sustainability

MDPI

Article

Dynamic Carpooling in Urban Areas: Design and Experimentation with a Multi-Objective Route Matching Algorithm

Matteo Mallus [1,2,*], Giuseppe Colistra [1,2], Luigi Atzori [1,2,3], Maurizio Murroni [1] and Virginia Pilloni [1]

[1] Dipartimento di Ingegneria Elettrica ed Elettronica (DIEE), University of Cagliari, 09123 Cagliari, Italy; giuseppe.colistra@diee.unica.it (G.C.); l.atzori@diee.unica.it (L.A.); murroni@diee.unica.it (M.M.); virginia.pilloni@diee.unica.it (V.P.)

[2] GreenShare SRL, 09128 Cagliari, Italy; giuseppe.colistra@green-share.it (G.C.); luigi.atzori@green-share.it (L.A.)

[3] Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), Unità di Ricerca di Cagliari, 09123 Cagliari, Italy

* Correspondence: matteo.mallus@green-share.it; Tel.: +39-070-6755902

Academic Editor: Harald Rohracher

Received: 30 December 2016; Accepted: 06 February 2017; Published: 10 February 2017

Abstract: This paper focuses on dynamic carpooling services in urban areas to address the needs of mobility in real-time by proposing a two-fold contribution: a solution with novel features with respect to the current state-of-the-art, which is named CLACSOON and is available on the market; the analysis of the carpooling services performance in the urban area of the city of Cagliari through emulations. Two new features characterize the proposed solution: partial ridesharing, according to which the riders can walk to reach the driver along his/her route when driving to the destination; the possibility to share the ride when the driver has already started the ride by modeling the mobility to reach the driver destination. To analyze which features of the population bring better performance to changing the characteristics of the users, we also conducted emulations. When compared with current solutions, CLACSOON allows for achieving a decrease in the waiting time of around 55% and an increase in the driver and passenger success rates of around 4% and 10%, respectively. Additionally, the proposed features allowed for having an increase in the reduction of the CO_2 emission by more than 10% with respect to the traditional carpooling service.

Keywords: real-time carpooling; smart city; Internet of Things; smart transport

1. Introduction

Vehicular traffic congestion is one of the main problems of most of our cities and towns [1]: it degrades the quality of life, leading to a wide set of social, economic and environmental impacts. It calls for a great effort in studying and deploying innovative and ambitious urban transport modes to reach a less car-dependent life-style, which is one of the main causes of urban traffic congestion.

The particular vehicles used for transport, the source of energy and the infrastructure used to implement the transport play a critical role for the evaluation of the social, environmental and climate impact [2]. Different alternative transport modes have been implemented for reducing air pollution, most of the time based on public transport services, where several options have been proposed and deployed according to the specific configurations of the cities' public transportation infrastructures, e.g., city buses, light rails, trains, subways, ferries. The use of public transport infrastructures is indeed certainly one of the best solutions to face the challenge of vehicular traffic congestion. However, conventional public transport denotes a service that follows fixed routes and schedules; it may not be

available in certain areas, and usually, the distance to the stop locations may be great in sparse areas. Therefore, public transport cannot accommodate all types of mobility needs, which would inevitably be met by the use of personal transportation means. Unfortunately, still, a significant number of people prefer the use of a personal car (or other personal transportation means) over public transportation, who however should be persuaded to change their mobility-style. A report by the U.S. Environmental Protection Agency revealed that light-duty vehicles are the source of nearly 25% of the country's greenhouse gas emissions [3]. Consequently, cutting this significant source of emissions is crucial, and a shift of single-occupancy vehicles towards shared cars could help to address these problems. In this context, carpooling has been a widely-accepted concept to implement Intelligent Transport Systems (ITS) in smart cities and to reduce the gas emissions caused by the use of the personal car. Carpooling (also known as ridesharing) is defined as the sharing of car journeys so that more than one person travels in one car, thus reducing travel costs, such as fuel costs, tolls, etc., but most importantly, from the societal and environmental point of view, reducing air pollution. While the concept of carpooling has existed for decades [4], currently, this service is having a lift thanks to the advancements in the ICT sector. In particular, the wide-spread availability of broadband Internet services allows for the deployment of powerful tools for carpoolers to meet potential companions and reach an agreement on the shared trips [5,6]. Most of the ridesharing systems operating today allow the passenger and drivers to find convenient trip arrangements over the Internet, to support trust building between registered users and to implement billing systems to charge passengers and compensate drivers [7]. These procedures must allow for quick and easy matching of carpoolers' needs, as well as for assisting in establishing itineraries, prices and payment methods [8].

Nowadays, the most widespread implementations of ridesharing services rely on a static approach: the carpoolers post the requests and the offers several hours in advance for a future transportation need, and shared rides have to be arranged before the trip starts. On the other hand, dynamic ridesharing is a relatively new type of carpooling: it is a system where an automated process employed by a ridesharing provider matches up drivers and riders on a very short notice [9], which can range from a few minutes to a few hours before the departure time. Dynamic ridesharing clearly brings several advantages over the static ridesharing approach.

This paper focuses on the challenge of a dynamic ridesharing service in urban areas. The major contributions of this paper are the following:

- A new carpooling platform named CLACSOON is presented [10], which is intended to make simple the interaction of the clacsooners, i.e., the platform users, to find a trip companion and interact during all of the phases of the sharing experience. This platform is currently working in the area of Cagliari, and it is available for the iOS and Android operating systems.
- A novel matching algorithm is proposed, which is a route matching algorithm that has two novel features with respect to the state-of-the-art: partial ridesharing, according to which the riders can walk to reach the driver along his/her route when driving to the destination; the possibility to share the ride when the driver has already started the ride by modeling the mobility to reach the driver destination.
- Due to the impossibility to control the characteristics of the real users, an emulation system is deployed to analyze the key parameters that affect the Quality of Experience (QoE) provided to the users when changing the characteristics of the population. The objective is to have key information on how to drive the creation of the population of users (through marketing operations) to reach the desired usage targets. The performance has been evaluated considering the ridesharing success rate for both driver and passenger, the waiting time and the total system CO_2 saved. The results have been compared with the case for which the novel proposed features were not used, showing significant improvements.

The remaining of the paper is organized as follows: Section 2 presents relevant past works and highlights the novelty of the proposed system; Section 3 describes the implemented platform; Section 4

presents CLACSOON's matching algorithm; Section 5 discusses the experimental results for the case study; conclusions and future work are drawn in last section.

2. Past Works and Introduced Innovation

Nowadays, most of the carpooling services implement a "static" approach: when using such a service, the carpoolers have to post ridesharing requests and offers several hours before their desired departure time, and the shared ride has to be arranged before the trip starts. This approach is shown to be effective for mid/long distance trips, while it is not suitable for short distance trips, which often occur in an urban scenario: in this case, a real-time approach fits better. Dynamic ridesharing is a relatively new type of carpooling: it is a system where an automated process employed by a ridesharing provider matches up drivers and riders on a very short notice [9], which can range from a few minutes to a few hours before departure time. In addition to using communication technologies, dynamic ridesharing systems must establish a procedure that enables travelers to form ridesharing instantaneously [11]. This type of carpooling generally makes use of three recent technological advances: GPS navigation devices to determine a driver's route and arrange the shared ride, smartphones for riders to request a ride from wherever they happen to be and social networks to establish trust and accountability between drivers and passengers. These elements are coordinated by a ridesharing provider, which can match rides using opportune route matching algorithms.

In the following two subsections, we review the past works, and we present the introduced innovation, respectively.

2.1. Past Works

The idea of dynamic ridesharing is not new, and major initiatives have been tried in the past in the field of business, for example by Flinc (www.flinc.org), Carma Carpooling (https://www.gocarma.com/) and Commutr (www.getcommutr.com). Dynamic ridesharing clearly brings several advantages over the static ridesharing approach, especially in terms of flexibility in satisfying the users needs. Because of its potential, also several research efforts have been conducted in the last few years, but the problem of matching ride requests and ride offers at a large scale remains challenging. Several matching agencies tried different approaches, but what constitutes the best procedure is still a matter of debate [8]. The ridesharing matching problem in the literature is often modeled as an optimization problem [8,11]. A commonly-used objective is to minimize the overall travel distances in the optimization problem or considering multiple objectives, including the minimization of the overall travel times, the maximization of the number of ride-matches and the minimization of the system response time. The main technical challenge is the complexity of the optimization problem and the matching process itself, along with the complexity of accurately modeling the carpoolers behavior. On the practical side, one of the main challenges regards the critical mass issue, which is faced by dynamic ridesharing services, in particular, in their startup phase; this problem consists of the difficulty in achieving a critical mass of users in order for the service to find an appropriate mate for the users requests, bringing an adequate matching success rate. This challenge is also related to the QoE perceived by the users, which depends on factors, such as safety, social discomfort and time flexibility. In Xing et al. [12], a ridesharing concept for short-distance travel within metropolitan areas is designed as a multi-agent system to handle spontaneous ridesharing requests of prospective passengers with transport opportunities available on a short call bases. The work in Arnould et al. [13] illustrates WiSafeCa (Wireless Traffic Safety Network Between Cars), a Eureka/Celtic-founded European project that consists of researching and prototyping efficient car-to-car and car-to-infrastructure networking mechanisms striving to reduce accidents and traffic congestion. In the scope of the project, a dynamic ridesharing system was designed, in order to serve real-time transport requests. In Agatz et al. [14] is considered the problem of matching drivers and riders for a dynamic ridesharing scenario, presenting a simulation study based on travel demand data for the city of Atlanta. The matching problem is

