Agent-based Framework for Self-Organization of Collective and Autonomous Shuttle Fleets

Antonio Bucchiarone* [iD], Martina De Sanctis† [iD], Nelly Bencomo‡ [iD],

*Fondazione Bruno Kessler, Trento, Italy
bucchiarone@fbk.eu

†Gran Sasso Science Institute (GSSI), L'Aquila, Italy
martina.desanctis@gssi.it

‡SEA, CS, Aston University, Birmigham, UK
nelly@acm.org

Abstract—The mobility of people is at the center of transportation planning and decision-making of the cities of the future. In order to accelerate the transition to zero-emissions and to maximize air quality benefits, smart cities are prioritizing walking, cycling, shared mobility services and public transport over the use of private cars. Extensive progress has been made in autonomous and electric cars. Autonomous Vehicles (AV) are increasingly capable of moving without full control of humans, automating some aspects of driving, such as steering or braking. For these reasons, cities are investing in the infrastructure and technology needed to support connected, multi-modal transit networks that include shared electric Autonomous Vehicles (AV). The relationship between traditional public transport and new mobility services is in the spotlight and need to be rethought. This paper proposes an agent-based simulation framework that allows for the creation and simulation of mobility scenarios to investigate the impact of new mobility modes on a city daily life. It lets traffic planners explore the cooperative integration of AV using a decentralized control approach. A prototype has been implemented and validated with data of the city of Trento.

Index Terms—Transportation Planning, Self-Organization, Agent-based Simulation, Autonomous Shuttles

I. INTRODUCTION

Transportation planning is challenging due to the uncertainty and complexity of caring for all feasible settings and corresponding adaptation actions [1]. Further, there is urgent need for mobility innovations to be able to meet the changing needs of users. Recent progress in simulation platforms and their integration with Geographical Information Systems (GIS) make it possible to create mobility planning simulations without requiring extensive expertise. Urban environments with details of traffic flows are now available from providers such as Google [2]. Agent-based simulations allow planners to "populate" a city or larger regions with a collection of agents that have travel patterns. Further, it is now possible to take into consideration Autonomous Vehicles (AV) in such dynamic environments [3], [4].

On the other hand, the impacts of self-driving cars portend significant changes to the transportation ecosystem[1]. Some forecast the end of parking spaces [5]. Others believe that AV will paradoxically increase traffic [6], [7]. Others predict

---

[1] http://tiny.cc/2f61pz

that there will be new classes of traffic problems that occur at scale [5], due to the homogeneity of these transportation systems. While companies are already providing and testing the first AV [8], we are still far from knowing their impact on communities, as there is significant uncertainty concerning AV actual costs and benefits [9], [10]. We argue that it is necessary to develop mechanisms that allow city planners to study how AV are likely to affect travel demands and planning design space [11] and decisions such as optimal road, parking and public transit supply [12], and to model, analyse, and present possible configurations in ways that the citizens can understand and participate [13].

Real scenarios present different challenges. A problem we take into account covers real-time dynamic ride-sharing. It consists of sharing a vehicle among individual travellers for a trip, splitting the travel costs, i.e., tolls and parking fees, thus reducing the costs for the system in terms of cars used, pollution, traffic, etc. Such transportation allows us to reduce the costs at the expense of convenience. Nevertheless, it is not always possible to consider this form of transportation as a valid alternative due to complexity, with the addition that it is generally a disorganised and informal activity. Indeed, it is not simple to coordinate the itineraries of groups of travellers with different origins and destinations, or to match constrains of travellers, drivers, and AV. The complex uncertain dynamics of an environment such as a city and the need of a real-time (last-minute) approach add further levels of complexity to the riders matching.

From this perspective, the growth of *autonomous shuttles* (AS) in urban public environments could enable new services to deal with the new challenges posed by large cities, which require the combination of the mobility of people [14]. In particular, several pilot experimentations prove significant technology development results, as well as the citizens' acceptance in many cities all over the world, in countries such as Germany, France, Switzerland, Finland, Sweden, The Netherlands, and Estonia, as presented by recent research [14], [15].

This paper presents a framework for city planners to model the introduction of innovative mobility scenarios using AS, and offers the algorithms to simulate several phenomena including the decentralized and participatory management of AS. The

goal of the framework is to support the decision-making of the city mobility planners in order to (i) investigate the impact of innovative mobility modes on the traffic of a city, and (ii) elicit information about the environment and uncertainty, with respect to unexpected situations such as breakdowns, heavy traffic and congestion when the scenarios are applied. The framework is multi-agent with decentralized control, and models a city (buildings, roads and intersections) with its inhabitants and cars to explore the benefits and challenges of the integration of AS into the traffic of the city. A main novelty aspect of our contribution is in the intersection between a decentralized collective approach and the participation of AS as agents of the collective of people, cars and elements of the city, to study the impacts of innovative mobility solutions on traffic. The paper also presents the prototype that has been implemented and validated with data of the city of Trento.

The paper is structured as follows: Section II presents the state of the art. The architecture and the models for self-organized systems in the mobility domain are described in Section III. Section IV introduces the algorithmic solution while validation results of the prototype are discussed in Section V. Section VI concludes and discusses future work.

II. STATE OF THE ART

AV are vehicles that can move without the control of humans. In 2018 the U.S. Department of Transportation has released a policy regarding automated vehicles and their safe integration in the transportation system [16]. Some levels of automation have already been integrated into cars that are on the market and many manufactures are venturing in this direction [17][2].

There are many advantages brought by AV, i.e., reduction of carbon production, traffic, and congestion but also fewer accidents, which are caused by driver errors, fatigue, alcohol or drugs up to a big extent [17], [18], [19]. However, even though the levels of security provided by such system are expected to further improve and different test on urban roads have already been made [20], [21], high level of penetration of completely autonomous vehicles cannot yet be guaranteed in the next decade [16], [18]. As such, simulation is paramount to evaluate the AV impacts and their cooperative adaption [9].

In [22] the impact of AV on safety is studied using VIS-SIM's car following models [23] to simulate both the human-driven vehicles and AV. Together with VISSIM, the Surrogate Safety Assessment Model (SSAM) [24] has been introduced to assess potential conflicts of a road network to be used during the simulation. The results proved that with a high penetration rates, the AV improve safety significantly. Other studies have focused directly on modelling systems to automating intersection crossing, thus to help in improving the traffic flow and to minimise collisions. An example can be found in [25], where it is shown how intersections are regulated by a Multi-Agent Autonomous Management (MA-AIM) system. Other studies [18], [26], [27] focused on the application of AV to ridesharing, to therefore study the waiting times of passengers and the impact of dynamic ride-sharing on travellers costs

[2] https://bit.ly/36gZ7JC

and the average service time in such applications. In [28], the authors focused on the decrease of carpooling users due to concerns regarding privacy and lack of trust towards other users (i.e. carpoolers). As a solution, the authors propose a decentralised manager network for carpooling, coupled with a reputation management protocol to help building trust. In [13] the authors present a self-adaptive framework, called TRAPP, which relies on the SUMO traffic simulator [29], [30]. The aim is to optimize traffic flows in a decentralized and participatory way. SUMO is further exploited, together with reinforcement learning, in [31]. The goal of the approach is to avoid traffic jams with dynamic self-organizing trip planning. The impact of connected and autonomous vehicles (CAVs) on the traffic flow has been investigated in [32], where the vehicle-to-vehicle connection via short-range communication is also considered.

As discussed above, the scientific literature has already investigated the potential impacts of AV use on the traffic systems in cities. Expectedly, AV have potential to improve efficiency and traffic conditions in cities [33]. However, no study has focused on the impact of the use of *Autonomous Shuttles* (AS). Autonomous shuttles provide an attractive and flexible solution to move people around in given areas such as industrial campuses and city centres, connecting those areas with main mass transit systems. AS can also be seen to offer new mobility/delivery services into the city center where narrow streets are not easily served by traditional buses. AS can also serve critical areas with minimal new infrastructure while reducing noise and pollution [34], [35]. The collaboration and interactions between different services that AS can provide and the needs of different entities of the city, such as neighbourhoods or employees of companies, need to be studied. Similar to some publications described above, a decentralized participatory mechanism is offered, yet we make emphasis on a decentralized agent-based collective of people, cars and elements of the city (e.g., buildings, roads).

To close this research gap identified, the focus of this paper is to realize a framework to *model*, *simulate* and *analyze* dynamic and self-organized mobility systems through a Multi-Agent System (MAS), in which heterogeneous ensembles (AS and travellers) [36], [37] are created. The aim is to provide a solution to allow the simulation and analysis of different mobility scenarios in journeys, in which passengers exploit AS to reach their destinations.

III. SYSTEM ARCHITECTURE AND MODELS

To provide a modular solution that can be customized for different self-organized applications in the mobility domain, we have defined a system architecture consisting of multiple layers, as shown in Figure 1, each one defined using the services of the previous ones. The system architecture and models are an extension of the work presented in [38]. While the initial work was focused only on autonomous vehicles and on a small portion of the Trento city map, here we consider a novel application scenario that involves AS to handle a much larger number of passengers spread over the complete map of the city. This led us to extend the architecture, the models, and to define and experiment a new distributed solution. We have also re-positioned the work w.r.t. the state of the art.

