Meeting points in ridesharing: A privacy-preserving approach

Ulrich Matchi Aïvodji, Sébastien Gambs, Marie-José Huguet, Marc-Olivier Killijian

Meeting Points in Ridesharing: a Privacy-Preserving Approach

Ulrich Matchi Aïvodji[a], Sébastien Gambs[b], Marie-José Huguet[c], Marc-Olivier Killijian[a]

[a] *LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France*
[b] *Université du Québec à Montréal, Montréal, Canada*
[c] *LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France*

Abstract

Nowadays, problems of congestion in urban areas due to the massive usage of cars, last-minute travel needs and progress in information and communication technologies have fostered the rise of new transportation modes such as ridesharing. In a ridesharing service, a car owner shares empty seats of his car with other travelers. Recent ridesharing approaches help to identify interesting meeting points to improve the efficiency of the ridesharing service (*i.e.*, the best pick-up and drop-off points so that the travel cost is competitive for both driver and rider). In particular, ridesharing services, such as Blablacar or Carma, have become a good mobility alternative for users in their daily life. However, this success has come at the cost of user privacy. Indeed in current's ridesharing services, users are not in control of their own data and have to trust the ridesharing operators with the management of their data.

In this paper, we aim at developing a privacy-preserving service to compute meeting points in ridesharing, such that each user remains in control of his location data. More precisely, we propose a decentralized architecture that provides strong security and privacy guarantees without sacrificing the usability of ridesharing services. In particular, our approach protects the privacy of location data of users. Following the privacy-by-design principle, we have integrated existing privacy enhancing technologies and multimodal shortest path algorithms to privately compute mutually interesting meeting points for both drivers and riders in ridesharing. In addition, we have built a prototype implementation of the proposed approach. The experiments, conducted on a real transportation network, have demonstrated that it is possible to reach a trade-off in which both the privacy and utility levels are satisfactory.

Keywords: Dynamic Ridesharing, Privacy Enhancing Technologies, Multimodal Shortest Path, Secure Multiparty Computation, Private Set Intersection.

1. Introduction

The ubiquitous world in which we live has fostered the development of technologies that take into account the context in which users are using them to deliver high quality services. For example, by providing personalized services based on the positions of users, *Location-Based Services* (LBS) Artigues et al. (2012); Quercia et al. (2010) have encouraged the emergence of ridesharing services.

Email addresses: umaivodj@laas.fr (Ulrich Matchi Aïvodji), gambs.sebastien@uqam.ca (Sébastien Gambs), huguet@laas.fr (Marie-José Huguet), killijian@laas.fr (Marc-Olivier Killijian)

In classical ridesharing services, the most studied problem is to assign riders to drivers given various constraints or objectives Furuhata et al. (2013); Agatz et al. (2012). However, as stated in Agatz et al. (2012); Stiglic et al. (2015), another problem arises when considering meeting points between riders and drivers to satisfy more assignments or to produce better solutions with respect to some objectives including the reduction of the travel time spent by the rider, the partial use of the public transportation system or simply the walking distance before reaching the pick-up location. Instead of putting the burden of finding the meeting points on the users, the system itself should be able to identify the optimal meeting points to make the trip as short as possible for both participants.

Recent academic research Bit-Monnot et al. (2013) has considered this issue for multimodal transportation networks and proposed an exact method for automatically solving it on a centralized solver in polynomial time. In Varone and Aissat (2015), some heuristic approaches are also proposed, that aim at reducing the computation time, by limiting the set of meeting points on the rider itinerary.

The main limitation of the current ridesharing systems is related to the centralized nature of their infrastructures. Indeed, most (if not all) existing ridesharing services rely on centralized platforms in charge of collecting and storing potentially sensitive data from ridesharing users, such as their positions and identities, to compute their itineraries. This situation could lead to important damages to their privacy Furuhata et al. (2013); Cottrill and Thakuriah (2015). For instance, a malicious person, hereafter referred to as the *adversary*, can use the location data stored on the centralized platform to cause a privacy breach Gambs et al. (2011). In particular, this adversary can learn the Points Of Interests (POIs) of ridesharing users, compute their mobility models to infer their future movements or even de-anonymize them in another location dataset Gambs, Killijian and del Prado Cortez (2014). Centralized platforms are exposed to the single point of failure problem (*i.e,* a privacy breach may lead to the violation of the privacy of nearly all system's users). We believe that it is of paramount importance to explore alternative approaches to securely implement ridesharing services while preserving the privacy of users.

The ridesharing problem can be summarized as follows. We consider a driver and a rider, each having their own origin and a destination (see Figure 1).

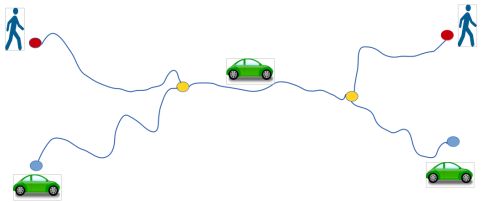

Figure 1: Meeting points for two users in a ridesharing system.

The driver is looking for a rider to pick-up and is willing to make a small detour while the rider is looking for an itinerary in which he may use ridesharing as part of his journey. The participants expect to get a response to their request as soon as possible. The objective of the routing component of the ridesharing platform is to find for both of them a pick-up and drop-off locations and optimal itineraries for their journeys. The optimization function considered is the minimization of the arrival times of both users (*i.e.*, the sum of their arrival times).

More precisely, in this work we focus on casual, one-time and irregular ridesharing scenarios.

Irregular ridesharing scenarios differ from regular ones by the fact that in regular ridesharing a trust is supposed to be already established between riders and drivers. Furthermore, in our scenario, we expect an almost real-time response for a better user experience. In this context, the objective of our approach is to help ridesharing users to synchronize their itineraries without having to reveal the origins and destinations of their journeys.

Therefore, we propose a distributed architecture for privacy-preserving computation of interesting meeting points for drivers and riders participating to a ridesharing system. More precisely, we show that designing multimodal routing algorithms with privacy in mind can greatly enhance the privacy in ridesharing services. In particular, we follow the *privacy-by-design* principle Cavoukian (2009), which recommends to integrate privacy as soon as the design phase of the system (*e.g.*, system structure, hardware design, data processing, applications, etc.) instead of adding *a posteriori* mechanisms to enhance privacy once the system is already built and deployed. To the best of our knowledge, our work is the first that uses private set intersection conjointly with multimodal routing algorithms to design usable privacy-protection mechanisms for ridesharing.

The remainder of this paper is organized as follows. In Section 2, we provide an overview of the related literature. In Section 3, we introduce tools related to both cryptography and graph theory that have been used in our approach. In Section 4, we detail how the system works. In Section 5 and 6, we motivate and discuss security and privacy of the system as well as its computational performance from both analytic and experimental points of view. In Section 7, we discuss the scalability and real world implementation of the system. We conclude with a summary of findings and future works in Section 8.

2. Related work

In this section, we present an overview on privacy research in transportation applications, before focusing on the computation of meeting points for ridesharing.

2.1. Privacy research in transportation

In Beresford and Stajano (2003), the authors define *location privacy* as preventing an unwanted entity to learn the past, present and future geographic position of an individual. Location privacy methods have been classified in two main categories by Cottrill (2009) : *policy-based* and *technique-based* approaches. Although policy-based approaches can deter an adversary from committing a privacy breach due to the threat of heavy penalties, their actions are really visible *a posteriori*, when the offense (*i.e.*, privacy violation) is already committed. In the other hand, technique-based approaches, such as the use of *privacy enhancing technologies* (PETs), help in reducing the success rate of the adversary when performing an attack by the mean of several techniques ranging from pseudonymization to advanced cryptographic primitives.

The main objective of these techniques is to prevent the adversary from learning the mobility traces of users. We summarize in Table 1 the most commonly available techniques. Although these techniques are quite generic, the specificity of the service considered impact the choice of the method to adopt.

