Practically Preserving and Evaluating Location Privacy

Vincent Primault

N°d'ordre NNT :

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
INSA Lyon

Ecole Doctorale N° 512
InfoMaths

Spécialité de doctorat : Informatique

Soutenue publiquement le 01/03/2018, par :
Vincent Primault

Practically Preserving and Evaluating Location Privacy

Devant le jury composé de :

Chbeir, Richard	Professeur, IUT de Bayonne et du Pays Basque	Rapporteur
Nguyen, Benjamin	Professeur, INSA-CVL	Rapporteur
Capra, Licia	Professeur, UCL	Examinatrice
Huguenin, Kévin	Professeur assistant, UNIL	Examinateur
Brunie, Lionel	Professeur, INSA-LYON	Directeur de thèse
Ben Mokhtar, Sonia	Chargée de recherche, INSA-LYON	Examinatrice
Lauradoux, Cédric	Chargé de recherche, INRIA	Examinateur

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec : Renée EL MELHEM Bat Blaise Pascal 3e etage secretariat@edchimie-lyon.fr Insa : R. GOURDON	M. Stéphane DANIELE Institut de Recherches sur la Catalyse et l'Environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 avenue Albert Einstein 69626 Villeurbanne cedex directeur@edchimie-lyon.fr
E.E.A.	**ELECTRONIQUE, ELECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec : M.C. HAVGOUDOUKIAN Ecole-Doctorale.eea@ec-lyon.fr	M. Gérard SCORLETTI Ecole Centrale de Lyon 36 avenue Guy de Collongue 69134 ECULLY Tél : 04.72.18 60.97 Fax : 04 78 43 37 17 Gerard.scorletti@ec-lyon.fr
E2M2	**EVOLUTION, ECOSYSTEME, MICROBIOLOGIE, MODELISATION** **http://e2m2.universite-lyon.fr** Sec : Sylvie ROBERJOT Bât Atrium - UCB Lyon 1 04.72.44.83.62 Insa : H. CHARLES secretariat.e2m2@univ-lyon1.fr	M. Fabrice CORDEY CNRS UMR 5276 Lab. de géologie de Lyon Université Claude Bernard Lyon 1 Bât Géode 2 rue Raphaël Dubois 69622 VILLEURBANNE Cédex Tél : 06.07.53.89.13 cordey@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTE** **http://www.ediss-lyon.fr** Sec : Sylvie ROBERJOT Bât Atrium - UCB Lyon 1 04.72.44.83.62 Insa : M. LAGARDE secretariat.ediss@univ-lyon1.fr	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 avenue Jean Capelle INSA de Lyon 696621 Villeurbanne Tél : 04.72.68.49.09 Fax :04 72 68 49 16 Emmanuelle.canet@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHEMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec :Renée EL MELHEM Bat Blaise Pascal, 3e étage Tél : 04.72. 43. 80. 46 Fax : 04.72.43.16.87 infomaths@univ-lyon1.fr	M. Luca ZAMBONI Bâtiment Braconnier 43 Boulevard du 11 novembre 1918 69622 VILLEURBANNE Cedex Tél :04 26 23 45 52 zamboni@maths.univ-lyon1.fr
Matériaux	**MATERIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec : Marion COMBE Tél:04-72-43-71-70 –Fax : 87.12 Bat. Direction ed.materiaux@insa-lyon.fr	M. Jean-Yves BUFFIERE INSA de Lyon MATEIS Bâtiment Saint Exupéry 7 avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72.43 71.70 Fax 04 72 43 85 28 Ed.materiaux@insa-lyon.fr
MEGA	**MECANIQUE, ENERGETIQUE, GENIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr/** Sec : Marion COMBE Tél:04-72-43-71-70 –Fax : 87.12 Bat. Direction mega@insa-lyon.fr	M. Philippe BOISSE INSA de Lyon Laboratoire LAMCOS Bâtiment Jacquard 25 bis avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72 .43.71.70 Fax : 04 72 43 72 37 Philippe.boisse@insa-lyon.fr
ScSo	**ScSo*** **http://ed483.univ-lyon2.fr/** Sec : Viviane POLSINELLI Brigitte DUBOIS Insa : J.Y. TOUSSAINT Tél : 04 78 69 72 76 viviane.polsinelli@univ-lyon2.fr	M. Christian MONTES Université Lyon 2 86 rue Pasteur 69365 LYON Cedex 07 Christian.montes@univ-lyon2.fr

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

"Privacy is one of the biggest problems in this new electronic age."

– Andy Grove
Former chairman of Intel

Acknowledgements

At the beginning of this manuscript and the end of this journey, I would like to thank all those who helped me to complete this work. Firstly, I would like to thank my academic supervisor, Prof. Lionel Brunie for introducing me to research, proposing me an exciting PhD topic, and his guidance all along my thesis. I would also like to express my gratitude to Dr Sonia Ben Mokhtar, for the numerous scientific discussions, the insightful inputs on my work, and her patience and encouragements when they were the most needed.

I would also like to thank the members of the jury who evaluated my work: Prof. Richard Chbeir and Prof. Benjamin Nguyen for reviewing my manuscript, as well as Prof. Licia Capra, Dr Cédric Lauradoux and Dr Kévin Huguenin for being part of my jury and their interest in my work.

Although a PhD may sometimes feel rather lonely, I am proud to have had the opportunity to work with other people. I am especially grateful to Antoine Boutet, Cédric Lauradoux, Sara Bouchenak, Mohamed Maouche and Sophie Cerf for all those scientific discussions, for helping me to think differently, and for the fruitful collaborations we had. I would also like to thank Juliette, Benoît and all the teaching team of the marketing department (T.C.) of the IUT Lyon 1, for welcoming me warmly, and helping me during the one year I enjoyed spending there. I am also grateful for the opportunity that was given to me to join the University College London for my last year. I thank Ingemar and Emiliano for accepting me, for their guidance here, and for their support during the very last months of my PhD.

I would also like to thank my fellow PhD students, in no particular order: Albin, Vincent (B.), Diana, Mohamed, Romain, Rémi, Sophie, Tarek, Sullivan, Guido, Manel, Loïc, Sébastien, Pierre-Louis, Mazen, Guillaume, Aimene, Adnene. Thank you so much for all the passionate discussions (scientific... or not!) we shared and for the support (scientific... or not!) I received. Besides the laboratory, I really enjoyed my time in Lyon, and I am grateful for all the people I met. They are too numerous to be named individually, but they were for sure an important part of my life during these years.

Finally, I thank my parents, for giving me life and for their constant support thereafter.

Abstract

In the past decades, the usage of GPS-enabled smartphones has dramatically risen, opening the way to new exciting usages such as Google Maps, Foursquare or even Pokemon GO. All these geolocated services use the actual location of a user to give him a contextualized service. However, all these usages do not come without privacy threats. Indeed, location data that users are sending to these services can be used to infer sensitive knowledge about them, such as where they live, where they work, what they use to do in the evenings, who are their acquaintances, etc.

The revelations of Edward Snowden on the NSA's methods are a striking example of threats associated with mobility data. Starting from June 2013, this former NSA employee started revealing the methods used by this American agency to spy on people, organizations and countries. One of these programs, FASCIA, is a database that is assumed to contain trillions of location records, adding 5 billion records every day. Besides data coming from phone operators, third parties applications are also involved in this massive collection scheme. A 2008 GCHQ (a British intelligence agency collaborating with the NSA) report leaked by Snowden stated that "anyone using Google Maps on a smartphone is working in support of a GCHQ system". Though this massive and unprecedented collection scheme was organized by several governments, threats have also been demonstrated at smaller scales. For example, the Uber application was pinpointed in 2016 because it was collecting mobility data even after a ride ended, not to mention the multiple utility applications (e.g., flashlights) that require access to the GPS sensor. It is also well-known that Google, Twitter or Facebook actively use users' personal data to target their advertisements.

This is were protection mechanisms come into play. Their goal is to let users use geolocated services (e.g., Google Maps, Uber) on their mobile devices while giving them control on their privacy. These mechanisms all work by altering mobility data in some way (whether by distorting it, deleting some parts or even creating fake records), which creates a trade-off between privacy (the level of protection) and utility (the quality of service) one gets while using a protection mechanism. In this thesis, we are interested in building new protection mechanisms, featuring original and interesting properties, and in evaluating the efficiency of protection mechanisms, both existing ones and our owns.

Towards this purpose, we start by surveying existing protection mechanisms and metrics used to evaluate them. We formally define seven of them, and apply them on a state-of-the-art protection mechanism, practically demonstrating its strengths and weaknesses.

This first analysis highlights a particularly sensitive information, namely the points of interest. These are all the places where users use to spend most of their time, such as work, home or a non-profit they are involved in. This then leads us towards building a new protection mechanism, PROMESSE, whose main goal is to hide these points of interest. We demonstrate that it fulfills this objective, while offering a better spatial precision than previous state-of-the-art protection mechanisms.

Protection mechanisms tend to be configured by parameters, which highly impact their effectiveness in terms of privacy and utility. During the evaluation of PROMESSE, we found out that there is a particular configuration that is optimal, with respect to the information a user wants to hide. Consequently, we propose ALP, a solution to help users to configure their protection mechanisms. With our solution, users specify objectives in terms of privacy and utility, that are then automatically converted into actual parameters. The evaluation shows that ALP can generate good-quality configurations, while being adaptive. Indeed, as the behavior of users change (e.g., they may move to another home or accept a new job), the configuration is dynamically updated.

Finally, we introduce ACCIO, which is a framework and prototype encompassing most of our work. Its goal is to allow to easily launch location privacy experiments by only writing a few lines of JSON, thus enforcing reproducibility and driving experimentation by encouraging researchers to test alternative scenarii. It has already been used outside of the context of this thesis by other researchers interested in location privacy.

Résumé

Depuis quelques dizaines d'années, l'utilisation de téléphones contenant un capteur GPS a fortement augmenté, ouvrant la voie à de nouveaux usages tels que Google Maps, Foursquare ou même Pokemon GO. Toutes ces applications géolocalisées utilisent la localisation actuelle de l'utilisateur pour lui fournir un service contextualisé. Cependant, tous ces usages ne sont pas sans menace pour la vie privée des utilisateurs. En effet, les données de mobilité qu'ils envoient à ces services peuvent être utilisées pour inférer des informations sensibles telles que leur domicile, leur lieu de travail, leur bars préférés ou encore leurs amis.

Les révélations d'Edward Snowden sur les méthodes de la NSA sont un exemple frappant de l'utilisation qui peut être faite de ces données. À partir de juin 2013, cet ancien employé de la NSA a commencé à révéler les méthodes utilisées par cette agence américaine pour espionner des individus, organisations et nations. L'un de ces programmes, FASCIA, est une base de données qui contient des billions d'enregistrements de localisation, ajoutés au rythme de 5 milliards nouveaux enregistrements quotidiens. En plus des données venant directement des opérateurs téléphoniques, des applications tierces sont aussi impliquées dans ce schéma de surveillance. Un rapport de 2008 du GCHQ (une agence de renseignement britannique, collaborant avec la NSA), révélé par Snowden, affirmait que "toute personne utilisant Google Maps sur un téléphone participait à la collecte du GCHQ". Bien que cette collecte massive et sans précédent ait été orchestrée par plusieurs gouvernements, des menaces existent aussi à plus petite échelle. Par exemple, l'application Uber a été pointée du doigt en 2016 car elle continuait à collecter les données de mobilité de ses utilisateurs après que leur trajet ait prit fin, sans oublier les multiple utilitaires (par exemple des lampes torches) qui demandent à avoir accès au capteur GPS. Il est également de notoriété publique que Google, Twitter ou Facebook utilisent activement les données personnelles de leurs utilisateurs pour cibler leur publicité.

C'est à ce moment qu'entrent en action les mécanismes de protection. Leur objectif est de permettre aux utilisateurs de profiter des applications géolocalisées (par exemple Google Maps ou Uber) sur leurs téléphones et tablettes, tout en leur redonnant le contrôle sur leur vie privée. Ces mécanismes fonctionnent tous en altérant les données de localisation d'une façon ou d'une autre (que ce soit en les transformant, supprimant des portions ou encore en créant de fausses données). Cela donne naissance à un compromis entre vie privée (le niveau de protection) et utilité (la qualité de service) qu'on peut obtenir en utilisant un tel mécanisme. Dans cette thèse, nous nous sommes intéressés à la création de nouveaux mécanismes de protection, proposant des propriétés originales,

et à l'évaluation et l'efficacité de ces mécanismes, à la fois ceux qui existaient déjà et les nôtres.

À cette fin, nous commençons par répertorier les mécanismes de protection existants et les métriques utilisées pour les évaluer. Nous définissons formellement sept d'entre elles, et les appliquons à un mécanisme de protection de l'état de l'art afin de démontrer en pratique ses forces et ses faiblesses. Cette première analyse met en avant une information particulièrement sensible : les points d'intérêt. Ces derniers représentent tous les lieux où les utilisateurs passent la majeure partie de leur temps, comme leur travail, leur domicile ou encore une association dans laquelle ils sont investis. Cela nous conduit ensuite à concevoir un nouveau mécanisme de protection, PROMESSE, dont le but principal est de cacher ces points d'intérêt. Nous montrons qu'il remplit cet objectif tout en offrant une meilleure précision spatiale que des travaux précédents de l'état de l'art.

