Improving the On-Vehicle Experience of Passengers through SC-M*: A Scalable Multi-Passenger Multi-Criteria Mobility Planner

Rongye Shi, *Member, IEEE,* Peter Steenkiste, *Fellow, IEEE,* and Manuela M. Veloso, *Fellow, IEEE*

***Abstract*—The rapid growth in urban population poses significant challenges to moving city dwellers in a fast and convenient manner. This paper contributes to solving the challenges from the viewpoint of passengers by improving their on-vehicle experience. Specifically, we focus on the problem: Given an urban public transit network and a number of passengers, with some of them controllable and the rest uncontrollable, how can we plan for the controllable passengers to improve their experience in terms of their service preference? We formalize this problem as a multi-agent path planning (MAPP) problem with soft collisions, where multiple controllable passengers are allowed to share on-vehicle service resources with one another under certain constraints. We then propose a customized version of the SC-M* algorithm to efficiently solve the MAPP task for bus transit system in complex urban environments, where we have a large passenger size and multiple types of passengers requesting various types of service resources. We demonstrate the use of SC-M* in a case study of the bus transit system in Porto, Portugal. In the case study, we implement a data-driven on-vehicle experience simulator for the bus transit system, which simulates the passenger behaviors and on-vehicle resource dynamics, and evaluate the SC-M* on it. The experimental results show the advantages of the SC-M* in terms of path cost, collision-free constraint, and the scalability in run time and success rate.**

***Index Terms*—Public transit system, multi-agent systems, path planning, on-vehicle experience simulation, passenger behavior modeling, time-expanded graph.**

I. Introduction

MOBILITY is a central service a city needs to provide to the people. To satisfy the mobility needs given the rapid growth in city population, *Intelligent Transpiration System* (ITS) is proposed to utilize urban informatics and synergistic techniques to improve the transportation efficiency. *Intelligent public transit system* (IPTS) is one important member in the ITS family, which has attracted increasing interest in recent years.

Traditional research on IPTS focuses on using information about the state of the transportation networks for scheduling and dispatching mass transit [1]. Although a lot of progress has been made improving the IPTS in terms of infrastructure construction, information sharing and fleet scheduling, limited attention is paid to the people inside the vehicles, especially to the travel experience of the passengers that use the public transits. For example, in public transit systems, different passengers have different preferences, even necessities, in terms of public resources, such as seat availability or on-vehicle Wi-Fi. These factors seriously affect their satisfaction level while traveling.

We advocate that, to develop an efficient IPTS, *passenger-centered* research is a necessity, providing customized mobility plans directly to each passenger for a trip with a pleasant experience. This insight comes from the fact that an individual passenger can neither change the existing public transportation infrastructures nor the massive passenger and traffic flow, but rather, can choose an alternative well-designed strategy to accomplish the travel demand while making his or her experience pleasant. It is not easy to help passengers achieve this goal. On one hand, research to developing a model or a simulator that captures the road traffic, passenger behaviors and other factors relevant to the on-vehicle experience is a prerequisite to the goal but has not been considered previously. Specifically, the on-vehicle experience is affected by the surrounding passengers, traffic conditions, and facilities that support the public on-vehicle services. On the other hand, the mobility planner should consider the interactions among passengers as they call for the same service, which is challenging in the field of planning and scheduling, especially on a city scale.

In this paper, for the first time, we address the challenges from the viewpoint of passengers by considering the important role of passenger experience when planning for pleasant trips through the urban public transit systems. This paper contributes a passenger-centered planner for multiple passengers requesting multiple types of public services while limiting the level of interference among them. We assume that the total set of passengers is divided into two subsets – one with passengers who request planning service from the planner, called *client passengers*, and the other *non-client passengers* who do not. By viewing each client passenger as an agent, we formalize this problem as a *multi-agent path planning* (MAPP) problem, where multiple client passengers may interfere with one another in terms of some service resources. Grounded on this setting, we made the following contributions:

- Model the collisions among passengers as *soft collisions*,

Manuscript received month day, year; revised month day, year.
This work was supported by the FCT under the Carnegie Mellon–Portugal ERI S2MovingCities project (Grant CMUP-ERI/TIC/0010/2014).

R. Shi (corresponding author) is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213 USA (e-mail: rongyeshi@cmu.edu).
P. Steenkiste is with the Computer Science Department and the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213 USA (e-mail: prs@cs.cmu.edu).
M. M. Veloso is with the Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, 15213 USA (e-mail: mmv@cs.cmu.edu).

leading to a soft-collision-based (SC-based) MAPP problem, where a collision among client passengers is soft, quantified using a *collision score*, and different passengers have different scores according to their experiences. Based on this setting, the soft-collision M* (SC-M*), which is originally proposed by the authors [2], can be customized and applied to solve the SC-based MAPP;

- Customize the SC-M* to the bus transit system of Porto in the simulator. Specifically, we propose the *time-expanded expectation graph* to model the Porto bus network and introduce a technical detail to SC-M* (i.e., the *forward-the-tail-agent* approach) to handle the non-synchronization issue in the real system.
- Develop a data-driven on-vehicle experience simulator for the bus transit system of Porto, Portugal that simulates the passenger behaviors and the dynamics of certain on-vehicle resources (i.e., Wi-Fi quality and space availability). We then apply the customized SC-M* to generate the mobility plans based on the time-expanded expectation graph, which is constructed using the logs of simulations driven by the historical data, and test the plans' execution based on the *testing simulations* driven by the future unseen data;
- Evaluate the SC-M* method in terms of path cost, soft-collision constraint, and scalability in run time and success rate. The experimental results show the advantage in scalability and flexibility of SC-M* for handling complex environments with multiple types of passengers requesting multiple types of service resources.

The rest of the paper is organized as follows. Section II discusses related work and motivations. Section III gives a description of the MAPP definition and the SC-M* methodology. Section IV and V detail the implementation of the on-vehicle experience simulator and the customization of SC-M* to real systems. Section VI discusses experiments. Section VII concludes our work.

II. Related Work and Motivations

A. Related Work

The research on planning for passenger-level mobility in the context of public transit networks traditionally finds the shortest path for independent individuals, which is also known as *route planning* [3]. Traditional path planning on public transit networks is constrained on the scheduled networks, consisting of a set of stops, a set of routes, and a set of timed trips. Upon the scheduled networks, basic shortest path algorithms, such as the *Dijkstra's* algorithm [4], A* search [5] and bidirectional search [6], can apply.

When concerning passenger's experience, shortest path algorithms, which only consider distance as the optimization criterion, are not enough. For public transit networks, other criteria are just as important, leading to a multi-criteria planning problem. There are mainly two versions of multi-criteria search: The Pareto set search and the scalarization-based search. The first one is to find a *Pareto set* (i.e., a maximum set of incomparable paths [7], [8], [9]). This type of search treats each criterion as equally important, which incurs a high computational complexity and is not necessarily true in practice. An alternative is scalarization [10], using a linear combination of criteria as the optimization objective. This approach is to transform the multi-criteria search to a single-criterion search using weighted sum of criteria, which is more practical and computationally feasible.

As one case of multi-criteria planning, a multi-agent path planning (MAPP) finds a joint path with optimal objective for multiple agents conditioned on certain constraints. Approaches for MAPP can be folded into three main categories: coupled, decoupled, and intermediate. Coupled approaches search the *joint configuration space* of the multi-agent system, which is the *tensor product* of the free configuration spaces of all the individual agents. Decoupled approaches plan for each agent separately and then adjust the path to avoid collisions. Algorithms from this category are generally faster because the graph search and collision-avoidance adjustment are performed in low-dimensional spaces. However, optimality and completeness are not guaranteed. Intermediate approaches lie between coupled and decoupled ones because they dynamically couple agents and grow the search space during the planning. For a more review, we refer readers to [13]. Most MAPP approaches assume hard collisions, i.e., they do not allow any two agents to co-exist at the same node or edge. Our recent work [2] proposes and studies the soft-collision M* algorithm (SC-M*) to handle the MAPP under the soft collision context, which is the core algorithm in this paper.

B. Motivations for Multi-Passenger Mobility Planning

In mass transit systems, passengers have various preferences, even necessities, in terms of *public resources*, such as seat availability (necessary for seniors) or on-vehicle Wi-Fi supplies (preferred by video viewers and game players during the trip). When concerning multiple passengers' mobility on a city scale, it is not practical to plan for each individual passenger[1] independently because passengers may interfere with one another regarding public resources. Individually optimal paths can cause serious interference, leading to low-quality experiences. Interference between passengers is soft because it is possible that they do not call for the same resource when they are on the same public vehicle. Also, they are able to tolerate each other over a short time and distance. Intuitively, how likely a collision (intolerable interference) actually happens depends on 1) whether the resource supply is less than the demands and 2) how long the lack-of-supply condition lasts in terms of the time and distance that the passengers stay together.

Passengers can be viewed as agents moving through the urban public transit network. When we plan for all the agents, eliminating any hard collision is neither necessary nor feasible.

[1]In this paper, the concept of individual passenger is a generalized one, which can also refer to a group of *similar passengers* sharing the same preferences and the same Origin-Destination demands, and thus, the influence between them is considerable. To use a group of traffic subjects, such as vehicles, as a unit is a feasible way to model and control the mobility of the city [11].

Thus, we are more interested in another problem: *how can we plan for the client passengers (who request the planning service) to optimize the public resources received by them such that the probability of collision among them is bounded?*

This paper formalizes the problem as a MAPP problem under the soft-collision context, and we optimize the mobility of each client passenger over multiple public resources. To solve this practical problem efficiently, for the first time, we apply and customize SC-M*, which is originally introduced in [2], to plan for and accommodate a large number of client passengers simultaneously. Our work contributes to showing that a SC-based MAPP solver can be extended to the public transportation domain and can accommodate the passengers' mobility demands on a city scale. SC-M* is expected to help with the congestion and resource management issues in the context of rising population and mobility needs in cities.

III. Soft-Collision M* (SC-M*) for Passenger Mobility Planning

This section defines the traditional MAPP problem and its state-of-the-art solver, M* [12], [13], and proposes soft-collision M* planning approach.