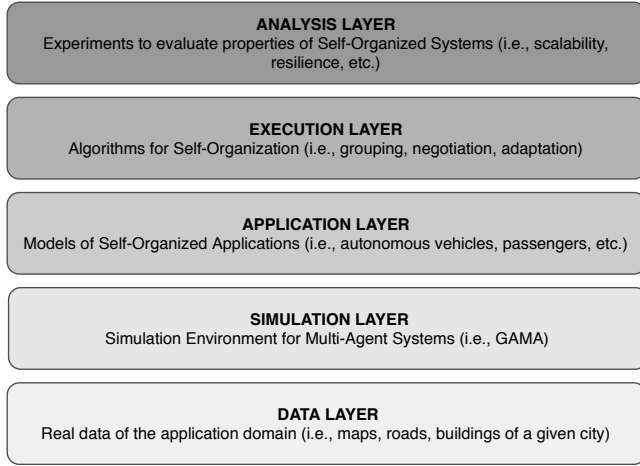In the following sections we describe each layer with the aim of giving details on how our solution works.

Fig. 1: Multi-Layer Architecture for Self-Organized Systems.

A. Data Layer

The DATA LAYER is responsible for creating a virtual environment that replicates a realistic urban scenario. It has access to real data about *buildings*, *roads* and *intersections* of a specific city. Specifically, we use OpenStreetMap[3] as the framework for gathering geodata. Geodata refers to information about geographic locations that is stored in a format that can be used with any geographic information system.

Roads and intersections provide the basis for generating a *road graph*, which support the agents within the simulation. Next, we discuss these concepts and how they integrate into the simulated transportation network.

Buildings are polygons of diverse types stored as a map of `key:value` pairs with the type of building as value. In the GAMA framework [4], buildings are grouped into two categories, namely *residential* and *industrial*.

Roads are polyline composed of different segments. Roads come with attributes that describe characteristics, such as the driving side, or the number of lanes. Roads are implemented as agents as well, with basic behavior. In particular, the attribute `speed_coefficient` helps to model traffic on the roads by slowing down the speed of shuttles in streets with high density. Traffic is also computed.

Crossings, traffic signals, traffic roundabouts, stops and yield signals are tagged as `highway`, along with bus stops and street lamps. In our approach, traffic signals and other relevant intersections were represented as the type `intersection` in compliance with the codes created by Patric Taillandier [39]. Intersections are also modeled as agents.

B. Simulation Layer

At the SIMULATION LAYER, we use the GAMA modelling and simulation framework. GAMA enables the modeling and simulation of spatially explicit agent-based systems where

geodata about real-world maps, streets, buildings, etc. are integrated using GIS data. In GAMA, different types of agents can be programmed with their own *behavior* and *attributes*. The behavior of each agent is supported by functions, in the form of *reflexes* (called automatically at each time step) or *actions* (called by an instance of species). Agents can have *skills*, i.e., built-in modules with a set of skill-related attributes and actions. GAMA provides an IDE and a language, the GAma Modelling Language (GAML). It is an agent-oriented programming language that enables users to define agents, as *species*, whose behaviour is defined by actions and reflexes.

Behind the concepts and the operational semantics of GAML, there is a meta-model made by three main categories of abstract classes, namely *Entities* (i.e., agent and population of agents), *Space*, *Time* and *Species* that combines all the previous classes.

The concept of *species* is similar to that of *class* in object-oriented programming. It defines the attributes, characteristics and functions of the entity it represents. In our approach, three species of agents have been modelled: *people* who are looking for a lift to commute to work and back, *autonomous shuttles* (AS) that offer lifts, and *common cars*, such as traditional non-autonomous cars with drivers. In addition, GAMA supports the development of multi-level agent-based models [40] through a *containment* relationship, thus enabling the definition of *macro-species* and *micro-species*. A *micro-species* can be nested inside a species, i.e., a *macro-species*. Hence, it is possible for a species to host a population of micro-species. There may be several reasons for a multi-level representation of agents. For instance, there may be need to consider agents that are part of organizations and therefore to stay at different levels. In our approach, this relation was useful in order to represent people that enter into a shuttle by defining a micro-species inside the AS species.

Since the subject of this study is in the mobility domain, among the available built-in skills we adopted the following:

- *Movement skills*: Implement actions to simulate the movement of agents in an open space or along a graph. It is employed by the *people* and the *passenger* species.
- *Road skills* [39]: Register agents on the road and is used by the *road* species.
- *Advanced Driving skill* [39]: Simulates agents capable of driving. Both, *cars* and *AS* species have this skill.

The GAMA framework also provides an interactive visualization tool, which gives feedback to the user during the simulation. It also allows users to automatically instantiate agents and create realistic models and simulation of a given geographic area.

C. Application Layer

The APPLICATION LAYER specializes the simulation layer and supports the modeling of dedicated self-organized applications. In this layer we can model the behaviors of all the agents involved in the target application. In our specific case, `People` in the application are individuals who search for both a lift to their working place before a given time, and a lift at a given time to return home. Such agents are

---

modelled by the species *people*. Each agent is initialized with a *living place* and a *working place*, a *starting* and an *ending* working hour and a *list of intersections* at a given distance from its location, used by AS to recognize people that are looking for a lift. This species has the *moving* skill to move on different roads topologies. It can access the attributes *starting point*, *destination point*, *current_path* between these two points, *current_edge*, that is the current agent's location, and the *speed* on the path.

To model people behaviours, two actions are defined, `search_path` and `move`. The former simulates the search for a path in the road graph from the current location, constrained by topology. This is analogous to what happens in real life as nowadays, and before commuting to work, it is common to search the best option on, e.g., Google maps as there may be traffic and taking the usual route may lead to arriving late. Information about the time required to reach the destination (*time_to_cover*) and the distance the agent will cover (*distance_to_cover*) are stored, and the cost for the agent to move is computed. It keeps track of the time needed by the agent to arrive to its destination and the distance it covered. It also computes the distance the agent covered (*dist_covered_alone*). Two further variables, *late* and *actual_time_in* are used to track whether the agent was late at work, and the time the agent started working.

Eventually, the agents' behavior can also be represented as a Finite State Machine (FSM) describing the different states in which the agent can be and the transitions among them, as represented in Figure 2 left side, and further described below:

- **resting**: at a given hour before the start of working time, the agent will switch its state to the state *search_lift_to_work*.
- **search_lift_to_work**: at this state, the agent will set its target to the working place location, search for a route to reach it and then wait for an established amount of time for an available shuttle nearby that could pick it up. After this interval of time has passed, the agent changes its state to *go_work* if no shuttle is found, or to *wait_for_lift* if a shuttle has been found.
- **wait_for_lift**: if the shuttle that can give a lift to the passenger is still far away, the agent will wait in this state for the shuttle to arrive.
- **go_work**: this state can represent two different situations: (i) there are not available shuttles, the agent moves alone towards its working place (the previous state is *search_lift_to_work*); (ii) the shuttle it was waiting for arrives, the agent starts its lift (the previous state is *wait_for_lift*). The agent will change its state to *working* as soon as its location corresponds to its working place.
- **working**: the agent is working. When it is time for the agent to go home, it will change to the state *search_lift_to_home*.
- **search_lift_to_home**: after work, the agent will set its target to its house location and search for the best route to return home. Then it will wait a given amount of time for a potential lift from a shuttle that is passing nearby. After this interval of time has passed, the agent changes its state to *go_home* if no shuttle is found, or to *wait_for_lift* if a shuttle has been found.
- **go_home**: after the given amount of time established to wait for a shuttle, or when the found shuttle arrives, the agent moves towards its home. Eventually, it will change its state to *resting* as soon as its position corresponds to its living place.
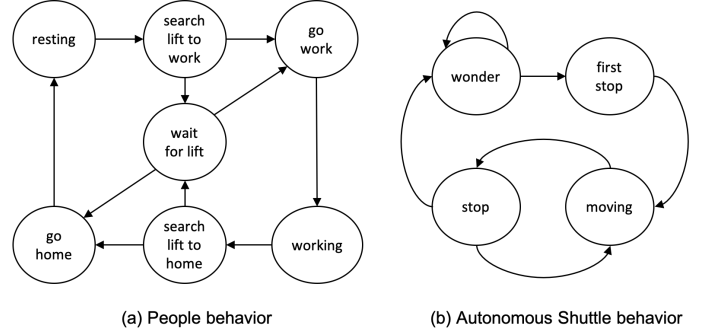

Fig. 2: FSM for the People and Autonomous Shuttle behaviors.

The described FSM represents an abstraction of the overall people agents behaviour, which includes all the required states for our approach. Each state implicitly includes relevant parameters, such as timing concerns (e.g., in state *search_lift_to_work*), even if these are not visible in the FSM representation. Moreover, the FSM could be further specialized to include other parameters, if required by the analysis that is intended to be performed.