Les mécanismes de protection sont en général configurés par des paramètres, qui ont un grand impact sur l'efficacité des mécanismes en termes de vie privée et d'utilité. L'évaluation de PROMESSE a mis en évidence qu'il existe une configuration particulière de ce dernier qui est optimale, en fonction de l'information que l'utilisateur veut cacher. C'est ainsi que nous proposons ALP, une solution destinée à aider les utilisateurs à configurer leurs mécanismes de protection. Avec notre solution, les utilisateurs spécifient des objectifs en termes de vie privée et d'utilité, qui sont ensuite automatiquement convertis en paramètres par notre système. L'évaluation montre qu'ALP génère des configurations de bonne qualité, tout en offrant un caractère adaptatif. En effet, au fur et à mesure que le comportement des utilisateurs change (par exemple ils peuvent déménager ou changer de travail), la configuration est mise à jour dynamiquement.

Enfin, nous présentons ACCIO, qui est un prototype regroupant la majeure partie du travail de cette thèse. Son objectif est de permettre de lancer facilement des expériences destinées à étudier des mécanismes de protection, en écrivant simplement quelques lignes de JSON. Il permet de renforcer la reproductibilité des expériences et d'encourager les chercheurs à tester des scénarios alternatifs. Cet outil a déjà été utilisé en dehors du contexte de cette thèse par d'autres chercheurs.

Table of contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction

1.1 Context

Privacy has been recently in the spotlight because of a set of very unlucky events, which ultimately resulted in privacy breaches. In 2002, Sweeney was able to identify people from an "anonymized" health dataset that was released by the organization responsible for collecting it [144]. Data did not contain first and last names, but information such as sex, birth date and zip code of patients. By correlating these fields with a voters list she obtained for US$20, she could identify the governor of Massachusetts, who was the only male patient with his birth date and zip code. Similarly, in 2006, AOL voluntarily released web search logs about 650,000 of its users, with the goal to give academics a new dataset for their research. Once again, data contained no physical identity, but a unique identifier for each user and the history of their search queries as well as the links they clicked on. Despite good initial intentions, this publication resulted in a failure when it became apparent that it was possible to re-identify some users from the released data. Thelma Arnold, a 62-year-old widow living in Lilburn (Georgia, USA), was ultimately identified by queries such as "60 single men" or "landscapers in Lilburn, Ga" [12]. Yet another example is the Netflix prize, a competition organized by Netflix to improve their recommendation algorithms, which started in 2006. Netflix provided participants a training dataset containing 100 million ratings of about 500,000 users. Users were represented by an integer, whereas film names were clearly accessible. Once again, researchers were able to de-anonymize part of this dataset, by using external knowledge publicly accessible on IMDB [109].

These examples have definitively fostered research on privacy in the last two decades. In this thesis, we focus on a specific topic which is *location privacy*, i.e., privacy applied to mobility data. Indeed, location privacy comes with its specific challenges and solutions. More and more people carry handheld devices every day (e.g., smartphones, tablets) equipped with geolocational capabilities (e.g., embedded GPS chips), allowing them to access a wide variety of online services on the move. These services, often called *location-based services* (later abbreviated LBSs), provide users with contextual information depending on their current location. We give here a non-exhaustive list of common use cases that have been enabled by the rise of LBSs.

- *Directions & navigation applications:* These services allow users to get directions to (almost) any destination, and then to navigate towards it by simply following spoken instructions. Location data is used to provide real-time directions, recalculated as the user is moving. Well-known players here include Google Maps [59] and Waze [150].
- *Weather applications:* These services provide current weather conditions as well as forecasts. Location data is used to give the user relevant information for the city he is currently located in. Yahoo! Weather [154] is an application providing such a service on Android and iOS.
- *Venue finders:* These services give users information about interesting places in the user's vicinity. Most of the time, they include recommendations based on other persons' experience. Location data is used to show only places in immediate user's neighborhood. Foursquare [44] and Yelp [156] are two applications helping to find such interesting places, with an added social dimension.
- *Social games:* These services turn any urban walk into an ever-changing game, where each new place becomes a new playground. Location data is used to make

the game evolve depending on the user's city and his immediate surroundings, sometimes allowing to compete with nearby other users. Examples of such games are Pokemon GO [112] and City Domination [29].

- *Crowd-sensing applications:* These services enable participatory sensing, where a crowd of users use their smartphones to monitor their environment and share their results through an LBS server. Crowd-sensing benefits to a large variety of domains such as traffic monitoring (e.g., Nericell [104]) or health monitoring (e.g., PEIR [108]). APISENSE [64] and Funf [7] are two applications allowing to run crowd-sensing campaigns.

Whatever their exact nature, LBSs share similar objectives: on the one side, they use location data provided by their users to provide them with an accurate and contextual service; and on the other side, they make business out of the collected data and use it to continuously improve their service. Mobility data gathered by such companies can then be either used internally (e.g., for marketing purposes), or be given/sold to external parties (e.g., release of jogging/cycling traces, publication of pictures with location metadata on Flickr). Indeed, the market related to LBSs is enormous: the total revenue of the US-only LBS industry was already estimated to $75 billion in 2012 [66]. Furthermore, the high value of the location data leads many applications to commercially exploit the collected data for analysis or advertisement targeting purposes. For instance, Foursquare Enterprise [42] is a service of Foursquare offered to enterprises, whose goal is to give them tools to better understand their business. For that, they use the huge amount of mobility data that Foursquare is collecting every minute to give businesses insights about who is visiting their stores and passing in front of them, as well as for competitors.

Obviously, such an amount of information does not come without privacy threats. First, some applications exploit private information about users stored on their mobile devices. For instance, TaintDroid [40] and Mobilitics [4] showed that several high-rated applications are suspected to exfiltrate sensitive data to third parties. But even with non-malicious behaviors, sharing so much mobility data can lead to privacy breaches. Indeed, users are often not aware of the quantity of sensitive knowledge that can be inferred from their mobility data. Analyzing mobility traces of users can reveal their *Points of Interest* [49] (later abbreviated POIs), which are meaningful places such as home or work. It can also reveal the other users they frequently meet [136], or lead to predicting their future mobility [134]. It is also possible to semantically label these mobility traces [84] in order to infer the actual user's activity (e.g., working, shopping, watching a film). Besides the continuous tracking of a user's activities, POIs can lead to leak even more sensitive information (e.g., religious or political beliefs if one regularly goes to a worship place or to the headquarters of a political party). As an example, it is possible to find out which taxi drivers are Muslim by correlating the time at which they are in pause with mandatory prayer times [45].

Consequently, a number of solutions have been investigated in the literature to protect users' privacy while still allowing them to enjoy LBSs. These solutions are called *Location Privacy Protection Mechanisms* (later abbreviated LPPMs). There is a rich literature about existing LPPMs. Some of them are rather generic and can adapt to a lot of situations while others are very specific to a single use case. LPPMs rely on a wide array of techniques, ranging from data perturbation (e.g., [9, 54, 75]) to data encryp-

tion (e.g., [97,123,163]), and including fake data generation (e.g., [77,117,131]). In this manuscript, we distinguish between three classes of use cases for LPPMs. In *real-time use cases*, users query an LBS and expect an immediate answer. We include in this category the usage of navigation applications, weather applications, venue finders and social games. The main challenge for real-time LPPMs (e.g., [9, 54, 123]) is that they only have at their disposal actual and historical locations; they obviously do not know the future state of the system. Besides the scope of our work, the literature also contains more radical approaches to tackle these use cases by proposing to replace existing LBSs with new privacy-by-design architectures[1] (e.g., [62,119]). *Offline use cases* come into play once an LBS has collected mobility data and wants to publish it, whether it is for commercial or non-profit purposes. For example, an enterprise may give mobility data it gathered to its marketing department in the hope to find out interesting trends. More threatening, such data may also be released in the open to researchers, as it was the case with the aforementioned AOL and Netflix datasets. The usage of any LBS may trigger an offline use case once the organization collecting data wants to publish it in a privacy-preserving way. Although such data collections are subject to privacy policies, the latter are usually very liberal and regularly pinned down. For example, the French agency regulating Internet liberties criticized Google's 2014 updated privacy policy as non-compliant with European directives [30]. Consequently, the usage of offline LPPMs could definitely improve offered privacy, while helping companies to comply with regulatory laws. Instead of protecting locations on-the-fly, offline LPPMs (e.g., [2, 60, 103]) protect whole mobility datasets at once, possibly leveraging the knowledge of the behavior of all users in the system to apply more efficient and subtle schemes. Besides the scope of our work, offline use cases also include interactive querying of gathered mobility data[2] (e.g, [117]). In *batch use cases*, users regularly send their data to an LBS (e.g., every hour) and expect it to publish back aggregated results. We include in this category the usage of crowd-sensing applications [107]. Batch use cases are a middle-ground situation between real-time and offline use cases. Batch LPPMs differ from real-time LPPMs in that they are less sensitive to latency, and they send more data at once. They also differ from offline LPPMs because they do not have the global knowledge of where all users are located at their disposal, whereas offline LPPMs protected entire datasets and hence know where everyone is located. Consequently, real-time LPPMs can be used for batch use cases too, as well as offline LPPMs that protect each user independently of the others. There are also LPPMs designed specifically for batch use cases, aiming at protecting small batches of data belonging to a single user at once. For example, [75] is targeted towards protecting trajectories, i.e., small portions of data belonging to a single user; trajectories are protected as a whole (instead of protecting each point independently as with an online LPPM) and independently of the other users.

Figure 1.1 summarizes the interactions between our three families of uses cases and involved actors. As depicted, we distinguish between two phases when using an LPPM: what happens *online*, during the *collection*, i.e., between a user and an LBS, and what happens *offline*, during the *publication*, i.e., between an LBS and an analyst. Depending

[1] Although these solutions look elegant and promising to us, they serve a slightly different goal than ours. In our work, we place as a priority to achieve interoperability with existing LBSs, which is impossible with those approaches.

[2] In these schemes, analysts can interactively submit queries (e.g., SQL) and get answers, with respect to some privacy policy. However, it does not fit our goal to protected entire datasets, which gives analysts more flexibility instead of being limited by the expressiveness of a query language.

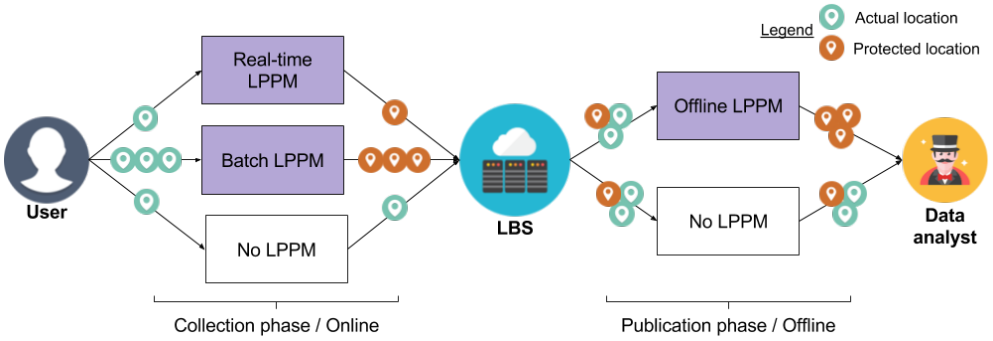


Figure 1.1: Summarization of the three families of use cases and involved actors, and the way they interact.

on their nature, LBSs fall either in the real-time or batch family of use cases. Furthermore, offline use cases appear as soon as one of these LBSs is willing to publish the gathered mobility data in a privacy-preserving way. At this point, the LBS has received either protected or unprotected data, depending on whether users were using an online LPPM before sending their mobility data to the LBS. This is perfectly fine, though we will not make the distinction in this manuscript about whether an offline LPPM is acting on protected or unprotected data. We consider in those cases that the LBS has to protect all the data in the same manner, as if it was working with unprotected data. In other words, real-time/batch use cases and offline use cases are complementary and handled by different parties: applying an online LPPM is indeed the responsibility of the user, whereas applying an offline LPPM is the responsibility of the LBS itself.

In all use cases, we consider the adversary to be *honest-but-curious*. In real-time and batch use cases, the adversary is the LBS itself. We consider the latter provides to the user a service at the best of its abilities ("honest"), but while continuously analyzing gathered data in order to infer knowledge about users ("curious"). For example, this is exactly what Google, Facebook or Twitter are doing. They provide high-quality and contextualized services to their users, but in exchange they make money from targeted advertisements exploiting users' personal data. In offline use cases, data has already been collected by an LBS, which is not anymore considered as the adversary. In these use cases, the adversary is any person having access to data published by the LBS. We consider that these persons have a legitimate access to published data ("honest"), but are actively trying to break users' privacy, beyond the initial intent of the data publisher ("curious"). This is what happened with the aforementioned Netflix Prize scandal [109], where the data was published with the aim that researchers would design or improve machine learning algorithms, but it was actually de-anonymized.

Because there are so many different LPPMs and use cases, researchers have proposed a large variety of metrics to evaluate them. These metrics can be divided in three categories. *Privacy metrics* quantify the level of privacy a user can expect while using a given LPPM. One popular way to evaluate privacy is to compare the effect of a privacy attack before and after applying an LPPM (e.g., [139]). *Utility metrics* measure the usefulness (also called quality of service) that can still be obtained while using an LPPM, which largely depends on the targeted LBS and its role. There is an inherent trade-off between privacy and utility. Indeed, if no mobility data is sent, privacy is perfectly preserved, while utility is null. Conversely, sending unprotected data results

in a perfect utility at the cost of no privacy. Finally, *performance metrics* measure the algorithm efficiency or cost of a given LPPM. Typical performance metrics are the execution time, the ability to scale or the tolerance to faults. These metrics are orthogonal and do not participate to the privacy/utility trade-off, but still are important because they impact the usability of LPPMs.