A. MAPP Problem Definition and M*

In MAPP problem, we have m agents indexed by the set $I = \{1, \ldots, m\}$. Let the free configuration space of agent j be represented by the directed graph $G^j = \{V^j, E^j\}$. Each agent has its own free configuration space. The joint configuration space, which describes the set of all possible states of the multi-agent system, is represented by the *tensor product* of the graphs of all individual agents: $G = G^1 \otimes \cdots \otimes G^m$. G consists of a *joint vertex* set V and a *joint edge* set E. As an example, in a 2-D joint configuration space given by the agents j and k, the two 2-D joint vertexes $v_p = (v_p^j, v_p^k)$ and $v_q = (v_q^j, v_q^k)$ is connected by the joint edge (e_{pq}^j, e_{pq}^k). Note that $v_p^j \in V^j$ and $e_{pq}^j \in E^j$. Let $\pi^j(v_p^j, v_q^j)$ denote a sequence of joint vertexes, called a *path* in G^j from v_p^j to v_q^j. The cost of a path $\pi(v_p, v_q)$ in G is defined as

$$g(\pi(v_p, v_q)) = \sum_{j=1}^{m} g(\pi^j(v_p^j, v_q^j)), \tag{1}$$

where $g(\pi)$ is the sum of all edge costs involved in the joint path π.

The problem of MAPP is to find a collision-free path, which is optimal with respect to minimal cost, from the source configuration $v_s = v_s^1 \otimes \cdots \otimes v_s^m$ to the destination configuration $v_d = v_d^1 \otimes \cdots \otimes v_d^m$. To determine the collision between agents, a collision function $\psi(v_p)$ is defined to return the set of conflicting agents at v_p.

Most fundamental MAPP approaches usually default hard collisions that no intersection is allowed between every two agents in terms of the occupation of any *resource*, such as space. This implies that the capacity of each resource can support only one agent at a time (i.e., a collision happens immediately once agents intersect at any resource). Suppose we have a set of resources $A = \{A_1, \ldots, A_L\}$ requested by each agent in the multi-agent system, where A_k is the set of resource of Type k on all edges and vertexes. The traditional hard-collision-free constrained MAPP problem is formulated as follows:

$$\begin{aligned} &\min_{\pi} g\left(\pi\left(v_s, v_d\right)\right) \\ s.t.\ & \\ &\bigcup_{\forall i \neq j \in I} \left(A_k(v_p^i) \cap A_k(v_p^j)\right) = \emptyset,\ \forall A_k \in A,\ \forall v_p \in \pi, \\ &\bigcup_{\forall i \neq j \in I} \left(A_k(e_{pq}^i) \cap A_k(e_{pq}^j)\right) = \emptyset,\ \forall A_k \in A,\ \forall e_{pq} \in E(\pi), \end{aligned} \tag{2}$$

where $A_k(v_p^j)$ and $A_k(e_{pq}^j)$ denotes the subset of resource A_k occupied by the agent j at the joint vertex v_p and at the joint edge e_{pq}, respectively; $E(\pi)$ is the set of joint edges involved in path π. This definition can be adjusted to consider edge collision or vertex collision only according to practice.

One state-of-the-art solver to this problem is M* [12], [13], which uses the *sub-dimensional expansion* strategy to dynamically increase the dimensionality of the search space in regions featuring some agent collisions. M* enables a relatively cheaper graph search under the strict hard-collision-free constraint.

The M* assumes a hard-collision-free constraint, which does not apply to many real-world applications. Our recent work on SC-M* algorithm in [2] generalizes the M* to soft-collision context, which can be applied to the multi-passenger mobility planning.

B. Soft-Collision Constrained MAPP

In this subsection, we formalize the multi-passenger mobility planning problem as a MAPP problem under the soft-collision context (i.e., SC-based MAPP) and apply the SC-M* to efficiently solve it.

To comply to the real-world scenario, we introduce the soft-collision property to the behavioral model of a client passenger. In our model, each client passenger is defined to have the following properties: 1) a collision among client passengers is "soft", quantified using some collision scores and 2) different client passengers have different collision scores, according to their individual experiences through a path. The influence from the non-client passengers to client passengers is encoded as a part of the edge cost, which is not involved in the collision among client ones. To obtain the properties, we introduce to each client passenger an attribute called *resource experience* and use the resource experience to calculate the *collision score*.

1) Definition 1 (Resource Experience): We define *resource experience* to quantify the dissatisfying experience per resource about which a client passenger cares.

Let

- $\pi = \pi\left(v_s, v_b\right)$ be a path from the source v_s to some v_b;
- $v_q = \pi\left(v_p\right)$ be the immediate successor of v_p along the path π;
- $A_k(e_{pq}^j)$ be the capacity (amount) of the subset of the resource A_k on the edge e_{pq}^j, given by the graph model;

- $A_k^j(e_{pq}^j)$ be the amount of the subset of the resource A_k *actually allocated* to the client passenger j on the edge e_{pq}^j, called the *allocated resource value*.

The *resource experience* is then defined as the *dissatisfying experience* of client passenger j on resource A_k along the path π^j:

$$D\left(\pi^j, A_k\right) = \sum_{v_p \in \pi/v_b} \mathbb{1}\left(A_k(e_{pq}^j) \geq \varepsilon_k \wedge A_k^j(e_{pq}^j) < \varepsilon_k\right) \cdot g(e_{pq}^j), \tag{3}$$

where $\mathbb{1}(\cdot)$ is the indicator function, whose value is one if the logical condition is true, else zero; $\varepsilon_k \in \varepsilon = \{\varepsilon_1, \ldots, \varepsilon_L\}$ is the *satisfying value* regarding the resource A_k, which is a positive real value; $g(e_{pq}^j)$ is the edge cost regarding travel time, distance, negative influence from non-client passengers given by the graph model; $A_k^j(e_{pq}^j)$ is formulated as:

$$A_k^j(e_{pq}^j) = \frac{A_k(e_{pq}^j)}{\sum_{i \in I} \mathbb{1}\left(e_{pq}^i = e_{pq}^j\right)}. \tag{4}$$

The numerator of Eq. (4) is the available resource k on edge e_{pq}^j. The denominator denotes the number of agents moving to the same edge e_{pq}^j. This is judged by the logic condition, $e_{pq}^i = e_{pq}^j$, i.e., whether or not the id of edge occupied by agent i is the same to the id of edge occupied by agent j. So the equation defines the actual allocation of resource allocated to agent j. According to Eq. (4), $A_k^j(e_{pq}^j) = A_k(e_{pq}^j)$ if and only if no other client passengers are physically moving along with client passenger j on the edge e_{pq}^j. The allocated resource value $A_k^j(e_{pq}^j)$ quantifies the level of interference incurred by other client passengers when they physically move together.

The definition of resource experience in Eq. (3) actually defines a property of a client passenger: Only the situation, in which the resource allocated to a client passenger is dissatisfying because of the co-existence of other client passengers, will contribute to the dissatisfying experience of that client passenger. Furthermore, each dissatisfying condition is weighted by the edge cost $g(e_{pq}^j)$. In this way, we can quantify how serious such a dissatisfying condition is in terms of travel time, distance and crowdedness level, etc, which are given by $g(e_{pq}^j)$ in the graph. According to different application purpose, Eq. (3) is open to modification for a more realistic model. For example, we can multiply $g(e_{pq}^j)$ by a function of $A_k^j(e_{pq}^j)$ to reflect the level of dissatisfying condition on the edge. Also, different passengers may have different ϵ_k for a given resource in real world.

2) Definition 2 (Collision Score): We use the resource experience results from Definition 1 to calculate the *collision scores*. This is defined from the view point of collision probability, that must be constrained under some threshold.

Let

- Col_j be the event that client passenger j announces a collision (i.e., when client passenger j calls for one of the resources, the allocated resource is less than satisfying);
- $D^j = \{D_1^j, \ldots, D_L^j\}$, where $D_k^j = D\left(\pi^j, A_k\right)$, be the set of dissatisfying experiences of client passenger j along path π^j on the resource A_k;
- $f_k \in f = \{f_1, \ldots, f_L\}$ be a customized cumulative distribution function (CDF) defined on $[0, +\infty)$, mapping the resource experience D to a probability of collision on the resource A_k.

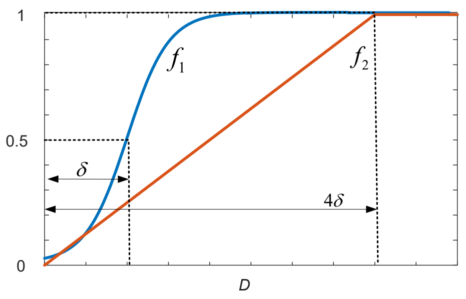


Fig. 1. Example designs of CDFs, mapping the resource experience D of a client passenger to a collision probability on certain resource. f_1: sigmoid-based CDF for important (sensitive) resources. f_2: linear CDF for trivial (insensitive) resource. δ: offset parameter adjusting the tolerance level.

The *collision score* of the client passenger j is defined as the probability of the occurrence of a collision, i.e., the client passenger j can no longer tolerate the experience and complain on *at least* one of the resources given the resource experience D^j:

$$P\left(Col_j \middle| D^j\right) = 1 - \prod_{k \in \{1,\ldots,L\}} \left(1 - f_k(D_k^j)\right). \tag{5}$$

Note that $P\left(Col_j \middle| D^j\right)$ calculates the *complement* of the success probability – the joint probability of being collision-free on all resources.

Fig. 1 shows two example designs of f: $f_1(D) = sigmoid(D - \delta)$, with a discontinuity point $f_1(0) = 0$, is a sigmoid-based CDF function, featuring a surge in the collision score (i.e., the derivative is bump-shaped). This function is suitable to important resources that are sensitive to the client passenger; $f_2(D) = min(1, D/(4\delta))$ is a linear CDF with a shallow slope (i.e., the derivative is a constant). This function can apply to trivial resources that are not very sensitive to the client passenger but still accumulate to contribute to the collision score. We use the *offset parameter* δ to adjust the *tolerance level* of the dissatisfying experience. With larger δ, a client passenger will tolerate a more dissatisfying experience before announcing a collision.

Although the definition of the collision score can be customized according to different practices, the probabilistic definition of collision score introduced here is a general one: Different types of resources may have different value ranges, and Eq. (5) standardizes the resource ranges, mapping them to a value between $[0, 1]$ and enabling an efficient integration of different types of resources to the framework.

3) Definition 3 (Soft-Collision Function $\widetilde{\psi}$): According to the collision scores from Definition 2, we want to pick out the above-threshold client passengers and place them into the soft-collision set via the *soft-collision function*.