The `Driving Model` is based on the one proposed by [41] and re-elaborated by [39] using the GAMA platform. The agents compute a path given an origin and a destination, which may also be chosen randomly, on the road network. This trajectory is composed of a sequence of edges. In the model presented in [41], the movements on an edge were inspired by the Intelligent Driver Model presented in [42], with the addition of the possibility for the drivers to change lane at any time, both when entering a new edge or when already on it. Additional properties define maximum speed, vehicle length, or the "personality" of the driver, such as their attitude towards following the driving rules. In our system, two species of vehicles were simulated: a *common* car and an *autonomous* shuttle agents. Various instances of the former are generated in order to simulate traffic on the roads by letting them wander on the roads. For this reason, common cars have two reflexes: `time_to_go` and `move`. The former is used to set a random target for the agent towards which to move, the latter makes the agent move exploiting the action `drive`. The latter also checks whether the agent has slowed down to a speed below 5 kms/hour in order to decide whether or not to turn around and change route.

Figure 2 right side shows the FSM model of the AS behavior. All the AS start at the *wander* state. They will change state from *wander* to *first_stop* when they find a person or a group of people searching for a lift on the road, triggering the decision-making mechanism. In this case, a final destination is chosen among the targets of the first person or group of people found in order to set an *objective for the shuttle*. Once this objective is set, the shuttle can create a path proposing the

related costs to the first set of probable passengers. After the first set of passengers, who accepted the shuttle offer to get on board, the agent changes its state to *moving*. If there are still open seats, it may propose a lift to passengers found on the road, by changing its state to *stop*, and therefore updating the costs for all passengers.

AS keep track of the distance and the expected time needed to reach their next destination. Data also includes the time that was actually necessary to reach their destinations, the passengers on board and all the stops made to either take on or drop off passengers.

The next two layers, EXECUTION and ANALYSIS, are on top of the previous ones shown above, and are described next.

IV. EXECUTION LAYER

In this Section, we propose a decentralized approach for the self-organization of fleets of autonomous shuttles, as part of the EXECUTION LAYER. The computational power for the grouping of agents and the computation of travel costs are distributed among the AS. Each AS is capable of dynamically finding passengers on the road it is currently travelling and automatically check whether to offer them a lift, based on certain conditions. For instance, if their initial positions and/or
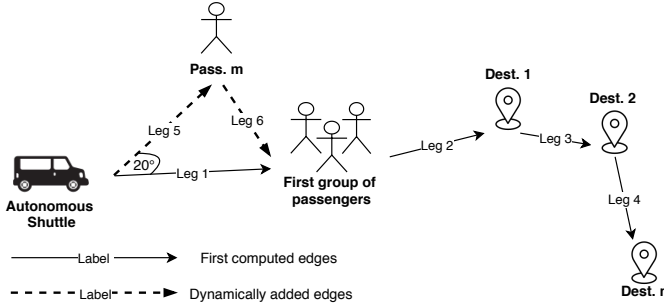

Fig. 3: Dynamic addition of new passengers and path update.

final destinations are too far from the current path of the AS, offering them a lift might lead to an increase of travel time and cost for those passengers already on board, which would be taken into account when deciding to accept the lift or not.

Once started, AS identify their first passenger or group of passengers among those who made a lift request. When the first passengers are found, the AS compute an initial path based on their current positions and destinations. A graphical representation is given in Figure 3, where the `first group of passengers` is identified. The initial path made by their position and destinations is composed by `leg 1`, `leg 2`, `leg 3` and `leg 4`. The final target of the AS travel is computed on the base of the passenger whose destination is the farthermost away (i.e., `Dest. n` in Figure 3). This would also increase the possibility of finding and adding more passengers on the road (i.e., `Pass. m` in Figure 3 that makes a lift request after the initial path has been computed). In the presence of other passengers on the path, the AS will decide whether or not to offer a lift to them, based on the objectives of the passengers already on board. Accepting new passengers lead AS to dynamically update their path (i.e., by adding `leg 5` and `leg 6` in place of `leg 1` in

Figure 3 when `Pass. m` is accepted on board). Moreover, an optimized decision process supporting the dynamic addition of passengers would be beneficial for both AS, which can reach their full occupancy, and for the passengers, who would share the travel costs. Of course, not all passengers on the path can be accepted if it means that having them on board causes a degradation of the travel duration. The latter can be particularly relevant for those passengers on their way to work.

The proposed solution is supported by a set of algorithms: (*i*) to compute an initial path on the base of the first accepted passenger(s), (*ii*) to compute the various legs of the AS path, with the related costs, (*iii*) to dynamically add passengers on the way, and (*iv*) to update the costs each time a new passenger(s) is added to a travelling AS [5]. Algorithm 1 presents the algorithm to manage self-organized AS, when new passengers should be added, i.e., the core of the approach.

Each time a shuttle finds passengers on the road, it will set an initial list of new *potential passengers* with their destinations. To decide about their acceptance on board, a previous analysis needs to be performed. Consequently, this list will be first filtered by the *angle* between the passengers (both on board and new) destinations and the current location of the car. Each new passenger whose computed angle is greater than 20° would be removed from the list, as they are considered too far from the AS current location and serving them would imply a significant negative effect on the time of travel. For instance, Figure 3 shows that `Pass. m` has been accepted since the angle between his/her current position and the position of the AS is equal to 20°. Indeed, the greater the angle is, the greater the travel time will be. Thus, the angle degree must be kept under a certain upper bound value of 20° that we get after running several experiments showing that for values higher than 20° a constant worsening of performance could be observed. After, the list will be filtered by the stops already visited by the AS. In fact, in order to avoid AS to significantly increase its total path, those passengers whose destinations correspond to stops that have already been visited are not accepted for the lift. This helps to avoid AS to make a non-optimized route by revisiting already visited stops on its path to the final destination.

After the initial checks, if there are still passengers in the list, a loop over the new potential destinations, namely *d*, starts in order to further understand whether it is possible to include them as future stops or if the passengers should either wait for another AS or go by their own to find an alternative solution to arrive to their destinations. This loop is the core part of Algorithm 1 (lines **5–40**). First, the last index, namely *max_index* of the *targets* list is acquired (line **7**), then a check is made over the length of the passengers that will be dropped off by the AS at destination *d*: only the number of passengers needed to fill the remaining available seats are considered. If there are no passengers left or, if there are no available seats left (lines **8-9**), the cycle is finished. Otherwise, it is checked whether or not the destination of passengers is already among the next stops of the AS. If

---

that is the case, the passengers are added on board (lines **11-12**). Otherwise, a loop starts (lines **14-38**) over the next stops in order to insert the destination at the appropriate index. The above implies several calculations. The path between the current target and the next target, namely $t2n$, and the path between the current target and the destination d, namely $t2d$, are computed (lines **15-20**). If they are defined for the first time (line **15**), the current target will correspond to the *new_origin* (lines **16-17**), otherwise it will be the destination at index $i-1$ (lines **19-20**). This distinction is necessary since the origin is not in the list of targets, but only in the list of global stops. The length of these two paths are compared (line **21**) and the best option is chosen accordingly. Specifically, if the length to the destination $t2d$ is shorter than the one to the next stop $t2n$, the path from the destination d to the next stop is computed (line **22**). Before the addition of the passengers and their destinations, some additional checks about the very first passenger are needed, as for example, if he/she is going to work (line **24**). An approximation about the time needed to reach his/her working place considering the path variation due to the addition of new passengers is then computed (line **25**). Thereafter, it is checked whether the time variation is affordable, in order avoid lengthening the path unnecessarily. This information is stored in the *check* variable (line **26**) that is used to decide whether adding or not the new passengers (lines **27-28**). At this point, if necessary, the AS current path is recomputed, based on the updated list of stops. If the length to the destination $t2d$ is longer than the one to the next stop $t2n$, the counter i will be incremented and therefore, the loop will start again. Eventually, when the value of i corresponds to the last index of the *targets* list, the destination will be added to the *targets* and to the *stops* as the last stop (lines **33-37**).

After each cycle of the loop over the destinations, the passengers will be removed from the list of potential new passengers (line **40**). Lastly, if there have not been any additional passengers at the current location, i.e., *tot_added_passengers* = 0, the *new_origin*, will be removed from the global list of stops, unless it is the origin of the travel (lines **41-43**).

After the actual new passengers have been identified, the following further steps are performed. The current position of each passenger (i.e. the origin of their travel) is added to the list of the AS' stops before the next stop. Moreover, the path from the AS current location to the destination of the first passenger who boarded the vehicle (and therefore, whose destination coincides with the final target of the AS), is computed again in order to include the new stops. The time to the destination is also updated for each passenger to therefore, make sure they are not running too late and, eventually, the number of remaining seats is computed.

In terms of temporal complexity, the decentralized algorithm just described is *subquadratic* with $O(n*m)$. In particular, n is the number of passengers that must be managed by an AS while m is the number of stops that the same AS must handle. Moreover, since each AS can serve at most 12 passengers at a time, and new passengers cannot be taken on board if there are no more seats, in the worst case m will be equal to 11.