1.2 Problem statement

In this context, the research problem we tackle in this thesis can be summarized with the following problem statement:

> How to build and evaluate privacy- and utility-preserving location privacy protection mechanisms?

We decompose this problem into four sub-problems.

P1 – Evaluating and comparing LPPMs.
Because there is already a very rich literature about LPPMs[3], evaluating and comparing the guarantees they offer in terms of both privacy and utility turns out to be a cumbersome task. Indeed, each paper proposes its own evaluation metrics, often tailored for a specific LPPM and use case, and evaluates its solution against competitors with those metrics. This results in a large number of metrics in the literature, and making a fair comparison between different LPPMs looks like comparing pears and apples. Consequently, there is no standard methodology to evaluate such different LPPMs, which is a weakness of the literature. Moreover, some LPPMs come with theoretical guarantees, which give a bound on the impact of a privacy leak. This is a strength, because such a guarantee is generic and defined independently from any specific privacy attack, but can also reveal to be a weakness, because some attacks may still be possible and efficient and this very fact is hidden.
We therefore consider the following research questions:

- **P1.1** – Which metrics to use to evaluate an LPPM in terms of privacy, utility and performance?
- **P1.2** – What is the practical impact, in terms of privacy and utility, of theoretical LPPM guarantees?

P2 – Designing privacy- and utility-preserving LPPMs.
Although many LPPMs have been proposed by researchers, there are still unexplored areas. LPPMs attempt to draw a trade-off between privacy and utility. On the privacy side, we found out that POIs are of a huge importance and should be protected. Moreover, we found out that LPPMs tend to introduce a large spatial distortion (i.e., protected locations are far away from original ones), because they are usually designed with a focus on privacy objectives. Although they are evaluated in terms of both privacy and utility, they are not designed from the ground with the goal of maximizing an effective utility. As a result, privacy is well preserved, but utility often remains best-effort.
We therefore consider the following research questions:

[3] In a recent survey we wrote, we categorized no less than 55 different LPPMs across all use cases.

- **P2.1** – Which under-explored privacy and utility guarantees are worth considering when designing a new LPPM?
- **P2.2** – How to design an LPPM considering both privacy and utility as equally important objectives?

P3 – Configuring LPPMs.
Research papers usually evaluate LPPMs after they have been carefully parametrized by people who created them. Choosing a correct parametrization is indeed crucial; if badly configured, an LPPM can become totally ineffective. However, in real-life, we do not always have a location privacy expert available to help us with LPPM parametrization. Indeed, it happens that this task is far from easy, because configuration parameters can be numerous (e.g., up to 7 different parameters in [2]) and obscure for non-experts (e.g., the unitless ϵ parameter of differentially private LPPMs such as [9], whose impact follows a logarithmic scale). Moreover, as time passes, user's behavior is likely to change. A user may move to another city, change his favorite cinema, become involved in politics, make new friends, etc. Consequently, a configuration may become obsolete, and needs to be continuously re-evaluated and adapted. Furthermore, not all places have the same importance: going to a cinema is far less sensitive than going to a hospital, being at home reveals much more than walking in a crowded mall. Practically, this means that the parametrization of an LPPM cannot be determined once for all but has to be adaptive, with respect to data being actually protected.
We therefore consider the following research questions:

- **P3.1** – How to allow a final user to specify rich and expressive objectives in terms of privacy and utility?
- **P3.2** – How to transform these objectives into an effective set of parameters?
- **P3.3** – How to handle several different LPPMs?
- **P3.4** – How to adapt generated configuration to current user's behavior?

P4 – Experimenting with and productionizing LPPMs.
LPPMs come with their own system model and assumptions. For example, they can be designed either for real-time or offline use cases; they assume either a discrete or continuous time and space; they work either with local-only data or interact with other users to enrich their local knowledge. Consequently, there is no standard model to evaluate and compare such different LPPMs in a unified manner. Furthermore, we lack of production-grade and publicly available implementations of location privacy algorithms. This slows down innovation, as researchers have to reimplement again and again similar code, and certainly is a barrier for enterprises to consider integrating privacy in their processes. Finally, because implementations are not always made available and papers are not always precise enough, there is a lack of reproducibility of results[4].
We therefore consider the following research questions:

- **P4.1** – How to design a framework that would allow to evaluate and fairly compare different LPPMs?
- **P4.2** – How to allow researchers to easily express and launch large-scale location privacy experiments?
- **P4.3** – How to improve reproducibility of location privacy research results?

[4]Indeed, this a not a problem unique to our discipline, as regularly pointed out, e.g., [10].

1.3 A note on legal aspects

In this section, we give a hint about European Union (EU) and French laws regarding privacy and data protection. Our work in this thesis is purely technical; legal aspects are outside of the scope of this thesis and will not be evoked anymore later.

In the EU, the Data Protection Directive [148], adopted in 1995, was first regulating the protection of personal data. It is a directive, which means it has to be translated into a law in each of the EU countries to take effect. This directive is in the process of being superseded by the General Data Protection Regulation (GDPR) [149], adopted in April 2016 and planned to be enforceable in May 2018. The GDPR is a regulation, which means it has immediately the same power than a law in all EU countries. It applies to every organization collecting or processing data from EU residents, even if the organization itself is not based in the EU. Notably, it makes these organizations responsible and accountable for the way personal data is managed. They should be able to demonstrate that privacy-preserving measures have been integrated in their processes. Moreover, privacy must be implemented by design and by default.

In France, the CNIL (National Commission on Informatics and Liberty) is a governmental body responsible for enforcing French and EU laws regarding data privacy. Its current role includes gathering declarations from organizations collecting or processing personal data, warning non-compliant organizations, issuing fines and reporting to the judicial entities. The CNIL will become the French Supervisory Authority for the GDPR, cooperating with Supervisory Authorities of other EU countries. Its role will evolve, notably because declarations will no longer be required. The responsibility of doing privacy-preserving data processing will be upon the organizations, which are accountable for it. The CNIL, as a Supervisory Authority, will be able to control organizations, and check whether they are actually conformant with the GDPR. The amount of fines will also increase, up to 20 million euros or 4 % of the worldwide revenue of the guilty organization.

The GDPR does not require nor recommend any technical solution, only pseudonymization[5] is mentioned as an exemple. This is exactly where LPPMs fit: they provide a practical way to enforce this regulation. Therefore, technical work about location privacy is complimentary of the legal framework, and will likely become required, as these laws strengthen users's rights and oblige organizations to take technical measures to protect these rights.

1.4 Contributions summary

This manuscript is made of four contributions, each one addressing one of the research problem raised in Section 1.2.

C1 – A state-of-the-art of LPPMs & practical assessment. *In response to P1.* To clarify what exists and what are the limitations of actual solutions, we first present a state-of-the-art of LPPMs. We put an emphasis on metrics used to evaluate them

[5]Pseudonymization consists in replacing any personally identifying field, such as a first/last name or social security number, by a pseudonym, such as a random integer.

in terms of privacy, utility and performance. We then extract a list of eight evaluation metrics that will be used throughout the rest of this manuscript. They are either state-of-the-art metrics that we define formally in our model, or custom metrics that we crafted ourselves. These metrics provide a consistent view with respect to our research problem and to the solutions that we propose. Furthermore, we use these metrics to practically assess the efficiency of a representative state-of-the-art LPPM, geo-indistinguishability [9]. This analysis shows that, despite theoretical guarantees, a large proportion of points of interest are still exposed while using the latter LPPM.

C2 – A speed smoothing LPPM. *In response to P2.*
With the outcome of the previous analysis of geo-indistinguishability in mind, we propose PROMESSE, a new kind of LPPM whose goal is specifically to hide points of interest, while drastically reducing spatial distortion traditionally coming from using an LPPM. To achieve this, PROMESSE relies on speed smoothing, a novel technique that makes the user appear to be constantly on-the-move with a constant speed. PROMESSE is designed to be used in offline use cases, though it supports batch use cases too. We implement and evaluate it against two other state-of-the-art LPPMs, and show that it performs significantly better when it comes to hiding points of interest and adds almost no spatial distortion.

C3 – A system to assist in configuring LPPMs. *In response to P3.*
We introduce ALP, a solution assisting the users to configure their LPPM. ALP allows users to express their objectives in terms of privacy and utility, by providing them with a library of metrics. For example, they can express objectives such as "I do not want my home to be identifiable with a precision higher than 200 meters" (a privacy objective). Then, from a set of objectives, our solution proposes a "good enough" configuration for any parametrizable LPPM, taking into account current data being protected. This configuration is determined by using actual data to protect, thus guaranteeing that is will be tuned with respect to the sensitivity of the data under consideration. ALP is designed with batch use cases in mind, though it supports offline use cases too. Our solution is shown to provide more efficient configurations than static ones in terms of privacy and utility, and to ease the burden of configuring an LPPM for non-technical users.

C4 – A location privacy experimentation framework. *In response to P4.*
We propose ACCIO, a framework that proposes a unified model to represent and implement LPPMs (across all use cases) and these metrics. ACCIO comes with a JSON-based language to easily express simple as well as complex experiments. We demonstrate its flexibility with a case study, starting from a simple situation and enriching it block by block, up to three different LPPMs evaluated with two different datasets and five different metrics. ACCIO encourages researchers to test unexplored scenarii by alleviating the burden of launching experiments, even large-scale ones. It is designed to be extensible and it has already been used outside of the context of this thesis by other researchers.

These contributions were the basis for five publications in international conferences and workshops.

- Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar and Lionel Brunie. Adap-

tive Location Privacy with ALP. In *Proceedings of the 35th Symposium on Reliable Distributed Systems (SRDS)*, September 2016, Budapest, Hungary. pp.269-278.

- Vincent Primault, Sonia Ben Mokhtar, Cédric Lauradoux and Lionel Brunie. Time Distortion Anonymization for the Publication of Mobility Data with High Utility. In *Proceedings of the 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, August 2015, Helsinki, Finland. pp.539-546.
- Vincent Primault, Sonia Ben Mokhtar and Lionel Brunie. Privacy-preserving Publication of Mobility Data with High Utility. In *Proceedings of the 2015 35th IEEE International Conference on Distributed Computed Systems (ICDCS)*, June 2015, Columbus, Ohio, USA. pp.802-803.
- Nicolas Haderer, Vincent Primault, Patrice Raveneau, Christophe Ribeiro, Romain Rouvoy and Sonia Ben Mokhtar. Towards a Practical Deployment of Privacy-preserving Crowd-sensing Tasks. In *Middleware Posters and Demos '14*, December 2014, Bordeaux, France. pp.43-44.
- Vincent Primault, Sonia Ben-Mokhtar, Cédric Lauradoux and Lionel Brunie. Differentially Private Location Privacy in Practice. In *Proceedings of the 2014 Mobile Security Technologies Conference (MoST)*, May 2014, San Jose, California, USA.

There was also one other publication in an international conference that builds on the tools and methodology developed in this manuscript, though not directly part of this thesis.

- Sophie Cerf, Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, Robert Birke, Lydia Y. Chen, Sara Bouchenak, Nicolas Marchand and Bogdan Robu. Achieving privacy and utility trade-off in mobility database with PULP. In *Proceedings of the 36th Symposium on Reliable Distributed Systems (SRDS)*, September 2017, Hong Kong, China.

1.5 Organization of the manuscript

The remaining of this manuscript is organized as follows. Chapter 2 formalizes the location privacy problem and presents state-of-the-art LPPMs. Chapter 3 introduces a library of evaluation metrics and shows they usage on an LPPM. Chapters 4, 5 and 6 introduce our three remaining contributions, besides state-of-the-art. Chapter 7 concludes this manuscript and presents future work.
More specifically, each chapter revolves around the following aspects.

- **Chapter 2** is a thorough introduction to location privacy. We provide an analysis of state-of-the-art location privacy attacks, thus highlighting the specific problems LPPMs have to solve. We also introduce a formalization of our problem with notations that will be instrumental in the remaining of this manuscript. Finally, we survey state-of-the-art LPPMs that are relevant to the research problems we address.
- **Chapter 3** resumes the literature work by presenting metrics that will be used in this thesis to evaluate LPPMs. Metrics either come directly from previous research papers, are inspired by previous papers or are original contributions. Even for state-of-the-art metrics, we had to (re)define them precisely in our system model.

Finally, we show a first practical application of these metrics by applying them on a state-of-the-art LPPM, geo-indistinguishability [9]. We leverage our metrics to outline strengths and weaknesses of this particular LPPM, and motivate our subsequent contributions from these results.