In previous works, following the privacy-by-design principle, a privacy-preserving mechanism Sun et al. (2013) has been proposed to design fine-grained urban traffic modeling using mobile sensors. The proposed method, which uses a combination of access control and sanitization through the so-called notion *virtual trip line* (VTL) zones, ensures the unlinkability of

Method	Description
Aggregation	Aggregate mobility traces over time and space to reduce individual identification. Cohen et al. (2001); Castelluccia (2007)
Anonymity	k-anonymity requires that each equivalence class (*i.e.*, a set of mobility traces that are indistinguishable from each other) contains at least k records. Sweeney (2002); Chen et al. (2013)
Cloaking	Blur the mobility traces of a user into a region with lower spatiotemporal resolution. Gedik and Liu (2006); Cheng et al. (2006)
Encryption	Encrypt the mobility traces of a user before disclosing them. Xi et al. (2014); Bilogrevic et al. (2014)
Geographical masking	Perturb the original location through the addition of noise. Armstrong et al. (1999); Kwan et al. (2004)
Mix zone	Mix-zones are regions wherein the locations of users are not recorded. In addition, the pseudonym of a user entering this zone differs from the one that he will have when he exits. The overall objective is to increase the unlinkability of the user.Beresford and Stajano (2003); Freudiger et al. (2007)
Pseudonyms	Replace the identifier of a user with a pseudonym. Bettini et al. (2009); Buttyán et al. (2007)

Table 1: Overview of main location privacy techniques.

mobility traces related to users (*i.e.*, it is difficult for an adversary to assign traces to specific users). The released information can then be used for transportation modeling purposes. In the same line of work, authors in Ghasemzadeh et al. (2014) use anonymization techniques to construct privacy-preserving passengers' flow graph based on trajectory data. Another example of transportation systems with privacy in mind can be found in Zangui et al. (2013) in which authors have considered the value of privacy and proposed an opt-in approach to address location privacy concerns when using vehicle-tracking technologies to classify users with respect to their travel characteristics or attributes for a better congestion pricing policy. A survey in Fries et al. (2012) ends up with the conclusion that PETs, especially aggregation and masking, turn out to be the most appropriate way to protect the privacy of motorists when it comes to collecting travel times and speeds. In another work Bilogrevic et al. (2014), the authors have proposed algorithms based on homomorphic encryption schemes to find the optimal meeting point for a group of individuals while preserving their privacy. In Xi et al. (2014), the authors proposed a privacy-preserving shortest path algorithm based on the use of a cryptographic primitive known as *Private Information Retrieval* (PIR) Chor et al. (1998); Melchor et al. (2014). In this method, a user can privately build his itinerary on a graph from an origin point to a destination point by interrogating a path reconstruction matrix computed on an unreliable server with the Floyd-Warshall algorithm Floyd (1962). By doing so, the user can fully benefit fully from the navigation service but avoid the disclosure of his location information. In a relatively similar workWu et al. (2016), researchers have applied a graph compression algorithm on road networks to improve PIR-based privacy-preserving shortest path computation's runtime. Another line of research fo-

cus on how to formally design optimal obfuscation mechanisms in a way that both privacy and utility are satisfied. For instance, authors in Shokri et al. (2012) propose the optimal location-privacy preserving mechanism that maximize the expected distortion that an adversary using the optimal inference attack incurred when reconstructing the user location while heeding the user's service-quality requirement.

2.2. Meeting points for ridesharing

In a classical ridesharing the driver has to pick the rider at his origin and drop him off at his destination Agatz et al. (2012). This strategy turns out to be inappropriate as it highly increases the driver's detour and authors in Stiglic et al. (2015) have shown that considering meeting points other than origins and destinations improve the number of matching and thus the overall functionality of the ridesharing system.

The most relevant work to our paper is by Bit-Monnot and co-authors Bit-Monnot et al. (2013). They have solved the dynamic ridesharing problem presented in Section 1 by introducing the 2 Synchronization Points Shortest Path problem (`2SP-SP`). In the `2SP-SP`, given a multimodal graph $G = (V, E, \Sigma)$, a driver D and a rider R each having their respective origin (O_d and O_r), destination (D_d and D_r) and departure time (t_d and t_r), the objective is to find the optimal pick-up point i and drop-off point j, such that the sum of arrival times for both users is minimized. Several variants were studied, such as the situation in which there is no limitation on the detour for both users and all points in the network may be a meeting point. In a second variant, the authors introduce detour limitations for the rider. The proposed methods are based on several shortest path algorithms, in forward or in backward search, in the multimodal context. The main difficulty is related to the time-dependency, which impacts the performance of backward search algorithms and the consistency between arrival times and cost (total travel time) on nodes during the search. With this approach, the `2SP-SP` is solved in polynomial time with worst case complexity of $O(|E| \times |V|^2)$. Considering detour limitations for the rider usually improves significantly the efficiency of the method.

In Aissat and Oulamara (2014), the authors consider the `2SP-SP` on road networks with constraints to also limit the detour for the driver. They proposed heuristic methods based on several shortest path algorithms and on a sub-graph of potential pick-up and drop-off points. In another paper Varone and Aissat (2015), the multimodality is also taken into account for the rider. More precisely, the authors propose to limit the meeting points on the rider path from his origin to his destination and still consider the detour constraint for the driver.

In these seminal works, privacy-preserving constraints were not considered as each user communicates his full location information to a centralized server. In our work we consider the centralized exact approach from Bit-Monnot et al. (2013) as a baseline to compare our work against.

3. Requirements

In the proposed approach, we will use PETs and multimodal shortest path algorithms that we briefly review in this section.

3.1. Secure multiparty computation

Secure Multiparty Computation (SMC) is a branch of cryptography that aims at computing a function depending on the inputs of several parties in a distributed manner, so that only the result of the computation is revealed while the inputs of each party remain secret Lindell and Pinkas

(2009). The gold standard would be the existence of a *trusted third party* that would perform the entire computation and returns the output to all the parties involved while erasing afterwards his memory. In short, the objective of SMC is to achieve the same functionality but without having access to this trusted third party. Yao first defined the *two-party comparison problem*, now known as Yao's Millionaires' problem, and developed a provably secure solution for this problem Yao (1986). Since this seminal work, a lot of works have been done in the field of secure computation. SMC techniques are used for numerous tasks including coin-tossing, broadcasting, electronic voting, electronic auctions, electronic cash, contract signing, anonymous transactions and private information retrieval schemes. Hereafter, we focus solely on cryptographic primitives for solving the private set intersection problem, which is the basis of our work.

3.2. Private set intersection

In the field of SMC, series of work have been done Freedman et al. (2004); Kissner and Song (2005); Jarecki and Liu (2009); Hazay and Lindell (2008); De Cristofaro and Tsudik (2010); Dong et al. (2013) on a cryptographic primitive called Private Set Intersection (PSI) and a variant known as Private Set Intersection Cardinality (PSI-CA). In these tasks, two parties jointly compute respectively the intersection or the intersection's cardinality of their private input sets without leaking any additional information. Authors in Freedman et al. (2004) first introduced PSI based on *Oblivious Polynomial Evaluations*. Additive homomorphic encryption is used to obliviously evaluate a polynomial that represents a private input set. Following the same principle, Kissner and Song (2005) explores the power of polynomial representation of multisets, using operations on polynomials to obtain composable privacy-preserving multisets operations such as set intersection, union, cardinality and over-threshold operations. A recent work Aguilar-Melchor et al. (2016) introduced NFLlib, an optimized open-source C++ library designed to handle polynomials over $\mathbb{Z}p[x]/(x^n + 1)$. NFLlib includes optimized subroutines to perform arithmetic operations over polynomials and allows to easily implement ideal lattice-based cryptography. We rely on this library to implement PSI with linear complexity. *Committed Oblivious Pseudorandom Function Evaluation*, another approach to implement PSI using secure pseudorandom function evaluations, has been introduced by Hazay and Lindell (2008) and lately improved by Jarecki and Liu (2009) and De Cristofaro and Tsudik (2010). Finally, another line of work Dong et al. (2013) proposed *Oblivious Bloom Intersection* in which private input sets are inserted in Bloom filters followed by the intersection of the Bloom filters. The proposed approach run in linear time. However, the use of hash functions in Bloom filters leads to the existence of false positives impacting the accuracy of Bloom filter-based PSI.