described as the minimization of the total system-wide vehicle miles incurred by users and their individual travel costs. The simulation results indicated that the use of sophisticated optimization methods based on a rolling horizon approach substantially improve the performance of ridesharing systems over a greedy matching algorithm. In the definition of their study, an important assumption is that a driver could make only one pickup and one delivery: this constraint makes the problem easier to solve, but it prevents the driver from serving some riders even if they are on his/her desired route. Another important assumption for the study was that a shared ride must be agreed before the start of the driver's trip; moreover, the dynamics of the positions and the speeds of all of the shared vehicles are omitted. The work in Herbawi et al. [15] addresses the dynamic ride-matching problem with time windows, optimizing a multi-criteria objective function. Extending the work proposed by Agatz, they propose a genetic and insertion-based heuristic algorithm for solving the optimization problem, also considering the multiple ride problem (i.e., more than one rider for a single driver). The problem is represented using a maximum-weight bipartite matching model, and the optimization software CPLEX is used to solve it. In Di Febbraro et al. [16], the proposed ridesharing system considers the interactions between drivers, riders and the system manager using a model based on mixed continuous-integer linear programming to maximize the performance of dynamic ridesharing systems. The dynamics of the positions and the speeds of all of the shared vehicles are omitted for simplicity, and it is assumed that users can meet only at a priori fixed delivery stations, such as near bus stops, intersections and the corners of squares. The performance of the proposed model has been analyzed through a simulation based on the modeling framework for Discrete Event Systems (DES). In Mallig et al. [17], the authors describe a former implementation of the agent-based travel demand model mobiTopp, with the aim of realizing a realistic model for ridesharing as an agent-based travel demand model. The model has the limitation that it currently supports only end-to-end ridesharing, i.e., only matching between origin-destination (O/D) zones.

Some works have analyzed the benefits of the proposed carpooling solutions. In Cho [18], the authors present a case study analyzing 12 carpooling services in Europe and the United States and conclude that interpersonal interactions in the service encounter (which depend on the application/service interface) play a significant role in the QoE perceived by the users. In Cici et al. [19], the authors investigate and assess the potentials of ridesharing by developing an algorithm that matches users characterized by similar mobility patterns on the basis of departure time, O/D locations and social distance based on data from popular social networks. The results provide an upper bound to the potential of ridesharing performance, indicating that the decrease in the number of cars in a city can be as high as about 30% when the users are willing to share a ride with friends of friends. In Tsao et al. [20], the authors present the potentials of carpooling for reducing traffic congestion in a hypothetical metropolitan area, assuming a uniform distribution of O/D locations. This model attempts to measure the potentials of ridesharing based on spatial and temporal factors, but assumes that only people who live in common home/work zones would consider sharing a ride with one another. Whereas this study is one of the most comprehensive studies in estimating the ridesharing performance in terms of traffic reduction, it has made important simplifications that have most probably underestimated the achievable results with respect to more realistic deployment scenarios [21].

Sharing a ride can also lead to some side effects: for drivers, making a detour to reach the riders' pick-up and drop-off points could represent a waste of time and money when these points are not close to the driver's route, since that behavior increases the total miles traveled by the driver. This drawback has generally a lower impact when compared to the total savings in CO_2 emissions due to the sharing of the ride, but it points out some fields of improvement for ridesharing systems. For example, this side effect can be mitigated by a carpooling system that evaluates only pickup points on the driver's route.

2.2. Proposed Innovation

As it resulted from the previous review, many works have focused on the optimization problem, but only a few have worked on the modeling of the driver mobility to find better matches. One

common assumption is that a shared ride must be agreed before the starting of the driver's trip, whereas the partial ridesharing (a partial ridesharing [8] happens when the pick-up and drop-off locations are located on the driver's original route, either if their origins and destinations are located on major streets or determined by negotiations) mode is not currently facilitated by matching agencies [8], and to the best of our knowledge, its benefits have not been investigated in the literature yet.

Based on these considerations, the novel carpooling solution for dynamic ridesharing service proposed in this paper and named CLACSOON includes the partial ridesharing mode. In this way, the driver avoids taking a detour whenever possible; therefore, it leads to an increment in the total system-wide CO_2 savings. Clearly, it calls for the riders to walk to reach the driver along his/her route when driving to the destination. Additionally, by introducing the modeling of the position of the driver's vehicle, only the remaining part of the route that a driver has to travel is considered when evaluating the matching. Therefore, this approach enables the possibility for shared rides to be agreed on the fly after the starting of the driver's trip, when a rider happens to be close to the remaining part of a driver's route. This approach leads to an increment in the number of total shared rides.

Another important contribution of this paper is that, to evaluate the impact on the performance of the system changing the population characteristics, an emulation system has been deployed to generate increasing numbers of users that interact with the CLACSOON platform, and extensive trials are implemented to analyze some performance indicators, varying the characteristics of the population in the city of Cagliari (Italy). In particular, the passenger success rate, the driver success rate and the total system-wide CO_2 saved have been evaluated with respect to the characteristics of the population. The results shows that introducing the aforementioned features in a route matching algorithm leads to a substantial performance improvement.

3. The CLACSOON System Architecture

The CLACSOON system has been designed and implemented considering an urban scenario where the aim is to offer a real-time, i.e., dynamic, carpooling service. The objective is to satisfy the needs of users that have an unplanned (or predicable) need of mobility in the city that could not be scheduled in advance. Accordingly, the system architecture needed to implement a service that simplifies and automatizes the provisioning of the carpooling processes, considering also the users QoE and creating an incisive user persuasion strategy. The main functional requirements to develop such a service can be briefly described as follows:

- Accounting: to allow the user to access the service. Each user has a profile where various kinds of information are stored, such as name, age, type of car and received feedback, which are very important to build the reputation level.
- Request and offer insertion: to allow each user to insert an offer or request a ride. Each ride is identified by a departure point, an arrival point, time flexibility parameters and a search radius representing the maximum detour from the scheduled trip.
- Automatic matching: the server dynamically evaluates the possible matching between a ride (either an offer or request) and the sets of complementary rides.
- Matching notification: if a matching is found, the system notifies the users. Each matching notification contains the pick-up point, the drop-off point and the expected driver arrival time. Each user can accept or refuse the notification.

The system has to be used by users in mobility, so the access to the system has to be guaranteed by mobile devices. Accordingly, the design of the system architecture considers this facility, and the front-end layer has been designed for mobile devices, considering the major operating systems. As for the back-end side of the system, it is deployed in the cloud to offer good reliability considering the high number of expected connections and then to provide good availability and capability features [22]. In the implementation of the CLACSOON platform, the technology chosen is Google App Engine and its tools for cloud solutions. Others services of the third parties (e.g., Facebook APIs, Direction APIs) are used to build the proposed service.

As already mentioned, the system follows the mobile-cloud paradigm. Figure 1 shows the major components:

- The mobile client allows the user to access the carpooling service in mobility. Its sensors (e.g., GPS) are used to simplify the access to the service and to enhance the user experience [23]. For all communications toward the server, the JSON format is used.
- The cloud application server is the core of the system. It enables the access of users, processes all requests and offers for rides and calculates the matching between requests and offers.
- The cloud database has the task to store all data useful for the service: user profile, ride offers, ride requests, trips, payments, feedback and other information.
- The Facebook APIs are used to simplify the process of registration by offering a quick and easy service to access the system. Using the Facebook social graph, the aim is to increase the social participation of users.
- A directions provider is used to evaluate the information concerning the route between departure and destination locations chosen by the user for his/her ride. This information includes travel directions, estimated path length, estimated travel time and likely speeds derived from road types.
- The push notification services are used to enable the push notification toward the smartphones. This feature is a milestone to obtain the real-time requirement [24].

The CLACSOON application can be downloaded from the iOS and Android markets.