- **Chapter 4** introduces PROMESSE, a new LPPM designed specifically to protect users' POIs, following the results of our previous analysis that highlighted the importance of POIs for privacy. On the utility side, our LPPM is designed with the goal of achieving a better utility than classical perturbation-based LPPMs that add some noise to users' locations. We study an experimental evaluation, comparing PROMESSE with two representative state-of-the-art LPPMs, featuring similar results in terms of privacy while achieving significantly better results in utility.
- **Chapter 5** unveils ALP, a solution assisting users during the configuration of LPPMs by using an objective-driven approach. These objectives are then converted into a configuration for any parametrizable LPPM, by using a classical optimization routine. Moreover, ALP dynamically adapts the generated configuration to underlying data, making it adaptive as a user moves and his behavior changes. Our experimental evaluation shows that we are able to produce good-quality configurations and to outperform statically configured LPPMs.
- **Chapter 6** details ACCIO, a framework that was developed to assist researchers willing to study location privacy. ACCIO provides a unified model allowing to represent very different LPPMs and metrics, and comes with implementations for the ones used in this thesis. Its JSON-based language enables to express experiments, including large-scale ones, with a well-known and easy-to-use syntax. We show that ACCIO can effectively elegantly handle a large variety of scenarii, even ones featuring original combinations of LPPMs and metrics.
- **Chapter 7** concludes this manuscript and presents research perspectives and open challenges.

CHAPTER 2

Location Privacy: A State of the Art

2.1 Introduction

As already outlined, using LBSs does not come without privacy threats. Curious LBSs can infer a lot of sensitive information from data provided by their users, such as their activities (e.g., [84]), their social relationships (e.g., [17]) or even predicting where they will be in the future (e.g., [134]). A prominent threat that we are particularly interested in rely on the extraction of *points of interest* (abbreviated POIs) [49, 81], which are significant places where users spend most of their time like a work place, a home, a mall, etc. To limit these privacy problems, many *location privacy protection mechanisms* (abbreviated LPPMs) have been proposed in the literature. Their goal is to protect users' location data while still allowing them to enjoy geolocated services. LPPMs come in a large variety in terms of architectures (e.g., purely local, peer-to-peer), of tackled use cases (real-time, batch or offline) and offered guarantees, each of them with their pros and cons. To face this diversity and allow us to compare LPPMs, we therefore begin by introducing a formalization of the location privacy problem. This formalization will be instrumental in the remaining of this manuscript, and the basis for all our algorithms, thus unifying our discourse and contributions under a common framework.

LPPMs are usually evaluated using metrics, which can be divided into three families: *privacy metrics*, *utility metrics* (i.e., the quality of service) and *performance metrics*. Because privacy and utility work in opposite ways, there is an inherent trade-off between them. Indeed, if someone is not using any LBS, his location privacy is perfectly protected but he gets nothing useful from these services (obviously). Conversely, if someone is using an LBS without any LPPM, his location privacy is null while he gets a maximal utility[1]. For now, as we do not have yet an LPPM providing a perfect privacy and a perfect utility, there is always a cursor to adjust between the two, with choices and concessions to make. Performance is orthogonal to the privacy/utility trade-off, but still has to be considered because of its impact on the user experience. Metrics are hence used to evaluate LPPMs along those three axes, although there is no standard evaluation methodology; each paper tends to use its own specifically tailored metrics. Therefore, we perform in this chapter an analysis of metrics used to evaluate LPPMs in the literature, and propose a first classification inside each family (related to research problem **P1**). Moreover, we provide an up-to-date review of the literature on LPPMs (related to research problem **P2**), and we categorize them into five categories, according to the techniques they use: mix-zones, generalization-based, dummies-based, perturbation-based and rules-based. We furthermore highlight for each LPPM the metrics that were used to evaluate it, in terms of privacy, utility and performance, and we study their architecture and the associated impacts (e.g., latency or scalability).

The remaining of this chapter is structured as follows. We start by introducing our system model in Section 2.2. We then give a brief overview about location privacy threats in Section 2.3. We present how LPPMs are evaluated in Section 2.4 before surveying state-of-the-art LPPMs in Section 2.5. We eventually present other related approaches in Section 2.6 before summarizing the content of this chapter in Section 2.7.

[1] It is actually the situation for most LBSs' users right now.

In a nutshell. Our original contributions in this chapter are the following:

- A formalization of the location privacy problem;
- A survey of LPPMs across our three use cases, organized into five categories, along with related metrics.

2.2 System model

This section presents a formalization of the location privacy problem by defining precisely what "mobility data" is and how an LPPM interacts with it. Roughly speaking, mobility data is collected under the form of *events* (Section 2.2.1), aggregated inside *datasets* and *mobility traces* (Section 2.2.2). LPPMs are later applied on datasets to protect them (Section 2.2.3), and evaluated with metrics (Section 2.2.4). We also define the notion of *point of interest*, which will be one of our prominent concerns in this thesis (Section 2.2.5). Eventually, we clearly specify our assumptions in Section 2.2.6.

The notations we introduce all along this section are summarized in Table 2.1.

Table 2.1: Notations.

$\mathcal{U}$	Set of all user identifiers
$\mathcal{L}$	Set of all locations (including POIs)
$d_{\mathcal{X}}$	Distance function between two locations
Θ	Set of all timestamps (with total order)
$\mathcal{E}$	Set of all possible events
$\mathcal{D}$	Set of all possible datasets
$d_i \in \mathcal{E}$	i-th event in dataset $d \in \mathcal{D}$, $i \in \mathbb{N}$
$\mathcal{D}_u \subset \mathcal{D}$	Set of all possible traces of user $u \in \mathcal{U}$
$d_u \in \mathcal{D}$	Trace of user $u \in \mathcal{U}$ inside dataset $d \in \mathcal{D}$
Π	Set of all possible LPPMs
$\mathcal{M}$	Set of all possible metrics
$\hat{x}$	Protected version of x (i.e., after applying an LPPM)

2.2.1 Event

The most basic information we are collecting in our model is called an *event*, which is composed of a user identifier, a location and a timestamp. We also define special types of events, namely *call detail records* and *check-ins*.

Physical and logical user. An individual, also called a *physical user*, is a person with an identity (e.g., first and last name, social security number). A *logical user*, is a consistent source of mobility data associated with a single physical user (e.g., a smartphone, a GPS embedded inside a car). A physical user can be associated with several logical users (e.g., a single physical user can use several devices). Although this distinction is worth being made, we consider logical users as the canonical "users" in the rest of this manuscript, so we will explicitly refer to "physical users" when talking about them. Typically, a user identifier can be an IP address, a numerical ID or a random string. The set of all user identifiers is noted $\mathcal{U}$.

Space and time. A *location* is a point at the Earth's surface. It can be represented in many ways, e.g., a latitude/longitude pair or a projection in Cartesian coordinates. We abstract this by considering locations as elements of a set $\mathcal{L}$, equipped with a distance function $d_{\mathcal{X}} : \mathcal{L}^2 \to \mathbb{R}^+$. For example, when representing locations in Cartesian coordinates, we may consider $d_{\mathcal{X}}(\ell, \ell') = \|\ell - \ell'\|_2$, i.e., the Euclidean norm. A *timestamp* is an absolute instant in time (i.e., it does not include timezone information). A timestamp can be represented in several ways, like a Unix timestamp or an ISO 8601-formatted string. We abstract this by considering timestamps as elements of a set Θ, equipped with a total order (according to the chronological order). Although we always consider the case where time and space are continuous, this general representation works also with discrete values.

Event. An event represents the location of a given user at a given time. More specifically, it is a triplet $\langle u, \ell, t \rangle$, where $u \in \mathcal{U}$ is the identifier of the user who generated the event, $t \in \Theta$ is the timestamp at which the event occurred and $\ell \in \mathcal{L}$ is the location where the event happened. The set of all possible events is noted $\mathcal{E}$. Let $\mathbf{user} : \mathcal{E} \longrightarrow \mathcal{U}$, $\mathbf{loc} : \mathcal{E} \longrightarrow \mathcal{L}$ and $\mathbf{time} : \mathcal{E} \longrightarrow \Theta$ be functions to access the attributes of an event, i.e., $\forall e = \langle u, \ell, t \rangle \in \mathcal{E}$, $\mathbf{user}(e) = u$, $\mathbf{loc}(e) = \ell$, $\mathbf{time}(e) = t$. $\mathcal{E}$ comes with a total order, which is the taken on the timestamps; more precisely $\forall (e_1, e_2) \in \mathcal{E}^2, e_1 \leq e_2 \Leftrightarrow \mathbf{time}(e_1) \leq \mathbf{time}(e_2)$.

Call detail record. A call detail record (abbreviated CDR) is a particular type of event that is produced by a cell phone operator. A CDR is created for each phone call performed by a user as well as some other operations such as sending a text message. It associates to a user identifier the time at which a communication (i.e., voice call or text message) occurred and the location, determined from the location of the cell tower the mobile device was connected to. Although we do not consider them in our model, CDRs are actually more detailed and come with more metadata, such as the duration of the communication (if it was a phone call) or the recipient's identifier. With CDRs, location comes at a coarser grain than classical events whose location is determined via other geolocation means such as GPS, and at a lower sampling rate, because it depends on the frequency at which users use their phone during the day.

Check-ins. A check-in is another particular type of event that is related to social networks (e.g., Swarm [43]). They are generated by users voluntarily informing the LBS that they are at a given location. This way, their friends are notified that they actually are at this cool restaurant or at a terrific concert of their favorite band. Being an event, a check-in contains the identifier of the user who checked-in, the time at which he did and the place at which he was at this moment. Check-ins usually have a lower sampling rate than classical events because they are only generated at the user's will. However, check-ins are particularly useful because they usually can be combined with other knowledge such as a graph of social relationships between users.

2.2.2 Trace & dataset

Although we can work directly with events, most of the time they are manipulated via mobility traces and datasets.

Dataset. A set of events is called a dataset. All events inside a dataset usually come

from a single collection campaign and feature similar characteristics (e.g., the same sampling rate or the geographical area). Formally, we note $\mathcal{D}$ the set of all possible datasets, defined as[2]:

$$\mathcal{D} = \mathcal{P}(\mathcal{E}).$$

Because there is a total order defined on events, datasets are also ordered. We note $d_i, i \leq |d|$ the i-th event of a dataset $d \in \mathcal{D}$, according to this order. Moreover, for any function $\mathbf{f}$ defined over events (e.g., **user**, **loc**), we note $\vec{\mathbf{f}}$ its image over a set of events $d \in \mathcal{D}$, defined such that $\vec{\mathbf{f}}(d) = \{\mathbf{f}(e) \,|\, e \in d\}$.

Mobility trace. A mobility trace (usually simply referred to as a *trace*) is a set of events all belonging to the same user. Therefore, a mobility trace is a subset of a dataset, and a dataset can be partitioned into a set of non-overlapping traces, one for each user. We note $\mathcal{D}_u$ the set of all possible traces belonging to user $u \in \mathcal{U}$, defined as:

$$\mathcal{D}_u = \{d \in \mathcal{D} \mid \forall e \in d, \mathbf{user}(e) = u\}.$$

Similarly, we note d_u the mobility trace of user $u \in \mathcal{U}$ inside a dataset $d \in \mathcal{D}$:

$$d_u = \{e \in d \mid \mathbf{user}(e) = u\}.$$

2.2.3 Location-privacy protection mechanism

Because of the privacy threats associated with mobility data collection, researchers have developed location privacy protection mechanisms (abbreviated LPPMs, or simply referred to as *protection mechanisms*). An LPPM is a function transforming a dataset into another dataset, formally $\mathcal{D} \longrightarrow \mathcal{D}$. The set of all LPPMs is noted Π. A non-protected dataset, i.e., a dataset on which an LPPM has never been applied, is called an *actual* dataset; it is usually the input of an LPPM. A dataset that is produced by an LPPM is called a *protected* dataset. Similarly we use the term of actual traces (resp. events) for traces (resp. events) belonging to an actual dataset, and the term of protected traces (resp. events) for traces (resp. events) belonging to a protected dataset. In other words, an LPPM produces a protected dataset from an actual dataset. As a convention, we note protected events and datasets with a hat, e.g., $\hat{e} \in \mathcal{E}$ and $\hat{d} \in \mathcal{D}$.

With this definition we consider the most general definition of an LPPM; depending on the use cases under consideration, its effective usage will vary. In offline use cases, when mobility data coming from multiple users is already aggregated on the LBS-side, LPPMs consume and produce datasets containing data for all users inside the system. In online use cases, users protect only their own mobility data and do not usually even have access to mobility data of other users. Therefore, online LPPMs consume and produce a single trace, i.e., an element of $\mathcal{D}_u \subset \mathcal{D}$ for some user $u \in \mathcal{U}$. More precisely, batch LPPMs work with traces usually containing multiple events while real-time LPPMs work with singleton datasets, i.e., composed of a single event. A protected dataset can possibly be empty if it was not possible to publish anything without endangering privacy.

Some LPPMs may need background knowledge to work. It may be generic information about the user's surroundings (e.g., population density, venues around) or information

[2]We remind that the power set of a set S, noted $\mathcal{P}(S)$, is the set of all subsets of S, including the empty set and S itself.

about the system state, i.e., past or actual locations of some (possibly all) users. For example, there exist LPPMs that implement a protocol to communicate with other users and get access to their partial traces (e.g. [28, 54]). Such knowledge is not included as part of the LPPM interface, because it may vary greatly from one LPPM to another.

