Multimodal route planning with public transport and carpooling

Huang, Haosheng ; Bucher, Dominik ; Kissling, Julian ; Weibel, Robert ; Raubal, Martin

Abstract: Increasing mobility demands raise the pressure on existing transport networks. As the most used mode of transport, private cars have a particularly strong environmental impact and produce congestion. Ridesharing or carpooling, where a driver and several riders form a carpool, can help to address these issues by increasing the number of persons per car. Therefore, recent years have seen a strong interest in carpooling. However, there exists no effective method of integrating carpooling into transport trip planners as of now, mainly due to the fuzzy and flexible nature e.g., no fixed stops, possibility of making detours of carpooling. This hinders the acceptance of carpooling by the general public. This paper proposes a new method to merge public transport and carpooling networks for multimodal route planning, considering the fuzziness and flexibility brought by carpooling. It is based on the concept of drive-time areas and points of action. The evaluation with real-world data sets shows that, compared with the state-of-the-art method, the proposed method merges static i.e., public transport and dynamic/fuzzy i.e., carpooling networks better, while retaining the desired flexibility offered by the latter, and thus creates a higher interconnectivity between the networks. Meanwhile, the merged network enables multimodal route planning, which can provide users with trips from an origin to a destination using different combinations of modes.

DOI: https://doi.org/10.1109/tits.2018.2876570

Multimodal Route Planning With Public Transport and Carpooling

Haosheng Huang, Dominik Bucher, Julian Kissling, Robert Weibel, and Martin Raubal

***Abstract*— Increasing mobility demands raise the pressure on existing transport networks. As the most used mode of transport, private cars have a particularly strong environmental impact and produce congestion. Ridesharing or carpooling, where a driver and several riders form a carpool, can help to address these issues by increasing the number of persons per car. Therefore, recent years have seen a strong interest in carpooling. However, there exists no effective method of integrating carpooling into transport trip planners as of now, mainly due to the fuzzy and flexible nature (e.g., no fixed stops, possibility of making detours) of carpooling. This hinders the acceptance of carpooling by the general public. This paper proposes a new method to merge public transport and carpooling networks for multimodal route planning, considering the fuzziness and flexibility brought by carpooling. It is based on the concept of drive-time areas and points of action. The evaluation with real-world data sets shows that, compared with the state-of-the-art method, the proposed method merges static (i.e., public transport) and dynamic/fuzzy (i.e., carpooling) networks better, while retaining the desired flexibility offered by the latter, and thus creates a higher interconnectivity between the networks. Meanwhile, the merged network enables multimodal route planning, which can provide users with trips from an origin to a destination using different combinations of modes.**

***Index Terms*— Carpooling, ridesharing, multimodal route planning, public transport, network linking.**

I. Introduction

MOBILITY has become more important over the past few decades and will very likely continue to do so. Increasing mobility in and between cities raises the pressure on existing transport networks. To meet this increasing demand, many cities around the world continuously improve their public transport infrastructure, and motivate people to use public transport such as subways, buses and trains. However, private cars are still the most used means of transport, as shown in a study which investigates the usage patterns of different transport modes in 218 European cities between 2001 and 2011 [23]. The extensive use of private cars has led to traffic congestions in urban and sub-urban areas, and brings about many societal and environmental issues. Carpooling or ridesharing, where a car driver and one or more riders form a carpool, is a promising solution to address these issues by increasing the number of persons per car [3], [6], [14]. Many carpooling platforms exist on the Internet, such as BlaBlaCar, e-carpooling.ch, and carpoolworld.com.

Specifically, in carpooling, a driver and one or more riders share a part of their common itinerary using the driver's car as well as a part of the travel expenses [1]. Most drivers are willing to make a short detour to pick up riders, which increases flexibility. In addition, traveling using carpooling is often much cheaper compared to public transport. All these circumstances make carpooling a very attractive mode of transport. However, stops defined by carpooling drivers are often not the same as a user's desired origin and destination, which prevents users from sharing rides. To further promote and make full use of carpooling, there is a strong need to develop multimodal transport planning systems that combine public transport and carpooling [1], [6]. These multimodal planners will allow users to search for a trip from an origin to a destination using different combinations of public transport and carpooling.

Research on merging public transport and carpooling for multimodal route planning is still at an early stage, and only a few studies exist so far [1], [6]. This might be due to several reasons. On the one hand, carpooling is still considered as a niche market and therefore not attractive financially. On the other hand, the dynamism, fuzziness and flexibility of carpooling make it challenging to integrate carpooling with public transport for multimodal route planning. Normally, multimodal routing networks are built on static networks, which consist of fixed routes and stops (e.g., bus stops) [4]. The basic principle of multimodal networks is network merging, where different transport networks are merged into one single routable graph. This is usually done by applying spatial methods like nearest neighbor algorithms to link nearby nodes in different networks [4], [21]. The crucial point is that this cannot be done easily or efficiently with carpooling, since no fixed stops exist and carpooling is of a fuzzy nature, which means detours can be made and therefore new stops can be exploited. New methods must be developed to efficiently merge the static (i.e., public transport) and dynamic/fuzzy (i.e., carpooling) networks, while retaining the desired flexibility offered by carpooling.

Manuscript received May 14, 2018; revised September 28, 2018; accepted October 4, 2018. This work was supported in part by the Swiss National Science Foundation within NRP 71 Managing Energy Consumption and in part by the Swiss Innovation Agency Innosuisse within the Swiss Competence Center for Energy Research (SCCER) Mobility. The Associate Editor for this paper was M. Mesbah. *(Corresponding author: Haosheng Huang.)*

H. Huang, J. Kissling, and R. Weibel are with the GIScience Center, University of Zurich, 8057 Zürich, Switzerland (e-mail: haosheng.huang@geo.uzh.ch; robert.weibel@geo.uzh.ch; ju.kissling@gmail.com).

D. Bucher and M. Raubal are with the Institute of Cartography and Geoinformation, ETH Zürich, 8093 Zürich, Switzerland (e-mail: dobucher@ethz.ch; mraubal@ethz.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2876570

In recognition of this research gap, this article proposes a new method to merge public transport and carpooling networks to facilitate multimodal route planning. Firstly, we apply the principle of time-expanded models to represent and model public transport and carpooling networks. Secondly, a merging technique, which can represent fuzzy locations and retain flexibility, is presented to merge the two networks into one single routable graph. In particular, we apply the concept of drive-time areas and points of action for this process. The proposed method is evaluated in an experiment with the Swiss railway network and real-life carpooling offers. We demonstrate that the proposed method creates a high interconnectivity between carpooling and the railway network. In addition, the merged network enables multimodal route planning, which can provide users with a trip from an origin to a destination using different combinations of public transport and carpooling.

The remainder of this article is organized as follows. Section II presents related work. In Section III, we describe the proposed method. Section IV reports on the evaluation, and Section V discusses the results. We draw conclusions and present an outlook on future work in Section VI.

II. Related Work

A. Network Modeling and Route Planning

Route planning can be described as the problem of finding a path through a graph. A graph consists of *nodes* or *vertices*, which are connected by *edges* (often carrying a weight, such as distance or travel time). In many cases, the optimal path is defined by having the minimal sum of weights (i.e., minimal cost) along edges from a source to a destination node. Dijkstra's shortest path algorithm [12] arguably is the best known route planning algorithm. Based on it, many speedup and optimization techniques have been proposed, resulting in several other route planning algorithms, such as A* [18], ALT (A*, landmarks and triangle inequality) [17], ArcFlags [19] and hierarchical approaches [15].

While a road network can usually be mapped to a graph relatively straightforwardly, the case is less clear for other transport networks, such as public transport. Even though trains and buses also run on static routes, their time-dependence (via timetables) requires specific modeling techniques [20], [21]. Three popular models exist for representing public transport and their timetables (which include a set of time-dependent events, i.e., departure and arrival times): *time-expanded*, *time-dependent*, and *frequency-based* models.

- In *time-expanded* models [4], [21], [22], public transport stops (e.g., bus stops) are simply modeled as "stop"-labeled nodes. Other nodes represent time events from the timetable, and directed edges between these nodes are added whenever it is possible to transfer from one to the other (i.e., the departure of one transport vehicle is after the arrival of the other). These time nodes are also linked to nodes of their owning stops. While these models often lead to a large number of nodes and edges, standard route planning algorithm and their speedup techniques can be directly applied on the resulting graph.
- *Time-dependent* models contain an edge between stop nodes if at least one connection exists (e.g., a bus line). Weights of these edges are determined as a function of time to reflect the time events of these stops [4], [7]. These models often have much smaller graph sizes than time-expanded models, however, route planning based on them becomes more complicated and time-consuming due to the dynamic nature of the edge weights.
- *Frequency-based* models [5] use the regularity of many public transport networks to compress the graph and reduce the query times. They store departure times, intervals and frequencies instead of time-dependent functions.