Given a path $\pi = \pi(v_s, v_b)$ and corresponding resource experience D^j for the client passenger j, the *soft-collision* *function* of client passenger j is

$$\widetilde{\psi}^j(v_b) = \begin{cases} \{j\}, & for\ P\left(Col_j \middle| D^j\right) \geq T \\ \emptyset, & otherwise \end{cases}, \quad (6)$$

where T is the *threshold of collision*. The definition of the *global soft-collision function* is then defined as

$$\widetilde{\psi}(v_b) = \bigcup_{j \in I} \widetilde{\psi}^j(v_b). \quad (7)$$

Based on Definition 3, we can formally model the multi-passenger mobility planning as a SC-based MAPP problem:

$$\begin{aligned} &\min_{\pi} g(\pi(v_s, v_d)) \\ s.t. & \qquad\qquad\qquad\qquad (8) \\ &\widetilde{\psi}(v_p) = \emptyset, \quad \forall v_p \in \pi. \end{aligned}$$

The interpretation of Eq. (8) is as follows: We want to move m client passengers from the source configuration $v_s = v_s^1 \otimes \cdots \otimes v_s^m$ to the destination configuration $v_d = v_d^1 \otimes \cdots \otimes v_d^m$ with minimal cost, constrained on that the soft-collision set is empty (i.e., the soft-collision constraint).

C. Illustrating the Principle of M* and SC-M*

SC-M* is a general solver for the SC-based MAPP problem in Eq. (8) and a description of the algorithm is provided in Appendix A. For more details, we refer readers to reference [2]. In this section, we give an example to illustrate the principle of M* and SC-M* and compare these two algorithms.

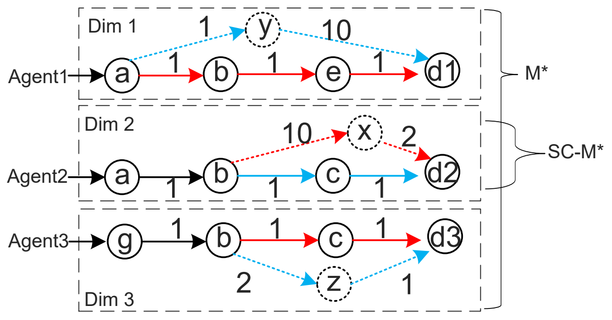


Fig. 2. Example of applying M* and SC-M* to MAPP problem with three agents. Solid vertexes and arrows denote individually optimal paths. Dotted vertexes and arrows are the nodes and edges added during expansion. Blue and red arrows are specific edges picked by M* and SC-M*, respectively. Black arrows are common edges picked by both algorithms. Traditional M* has to expand all three dimensions due to hard-collision constraints while SC-M* considers soft-collision constraints and only expands dimension 2.

In Fig. 2, we show a three-agent MAPP planning problem. Agents 1, 2, and 3 attempt to move from source vertexes a, a, and g to destination vertexes $d1$, $d2$, and $d3$, respectively. Assume that only edge collisions are considered and no collisions occur in vertexes. The individually optimal paths (with shortest distance) are $a \rightarrow b \rightarrow e \rightarrow d1$ with distance 3 for Agent1, $a \rightarrow b \rightarrow c \rightarrow d2$ with distance 3 for Agent2 and $g \rightarrow b \rightarrow c \rightarrow d3$ with distance 3 for Agent3. The total cost of the joint individually optimal path is 9.

M* solves this MAPP considering hard-collision constraints. It starts searching in the initial *search graph* consisting of individually optimal paths and increases the space gradually when collision occurs. At the beginning, M* explores the joint vertexes along the individually optimal path. When moving from the initial joint vertex (a, a, g) to (b, b, b), the algorithm detects the collision on edge $a \rightarrow b$ between Agent1 and Agent2, which triggers the expansion of dimension1 and dimension2 according to *sub-dimensional expansion* strategy. Then, vertexes y and x are included into the search graph. Next, M* will reroute Agent1 to vertex y to avoid collision and let Agent2 keep on its individually optimal path to move to vertex b and then to c. Following this moving, the collision on edge $b \rightarrow c$ between Agent2 and Agent3 is detected, triggering the sub-dimensional expansion of dimension3 to include vertex z to the search graph. In next step, M* reroutes Agent3 to vertex z to get to destination $d3$ (because edge $b \rightarrow x$ is too expensive for Agent2) and let Agent2 finish its individually optimal path. Finally, the collision-free joint path (with black and blue edges) is found including $a \rightarrow y \rightarrow d1$ for Agent1, $a \rightarrow b \rightarrow c \rightarrow d2$ for Agent2 and $g \rightarrow b \rightarrow z \rightarrow d3$ for Agent3. The total cost is $11 + 3 + 4 = 18$.

SC-M* solves this MAPP using soft-collision constraints. Assume that the collision threshold T in Eq (6) is properly set and agents can tolerate an accumulated collision distance of 1.5. In this way, following the individually optimal paths, only Agent2 experiences a collision distance of 2 (edges $a \rightarrow b$ and $b \rightarrow c$). Therefore, only dimension2 is expanded and SC-M* includes vertex x into search graph. As a result, Agent2 is rerouted through vertex x and other two agents stick to their individually optimal paths. Through this joint path (with black and red edges) , no agents experience >1.5 collision distance and the soft-collision constraint is satisfied. The total cost is $3 + 13 + 3 = 19$.

From this example, we demonstrate that by introducing a tunable soft-collision constraint, SC-M* reduces the search space and thus the execution time. However, it is sub-optimal. The reason is that the soft-collision set in Eq. (6) and Eq. (7) ignores the soft-collision conflicts in the paths of Agent1 and Agent3 in Fig. 2. Thus, the dimensions of soft-conflicting agents are not fully searched. Though SC-M* is sub-optimal, it is flexible to tune the built-in parameters to control the level of sub-optimalilty.

D. Properties and Advantages of SC-M*

We recently conducted research on SC-M* [2], in which we have obtained the following properties. SC-M* is:

- Theoretically complete: Given a finite graph, it guarantees (in finite time) to either return a solution or to determine that no solution exists;
- Theoretically suboptimal: It may not guarantee to always return the optimal path. As discussed in Section III-C, this is because the soft-collision set in (6) and (7) ignores the soft-conflicting agents in the path in exchange for improved scalability;
- Empirically flexible in performance by tuning the parameters T and δ: We theoretically show that SC-M* can make a transition from a decoupled individually optimal planner (e.g., individually optimal A*) at $T = 1$ to a standard hard-collision-free constrained M* at $T = 0$. And

empirically, we demonstrate the tunable performance via experiments in a small-scale (20×20) grid environment.

The work in [2] shows that SC-M* has significant overall advantages in terms of cost-optimality (i.e., path cost) and scalability (i.e., run-time and success rate under a large agent size) over other alternative SC-based MAPP solvers, such as SC-A* (using the standard A* to solve the SC-based MAPP) and SC-CBS (combining the soft-collision constraint with the state-of-the-art conflict-based-search algorithm [14]). These advantages make it a promising candidate for solving the city-scale passenger mobility planning problem.

IV. On-Vehicle Experience Simulator

This paper focuses on improving the on-vehicle experience of passengers based on SC-M*. To evaluate the performance of the algorithm, we develop a passenger-level simulator that combines: the 1) urban traffic simulation, 2) behavioral model of passengers (for both the client and non-client), and 3) infrastructures/facilities that support certain public resources. This section proposes an on-vehicle passenger experience simulator for a case study of the bus transit system in Porto, Portugal. The on-vehicle passenger experience simulator is based on the widely-used traffic simulator SUMO[2] and extended from the *joint-traffic-passenger-modeling-simulation* framework we developed in [15].

A. Porto Bus Transit System Establishment

The Porto bus transit system is established in SUMO. The Porto bus service operator STCP has a service website[3], where detailed information about routes, station geographical locations, and a timetable of bus trips are provided. As shown in Fig. 3, using the STCP bus service information and urban map resource (OpenStreetMap, etc.), we build the bus transit system as well as the urban road network of the selected central city area of Porto (E: -8.559543°, W: -8.66191°, S: 41.136044°, N: 41.185110°). The imported bus network contains 136 routes, 855 bus stops, and 5723 bus trips in a normal weekday. The simulation tests confirmed that the bus performance matches the actual Porto bus transit system: Each bus departs at the scheduled time, runs along its designated route, and pulls up at the designated stops correctly.

B. Real AFC Bus Passenger Data Processing

For passenger-level simulation, we should configure the non-client passenger flow based on real data. The Porto Automated Fare Collection (AFC) data is a valuable resource that is directly suitable for such a configuration.

The AFC data contains bus passenger transaction records that occurred in January, April, and May of 2010. The data was collected by the AFC system installed in buses operated by STCP in Porto. The AFC system called "Andante" is an entry-only system. Each transaction record contains several attributes of which we are interested in the following: the 1) ID; 2) transaction timestamp; 3) bus stop where the transaction

[2]http://sumo.sourceforge.net

[3]https://www.stcp.pt

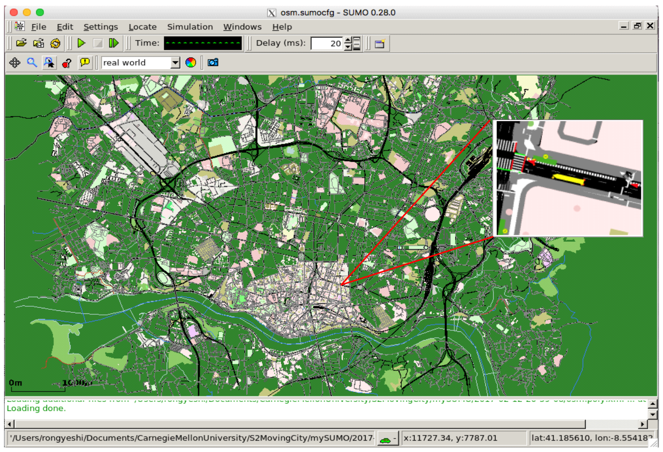

osm.sumocfg - SUMO 0.28.0
File Edit Settings Locate Simulation Windows Help
Time:
Delay (ms): 20
real world
0m
1000m
Loading done.
/Users/rongyeshi/Documents/CarnegieMellonUniversity/S2MovingCity/mySUMO/2017
x:11727.34, y:7787.01
lat:41.185610, lon:-8.55418

Fig. 3. Virtual traffic and bus transit network in SUMO.

occurred; 4) route; and 5) route direction. There are 2,374 bus stops in a broad area and 132 bus routes (4 late-night routes were missing in 2010). We select transaction records on Wednesdays of the three months, totaling 12 Wednesdays with 2,422,079 transactions. Those Wednesdays are normal weekdays and any local holidays are removed. We infer the missing destination information from the raw passenger data by applying the *semi-supervised self-training method* [16] and >90% of the Origin-Destination (O-D) demands are retrieved with high inference confidence. The post-processed O-D demands serve as the set-up configuration of the non-client passenger flow on a weekday.