2.2.4 Metric

An important part of the approach we develop in this thesis rely on ways to evaluate LPPMs, both state-of-the-art ones and our own proposals. Towards this purpose, we use evaluation metrics. The set of all possible metrics is noted $\mathcal{M}$, a metric being a function whose goal is to evaluate the quality of a protected dataset compared to the actual one, i.e., $\mathcal{D}, \mathcal{D} \longrightarrow \mathbb{R}^n, n \in \mathbb{N}$. The first input dataset is a protected dataset and the second one is the actual dataset from which the protected dataset was derived. The actual dataset may not always be needed, e.g., the entropy is a privacy metric [37] that can be computed directly on the protected dataset, while it may be required, e.g., for distortion-based metrics [138]. The output is a vector of real numbers, whose exact meaning depends on the metric; this vector can possibly be reduced to a singleton. Consequently, evaluation results are often depicted as a cumulative distribution function, allowing to precisely figure out the distribution of values. When it makes sense, we may also only provide aggregated information, such as the average or median value.

Some metrics may need background knowledge to work, e.g., topological information about the surroundings of a location or white pages. Such knowledge is not included as part of the metric interface, because it may vary greatly from one metric to another.

Moreover, in order to simplify the definition of our metrics, we make an important assumption on the LPPM: we assume that LPPMs we formally evaluate in the remaining of this thesis do not change the user identifier. It is not a limitation of our model *per se*, but rather a simplification because we do not evaluate such LPPMs in this thesis. Indeed, as soon as an LPPM is allowed to change user identifiers, it becomes much more difficult to compare information at the trace level, as a user a in the actual dataset may become a user b in the protected dataset.

2.2.5 Point of interest

Points of interest (abbreviated POIs) are places where a user regularly spends some time, such as home/work places, a cinema he goes to or a non-profit organization he is involved in [49, 51]. They are "of interest" because they characterize the mobility patterns of users; as such, they are a sensitive information that users generally do not want to be leaked. For instance, points of interest can represent particularly sensitive places such as home, a worship place or the headquarters of a political party. Although they can contain rich information (e.g., the time spent inside the POI or the number of times the user went to it), we consider in our model points of interest as being pure locations, i.e., elements of $\mathcal{L}$. We leave as future work to enrich this definition and develop new algorithms relying on additional properties.

POIs are usually extracted from mobility traces (and not whole datasets) to characterize the mobility of individuals. A popular way to get them is by using a density-based

clustering algorithms [80], which create clusters from areas with a higher density in terms of events than other less represented areas. They are opposed to other types of clustering algorithms, such as centroid-based ones that attempt to assign every event to a cluster[3]. Clustering algorithms specifically targeting the extraction of POIs have been proposed, e.g., [65, 164]. As part of our work, we introduce our own POIs extraction algorithm, which is later detailed in Section 3.2.1.

2.2.6 Assumptions

We conclude our modelization by explaining the assumptions that are done in the remaining of our manuscript.

First, we assume that all communications happen in a secure manner. More specifically, the user is expected to send his mobility data to the LBS using a secured channel, and the analyst is expected to retrieve mobility dataset using a secured channel. Using TLS-encrypted channels is a well-known solution to this problem. This prevents man-in-the-middle attacks (e.g., wiretapping and eavesdropping) during the collection phase. The only adversary we consider is the LBS.

Second, we expect that users do not sending personal identifying information with their (hopefully protected) mobility data. Indeed, metadata attached to each LBS request can be used as a side channel and therefore can leak sensitive information. For example, the IP address that comes with each HTTP request is a well-known source of locational knowledge. APIs such as MaxMind's GeoIP [99] are able to convert an IP address into a geographical location. MaxMind claims a 99,8 % accuracy at the country-level, 90 % accuracy at the state-level and 83 % accuracy at the city-level within a 40 kilometers radius in the US, though these figures can vary from one country to another. In practice, it shows that an IP address is sufficient to infer the country and the city with a reasonable accuracy, but should not provide enough precision to get the exact address of a user. Still, a, IP address leaks some information and should be hidden to get the most of LPPMs. A simple proxy is a possible way to mitigate this issue, although literature contains more sophisticated solutions such as Tor [38] or the PEAS privacy proxy proposed by our team [118].

Third, the way a smartphone acquires its location has to be privacy-preserving too. There is no point of protecting mobility data with an LPPM if the location itself is obtained from an external service in an unsafe manner. For example, the Google Maps Geolocation API [58] allows to get the location from the list of nearby cell towers or Wi-Fi routers. The advantage is that it can be extremely precise in dense urban environments and it is cheaper in terms of battery life than GPS, but it has the drawback of giving access to the service provider (e.g., Google) to the user's location. In that case, the service provider becomes a new attacker that has access to the raw location. To mitigate that, users should use a privacy-preserving solution to acquire their location, such as the Global Positioning System[4].

[3] The parent of centroid-based clustering algorithms is k-means [96].

[4] GPS satellites are constantly broadcasting a signal that is then received by GPS-enabled devices on the ground. Communication happens only from space to ground, thus preventing satellites to learn anything from listening devices.

2.3 Privacy threats

Although the usefulness of LBSs does not need to be proven anymore, users are not always aware of the risks associated with the disclosure of their location during their daily life. The website *Please Rob Me* [122] aims to "raise awareness about over-sharing". They use geolocated tweets to infer whether a user is at home, and hence if the way is free for potential thieves. Indeed, sharing mobility data to LBSs does not come without risks, not only because of what the LBS can learn but also because of what the other users can learn from publicly published data. This section is dedicated to presenting the main practical threats, related to the exploitation of mobility data.

2.3.1 Points of interest

POIs are spatially delimited places where a user spends some time[5]. They are particularly sensitive as they convey information about what users are doing and their habits. Figure 2.1 presents a sample mobility trace and the result of an hypothetical POIs extraction. The user represented in this figure is moving in the center of Paris. Three POIs were extracted, one in front of the opera, one in a cinema and a third one at the crossing between two streets. Just by looking at the map, a quick analysis suggests that this person could have been waiting someone in front of the opera and is likely to have spent some time in a cinema (probably watching a movie). This (not so) imaginary analysis shows what kind of information it is possible to infer from POIs. Of course, a more rigorous analysis would use temporal information (to determine how much time was spent inside each POI) and more powerful tools such as geocoding (to get the exact address where this person was from his location) or semantic knowledge (such as Foursquare [44] or OpenStreetMap [114]).

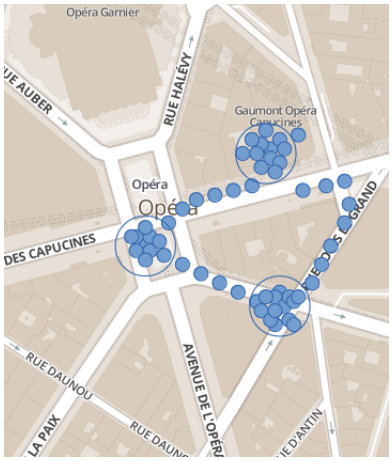


Figure 2.1: Three POIs have been extracted from this mobility trace.

Gambs et al. [49] made an attack on a dataset containing mobility data of taxi drivers in the San Francisco Bay Area. By finding points where the taxi's GPS sensor was off for a long period of time (e.g. 2 hours), they were able to infer POIs of the drivers. For 20 out of 90 users analyzed, they were able to locate a plausible home in a small

[5]The precise amount of time depends on the granularity one is concerned about. An attacker may be interested in very fine POIs, and thus willing to capture 5-minutes stops, or he may only be interested in coarse POIs, and thus only capturing stops of at least 2 hours.

neighborhood. They even confirmed these results for 10 users by using a satellite view of the area: it showed the presence of a yellow cab parked in front of the supposed driver's home.

Krumm [84] introduced *Placer*, a system using machine learning to automatically label places into 14 categories (home, work, shopping, transportation, place of worship, etc.). The author validated his solution by using two publicly available datasets, the American Time Use Survey and the Puget Sound Regional Council Household Activity Survey, which are diary surveys where subjects are asked to keep a track of all their activities for a few days. He reported an overall accuracy of 73 % and 74 % on the two datasets, mostly thanks to home and work places which are the easiest ones to label because this is where people spend the most of their time.

Deneau [45] created a visualization tool to analyze active and inactive periods of taxi drivers over the day. By correlating their time of inactivity with the five times of prayer per day observed by practicing Muslims, it was possible to find out which drivers are likely to be Muslims. From a 20 Gb dataset containing 173 million taxi rides in New York City in 2013, he was able to identify four examples of drivers that could be Muslims[6]. Tockar [147] showed that this same dataset allows to stalk at celebrities. With the support of publicly available photos of celebrities taking the taxi, he was able to reconstruct the journey of two of them (Bradley Cooper and Jessica Alba) and thus get additional information like the pick-up and drop-off locations or whether they tipped their driver. Tockar went further by extracting drop-off addresses of people frequently spending their night in a "gentlemen's" club. Correlating this address with Google and Facebook led to associate to one of these individuals a name and even a photo[7].

2.3.2 Social relationships

Comparing mobility data of several users allows one to infer relationships between them. The idea is rather simple: if two (or more) persons spend some time within the same area at the same moment, they are likely to be connected by some social link. Bilogrevic et al. [17] studied this threat by using malicious Wi-Fi access points deployed on the EPFL campus (in Switzerland) that were able to locate devices communicating with them. With two appropriate thresholds to detect a stop (typically at least 5 minutes) and the proximity of users (typically at most 20 meters), they could detect meetings between people. Then, they split the dataset in two parts to obtain a training dataset and a testing dataset. The training dataset was used, combined with a questionnaire and a database of courses, as ground truth. It allowed to build a model characterizing social links between students, whether they are classmates, friends or other, mainly depending on the place where they met and the time they spent together. The testing dataset was then used to validate this model. They achieved their best results when classifying friends, with a true positive rate of 84 % and a false positive rate of 27 %.

[6] From the information provided by the article, Deneau apparently did not push further its investigation once he demonstrated it was indeed possible to identify Muslim drivers. It "seems possible" that much more are identifiable.

[7] Here again, the goal of Tockar was to validate his methodology, though he did not apparently try to reproduce his experiment on other users.

2.3.3 Re-identification

Mobility data can ultimately lead to re-identifying physical users, i.e., to associate an identity to mobility traces. Krumm [81] used two months of mobility data and tried to infer users' home address with four different heuristics: the last destination closest to 3 a.m., the median location (weighted by the time spent at each location), the largest cluster and the best time (the likelihood that a user is at home at a given time of the day). He tested these heuristics with a dataset collected for research purposes by loaning GPS devices to car drivers. By combining the best performing heuristics (median location and largest cluster) with white pages, it was possible to retrieve correctly the name for 9 out of 172 drivers. Although this rate is not high, it shows such a threat is praticable. The author proposed three explanations: GPS imprecision, inaccurate geocoding/white pages, and erratic subject behavior. Gambs et al. [51] proposed a re-identification approach based on mobility Markov chains. The latter were used to model mobility patterns of users, more specifically the transitions between POIs. They designed eight different distance metrics to quantify the similarity between two Markov chains and used them to re-identify users by associating each unknown Markov chain to the closest Markov chain belonging to a known user. They validated their results against five datasets and achieved up to 45 % of good matchings with Geolife (a dataset of 178 users moving around Beijing during five years), which was significantly better than other state-of-the-art attacks.

These results are made possible by the high degree of uniqueness of human mobility. Indeed, De Montjoye et al. [35] showed that, with a CDR dataset containing 1.5 million users, only four randomly chosen events inside a trace were sufficient to uniquely identify 95 % of the users, while two randomly chosen events allowed to identify 50 % of the users. It means that the mobility of every individual acts like a unique fingerprint, even among a large number of users. In the same way, Golle et al. [56] studied the uniqueness of the home/work pair with a dataset from the US Census Bureau containing home and work locations for more than 103 million workers. Using the census district where people live and work, it was possible to uniquely identify 5 % of them, and for 40 % of them, there were only 9 other persons living and working in the same district (thus offering very little anonymity if this information was to be revealed). Zang et al. [160] improved the previous study by considering the top-N locations of a large CDR dataset of a US nation-wide cell phone operator, containing more than 30 billion call records made by 25 million users. They showed it was possible to uniquely identify 35 % of the users by using their top-two locations and 85 % of them by using their top-three locations. With Boutet et al. [19], we also demonstrated the highly unique nature of mobility traces constructed from different sensors, namely GPS, Wi-Fi and GSM (i.e., cell towers). For that purpose, we used two multi-sensors datasets, MDC (a dataset of 185 users moving around Geneva during three years) and Privamov (a dataset of 100 users moving around Lyon, France, during 16 months). Despite a much smaller dataset we showed again, confirming results of [35], that four random events inside a trace were sufficient to uniquely identify 94 % of the users. Among other observations, we noticed that the temporal dimension alone (i.e., only considering whether a user is moving or inside a POI) is as discriminative as the spatial dimension alone. For example, with locations inferred from the Wi-Fi sensor and four random events, the temporal information only allowed to uniquely identify 68 % of users (against 70 % with the spatial information only, and 94 % with both). We also pointed out that it

was still possible to re-identify users with a high success rate (between 80 % and 98 %) by using an appropriate attack relying on finding out the most visited POI, even if the data was protected by classical LPPMs [2, 9].

2.3.4 Future mobility prediction

Knowing past mobility of a user can help to model his habits and hence allow one to predict where he will be in a future time. Noulas et al. [113] focused on Foursquare check-ins. They collected a dataset containing more than 35 million check-ins from Foursquare over 5 months. They built a supervised learning model aiming at predicting places where users are likely to leave their next check-in. Precision was maximal during morning and at noon, when they achieved an accuracy of 65 %. It was more difficult to predict the next check-in in the night, during which accuracy dropped to 50 %.