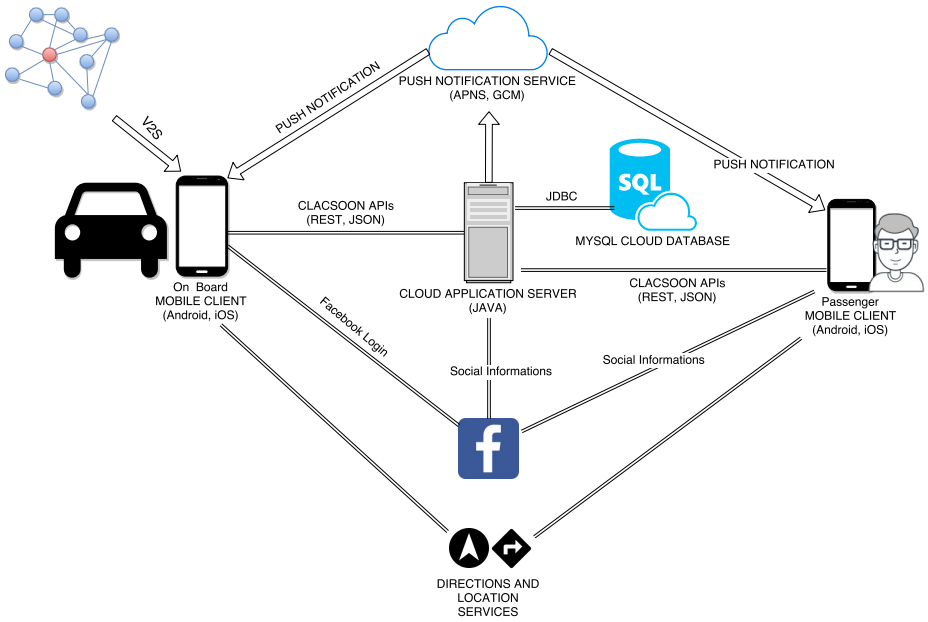

Figure 1. The CLACSOONsystem: a sketch of the functional blocks.

4. The Route Matching Algorithm

This section describes the CLACSOON's route matching algorithm and its ridesharing model. The proposed algorithm contemplates the partial ridesharing mode, i.e., it takes into account the possibility for a rider to reach the driver along his/her route, thus avoiding that the driver takes a detour whenever possible. Furthermore, the algorithm implements a method for estimating the position of the driver's vehicle in an urban context, which enables the possibility for sharing rides after the starting of the driver's trip. During the design of the matching algorithm, we considered a dynamic ridesharing scenario in which a ridesharing provider receives all of the trip announcements for each participant. We also assume that the ridesharing provider relies on the availability of a directions provider, which provides the information concerning the route between a departure and a destination

location. This information includes travel directions, estimated path length, estimated travel time, expected speed derived from road types, which may or may not depend on the historical average speed data over certain time periods. Such a service is provided by many agencies; an example is the Google Maps Directions API [25], which is a service that calculates directions between locations using HTTP requests. Bing Map [26] is another map service provider, which calculates and display directions and routes on the Map with Direction API module or with Bing Map Rest Services. Several alternatives can also be used for those ridesharing providers who opt for a self-hosted direction provider: a great example is The Open Source Routing Machine [27], which is a high performance routing engine written in C++ designed to run on OpenStreetMap data.

Each rider and driver request includes the desired departure and arrival locations. Each ride offer or request includes a timeout, which has to be intended as the maximum time the user is willing to wait before finding a mate. Furthermore, each announcement includes a search radius, which has to be intended as the maximum detour that the driver is willing to make from his/her original route or the maximum distance the rider is willing to walk to reach the pickup point. With this information, the proposed service automatically establishes shared rides over time, matching potential drivers with riders. For the purpose of describing the route matching algorithm, we assume that at a given time t:

- D is the set of drivers;
- P is the set of riders;
- $U = D \cup P$ is the whole population of the dynamic ridesharing system;
- t_d^{DEP}, t_p^{DEP} are the desired departure time for a driver d and a rider p, respectively;
- R_d is the search radius of the driver $d, \forall d \in D$, indicating the maximum detour from his/her scheduled trip that the driver is willing to travel;
- R_p is the search radius of the rider $p, \forall p \in P$, indicating the maximum distance the rider is willing to walk to reach a pickup point.

Furthermore, we assume that:

- $\vec{D}_d, \vec{D}_p$ are the coordinates of the desired departure location for the driver d and rider p, respectively;
- $\vec{A}_d, \vec{A}_p$ are the coordinates of the desired arrival location for the driver d and the rider p, respectively;
- $\vec{\alpha}_d$ is the desired route for a driver d, which connects $\vec{D}_d$ to $\vec{A}_d$. This information is provided by the directions provider and is encoded in a matrix of two columns where each row corresponds to a point in the path;
- τ_d is the estimated travel duration of $\vec{\alpha}_d$, provided by the directions provider;
- $\overline{V_d}$ is the average theoretical speed for $\vec{\alpha}_d$, provided by the directions provider;
- $s_d(t)$ is the number of spare seats for a driver d at the time t, and $s_d(0)$ is the initial vehicle capacity.

The problem of finding the matching between drivers and riders can be formulated as described in the following. The matching algorithm has to satisfy the following constraints:

1. The total number of riders in a vehicle must not exceed the number of spare seats specified by the driver;
2. The entire commuting route must start at the departure and end at the destination locations specified by the driver;
3. Each rider must be picked up before he/she can be dropped off. This constraint seems obvious, but it must be made explicit in a carpool matching algorithm.
4. The maximum distance that a rider p has to walk for reaching the pickup point cannot exceed the search radius R_p;
5. The maximum detour that a driver d has to take with respect to his/her route, for picking up a rider, cannot exceed the search radius R_d;

6. The rider and the driver can wait to find a matched mate for a shared ride at most the timeouts T_p and T_d, respectively.

The constraints from 1 to 3 are usual for a commute process [28], while the constraints from 4 to 6 are specific for the proposed dynamic ridesharing system. As mentioned previously, the potential route of a driver is encoded with a polyline $\vec{\alpha_d}$, which is a matrix with two columns where each row represents the coordinate of each point in the polyline. Accordingly:

$$\vec{\alpha_d} = \cup_{i=0}^{n-1} \vec{\alpha}_{d_i} \tag{1}$$

where i indexes the points in the route and $\vec{\alpha}_{d_0} = \vec{D}_d$ and $\vec{\alpha}_{d_{n-1}} = \vec{A}_d$. The number of points in this matrix (n) is clearly variable and depends on the departure and arrival points, as well as on the route solution proposed by direction providers.

The ridesharing service should be implemented in a way to require the minimal intervention from the users to maximize usability, but at the same time giving him/her the freedom to chose among a possible list of mates. This implies that the ridesharing service finds all the matches and notifies the user with a list of suitable travel companions. The proposed matching algorithm works as follows: the algorithm first searches for one (or more) suitable matching and then, when the matching is found, the arrangement of the shared ride is proposed to the participants. The driver and the rider then can accept or refuse it. In most studies, the objective of the matching algorithm is the maximizing of the system-wide miles saved, the maximizing of the success rate (the percentage of satisfied drivers and riders) or the minimizing of the waiting time of drivers and riders. Clearly, these objectives partially conflict with each other. Depending on the policy of the ridesharing provider, one (or a combination) of the aforementioned objectives is selected for the implementation of the matching algorithm. In our solution, we consider a weighting of the length of the shared trip and needed detour. As already stated, this differentiates with respect to the alternative proposals, as we consider the partial ridesharing mode and the detour of the riders. The proposed route matching algorithm relies on the following three sequential functions that are executed:

- Temporal matching: for each new user (either a rider or driver), the system evaluates whether the a time constraint is satisfied for each possible travel companions, given the timeout T, the driver's travel duration and the current shared rides allocation, but without considering any geographical constraint;
- Geographical matching: this is the evaluation of the matching between a driver and a rider on the basis of the distance from their paths. This step is performed for each pair (d, p) of drivers and riders that satisfied the previous matching. This step also takes into account the theoretical future position of the driver's vehicle, from the beginning till the end of his/her ride.
- Cost function evaluation: this evaluates the cost $C_{d,p}$ for a shared ride between each driver d and rider p that satisfied both the temporal matching and the geographical matching constraints.

The details of these steps will be explained in detail in the following sections.

The list of possible travel companions is then ordered by the value of the cost $C_{d,p}$. This result represents the output of CLACSOON's matching algorithm. This list is then proposed to riders and drivers.

4.1. Temporal Matching

For the temporal matching, it is necessary to consider the effect of the timeout (T_d and T_p), which is the maximum time the user is willing to wait to find a mate, and after this amount of time, the ride request is considered to be expired. For the drivers, it is also important to consider the estimated travel

duration τ_d, as after this amount of time, the ride offer is considered to be over. In case the rider starts the ride after the driver, then the following two conditions must be verified:

$$t_d^{DEP} \leq t_p^{DEP} \leq (t_d^{DEP} + T_d) \tag{2}$$

$$t_d^{DEP} \leq t_p^{DEP} \leq (t_d^{DEP} + \tau_d) \tag{3}$$

which check that the rider arrives before the driver timeout and before the ending of his/her trip.

Differently, in case the driver starts the ride after the rider, the following condition must be verified:

$$t_p^{DEP} \leq (t_d^{DEP}) \leq (t_p^{DEP} + T_p) \tag{4}$$

which check that the driver arrives before the rider timeout.

Each pair (d, r) that satisfies this step is then evaluated in cascade by the geographical matching algorithm.