As introduced above, the real-time dynamic ride-sharing is built on the idea of sharing a vehicle among individual

---

Algorithm 1: Decentralised algorithm for adding new passengers after the first passenger or group.

Data: P; *targets*;*original*;
Result: $p \in Passengers$; updated *stops*; updated *targets*; updated *cost_legs*;

```
1   tot_added_passengers ← 0;
2   up_costs_pass ← false;
3   added ← false;
4   open_seats ← max_pass - |passengers|;
5   foreach d ∈ destinations do
6       i ← 0;
7       max_index ← |targets|-1;
8       if open_seats = 0 or |destinations[d]| = 0 then
9           break;
10      else
11          if d ∈ targets then
12              Add_Passengers;
13          else
14              while added = false do
15                  if i = 1 then
16                      t2d ← path(new_origin, d);
17                      t2n ← path(new_origin, targets[i]);
18                  else if i < max_index then
19                      t2d ← path(targets[i-1], d);
20                      t2n ← path(targets[i-1], targets[i]);
                        /* where i−1 is the index of
                           the current stop          */
21                  if |t2d| < |t2n| then
22                      d2t ← path(d, targets[i]);
23                      check ← true;
24                      if 'work' in first.next_state then
25                          change ← t_a(original) - t_a(t2n) +
                                time_a(t2d) + t_a(d2n);
26                          check ← (change<(t_a(original)×3/2))
                                and (av_time≥change);
27                      if check then
28                          Add_Passengers;
29                          if i = 1 then
30                              Change_Path;
31                      else
32                          break;
33                  if i = max_index and not added then
34                      targets ← targets + d;
35                      stops ← stops + d;
36                      Add_Passengers;
37                      as_last ← true;
38                  break;
39              i ← i+1;
40          P ← P - destinations[d];
41  if tot_added_passengers = 0 then
42      if stops index_of new_origin ≠ 0 then
43          stops ← stops - new_origin;
```

---

travellers for a trip, while splitting the travel costs. This allows users to reduce the costs, although it may potentially occur at the expense of passenger's convenience. Computing the final cost for each passenger is not easy when passengers are dynamically boarding and leaving cars. Specifically, the total path of each AS corresponds to a sequence of legs, which is continuously modified due to the addition of passengers on board. Each passenger must contribute only to the costs of those legs for which he/she was on board, independently on the entire path that the AS covers. Accordingly, in our approach the travel cost is calculated based on the computation of the

costs of legs shown by $cost_{leg}$ in equation (1), where the length of the leg and the cost per km are considered.

$$cost_{leg} = \frac{length(leg)}{1000} \times cost_{Km} \qquad (1)$$

Then, for each AS, the travel cost corresponding to the entire path it covered is calculated by the sum of the costs of the legs in the path, $cost_{path}$ in equation (2).

$$cost_{path} = \sum_{i=1}^{n} cost_{leg_i} \qquad (2)$$

Eventually, to compute the cost that each passenger must pay on a given leg for which he/she was on board, namely $cost_{leg_{pass}}$, the $cost_{leg}$ is divided by the number of passengers for that leg, as implied by equation (3).

$$cost_{leg_{pass}} = \frac{cost_{leg}}{|passengers\_on\_board|} \qquad (3)$$

Of course, the total cost for each passenger will be given by the sum of the $cost_{leg_{pass}}$ of each leg he/she has to contribute.

V. ANALYSIS LAYER

The ANALYSIS LAYER executes experiments on top of the EXECUTION LAYER to evaluate properties of self-organized systems. We have specified and executed a set of experiments to evaluate the performance of the proposed solution. Specifically, we are interested to figure out the efficiency of the decentralized approach in terms of served users, waiting time, travel costs, AS usage, in different settings involving diverse number of users and/or AS. We aim to show how simulations can support the mobility service provider to infer the best size of the fleet for serving a given number of users, while guaranteeing the best costs and services for the passengers, as well as avoiding underused shuttles.

The simulations are built on top of real data of the city of Trento (Italy). The experiments were run on a laptop equipped with a quad-core CPU running at 2.7GHz, and 16Gb memory. In the following, we discuss three main evaluations that have been performed.

A. Evaluating the approach with multiple vs. a few passengers destinations

This evaluation aimed to understand the impact of the approach on the efficiency of passengers allocation when considering the following complementary scenarios: (i) each user is assigned a different working place, meaning that diversified and mainly non overlapping travel destinations are considered; (ii) each user is assigned one of the only two available working places; (iii) each user is assigned the same unique working place. The last two scenarios differ from the first one, since they model those cases in which a company can decide to adhere to the AS mobility service and offer it to its employees. This impacts on the final destination of users, used for planning their trip to work, which will coincide. Considering that users' destinations are exploited by AS to decide if get users on board or not, we aim to investigate the impact of only one or a few shared destinations.
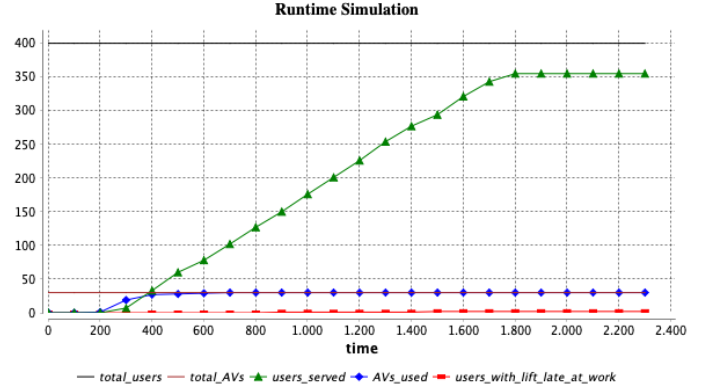

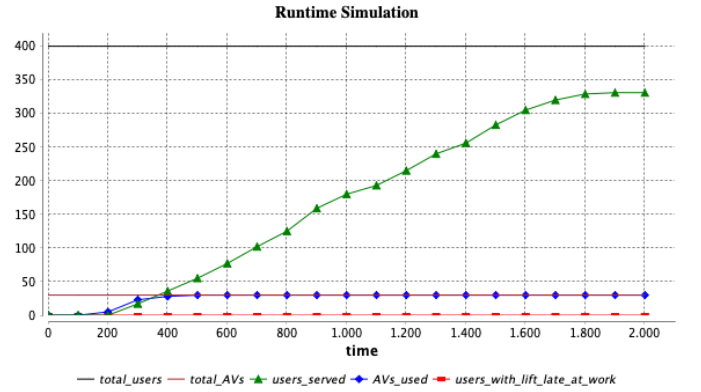Fig. 4: *Exp1* with diversified working places.
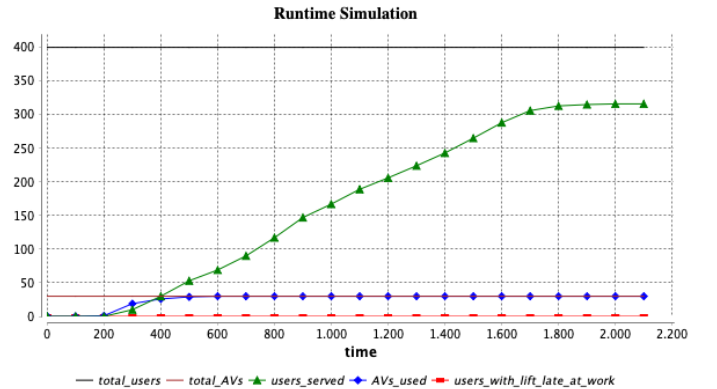

Fig. 5: *Exp2* with two working places.

Fig. 6: *Exp3* with only one working place.

The initial setting for this first evaluation considers a fixed set of agents modeling a fleet of 30 shuttles of 12-places each and 400 users. The cost assumed for the service is of EURO 1 per kilometer (Km), which is then shared among the users sharing the same travel. The time interval for the users to start working ranges between 8:00 am and 10:00 am, meaning that they do not look for a lift to work all at the same time. We have run three experiments for this evaluation, *Exp1*, *Exp2* and *Exp3*, simulating the three scenarios described above, in which diversified, two and only one working places are considered.

In Figures 4, 5, and 6, we report the charts of the three experiments showing the simulation progress till its end.

Specifically, in the *x-axis* the number of simulation cycles are reported, where the minimum duration of a cycle corresponds to 0.01 seconds. Then, each chart displays the number of users served by the fleet of AS, on top of the 400 users requesting for a lift, the total number of AS used and the number of served users arriving late at work. At first sight, it can be noticed that the number of served users slightly decreases while moving from *Exp*1 to *Exp*3.

In Table I, further details about the first evaluation are reported. In particular, first column of Table I lists the most interesting outcomes that have been measured during the simulation, and which are grouped as follows: in the upper part of the table (from the second to the fifth row) there are parameters referring to the users; in the lower part of the table (last three rows) there are parameters referring to the AS. Several observations can be made by looking at the table. As already seen from Figures 4, 5, and 6, the number of served users slightly decreases from 355 in *Exp*1, to 331 in *Exp*2 and 316 in *Exp*3, while only in *Exp*1 we registered two served users arrived late at work.