3.3. Multimodal routing

A multimodal network can be represented as a graph $G = (V, E, \Sigma)$ and an automaton $\mathcal{A} = (Q, \Sigma, \delta, S, F)$ in which V is a set of vertices, E a set of edges and Σ a set of labels representing all available transportation modes (*e.g.*, walk, bike, car, public transportation ...). The automaton A models the constraints of feasible paths corresponding either to physical constraints or to user's preferences. $\mathcal{A}$ consists of a finite set Q of states, the alphabet Σ, a transition function $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$, a set of initial states S and a set F of final states.

The computation on multimodal shortest paths rely on the *Regular Language Constrained Shortest Path Problem* Barrett et al. (2000) that uses a regular language L to model constraints on transportation modes. This algorithm implicitly constructs a product network $G \times \mathcal{A}$ before applying a Dijkstra algorithm on this product network. The considered shortest path problem

remains polynomial and can be solved using a modified version of Dijkstra's algorithm Dijkstra (1959) known as D_{RegLC}. In ridesharing systems, each user (*i.e.*, driver and rider) has his own automaton linked to his set of modes. More precisely, for the driver we consider $\Sigma^d = \{\,\text{car}\,\}$ and $\Sigma^r = \{\,\text{walk}, \text{public transportation}\,\}$ for the rider. Based on these sets of modes, we have two D_{RegLC} algorithms : ${D_{\text{RegLC}}}^r$ for the rider and ${D_{\text{RegLC}}}^d$ for the driver.

We particularly rely on *isochrones* to compute potential meeting points. Given an origin vertex s and a radius r, an *isochrone* is the set of vertices in the shortest-path-tree T rooted at vertex s such that the path distance from root s to any other vertex $v \in T$ is less or equal to r. Isochrones are computed by the means of the D_{RegLC}'s algorithm. For instance, Figure 2 shows 7 isochrones of radius ranging from 10 to 60 minutes for a rider starting his journey at our laboratory LAAS-CNRS located at (43.563725, 1.476744) in Toulouse.

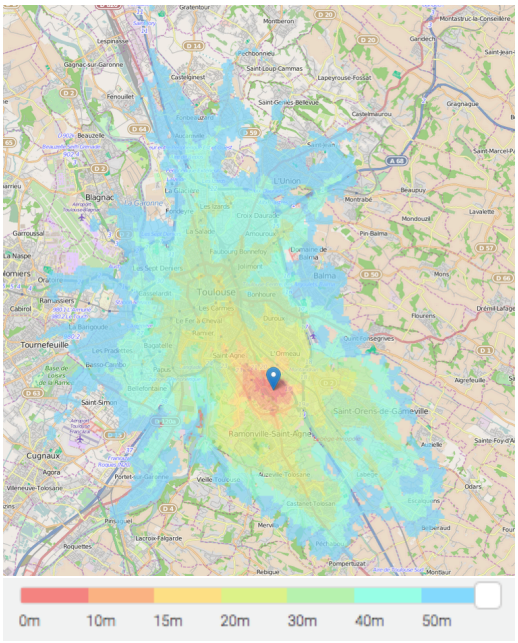

Figure 2: Example of isochrone.

4. Proposed approach to solve the Privacy-preserving `2SP-SP`

The privacy-preserving `2SP-SP` problem, hereafter referred to as `Priv-2SP-SP`, can be considered as an instance of `2SP-SP` with privacy constraints in addition to existing optimization constraints. `Priv-2SP-SP` should allow us to produce solutions as close as possible to `2SP-SP`'s ones but without having to disclose the location information of any participant to a centralized entity as well as to other users. In this section, we describe the core of our approach that combines secure multiparty computation and multimodal routing to implement a ridesharing service in a privacy-preserving manner.

4.1. Overview of the approach

4.1.1. User-centric architecture

In our architecture, the decision taken to enhance privacy (*e.g.*, hiding or perturbing location data) is made locally by the user. The rationale behind this choice is principally related to privacy. In fact, with this architecture, users have full control of their personal data and can decide the granularity with which sensitive information are disclosed. As a consequence, the quality of the service the user receives is dependent of the granularity of released information.

4.1.2. Functional model

We consider a pair composed of a driver D and a rider R wanting to ride together and who have already been put in contact by an external third party. Each user has an origin and destination location considered as being his private information. The driver will use his/her car throughout the trip while the rider may begin his/her journey with public transportation and walking mode, then will share a car with a driver to speed up the trip and joint his/her final destination using the same mode as in the beginning. We assume that D and R have both the same knowledge of the multimodal network G on which the trip will take place and that each user has an automaton $\mathcal{A}$ to model his mobility constraints. They want to determine a pick-up location and a drop-off location such that the trip is as cost effective as possible for both users and without having to disclose their origin and destination locations to each other or to any third party. For the sake of simplicity and to be in accordance with `2SP-SP`, we assume that D and R will start their journey at the same time but this assumption can easily be relaxed in practice without impacting too much the performance.

4.1.3. Security

For us, the primary security requirement is the privacy of user data. More precisely, users should be able to control their personal data and no user should be able to access the location data of another user unless explicitly authorized. Users communicate over a secure channel and have sufficient computing capabilities on their own platforms (*e.g.*, personal computer or smartphone) to perform the tasks requiring local computations such as the cryptographic ones or the computation of the isochrone.

4.1.4. Adversary model

We assume that the participants in the decentralized ridesharing system may be *honest-but-curious* adversaries in the cryptographic sense. In practice, this means that they will follow the directives of the established protocol while trying to infer additional information from the output of the computation, its intermediary results and the ciphered inputs of other participants.

4.2. Brute-force approach

In the transportation network $G = (V, E, \Sigma)$, all nodes reachable by both the rider and the driver may be a potential pick-up or drop-off. We denote by V' this set of nodes (*i.e.*, in practice $|V'|$ is close to $|V|$). A naive method will consider all combinations of these two sets, which corresponds to $|V'|^2$ solutions for the studied ridesharing problem. This method for computing the potential sets will preserve the privacy of each user and shift the privacy-preserving constraint on the second step dedicated to the selection of the best solution. However, computing all the solutions will involve high running time to obtain the cost of the $|V'|^2$ solutions. More precisely, this would be equal to $|V'|^2$ executions of a shortest path algorithm between the two potential sets

plus 2 shortest path algorithms for each user to evaluate the cost from their origin to each node in the potential pick-up set and from their destination to each node in the potential drop-off set.

4.3. Heuristic approach

To provide both privacy and efficiency, we propose another method to compute a set of potential interesting pick-up (respectively drop-off) points. In this method, we consider that each user computes by himself the two sets of potential meeting points. Then, the potential pick-up set for the driver, L^d_{up}, and the potential pick-up set for the rider, L^r_{up}, are intersected to obtain the set of shared potential pick-up points, L_{up}. The same intersection protocol is also used to produce the set of shared potential drop-off points, $L_{off} = L^d_{off} \cap L^r_{off}$. Afterwards, privacy-preserving set intersection is used to reduce the amount of information exchanged between users.

Rather than working in a centralized way as in `2SP-SP`, `Priv-2SP-SP` is a distributed approach and consists in three main steps: (1) local computation of potential meeting points (2) secure computation of shared pick-up and drop-off and (3) selection of the best pick-up and drop-off points. In Table 2, we summarize the notations used throughout the rest of the paper.