C. Modeling the On-Vehicle WiFi Service

The on-vehicle services are termed *public resources* because they are limited in amount and used by each individual passenger to improve his or her experience. This paper considers two public resources: on-vehicle Wi-Fi service and space availability.

The modeling of the on-vehicle Wi-Fi service is driven by the following concerns: In a city, buses can connect to any wireless backhaul channel to provide Internet connectivity to passengers, including cellular towers and city hotspots. The connection based on cellular networks is expensive because they are owned and operated by telecommunication companies (e.g., T-Mobile and AT&T). Those companies have to build and maintain the infrastructures, auction against peers to gain a spectrum license from the Federal Communications Commission (FCC) and support many other costs. In contrast, urban hotspot-based infrastructures are public facilities, and the wireless connection between a bus and a hotspot is free. The cost, which includes Ethernet fees, hotspot maintenance, etc., is much cheaper, and is affordable to urban governance. In this sense, in the paper, the on-vehicle free Wi-Fi connection is assumed to be supported by the city hotspots deployed across the city. Passengers are assumed to seek the bus Wi-Fi service first before using their own cellular data packages.

We construct the on-vehicle Wi-Fi service map based on the urban hotspot location data and the free-space-path-loss model. Each bus will connect to the closest hotspot automatically,

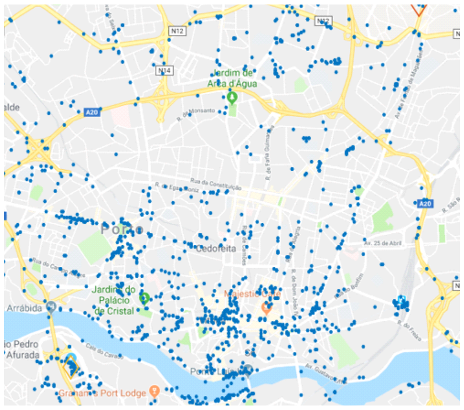


Fig. 4. Map of Wi-Fi hotspot locations in Porto.

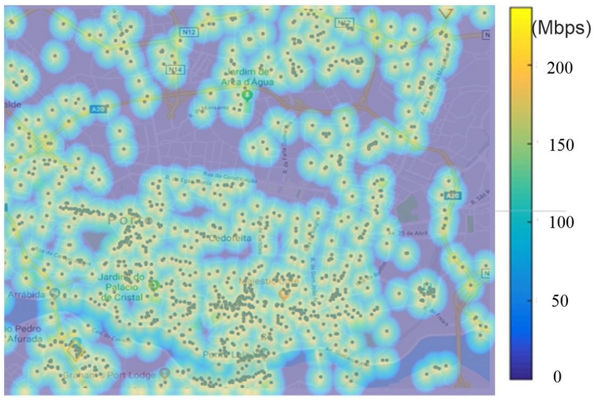


Fig. 5. Map of the on-vehicle Wi-Fi capacity in megabit per second (Mbps).

ignoring the handoff delay between Wi-Fi hotspots. Without considering the surrounding obstacles, multipath interference, and other factors affecting the wireless signals, the capacity of a link in bits per second (bps) follows the *Shannon's* Theory [17]:

$$C = B \cdot log_2 \left(1 + \frac{S}{N}\right), \tag{9}$$

where B = 40 MHz for 801.11ac wireless networking protocol [18], the signal-to-noise-ratio (SNR) = 40 dB for a good outdoor environment 2.5 meters from the hotspot [19]. The SNR is linear to $\frac{1}{distance^2}$ based on the free-space-path-loss model.

We use the hotspot location data to form the map of the on-vehicle Wi-Fi service capacity. The real Wi-Fi hotspot location data for Porto is presented in Fig. 4, where each blue point is a hotspot location. Fig. 5 is the map of the on-vehicle Wi-Fi service capacity using Eq. (9), where we assume that the capacity is not affected by whether two or more buses are connecting to the same hotspot. The capacity decays with respect to the distance from the hotspot and the areas with higher hotspot distribution density will provide higher and more robust quality of Wi-Fi service.

D. Bus Passenger Behavior Modeling + SUMO Simulation

Given a bus passenger mobility plan generated by some planner, we develop a *joint simulator* to simulate how the passenger moves through the bus transit network and interacts with buses and traffic to ultimately reach the destination. One of the core functions of the simulator is the bus passenger behavioral model which runs jointly with SUMO via a monitor-control algorithm. The algorithm is based on TraCI, which is a traffic control interface of SUMO. Specifically, the algorithm is to update the states of buses, simulate passenger behaviors and record important moments (e.g., bus arrival time, etc.). We refer readers to [15] for detailed descriptions, and here we highlight some points about the joint simulator:

- The interactions between buses and passengers take place at the bus stop, and the bus dwell time is affected by the number of boarding and alighting passengers;
- The interactions between buses and other road traffic are simulated by SUMO, where the travel time of a bus on the road varies according to the traffic conditions and the dwell time at stops;
- The simulation log stores detailed passenger experience information (wating/boarding/alighting time at the stop, Wi-Fi quality on bus, cost, etc.) and bus state information (bus arrival time at each stop, on/off passengers' IDs at each stop, stop dwell time, passenger volume, etc.).

Using the joint simulator and the on-vehicle Wi-Fi service model, we record the simulated capacity of the Wi-Fi access rate (Mbps) on each bus trip and add the records to the simulation output log. Specifically, at every 10 seconds in the simulation, we associate the geographical location of each bus trip to the closest Wi-Fi hotspot and calculate the Wi-Fi access rate using Eq. (9). This capacity becomes an attribute of the bus state record. The average Wi-Fi access rate allocated to each passenger becomes an attribute of the passenger's experience record.

Fig. 6 illustrates how the average on-vehicle Wi-Fi service allocated to each passenger is obtained from the simulation. We record the Wi-Fi capacity along Trip 801-1, which departed at 6 am on the Wednesday of May 5, 2010; collect the passenger volume; and take the ratio of the two. This trip starts from the central city area, and the Wi-Fi quality is high. However, at around 2000 meters along the trip, the passenger volume increases and the average Wi-Fi quality declines to around 25 Mbps. After 4000 meters, the bus moves into poor Wi-Fi area while the passenger volume is large, and thus, the Wi-Fi service is low from the viewpoint of each passenger.

Because the passenger volume will change from day to day, the average Wi-Fi profile is unique on a daily basis. However, instead of complete randomness, human trajectories show a high degree of temporal and spatial regularity [20], and thus, the Wi-Fi quality pattern along Trip 801-1 may still apply to a future normal Wednesday. Based on this dynamic pattern of the Wi-Fi quality on Trip 801-1, a planner could encourage the passenger who has a high Wi-Fi preference to take the segment of the trip before 2000 meters and avoid traveling through the segment after 4000 meters.

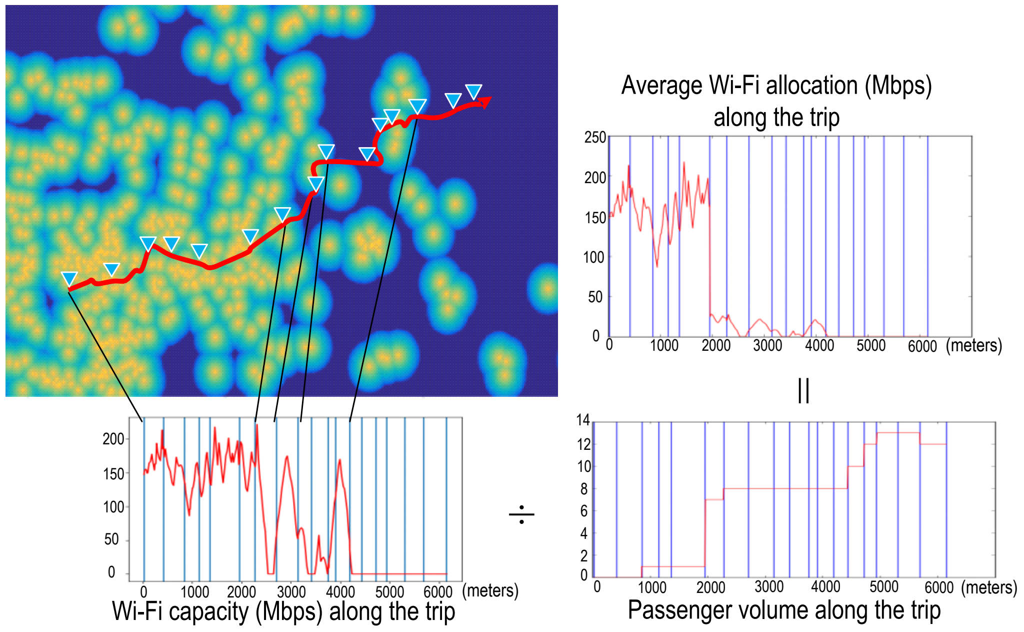


Fig. 6. Example of the average Wi-Fi quality profile along Trip 801-1 on the Wednesday of May 5, 2010.

As will be discussed later in the next section, the simulation log will be used to construct the *expectation graph*. The expectation graph is a model of the public transit network, which is a prerequisite for the planning. Based on model, the SC-M* is applied. We run the simulation for all the four Wednesdays in May of 2010, using the simulation logs of the first three Wednesdays (May 5, 12, and 19) as the historical data for constructing the expectation graph, based on which we apply the SC-M* to generate mobility plans. The generated plans are tested by executing them in the *testing simulation* of the unseen Wednesday on May 26, 2010.

V. Customizing SC-M* to Real-World Bus Transit Systems

Before applying the SC-M* to generate mobility plans, we need to 1) pre-process the simulation-log data to build the *expectation graph* of the bus transit network and 2) customize certain technical details of the SC-M* to the environment of bus transit systems because the real systems differ from the traditional MAPP environments. To this end, this section introduces the *time-expanded expectation graph* model and the *forward-the-tail-agent* approach.