Interestingly, in the first scenario, where each user is assigned a different working place, thus diversified destinations are considered (*Exp*1), we can observe a higher average travel cost for users, namely EURO 2,60 w.r.t. EURO 1,50 and EURO 1,45 in *Exp*2 and *Exp*3 respectively, and a considerably lower average waiting time of 2,61 minutes, against the 5,44 and 7,18 minutes in *Exp*2 and *Exp*3, respectively. The higher average travel cost is supported by the Kms traveled by the AS, on average, that in *Exp*1 is roughly the double w.r.t. *Exp*2 and *Exp*3. Lastly, we also measured the cumulative user lifts per each AS, i.e., the total number of users served by an AS at the end of its work. There is not a significant difference on the cumulative user lifts given on average by each AS, in the three scenarios, which is only slightly greater in *Exp*1 where, however, more users are served. Considering that it ranges roughly between 7 and 17 in the scenarios, we can also state that a fleet of 30 shuttles is enough for a total of 400 users.

TABLE I: Comparison between *random* (*Exp*1) and *fixed* (*Exp*2 and *Exp*3) number of working places.

	*Exp*1	*Exp*2	*Exp*3
Users served	355	331	316
Served users late at work	2	0	0
Avg. travel costs	EURO 2,60	EURO 1,50	EURO 1,45
Avg. waiting time	2,61 min	5,44 min	7,18 min
Avg. cumulative user lifts [range]	12 [8-17]	11 [7-16]	11 [7-15]
Tot. gain for the AS service	EURO 923,95	EURO 497,04	EURO 457,62
Avg. Km traveled by AS	29,42 km	16,60 km	15,25 km

Figure 7 shows the distribution of travel costs in the three scenarios, where the median value is reported. It highlights a wider distribution of costs in the case in which diversified working places are considered, bringing to the higher average travel cost discussed in Table I. From this evaluation, it results that managing a considerable set of users sharing the same working place contributes to making the AS service less expensive and thus, more convenient for companies who may at the same time wish to offer it to their employees, at the expense of increase of waiting time.
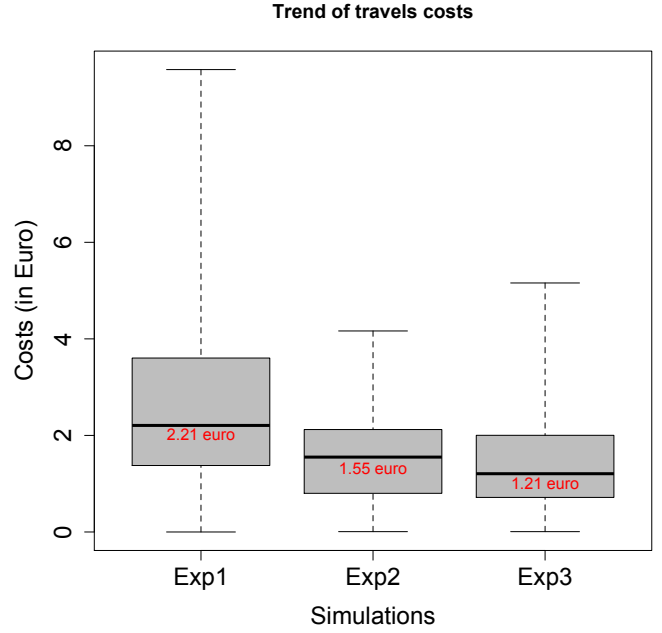

Fig. 7: Trend of travel costs in the three experiments.

However, *Exp*1 shows that when diversified working places are considered, travel costs are subject to a high variance that might discourage the use of a shared ride. This requires the need of keeping the travel costs bounded, in such a way to make the approach reliable. Considering that the travel cost mainly depends on the distance AS cover (see equations (1) and (2)), at a first stage we are considering to bound such distance. In particular, the high variance shown in Figure 7 depends on the way in which AS identify their first group of passengers among those who made a lift request, as described in Algorithm 1. AS compute their initial path based on the passengers destinations, by selecting the farthermost away as their final target. Our idea for future work is that of modifying our approach by computing an upper bound to the final target distance from the origin of AS, by gradually excluding the passenger whose destination is the farthermost away but overpasses the established upper bound. Eventually, since travel costs are also affected by the number of passengers sharing a ride (see equation (3)), we plan to further extend our approach in order to maximize shared route, as done in [43].

B. Evaluating the efficiency of a fleet of a fixed number of AS

This evaluation aimed to measure the efficiency of a fixed fleet made by 50 shuttles of 12-places each, with varying numbers of total users, namely 250, 500, 750 and 1000. As in the first evaluation, the cost for the service is of EURO 1 per Km whereas the time interval for the users to start working ranges between 8:00 am and 10:00 am. In Figure 8 left side we show the percentage of served users (per each users total number) as well as the percentage of served users arriving late at work. At the same time, for each simulation we further measured the average of cumulative user lifts given by the AS in the fleet, in Figure 8 right side, in order to understand to what extend AS reach or surpass their full occupancy capacity.
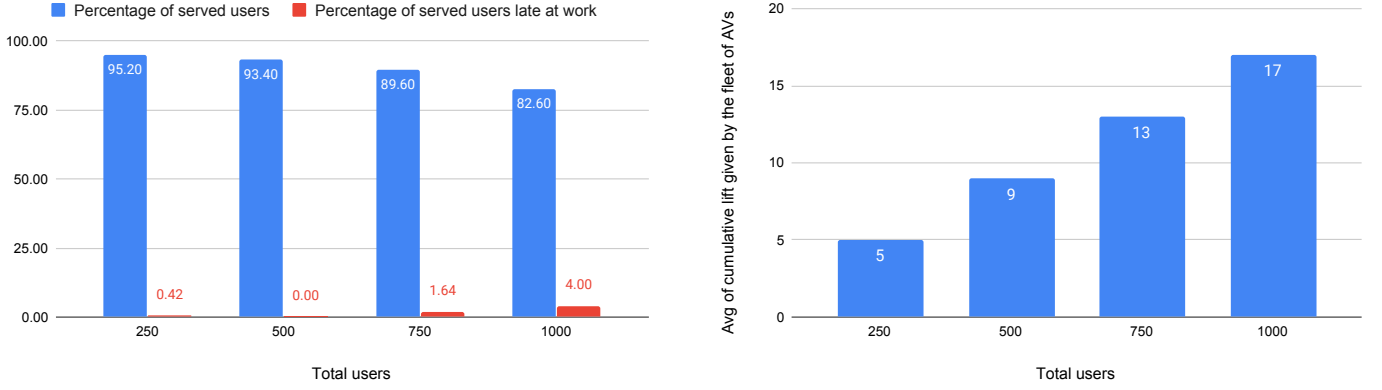
Fig. 8: Fixed fleet of 50 shuttles in scenarios with varying number of total users, from 250 to 1000. **(Left side)** Percentage of served users over the total number of users and percentage of users late at work among the served users; **(Right side)** Average of cumulative user lifts given by the fleet in the different scenarios with varying number of users.

It can be observed that, in terms of served users, the fleet of 50 AS guarantees good performances in all the four scenarios considered, from the 95,20% of served users over a total of 250 users, to the 82,60% over 1000 users. In contrast, the percentage of served users arriving late at work has incremented as the total number of users requesting lifts increases, even if the number of users arriving late at work remains significantly low. The most relevant numbers are registered in the case of 750 and 1000 users, where 1,64% (i.e., 11 users) and 4,00% (i.e., 33 users) among served users arrive late at work, with an average late time of 6,11 and 6,70 minutes, respectively. However, to have a complete overview of the performance of the AS fleet, it is important to observe the average number of cumulative user lifts given by the AS, in each scenario. Indeed, from Figure 8 right side it results that, when serving 250 and 500 users, AS do not reach their full occupancy capacity (i.e., 5 and 9 cumulative user lifts in average) and this might negatively impact on the travel cost split among users sharing the same travel. To the contrary, when serving 750 and 1000 users, the registered average of cumulative user lifts is of 13 and 17, respectively, representing an optimized use of the entire fleet. Eventually, we measured the waiting time for users whose request for a lift has been accepted, in the four analyzed scenarios. The average waiting time increases with the amount of the number of total users. Specifically, it ranges from 1,23 minutes with 250 users to 3,31 minutes in the scenario with 1000 users.

In conclusion, this second evaluation shows that a fleet of 50 AS achieves a good trade-off between the percentage of served users and the usage of AS when serving a number of users in between 750 and 1000, while it is more efficient in terms of served users but, it is also less efficient in terms of AS usage, with less than 750 users approximately.