In essence, time-dependent and frequency-based models shift processing complexity to route planning algorithms, while time-expanded models can be preprocessed better. Delays and uncertainty are common in public transport which makes it of essence that a graph can quickly be updated according to the traffic state of the transport network [10], [13]. Several methods have been proposed to make these updates [4]: *repairing*, *bypassing* wrong parts and *metric splits*.

Route planning on public transport networks is hardly purely based on travel distance, but often considers other criteria, such as duration, and costs [16], [21]. Several typical routing problems are often discussed in the literature: routing with a given departure time range, routing with a given arrival time range, and multi-criteria routing.

B. Multimodal Route Planning

Integrating different modes of transport (such as walking, cycling, car travel, or public transport) is commonly referred to as multimodal route planning if the involved networks (physically or conceptually) differ [6], [8]. Multimodal route planning can be achieved by merging the involved network graphs into a single graph, and applying routing algorithms on the merged graph. A common approach for merging different networks is to solve the nearest neighbor problem (NN) [21], which simply connects spatially close stops from different networks. Specifically, a stop node A in network G is connected with its closest stop node B in another network G'. Additional conditions are often respected, e.g., the distance between A and B should be shorter than a certain threshold.

To retain the transport mode information after linking different networks, nodes and edges in the merged multimodal graph usually carry a label denoting the mode of transport [11].

C. Carpooling

In carpooling or ridersharing, a driver and one or several riders share part of their common trips using the driver's car [1]. This potentially helps to reduce the societal and environmental impacts brought by car travels. Therefore, carpooling has become more and more popular in recent years.

While carpooling offers several economical and ecological benefits, its integration into transport trip planners has not been studied well previously. Aissat and Varone present one of only few studies on this aspect [1]. They simply use an existing multimodal transport planner and try to substitute

each of the individual parts of a potential trip with carpooling. In their approach, an original part of the trip is only replaced if the carpooling substitution (including waiting times) is faster than the original. To find potential carpooling offers, they compute all possible trips from carpooling pickup locations to later stops on the original trip. Another approach was proposed by [6]. They define the public transport network as the riders' network and the road network as the carpooling drivers' network. The idea, then, is to compute multiple shortest paths, and synchronize the trips of riders and drivers. This two-synchronization-point shortest path problem is made up of a composition of 5 sub-paths, where two of them lead to a carpooling pickup point, one from the pickup to the carpooling drop-off, and two from the drop-off location to the final destination of rider and driver. The pickup and drop-off points are chosen by minimizing the summarized travel cost of the involved parties. There are also many studies developing methods to identify meeting points for car drivers and riders [2], [9]. However, these methods always keep a specific carpooling offer in mind, and do not try to find a best carpooling offer from a set of possible choices.

While the above studies present some first attempts to integrate carpooling into multimodal transport networks, they still do not fully respect the inherent flexibility offered by carpooling. For example, carpooling is not restricted to a fixed route, and often allows detours. This potentially allows the driver to pick up someone not directly at the stops he/she defined or somewhere along the planned route. In other words, many other pickup locations are potentially possible. The state-of-the-art method NNP [4], [21], which is commonly used for integrating different transport modes, also cannot effectively address these issues. In recognition of this research gap, this article proposes a new method to merge public transport and carpooling networks for multimodal route planning, considering the fuzziness and flexibility brought by carpooling.

III. Methodology

This section introduces our methodology for multimodal route planning, combining public transport and carpooling. We first explore how public transport and carpooling networks can be modeled using time-expanded models (Sections III-A and III-B). Next, a merging technique is proposed to link these two networks, considering the fuzziness/flexibility of carpooling (Section III-C). Finally, we illustrate how the merged network can be used for multimodal routing (Section III-D).

A. Modeling of Public Transport Networks

In general, public transport relies on a timetable and operates on established routes. A timetable consists of a set of stations S, a set of public transport vehicles V (e.g., trains, subways, and buses), a set of times T, and a set of elementary connections C. An elementary connection $c \in C$ is a tuple $c = (v, s, t_s, e, t_e)$, denoting that a public transport vehicle $v \in V$ starts at a specified station $s \in S$ at time $t_s \in T$, and arrives at station $e \in S$ at time $t_e \in T$. Therefore, the travel time of a connection is simply $t_e - t_s$. Please note that different vehicles lead to different t_s and t_e values. An elementary connection has no stops in between. It can therefore be interpreted as a leg of a trip $l \ni c$. A trip l is a list of sequential connections, and can be described as a tuple $l = (c_1, \ldots, c_n)$. In a trip l, every connection c has the same public transport vehicle v. Thus a trip represents the whole travel of a certain vehicle (at a fixed time). A route r consists of multiple trips visiting the same stops: $r = \{l_1, \ldots, l_n\}$.

To illustrate the above definitions, we use a fictional example. Assume that a public transport agency offers journeys from *New York* (NY) via *Washington* (DC) to *Miami* (MI) without any stops in between. Hence, it offers the route $r_{(NY\text{-}DC\text{-}MI)}$. There are four trains per day undertaking the route. Hence four trips l_{1-4} exist. These trips are time-dependent, since they leave at specific times. On each trip the trains stop in DC. Thus, each trip consists of two legs or connections c (NY-DC and DC-MI). If the first train leaves at 10:00, arrives at 14:00 in DC, leaves there at 14:05, and finally reaches MI at 23:00, the connections are defined as $c_1 =$ (train1, NY, 10:00, DC, 14:00), and $c_2 =$ (train1, DC, 14:05, MI, 23:00).

In the following, we transform the above defined public transport network into a time-expanded graph model. At this point, the following nodes and edges are defined (Figure 1).

- Stop node $s_i^{PT} \in S$. Each stop node holds information about a specific stop, e.g., name, geographic location, and wheelchair accessibility.
- Time node t_i^{PT}: A time node defines both the arrival and departure time of a public transport vehicle v at the stop s_i^{PT}. Let $Arrival(t_i^{PT})$ and $Departure(t_i^{PT})$ be the arrival and departure time. Thus, $Arrival(t_i^{PT}) < Departure(t_i^{PT})$.
- Edge (s_i^{PT}, t_i^{PT}): Mark the affiliation of a time node t_i^{PT} to a stop s_i^{PT}.
- Edge (t_i^{PT}, t_{i+1}^{PT}): The edge between two time nodes t_i^{PT} and t_{i+1}^{PT} represents the actual connection or a leg of a trip. It is a directed edge. This edge is weighted by a certain metric. In our case, the weight is set as the travel duration $Arrival(t_{i+1}^{PT}) - Departure(t_i^{PT})$.
- Trip node l^{PT}: A trip node represents a set of connections. It can hold information about a specific trip, e.g., vehicle types, and wheelchair accessibility.
- Edge (t_i^{PT}, l^{PT}): Mark the affiliation of a time node t_i^{PT} to a trip l^{PT}. Hence, each trip l^{PT} has edges with multiple time nodes.
- Route node r^{PT}: A route node represents a set of trips. It can hold information about a specific route, e.g., names.
- Edge (r^{PT}, l^{PT}): Mark the affiliation of a trip node l^{PT} to a route r^{PT}. Hence, each route r^{PT} has edges with multiple trip nodes.

This model works for single ride journeys as trips are represented separately. To retrieve a shortest or fastest path, standard routing algorithms such as Dijkstra's and A* can be applied on the sub-graph of the time nodes.

Modeling transfers needs further adaptations to the model. Other than [21], we do not implement an additional transfer node, but an additional edge between time nodes of different trips if a transfer is feasible there. For example, in Figure 2, a transfer edge (t_i^{PT}, t_j^{PT}) is created between time nodes t_i^{PT}

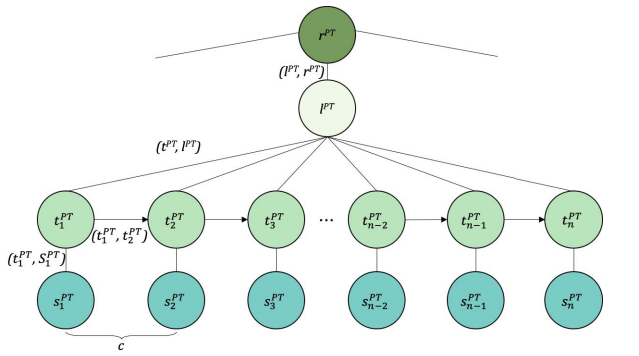


Fig. 1. A time-expanded model of a public transport network: A trip is composed of a sequence of time nodes (t_i^{PT}) at different stop nodes (s_i^{PT}), and it belongs to a route.