Sadilek and Krumm [134] proposed *Far Out*, a system to predict the location of a user in the long term, i.e., in a far away future date[8] and within a time window of one hour. They leveraged Fourier analysis and principal component analysis to extract repetitive patterns from mobility data and build a model for supervised learning. These patterns were associated to a week day and an hour in the day. They tested their solution against a dataset containing more than 32,000 days of data for 703 users. Their system featured an accuracy in their predictions ranging from 77 % to 93 %.

Gambs et al. [47] modeled movement habits of people by using Markov chains. Each frequent POI becomes a state in the chain and a probability is assigned to each possible transition. They extended this model to incorporate not only the past POI, but the past n POIs in the Markov chain. With two different datasets, Geolife (a dataset of 178 users moving around Beijing during five years) and Phonetic (a dataset of 6 users collected over 15 months), they achieved a correct prediction rate between 70 % and 95 % when $n = 2$.

2.4 Evaluating LPPMs

Unfortunately, there is no standard way to evaluate LPPMs. This lack of well defined evaluation methodology leads to a multiplication of metrics used towards this purpose. To the best of our knowledge, only the work of Shokri et al. [139] focuses on the evaluation of LPPMs. Although it is only interested in quantifying privacy, it defines solid foundations towards building a complete evaluation methodology. In this section, we review the different evaluation metrics used in the literature to assess LPPMs in a quantitative manner. We start by introducing classical privacy notions in Section 2.4.1. We then group and present evaluation metrics through three complementary families, namely privacy metrics in Section 2.4.2, utility metrics in Section 2.4.3 and performance metrics in Section 2.4.4. We conclude this section by presenting the four architectures used to implement an LPPM in Section 2.4.5.

[8] In practice, how "far away" depended on how much data they had for a given user. On average, predictions were done for dates up to 23 days after the last known position.

2.4.1 Classical privacy notions

Two general definitions of privacy have emerged and have been widely adopted by the community since. We present them at the beginning of this manuscript because they are still the foundation for most of subsequent works. Those definitions propose generic privacy guarantees that were originally not specific to location privacy, but have been later successfully applied to location privacy. In this section (and only this one), the concept of dataset is not limited to mobility datasets as defined in Section 2.2.2 but to generic datasets, i.e., a list of records with attributes.

k-anonymity

The concept of k-anonymity has been introduced by Sweeney in 2002 [144]. The idea is to prevent one to uniquely identify individuals from a small subset of their attributes, called a *quasi-identifier*. The subset of attributes to protect, which is not part of the quasi-identifier, form the *sensitive attributes*. For instance, within medical records, the birth date, sex and zip code triplet is a quasi-identifier that is enough to uniquely identify some individuals, while the disease is a sensitive attribute. k-anonymity states that to be protected, a user must be indistinguishable among at least $k-1$ other users. To achieve that, all k indistinguishable users must have the same values for all attributes forming their quasi-identifier. This makes them look similar and forms what is called an *anonymity group*. Therefore, the probability of an attacker without external knowledge to re-identify someone among k similar users is at most $1/k$.

Definition 1. *Let d be a sequence of records with n attributes $a_1, ..., a_n$ and $Q_d = \{a_i, ..., a_j\} \subseteq \{a_1, ..., a_n\}$ be the quasi-identifier associated with d. Let d_k be the k-th record of d and $r[Q_d]$ the projection of record $r \in d$ on Q_d, i.e., the $|Q_d|$-tuple formed of values for only the attributes of Q_d in r. d is said to satisfy k-anonymity if and only if each unique sequence of values in the quasi-identifier appears with at least k occurrences in d, or formally:*

$$\forall s \in \{r[Q_d] \,|\, r \in d\}, |\{i \in \mathbb{N} \,|\, d_i[Q_d] = s \ \}| \geq k$$

For example, Table 2.2 shows a sample medical dataset exposing a k-anonymity guarantee, where the quasi-identifier is $\{Birth, Sex, Zip\}$ and the sensitive attributes are $\{Disease\}$, for $k = 2$. Here, there are three unique $\{Birth, Sex, Zip\}$ triplets, i.e., $\langle 1970, M, 0247\rangle$, $\langle 1970, F, 0247\rangle$ and $\langle 1969, M, 0232\rangle$. For each of those triplets, there are respectively two, three and two different records. Consequently, there is a minimum of two different records for each triplet of values taken by the quasi-identifier: this table guarantees 2-anonymity. This way, knowing the birth year, sex and zip code of some individual should not leak his disease, as there is at least one other person with the same quasi-identifier.

However, despite providing 2-anonymity, there is a problem in Table 2.2 for male patients born in 1969 and living in the area with 0232 zip code (i.e., the last two records). Indeed, they share the same value for their sensitive attribute (i.e., they have the same disease), which leaves them unprotected. This concern has been addressed by the introduction of ℓ-diversity [95]. It extends k-anonymity by additionally enforcing that within anonymity groups, there should be at least ℓ "well-represented" values. More precisely, it enforces a particular distribution of values for sensitive attributes across

Figure 2.2: Example of a dataset with k-anonymity where $k = 2$.

Birth	Sex	Zip	Disease
1970	M	0247	Migraine
1970	M	0247	Chest pain
1970	F	0247	Asthma
1970	F	0247	Migraine
1970	F	0247	Asthma
1969	M	0232	Appendicitis
1969	M	0232	Appendicitis

each anonymity group. This "well-represented" notion is formally defined in three different ways in [95]. The simplest one is called distinct ℓ-diversity and states that there must be at least ℓ distinct values for each sensitive field for each anonymity group.

Differential privacy

Differential privacy is a more recent concept introduced by Dwork [39] defining a formal and provable privacy guarantee. The idea is that an aggregate result computed over a dataset should be "almost" the same whether or not a single element is present inside the dataset. In other words, the addition or removal of one single element shall not change significantly the probability of any outcome of an aggregate function. Unlike k-anonymity, the differential privacy definition is not affected by the external knowledge an attacker may have.

Definition 2. *Let $\epsilon \in \mathbb{R}^{+*}$ and $\mathcal{K}$ be a randomized function that takes a dataset as input. Let image($\mathcal{K}$) be the image of $\mathcal{K}$. $\mathcal{K}$ gives ϵ-differential privacy if for all datasets D_1 and D_2 differing on at most one element, and for all $S \subseteq$ image($\mathcal{K}$),*

$$\Pr[\mathcal{K}(D_1) \in S] \leq e^{\epsilon} \times \Pr[\mathcal{K}(D_2) \in S]$$

For example, Table 2.3 shows two versions of a sample dataset listing whether individuals are subject to chronic migraines. Let us suppose that an analyst has access to these two datasets, and to a query Q that takes a dataset as input and returns the number of persons having chronic migraines. By computing $Q(D_2) - Q(D_1) = 3 - 2 = 1$, our curious analyst can infer that Joe is indeed subject to chronic migraines.

Figure 2.3: Two datasets differing on one single element.

(a) Dataset D_1, without Joe.

Name	Has chronic migraines
Agatha	True
Anna	False
John	True
Mark	False
Mary	False

(b) Dataset D_2, with Joe.

Name	Has chronic migraines
Agatha	True
Anna	False
Joe	True
John	True
Mark	False
Mary	False

Several methods have been proposed to practically achieve differential privacy. We present one of them, called the Laplace mechanism, that can is used for numerical

values, and hence in the location privacy context. It relies on adding random noise, whose magnitude depends on the *sensitivity* of the query function issued on the dataset. Intuitively, the sensitivity of a query function quantifies the impact that the addition or removal of a single element of a dataset could have on the output of this function.

Definition 3. *Let f be a function that takes a dataset as input and produces a vector of reals, i.e., $f : \mathcal{D} \longrightarrow \mathbb{R}^n, n \in \mathbb{N}$. Let D_1 and D_2 be two datasets differing on at most one element. The sensitivity of f is noted Δf and defined, for all such datasets D_1 and D_2, as:*

$$\Delta f = \max_{D_1, D_2} ||f(D_1) - f(D_2)||_1.$$

The sensitivity is defined independently of the underlying data, and only depends on the function under consideration. In particular, for queries that are counting records (such as Q in our previous example), $\Delta Q = 1$ because the addition or removal of a single record affects the count result by increasing or decreasing its value by 1. Then, the Laplace mechanism adds Laplacian noise with mean 0 and scale parameter $\Delta f/\epsilon$ to the query's result[9]. Consequently, the ϵ-differentially privacy version of Q is defined as $\hat{Q}(D) = Q(D) + Y$, where $Y \sim \text{Lap}(1/\epsilon)$. That way, computing $Q(D_2) - Q(D_1)$ does not automatically result in 1, because of the added Laplacian noise. The Laplace mechanism is of course only suitable for queries producing numerical results; another method exists for categorical values [100], but it is outside of the scope of this thesis.

Differential privacy supports the composition of functions, and the potential information leakage resulting of this composition can be quantified. In the general case, when applying n randomized independent algorithms $\mathcal{K}_1, ..., \mathcal{K}_n$ that provide $\epsilon_1, ..., \epsilon_n$-differential privacy, any composition of those algorithms provides $(\Sigma_i \epsilon_i)$-differential privacy. This is known as *sequential composition.*

This protection model assumes that each analyst has a global privacy budget. Each time he issues an ϵ-differentially private query, his privacy budget is reduced by ϵ. Once the budget is totally consumed, all subsequent queries from this analyst should be rejected. It models the fact that once an information is learnt, it cannot be forgotten. In practice, determining this privacy budget and its instantiation (global, per user, etc.) remains largely an open question that has not really been studied in the literature. As authors of [22] note, "when the user runs out of budget, he should in principle stop using the system. This is typical in the area of differential privacy where a database should not being queried after the budget is exhausted. In practice, of course, this is not realistic, and new queries can be allowed by resetting the budget, essentially assuming either that there is no correlation between the old and new data, or that the correlation is weak and cannot be exploited by the adversary. In the case of location privacy we could, for instance, reset the budget at the end of each day. [...] The question of resetting the budget is open in the field of differential privacy and is orthogonal to our goal of making an efficient use of it." However, because of the importance of this question, recent works take interest in it, e.g., [70].

[9] Proof of this is provided in [39].

2.4.2 Privacy metrics

To quantify the level of protection offered by an LPPM, we identify three categories of privacy metrics.

- *Formal guarantee* metrics adopt a theoretical approach to quantify the effect of an LPPM on mobility data. They use a well-defined and unambiguous framework to guarantee that a protected dataset has a certain level of privacy. As of now, there are two such guarantees commonly offered by LPPMs: k-anonymity and differential privacy (cf. Section 2.4.1). *k-anonymity*, applied to location privacy, states that during a given time window and inside a given area, there should be at least k users. LPPMs then take different approaches to enforce this guarantee, for example by allowing to specify the size of these areas or time windows as parameters, or by automatically adjusting them, such as they contain k users. *ϵ-differential privacy* has been instantiated differently by different LPPMs. Usually, instead of protecting the presence or absence of individual users, as it is the case with classical differential privacy, LPPMs attempt to protect the presence or absence of individual locations. Hence, the goal is not anymore to hide that a user is part of a dataset, but to hide where he went.
- *Data distortion* metrics compare privacy-related properties of mobility data before and after applying an LPPM on it. Indeed, using an LPPM is expected to hide sensitive information that was otherwise possible to obtain from actual mobility data. Examples of such metrics include computing the entropy of protected data or evaluating whether POIs can still be retrieved.
- *Attack correctness* metrics evaluate the impact of a location privacy attack that could be ran by an adversary in order to gain knowledge about users. [139] did an extensive work on the usage of attacks to quantify location privacy. They distinguish between three axes when evaluating the effectiveness of an attack: certainty, accuracy and correctness. Certainty is about the ambiguity of the attack's result; for example there is some uncertainty if a re-identification attack outputs three possible users, while the uncertainty is null if the same attack outputs a single user (independently of whether it is the correct answer). Accuracy is about taking into account that the attacker does not have unlimited computational resources; consequently, the output of his attack may be only an approximate response, e.g., by only taking into account a sample of all data at his disposal. Correctness quantifies the distance between the attack's result and the truth; it is what actually quantifies location privacy. An LPPM is expected to mitigate privacy attacks and lower (or even suppress) their harmful effects. As opposed to data distortion metrics, attack correctness metrics do not compare the effect of an attack before and after applying an LPPM, but rather evaluate directly the attack on a protected dataset, and use the actual dataset as ground truth to evaluate whether the attack was successful.

2.4.3 Utility metrics

To evaluate the quality of service while being protected by an LPPM, we identify two categories of utility metrics.

- *Data distortion* metrics compare utility-related properties of mobility before and

after applying an LPPM on it. Indeed, we expect that the LPPM will not distort all properties of a dataset and make it unusable. Examples of such metrics include evaluating the spatial/temporal imprecision and comparing the covered area. It is of purpose that we name this category the same way as for privacy metrics, because they do represent the same thing, but applied on different properties (privacy- or utility-related). If we go even further, it happens that some data distortion metrics are used one time as a privacy metric and the other time as a utility metric[10].

- *Task distortion* metrics compare the result of some practical task on the data before and after applying an LPPM. For instance, these metrics can be interested in data mining tasks or analytics queries. As opposed to data distortion metrics, which compare directly the properties of two datasets, task distortion metrics compare the outcome of a (possibly complex) task executed on a dataset. While data distortion metrics remain rather generic, task distortion metrics are more specialized and usually specific to a given use case.