4.2. Geographical Matching

Each pair (d, p) that satisfies the temporal matching constraints is evaluated by the geographical matching algorithm. For this purpose, we propose a method for estimating the driver's position at the time t on the basis of his/her destination. Modeling the position of the driver's vehicle enables sharing rides even after the starting of the driver's trip. As mentioned before, $\vec{\alpha_d}$ is the desired route for a driver d, which connects the point $\vec{D}_d = \vec{\alpha_{d_0}}$ with the point $\vec{A}_d = \vec{\alpha_{d_{n-1}}}$ (Section 4.1).

Recall that n is the number of segments in the polyline. Assuming that τ_{d_i} is the travel duration between the point i and the point $i+1$, we can decompose the total travel duration τ_d as the sum of the travel duration of each single segment of the route:

$$\tau_d = \sum_{i=0}^{n-1} \tau_{d_i} \tag{5}$$

For simplicity, we can assume that:

$$\tau_{d_i} = \frac{\tau_d}{n}, \forall i \in (0, ..., n-1) \tag{6}$$

We then choose to estimate the driver position at the time $t = t_d^{DEP} + \Delta t$ as the point of the route with index $K_d(t)$:

$$K_d(t) = \lfloor \frac{\Delta t}{\tau_d} \rfloor \cdot (n-1) \qquad \text{if } t_d^{DEP} \leq t \leq (t_d^{DEP} + \tau_d) \tag{7}$$

Given that the constraints discussed in Section 4.1 have just been satisfied, note that this equation has to be considered in the range $t_d^{DEP} \leq t \leq (t_d^{DEP} + \tau_d)$, i.e., the position of the driver is considered to be undefined before the beginning of the ride and after the ride is over.

The remaining part of the path that a driver d has to travel at a time t can then be expressed as:

$$\vec{\beta_d}(t) = \cup_{i=K_d(t)}^{n-1} \vec{\alpha}_{d_i} \tag{8}$$

Accordingly, $\vec{\beta_d}(t)$ is a subset of $\vec{\alpha_d}$, which does not contain the points with index $i < K_d(t)$ that the driver d should have passed by at time t. In other words, $\vec{\beta_d}(t)$ represents the part of the route that theoretically the driver has to travel after the time t. When evaluating the matching at the time t between the offer d and a request p, only the remaining route points $\vec{\beta_d}(t)$ are considered, against the departure $\vec{D}_p$ and destination $\vec{A}_p$ points of the rider: this feature enables drivers to pick up riders on the fly, if the pickup point is close to the remaining route points in $\vec{\beta_d}(t)$. This situation is depicted in Figure 2. For the purpose of describing the geographical matching constraints, we assume that:

- $\Delta dep_{d,p}(t)$ is the minimum distance between the set of route points in $\vec{\beta}_d(t)$ and the rider's departure $\vec{A}_p$;
- $\Delta dst_{d,p}(t)$ is the minimum distance between the set of route points in $\vec{\beta}_d(t)$ and the rider's destination $\vec{D}_p$;
- $\beta_d^{\vec{dep}}(t)$ and $\beta_d^{\vec{dst}}(t)$ are the two points on the driver's route with the minimum distance from $\vec{D}_p$ and $\vec{A}_p$, respectively. These points represent the pick-up points on the driver's route.

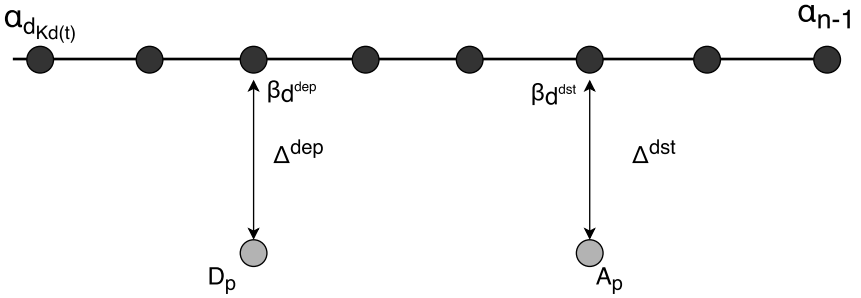

Figure 2. Geographical matching: graphical representation.

A first constraint for the matching to be found is that the index of the point $\beta_d^{\vec{dst}}(t)$ has to be greater than the index of $\beta_d^{\vec{dep}}(t)$, so that the rider must be picked up before he/she can be dropped off.

To take the partial ridesharing mode into account, when the search radius R_p specified by a rider allows him/her to reach a departure pickup point on the driver's route, the system places the pickup point on the point $\beta_d^{\vec{dep}}(t)$. A similar method is used for the evaluation of the destination pickup point $\beta_d^{\vec{dst}}(t)$. In this way, this setting avoids the driver taking a detour when possible, and thus, it is expected that it leads to an increasing of the total system-wide CO_2 savings. Depending on the values of the rider's and the driver's search radius, the matching algorithm assigns the departure pickup point $\vec{P_{dep}}(t)$ and the destination drop-off point $\vec{P_{dest}}(t)$ with the following method:

$$P_{d,p}^{\vec{dep}}(t) = \begin{cases} \beta_d^{\vec{dep}}(t) & 0 \leq \Delta dep_{d,p}(t) \leq R_p \\ \vec{D}_p & R_p < \Delta dep_{d,p}(t) < R_d \\ \uparrow & \Delta dep_{d,p}(t) > R_d \end{cases} \tag{9}$$

$$P_{d,p}^{\vec{dst}}(t) = \begin{cases} \beta_d^{\vec{dest}}(t) & 0 \leq \Delta dst_{d,p}(t) \leq R_p \\ \vec{A}_p & R_p < \Delta dst_{d,p}(t) < R_d \\ \uparrow & \Delta dst_{d,p}(t) > R_d \end{cases} \tag{10}$$

The three conditions in both of the previous equations have been derived from the following motivations:

Condition (1) if the search radius of the rider R_p is greater or equal to the distance $\Delta dep_{d,p}(t)$ between his/her departure and the driver's route, the matching algorithm specifies the location $\beta_d^{\vec{dep}}(t)$ to be the departure pickup point.

Condition (2) if the search radius of the rider R_p is lower than the distance $\Delta dep_{d,p}(t)$, but the search radius of the driver R_d is greater or equal to the distance $\Delta dep_{d,p}(t)$, the pickup point is assigned to be on the rider's departure point D_p.

Condition (3) if both Conditions (1) and (2) are not satisfied, a pickup point does not exist, an then, a matching between p and d does not exist.

An analogous approach is followed in calculating the drop-off point.

The total driver's deviation (for simplicity, we consider the deviation to be in a straight line) from his/her original path can be expressed as:

$$dev_{p,d}(t) = w^D_{d,p} \cdot \Delta dep_{d,p}(t) + w^A_{d,p} \cdot \Delta dst_{d,p}(t) \tag{11}$$

where $w^D_{d,p}$ is a binary variable set to one if $P_{dep} \equiv D_p$, i.e., if a detour from the driver's original path is needed. Likewise, $w^A_{d,p}$ is a binary variable set to one if $P_{dest} \equiv A_p$ and set to zero otherwise. The last constraint to be satisfied is that the total detour that a driver should take in order to reach the pick-up and drop-off points cannot be higher than the distance $\Delta km_{p,d}$ covered by the shared route, i.e., the shared ride provides positive cost savings. If this constraint is not satisfied, there would not be a benefit for the driver to take the detour in order to share the ride.

$$dev_{p,d}(t) \leq \Delta km_{p,d} \tag{12}$$

If both $P_{dest}(t)$ and $P_{dep}(t)$ are defined, the matching is assumed to be found.

4.3. Cost Function

For the pairs of offers and requests (d, p), which satisfy the temporal and geographical constraints, the value of a cost function $C_{d,p}(t)$ for a shared ride is evaluated.

It takes into account the following two elements: $\Delta km_{p,d}$, which is the length of the shared ride; $dev_{p,d}$, which is the length of the needed detour.

The cost function is defined as:

$$C_{d,p}(t) = f(dev_{p,d}(t), \Delta km_{p,d}) = \Theta \cdot dev_{p,d}(t) - \Psi \cdot \Delta km_{p,d} \tag{13}$$

where Θ, Ψ are tuning parameters that respectively determine the importance of the detour from the original path and with respect to the travel sharing. Accordingly, a list of suitable travel companions is ordered, which are associated with:

- the departure pickup point $P_{dep}(t)$
- the destination pickup point $P_{dest}(t)$
- the cost $C_{d,p}(t)$

The list of suitable travel companions for a user represents the output of CLACSOON's matching algorithm. The user is then left with the option to select the best mates according to his/her personal interests. In the next section on the performance evaluation, we assume that the user always selects the mate corresponding to the lowest cost function.