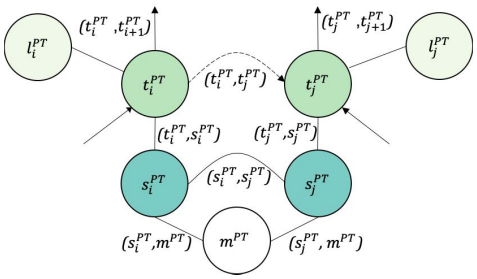


Fig. 2. Representing transfers in the time-expanded model. The dashed edge (t_i^{PT}, t_j^{PT}) between two time nodes indicates that a transfer from trip l_i^{PT} to trip l_j^{PT} is possible at stops s_i^{PT} and s_j^{PT}, both of which belong to the meta-stop m^{PT}.

and t_j^{PT}, where t_i^{PT} belongs to trip l_i^{PT} and t_j^{PT} belongs to another trip l_j^{PT}. The weight of this edge is set as the duration $Departure(t_j^{PT}) - Arrival(t_i^{PT})$. Please note that multiple transfer edges might be created, as several transfers from t_i^{PT} to other time nodes might be feasible. These transfer edges are labeled as "transfer" to differentiate them from original parts of a trip. Threshold values are often used to check whether a transfer is feasible or not. For example, if (3 min $< Departure(t_j^{PT}) - Arrival(t_i^{PT}) <$ 10 min) holds, only then the transfer edge (t_i^{PT}, t_j^{PT}) is created. This condition defines transfer time in a static manner: the transfer time from stop s_i^{PT} to s_j^{PT} should be greater than 3 minutes, but no longer than 10 minutes. Considering the fact that multiple stops, or more precisely, stop points (e.g., platforms), might belong to a same meta-stop (e.g., a train station), we introduce a meta-stop node m^{PT} and relate it with its child stops. Furthermore, an edge (s_i^{PT}, s_j^{PT}) between every child stop is created representing the variable transfer time. Please note that if two stops do not belong to the same meta-stop but are within walking distance, an edge between them can also be created. The conditional for creating transfer edge (t_i^{PT}, t_j^{PT}) can consequently be retrieved from (s_i^{PT}, s_j^{PT}). Implementing variable transfer times adds complexity to the model. Information regarding the edge (s_i^{PT}, s_j^{PT}) does not necessarily have to be stored in the graph itself. An alternative solution is to store pairs of child stops and their min./max. transfer times in an external data table or even a text file.

Algorithm 1 shows the algorithm to create transfer edges between time nodes. It has two inputs: a time-expanded graph representing all single ride journeys, and a set of transfer conditions between stops. Each condition defines what makes a transfer between two stops feasible. The algorithm checks the difference between the arrival time at a stop and the departure time at another stop with the transfer condition of these two stops. If the condition is met, a directed transfer edge with the time difference as its weight is created.

By this adaptation to the original time-expanded model, journeys with transfers (e.g., a journey from A to C, with transfer at B) can be retrieved by running a routing algorithm on the resulting graph.

B. Modeling of Carpooling

Before modeling carpooling data, we need to gain a better understanding of the characteristics of these data. By analyzing offers from several existing carpooling platforms (BlaBlaCar, e-carpooling.ch, carpoolworld.com), we found that a carpooling offer typically consists of an *origin* (pick-up point), a *destination* (drop-off point), and optionally a set of *stopover* points that are along the way. The start time of the journey is also given, as well as prices for the whole journey and sometimes for each leg. The stop locations are usually not given as exact longitude/latitude pairs, but rather as city names or town names. Drivers can optionally specify a detour time, which is usually between 0 and 30 minutes. This fuzziness in terms of stop locations and detour times is a major difference compared to public transport. In the following, we illustrate how carpooling data can be represented in a time-expanded model, and in Section III-C, we will show how this fuzziness helps to link public transport and carpooling network graphs.

Similarly to how we modeled public transport network in Section III-A, we model carpooling as a time-expanded graph, as shown in Figure 3. It consists of a set of stop nodes, representing the origin (s_1^{CP}), the destination (s_n^{CP}) and a set of stopovers ($s_2^{CP}, \ldots, s_{n-1}^{CP}$) of the carpooling offer. In contrast to public transport, carpooling does not directly provide the information of arrival times at the stopover locations and the destination, which have to be derived by routing the offer on a road network. Therefore, we label the time nodes of stopovers and destination as "derived" time nodes ($t_2^{CP\prime}, \ldots, t_n^{CP\prime}$). For each derived time node, we simply set its departure time the same as its arrival time, as dropping off a passenger is often very fast.

To enable transfers between different carpooling offers, the same procedure used for public transport can be applied.

C. Model Merging and Linking

After modeling public transport and carpooling as time-expanded graphs, the next step is to merge and link these graphs, considering the fuzziness and flexibility brought by carpooling. As both graphs use the same types of nodes, merging them is straightforward. However, linking individual nodes from the two merged graphs requires more work.

Algorithm 1 Creating Transfer Edges Between Time Nodes

Input. A time-expanded graph $G = (V, E)$ representing all single ride journeys; a set of transfer conditions TC between stops, and each transfer condition tc as a tuple $(s_1, s_2, min_{dur}, max_{dur})$, meaning that the transfer time from stop s_1 to s_2 should be greater than min_{dur} and no longer than max_{dur}.
Output. A new time-expanded graph $G' = (V, E')$ that is the original graph G enriched with possible transfer edges.

$G' \leftarrow G$
for all tc in TC **do**
  $V_1 \leftarrow$ all the time nodes that are linked to $tc.s_1$
  $V_2 \leftarrow$ all the time nodes that are linked to $tc.s_2$
  // Find feasible transfers from V_1 to V_2.
  for all time node v_1 in V_1 **do**
    for all time node v_2 in V_2 **do**
      $dur \leftarrow$ departure time of v_2 - arrival time of v_1
      // Check whether a transfer is feasible here; If yes, add a directed transfer edge from time node v_1 to v_2 to G'.
      if $dur > tc.min_{dur}$ and $dur < tc.max_{dur}$ **then**
        Add the directed edge (v_1, v_2) to G'; Set its weight as dur; Label the edge as "transfer"
return G'

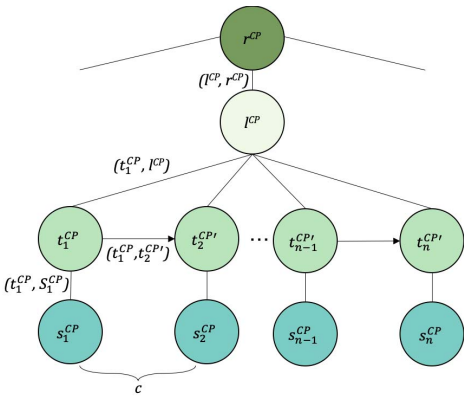


Fig. 3. A time-expanded model of carpooling. Similar to public transport, a trip is composed of a sequence of time nodes (t_i^{CP}) at different stop nodes (s_i^{CP}), and it belongs to a route.

1) Linking via Origins, Destinations, and Stopovers of Carpooling: As mentioned before, carpooling offers usually do not specify exact locations for origins, destinations, and stopovers. It is common to only mention the cities an offer starts from, arrives in, and passes by. This inaccuracy brings some challenges to link nodes from the public transport and carpooling graphs. As carpooling does not have exact stops, it is unclear how and to which public transport stop a carpooling offer can be connected. In case of connecting a carpooling stop (either start, destination, or stopover) to its nearest public transport stop in the same city using the state-of-the-art method, the flexibility of carpooling is lost, because theoretically, the carpooling driver could pick up or drop off the passenger at any stop within the city. However, connecting a carpooling stop to every public transport stop within the same city does not seem to be ideal, either. In a large city, it would be inconvenient for the driver to deviate much from the shortest path, simply to drop someone off.