A. Expectation Graph of Bus Transit Network

One key point to planning for improved experience with soft-collision constraints is to construct a model of the urban public transit network with necessary information. In addition to the static infrastructure information about the bus routes and stop locations, dynamic information obtained from the simulation (e.g., the timing of the bus trips, Wi-Fi quality, and passenger volume) is needed. We use a *time-expanded model* to model the static and dynamic information about the public transit system in a weekday, and the average of the time-expanded models of several weekdays produces the *expectation graph*, upon which we apply the SC-M* during the planning phase.

1) Time-Expanded Model (An Off-Line Model for One Weekday): The public transit network is based on a *timetable*, consisting of a set of stops, routes, and trips. A trip of a route corresponds to a vehicle visiting a list of stops along the route at specific times of day [3]. The main approach for constructing a graph of the public transit network from the timetable is *time-expanded* modeling [21]. The time-expanded model creates a vertex for every event (e.g., a bus-arrival event at a stop) in the timetable and uses edges to connect subsequent events in the direction of time flow. In our research, we replace the traditionally used timetable events with the bus-arrival events in the simulation log to build the time-expanded model.

As illustrated in Fig. 7, we consider the bus transit network as a *directed graph* G (note that the graph G here is the same as G^j in Section III for simplification). Each vertex corresponds to an event in the simulation log. An event is defined as a bus arriving at a stop during a bus trip of a certain route. In Fig. 7(a), for example, the blue route (with blue *solid* arrows) has three bus trips, and $v_{2,1}$ corresponds to a bus-arrival event at the bus stop V_1 during the blue bus trip 2 (i.e., the second row of blue vertexes in the figure) at the time $t_{2,1}$. V_1 is defined as the set of the arrival events occurring at the same stop (i.e., Stop 1). The arrow edge connects two subsequent bus-arrival events along a bus trip. The color of the edge indicates the route the bus trip belongs to. For example, the blue edge from $v_{1,1}$ to $v_{1,2}$ indicates that, during the blue bus trip 1, the arrival event at the Stop 1 (associated with the set V_1) at time $t_{1,1}$ precedes the arrival event at the successor Stop 2 (associated with the set V_2) at time $t_{1,2}$. In the graph, passengers can transfer to another bus trip of the same route. The transfer connections are represented

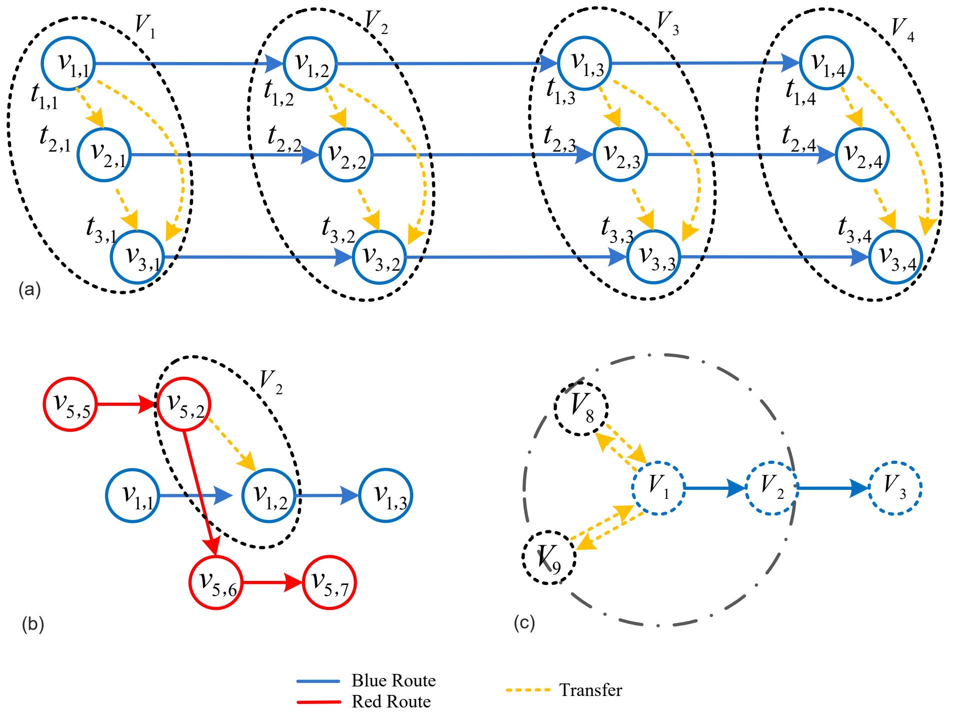


Fig. 7. Time-expanded graph for the bus transit network of one weekday. (a) shows the time expanded graph for the blue route and (b) shows how passengers can transfer between two different routes. (c) shows the transfer links between walking-reachable stops. All dashed vertexes/ovals (e.g., V_1 in [a] and V_8 in [c]) indicate the set of events occurring at the same stop.

by the yellow *dotted* arrow edges. The events at the same stop are fully connected, and any pair of subsequent events has a yellow dotted connection.

For the time-expanded graph, Fig. 7(b) illustrates that the red route (with red *dashed* arrows) and the blue route (with blue *solid* arrows) share the same stop (i.e., Stop 2 associated with the set V_2), and that passengers can transfer from the red route to the blue route at this stop ($t_{5,2} < t_{1,2}$ is a prerequisite for creating such a transfer edge). In addition to transfer edges at the same bus stop, walking transfer edges are introduced into the graph, as illustrated in Fig. 7(c). Stops within a short geographical distance (e.g., 640 meters) are considered as reachable via walking, and passengers would be willing to walk a few more meters to transfer routes there. The transfer edge between bus stops (dashed circles/ovals) in Fig. 7(c) indicates the full connection between any pair of subsequent events occurring at the two stops.

2) Expectation Graph (An Off-Line Average Model of Several Weekdays): After obtaining the simulation log of a weekday from the on-vehicle experience simulator , we extract the simulated travel time, average Wi-Fi quality, and passenger volume as each edge's attributes. The yellow dotted transfer edges are outside of the vehicles; thus, their Wi-Fi and passenger volume attributes are set to zero. We construct the *time-expanded models* of three simulated Wednesdays in May (i.e., May 5, 12, and 19). By averaging the attributes over the three models, we obtain the *expectation graph* of the bus transit network. SC-M* uses the graph to plan for the client passengers in the planning phase. Using this graph, passengers who reach one of the events at the destination stop fulfill their individual travel plan.

One disadvantage of the time-expanded model is that the resulting graph is quite large. To handle this issue, the expectation graph only contains bus trips of which departure times are within a 30-minute time window. For example, our experiments use a time window between 14:00 and 14:30.

B. Forward-the-Tail-Agent Approach

In contrast to the traditional MAPP settings, in which the agents are assumed to reach certain vertexes at a time step (i.e., the time spent on each edge is the same), the agents in real-world networks do not synchronize because of the various travel times spent on different edges. As shown in Fig. 8(a) three agents in the traditional MAPP context are assumed to synchronize at the joint vertexes $(v_1^1,\ v_4^2,\ v_7^3)$, $(v_2^1,\ v_5^2,\ v_2^3)$, and $(v_3^1,\ v_6^2,\ v_3^3)$ at the times t_1, t_2, and t_3, respectively. Therefore, the planner will know that the Agent 1 and Agent 3 meet up at v_2 and interfere on the edge from v_2 to v_3. Note that the superscript indicates the index of an agent and subscript indicates the index of a vertex. However, when traditional MAPP planners are applied directly to real-world networks, as shown in Fig. 8(b), agents will move from the joint vertex $(v_1^1,\ v_3^2,\ v_5^3)$ to $(v_2^1,\ v_4^2,\ v_1^3)$ after the first step, and the interference on the edge from v_2 to v_3 cannot be detected in real time. Thus, without the synchronization property, SC-M* and other MAPP algorithms cannot be directly applied. Note that identical vertexes are shaded in colors and we have $t_1 < t_2 < t_3 < t_4 < t_5$ in Fig. 8.

To deal with this practical issue, we introduce the *forward-the-tail-agent* approach. At each time step, this approach only

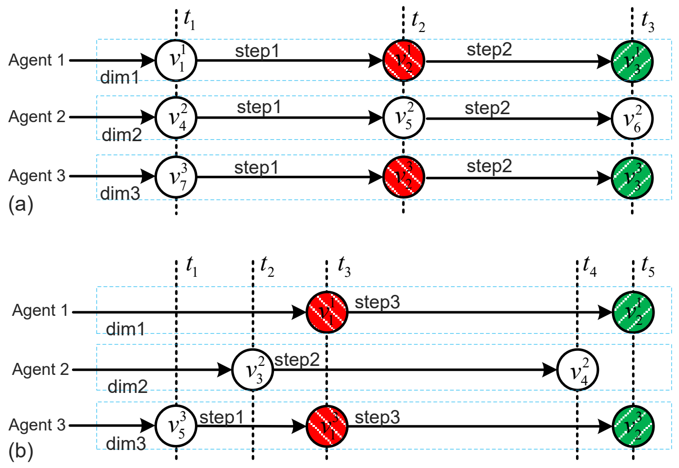


Fig. 8. Forward-the-tail-agent approach to handle non-synchronization in real systems. v_3^2 indicates the event that Agent 2 arrives at vertex v_3.

forwards the agent at the tail (i.e., the last agent in the time flow). If multiple agents synchronize at the tail, they will all be forwarded simultaneously. In Fig. 8(b), three agents start with the joint vertex $(v_1^1,\ v_3^2,\ \underline{v_5^3})$ and the approach only forwards Agent 3 (Agent 3 is at the tail in time, which is underscored) from v_5^3 to v_1^3 at the first step, and the joint vertex $(v_1^1,\ \underline{v_3^2},\ v_1^3)$ is reached. At step 2, Agent 2 is at the tail and is forwarded from v_3^2 to v_4^2. Then, the joint vertex $(\underline{v_1^1},\ v_4^2,\ \underline{v_1^3})$ is reached, in which both Agent 1 and Agent 3 are at the tail. At step 3, both Agent 1 and Agent 3 are moved to v_2^1 and v_2^3, respectively, and the interference on the edge from v_1 to v_2 is detected at this step. Compared to the three joint vertexes in the traditional MAPP setting in Fig. 8(a), Fig. 8(b) shows a sequence of four joint vertexes by using the proposed approach: $(v_1^1,\ v_3^2,\ \underline{v_5^3})$, $(v_1^1,\ \underline{v_3^2},\ v_1^3)$, $(\underline{v_1^1},\ v_4^2,\ \underline{v_1^3})$, and $(v_2^1,\ \underline{v_4^2},\ v_2^3)$. The forward-the-tail-agent approach can also be directly applied to the synchronization scenario in Fig. 8(a), performing exactly the same as the traditional MAPP algorithms.