5. Analysis of the Experimental Results

The CLACSOON platform has been implemented, and the relevant service is publicly available for the major mobile operating systems (iOS and Android). Currently, the service is operating at a small scale, and it has attracted three thousand users, mostly located within the area of Cagliari. Since we were facing the critical mass issue, we were interested in analyzing the performance of the system, in relation with the population characteristics. For this purpose we implemented an emulation system, as the current population of CLACSOON users is limited and because we were interested in analyzing the performance with different population characteristics, which cannot be controlled in real scenarios. The place we selected for the emulation scenario is the city of Cagliari, which is an Italian municipality with nearly 150,000 inhabitants, with a metropolitan area (including the surrounding 15 municipalities) of more than 420,000 inhabitants [29]. Considering a real area, we were able to emulate the mobility patterns in real urban conditions, including real roads in the city and real paths between any departure and destination (e.g., pedestrian zone, one-way roads, limited traffic zones). Three Key Performance Indicators (KPI) have been analyzed: the number of shared rides, the waiting

time to find a ride and the average total system-wide CO_2 savings. The following subsections describe the emulation system, present the experimental setup, analyze the achieved performance results and provide a comparison with alternative approaches.

5.1. Description of the Executed Emulations

In this section, we describe the experimental setup and the emulation system. We implemented an agent-based emulator that generates the ride offers and requests on behalf of real uses, evaluates the matching between them and emulates the sharing of rides. The emulator is implemented in Java, and it is based on the core of the CLACSOON platform (indeed, the matching is exactly the service in production, but executed in the emulation environment). In our experiments, we ran several scenarios, each one characterized by a combination of parameter settings as explained in the following. In the following, we describe the processes we followed for the configuration, setup, run and evaluation phases in our experiments.

5.1.1. Configuration

During the configuration step, a list of scenarios is generated: each scenario represents the configuration of a population. During the experiments, we have changed some parameters of the population to evaluate the effects on the KPIs; these parameters are listed in Table 1. The performed experiments have been conducted by selecting an area of interest in the city of Cagliari (centered at: 39.23, 9.14) and with an area (A) of about 64 km^2 , which is where the users can operate. This area is of interest for this study since the majority of CLACSOON'S users mainly operate inside this boundary. Furthermore, this area is representative of medium-small cities with numerous residential areas, commercial sites, factories and historic neighborhoods within its metropolitan boundaries. Figure 3 shows the area selected for this case study where the area of interest is delimited by a black line. Each run lasts for S hours, during which a total of N users act as either passengers or drivers. When evaluating the performance of the system with respect to the spatial clacsooners density, we have varied both the population density (N_k) and the ratio between the number of drivers and the number of passengers, i.e., L_d/L_p. As shown in the table, N ranges from 600 to 2500, which correspond to a different population density given the size of the reference geographical area, and the ratio L_d/L_p ranges from 1/8 to eight. In the performed emulations, we also refer to the timeout T, which ranges from 1 to 30 min. For simplicity, we assume the same timeout T for each rider and each driver.

Table 1. Values of parameters varied during the experiment.

System Parameters		
Time window	S	4 h
Total users	N	from 600 to 2500
Population percentage	N_k	from 10 to 40 users/km^2
Number of drivers Number of passengers	L_d L_p	L_d/L_p from 1/8 to 8
Temporal rate of ride offers Temporal rate of ride requests	f_d f_p	from 10 to 600 users/h
Timeout	T	From 1 min to 30 min
Search radius of passengers	R_p	300 m
Search radius of drivers	R_d	1/10 of the travel length
Cost function tuning parameters	Ψ, Θ	$\Psi/\Theta = 1$

Each scenario represents a combination of the parameters listed in Table 1. To perform the simulations proposed in this case study, we evaluated the KPIs for three sizes of the population N, eight levels for the ratio L_p/L_d and eight values for the timeout T, for a total of 192 scenarios.

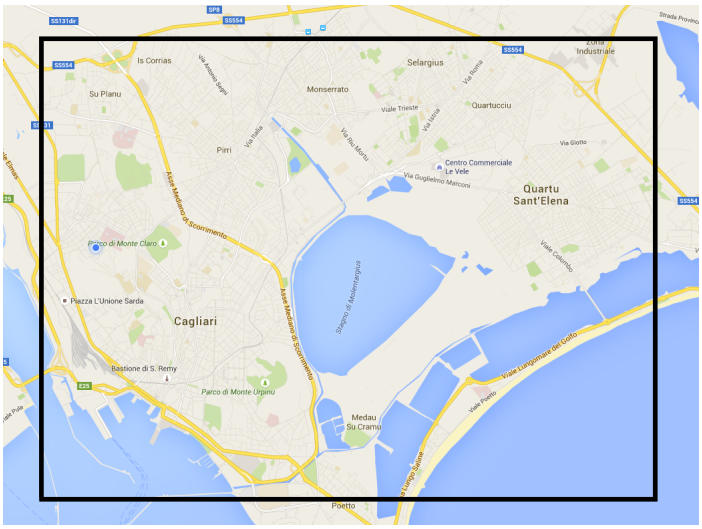

Figure 3. The area for the case study.

5.1.2. Setup

The setup step consists of the generation of each member of the population for a single run. During this step, the total population N is divided into L_d drivers and L_p passengers. The emulator assigns each user departure and destination locations chosen randomly and uniformly in the selected area, with the following two constraints:

(1) both locations fall on a street
(2) it is actually possible to travel from the departure to the destination, i.e., a path exists between these points

Each trip is assigned the shortest path between the departure and destination points, which is calculated by the directions provider. Generating random paths within this area leads to an average travel duration of approximately 13 min with a standard deviation of approximately 6 min (Figure 4). We chose to select randomly and uniformly the starting and the arrival points due to the lack of mobility models for ridesharing users for the city of Cagliari. A similar simplifying assumption has been made in Tsao et al. [20], where the authors, due to the lack of data, assumed a uniform distribution of departure and destination locations in a hypothetical metropolitan area. In Cici et al. [19] and Amey [21] , the authors pointed out that this simplifying assumption should lead to an underestimation of the carpooling performance. Therefore, recent mobility surveys for the city of Cagliari would be needed to assess more accurately the performance of the proposed solution.

The emulator also assigns to each driver and each rider the desired departure time t_u^{DEP}. The time interval between two successive departure times is set to have an exponential distribution within the time window S. If we consider this period and a given number of drivers L_d and of riders L_p, we obtain an expected time interval between offers μ_d and an expected time interval between requests μ_p:

$$\begin{cases} \mu_d = \dfrac{S}{L_d} \\ \mu_p = \dfrac{S}{L_p} \end{cases} \tag{14}$$

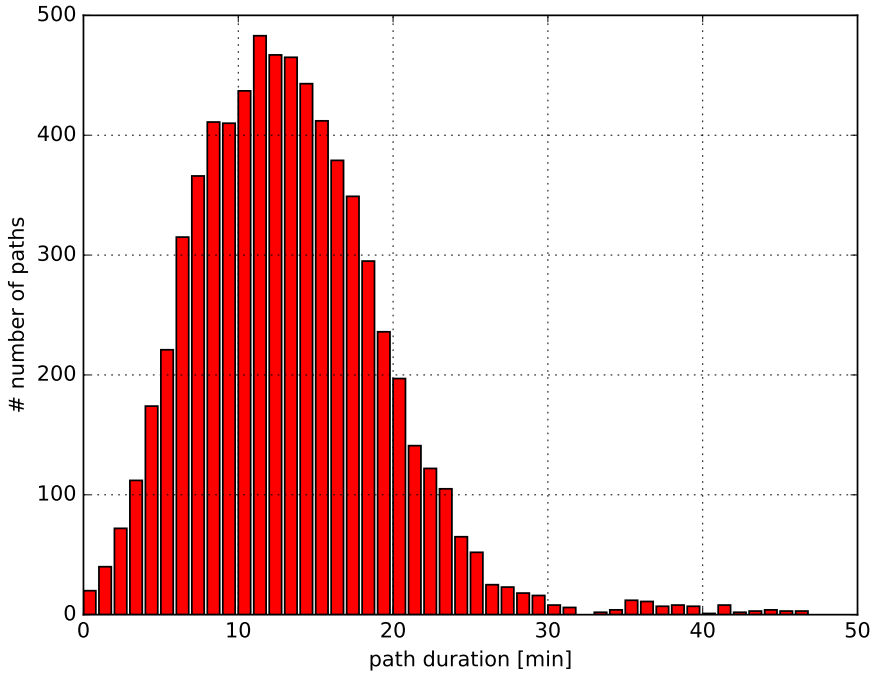

Figure 4. Average travel duration within the selected area.