To address the above issues, we firstly use existing geocoding APIs to convert each carpooling stop name into geographic coordinates (i.e., a longitude/latitude pair). Potential geocoding solutions are provided by Google, Geonames and others. At a second step, we check which public transport stops can potentially be reached from this longitude/latitude pair, considering the maximum detour restriction. To achieve this aim, we apply the concept of drive-time areas (DTAs). A DTA is a zone/polygon around a spatial feature measured in units of time needed for travel by car.[1] For example, a supermarket's 20-minute DTA defines the area in which drivers can reach the supermarket in 20 minutes or less. DTA can be generated by using actual street networks and speed limits or more accurately with approximated driving times derived from historical traffic data. Some existing solutions exist for this purpose, such as ESRI SteetMap Premium. We compute DTAs around public transport stops, and check if a carpooling stop lies within a DTA. Computing DTAs around public transport stops instead of carpooling stops allows reusing these DTAs when linking other carpooling offers.

Thus, drive-time areas DTAs are implemented around public transport stops s^{PT}, using the detour time as parameter. Rather than just finding the nearest neighboring PT stop of a geocoded carpooling stop s^{CP}, as in the state-of-the-art method, a set of PT stops are determined by checking whether their DTAs contain s^{CP}. If the DTA of a particular PT stop s^{PT} contains the CP stop s^{CP}, it means that s^{PT} can be reached from s^{CP} within the detour limit. Therefore, s^{CP} can be linked to s^{PT}. With this, each carpooling stop can be linked to multiple PT stops. To reduce the number of edges in the merged graph, we only create an edge between s^{CP} and each s^{PT}'s meta-stop. Figure 4 shows an example where a carpooling stop s^{CP} falls within the DTAs of s_1^{PT} and s_2^{PT}. Consequently, edges for (m_1^{PT}, s^{CP}) and (m_2^{PT}, s^{CP}) are created, where m_1^{PT} and m_2^{PT} are the meta-stops of s_1^{PT} and s_2^{PT}.

[1]https://support.esri.com/en/other-resources/gis-dictionary/term/drive-time%20areahttps://support.esri.com/en/other-resources/gis-dictionary/term/drive-time%20area

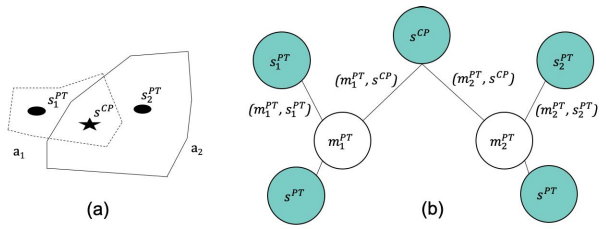


Fig. 4. A carpooling stop is contained by drive-time areas of two public transport stops. (a) the example; (b) schematic representation of the carpooling stop linked to the meta-stops of the two public transport stops.

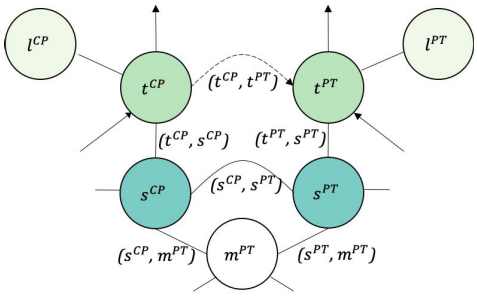


Fig. 5. Schematic representation of a transfer between carpooling and public transport. If a given conditional is true, a transfer edge (t^{CP}, t^{PT}) is created.

After linking the public transport and carpooling stops, we can then check whether a transfer from carpooling to public transport, or vice versa, is possible. Following the same procedure used for representing transfers in public transport (Section III-A), we employ threshold values to check whether a transfer is feasible or not. Hence, a directed transfer edge (t^{CP}, t^{PT}) is created if a certain conditional, e.g., (3min $< Departure(t^{PT}) - Arrival(t^{CP}) <$ 10 min), is true (Figure 5). The weight of the edge is set as $Departure(t^{PT}) - Arrival(t^{CP})$. Note that multiple transfer edges might be created, as several transfers from t^{CP} to other time nodes might be feasible. Similarly, these transfer edges are labeled as "transfer" to differentiate them from original parts of a trip.

With this, the fuzziness of the inaccurate geographic locations of the origin, destination, and stopovers of carpooling offers can be represented. Furthermore, this leads to a more flexible system, since transfers to public transport can not only happen at a specific stop, but at multiple stops. This reflects the flexibility offered by carpooling.

2) Representing Fuzziness and Flexibility and Linking Along Carpooling Routes: Carpooling is potentially more flexible than public transport because it is not bound to predefined routes. Hence, a carpooling driver can drive detours which exploit new stops that are not directly on the planned route. In the following, we show how this flexibility can be used to link potential public transport stops around a carpooling route. As a prerequisite, the carpooling route will be derived by applying shortest-path algorithms on the given origin, destination, and stopovers. Several mapping APIs exist for this purpose, such as OpenRouteServices and the Google Directions API.

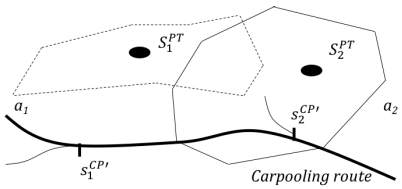


Fig. 6. A carpooling route crosses the drive-time area of a public transport stop s_2^{PT}.

We then again apply the concept of DTAs to exploit potential stops $s_i^{CP\prime}$ along a derived carpooling route. Half of the detour time is used as parameter when computing DTAs, as the driver needs to drive there and back. Similarly to the stop merging shown in the previous section, rather than relating $s_i^{CP\prime}$ to a specific s_j^{PT} stop, an edge $(m_j^{PT}, s_i^{CP\prime})$ is created between the meta-stop of s_j^{PT} and $s_i^{CP\prime}$. Figure 6 shows an example where a carpooling route crosses the DTA of s_2^{PT}, but not the DTA of s_1^{PT}. Therefore, an edge will be created between $s_2^{CP\prime}$ and the meta-stop of s_2^{PT}.

Please note that $s_i^{CP\prime}$ along a carpooling route cannot be chosen randomly or by checking the Euclidean distance between the route and s_j^{PT}, as s_j^{PT} might not be reachable from $s_i^{CP\prime}$. For example, if the carpooling route is a highway crossing the DTA of s_j^{PT} but without a ramp in the DTA, a driver would not be able to reach s_j^{PT}. To address the issue, we introduce the concept of Points of Action (POAs). A POA is defined as a point along the carpooling route where the driver needs to take an action, such as turning left, turning right, or continuing without changing direction. POAs are mainly crossings, junctions, or highway ramps. If a driver does not have to take an action, there will not be a POA. POAs of a carpooling route can be identified by overlapping the route with the road network. Consequently, for each POA, we check whether it is within the drive-time areas of public transport stops, as shown in Figure 6. If yes, this POA is a potential stop along the carpooling route, and we create an edge between this POA and the meta-stops of these public transport stops.

After linking public transport stops and POAs along the carpooling route, we can then check whether a transfer from carpooling to public transport, or vice versa, is possible. Following the same procedure used for representing transfers before (Sections III-A and III-C1), we employ threshold values to check whether a transfer is feasible or not. Hence, if a certain conditional, e.g., (10 min $< Departure(t^{PT}) - Arrival(t_{POA}^{CP'}) - travel\_time(s_{POA}^{CP}, s^{PT}) <$ 30 min), is true, we will create a time node t_{POA}^{CP} for this POA, and create a directed transfer edge $(t_{POA}^{CP'}, t^{PT})$, as shown in Figure 7. $Travel\_time(s_{POA}^{CP}, s^{PT})$ is the travel time from s_{POA}^{CP} to s^{PT}, and in the worst case, it is equal to half of the maximum allowed detour time. Therefore, in approximation, it can be set as half of the allowed detour time. The weight of the transfer edge is set as $Departure(t^{PT}) - Arrival(t_{POA}^{CP'})$. Note that multiple transfer edges might be created, as several transfers from $t_{POA}^{CP'}$ to other time nodes might be feasible.

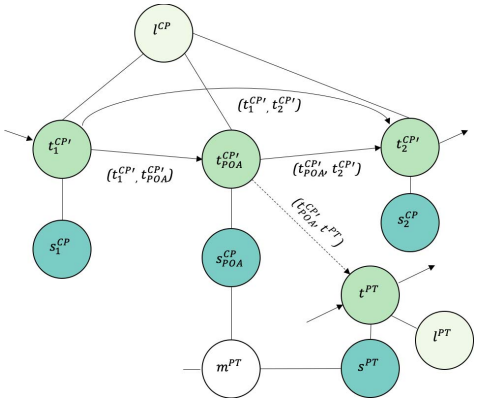


Fig. 7. Schematic representation of a transfer from a POA s_{POA}^{CP} of carpooling to a public transport stop s^{PT}. If a given conditional is true, a transfer edge $(t_{POA}^{CP'}, t^{PT})$ is created.