Based on the expectation graph and the forward-the-tail-agent approach, the SC-M* planner can be generalized to the real Porto bus transit system to generate mobility plans for client passengers. The client passengers' mobility planning is based on the expectation graph in the planning phase and executed in the testing environment of a future unseen weekday (provided by the testing simulation) in the testing phase.

VI. Experiments and Evaluations

This section evaluates the customized SC-M* using the on-vehicle experience simulator. The evaluation has two parts:

- First, we start with the planning tasks for a small number of client passengers under the strict hard-collision-free constraint. For this experiment, we can conduct a *one-by-one checking* on the quality of each generated mobility plan.
- Second, we then apply the SC-M* to a large number of passengers under the soft-collision context, in which we can conduct a *property checking* on the scalability (i.e., the run time and success rate under various client passenger sizes) and on the satisfaction of the soft-collision constraint of all generated mobility plans. In this experiment, the planning tasks are O-D pairs of client passengers randomly generated over the city.

We run SC-M* with an Intel Core i7-6700 CPU at 3.4 GHz with 16 GB RAM. We consider two public resources: Wi-Fi and available space on public vehicles. The goal of SC-M* is to find a joint path in the public transit system for multiple client passengers, optimizing the overall multi-criteria objective constrained on the soft-collision-free condition.

The cost objective, which is customized according to passengers' preference, is defined in terms of three criteria: travel time, Wi-Fi cost, and space cost. The Wi-Fi cost of an edge is given by $max(0,\ 150 - ave\_WiFi\_data\_rate)$, which is a rectified linear function of the negative average Wi-Fi quality in Mbps. The cost regarding the lack of space is quantified using the passenger volume on the edge. Note that the travel time, average Wi-Fi quality, and passenger volume are the three attributes of an edge given by the expectation graph, which is defined in Section V-A. We apply the scalarization method to define the cost objective in SC-M* and optimize over the linear combination of the three criteria. The overall cost of an edge used in the SC-M* is given by:

$$cost_{edge} = \alpha \cdot cost_{time} + \beta \cdot cost_{WiFi} + \gamma \cdot cost_{space}, \quad (10)$$

where α, β, and γ are the scale coefficients.

A. Evaluations with Respect to Path Cost and the Hard-Collision-Free Constraint

1) Experimental Setup: We first apply the SC-M* to a small client passenger size and investigate each mobility plan's quality with respect to path cost and satisfaction of the hard-collision-free constraint. Specifically, each experiment corresponds to a planning task for five client passengers under the strict hard-collision-free constraint (i.e., the collision threshold T in Eq. (6) is set to zero). To start with a small number of client passengers and the hard-collision context is necessary because we can easily check and evaluate the quality of each generated plan one by one.

The planning task of each experiment is designed to encounter at least one collision along the individually optimal path. Thus, the SC-M* paths must differ from the individually optimal paths. We use the standard A* as the individually optimal planner. We evaluate the quality of each plan generated by SC-M* and individually optimal A* via the *testing simulation* of the final Wednesday in May, which is unseen during the planning phase. Because the expectation graph is based on the other three Wednesdays in May, it is expected that the environment in the testing simulation is similar to the expectation graph in terms of the non-client passenger flow and road traffic. Along this line, the planner should achieve a reasonably low path cost with the collision-free constraint satisfied for each planning task.

We design five planning tasks (one task per experiment). In each experiment, we

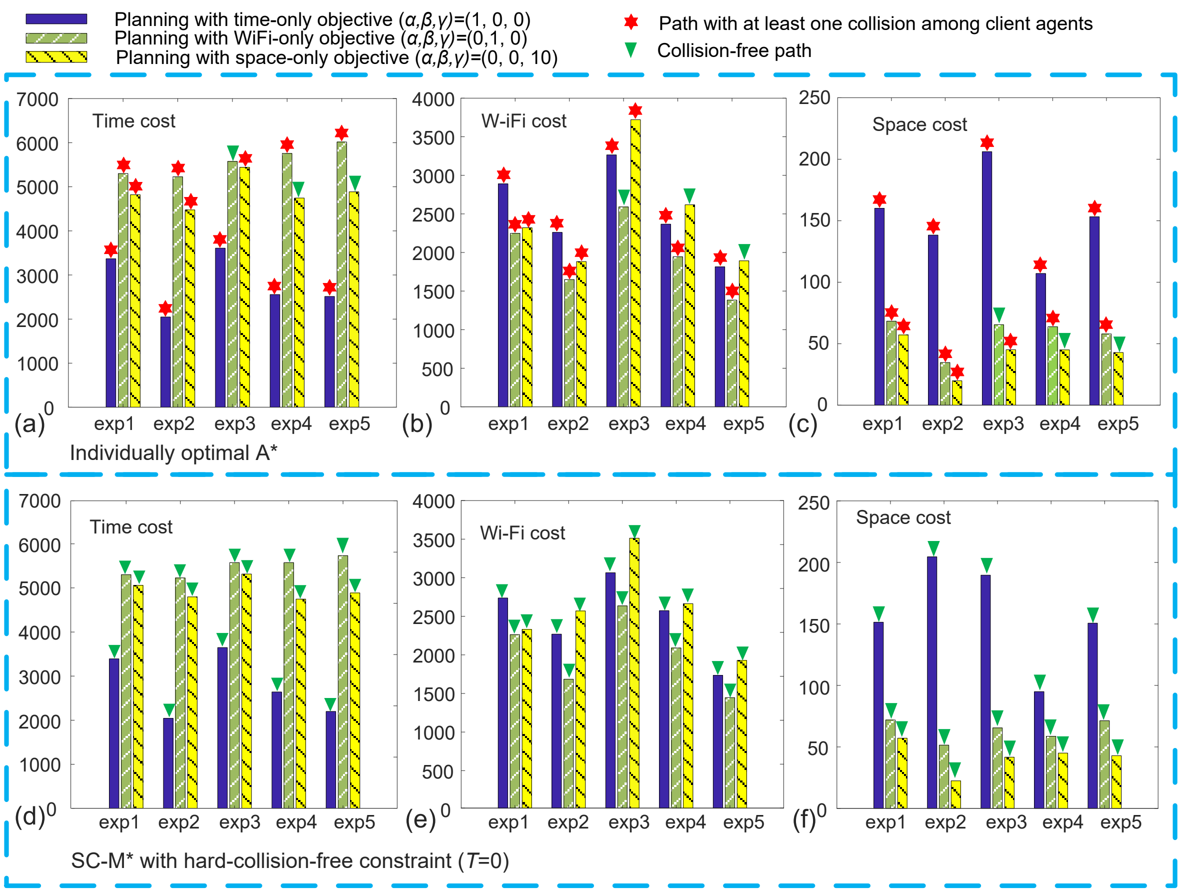


Fig. 9. Comparison of the path costs between individually optimal A* and SC-M* ($T = 0$).

- Apply the two MAPP planners (individually optimal A* and SC-M* [$T = 0$]) separately to the task;
- Each planner generates three independent plans for each task using the time-only (α, β, γ = 1, 0, 0), Wi-Fi-only (α, β, γ = 0, 1, 0) and space-only (α, β, γ = 0, 0, 10) objectives, respectively.

2) Experimental Results: We execute the planned paths in the testing simulations and obtain the path cost and collision information of the planned paths, as presented in Fig. 9. From the figure, we observe that:

- The time cost (i.e., the travel time of the planned path) is minimized by both planners under the time-only objective (see Fig. 9(a) and Fig. 9(d)) in the five experiments. The same phenomenon consistently occurs to the Wi-Fi cost under the Wi-Fi-only objective (Fig. 9(b) and Fig. 9(e)) and to the space cost under the space-only objective (Fig. 9(c) and Fig. 9(f)). These results verify the basic function of both planners of minimizing the customized cost objectives;
- Most individually optimal A* paths encounter collisions (marked with red asteroid indicators), whereas the paths from the SC-M* are all collision-free (marked with green triangle indicators).

The experimental results demonstrate that the SC-M* can optimize the path cost while preventing each client passenger from collisions, and thus, improving the client passengers' travel experience conditioned on the collision-free constraint.

B. Evaluations with Respect to Scalability

This paper attempts to mitigate the scalability problem in real world: The global search space grows exponentially with the number of agents and will become prohibitively large when planning on a city scale; the involvement of multiple preference types and resources produces more collisions, leading to a quick increase in dimensional expansion. Three techniques can improve the scalability: To 1) choose a reasonable time window (e.g., 40 seconds) to limit the number of client passengers that SC-M* must deal with at a time for the whole city; 2) reduce the chance of collisions by tuning the build-in parameters of SC-M*; and 3) group several similar passengers (i.e., passengers sharing the same preferences and the same O-D demands) as a single agent when running the planner.

In this section, we evaluate the scalability of SC-M* in real systems. In the experiments,each client passenger attempts to move from the origin to the destination with a low path cost and with bounded collision scores (i.e., the probability of collision [defined in Eq. (5)] is bounded).

1) Two-Resource-Two-Passenger-Type Setting: In this experiment, we consider two types of client passenger requesting two public resources: Wi-Fi and space (i.e., $A = \{A_1 : “WiFi”, A_2 : “Space”\}$). We use client passenger group as an agent which contains a group of ten client passengers. The

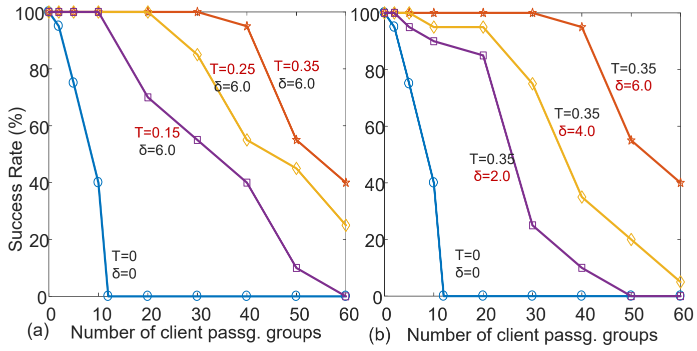


Fig. 10. (a) The impact of the collision threshold T on the success rate of SC-M*, given $\delta = 6.0$; (b) The impact of the offset parameter δ on the success rate, given $T = 0.35$.