C. Evaluating the efficiency of the approach in finding the best size of a AS fleet for serving a given amount of users

The third evaluation aimed to establish the best size of a AS fleet for serving a fixed amount of 1000 users, by running simulations with a fleet of different size, i.e., made by 40,

50, 60 and 70 AS. The basic setting is the same as in the first and second evaluations. In Figure 9 left side we show the percentage of served users per each fleet size as well as the percentage of served users arriving late at work. At the same time, for each fleet we measured the average of cumulative user lifts given by the AS, in Figure 9 right side, to observe the trend of AS occupancy capacity.

As expected, the left side of Figure 9 shows that the number of served users increases with the increasing size of the fleet and it ranges from 73,9% (when the fleet is made by 40 AS) to 92% (with a fleet of 70 AS). On the other hand, the number of users served who arrived late at work decreases with the increase of the size of the fleet. It varies from 4,87% (i.e., 36 users) over the served users, for which we registered an average late time of 6,36 minutes, to 0,87% (i.e., 8 users) of late users with an average late time of 2,7 minutes. Instead, the right side of Figure 9 shows how fleets of different size perform in terms of AS occupancy capacity, given the fixed number of 1000 users. We can observe that, in all scenarios, on average the number of cumulative user lifts provided by the AS surpass the number of their occupancy capacity, i.e., 12, ranging from 13 to 19. Of course, the average of cumulative user lifts is higher when the fleet is made by 40 AS, and it slowly decreases when the size of the fleet rises. This basically shows a good usage of the entire fleet. Further, it allows us to predict that increasing the fleet size over 70 might not be convenient, since it can lead to an under-utilisation of the fleet, increasing the number of AS that do not reach their full occupancy capacity. Moreover, as we have seen, a fleet of 70 shuttles served the 92% of 1000 users, which can be considered as a good result. This simulation further shows that a fleet of 60 AS presents as well a good performance in terms of both served users (91.1%) and AS utilization (15 cumulative user lifts on average). The latter will be seen as convenient or not based on the objective of the service provider.

Lastly, the waiting time for users served by the AS, decreases with the increasing size of the fleet, from 4,01 minutes whit a fleet of 40 AS, to 2,13 minutes whit a fleet of 70 AS.

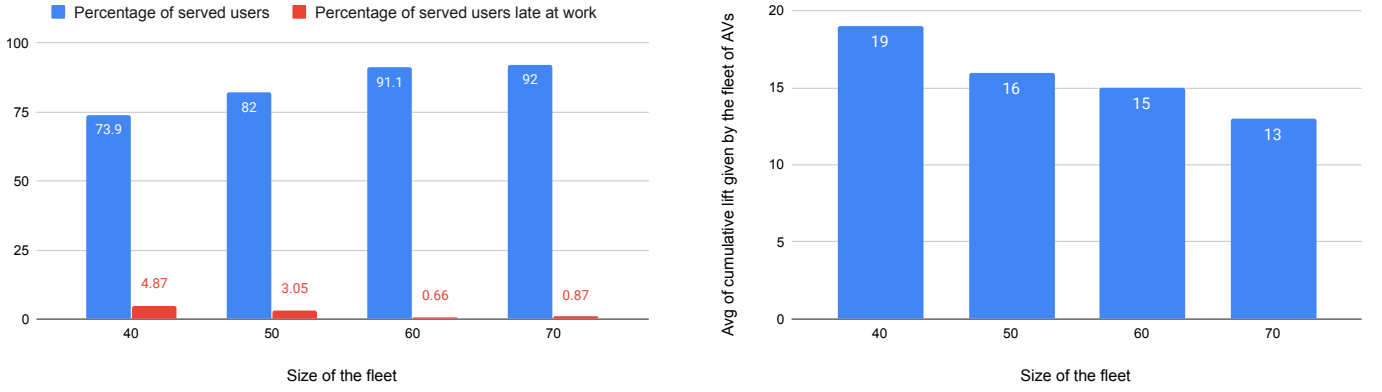At last, to have a complete overview of the cost for the

Fig. 9: Fixed number of 1000 users in scenarios with varying number of AS in the fleet, from 40 to 70. (**Left side**) Percentage of served users over the total users and percentage of users late at work among served users; (**Right side**) Average of cumulative user lifts given by the fleet in the different scenarios with varying number of AS.

service provided by the AS, in Figure 10 we plotted the travel costs for the served users in both the second evaluation with the fixed fleet of 50 AS and varying number of total users (left side) and the third evaluation with the fixed number of 1000 users and varying size of the fleet (right side). The median value is reported in red. We can see that, on average, the cost of the service is stable in all the scenarios studied, which it is of about EURO 2 for the majority of served users. However, there are cases in which the travel is more expensive, reaching peaks of roughly EURO 12 and EURO 20. This can be due to longer distances covered by the AS but also to a low level of occupancy capacity of AS, which leads to the situation where the travel cost is shared among just few users.

a) Threats to validity: Our experimentation may be internally biased from the settings of input parameters, namely the *time interval* for the users to start working, the *cost* per Km, the *angle* between the passengers destinations and the AS's current location used to decide if accepting or not passengers on the road. Variations to these parameters could affect travel time and costs, and the number of passengers being late at work, whereas the overall procedure is not affected. We have chosen input parameters based on real working settings we experience in our daily life, although we are aware that multiple combinations of them are possible. As external threats to validity, we are aware that the application of the approach to other case studies, i.e., different cities, has not been performed. This might lead to observe different results due to, e.g., the diverse dimensions of cities, with different road graphs structure, but we leave this point for our future work.

D. Discussion

From the described evaluations above, we can draw several conclusions. The first evaluation highlights two different perspectives in which the mobility service providing AS fleets can be exploited. The first refers to the perspective of users, where each user decides if adhering or not to service, based on their working place or routines. The second instead, gives the perspective of companies that may decide to adhere to the AS mobility service in order to offer it to their employees.

For instance, a company can decide to support collective mobility for environmental regarding matters. Of course, these two alternatives are not complementary but they can be used simultaneously in order to perform trade-offs. We have performed a preliminary evaluation of these alternatives, and analysed in isolation, and discovered that the scenario in which a consistent amount of users work for the same company (i.e., in the same working place) leads to lower travel costs and to a lower amount of Kms traveled by the AS, at the expense of the users waiting time, which even if tending to increase they also remain within acceptable levels. The second and third evaluations instead, show how the proposed agent-based framework can support the service provider in delivering useful insights on the number of users to be served by a fleet, and the size of a fleet serving a given amount of users that should be adopted. Further simulations allowed us to evaluate up to which extent a fleet can guarantee good performance in terms of served users, while also keeping low the number of users reaching their working place late, and the cumulative user lifts per AS as higher as possible.

All the previous evaluations allow us to conclude with the following thoughts. A fundamental aspect of a smart and sustainable city is to be able to sense the pulse of the city, to perform short and long-term analysis of phenomena and to provide valuable information to decision makers. Simulations are relevant in this context as they support administrators, operators and users in the assessment of how innovations and proposed solutions for mobility systems will meet their needs to help plan for the future. With the framework proposed in this paper [6], we provide a solution that is able to: (1) exploit the data collected by the different simulations, (2) enable a deep analysis of the impact of AS in a city in different scenarios and, (3) help administrators, companies and citizens understand their city and its traffic, and how it reacts when

---

[6] The source of the framework is available for its usage at https://bit.ly/30SdGTa. It contains: (1) the GAMA PROJECT and the MODELS DESCRIPTIONS; these can be further used by the research community to model other transportation modes and simulate their impact on cities. (2) the SCRIPTS to create the Trento city map and for decentralised simulations, and (3) the installation, documentation and usage instructions.
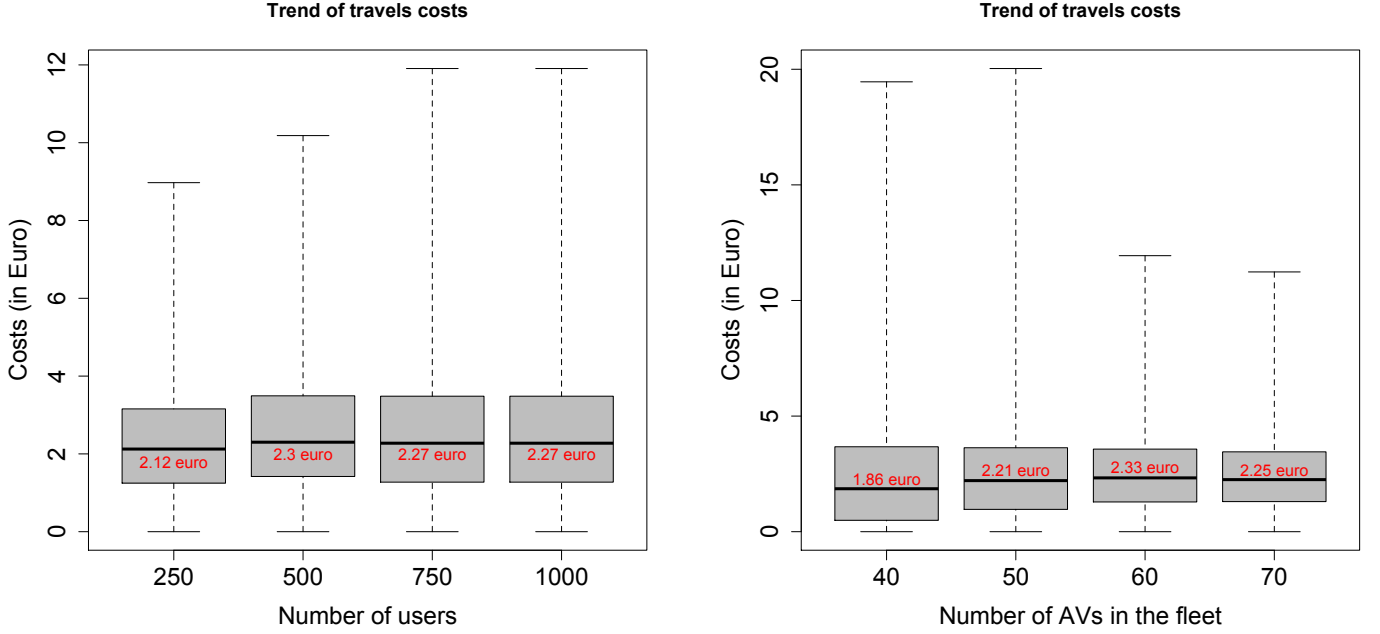
Fig. 10: Trend of travel costs. **(Left side)** Scenario with fixed fleet of 50 AS and varying number of total users; **(Right side)** Fixed number of 1000 users and varying number of AS in the fleet.