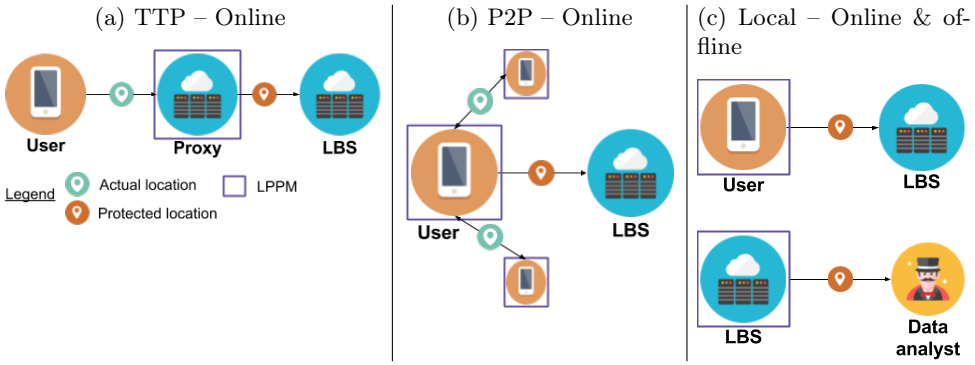
2.4.4 Performance metrics

To evaluate the performance an LPPM, four categories of metrics are commonly used.

- *Execution time* is a simple quantification of the time it takes for an LPPM to protect data. Of course, it does not have the same impact for real-time use cases, where a response is expected in a very short time frame (a few milliseconds, a few seconds at most), than for batch or offline use cases that do not expect an immediate answer. However, even for the latter, it is of importance as computational resources have a cost ("time is money"). This execution time can be measured in various ways, for example in seconds or in CPU cycles.
- *Communication overhead* quantifies the negative impact of applying an LPPM on the quantity of information that will be produced and exchanged through the network in online use cases. For online use cases, some LPPMs need to exchange more messages, or more answers are received from the LBS. Obviously, it has an impact on the execution time, but it can be measured separately. For offline use cases it is related to the size of the protected dataset; if bigger or more complex that the actual one, it can slow down the job of analysts and affect their experience when working with the dataset.
- *Energy overhead* measures the negative impact on the battery lifetime implied by using a given LPPM, when running it as an application on a mobile device. It is important to be quantified because it impacts the usability and adoption by end users. It is only applicable to online LPPMs.
- *Scalability* measures how well an LPPM can face a high workload. For online LPPMs, scalability metrics are mostly related to the capability of handling a high volume of concurrent requests, while for offline LPPMs it concerns the ability to deal with datasets of large sizes.

[10] A common example is a metric whose goal is to compare the distance between actual locations and protected locations. It can be viewed either as a privacy metric, because by distorting locations we hide where users were, or as a utility metric, if the LBS that we use or the task that the analyst wants to run requires spatial precision.

Figure 2.4: Three different architectures for LPPMs.



2.4.5 Architectures of LPPMs

LPPMs can leverage three different architectures, which are depicted in Figure 2.4. The local architecture is used by both online and offline LPPMs, while the TTP and P2P architectures are unique to online LPPMs.

- The *TTP* architecture requires a trusted third party proxy server. It means there is an external entity that has access to the actual data coming from all users.
- The *P2P* architecture requires no external server, but it requires users taking part in the system to exchange information in a peer-to-peer fashion in order to protect their data. Such LPPMs engage users in a collaborative privacy protocol before they send their data to an LBS.
- The *Local* architecture does not require any communication with another party to protect data. LPPMs entirely autonomous and process everything locally, on the device on which they are executed. They may need access to external databases, in which case the latter are expected to be entirely available locally.

2.5 Preserving privacy with LPPMs

LPPMs have been introduced to mitigate location privacy threats such as the ones presented in Section 2.3. According to our scenario presented in Figure 1.1, we distinguish between two phases when using an LBS with an LPPM: the collection and the publication. During the collection phase, either a real-time LPPM, a batch LPPM or no LPPM at all can be used. Although some of them can be designed explicitly with one use case or the other in mind, real-time and batch LPPMs are largely interchangeable and the denomination depends mainly on the target LBS. For this reason, we will not distinguish in this chapter between real-time and batch LPPMs. LPPMs used during the collection phase are all labelled as *online LPPMs*, while LPPMs used during the publication phase are labelled as *offline LPPMs*.

Figure 2.5 summarizes our approach when classifying LPPMs. It highlights the four axis that are simultaneously used to qualify an LPPM: its use case, the way it is evaluated, its architecture and its family. Indeed, we present in this section the state-of-the-art LPPMs that we surveyed organized across five families. We summarize all LPPMs and their categorization in Table 2.2 and Table 2.3 for online and offline LPPMs, respectively. Moreover, we indicate for each LPPM its architecture and the family

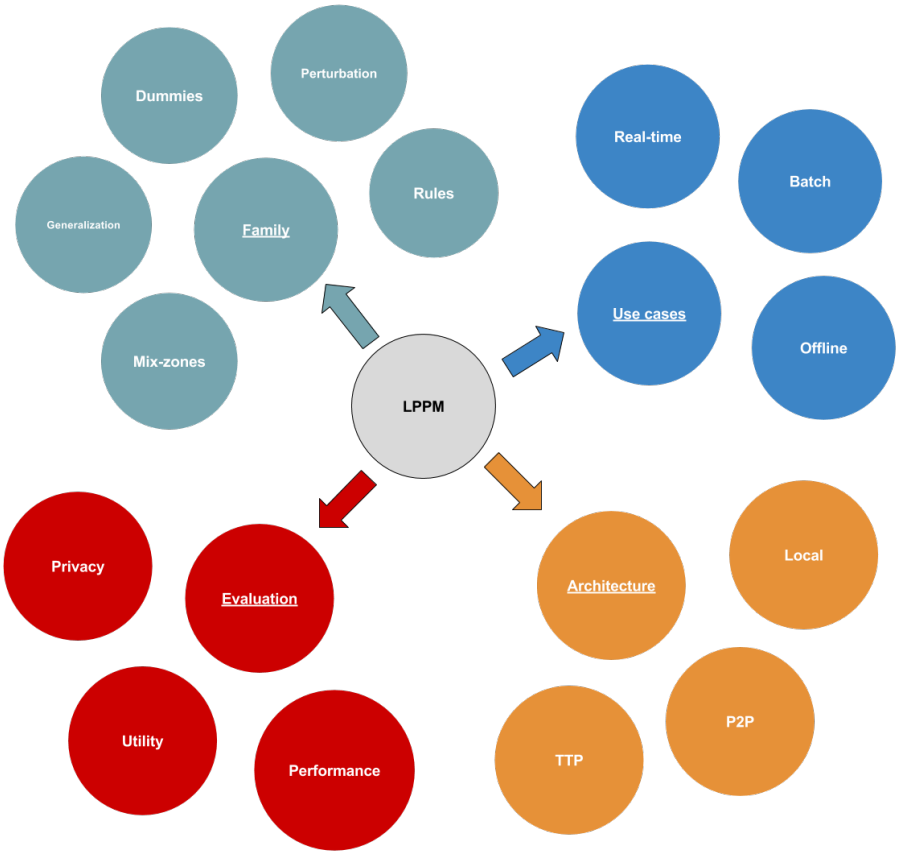


Figure 2.5: Taxonomy of location privacy threats and state-of-the-art LPPMs.

of metrics that were used to evaluate it, according to the methodology presented in Section 2.4. For exhaustivity, we distinguish in these tables between differential privacy and k-anonymity for privacy formal guarantees, and mention when an ad-hoc metric was used to evaluate LPPMs. Ad-hoc metrics encompass metrics that do not fit in our classification, usually because they measure something that is unique to the way the LPPM works., e.g., something related to its algorithm and that cannot be made generic to all LPPMs.

2.5.1 Mix-zones

Mix-zones is a concept introduced by Beresford and Stajano [15], taking its roots in the seminal work of Chaum [24] about mix networks, and further refined in [14]. A mix-zone is defined as an area where movements of users are not tracked, and consequently where users cannot communicate with an LBS. When a user leaves a mix-zone, he receives a new identifier, usually chosen among those of users still inside the mix-zone. It means that when k users are inside a mix-zone at the same time, their identities will be shuffled, providing some sort of k-anonymity (the actual identity of each user is hidden among $k-1$ other users) and resulting in an attacker's confusion. Mix-zones are usually placed at the crossing of several roads, to maximize the confusion (because users are expected to change of direction at such places). This model applies only during the collection phase.

Table 2.2: List of *online* LPPMs studied in this chapter, with their architecture and metrics used by their authors to evaluate them.

		Privacy					Utility			Performance				
Protection mechanism	Architecture	Differential privacy	k-anonymity	Data distortion	Attack correctness	Ad-hoc metric	Data distortion	Task distortion	Ad-hoc metric	Execution time	Communication overhead	Energy overhead	Scalability	Ad-hoc metric
Mix-zones														
Freudiger et al. [46]	TTP				✓									
Traffic-aware mix-zones [91]	TTP				✓					✓				
MobiMix [115]	TTP				✓					✓			✓	
Generalization-based mechanisms														
CliqueCloak [52]	TTP		✓					✓		✓	✓			
Casper [105]	TTP		✓							✓				
P2P cloaking [28]	P2P		✓					✓		✓	✓			
PRIVÉ [54]	P2P		✓			✓				✓	✓			✓
PrivacyGrid [11]	TTP		✓					✓		✓			✓	
Agir et al. [6]	Local			✓		✓	✓							
Ngo et al. [111]	Local	✓		✓					✓	✓				
Dummies-based mechanisms														
Kido et al. [77]	Local		✓			✓								
You et al. [157]	Local		✓			✓			✓					
MobiPriv [143]	TTP		✓	✓				✓		✓				
Kato et al. [76]	Local		✓	✓		✓								
SybilQuery [135]	Local		✓			✓				✓	✓			
Realistic fake trips [83]	Local													
Perturbation-based mechanisms														
Geo-indistinguishability [9]	Local	✓						✓			✓			
Path cloaking [69]	TTP			✓			✓							
CAP [120]	Local		✓					✓		✓				
Location truncation [102]	Local							✓						
Predictive geo-indistinguishability [22]	Local	✓					✓		✓					
Elastic geo-indistinguishability [23]	Local	✓		✓			✓							
Rules-based mechanisms														
ipShield [21]	Local									✓		✓		
LP-Guardian [41]	Local						✓			✓		✓		

The main issue that arises is where to place them. Indeed, too many mix-zones will results in a great loss of utility, as users will not be able to use a LBS when they need it, while too few mix-zones will greatly reduce the probability that a user ever enters one. So the question is how to find optimal locations for these mix-zones. One solution to the mix-zones placement problem was proposed by Liu et al. [91]. They modeled the city as a graph, where nodes are venues (i.e., places of interest inside a city such as monuments, restaurants, cinemas, etc.) and the road network is used to create edges connecting those venues. On the one hand, an LBS can have side information on this graph and use it to re-identify users. On the other hand, information about traffic is used to compute the optimal placement of mix-zones as an optimization problem. Other solutions to this problem were proposed, e.g., [46, 115]. Overall, mix-zones LPPMs suffer of a non-negligible weakness which is that they need "a lot" of users to be effective. Indeed, if too few users participate to the system, it is not very likely that they will meet at any time during the day. We believe that this critical mass of users is too important to make

Table 2.3: List of *offline* LPPMs studied in this chapter, with metrics used by their authors to evaluate them.

	Privacy					Utility			Performance	
Protection mechanism	Differential privacy	k-anonymity	Data distortion	Attack correctness	Ad-hoc metric	Data distortion	Task distortion	Ad-hoc metric	Execution time	Scalability
Generalization-based mechanisms										
Nergiz et al. [110]		✓					✓	✓	✓	
Never Walk Alone [1]		✓	✓				✓		✓	
Wait for Me [2]		✓	✓			✓	✓		✓	✓
Differentially private grids [89]	✓						✓			
GLOVE [60]		✓				✓				
Perturbation-based mechanisms										
Geo-indistinguishability [9]	✓						✓			
Path confusion [68]				✓	✓	✓				
Chen et al. [25]	✓						✓		✓	✓
Jiang et al. [75]	✓					✓				
Riboni et al. [132]	✓	✓				✓	✓			
Synthetic fake trips [18]				✓			✓	✓		
DP-WHERE [103]	✓					✓				

mix-zones usable for individual users willing to protect their privacy. This family of LPPMs seems more suited to be integrated either by hardware manufacturers of GPS-enabled devices (e.g., Google, Apple, TomTom) or directly inside the OS (e.g., iOS, Android), because there would be a significant base of users adopting such an LPPM. Therefore, this solution would break one of our goal which is to achieve interoperability with existing LBSs, by letting the user decide whether and how he wants his location privacy to be protected. This is why we will not consider furthermore mix-zones in the remaining of this thesis, as it seems not practicable, with respect to our goals.

2.5.2 Generalization-based mechanisms

Generalization-based methods are the application of k-anonymity to location privacy. More specifically, it has been theorized as the the concept of spatial cloaking introduced by Gruteser et al. [61]. The idea is to create and maintain cloaking areas in which at least k users are located at any given moment. Moreover, reducing the precision of the spatial information (users are reported to be in areas instead of at a precise location) also adds some privacy, because it is not possible anymore to infer exactly where they are and what they are doing.