Similarly, these transfer edges are labeled as "transfer" to differentiate them from original parts of a trip. Furthermore, we also keep the original edge $(t_1^{CP'}, t_2^{CP'})$ to show that a stop at the POA is not a must, but rather optional.

The concept of POA has further benefits. Since POAs are located at crossings, junctions, and other significant points on the road network, different carpooling routes may exploit the same POA if they are passing the same crossing. Consequently, transfers between carpooling routes which intersect at a POA can be created using the same approach.

In case a driver makes a detour to s^{PT} via s_{POA}^{CP}, arrival and departure times at all later stops will change. In this paper, we assume that beyond the predefined origin, destination, and stopovers, a carpooling driver will drive maximally one detour along the route. This assumption is based on the observation that allowing an undefined or high number of detours would require the driver to drive many short trips with many detours, which is normally not acceptable for carpooling drivers and passengers. With this assumption, the arrival and departure times at each later stop do not need to be adjusted.

Algorithm 2 shows the algorithm combining the steps described above in Section III-C1 and in this section. When a new carpooling offer is created, the algorithm is used to add the offer to the public transport graph. By adding all the available offers, multimodal journeys with transfers between public transport and carpooling, between public transport, and between carpooling can be retrieved by running routing algorithms on the resulting directed graph.

D. Routing

By applying the above method, a time-expanded multi-model graph combining public transport and carpooling can be created. Standard routing algorithms such as Dijkstra's can be directly applied on the resulting graph to provide routes from an origin to a destination. In the following, we introduce techniques to provide routes that are often requested by users when using public transport or other time-dependent transport.

1) Routing With a Given Departure Time Range: Usually, a user wants to query routes from an origin A to a destination B, given a departure time range (e.g., leaving between 8:00 and 8:15). To query these routes, firstly, we identify all time nodes T_A that belong to the child stops of A and whose departure times are within the range. Secondly, all the times nodes T_B that belong to the child stops of B are identified. In the last step, the routing algorithm then computes routes from any of the time nodes in T_A to any of the time nodes in T_B, based on the sub-graph of the time nodes of the merged network. This will return a set of routes that depart from the origin within the given time range, which can then be ordered according to different criteria, such as the total travel time, the number of transfers, or simply arrival time at the destination. This gives users the possibility to choose the route they want to take.

To improve the performance of the above routing, some optimization can be introduced, such as instead of using all the time nodes at B, we can estimate the travel time between A and B, and use this to approximately select time nodes whose departure times are within a feasible range.

2) Routing With a Given Arrival Time Range: Similar to the above task, a user might want to arrive at the destination B by a certain time, but does not know when she should leave at the origin A. The same procedure can be applied. However, instead of reducing the set of time nodes at the origin, the set of time nodes at the destination is reduced according to the pre-defined arrival time range. This routing task can be solved in either a forward search or a backwards search. In the forward approach, the first step finds all the time nodes at the destination B that lies within the given arrival time range. In a second step, a standard routing algorithm is executed from any time node at the origin A to the selected time nodes at B. In the backwards approach, routing is performed in the opposite direction of the edge direction. Thus, a routing algorithm is executed from the selected time node at B to any time nodes at A. Similarly, the resulting routes can be ordered according to different criteria to help users choose a route to take.

IV. Evaluation and Results

To test the feasibility of the proposed method, we evaluate it using two real-world datasets from the Swiss Federal Railways (SBB) and a large European carpooling platform. The evaluation was conducted on a Lenovo ThinkPad T430s with an Intel i7-3520 CPU 2.9 GHz and 16 GB memory. The network graph was built and queried using Neo4j 3.0.8, a widely-used graph database. Neo4j was used for simplicity reasons, as it provides efficient graph storage, allows labeling nodes/edges, and comes with standard routing algorithms such as Dijkstra's.

A. Data

The railway network data in Switzerland is used in the evaluation as a representative of public transport data. While the mode of train may seem different from other modes (e.g., bus and subway) from an end user's perspective, it shares the main characteristics of public transport, which relies on a timetable and operates relatively stable on established routes. We retrieved the railway data in the

Algorithm 2 Merging Public Transport and Carpooling Graphs

Input. A time-expanded graph $G^{PT} = (V^{PT}, E^{PT})$ representing all public transport trips and transfers; a time-expanded graph $G^{CP} = (V^{CP}, E^{CP})$ representing a carpooling offer; a maximum detour time t_{detour} given by the CP driver; stops S^{CP} (i.e., origin s_o^{CP}, destination s_d^{CP} and stopovers S_{so}^{CP}) of the CP defined by the driver; a set of transfer conditions $tc' \in TC'$ at each PT stop where $tc' = (x, min_{dur}, max_{dur})$, defining the min. and max. time required for transferring between PT and CP at this PT stop.

Output. A new time-expanded graph $G' = (V', E')$ combining the PT and CP graphs.

$G' \leftarrow (V^{PT} \cup V^{CP}, E^{PT} \cup E^{CP})$ // Combine the PT and CP graphs.
Use existing geocoding API to get the geographic coordinates of all CP stops S^{CP}
Use existing routing API to get shortest route r from s_o^{CP} to s_d^{CP} via S_{so}^{CP}
Estimate arrival times at stopovers S_{so}^{CP} and destination s_d^{CP}

// Link each given CP stop to the PT graph (cf. Section III-C1).
for all CP stop $s^{CP} \in S^{CP}$ **do** // Find all nearby PT stops in an approximate way; they will be refined later.
  $S^{PT} \leftarrow$ all PT meta-stops where the Euclidean dist. to $s^{CP} < (t_{detour} \cdot 100$ km/h)
  for all PT stop $s^{PT} \in S^{PT}$ **do**
    $a \leftarrow$ DTA of s^{PT}, using t_{detour} as parameter
    if s^{CP} is within a **then** // This means that s^{PT} can be reached from s^{CP} within t_{detour}.
      Add edge (s^{CP}, s^{PT}) to G'
      // Check whether transfers between CP and PT are possible here.
      $V^{PT} \leftarrow$ all the time nodes that are linked to the child PT stops of s^{PT}
      for all Time node $v^{PT} \in V^{PT}$ **do**
        if Arrival time at $s^{CP} <$ departure time at v^{PT} **then**// This potentially allows transferring from CP to PT.
          $dur \leftarrow$ departure time at v^{PT} - arrival time at s^{CP}
          if $dur > min_{dur}$ at s^{PT} and $dur < max_{dur}$ at s^{PT} **then**
            Add directed edge $e = (t^{CP'}, v^{PT})$ to G'; // $t^{CP'}$: time node of s^{CP}
            Set e's weight as dur and its label as "transfer"
        if Arrival time at $v^{PT} <$ departure time at s^{CP} **then**// This potentially allows transferring from PT to CP.
          $dur \leftarrow$ departure time at s^{CP} - arrival time at v^{PT}
          if $dur > min_{dur}$ at s^{PT} and $dur < max_{dur}$ at s^{PT} **then**
            Add directed edge $e = (v^{PT}, t^{CP'})$ to G'; // $t^{CP'}$: time node of s^{CP}
            Set e's weight as dur and its label as "transfer"