5.1.3. Run

Once the population's details have been set, the run step is executed. Each run for a given scenario has been repeated for 20 cycles, in order to reduce the width of the confidence interval. In particular, we checked the 95% confidence interval for one of the most important KPI, i.e., the passenger success rate, whose results are shown in Figure 5, and we checked that it was very small, so that we almost had no overlaps among the curves. Specifically, it was lower than 0.01, which was very low. The emulator models a situation in which a user u joins the population at his/her desired departure time t_u^{dep}, simulating the publication of an offer or request through the CLACSOON mobile application. Strictly after the user u joins the population, the matching algorithm is evaluated between this user and the set of complementary users: if the user is a driver, the matching is evaluated against a set of riders, and vice versa. If no matching is found, the user is given a time T to be contacted by another user. When a new rider joins the population, a set of offers that satisfy the constraints specified by the temporal matching is retrieved from the database. These offers are then evaluated by the geographical matching algorithm. If one or more offers satisfy the geographical matching, the value of the cost function is evaluated for these offers, and the ride is agreed upon with the offer, which leads to less cost. Moreover, the number of empty seats for the offer is reduced by one, and the ride request is marked as busy, i.e., it is not possible for other drivers to give a lift for this request. On the other side, when a new driver joins the population, the list of the existing ride requests is retrieved from the database. Those rides have to satisfy the constraint of not being busy, i.e., the ride request must not be committed to another driver. If the aforementioned constraint is satisfied, then the matching between the offer and the request is evaluated by the temporal matching and the geographical matching algorithm. If a matching is found, the shared ride is considered to be agreed upon; the ride request is marked as busy; and the spare seats for the ride offer are decremented by one.

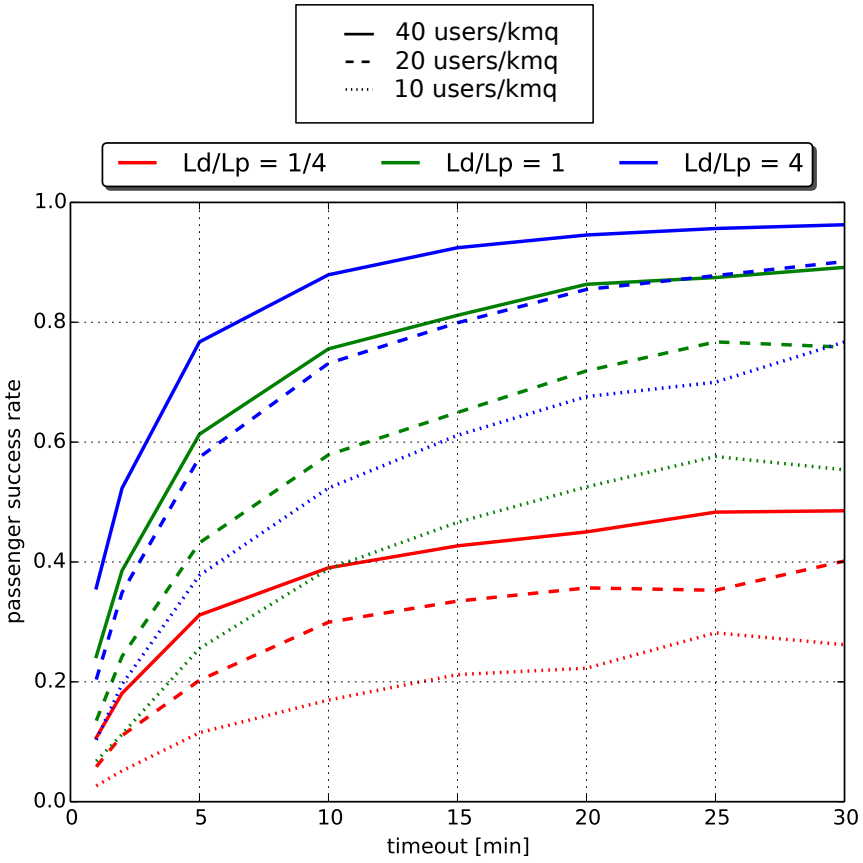

Figure 5. Passenger success rate.

5.1.4. Evaluation

After the end of a run, the following KPIs are computed:

- Passenger waiting time: the average of the time that the riders that found a match had to wait before finding that match
- Passenger success rate: the percentage of riders that found a ride
- Driver success rate: the percentage of drivers that shared a ride
- Total system-wide CO_2 saved: the sum of the estimation of the CO_2 saved for each shared ride

Like the cost function (13) defined in the Section 4.3, the estimation of the total system-wide CO_2 saved is based on the following two elements: (i) the length of the shared ride, i.e., the distance that the riders should have traveled alone if the shared ride was not agreed upon; (ii) the length of the needed detour that the drivers had to take to reach the pick-up and drop-off point. Therefore, this estimation takes into account that drivers could drive longer distances to pick up the riders. The results of the emulations from this case study have been computed as an average of the KPIs obtained from 20 runs for each scenario. As mentioned before in Section 5.1.1, a total number of 192 scenarios has been executed, leading to a total number of about 4000 runs.

5.2. Experimental Results

The performance in relation with the time distribution of the service utilization has been evaluated varying the rate of ride offers (f_d) and the rate of ride requests (f_p). The performance has also been evaluated in relation with the maximum waiting time of the users T.

Figures 5 to 8 show the results according to the mentioned KPIs varying the parameters with the values indicated in Table 1.

5.2.1. Passenger Success Rate

Figure 5 shows the passenger success rate, which is the percentage of passengers that find a ride. This chart shows the trend of this indicator in relation with the timeout, for three levels of the ratio L_d/L_p and three levels of the population N. The first highlighted trend is that the success rate increases with the population, which is something that is expected, since if the spatial density of users is low, the probability to have a matching is small, as well. The growth in relation with the timeout is significant until the value of T is around 15 min; after this value, any further increase in the timeout does not have a big impact. This is due to the fact that the random paths generated in the selected area result in an average travel length of 13 min. This value of T is comparable to the bus transit frequency within the city of Cagliari. If we consider a threshold in the success rate of 80%, we see that this can be achieved with a balance between the numbers of drivers and of passengers (i.e., $L_d/L_p = 1$) if the latter have the patience to wait for up to 13 min in the case that the clacsooner density is 40 users/km^2. Otherwise, if the percentage of drivers is high, having a ratio $L_d/L_p = 4$, then the passengers would have to wait only 6 min. This result tells us that depending on the patience of the customers, a different marketing action should be followed to reach the needed percentage of drivers in the clacsooner population.

5.2.2. Driver Success Rate

Figure 6 shows the driver success rate. The first highlighted trend is that the success rate increases when the population increase and when the ratio L_d/L_p decreases (i.e., the greater the number of riders request a ride, the higher the driver success rate).

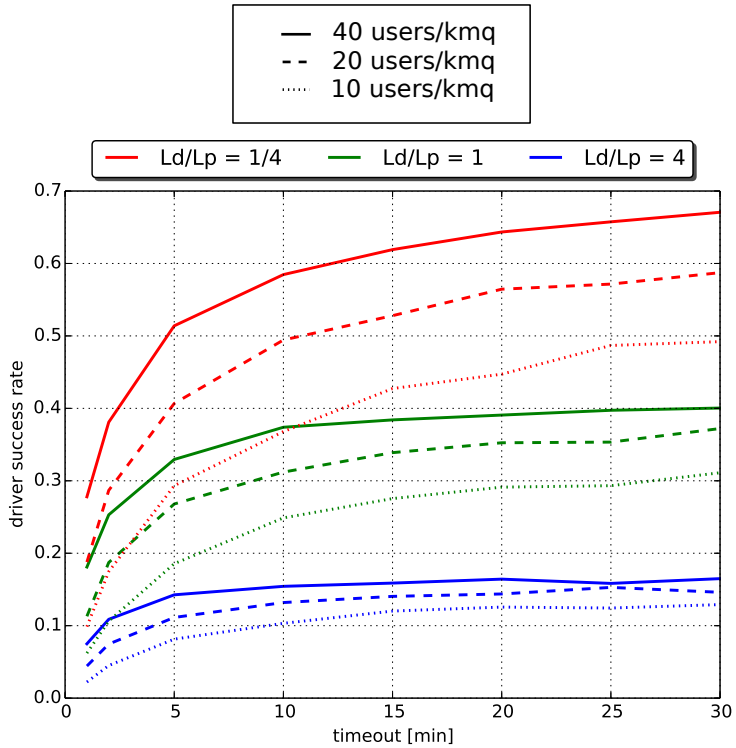

Figure 6. Driver success rate.

Figures 5 and 6 show that, with the same population level, the rider success rate is higher than the driver success rate. This difference is related to the different nature of these two agents: a single driver could give a ride to more than one passenger. This situation is more likely to happen if the number of drivers is higher than the number of riders and when the number of users is high. Moreover, an increase in the timeout always leads to an increase in the success rate for riders, but for drivers,

this effect is limited: the driver success rate increases slowly after 15 min. This is due to the fact that the random paths generated in the selected area result in an average travel length of 13 min. The emulator models a situation in which riders insert their trip when they arrive in the desired pickup point, and then, they can wait for a match at most until their waiting time reaches the timeout. Drivers, as opposed to riders, insert their trip when they are ready to start the trip, and they could obtain a match at most until their trip is over. Therefore, a further increase in the timeout does not lead to a big benefit for drivers. Moreover, when L_d becomes higher than L_p and for a high level of population, there is another important trend. The system results in being unbalanced in favor of the riders, so most of them can find a ride. The remaining drivers have a low probability to find a passenger, because the majority of passengers have just agreed to take a ride from a driver.