Symbol	**Meaning**
u, d, r	Ridesharing user, the driver, the rider
O_u, D_u	Origin of user u, destination of user u
L^u_{up}	Set of potential pick-up locations for the user u
L^u_{off}	Set of potential drop-off locations for the user u
$L_{up} = L^d_{up} \cap L^r_{up}$	Set of common pick-up locations both to the driver and the rider
$L_{off} = L^d_{off} \cap L^r_{off}$	Set of common drop-off locations both to the driver and the rider
m, n	$\lvert L_{up}\rvert$, $\lvert L_{off}\rvert$
$\pi_{i,j}$	Path from location i to location j
$cost^{\Sigma}(\pi_{i,j})$	Cost of the path $\pi_{i,j}$ using the set of modes Σ
$Trip^u(\pi_{i,j})$	Cost of the trip for the user u while taking the shared path $\pi_{i,j}$
$rk^u(\pi_{i,j})$	Rank of the path $\pi_{i,j}$ in the user's sorted list of shared paths
$sc^u(\pi_{i,j})$	Score given by the user to path $\pi_{i,j}$

Table 2: Summary of notations.

Table 3 describes an overview of the proposed approach.

4.3.1. Local computation of potential ridesharing locations

To get the sets of potential pick-up and drop-off locations, each user computes two isochrones, one from the origin location and the other from the destination location. Theses algorithms (${D_{\mathsf{RegLC}}}^d$ for the driver and ${D_{\mathsf{RegLC}}}^r$ for the rider) output the sets of potential pick-up and drop-off locations for both the driver and the rider. In each set, each scanned vertex will be represented with an unique and publicly available identifier as well as a confidential cost corresponding to the time spend by the user to reach the vertex in question. This cost has to remain confidential to avoid the possibility of *triangulation attacks*. In a triangulation attack, an adversary infers the location of the user based on the travel costs needed by him to reach some locations.

Priv-2SP-SP algorithm

Participants: driver D, rider R

Output: ideal pick-up and drop-off locations

1 - Local computation of potential ridesharing locations

- D: computes isochrones from origin and destination and obtains L^d_{up} and L^d_{off}
- R: computes isochrones from origin and destination and obtains L^r_{up} and L^r_{off}

2 - Secure collaboration to get common ridesharing locations

D and R use PSI to get L_{up} and L_{off}

3 - Computation of shared paths

- D: Computes the $m * n$ paths between L_{up} and L_{off}

$\longrightarrow \mathcal{P} = \left\{ \pi_{i,j} \;\middle|\; i \in L_{up}, j \in L_{off} \right\}, \left\{ cost^{\Sigma^d}(\pi_{i,j}) \;\middle|\; \pi_{i,j} \in \mathcal{P} \right\}$

4 - Local computation of trip costs associated to the use of each shared path

- D: computes $Trip^d(\pi_{i,j}) = cost^{\Sigma^d}(\pi_{O_d,i}) + cost^{\Sigma^d}(\pi_{i,j}) + cost^{\Sigma^d}(\pi_{j,D_d})$
- R: computes $Trip^r(\pi_{i,j}) = cost^{\Sigma^r}(\pi_{O_r,i}) + cost^{\Sigma^d}(\pi_{i,j}) + cost^{\Sigma^r}(\pi_{j,D_r})$

5 - Update of scores associated to shared paths

- D computes $sc^d(\pi_{i,j}) = m * n - rk^d(\pi_{i,j})$ for each shared path $\pi_{i,j}$.
- R computes $sc^r(\pi_{i,j}) = m * n - rk^r(\pi_{i,j})$ for each shared path $\pi_{i,j}$.
- To obtain $rk^u(\pi_{i,j})$ each user locally sorts the list of shared paths according to $Trip^u(\pi_{i,j})$

$\longrightarrow \left\{ sc^d(\pi_{i,j}) \right\}, \left\{ sc^r(\pi_{i,j}) \right\}$

6 - Election of ideal pick-up and drop-off locations

- D and R choose the path $\pi_{i,j}$ with the highest cumulative score.

$\longrightarrow (i^*, j^*) = \text{argmax}_{(i,j)} \left(sc^d(\pi_{i,j}) + sc^r(\pi_{i,j}) \right)$.

Table 3: Overview of the privacy-preserving approach for pick-up and drop-off computations.

4.3.2. Intersection of potential meeting locations

Once potential ridesharing locations are locally identified, both users collaboratively compute common pick-ups L_{up} and common drop-offs L_{off} with the PSI method. The private set intersection is chosen instead of the regular set intersection to follow the data minimization principle as recommended by the *privacy-by-design* approach Cavoukian (2009). Consequently, at the end of this step users will get only common ridesharing locations. To achieve this goal, each user constructs two private input sets. The first contains identifiers of potential pick-up locations represented as integers while the second consists of identifiers of potential drop-off locations also represented as integers. Then, the users encrypt the private inputs and launch the PSI method, which results in the obtaining of L_{up} and L_{off}.

4.3.3. Computation of shared paths

In this step, the driver computes all the possible paths $\pi_{i,j}$ that are accessible by car between L_{up} and L_{off} using ${D_{\mathsf{RegLC}}}^d$ algorithms. This task requires the computation of $\min(m, n)$ ${D_{\mathsf{RegLC}}}^d$ algorithms to get all the $m * n$ paths between L_{up} and L_{off}. Each path $\pi_{i,j}$ is associated with a unique identifier, a cost and two scores given by users. The scores that driver and rider associate to each path $\pi_{i,j}$ are initialize to 0 at this step.

We deliberately leave the driver in charge of this computation for scenarios in which we would like to take into account more complex itinerary's constraints associated to the driver that he only can express with his automaton (*e.g.*, taking a particular highway or using a particular parking station). In the absence of such constraint, both driver and rider can compute the shared path locally.

If the number of common pick-up and drop-off locations is important, this step may result in a high running time as it requires several multimodal shortest paths computation. To increase the efficiency of the proposed approach, we consider that the two users iteratively compute the intersection of their isochrone's subsets by gradually growing the radius from r_{min} to r. Iterations may stop as soon as a condition on the size of the intersection set is reached. This strategy helps in reducing the number of shared paths computed.

4.3.4. Local computation of trip costs and ranks

Driver computes $Trip^d(\pi_{i,j}) = cost^{\Sigma^d}(\pi_{O_d,i}) + cost^{\Sigma^d}(\pi_{i,j}) + cost^{\Sigma^d}(\pi_{j,D_d})$ for each path $\pi_{i,j}$ locally. More precisely, he adds the private costs $cost^{\Sigma^d}(\pi_{O_d,i})$ and $cost^{\Sigma^d}(\pi_{j,D_d})$, coming from the local computations of isochrones, to the publicly available cost $cost^{\Sigma^d}(\pi_{i,j})$ to obtain $Trip^d(\pi_{i,j})$. This cost is an estimation of the cost of his journey if he decides to take the path $\pi_{i,j}$ as a sub-path in his trip. Finally, the driver sorts $Trip^d(\pi_{i,j})$ to obtain a rank $rk^d(\pi_{i,j})$ for each shared path.

The rider computes $Trip^r(\pi_{i,j}) = cost^{\Sigma^r}(\pi_{O_r,i}) + cost^{\Sigma^d}(\pi_{i,j}) + cost^{\Sigma^r}(\pi_{j,D_r})$ following the same approach as the driver, which leads to a ranking of each shared paths. Note, that in this step, the rider needs to re-compute the cost from the drop-off locations to his destination due to the time-dependency of his transportation network.

At the end of this step, the shared path $\pi_{i,j}$ such that $Trip^d(\pi_{i,j})$ (respectively $Trip^r(\pi_{i,j})$) is minimized will be the most relevant for the driver (respectively the rider).

4.3.5. Update of scores associated to shared paths

The driver outputs for each shared path $\pi_{i,j}$ a score $sc^d(\pi_{i,j}) = f(rk^d(\pi_{i,j}))$ reflecting his willingness to use it as a sub-path during the ridesharing trip. For simplicity we define the

function f as:

$$f : \left| \begin{array}{ccc} \mathcal{P} & \longrightarrow & \mathbb{N} \\ \pi_{i,j} & \longmapsto & m * n - rk^d(\pi_{i,j}) \end{array} \right.$$

The rider also outputs a score $sc^r(\pi_{i,j}) = g(rk^r(\pi_{i,j}))$ for each shared path $\pi_{i,j}$ in which g is defined as:

$$g : \left| \begin{array}{ccc} \mathcal{P} & \longrightarrow & \mathbb{N} \\ \pi_{i,j} & \longmapsto & m * n - rk^r(\pi_{i,j}) \end{array} \right.$$

In both cases, the better the rank of a shared path is, the higher is the score associated with this path. For simplicity, we design the rank conversion function as the complementary of the rank. To improve the privacy, it would be possible to randomize this function to make it more difficult for an adversary to infer the rank from the score. The conversion of the trip cost to a score has the advantage of adding more privacy on the choice of each user because it discloses less information than the trip cost itself.