novel mobility solutions are taken into account.

VI. Conclusions and Future Work

Planning for traffic patterns that will incorporate intelligent vehicles and deal with the inherent uncertainty is a daunting task. It requires the ability to analyze the effects of systems that do not exist yet, while at the same time committing to the building of transportation infrastructures envisioned to last a long term. Hence, the ability for city planners to plan for these potential futures is critical. The scientific literature has already investigated the potential impacts of dynamic ride-sharing with automated vehicles. Instead, this paper has described a system that enables city mobility planners to explore the integration of AS, and the potential of tackling the uncertainty underlying, using a decentralized control. Expectedly, the integration of AS has significant potential to improve traffic capacity and efficiency of mobility systems, assuming different trade-offs.

As seen in more detail in the experiments, a mix of traditional and intelligent vehicles can also be analyzed using the proposed framework. A final goal is to additionally develop the framework into a further robust, off-the-shelf tool that planners can use to optimize novel technologies in towns.

The insertion of cooperative behaviours among AS have great potential in the framework, to therefore help when AS face unexpected situations such as breakdowns, heavy traffic and congestion. We are experimenting with new cooperative behaviours to face such problems. As such, future directions include the implementation of adaptation strategies on top of those self-organization provided to handle adaptation needs, such as, i) the dynamic rescheduling of the available AS in cases in which there are passengers who cannot longer be reached by the assigned car; ii) the dynamic re-grouping of passengers of a faulty car, among the closest AS around.

The behaviour of systems with AS is emerging and uncertain, and can therefore surprise the stakeholders [44]. Therefore, we also plan to add explanation capabilities about the decision-making and impact of self-organized and adaptive cooperation of AS to the framework [45], [46], [47], [48]. One of the potential benefits of creating this framework is that it allows planners to "score" a configuration along multiple parameters, including cost, traveller waiting time, carbon production, etc. Not only it provides flexibility to the planner, but also provides a sound basis for integrating machine learning into the planning process. Using a Reinforcement Learning (RL) approach [49], the outputs of the simulation can be used as a cost function that can train a system to find local (and potentially global) optima in a complex, multidimensional environment. A next step in this research effort is to integrate RL approaches into the traffic planning process to work according to the different "best" options according to given criteria. A goal is to have a planner outlining a region on a map to select options about the population, and let the RL decision-making calculate the different "best" options according to the criteria. Providing a RL based evaluation allows for the assessment of a broad range of possibilities to be further explored by human planners. For example, human planners would take care of edge cases such as those where passengers reject a vehicle's proposal, handling non-standard cargoes, or providing transportation services that need special coordination with other stakeholders such as parents.

Eventually, since the connection between autonomous vehicle technologies and estimated changes in energy consumption is receiving a great deal of attention [50], our plan it to extend our framework to deal with mobility scenarios including autonomous electric shuttles and simulate different situations with the objective to optimize energy resources in a city.

REFERENCES

[1] G. Lyons and C. Davidson, "Guidance for transport planning and policy making in the face of an uncertain future," *Transportation Research Part A: Policy and Practice*, vol. 88, pp. 104–116, 06 2016.

[2] Google, "Google maps platform," 2020. https://cloud.google.com/maps-platform/.

[3] S. Mariani, G. Cabri, and F. Zambonelli, "Coordination of autonomous vehicles: Taxonomy and survey," *CoRR*, vol. abs/2001.02443, 2020.

[4] M. Jeon, A. Riener, J. Sterkenburg, J. Lee, B. N. Walker, and I. J. Alvarez, "An international survey on automated and electric vehicles: Austria, germany, south korea, and USA," in *9th Int. Conf., DHM*, vol. 10917, pp. 579–587, Springer, 2018.

[5] T. Litman, *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute Victoria, Canada, 2017.

[6] S. Calvert, W. Schakel, and J. Van Lint, "Will automated vehicles negatively impact traffic flow?," *Journal of advanced transportation*, vol. 2017, 2017.

[7] N. Mehr and R. Horowitz, "Can the presence of autonomous vehicles worsen the equilibrium state of traffic networks?," *2018 IEEE Conf. on Decision and Control (CDC)*, pp. 1788–1793, 2018.

[8] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: Disengagements, accidents and reaction times.," *PLoS ONE*, vol. 11(12), 2016.

[9] W. Do, O. M. Rouhani, and L. Miranda-Moreno, "Simulation-based connected and automated vehicle models on highway sections: A literature review," *Journal of Advanced Transportation*, vol. 2019, pp. 1–14, 06 2019.

[10] X.-m. Chen, M. Jin, Y.-s. Miao, and Q. Zhang, "Driving decision-making analysis of car-following for autonomous vehicle under complex urban environment," *Journal of Central South University*, vol. 24, pp. 1476–1482, 06 2017.

[11] R. Lemos, H. Giese, H. Müller, J. Andersson, M. Litoiu, B. Schmerl, G. Tamura, N. Villegas, T. Vogel, D. Weyns, L. Baresi, B. Becker, N. Bencomo, Y. Brun, B. Cukic, R. Desmarais, S. Dustdar, G. Engels, and J. Wuttke, *Software Engineering for Self-Adaptive Systems: A Second Research Roadmap*, pp. 1–32. Springer, 01 2013.

[12] T. A. Litman, "Autonomous vehicle implementation predictions: Implications for transport planning," 2015.

[13] I. Gerostathopoulos and E. Pournaras, "Trapped in traffic? a self-adaptive framework for decentralized traffic optimization," in *2019 IEEE/ACM 14th Int. Symp. on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pp. 32–38, May 2019.

[14] J. Ainsalu, V. Arffman, M. Bellone, M. Ellner, T. Haapamaki, N. Haavisto, E. Josefson, A. Ismailogullari, B. Lee, O. Madland, R. Madzulis, J. Muur, and S, "State of the Art of Automated Buses," *Sustainability*, vol. 10, pp. 1–34, August 2018.

[15] M. Azad, N. Hoseinzadeh, C. Brakewood, C. R. Cherry, and L. D. Han, "Fully Autonomous Buses: A Literature Review and Future Research Directions," *Journal of Advanced Transportation*, pp. 1–16, 2019.

[16] N. H. T. S. Administration *et al.*, "U.s. department of transportation releases 'preparing for the future of transportation: Automated vehicles 3.0'," *US Department of Transportation*, 2018.

[17] J. Bierstedt, A. Gooze, C. Gray, J. Peterman, L. Raykin, and J. Walters, "Effects of next-generation vehicles on travel demand and highway capacity," *FP Think Working Group*, pp. 10–11, 2014.

[18] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations for capitalizing on self-driven vehicles," *Transportation Research*, vol. 20, 2014.

[19] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, "Autonomous vehicles: challenges, opportunities, and future implications for transportation policies," *Journal of modern transportation*, vol. 24, no. 4, pp. 284–303, 2016.

[20] A. Broggi, P. Cerri, S. Debattisti, M. C. Laghi, P. Medici, D. Molinari, M. Panciroli, and A. Prioletti, "Proud—public road urban driverless-car test," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3508–3519, 2015.

[21] S. L. Poczter and L. M. Jankovic, "The google car: driving toward a better future?," *Journal of Business Case Studies (Online)*, vol. 10, no. 1, p. 7, 2014.

[22] M. M. Morando, Q. Tian, L. T. Truong, and H. L. Vu, "Studying the safety impact of autonomous vehicles using simulation-based surrogate safety measures," *Journal of Advanced Transportation*, vol. 2018, 2018.