5.2.3. Passenger Waiting Time

Figure 7 shows the rider's average waiting time needed to find a ride. The trend is linear and decreases when the number of drivers increases. If the ratio L_d/L_p is high (i.e., more drivers than riders are in the system), the waiting time is low and vice versa. In case there are more drivers than riders and then there are many offers, the probability to find a ride quickly becomes high. Therefore, if the number of drivers is higher than or equal to the number of passengers, the waiting time increases slowly.

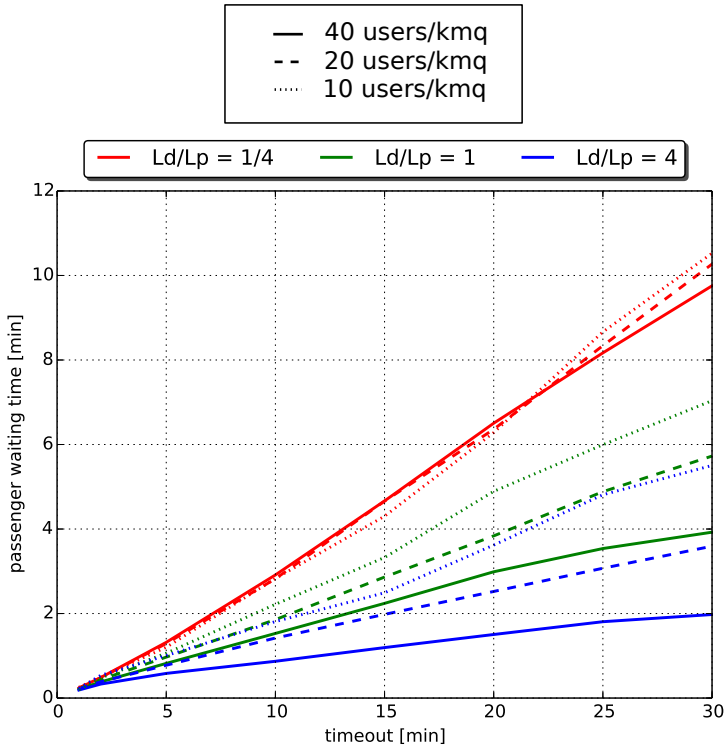

Figure 7. Passenger waiting time.

This figure does not consider the waiting time of rides that have not found a ride (in fact, a waiting time for these rides is undefined), so this figure has to be considered in conjunction with Figure 5.

5.2.4. Total System-Wide CO_2 Saved

The average success rate and the waiting time represent the performance from the single trip point of view. By collecting the travel length of every shared ride, we are able to compute the global system CO_2 saved. The result is shown in Figure 8. This chart shows the trend of this indicator in

relation to the the ratio L_d/L_p, for three levels of the population N and for three levels of the timeout T. It is important to note that the CO_2 saved is a KPI that describes the performance of the whole system.

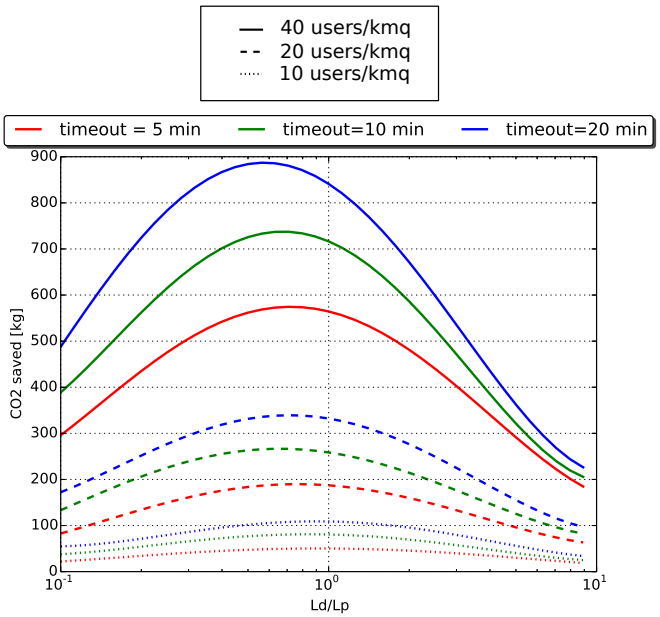

Figure 8. Total system-wide CO_2 savings.

The CO_2 saved is estimated as the product between the total travel shared (evaluated in km) and the average CO_2 emitted by a car (140 g of CO_2 per km) [30]. Since this parameter is assumed to be proportional to the shared travel length, it is also representative of the total cost savings generated by the carpooling system. The curve reaches the maximum value when the number of riders is close to the number of drivers, that is when the system is balanced. It increases when the timeout increases, as well as when the population increases.

Assuming the aforementioned value of CO_2 emitted per km, we can compute the value of emission savings in the time window S used in the simulation. It is notable that with a timeout value of 10 min and for $L_p \approx L_d$, the emission savings in this scenario are about 80 kg for 10 users/km^2, 250 kg for 20 users/km^2 and 720 kg for 40 users/km^2. It is clear that the trend is not linear, but follows an exponential increase with respect to the population.

5.3. Performance Comparison

Following the approach done in [11], to assess the value of CLACSOON's matching algorithm, in this section, we compare its performance with those of an alternative matching algorithm we developed (named "DUMMY"), which presents the two following simplifications with respect to CLACSOON's matching algorithm:

- when a match is found, the pickup and the drop-off points are assigned to be on the rider's desired departure ($\vec{D}_p$) and destination ($\vec{A}_p$) points, respectively.
- a driver cannot accept requests after his/her trip started: each shared ride must be agreed upon before the starting of the driver's trip, and the dynamics of the positions and the speeds of the vehicles are not taken into account

Note that the DUMMY algorithm not only contemplates the identical ridesharing (i.e., when the departure has to be the same for riders and drivers). Indeed, the DUMMY matching algorithm also contemplates the presence of intermediate meeting points that can: (i) be on the original route of the driver; (ii) not be on the way of an original route of the driver, so that a detour would be

needed to reach the pick-up and drop-off points. The difference with the CLACSOON algorithm is that, when rider's desired departure and destination locations are not on the way of the driver's original route, the driver should have to take a detour to reach the rider's departure and destination points. Therefore, the DUMMY matching algorithm does not contemplate the partial ridesharing mode and prevents drivers from picking up riders on the fly, even if the pickup points result in the driver's route, since the shared ride has to be arranged before the starting of the driver's trip. With this comparison, we intend to specifically evaluate the major two novel features introduced by our proposal. The following results shows the comparison of the indicators for a population density of 40 *users*/km^2. In the following, if not explicitly stated otherwise, the simulation environment parameters are equal to those listed in Table 1. Figure 9 clearly demonstrates that the CLACSOON matching algorithm performed better than the DUMMY matching algorithm in terms of all of the KPIs we computed. Figures 9a–c show the computed results in relation with the timeout T and for $L_d/L_p = 1$, i.e., the number of riders is set to be equal to the number of drivers. For instance, for a timeout of 10 min, the CLACSOON matching algorithm leads to a decrease in the waiting time of around 55% and an increase in the driver and passenger success rates of around 4% and 10%, respectively. However, note that the relative advantage for the success rate decreases with the timeout for both drivers and riders. Figure 9d shows the total system-wide CO_2 saved for a timeout of 10 min, in relation with the ratio L_d/L_p. It is notable that, for $L_p = L_d$, the CLACSOON algorithm leads to an increase (+21%) of the CO_2 saved, which corresponds to 120 kg of CO_2 emission savings over the value computed for the DUMMY matching algorithm. For the selected scenario, the overall CO_2 that riders would have emitted if each one had driven his/her car (estimated with respect to the total length of their desired route) is estimated to be approximately 1150 kg, so a participation rate of 40 *users*/km^2 leads to the 64% of emission savings for the CLACSOON matching algorithm and the 54% of emission savings for the DUMMY algorithm.

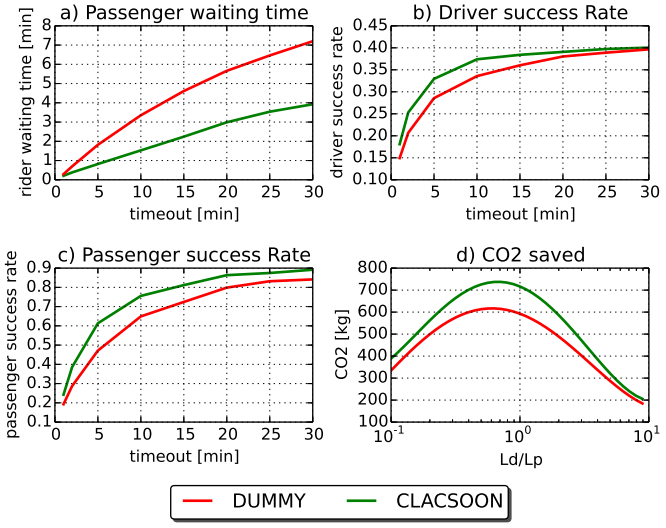

Figure 9. Comparison with the "DUMMY" matching algorithm.