4.3.6. Election of ideal pick-up and drop-off locations

In this step, each user communicates the score he attributes to each $\pi_{i,j}$. Then, a simple voting procedure is applied in which instead of taking into account the costs of journeys, the focus is made on the scores only (*i.e.*, the values of costs are kept confidential for privacy reasons).

Instead of using the simple voting procedure, other voting strategies would be possible provided that they are not computationally expensive. Note that the same procedure must be used by the two users to obtain the same result on both. `2SP-SP` sought to minimize the sum of the costs of journeys while `Priv-2SP-SP` reduces the problem to the maximization of the scores. To achieve this goal, each user selects the shared path $\pi_{i,j}$ such that $(sc^d(\pi_{i,j}) + sc^r(\pi_{i,j}))$ is maximized. More precisely, the pick-up and drop-off locations corresponding to the solution are obtained in the following manner: $(\mathrm{i}^*, \mathrm{j}^*) = \mathrm{argmax}_{(i,j)} \, (sc^d(\pi_{i,j}) + sc^r(\pi_{i,j}))$.

By default, the election works in a fair manner by deciding that a score assigned to a given path has the same weight for both parties. However, it is easy to design an unbalanced version of this vote by using a positive multiplicative factor reflecting the importance given to one party over the other. In this case, the best path will be the one such as $\alpha \times \mathsf{sc}^{\mathsf{d}}(\pi_{\mathrm{i,j}}) + \beta \times \mathsf{sc}^{\mathsf{r}}(\pi_{\mathrm{i,j}})$ is maximized with α and β representing respectively the influence of the driver and of the rider on the choice of a shared path.

4.3.7. Waiting time

In the proposed approach, there is no guarantee that the waiting time of each participant at the pick-up location will be the lowest one. To ensure that the waiting time meets the constraints of the schedule of each user, one can rely on homomorphic encryption to check if $|cost^{\Sigma^d}(\pi_{O_d,i}) - cost^{\Sigma^r}(\pi_{O_r,i})| \leq \max(wt^d(i), wt^r(i))$, in which $wt^u(i)$ represents the maximal waiting time of user u at the pick-up location i. To take into account the limitation of each user on the waiting time, in this last step it is important to check if the waiting time corresponding to the best selected path is valid using a secure comparison method. If this condition is not met (*i.e.*, the time is not valid), the users have to select a second path and so forth until the waiting time of the path considered matches their constraints.

5. Security and performance analysis

5.1. Security analysis

Since we solve `Priv-2SP-SP` by using a PSI method, the security of our scheme mainly depends on the security of the PSI. In Freedman et al. (2004), the authors proved the security of PSI in the semi-honest model. We recall hereafter the sketch of the scheme and the security proof.

Consider a party A running a PSI method with another party B. To summarize A starts by generating a public and private encryption keys of an homomorphic encryption scheme before defining a polynomial whose roots are A's inputs. Then, A sends the public key and the encryption of the polynomial coefficients to B. B, upon reception of these elements, evaluates in an oblivious fashion the polynomial with his inputs and returns the result to A. Finally, A decrypts an item if it is in the intersection of the two sets or a random number otherwise.

The *correctness* of the PSI method is ensured by the fact that it successfully evaluates with high probability. The *privacy of the input of A* is guaranteed if the encryption scheme used is semantically secure. Indeed in this case, the views of B for any two inputs of A are indistinguishable. The *privacy of the input of B* comes from the fact that for every party A' operating in the real world, there is a party A operating in the ideal model, such that for every input of B, the views of the parties A and B in the ideal model are indistinguishable from the views of A' and B in the real world.

5.2. Privacy analysis

Even if the PSI method is secure, the context in which it is applied may help an honest-but-curious adversary in inferring some information about users. For instance despite the fact that the input data are encrypted, each user has access to the size of the ciphertexts, which can indirectly disclose location information as we show hereafter. First, we look at three regions in the Toulouse area: countryside, suburban and downtown areas. Then, we randomly generate in each area 100 locations from which we launch 30 minutes isochrones for both rider and driver to see how the number of potential ridesharing location varies from one location to another and from one user to another. In Figure 3, we report on the cumulative distribution functions of the isochrone's size for each area and for each user.

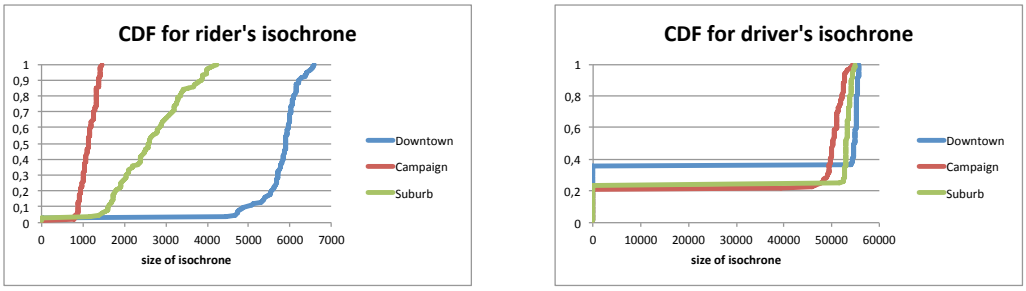

Figure 3: Inference analysis.

This analysis shows that drivers' isochrones are more indistinguishable than riders' ones. Thus, the observation of input size (*i.e.*, the size of isochrones) can help an adversary to guess the region in the transportation graph in which the rider is located. This problem comes from

the topology of the graph. In particular, for a rider it is much easier to reach any location from downtown because of the high availability of public transportation, than from suburban or countryside areas. In our approach, we prevent this type of attack by fixing the size of isochrone for each user in order to have an indistinguishability of both driver's and rider's isochrones. Another information disclosed by the computation is the score that each user attributes to shared paths, which is obtained by first converting the trip cost into rank and then to score. This transformation adds more privacy because of the loss of information that it incurs.

5.3. Communication and computational complexities

The complexity of the proposed approach is directly proportional to the complexity of the shortest path algorithms. More precisely, each participant runs 2 D_{RegLC} algorithms to get their potential pick-up and drop-off locations. Finding the common POIs using PSI has a linear complexity in the size of the input sets. Once common pick-up and drop-off locations are discovered, the participants run $|V|$ D_{RegLC} in the worst case to get all the $|V|^2$ possible shared paths. The scoring and ranking step requires the insertion of $|V|^2$ paths into a binary heap. All these steps lead to a global complexity of $O(|V|\times|E|\times\log|V|)$. Finally, the communication complexity is $O(|V|\times\log|V|)$ as the identifier of each node may have to be exchanged after encryption. The respective worst case complexities of the exact and centralized approach (`2SP-SP`) and the privacy-preserving approach (`Priv-2SP-SP`) are summarized in Table 4.

	Communication cost	**Computational complexity**
`2SP-SP`	$O(1)$	$O\left(\|E\|\times\|V\|^2\right)$
`Priv-2SP-SP`	$O(\|V\|\times\log\|V\|)$	$O(\|E\|\times\|V\|\times\log\|V\|)$

Table 4: Cost and runtime analysis.

6. Experiments

In this section, we report on the experiments that we have conducted to evaluate the effectiveness of our approach.