[23] PTV, *VISSIM 9 - User Manual*, 2016. https://www.scribd.com/document/353480439/Vissim-9-Manual.

[24] D. Gettman, L. Pu, T. Sayed, S. Shelby, and I. Siemens, "Surrogate safety assessment model and validation," tech. rep., United States. Federal Highway Administration., 2008.

[25] A. Buzachis, A. Celesti, A. Galletta, M. Fazio, and M. Villari, "A secure and dependable multi-agent autonomous intersection management (maaim) system leveraging blockchain facilities," in *2018 IEEE/ACM Int. Conf. on Utility and Cloud Computing Companion (UCC Companion)*, pp. 226–231, IEEE, 2018.

[26] M. W. Levin, K. M. Kockelman, S. D. Boyles, and T. Li, "A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application," *Computers, Environment and Urban Systems*, vol. 64, pp. 373–383, 2017.

[27] D. J. Fagnant and K. M. Kockelman, "Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in austin, texas," *Transportation*, vol. 45, no. 1, pp. 143–158, 2018.

[28] D. Sánchez, S. Martínez, and J. Domingo-Ferrer, "Co-utile p2p ridesharing via decentralization and reputation management," *Transportation Research Part C: Emerging Technologies*, vol. 73, pp. 147–166, 2016.

[29] "Sumo - simulation of urban mobility," 2020. https://www.eclipse.org/sumo/.

[30] P. Á. López, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. WieBner, "Microscopic traffic simulation using SUMO," in *21st Int. Conf. on Intelligent Transportation Systems, ITSC 2018*, pp. 2575–2582, 2018.

[31] T. Liebig and M. Sotzny, "On avoiding traffic jams with dynamic self-organizing trip planning," in *13th Int. Conf. on Spatial Information Theory, COSIT 2017*, pp. 17:1–17:12, 2017.

[32] L. Ye and T. Yamamoto, "Modeling connected and autonomous vehicles in heterogeneous traffic flow," *Physica A: Statistical Mechanics and its Applications*, vol. 490, pp. 269–277, 2018.

[33] Q. Lu, T. Tettamanti, D. Hörcher, and I. Varga, "The impact of autonomous vehicles on urban traffic network capacity: an experimental analysis by microscopic traffic simulation," *Transportation Letters*, pp. 1–10, 2019.

[34] A. R. et al., "The future of road transport. eur - scientific and technical research reports." https://www.kowi.de/Portaldata/2/Resources/horizon2020/coop/future-road-transport.pdf. Publications Office of the European Union, 2019.

[35] A. Bucchiarone, S. Battisti, A. Marconi, R. Maldacea, and D. C. Ponce, "Autonomous shuttle-as-a-service (asaas): Challenges, opportunities, and social implications," *CoRR*, vol. abs/2001.09763, 2020.

[36] M. M. Hölzl, N. Koch, M. Puviani, M. Wirsing, and F. Zambonelli, "The ensemble development life cycle and best practices for collective autonomic systems," in *Software Engineering for Collective Autonomic Systems - The ASCENS Approach*, pp. 325–354, Springer, 2015.

[37] F. Zambonelli, N. Bicocchi, G. Cabri, L. Leonardi, and M. Puviani, "On self-adaptation, self-expression, and self-awareness in autonomic service component ensembles," in *Fifth IEEE Conf. on Self-Adaptive and Self-Organizing Systems, SASOW 2011, Workshops Proceedings*, pp. 108–113, 2011.

[38] A. Congiu, *An Approach for Collective Adaptation for Self-Organized Autonomous Vehicles*. Master degree thesis, University of Trento, 2018/2019.

[39] P. Taillandier, "Traffic simulation with the GAMA platform," in *Int. Work. on Agents in Traffic and Transportation*, p. 8 p., 2014.

[40] A. Grignard, P. Taillandier, B. Gaudou, D. A. Vo, N. Q. Huynh, and A. Drogoul, "Gama 1.6: Advancing the art of complex agent-based modeling and simulation," in *PRIMA 2013: Principles and Practice of Multi-Agent Systems*, pp. 117–131, Springer Berlin Heidelberg, 2013.

[41] P. Tranouez, E. Daudé, and P. Langlois, "A multiagent urban traffic simulation," *Journal of Nonlinear Systems and Applications*, vol. 1, pp. 1–9, June 2010.

[42] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model mobil for car-following models," *Transportation Research Record*, vol. 1999, no. 1, pp. 86–94, 2007.

[43] N. Ta, G. Li, T. Zhao, J. Feng, H. Ma, and Z. Gong, "An efficient ridesharing framework for maximizing shared route," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 2, pp. 219–233, 2018.

[44] N. Bencomo and A. Belaggoun, "A world full of surprises: Bayesian theory of surprise to quantify degrees of uncertainty," in *ICSE Companion*, pp. 460–463, 2014.

[45] A. García-Domínguez, N. Bencomo, J. M. P. Ullauri, and L. H. G. Paucar, "Querying and annotating model histories with time-aware patterns," in *22nd ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems, MODELS 2019,*, pp. 194–204, 2019.

[46] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein, "Requirements-Aware Systems: A Research Agenda for RE for Self-adaptive Systems," in *2010 18th IEEE Int. Requirements Engineering Conference*, pp. 95–103, Sept. 2010.

[47] K. Welsh, N. Bencomo, P. Sawyer, and J. Whittle, "Self-explanation in adaptive systems based on runtime goal-based models," *Trans. Computational Collective Intelligence*, vol. 16, pp. 122–145, 2014.

[48] A. Bucchiarone, "Collective adaptation through multi-agents ensembles: The case of smart urban mobility," *TAAS*, vol. 14, no. 2, pp. 6:1–6:28, 2019.

[49] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[50] C. Ross and S. Guhathakurta, "Autonomous vehicles and energy impacts: A scenario analysis," *Energy Procedia*, vol. 143, pp. 47 – 52, 2017.

Nelly Bencomo is a Senior Lecturer in Computer Science at Aston University in the UK (since May 2013). In 2019, she was granted the Leverhulme Fellowship "QuantUn: quantification of uncertainty using Bayesian surprises." Nelly is the principal investigator of the research project Twenty20Insight funded under the EPSRCs to work on Software Engineering, RE, and AI (2020-2023). Before, she was an EU Marie Curie Fellow, from May 2011-May 2013 under a Marie-Curie Fellowship (Grant) Requirements@run.time: Requirements-aware Systems. She was a Senior Researcher at Lancaster University until May 2011 after being was awarded her Ph.D. in Computer Science by Lancaster University in 2008. Nelly exploits the interdisciplinary aspects of software engineering, comprising both technical and human concerns while developing techniques for intelligent, autonomous and highly-distributed systems. With other colleagues, she coined the research topics models@run.time and requirements@run.time. Nelly has actively participated in different European Projects and the EPSRC in the UK in the area of self-adaptive and autonomous systems. She was the program chair of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS) in 2014, and co-program chair of the 12th IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO) in 2018. Nelly is an Associate Editor of IEEE Transactions on Software Engineering (TSE) and a member of the Editorial Board of the Journal of Software and Systems Modeling.

Antonio Bucchiarone is Senior Researcher within the DAS Research Unit at Fondazione Bruno Kessler (FBK) of Trento, Italy. His main research interests include: Self-Adaptive (Collective) Systems, Domain Specific Languages for Socio-Technical System, and AI planning techniques for Automatic and Runtime Service Composition. He received a Ph.D. in Computer Science and Engineering from the IMT School for Advanced Studies Lucca in 2008 and since 2004 he has been a collaborator of Formal Methods and Tools Group at ISTI-CNR of Pisa (Italy). He has been actively involved in various European research projects in the field of Self-Adaptive Systems, Smart Mobility and Constructions and Service-Oriented Computing. He was the general chair of the 12th IEEE International Conference on Self-Adaptive and Self Organizing Systems (SASO 2018) and he is an Associate Editor of the IEEE Transactions on Intelligent Transportation Systems (T-ITS) Journal, the IEEE Software Journal and the IEEE Technology and Society Magazine.

Martina De Sanctis is Assistant Professor at the Computer Science department of the Gran Sasso Science Institute (GSSI), in L'Aquila, Italy, where she has previously been Postdoctoral researcher. Her research interests include behavioral and architectural adaptation of service- and IoT-based systems, collective aspects and modeling of multi-agent systems, dynamic adaptations and its application to several domains, i.e., mobility, smart cities, IoT, eHealth. She received a Ph.D. in Computer Science at the Doctoral School in Information and Communication Technology (2018), from the University of Trento and Fondazione Bruno Kessler (FBK) in Trento. From 2013 to 2018 she was PhD fellow and researcher at FBK, at the Distributed Adaptive Systems research unit where she was working on approaches for the dynamic adaptation of service-based systems, with focus on automated service composition, and their application in different domains. During her PhD studies she actively participated in European Projects in the large-scale collective systems (ICT-FET Proactive project) and digital industry (EIT Digital project) sectors. She has been previously working in companies as software developer in the business sectors of Geographic Information Systems (GIS) and Web-based software applications.