TABLE I
THE IMPACT OF T AND δ RUN TIME (IN SECONDS)

m	T=0 (M*)	T=0.15	T=0.25	T=0.35	T=1.0 (idv. A*)	δ=2	δ=4	δ=6
2	97.55	30.45	32.94	30.93	16.56	33.88	30.14	30.93
5	381.26	79.43	83.52	80.70	45.10	151.73	77.84	80.70
10	757.95	177.58	182.21	160.93	89.80	300.97	222.63	160.93
20	>1500	733.19	340.17	436.08	180.48	508.90	369.16	436.08
30	>1500	1023.95	648.30	602.10	269.99	1266.13	769.68	602.10
40	>1500	1254.28	1023.05	682.56	358.16	1408.25	1255.60	682.56
50	>1500	1460.95	1160.49	1134.24	450.35	>1500	1364.94	1134.24
60	>1500	>1500	1336.63	1357.88	540.84	>1500	1497.44	1357.88

Left: δ = 6.0 Right: T = 0.35

Wi-Fi capacity is the data rate of the edge and the space capacity is the number of the remaining seats. We set the satisfying values to $\varepsilon_1 = 60$ and $\varepsilon_2 = 5$ for Wi-Fi and space, respectively. Type I client passenger groups apply f_1 as the collision CDF for the Wi-Fi resource and the linear CDF f_2 for the space resource, implying that they view Wi-Fi and space as important and trivial, respectively. On the other hand, type II client passenger groups use f_2 for the Wi-Fi resource and f_1 for the space resource. Each client passenger group has a 50% chance of being type I. Both CDFs are adjusted using the offset parameter δ, and the collision score is constrained by the collision threshold parameter T. The parameters and symbols used here are defined in Section IV.

Each trial randomly generates O-D pairs for client passenger groups, and we give each trial 1500 seconds to find a solution. For each configuration (including the number of client passenger groups, the collision threshold T, and the offset parameter δ), we run 20 trials to calculate the average metrics (i.e., the success rate and run time).

2) Success Rate and Run Time Results: We study the influence of the collision threshold T and offset parameter δ on the SC-M* performance. Fig. 10(a) shows the success rate of the SC-M* in the planning phase under the collision thresholds of T = 0 (equivalent to the basic hard-collision-free M*), 0.15, 0.25 and 0.35 while the offset parameter is fixed to $\delta = 6.0$. Table I-Left shows the run time (in seconds) in the planning phase under various collision thresholds. Here, $T = 1.0$ is equivalent to individually optimal A*, denoted

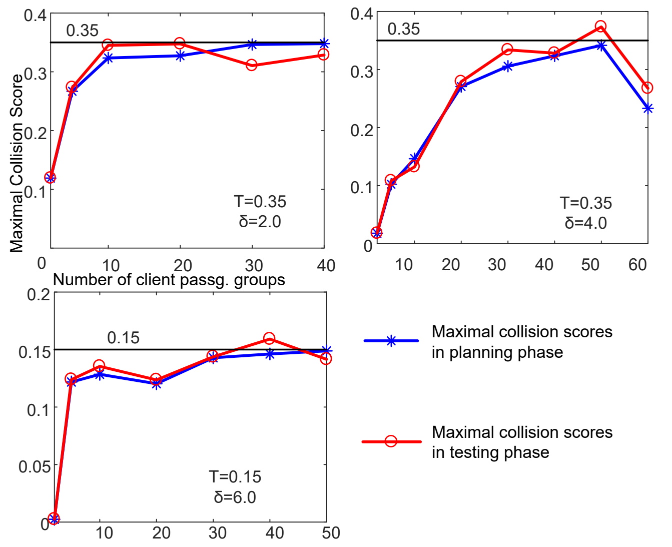


Fig. 11. Maximal collision scores of the planned paths in the planning phase and in the testing phase. To satisfy the constraint, the observed maximal collision score is expected to be less than the collision threshold T.

as idv. A*. Fig. 10(b) shows the success rate under the offset parameters of δ = 2.0, 4.0, and 6.0 while the collision threshold is fixed to $T = 0.35$. Table I-Right shows the corresponding run time.

The results clearly show that a large collision threshold T and offset parameter δ improve the scalability (i.e., improving the success rate of finding a solution and reducing the run time under large client passenger group sizes [$m > 40$] in the planning phase). The planning under the hard-collision-free constraint, which is equivalent to applying the basic M* to the public transit network, does not scale well as it can only handle the planning tasks for fewer than 12 client passenger groups.

The experimental results show that for the SC-M*, larger thresholds and offset parameters render more relaxed constraints; thus, each client passenger is expected to tolerate more interference on the resources before announcing a collision. With this property, one can tune the parameters to trade off the scalability against the slackness of the constraints, being flexible when planning on a city scale. This experiment also demonstrates that the SC-M* can feasibly handle a complex environment with multiple resources and multiple types of client passengers.

3) Satisfaction of the Constraints on Collision Scores: The scalability should also consider satisfying the soft-collision constraints in the testing phase. We execute all the planned paths in the testing simulation (i.e., the simulation of the unseen Wednesday on May 26, 2010) and extract the collision scores of each client passenger group to check the satisfaction of the constraints. Fig. 11 presents the maximal collision score observed for each configuration when executing the planned paths in the planning phase and in the testing phase, respectively. Note that each point in the figure corresponds to the maximal collision score in the 20 trials of a certain configuration. If the maximal collision score is less than the collision threshold T, the planning of the SC-M* under the

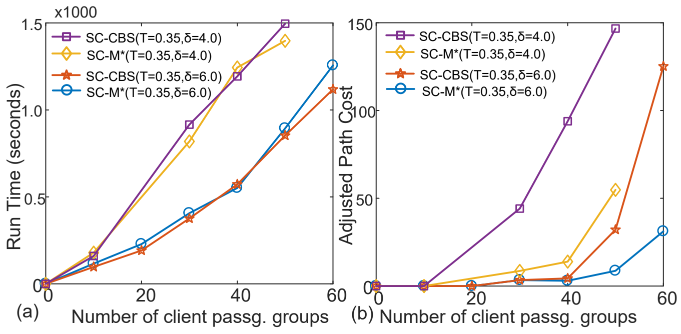


Fig. 12. Comparison between SC-M* and SC-CBS on run time and cost.

corresponding configuration is successful for all 20 trials.

From Fig. 11, we observe that all the collision scores in the planning phase (blue curves with asteroids) are below the given collision thresholds (black horizontal lines), indicating that the SC-M* plans strictly satisfy the soft-collision constraints in the environment described by the expectation graph. For the collision scores in the testing phase (red curves with circles), we observe that, despite a few outlier points slightly exceeding the collision thresholds, the executions of the planned paths in the testing simulation perform reasonably well and confine the collision score of the on-vehicle experience below the bound, even if the number of client passenger groups is large.

The experimental results support that the SC-M* can scale to assure the quality of the plans, satisfying the soft-collision constraints in the expectation graph and extending the soft-collision-free property to the real-world environment, as long as the expectation graph reasonably reflects the real public transit environment.

C. Comparison to Baseline

We compare the proposed SC-M* to another SC-based MAPP algorithm: SC-CBS. SC-CBS extends and customizes the state-of-the-art MAPP algorithm, conflict-based-search (CBS) [14] to the multi-passenger path planning problem in Eq. (8). SC-CBS is a two-level algorithm. At the high level, conflicts are added into a *conflict tree* (CT). At the low level, solutions consistent with the constraints given by the CT are found for each agent. Collisions are formatted as [agent j, edge e, step s], indicating that agent j collides on edge e at step s. The collision information is stored in tree nodes and inherited by their descendant nodes. In each iteration, SC-CBS selects a tree $leaf$ and conducts decoupled planning to find solutions consistent with the constraints associated to the leaf. If collisions occur in the newly-planned path, the conflict tree grows by adding new leaves, otherwise, a joint collision-free path is found.

We apply both SC-M* and SC-CBS to two-resource two-passenger-type settings. In each trial, both algorithms plan for multiple client passenger groups, and we give each algorithm 1500 seconds to find a solution. For each configuration, we run 20 trials to calculate the average run time and path cost.

Fig. 12 shows the results. The adjusted path cost denotes the additional path cost from the individually optimal A*. From Fig. 12(a), we observe that the two algorithms are comparable in terms of run time. However, in terms of path cost in Fig. 12(b), SC-M* is noticeably superior to SC-CBS, especially when the number of agents is large.

VII. Conclusion

In this paper, we propose a customized SC-M*, a scalable multi-passenger multi-criteria mobility planner, to improve the on-vehicle experience of passengers when travelling through the public transit system in a city. The proposed methodology can optimize the passengers' experience while limiting the probability of collisions among passengers to a bound. The parameters control the slackness of the constraints and brings advanced scalability to the SC-M*, making it a powerful tool in practice. The case study of the Porto city, Portugal shows the advantages of the SC-M* in terms of path cost, success rate and run time.

Future work is twofold: First, we will leverage advanced variants of M*, such as EPErM* and ODrM* [13], to remove the basic A* component in our planner. We believe that better performance can be obtained this way because these variants improve the coupled planner and policy generator (two important components in the basic M*), which are directly related to the M* bottlenecks that limit the planning scalability.

Second, we will consider on-line planning to deal with the failures in executing the plans in the real world. The SC-M* generates plans based on the off-line expectation graph in the planning phase, implying that failures may occur at some points during the real-world executions. Solving this problem requires a fast re-planner at the execution time from the location of failure to the destination given the state of public networks, which is a promising direction in the future.