6.1. Description of the dataset

The tests have been performed on 200 randomly generated instances of ridesharing problem using a multimodal transportation graph of the city of Toulouse in which we consider any vertex as potential ridesharing location. The multimodal graph has been generated using Openstreetmap's[1] data for the road network and Tisseo's GTFS[2] data for the public transportation network. The resulting graph has the following characteristics: $|V|$ = 75 837, $|E|$ = 527 053, Σ = {Walk, Car, Bus, Subway, Tramway}.

The multimodal routing algorithms have been implemented in C++ as well as the cryptographic primitives using the NFLlib library Aguilar-Melchor et al. (2016). Experiments were run on a virtual machine with 5GB RAM on a 2,9 GHz Intel Core i7 host machine.

[1] http://www.openstreetmap.org
[2] https://developers.google.com/transit/gtfs

To generate a ridesharing instance, we choose at random an origin and a destination for a pair of driver and rider. Both origins (respectively destinations) are chosen to geographically close in the transportation network to reflect the spatial matching that we assumed to be pre-arrange in our system assumption. More precisely, we create two groups each containing 100 instances of ridesharing. In Table 5, we summarize the main characteristics of each group.

	Group I	Group II
Distance between the both origins (respectively destinations)	2000 m	800 m
Travel time for the rider alone	1h15min	2h
Travel time for the driver alone	19min	20min

Table 5: Instances characteristics.

The main difference between the two datasets is the proximity between the driver and the rider. In the first group ridesharing candidates are more distant to each other. We launch the proposed `Priv-2SP-SP` method with the generated instances as input and compare our results to the optimal solutions obtained using the centralized method `2SP-SP` proposed in Bit-Monnot et al. (2013).

6.2. Experimental results

To qualitatively compare `2SP-SP` and `Priv-2SP-SP`, we define a common ridesharing cost as a function of the time each user spends to reach the pick-up location, the waiting time, the time they spend together on the shared path and the time each of them uses to reach final destination from the drop-off location. More formally the ridesharing cost value *Rcost* is:

$$\begin{aligned} Rcost = & \, cost^{\Sigma^d}(\pi_{O_d,Up^*}) + cost^{\Sigma^r}(\pi_{O_r,Up^*}) \\ & + |cost^{\Sigma^d}(\pi_{O_d,Up^*}) - cost^{\Sigma^r}(\pi_{O_r,Up^*})| \\ & + 2 * cost^{\Sigma^d}(\pi_{Up^*,Off^*}) \\ & + cost^{\Sigma^d}(\pi_{Off^*,D_d}) + cost^{\Sigma^r}(\pi_{Off^*,D_r}) \end{aligned}$$

Of course, other ridesharing cost functions can be defined but we believe that the one we propose is quite natural.

We summarize the results obtained with this synthetic data in Table 6.

	Rcost (s)		Runtime (s)	
	Group I	Group II	Group I	Group II
`2SP-SP`	3145.44	2872.03	1.09	0.75
`Priv-2SP-SP`	3283.53	2941.25	0.67	0.48

Table 6: Cost and runtime analysis.

On average over the 200 instances, the gap between the carpooling cost of `Priv-2SP-SP` and `2SP-SP` is acceptable and there is no major difference between the two approaches regarding the cost. More precisely, the average deviation compare to the centralized approach equals to 4.58%

in group I and 2.52% in group II. In Table 7, we report on the gap (δ in percentage) between 2SP-SP and Priv-2SP-SP with regard to the ridesharing cost in both group I and group II.

	$\delta = 0\%$	$\delta \in]0\%, 5\%]$	$\delta \in]5\%, 10\%]$	$\delta \in]10\%, 20\%]$
Group I	16	55	20	9
Group II	30	58	10	2

Table 7: Gap analysis of ridesharing cost

With respect to the runtime, Priv-2SP-SP is more efficient than 2SP-SP. This observation confirms our preliminary complexity analysis.

Figure 4 shows that in both group I and group II, Priv-2SP-SP is globally better than 2SP-SP from the runtime point of view.

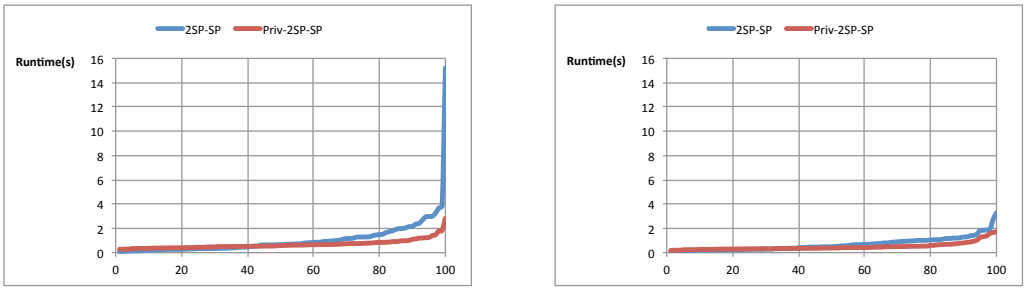

Figure 4: Runtime comparison in group I (left) and group II (right).

The two most time-consuming steps in Priv-2SP-SP are the PSI algorithm and the computation of shared paths. The execution time of the PSI depends on the size of the input data. Since all the vertices of the graph are considered as potential ridesharing locations, the running time measured represents the worst case scenario. An intuitive manner to decrease the runtime is to reduce the number of potential ridesharing locations. For instance, this could be done by using predefined ridesharing locations or by grouping geographically close positions into one location.

In addition to this runtime analysis, we report in Table 8 some characteristics of the ridesharing solutions that we consider important.

In this table, we report on the average value of the size of common pick-up and drop-off locations for the Priv-2SP-SP approach, the similarity between 2SP-SP and Priv-2SP-SP solutions with respect to the proximity of pick-up locations on the one hand and the proximity of drop-off locations on the other, the waiting time of both driver and rider, the detour of the driver, and finally, the gain in time of the rider when he use ridesharing as an alternative transportation mode.

This analysis reveals that the size of common pick-up and drop-off locations remains reasonable and that it is sufficient to obtain meeting points close to optimal meeting points. More precisely, the average distance between ridesharing locations is 230.06 meters for pick-ups and 202.13 meters for drop-offs in group I. In group II, values are 103.42 meters for pick-ups and 99.23 meters for drop-offs.

Waiting times at the pick-up are very short for both driver and rider in both 2SP-SP and Priv-2SP-SP approaches with an advantage for 2SP-SP. While we consider this gap acceptable,

Observation		**Group I**	**Group II**
Pick-ups size		23	15
Drop-offs size		17	14
Distance between pick-ups (m)		230.06	103.42
Distance between drop-offs (m)		202.13	99.23
Driver's waiting time	`2SP-SP`	2.23%	2.09%
	`Priv-2SP-SP`	6.44%	5.64%
Rider's waiting time	`2SP-SP`	0.33%	0.32%
	`Priv-2SP-SP`	0.16%	0.24%
Driver's detour	`2SP-SP`	53.46%	30.33%
	`Priv-2SP-SP`	51.29%	30.29%
Rider's gain	`2SP-SP`	231.06%	444.99%
	`Priv-2SP-SP`	210.75%	437.84%

Table 8: Characteristics of ridesharing solutions.

users can use the strategy we define in Section 4 to securely check the waiting time and accept or reject a solution.

We obtain driver's detour and rider's gain by computing the gap between their respective original trip cost (*i.e.*, the time it takes to do the origin destination trip alone) and the effective ridesharing cost (*i.e.*, the time each user spends while using ridesharing). Our results show an important gain in travel time for the rider (a reduction of 50 minutes in group I and 1 hour and 36 minutes in group II) against a slightly longer trip for the driver (an extra traveling time of 8 minutes in group I and 5 minutes in group II). In real world, the driver's detour will be offset by the rider's fare.

6.3. Limitation of driver's detour

In our experiments, the driver detour is not constrained and the objective is to reduce to total travel time for both users. Nevertheless, we can evaluate the number of solutions having a limited (reasonably) detour for the driver comparatively to his shortest path from his origin to his destination. In Table 9, we show the number of solutions having a driver's detour less or equal to 1.2 (respectively 1.4) times the *origin-destination* shortest path.