// Exploit potential stops along CP route (cf. Section III-C2).
$S_{POA}^{CP} \leftarrow$ all POAs along CP route r, computed by overlapping r with road network
for all POA $s_{POA}^{CP} \in S_{POA}^{CP}$ **do**
  $S^{PT} \leftarrow$ all PT meta-stops where the Euclid. dist. to $s_{POA}^{CP} < (t_{detour}/2 \cdot 100$ km/h)
  for all PT stop $s^{PT} \in S^{PT}$ **do**
    $a \leftarrow$ DTA of s^{PT} (using $t_{detour}/2$ as parameter) // $t_{detour}/2$, because the driver needs to go to PT and back.
    if s_{POA}^{CP} within a **then** // This means that s^{PT} can be reached from s_{POA}^{CP} within $t_{detour}/2$.
      Add a new stop node s_{POA}^{CP} to G'; Add edge (s_{POA}^{CP}, s^{PT}) to G'
      Estimate the arrival time t at the CP POA s_{POA}^{CP}
      Add new time node $t_{POA}^{CP'}$ to G', using the estimated t as arrival/departure time
      // Create the time node for s^{CP}, and link it to the required nodes.
      Add edges $(t_{POA}^{CP'}, s_{POA}^{CP})$ and $(t_{POA}^{CP'}, l^{CP})$ to G' // l^{CP}: trip node of the CP
      Add directed edges between $t_{POA}^{CP'}$ and adjacent time nodes to G'
      Set the edge weights as the travel time between the corresponding stop nodes
      // Check whether transfers between CP and PT are possible here.
      $V^{PT} \leftarrow$ all the time nodes that are linked to the child PT stops of s^{PT}
      for all Time node $v^{PT} \in V^{PT}$ **do**
        if Arrival time at $s_{POA}^{CP} <$ departure time at v^{PT} **then** // This potentially allows transferring from CP to PT.
          $dur \leftarrow$ departure time at v^{PT} - arrival time at s_{POA}^{CP}
          if $dur - t_{detour}/2 > min_{dur}$ at s^{PT} and $dur - t_{detour}/2 < max_{dur}$ at s^{PT} **then**
            Add directed edge $e = (t_{POA}^{CP'}, v^{PT})$ to G'; Set e's weight as dur and its label as "transfer"
        if Arrival time at $v^{PT} <$ departure time at s_{POA}^{CP} **then** // This potentially allows transferring from PT to CP.
          $dur \leftarrow$ departure time at s_{POA}^{CP} - arrival time at v^{PT}
          if $dur - t_{detour}/2 > min_{dur}$ at s^{PT} and $dur - t_{detour}/2 < max_{dur}$ at s^{PT} **then**
            Add directed edge $e = (v^{PT}, t_{POA}^{CP'})$ to G'; Set e's weight as dur and its label as "transfer"

return G'

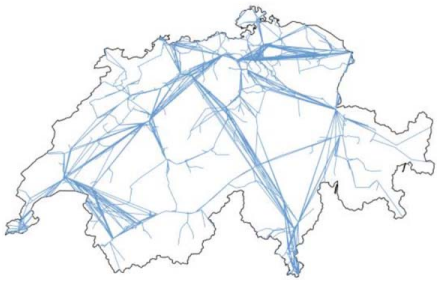


Fig. 8. Distribution of the 2'000 carpooling trips in the study.

Fig. 9. The 5-minute drive time areas computed around all train stops in Switzerland. Note that they are much smaller in the mountainous regions in the southern part of Switzerland.

General Transport Feed Specification format from geOps (http://gtfs.geops.ch/). In total, the data contains 1'912 railway stations and 28'455 routes, which consist of about 790'000 time-stamped stops at stations. The 1'912 stations consist of 4'683 platforms.

The carpooling data used within this work was retrieved for the largest 18 cities in Switzerland. Approx. 18'000 carpooling offers were retrieved from the platform within an 8-month period, and 2'000 of them were randomly selected for this evaluation to reduce the computational load. Each carpooling offer consists of a driver (who advertised the trip on the carpooling platform), an origin, a destination, a set of stopover points that are along the way (optional), a maximal detour time (optional), a price, and a capacity. To investigate the spatial characteristics of the carpooling data, we applied the geocoding and routing functions provided by Google's Directions API (http://developers.google.com/maps/documentation). Results show that carpooling offers tend to be for longer journeys, and the average length is 480 km. Please note that they often connect places not in the vicinity of a train station. We found that the 2'000 carpooling offers provided 878 independent routes where none of the origin, stopover, and destination stops are in the 5-minute driving vicinity of a train station. Figure 8 shows the distribution of the routes.

The merging process outlined in Algorithm 2 requires information about drive-time areas (DTAs) around public transport stops (i.e., train stations in this evaluation). The StreetMap Premium road network dataset from Esri (http://www.esri.com/data/streetmap) was used to compute these DTAs for all the stations, using 5 minutes as a threshold (Figure 9). We used this static value instead of a driver-defined detour time to reduce the computational load. However, it should be noted that the proposed method can deal with variable time durations set by the driver.

B. Network Graph Generation and Merging

After retrieving all the data, we then applied the method introduced in Section III-A on the railway data to generate its network graph. Algorithm 1 was then used to create transfers between different railway connections, and add them to the graph. We used the transfer conditional (3 min $<$ $Departure(t_b) - Arrival(t_a)$ $<$ 10 min) when creating transfers. In other words, we created respective transfer edges (e.g., between different trains) if the transfer time between two trains is greater than 3 minutes but no longer than 10 minutes. Setting this conditional is just for simplicity; the proposed method can deal with other user-defined ones. Meanwhile, each station node consists of its DTA (i.e., polygon) computed before.

We then added each carpooling offer to the railway network graph using the merging algorithm (Algorithm 2). The same transfer conditional used above was employed to check whether transferring from carpooling to railway, or vice versa, is feasible. Algorithm 2 also needs to have the Points of Action (POAs) of each carpooling route available. As mentioned in Section III-C2, POAs can be identified by overlapping the carpooling route with the road network. In this evaluation, we simply used the instruction steps returned by the Google Directions API (other existing routing APIs would also work) to identify POAs. In urban areas, the number of steps is rather high, for which reason we removed all POAs within less than 1 km of each other. In total, the 2'000 carpooling offers consist of 1'061 unique driver-defined stops (i.e., starts, destinations and stopovers), and 6'552 unique POAs.

After these, a multimodal network graph merging railway and carpooling networks was created. As introduced in Section III-D, standard routing algorithms such as Dijkstra's can be directly applied on the sub-graph of the time nodes in the merged network to provide routes from a start to an end.

C. Results

1) Comparison With Nearest Neighbor Algorithms: To compare the proposed method with the commonly employed nearest neighbor (NN) algorithm, which is a state-of-the-art method in merging different public transport networks [4], [21], we solved the NN problem for all driver-defined carpooling stops (i.e., starts, destinations and stopovers). Three different distances were used: 1 km, 2 km, and 5 km. Take 5 km as an example: we linked a driver-defined carpooling stop to its nearest railway station if their distance is smaller than 5 km. As using POAs is a new concept introduced in the proposed method, we did not apply NN to the carpooling POAs. Table I compares the number of links created for linking the railway and carpooling networks.

As can be seen, compared to the state-of-the-art method, the proposed method improves the connection of carpooling stops with railway stations. For example, for NN with

TABLE I
THE NUMBER OF CREATED CONNECTIONS BETWEEN PT AND CP STOPS

	Our method	NN (1 km)	NN (2 km)	NN (5 km)
# Links between driver-defined CP stops and railway stations	969	199	254	306
# Links between carpooling POAs and railway stations	5'683	-	-	-
Total	6'652	199	254	306

5 km, only 306 links were created for connecting carpooling stops and railway stations; while with the proposed method, 969 links were created. In addition, the proposed method was able to exploit new stops (via POAs) along a carpooling route, which created 5'683 further links to railway stations. In other words, the proposed method is capable of connecting carpooling offers to the railway network much better. It means that carpooling passengers can potentially transfer to many more railway stations, and vice versa.

2) Degree Centrality and PageRank Centrality: In total, about 5'572'593 transfer edges between time nodes were created, representing all possible transfers. Most of them are pure railway transfers (i.e., from one train to another, 4'812'926), and a smaller number for pure carpooling transfers (123'189). Additionally, 636'478 transfers exist with a modal change (either from carpooling to train, or vice versa). To investigate the effects of integrating carpooling into the railway network, we computed the degree centrality and PageRank centrality of each stop.

The degree centrality of a node counts all its connecting nodes. A high degree centrality means that the node links to many other nodes or many other nodes link to it. In this experiment, we particularly check whether integrating carpooling into the railway network improves the degree centrality of each railway station. The degree centrality of each station is computed by counting the number of its connecting railway stations, carpooling stops, and POAs. In general, the degree centrality scores of all railway stations in the merged multi-modal network have improved, meaning that integrating carpooling leads to the improvement of the number of connections at each railway station. We found that 8.1% of the stations are improved in their centrality by more than 10%, and 26% of the stations are improved by more than 3%. Figure 10 shows the heat map of the top 25% of railway stations with their degree centrality being improved by more than 5%. One can see that most of these stations lie in border as well as alpine regions. In these regions, the railway network (operated by SBB) is much less developed. This is particularly interesting because it indicates that carpooling can be a good addition to railway in less developed regions as well as for long distances where trains run less frequency.

