL: http://nhts.ornl.gov.

Sheffi, Y., 1985. Urban Transportation Networks, vol. 6. Prentice-Hall, Englewood Cliffs, NJ.

Shen, C.-W., Quadrifoglio, L., 2012. Evaluation of zoning design with transfers for paratransit services. Transp. Res. Rec. J. Transp. Res. Board 2277 (1), 82–89.

Stiglic, M., Agatz, N., Savelsbergh, M., Gradisar, M., 2015. The benefits of meeting points in ride-sharing systems. Transp. Res. Part B: Methodol. 82, 36–53. https://doi.org/10.1016/j.trb.2015.07.025.

Stiglic, M., Agatz, N., Savelsbergh, M., Gradisar, M., 2018. Enhancing urban mobility: Integrating ride-sharing and public transit. Comput. Oper. Res. 90, 12–21. https://doi.org/10.1016/j.cor.2017.08.016.

Sutherland, W., Jarrahi, M.H., 2018. The sharing economy and digital platforms: A review and research agenda. Int. J. Inf. Manage. 43 (February), 328–341. https://doi.org/10.1016/j.ijinfomgt.2018.07.004.

Tafreshian, A., Masoud, N., Yin, Y., 2020. Frontiers in Service Science: Ride Matching for Peer-to-Peer Ride Sharing: A Review and Future Directions. Service Science (November).

Waller, S.T., Ziliaskopoulos, A.K., 2002. On the online shortest path problem with limited Arc cost dependencies. Networks 40 (4), 216–227.

Wang, H., 2017, Routing and scheduling for a last-mile transportation system. Transp. Sci. 53(December 2018), trsc.2017.0753. URL: http://pubsonline.informs.org/doi/10.1287/trsc.2017.0753.

Wang, X., Agatz, N., Erera, A., 2017. Stable matching for dynamic ride-sharing systems. Transp. Sci. (November), trsc.2017.0768. URL: http://pubsonline.informs.org/doi/10.1287/trsc.2017.0768.

Wang, X., Yang, H., Zhu, D., 2018. Driver-Rider Cost-Sharing Strategies and Equilibria in a Ridesharing Program. Transportation Science Publication (July).

Webb, A., Khani, A., 2020. Park-and-ride choice behavior in a multimodal network with overlapping routes. Transp. Res. Rec. 2674 (3), 150–160. https://doi.org/10.1177/0361198120908866.