	1.2×SPP		1.4×SPP	
	Group I	Group II	Group I	Group II
Nb solutions `2SP-SP`	26	43	49	77
Nb solutions `Priv-2SP-SP`	29	42	53	79

Table 9: Feasible solutions with a limited driver's detour.

The proposed privacy-preserving approach is globally better than 2SP-SP when it comes to driver's detour. In addition, the better is the spatial matching (*i.e*, the distance between the driver and the rider at both origin and destination) the smaller is the detour and the better is the quality the solution in a broader sense.

To reduce the driver's detour in our solutions, while still preserving privacy, we proposed another variant of our approach. In this variant, the driver's isochrones are replaced by the set of vertices reached during a goal-directed shortest path algorithm (A*). This algorithm can be parameterized by the speed of the driver. When this speed decreases, the set of reached vertices is close to the driver shortest path. In Table 10, we summarize our results considering four different speeds: 10m/s, 15m/s, 20m/s and 25m/s. For simplicity, we consider only instances of group I for this particular aspect.

	cost gap	detour-driver (s)	gain-rider (s)	runtime (s)
A* – 10	44.80%	25.78%	125.17%	0.684
A* – 15	17.39%	41.57%	173.87%	0.465
A* – 20	9.17%	48.32%	195.05%	0.452
A* – 25	6.17%	50.88%	205.53%	0.481

Table 10: Results for the A* variants.

Using the A* variants instead of isochrone computation for the driver lead to reduce his detour especially when the algorithm A* is limited by a low speed. However, this detour limitation is at the detriment of the rider that has fewer gain, and leads to an increasing of the total ridesharing cost. Moreover, using a low speed in A* variants increases the runtime as it is more difficult to obtain common meeting points.

In Table 11, we give the number of solutions having a limited detour for each of A* variants.

	1.2×SPP	1.4×SPP
Nb solutions A* – 10	47	79
Nb solutions A* – 15	31	57
Nb solutions A* – 20	30	53
Nb solutions A* – 25	29	53

Table 11: Feasible solutions with a limited driver's detour - A* variants.

The only variant having a great impact on the number of solutions with a limited detour is the A* – 10 variant, the three other ones produce approximately the same number of solutions as the original method (*i.e.*, using isochrone computation).

7. Discussion

7.1. Scalability

In our experiments, we have considered the multimodal graph of *Toulouse*, a 118.3 km^2 large city, with 75837 vertices and 527053 edges. As shown in Section 5, the computational complexity of our approach is $O(|E|\times|V|\times\log|V|)$. More generally, we can consider the computational complexity as $O(|E|\times|N|\times\log|V|)$ in which N represents the set of potential ridesharing locations. We conduct the experiments with the worse case scenario in which $N = V$. The reduction of N to a set of user-defined ridesharing locations, highly desirable in practice, will make the algorithm more scalable as both the runtime required by the computation of shared paths and the runtime of the private set intersection grow with N.

7.2. Real world implementation

All our experiments have been done in simulation mode on virtual machines. In contrast in the real world, we can rely on a publish/subscribe model to match drivers and riders in the first place and then a peer-to-peer communication will start between each couple of rider and driver to run the proposed scheme and obtain meeting and arriving points. In Figure 5, we present a global architecture of the system. Our solution can be implemented with the existing network infrastructure and the current mobile technologies.

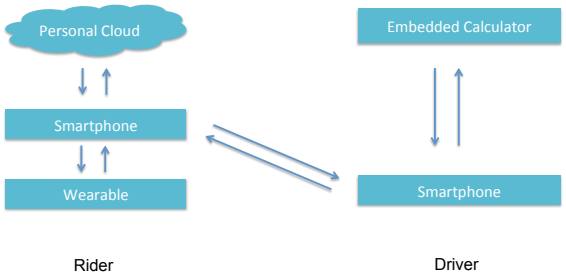

Figure 5: System architecture.

For the rider, wearable like smart watch can be used as user interface, smartphone will be used for computations and P2P communications and finally the most computationally expensive routines will be delegated to a remote personal cloud.

For the driver, computations can be balanced between an embedded processor on the car and a smartphone that will also take care of P2P communications.

7.3. The case of malicious adversary

While we have considered in our system the honest-but-curious adversary model, it is interesting to see how one can deal with malicious adversary's behavior. In contrast to honest-but-curious ones, malicious adversaries may cheat arbitrarily during the running of the protocol, for instance he may try to inject forged inputs like a fake isochrone to leak information about users. To thwart such kind of attack, one can rely on a privacy-preserving location proof system Zhu and Cao (2011); Gambs, Killijian, Roy and Traoré (2014) that by allowing to securely prove a location will serve as a mean to provide validate certificates on isochrones.

8. Conclusion

In this paper, we have proposed a distributed and secure algorithm enabling users of ridesharing services to interact in a private manner. We rely on local computations, secure multiparty computation (especially private set intersection) and multimodal routing algorithms, to obtain a ridesharing itinerary for users. Our results and analysis show that privacy enhancing technologies can help in solving the ridesharing problem while respecting the privacy of users and obtaining results qualitatively comparable to the one obtained from conventional centralized infrastructures. For the ridesharing application, the proposed privacy-preserving approach does not impact significantly the quality of the solutions with regard to optimal ones while leading to a lower running time as an additional benefit.

Our future work will investigate the generic version of ridesharing problem, involving multiple drivers and multiple riders, and study the corresponding privacy-preserving matching problem. Finally in a broader context, we will also explore how secure multiparty computation techniques can be used to solve other mobility problems. For instance in a geosocial network, one could be interesting in finding a common meeting points for a set of users by combining multimodal routing algorithm with their social preferences. Another application concerns the transportation of goods transportation in which different companies want to collaboratively manage the transportation of their goods to reduce costs without disclosing their private information such as their exact resources.

Agatz, N., Erera, A., Savelsbergh, M. and Wang, X. (2012), 'Optimization for dynamic ride-sharing: A review', *European Journal of Operational Research* **223**(2), 295–303.

Aguilar-Melchor, C., Barrier, J., Guelton, S., Guinet, A., Killijian, M.-O. and Lepoint, T. (2016), Nfllib: Ntt-based fast lattice library, *in* 'RSA Conference Cryptographers' Track'.

Aissat, K. and Oulamara, A. (2014), Dynamic ridesharing with intermediate locations, *in* 'Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2014 IEEE Symposium on', IEEE, pp. 36–42.

Armstrong, M. P., Rushton, G. and Zimmerman, D. L. (1999), 'Geographically masking health data to preserve confidentiality', *Statistics in Medicine* **18**(5), 497–525.

Artigues, C., Deswarte, Y., Guiochet, J., Huguet, M.-J., Killijian, M.-O., Powell, D., Roy, M., Bidan, C., Prigent, N., Anceaume, E., Gambs, S., Guette, G., Hurfin, M. and Schettini, F. (2012), Amores: An architecture for mobiquitous resilient systems, *in* 'Proceedings of the 1st European Workshop on AppRoaches to MObiquiTous Resilience', ARMOR '12, ACM, New York, NY, USA, pp. 7:1–7:6.
URL: *http://doi.acm.org/10.1145/2222436.2222443*

Barrett, C., Jacob, R. and Marathe, M. (2000), 'Formal-language-constrained path problems', *SIAM Journal on Computing* **30**(3), 809–837.

Beresford, A. R. and Stajano, F. (2003), 'Location privacy in pervasive computing', *Pervasive Computing* **2**(1), 46–55.

Bettini, C., Mascetti, S., Wang, X. S., Freni, D. and Jajodia, S. (2009), Anonymity and historical-anonymity in location-based services, *in* 'Privacy in Location-Based Applications', Springer, pp. 1–30.

Bilogrevic, I., Jadliwala, M., Joneja, V., Kalkan, K., Hubaux, J.-P. and Aad, I. (2014), 'Privacy-preserving optimal meeting location determination on mobile devices', *Information Forensics and Security, IEEE Transactions on* **9**(7), 1141–1156.