The PageRank centrality is based on the PageRank algorithm, and computes the importance of nodes based on the number of incoming links, the link propensity of the linkers and the centrality of the linkers. In PageRank centrality, a node is considered as important if it is linked by important nodes or it is highly linked in general. In transport, the PageRank shows the hubs in a transport network. In Figure 11, stops with a very high PageRank score (i.e., $>$ 1.5 standard deviation) are shown as yellow dots, while the underlying heat map shows the density of these stops. It can be seen that the railway network (left) focuses on the two hubs Zurich and Lausanne. Merging carpooling with railway network using the proposed method (right) leads to an improvement of the railway network, namely important stops (i.e., stops with high PageRank score) spread over larger areas, which indicates better coverage. This also means that more stops are well connected, i.e., people do not have to travel to the big hubs (like Zurich main station and Lausanne main station) as many other stops now also serve many destinations. Meanwhile, it can be stated that, with an increasing number of important stops, the network is more robust, as it is less affected by a failure of one or more stops.

In summary, integrating carpooling into the railway network using the proposed method improves the number of connections at each railway station. Meanwhile, it also leads to a

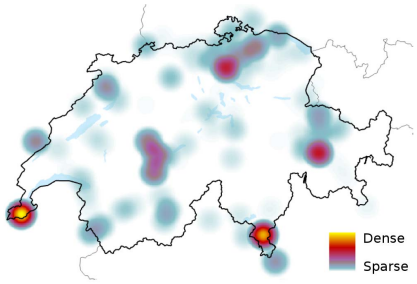


Fig. 10. Heat map of the top 25% of railway stations with their degree centrality being improved by more than 5%. These stations are mostly located in border or alpine areas. To view this figure in color, please refer to the online version of this article.

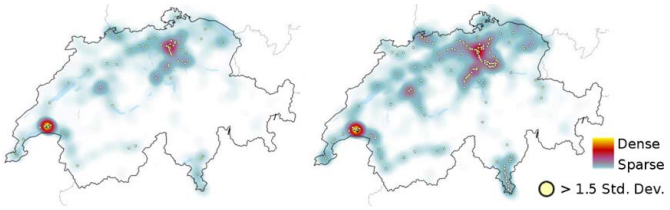


Fig. 11. The PageRank centrality measure, showing transport hubs: the density of stops with high PageRank score in railway network (left), and in the multimodal network resulting from merging the railway and carpooling network (right). To view this figure in color, please refer to the online version of this article.

Fig. 12. A multimodal trip from Bern Wankdorf to Olten, involving both carpooling and a train.

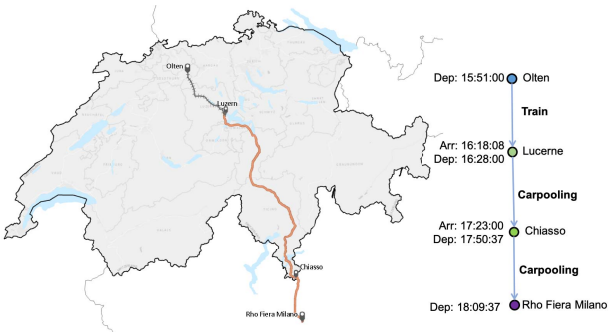


Fig. 13. A multimodal trip from Olten to Rho Fiera Milano, involving train and carpooling.

better spread of important stops (i.e., transport hubs) over the whole country, especially to the border and alpine areas.

3) Multimodal Route Planning: After merging the railway and the carpooling networks, we can then apply Dijkstra's on the sub-graph of the time nodes in the merged network to compute routes from an origin to a destination, using the method introduced in Section III-D. The resulting routes may be unimodal trips which only involve one mode, or multi-modal trips involving more than one mode. In the following, we particularly illustrate two examples with multimodal trips.

Assume that a soccer fan Adriana went to watch a game of her favorite team, FC Bern, at their Stadium in Bern Wankdorf. The game finishes at 16:00. Therefore, she is looking for a trip back to Olten at around 16:30. Querying the above merged network leads to a multimodal trip from Bern Wankdorf to Olten (Figure 12). The trip starts with carpooling from Bern Wankdorf to Egerkingen, and continues from Egerkingen by train. In total, it costs about 7.60 CHF, and takes 46 minutes. According to the Swiss Federal Railways (SBB) website (http://www.sbb.ch), the same trip with only trains, which involves one transfer at Bern main Station, costs 30 CHF and takes 47 minutes. In other words, the multimodal trip is less expensive in terms of travel time and fare. Please note that the resulting multimodal trip uses part of a carpooling offer, which is from Sierre (Switzerland) to Liège (Belgium). Both Bern Wankdorf and Egerkringen were actually not defined by the driver, but instead computed automatically via POAs, using the proposed method. Therefore, using the state-of-the-art method NNP would not be able to produce this multimodal trip.

Let's look at another example. Assume that a user living in Olten wants to visit the Expo in Milano at Rho Fiera Milano. Figure 13 shows a multimodal trip from Olten to Rho Fiera Milano, returned by applying the proposed method. The trip starts from Olten to Lucerne by train, from Lucerne to Chiasso by carpooling, and continues to the destination in another carpool. It takes about 220 minutes, and costs about 27.20 CHF. The trip returned by SBB takes 294 minutes, involves 2 changes, and costs more than 103 CHF. This example again shows that the multimodal trip combining trains and carpooling is less expensive in terms of travel time and fare. In this example, Lucerne is a stop defined by the driver of the first carpool. Chiasso and Rho Fiera Milano are again POA stops of the carpooling offers computed by the proposed method.

As can be seen from these examples, the proposed method is suitable to support multimodal route planning. It can not only make use of the driver-defined stops, but also explore potential stops that are along the carpooling route. The resulting trips seem to be cheaper, with shorter travel duration.

V. Discussion

As shown in the above evaluation with real-world data, the proposed method is suitable for merging public transport and carpooling networks to allow multimodal route planning, considering the fuzziness and flexibility of carpooling. Compared to the state-of-the-art approach (i.e., nearest neighbour method), the proposed method can better merge the static (i.e., public transport) and dynamic/fuzzy (i.e., carpooling) networks, while retaining the desired flexibility offered by the latter, and thus creates higher interconnectivity between the railway and carpooling networks. Meanwhile, the merged network enables multimodal route planning, which can provide users with a trip from an origin and a destination using different combinations of public transport and carpooling.

By comparing the proposed method with existing studies on multimodal route planning with carpooling [1], [6], we can see that existing studies always keep a specific carpooling offer in mind, and do not try to find the best carpooling offer from a set of possible choices for a passenger. Instead, the proposed method adds all valid carpooling offers to the public transport networks, and is able to find the best offer from this merged multimodal network.

In this study, we represented both public transport and carpooling networks using time-expanded models. On the one hand, this allows directly using standard routing algorithms such as Dijkstra's and A* as well as the associated speedup techniques to efficiently solve the route planning tasks. On the other hand, this approach leads to increased graph sizes, as each event (e.g., arrival and departure) is represented as a node, and edges between nodes are created. In short, this approach adds complexity to the network graph, rather than to the routing algorithm. While many graph databases, such as Neo4j and AllegroGraph, exist for efficiently managing large-scale graph data, it will be still very important to explore

techniques to reduce the size of the graph. Furthermore, techniques to remove expired carpooling data (e.g., carpooling trips that are already finished) should be developed, which will further reduce the size of the merged multimodal network graph.

The concept of drive-time areas (DTAs) was applied in this study to model the maximum detour allowed by carpooling drivers, and thus retain the flexibility offered by carpooling. It is important to note that computing DTAs is time-consuming, but can be done in advance. This will speed up the merging and linking process. Furthermore, this study computed DTAs around public transport stops instead of carpooling stops, which allows reusing these DTAs when linking/merging multiple carpooling offers. To further improve the performance of the proposed method, techniques to efficiently compute variable DTAs should be developed.

In the evaluation, we defined a minimal transfer time of 3 minutes and a maximum of 10 minutes. These values were derived by checking valid transfer results on the SBB website, considering that the time should be long enough to allow passengers to transfer but does not require them to wait for long. As shown in both algorithms, the transfer condition can be defined by the users. It can be expected that by setting a larger time range, more transfer edges between time nodes will be created, which will enable more transfers between railway and carpooling networks.