6. Conclusions and Future Works

In this work, we presented the CLACSOON platform that introduces some novel features with respected to the state-of-the-art. This platform has been implemented and is working mostly in the area of Cagliari. We introduced an important novel functionality according to which the route matching algorithm contemplates the partial ridesharing mode, i.e., riders can walk to reach the driver along his/her route when driving to the destination, resulting in higher total system-wide CO_2 savings. Moreover, we introduced the novel functionality according to which the matching algorithm models the position of the driver's vehicle in an urban context so that shared rides can be agreed upon

after the starting of the driver's trip. An emulation system has also been implemented to analyze the performances of the proposed matching algorithm in a simulated smart urban scenario, with respect to the characteristics of the users. The performance has been evaluated considering some key parameters that affect the Quality of Experience (QoE) provided to the users, i.e., ridesharing success rate for both driver and passenger and waiting time. We have also analyzed the total system CO_2 saved. An interesting result consists of the fact that the system shows the best level of performance, i.e., the maximum of the total system-wide CO_2 saved, when the system is balanced, i.e., when the number of drivers is near or equal to the number of riders. The results have also been compared with the case that the novel proposed features were not used. We observed that with CLACSOON, in the case that the users were keen on waiting till 10 min from the service request, it is possible to achieve a decrease in the waiting time of around 55% and an increase in the driver and passenger success rates of around 4% and 10%, respectively. Additionally, the proposed features allowed for having an increase in the reduction of the CO_2 emission of more than 10% with respect to the traditional carpooling service.

The results presented in this paper can be considered to evaluate the requirements to build a successful urban carpooling service. Future works will be focused on the use of even more real scenarios. One aspect to be considered is the generation of trips according to mobility surveys or studies that analyze the travel demand. These data would be needed to further validate the proposed analysis.

Acknowledgments: This work has been partially supported by the project Carpooling for Green Communities (COOG-IT), P.O.R. FESR 2007-2013 Regione Sardegna—Asse VI Competitivita—Linea di attivita 6.2.2 e 6.2.3, CUP E28C14000090007 and by the project FollowMe, Regione Sardegna L.R. 7.8.2007, n. 7., CUP F22I13000260009.

Author Contributions: Matteo Mallus has devised the overall system, implemented the application and conducted the experiments. Giuseppe Colistra has significantly contributed to the design of the system. The other authors have contributed equally to the supervision to all the activities of this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. EU Commission. *Green Paper, Towards a New Culture for Urban Mobility*; European Union: Brussels, Belgium, 2007.
2. Mihyeon Jeon, C.; Amekudzi, A. Addressing sustainability in transportation systems: Definitions, indicators, and metrics. *J. Infrastruct. Syst.* **2005**, *11*, 31–50.
3. United States Environmental Protection Agency. *US Transportation Sector Greenhouse Gas Emissions: 1990—2011, EPA-420-F-13-033a* ; United States Environmental Protection Agency, Office of Transportation and Air Quality, Washington, DC, USA, 2013.
4. Levofsky, A.; Greenberg, A. Organized dynamic ride sharing: The potential environmental benefits and the opportunity for advancing the concept. In Proceedings of the Transportation Research Board 2001 Annual Meeting, Washington, DC, USA, 7–11 January 2001.
5. Blablacar. Blablacar Ridesharing. Available online: https://www.blablacar.it/ (accessed on 30 July 2015).
6. CUTR. Ridematching Software—CUTR Center for Urban Transportation Research. 2015. Available online: http://www.nctr.usf.edu/programs/clearinghouse/ridematching-software/ (accessed on 30 July 2015).
7. Ferreira, J.; Trigo, P.; Filipe, P. Collaborative car pooling system. *World Acad. Sci. Eng. Technol.* **2009**, *54*, 721–725.
8. Furuhata, M.; Dessouky, M.; Ordóñez, F.; Brunet, M.E.; Wang, X.; Koenig, S. Ridesharing: The state-of-the-art and future directions. *Transp. Res. B Methodol.* **2013**, *57*, 28–46.
9. Agatz, N.; Erera, A.; Savelsbergh, M.; Wang, X. *The value of optimization in dynamic ridesharing: A simulation study in metro Atlanta*; Research paper, Report No. ERS-2010-034-LIS; Erasmus Research Institute of Management (ERIM): Rotterdam, The Netherlands, 2010.
10. CLACSOON. CLACSOON Urban Real-Time Carpooling. Available online: http://www.clacsoon.com (accessed on 30 July 2015).
11. Agatz, N.; Erera, A.; Savelsbergh, M.; Wang, X. Optimization for dynamic ridesharing: A review. *Eur. J. Oper. Res.* **2012**, *223*, 295–303.

12. Xing, X.; Warden, T.; Nicolai, T.; Herzog, O. Smize: A spontaneous ridesharing system for individual urban transit. In *German Conference on Multiagent System Technologies*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 165–176.
13. Arnould, G.; Khadraoui, D.; Armendáriz, M.; Burguillo, J.C.; Peleteiro, A. A transport based clearing system for dynamic carpooling business services. In Proceedings of the 2011 11th International Conference on ITS Telecommunications, Sankt-Peterburg, Russia, 23–25 August 2011; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2011; pp. 527–533.
14. Agatz, N.A.; Erera, A.L.; Savelsbergh, M.W.; Wang, X. Dynamic ridesharing: A simulation study in metro Atlanta. *Transp. Res. B Methodol.* **2011**, *45*, 1450–1464.
15. Herbawi, W.M.; Weber, M. A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows. In Proceedings of the 14th Annual Conference on Genetic And Evolutionary Computation, Philadelphia, PA, USA, 7–11 July 2012; ACM: New York, NY, USA, 2012; pp. 385–392.
16. Di Febbraro, A.; Gattorna, E.; Sacco, N. Optimization of dynamic ridesharing systems. *Transp. Res. Rec. J. Transp. Res. Board* **2013**, *2359*, 44–50.
17. Mallig, N.; Vortisch, P. Modeling Car Passenger Trips in mobiTopp. *Procedia Comput. Sci.* **2015**, *52*, 938–943.
18. Cho, E.J. Interpersonal interaction for pleasurable service experience. In Proceedings of the 2011 Conference on Designing Pleasurable Products and Interfaces, Milano, Italy, 22–25 June 2011; ACM: New York, NY, USA, 2011; p. 68.
19. Cici, B.; Markopoulou, A.; Frias-Martinez, E.; Laoutaris, N. Assessing the potential of ridesharing using mobile and social data: A tale of four cities. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, Seattle, WA, USA, 13–17 September 2014; ACM: New York, NY, USA, 2014; pp. 201–211.
20. Tsao, H.S.J.; Lin, D.J. *Spatial and temporal factors in estimating the potential of ridesharing for demand reduction*; California Partners for Advanced Transit and Highways (PATH) research report; University of California, Berkeley: Berkeley, California, 1999.
21. Amey, A. A proposed methodology for estimating rideshare viability within an organization, applied to the mit community. In *TRB Annual Meeting Procediings*; Transportation Research Board, The National Academies: Washington, DC, USA, 2011 ; pp. 1–16.
22. Fernando, N.; Loke, S.W.; Rahayu, W. Mobile cloud computing: A survey. *Future Gener. Comput. Syst.* **2013**, *29*, 84–106.
23. Chen, G.; Kotz, D.; others. *A Survey of Context-Aware Mobile Computing Research*; Technical Report, Technical Report TR2000-381; Department of Computer Science, Dartmouth College, Hanover, NH, USA, 2000.
24. Podnar, I.; Hauswirth, M.; Jazayeri, M. Mobile push: Delivering content to mobile users. In Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops, Vienna, Austria, 2–5 July 2002; IEEE: Piscataway, NJ, USA, 2002; pp. 563–568.
25. Developers, G. Google Directions API. 2016. Available online: https://developers.google.com/maps/documentation/directions/ (accessed on 30 November 2016).
26. Microsoft. Bing Maps for Enterprise. 2016. Available online:https://www.microsoft.com/maps/choose-your-bing-maps-API.aspx (accessed on 30 November 2016).
27. Luxen, D.; Vetter, C. Real-time routing with OpenStreetMap data. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Chicago, IL, USA, 1–4 November 2011; ACM: New York, NY, USA, 2011; pp. 513–516.
28. Xia, J. A New Model for a Carpool Matching Service. *PLoS ONE* **2015**, *10*, e0129257.
29. Wikipedia. Cagliari — Wikipedia, The Free Encyclopedia. 2015. Available online: https://en.wikipedia.org/w/index.php?title=Cagliari (accessed on 30 July 2015).
30. An, F.; Sauer, A. *Comparison of Passenger Vehicle Fuel Economy and Greenhouse Gas Emission Standards around the World*; Pew Center on Global Climate Change, Arlington, VA, USA, 2004; Volume 25.