Appendix A
SC-M* Description

SC-M* is a general solver to the MAPP problem in Eq. (8) and the pseudocode for SC-M* is presented in Algorithm 1, where critical commands relative to the soft-collision constraint are underscored. In this algorithm, Lines 1–7 initialize each vertex v in the vertex set V with infinite cost from the source v_s (the cost of v_s itself is zero), make dissatisfying experience to zero and make *collision set* C_k empty. The collision set is used to store indexes of collided agents, which is defined by Eq. (3) in [2]. The initial open list contains v_s only (Line 8). In each iteration, SC-M* expands the first-ranked vertex v_k in the open list ordered by the total cost $v_k.cost + heuristic[v_k]$ (Lines 10 and 11). The algorithm terminates and returns the result if the expanded v_k is the destination v_d (Lines 12–14) or jumps to the next iteration if immediate collision occurs at v_k, i.e., $\widetilde{\psi}(v_k) \neq \emptyset$ (Lines 15–17). Line 18 constructs the set of *limit neighbors* V_k^{nbh} of v_k. A limit neighbor is defined by Eq.(4) in [2] and is the core component of the *sub-dimensional expansion* strategy used by M* algorithm. For each vertex v_l in V_k^{nbh} (Line 19), SC-M* adds v_l to the *downstream vertex set* V_k of v_k (Line 20). The downstream vertex set is used for constructing and updating the collision set C_k and the definition is included in

Algorithm 1 Soft-collision M*

```
Input: $v_s$: source joint vertex; $v_d$: destination joint vertex; $\{V, E\}$: joint configuration graph;
    $A$: set of resources
Output: Path finding results
 1: for all $v_k$ in $V$ do
 2:     $v_k$.cost ← $+\infty$
 3:     $v_k$.exp ← all zero experience
 4:     $C_k \leftarrow \emptyset$    //$C_k$ is called collision set to store indexes of collided agents. See Eq.(3) in [2] for definition
 5:     $v_k$.traceBack ← $\emptyset$
 6: end for
 7: $v_s$.cost ← 0
 8: open ← $\{v_s\}$
 9: while open$\neq \emptyset$ do
10:     open.sort() by $v$.cost+heuristic[$v$]    //to sort the open list from small to large.
11:     $v_k$ ← open.pop()    //pop out the first-ranked vertex
12:     if $v_k = v_d$ then
13:         return back_track_path[$v_k$]    //optimal path found
14:     end if
15:     if $\psi(v_k) \neq \emptyset$ then
16:         continue    //skip the vertex in collisions
17:     end if
18:     construct the limited neighbours $V_k^{nbh}$ of $v_k$    // limited neighbours is the set of vertexes to expand. It is defined by Eq.(4) in [2]
19:     for $v_l$ in $V_k^{nbh}$ do
20:         add $v_l$ to $V_k$    //$V_k$ is the set of "downstream" vertexes existing in expended graph. It is used for constructing and updating collision set $C_k$
21:         Exp_update($v_l$, $A$)    //update experience of $v_l$ using Eq.(3)
22:         $C_l \leftarrow C_l \cup \widetilde{\psi}(v_l)$
23:         Backpropagate_update($v_k, C_l$, open)    // update open list and all the affected collision sets using backpropagation method given by Algorithm 1 in [13]
24:         if $\widetilde{\psi}(v_l) = \emptyset$ and $v_k$.cost+$e_{kl}$.cost $< v_l$.cost then
25:             $v_l$.cost ← $v_k$.cost+$e_{kl}$.cost
26:             $v_l$.traceBack← $v_k$
27:             open.add($v_l$)
28:         end if
29:     end for
30: end while
31: return no path exists
```

Eq.(3) in [2]. The algorithm then updates the dissatisfying experience of v_l using Eq.(3) (Line 21), and include the indexes of immediate colliding agents at v_l to its collision set C_l (Line 22). When the new collision set of v_l is obtained, SC-M* updates the open list and all the affected collision sets using *backpropagation* given by Algorithm 1 in [13]. After the updating operation, if v_l is free of collisions and has improved cost, SC-M* accepts the new cost by saving the trace-back information and adding v_l to the open list for future expansion (Lines 24–28). This process repeats until the open list is empty (Line 9) (i.e., no solution exists) or when the optimal solution is found (Lines 12–14).

ACKNOWLEDGMENT

This research was funded by the FCT under the Sensing and Serving a Moving City (S2MovingCity) project (Grant CMUP-ERI/TIC/0010/2014), which is part of the CarnegieMellon-Portugal Entrepreneurial Research Initiatives (ERIs). The authors would like to thank Teresa G. Dias, António A. Nunes, and João F. Cunha for providing the AFC data.

REFERENCES

[1] S. Elkosantini and S. Darmoul, "Intelligent public transportation systems: A review of architectures and enabling technologies", in *Proc. of the 2013 International Conference on Advanced Logistics and Transport (ICALT)*, Sousse, Tunisia, May 2013, pp. 233–238.

[2] R. Shi, P. Steenkiste, and M. M. Veloso, "SC-M*: A multi-agent path planning algorithm with soft-collision constraint on allocation of common resources," *Applied Sciences*, vol. 19, no. 4037, pp. 1–21, Sept. 2019.

[3] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, and R. F. Werneck, "Route planning in transportation networks," in *Algorithm Engineering*. Cham, Switzerland: Springer, 2016, pp. 19–80.

[4] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Systems Science and Cybernetics*, vol. 4, no. 2, pp.100–107, Jul. 1968.

[6] T. A. J. Nicholson, "Finding the shortest route between two points in a network," *The Computer Journal*, vol. 9, no. 3, pp.275–280, Nov. 1966.

[7] E. Q. V. Martins, "On a multicriteria shortest path problem," *European Journal of Operational Research*, vol. 16, no. 2, pp.236–245, May 1984.

[8] B. C. Dean, "Continuous-time Dynamic Shortest Path Algorithms," Ph.D. dissertation, Dept. EECS, M.I.T., Cambridge, MA, USA, 1999.

[9] D. Delling and D. Wagner, "Time-dependent route planning," in *Robust and Online Large-Scale Optimization*. Heidelberg, Berlin, Germany: Springer, 2009, pp. 207–230.

[10] R. Geisberger, M. Kobitzsch, and P. Sanders, "Route planning with flexible objective functions," in *Proc. of the 12th Workshop on Algorithm Engineering and Experiments (ALENEX)*, Austin, TX, USA, Jan. 2010, pp. 124–137.

[11] X.-F. Xie, S. F. Smith, L. Lu, and G. J. Barlow, "Schedule-driven intersection control," *Transportation Research Part C: Emerging Technologies*, vol. 24, no. 2012, pp.168–189, Oct. 2012.

[12] G. Wagner and H. Choset, "M*: A complete multirobot path planning algorithm with performance bounds," in *Proc. of the 2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Francisco, CA, USA, Sept. 2011, pp. 3260–3267.

[13] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artificial Intelligence*, vol. 219, pp. 1–24, Feb. 2015.

[14] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artificial Intelligence*, vol. 219, pp. 40–66, Feb. 2015.

[15] R. Shi, P. Steenkiste, and M. Veloso, "Generating Synthetic Passenger Data through Joint Traffic-Passenger Modeling and Simulation," in *Proc. of the 21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, Maui, HI, USA, Nov. 2018, pp. 3397–3402.

[16] R. Shi, P. Steenkiste, and M. Veloso, "Second-order destination inference using semi-supervised self-training for entry-only passenger data," in *Proc. of the 4th IEEE/ACM Int. Conf. on Big Data Computing, Applications and Technologies (BDCAT)*, Austin, TX, USA, Dec. 2017, pp. 255–264.

[17] L. L. Peterson and B. S. Davie, *Computer Networks, 5th Edition: A systems approach*. Burlington, MA, USA: Morgan Kaufmann Publishers, 2011.

[18] M. Antoni, "Comparison of 802.11ac and 802.11n PHY Layers," *Kwartalnik Naukowy Uczelni Vistula*, vol. 2, no. 40, pp. 111–123, 2014.

[19] Cisco Wireless Mesh Access Points, Design and Deployment Guide, Release 7.3.
[Online]. Available: https://www.cisco.com/c/en/us/td/docs/wireless/technology/mesh/7-3/design/guide/Mesh/Mesh_chapter_0100.html

[20] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, June 2008.

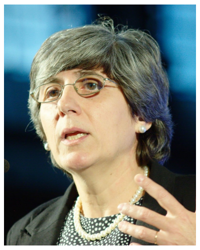
[21] M. Müller-Hannemann, F. Schulz, D. Wagner, and C. Zaroliagis, "Timetable information: Models and algorithms," in *Algorithmic Methods for Railway Optimization*. Heidelberg, Berlin, Germany: Springer, 2007, pp. 67–90.

Rongye Shi (S'17, M'19) received the M.S. degree in Electronics and Communication Engineering from Peking University, Beijing, China in 2014 and the Ph.D. degree in Electrical and Computer Engineering from Carnegie Mellon University in 2019. He is now a researcher at Columbia University.

His research is in the area of data science, machine learning, multi-agent path planning and their applications to intelligent urban mobility for people in cities. He received the 2014 Google Excellence Scholarship, the 2015 SONIC John Bardeen Student Research Award for excellence in nanodevices research, the NSF Student Travel Award for BDCAT'17, and the Best Paper Award for ACM GLSVLSI'17. Dr. Shi was a regular session co-chair for ITSC'18.

Peter Steenkiste (F'10) received the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University in 1983 and 1987, respectively. He is a professor of Computer Science, and Electrical and Computer Engineering at Carnegie Mellon University. His interests are in networking and distributed systems and he has participated in research projects in areas such as high-performance networking, quality of services, wireless networks, and networking in smart cities.

He has been on many program committees and he was the co-chair for the OPENSIG'99 workshop and IWQOS'00. He was also the program chair for HPDC'2000 and general co-chair for ACM SIGCOMM'02 and program co-chair for MOBICOM'08. He has been an associated editor for IEEE Transactions on Parallel and Distributed Systems (1998-1999), IEEE/ACM Transactions on Networking (2000–2003), and Cluster Computing (2000–2004). Professor Steenkiste is a member of the ACM and a fellow of the IEEE.



Manuela M. Veloso (F'11) received the Ph.D. degree in Computer Science from Carnegie Mellon University in 1992. She is the Herbert A. Simon University Professor in Computer Science at Carnegie Mellon University, where she was the head of Machine Learning Department from 2016–2018.

She researches in artificial intelligence, especially in the field of multi-agent systems and robotics. She founded and directs the CORAL research laboratory, for the study of autonomous agents that Collaborate, Observe, Reason, Act, and Learn. Recently, she extended her research to improve human life in cities with the focus to capture and process data about cities and people in them to have an impact on the universal quality of life. She is the past president of the AAAI (2011–2016) and is the co-founder, trustee, and past president of the RoboCup Federation. Professor Veloso is a fellow of the IEEE, ACM, AAAI and AAAS.