The evaluation merged the Swiss railway data and carpooling data from a large European carpooling platform for multimodal route planning. While railway timetables show the main characteristics of public transport timetables, it would be still interesting to see how well the proposed method works with other public transport networks, especially those covering smaller areas such as bus, subway and tram. We expect that similar results will be observed. It would also be interesting to see how well the proposed method works with frequency-based timetables, which define the trip frequency each route has. For instance, trams in Zurich service the same route on weekdays every 7 minutes during the day.

VI. Conclusion and Future Work

Recent years have seen much interest in carpooling, which aims to reduce the traffic congestions and travel costs. To further promote and facilitate the use of carpooling, there is a strong need to develop multimodal route planning systems that combine public transport and carpooling. Due to the fuzzy and flexible nature (e.g., no fixed stops, possibility to drive detours) of carpooling offers, existing state-of-the-art methods such as the nearest neighbor (NN) algorithm, which links nearby nodes in different networks, cannot be applied for merging public transport and carpooling networks. Based on the concepts of drive-time area (DTA) and point of action (POA), this article proposed a new method to merge public transport and carpooling networks to allow multimodal route planning, considering the fuzziness and flexibility of carpooling.

An evaluation with real-world datasets showed that the proposed method is feasible for merging public transport and carpooling networks into a multimodal routing network, while retaining the desired flexibility offered by carpooling. Compared to the state-of-the-art NN method, the proposed method can create higher interconnectivity between both networks. The evaluation also showed that by using standard Dijkstra's algorithm on the merged network, routes combining public transport and carpooling modes can be provided to users. The proposed method can be used to improve existing multimodal route planning applications by adding one additional mode–carpooling. This will enable the utilization of the full potential societal and environmental benefits brought by carpooling.

As a next step, we would like to extend the current method to support multi-criteria route planning, considering many other aspects, such as number of transfers, prices, and preferred modes. We also recommend further refining the method to reduce the size of the merged routing network, and to improve the performance of multimodal route planning. Experiments with public transport modes covering smaller areas and with frequency-based timetables can be also done to comprehensively evaluate the method. Furthermore, it is also useful to implement the method as a web application to provide multimodal route planning for the general public. To make multimodal route planning with carpooling work in real life, operational issues such as carpooling cost distribution policy, security, and availability of shared cars at transit stops on a return journey should be carefully considered.

Author Contributions

This paper is based on the Master Thesis of JK, supervised by HH and DB. HH and DB summarized and extended the work, shaping it into this article. RW and MR, as the head of the research units, oversaw this work, and provided comments to further improve the article.

References

[1] K. Aissat and S. Varone, "Carpooling as complement to multi-modal transportation," in *Proc. ICEIS*. Cham, Switzerland: Springer, 2015, pp. 236–255.

[2] U. M. Aïvodji, S. Gambs, M.-J. Huguet, and M.-O. Killijian, "Meeting points in ridesharing: A privacy-preserving approach," *Transp. Res. C, Emerg. Technol.*, vol. 72, pp. 239–253, Nov. 2016.

[3] S. Y. Amirkiaee and N. Evangelopoulos, "Why do people rideshare? An experimental study," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 55, pp. 9–24, May 2018.

[4] H. Bast *et al.*, "Route planning in transportation networks," in *Algorithm Engineering*. Berlin, Germany: Springer, 2016, pp. 19–80.

[5] H. Bast and S. Storandt, "Flow-based guidebook routing," in *Proc. Meeting Algorithm Eng. Exp.* Philadelphia, PA, USA: SIAM, 2014, pp. 155–165.

[6] A. Bit-Monnot, C. Artigues, M.-J. Huguet, and M.-O. Killijian, "Carpooling: The 2 synchronization points shortest paths problem," in *Proc. 13th Workshop Algorithmic Approaches Transp. Modeling, Optim., Syst. (ATMOS)*, 2013, p. 12.

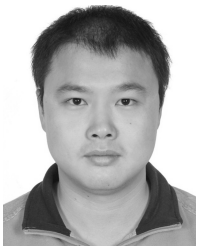
[7] G. S. Brodal and R. Jacob, "Time-dependent networks as models to achieve fast exact time-table queries," *Electron. Notes Theor. Comput. Sci.*, vol. 92, pp. 3–15, Feb. 2004.

[8] D. Bucher, D. Jonietz, and M. Raubal, "A heuristic for multi-modal route planning," in *Progress in Location-Based Services*. Cham, Switzerland: Springer, 2016, pp. 211–229.

[9] P. Czioska, A. Trifunović, S. Dennisen, and M. Sester, "Location- and time-dependent meeting point recommendations for shared interurban rides," *J. Location Based Services*, vol. 11, nos. 3–4, pp. 181–203, 2017.

[10] D. Delling, K. Giannakopoulou, D. Wagner, and C. Zaroliagis, "Timetable information updating in case of delays: Modeling issues," Tech. Rep. 133, 2008.

[11] J. Dibbelt, T. Pajor, and D. Wagner, "User-constrained multimodal route planning," *J. Exp. Algorithmics*, vol. 19, Feb. 2015, Art. no. 3.2.
[12] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
[13] D. Firmani, G. F. Italiano, L. Laura, and F. Santaroni, "Is timetabling routing always reliable for public transport," in *Proc. ATMOS Workshop*, vol. 33. Wadern, Germany: Schloss Dagstuhl—Leibniz-Zentrum fuer Informatik, 2013, pp. 15–26 .
[14] M. Furuhata, M. Dessouky, F. Ordóñez, M. E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transp. Res. B, Methodol.*, vol. 57, pp. 28–46, Nov. 2013.
[15] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter, "Exact routing in large road networks using contraction hierarchies," *Transp. Sci.*, vol. 46, no. 3, pp. 388–404, 2012.
[16] K. Goczyłla and J. Cielątkowski, "Optimal routing in a transportation network," *Eur. J. Oper. Res.*, vol. 87, no. 2, pp. 214–222, 1995.
[17] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory," in *Proc. SODA*. Philadelphia, PA, USA: SIAM, 2005, pp. 156–165.
[18] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.
[19] M. Hilger, E. Köhler, R. H. Möhring, and H. Schilling, "Fast point-to-point shortest path computations with arc-flags," in *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*, vol. 74. Providence, RI, USA: American Mathematical Society, 2009, pp. 41–72.
[20] M. Müller-Hannemann, F. Schulz, D. Wagner, and C. Zaroliagis, "Timetable information: Models and algorithms," in *Algorithmic Methods for Railway Optimization*. Berlin, Germany: Springer, 2007, pp. 67–90.
[21] T. Pajor, "Multi-modal route planning," M.S. thesis, Dept. Comput.-Sci., Univ. Karlsruhe, Karlsruhe, Germany, 2009.
[22] E. Pyrga, F. Schulz, D. Wagner, and C. Zaroliagis, "Efficient models for timetable information in public transportation systems," *J. Exp. Algorithmics*, vol. 12, Jun. 2008, Art. no. 2.4.
[23] G. Sierpiński, "Changes of the modal split of traffic in Europe," *Arch. Transport Syst. Telematics*, vol. 6, no. 1, pp. 45–48, 2013.



Haosheng Huang received the B.Sc. and M.Sc. degrees in computer science from South China Normal University, China, in 2004 and 2006, respectively, and the Ph.D. degree in GIScience from TU Wien, Austria, in 2013. He is currently a Research Group Leader and a Senior Lecturer with the GIScience Center, University of Zurich, Switzerland. His research interests include GIScience, location-based services, mobility, and transportation.

Dominik Bucher received the B.Sc. and M.Sc. degrees in electrical engineering and information technology from ETH Zürich, Switzerland, in 2011 and 2014, respectively, where he is currently pursuing the Ph.D. degree in geoinformation engineering. His research interests include location-based services, mobility and transportation, and big data analytics.

Julian Kissling received the M.Sc. degree in GIScience from the University of Zurich, Switzerland, in 2017. He is currently a Software Engineer with ESRI, Switzerland.

Robert Weibel received the M.Sc. and Ph.D. degrees in geography from the University of Zurich, Switzerland, in 1985 and 1989, respectively. He is currently a Full Professor of GIScience with the University of Zurich. His research interests include computational cartography, computational movement analysis, and language and space.

Martin Raubal received the M.Sc. and Ph.D. degrees in geoinformation from TU Wien, Austria, in 1998 and 2001, respectively. He is currently a Full Professor of geoinformation engineering with ETH Zurich, Switzerland. His research interests include location-based services, cognitive engineering for geospatial services, mobile eye-tracking, spatial cognition, and time geography.