Bit-Monnot, A., Artigues, C., Huguet, M.-J. and Killijian, M.-O. (2013), Carpooling: the 2 synchronization points shortest paths problem, *in* '13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)', pp. 12–p.

Buttyán, L., Holczer, T. and Vajda, I. (2007), On the effectiveness of changing pseudonyms to provide location privacy in vanets, *in* 'Security and Privacy in Ad-hoc and Sensor Networks', Springer, pp. 129–141.

Castelluccia, C. (2007), Securing very dynamic groups and data aggregation in wireless sensor networks, *in* 'Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on', IEEE, pp. 1–9.

Cavoukian, A. (2009), 'Privacy by design... take the challenge. information and privacy commissioner of ontario (canada)'.

Chen, R., Fung, B. C., Mohammed, N., Desai, B. C. and Wang, K. (2013), 'Privacy-preserving trajectory data publishing by local suppression', *Information Sciences* **231**, 83–97.

Cheng, R., Zhang, Y., Bertino, E. and Prabhakar, S. (2006), Preserving user location privacy in mobile data management infrastructures, *in* 'Privacy Enhancing Technologies', Springer, pp. 393–412.

Chor, B., Kushilevitz, E., Goldreich, O. and Sudan, M. (1998), 'Private information retrieval', *Journal of the ACM (JACM)* **45**(6), 965–981.

Cohen, N. H., Purakayastha, A., Turek, J., Wong, L. and Yeh, D. (2001), 'Challenges in flexible aggregation of pervasive data', *IBM Research Division* **1**.

Cottrill, C. D. (2009), 'Approaches to privacy preservation in intelligent transportation systems and vehicle-infrastructure integration initiative', *Transportation Research Record: Journal of the Transportation Research Board* **2129**, 9–15.

Cottrill, C. D. and Thakuriah, P. V. (2015), 'Location privacy preferences: A survey-based analysis of consumer awareness, trade-off and decision-making', *Transportation Research Part C: Emerging Technologies* **56**, 132–148.

De Cristofaro, E. and Tsudik, G. (2010), Practical private set intersection protocols with linear complexity, *in* 'Financial Cryptography and Data Security', Springer, pp. 143–159.

Dijkstra, E. W. (1959), 'A note on two problems in connexion with graphs', *Numerische mathematik* **1**(1), 269–271.

Dong, C., Chen, L. and Wen, Z. (2013), When private set intersection meets big data: an efficient and scalable protocol, *in* 'Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security', ACM, pp. 789–800.

Floyd, R. W. (1962), 'Algorithm 97: shortest path', *Communications of the ACM* **5**(6), 345.

Freedman, M. J., Nissim, K. and Pinkas, B. (2004), Efficient private matching and set intersection, *in* 'Advances in Cryptology-EUROCRYPT 2004', Springer, pp. 1–19.

Freudiger, J., Raya, M., Félegyházi, M., Papadimitratos, P. et al. (2007), Mix-zones for location privacy in vehicular networks, *in* 'ACM Workshop on Wireless Networking for Intelligent Transportation Systems (WiN-ITS)'.

Fries, R. N., Gahrooei, M. R., Chowdhury, M. and Conway, A. J. (2012), 'Meeting privacy challenges while advancing intelligent transportation systems', *Transportation Research Part C: Emerging Technologies* **25**, 34–45.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X. and Koenig, S. (2013), 'Ridesharing: The state-of-the-art and future directions', *Transportation Research Part B: Methodological* **57**, 28–46.

Gambs, S., Killijian, M. and del Prado Cortez, M. N. (2011), 'Show me how you move and I will tell you who you are', *Transactions on Data Privacy* **4**(2), 103–126.
URL: *http://www.tdp.cat/issues11/abs.a078a11.php*

Gambs, S., Killijian, M.-O. and del Prado Cortez, M. N. (2014), 'De-anonymization attack on geolocated data', *Journal of Computer and System Sciences* **80**(8), 1597–1614.

Gambs, S., Killijian, M.-O., Roy, M. and Traoré, M. (2014), Props: A privacy-preserving location proof system, *in* '2014 IEEE 33rd International Symposium on Reliable Distributed Systems', IEEE, pp. 1–10.

Gedik, B. and Liu, L. (2006), 'Mobieyes: A distributed location monitoring service using moving location queries', *Mobile Computing, IEEE Transactions on* **5**(10), 1384–1402.

Ghasemzadeh, M., Fung, B. C., Chen, R. and Awasthi, A. (2014), 'Anonymizing trajectory data for passenger flow analysis', *Transportation Research Part C: Emerging Technologies* **39**, 63–79.

Hazay, C. and Lindell, Y. (2008), Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries, *in* 'Theory of Cryptography', Springer, pp. 155–175.

Jarecki, S. and Liu, X. (2009), Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection, *in* 'Theory of Cryptography', Springer, pp. 577–594.

Kissner, L. and Song, D. (2005), Privacy-preserving set operations, *in* 'Advances in Cryptology–CRYPTO 2005', Springer, pp. 241–257.

Kwan, M.-P., Casas, I. and Schmitz, B. C. (2004), 'Protection of geoprivacy and accuracy of spatial information : How effective are geographical masks?', *Cartographica : The International Journal for Geographic Information and Geovisualization* **39**(2), 15– 28.

Lindell, Y. and Pinkas, B. (2009), 'Secure multiparty computation for privacy-preserving data mining', *Journal of Privacy and Confidentiality* **1**(1), 5.

Melchor, C. A., Barrier, J., Fousse, L. and Killijian, M.-O. (2014), 'Xpire: Private information retrieval for everyone.', *IACR Cryptology ePrint Archive* **2014**, 1025.

Quercia, D., Lathia, N., Calabrese, F., Di Lorenzo, G. and Crowcroft, J. (2010), Recommending social events from mobile phone location data, *in* 'Data Mining (ICDM), 2010 IEEE 10th International Conference on', IEEE, pp. 971–976.

Shokri, R., Theodorakopoulos, G., Troncoso, C., Hubaux, J.-P. and Le Boudec, J.-Y. (2012), Protecting location privacy: optimal strategy against localization attacks, *in* 'Proceedings of the 2012 ACM conference on Computer and communications security', ACM, pp. 617–627.

Stiglic, M., Agatz, N., Savelsbergh, M. and Gradisar, M. (2015), 'The benefits of meeting points in ride-sharing systems', *Transportation Research Part B: Methodological* **82**, 36–53.

Sun, Z., Zan, B., Ban, X. J. and Gruteser, M. (2013), 'Privacy protection method for fine-grained urban traffic modeling using mobile sensors', *Transportation Research Part B: Methodological* **56**, 50–69.

Sweeney, L. (2002), 'k-anonymity: A model for protecting privacy', *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **10**(05), 557–570.

Varone, S. and Aissat, K. (2015), Routing with public transport and ride-sharing, *in* 'Sixth International Workshop on Freight Transportation and Logistics (Odysseus)'.

Wu, D. J., Zimmerman, J., Planul, J. and Mitchell, J. C. (2016), 'Privacy-preserving shortest path computation', *arXiv preprint arXiv:1601.02281* .

Xi, Y., Schwiebert, L. and Shi, W. (2014), 'Privacy preserving shortest path routing with an application to navigation', *Pervasive and Mobile Computing* **13**, 142 – 149.

Yao, A. (1986), How to generate and exchange secrets, *in* 'Foundations of Computer Science, 1986., 27th Annual Symposium on', IEEE, pp. 162–167.

Zangui, M., Yin, Y., Lawphongpanich, S. and Chen, S. (2013), 'Differentiated congestion pricing of urban transportation networks with vehicle-tracking technologies', *Transportation Research Part C: Emerging Technologies* **36**, 434–445.

Zhu, Z. and Cao, G. (2011), Applaus: A privacy-preserving location proof updating system for location-based services, *in* 'INFOCOM, 2011 Proceedings IEEE', IEEE, pp. 1889–1897.