Online LPPMs

A representative online LPPM in this family is *Casper* [105], a spatial cloaking proxy proposed by Mokbel et al. It uses a location anonymizer (i.e., a trusted third party) which knows locations of all users and their privacy parameters (a k parameter and a minimal cloaking area size). When a user sends his query to the anonymizer, the latter

transforms it into a cloaked query and forwards it to the LBS. The latter needs to be able to understand such cloaked queries. The response is then sent back to the anonymizer, which refines it by using the actual user location and sends the final response to the user. Chow et al. proposed *P2P cloaking* [28], which is essentially an improvement of Casper. Like in Casper, users specify a privacy profile with a k parameter and a minimal cloaking area size. However, instead of using a central trusted anonymizer, a peer-to-peer protocol enables nearby peers to generate cloaking areas. Clients can then send themselves the query including the cloaking area, instead of their exact location, to an LBS. Their solution comes in two variants: the on-demand one, which executes the algorithm only when there is a need to query an LBS, but takes more time to completed, and the proactive one, which executes the algorithm in the background, but incurs a higher communication overhead. Other solutions were proposed, such as [11, 52, 54].

More recent works propose generalization-based approaches without the goal of providing k-anonymity. Agir et al. [6] introduced an adaptive mechanism to dynamically change the size of the cloaking areas hiding the exact location of users. More precisely, their solution locally evaluates the privacy level and enlarges the area until a target privacy level is achieved, or the information is too distorted (in which case the location cannot be released). Consequently, it may happen that this LPPM fails to deliver a query to an LBS. The privacy level is estimated from a linkability graph, whose nodes are events and edges represent connectivity (in time and space) between them. The goal of this graph is to evaluate the belief of an LBS about the authenticity of an event, by using side-channels such as the speed and the topology of the area. For example, a walking user is not expected to move 5 kilometers away in 5 minutes, or it may not be possible to go on the other side of a river without a bridge in 2 minutes. This solution is particularly interesting because it does not protect all locations the same way, but it adapts to the particular context to enforce a minimal distortion. Ngo and Kim [111] also proposed a generalization-based LPPM, but providing differential privacy guarantees (instead of k-anonymity, as it is usually the case with generalization-based LPPMs).

Overall, a problematic question that remains is the integrability of these solutions with existing LBSs. Indeed, LBSs usually work with locations (i.e., single points) and not areas. Therefore, generalization-based solutions are not always immediately usable with existing LBSs, or workarounds need to be implemented, e.g., report the barycenter of an area instead of the entire area if it is not supported. Moreover, these LPPMs all suffer from a default similar to mix-zones, which is that there is a need for a sufficient number of users to participate to the system to make it work. This is obvious for P2P architectures, which rely on other users to do the work, but it is also the case for TTP architectures, which need other users in the same area to generate cloaking areas. Consequently, either LPPMs enlarge areas until the privacy requirement is met, or they simply fail to protect data (e.g., [11, 52]). LPPMs with local architectures may be valid competitors for our propositions, because they do not have aforementioned drawbacks, relying on statistical models to evaluate the users' density instead of actual traffic.

Offline LPPMs

Abul et al. proposed *Never Walk Alone* [1], whose idea is to guarantee that at every instant each user is at a maximum distance δ of at least $k-1$ other users. More precisely, authors exploit the inherent incertitude that comes from location measurements to maintain confusion in the mind of an attacker and avoid distorting data too much.

They introduce the notion of (k, δ)-anonymity, where δ represents this uncertainty. As a result, their LPPM creates cylinders with a radius of δ within which at least k users are moving. Their algorithm enforces this property by spatially distorting events. Some events can also be suppressed (the maximum number of events that can be suppressed is a parameter of the algorithm) if they would be too difficult to protect or would require to distort the data too much. This mechanism has been later improved by *Wait for Me* [2], whose essential modification is to be time-tolerant. Consequently, events are distorted both temporally and spatially. This new algorithm also support the protection of batches of data taken from a larger dataset, allowing to scale to large datasets. However, even this improved algorithm still suffers of two major weaknesses. First, the δ chosen in experiments seems rather large, usually varying between a few meters and one kilometer. Indeed, the imprecision coming from a typical GPS sensor is between 5 and 15 meters (though it can vary depending on the exact environment), which is very far from a one kilometer. Second, the execution time is quite long, rapidly reaching thirty minutes for datasets with "only" 5 million events. This is because of the algorithmic complexity, which additionally makes the algorithm difficult to parallelize. Other approaches include [60, 89, 110].

In offline use cases, generalization-based LPPMs are valid competitors for our research problem. However, as outlined with the case of *Wait for Me*, these approaches are likely to suffer from a poor execution time, because forming clusters of k users is computationally expensive and is difficult to achieve in linear time.

2.5.3 Dummies-based mechanisms

Instead of relying on other users to be hidden among them and obtain k-anonymity, as with generalization-based approaches, it is possible instead to generate fake users, called dummies. The basic idea is for each user to send multiple queries to an LBS, instead of a single one. One of those queries contain his actual location, while the others contain fake locations. The LBS may be aware that there are dummies inside the data it got (obviously a user cannot be located in three different locations at the same moment), but it should not able to determine the actual user's location. The challenge here is to generate realistic fake data, indistinguishable from the real data. LPPMs belonging to this family are all online LPPMs. Theoretically, integrating dummies to enforce k-anonymity is an alternative to generalization-based methods, but it has never been applied to offline use cases, perhaps because the utility would be difficult to maintain.

Kido et al. [77] were the first to introduce a protection mechanism using dummies. They simply split the space into regions of a fixed size and generated dummies in neighboring regions. Overall, this method is rather naive and does not produce realistic dummies. Stenneth et al. [143] presented *MobiPriv*, which uses an anonymization proxy through which all queries transit before being sent to an LBS. Similarly to centralized protection mechanisms presented in Section 2.5.2, the proxy of *MobiPriv* enforces k-anonymity by generating realistic looking dummies. This way, instead of relying on other users to be in the same vicinity as it was the case with *Capser* [105] and other generalization-based methods, fake users are simply generated to simulate activity and protect the actual user. *MobiPriv* also leverages a history of previous queries to prevent attacks using the intersection of multiple queries' results to infer new knowledge. Kato et al. [76] and You et al. [157] presented other methods that both work with a local architecture.

Other solutions are focused on creating whole trips at once. The targeted use case is navigation applications, where start and end point are indeed known in advance (though they might change, but it is not considered in following works). As the user drives, he queries the LBS by sending his real location and $k-1$ other locations, according to the pre-computed fake trips. Shankar et al. introduced *SybilQuery* [135], which generates fake trips starting from and ending to different locations, while preserving properties such as the length of the trip and the semantics of the areas where endpoints are located (e.g., residential vs business areas). This step requires extensive external knowledge in order to effectively generate realistic-looking endpoints. A trajectory is generated between these endpoints using the LBS itself, which might only shift the problem (a better idea would probably be to use an embedded route planner, relying for example on OpenStreetMap data [114]). Krumm proposed another method to generate realistic-looking fake trips [83].

The main issue with dummies-based LPPMs is their ability to produce real-looking dummies. Indeed, a study of Peddinti et al. [116] showed that *SybilQuery* is very vulnerable to attacks based on machine learning. They developed an algorithm able to correlate traces, and tested it against a dataset containing data of 85 taxi drivers around San Francisco. SybilQuery was configured with $k = 5$, which means that each mobility event generated by a driver was hidden among four other dummy events. In the case of an attacker having access to a previous mobility dataset (forming the training dataset), their algorithm re-identified 93 % of the users. Furthermore, some of these algorithms (e.g., [83, 135]) use an extensive amount of external knowledge, such as a graph modeling the road network, a route planner or census statistics about the population. One could advocate that, it seems unrealistic that all of this data would fit comfortably on a mobile device. To give an order of magnitude, as of 2017/06/05, the entire OpenStreetMap XML planet file [114] takes 803 Gb. It could be reduced by only keeping necessary features, or by downloading only areas in which users move, but it still represents a large amount of data, difficult to process on a smartphone.

2.5.4 Perturbation-based mechanisms

Perturbation-based LPPM rely on pure data alteration to protect mobility data. This family basically encompasses all LPPMs that are neither related to mix-zones, generalization or dummies (rules-based LPPMs are rather aside, and will be presented in the next section). As a consequence, perturbation-based LPPMs rely on a large range of techniques with various objectives. Notable members of this family are differentially-private LPPMs.

Online LPPMs

Differential privacy has been generalized for location privacy by Andres et al. under the notion of *geo-indistinguishability* [9]. Geo-indistinguishability is a formal notion of location privacy that bounds the probability of two locations to be protected locations of the same real location within a given radius. Geo-indistinguishability states that two close locations ℓ and ℓ' should be perturbated into the same protected location $\hat{\ell}$ with a similar probability. As the distance between ℓ and ℓ' increases, their respective probabilities to be protected into the same location $\hat{\ell}$ differ. Authors proposed a way to provide geo-indistinguishability by adding noise drawn from a 2-dimensional Laplace distribu-

tion to an actual location[11], similarly to what is done in classical differential privacy. Due to the temporal correlations between locations inside a mobility trace (indeed the location at time t is strongly related to location at $t-1$, e.g., it is physically impossible for a driver to be 10 kilometers away in 2 seconds), differential privacy proposed in geo-indistinguishability can be problematic due to the cost to protect a whole trace. Indeed, per the sequential composition theorem (cf. Section 2.4.1), protecting each one of the n events of a mobility trace with ϵ-geo-indistinguishability results at the end in $n\epsilon$-geo-indistinguishability. Knowing that traces can have high sampling rates (e.g., one event per 30 seconds, or less), it results in a too high cost in terms of privacy budget. To overcome this limitation, Chatzikokolakis et al. proposed a predictive mechanism [22] using prediction to avoid spending too much budget for each location. If a protected location can be predicted by the LPPM, the latter uses this predicted location instead of spending budget to actually protect it. With two different ways of spending the privacy budget (fixed rate or fixed utility), this gives a substantial improvement over the original geo-indistinguishability LPPM. The same authors also proposed another extension of geo-indistinguishability that leverages contextual information to calibrate the amount of noise applied to disturb the mobility traces [23]. Indeed, they consider that not all locations have the same sensitivity, and that being in a dense urban environment with a lot of nearby venues is likely to reveal less information than being in a countryside area where there is only a few (or even a single) venues around. Consequently, they do not apply the same level of protection depending on the actual surrounding of the user. Another online differentially private LPPM was proposed [153], as well as LPPMs providing other kinds of guarantees besides differential privacy [69, 102, 120].

Perturbation-based LPPMs fit into our research problem and are valid competitors that we will consider when evaluating our proposals. Because they do not have any obvious disadvantage, besides the problem of choosing a budget for differentially private LPPMs (cf. 2.4.1 for more on this subject), we do not elaborate more here and conduct a thorough practical evaluation of geo-indistinguishability [9] in Section 3.6, as a representative LPPM of this family.

Offline LPPMs

Several works were interested in publishing differentially private datasets, such as [25,75, 132]. The authors of geo-indistinguishability [9] also presented an offline usage of their LPPM. However, other approaches are still possible. For example, Hoh and Gruteser [68] introduced the idea of path confusion. The idea is to force paths of close users to cross when they are close enough, in order to augment the confusion of an adversary about which path belongs to which user. If paths are already close enough, it minimally distorts the trace. They formulate and solve this problem as a constrained non-linear optimization problem.

During the publication phase, another way to guarantee that privacy is preserved is to publish a synthetic dataset instead of the actual dataset. A synthetic dataset is generated from scratch in such a way that some statistical properties exposed by the actual dataset are preserved. Such approaches typically work in two phases: (1) a model, containing statistical features to preserve, is learnt from the actual dataset (2) a new dataset is generated, according to this model. Though this approach has been largely

[11] Proof of this is given in [9].

explored for general databases (e.g., [88, 94]), few works apply this to location privacy. *DP-WHERE* [103] is a method introduced by Mir et al. to generate synthetic CDRs in a differentially private way. They start by building a model of real CDRs, formed of several histograms, and then add noise to each of them to achieve differential privacy. A synthetic CDR can be generated by using the private versions of the histograms. Bindschaedler and Shokri [18] proposed to generate synthetic mobility traces that are designed to be used instead of the real traces, thus presumably leaking no sensitive information. They build a mobility model for each trace and an aggregate probabilistic mobility model about the entire dataset, and use them to synthetize fake traces from these models. Moreover, they enforce that these traces satisfy a privacy test before being actually released.

2.5.5 Rules-based mechanisms

Some believe that one-size-fits-all protection mechanisms are unrealistic. This is why some protection mechanisms implement several state-of-the-art solutions and follow a set of rules to decide of the most appropriate countermeasure to take in the current situation. They can be viewed as an aggregation of solutions presented in previous sections, with a rules engine deciding which one to apply. Mechanisms presented in this section are all online LPPMs.

Chakraborty et al. proposed *ipShield* [21], which is a framework, implemented on Android, leveraging a rules engine to protect location privacy. Users define which threats they want to be protected against, with a priority level. The system then leverages a database of inference attacks to recommend protection rules to apply on each sensor (i.e., not only the GPS but also the accelerometer, the gyroscope, etc.). Therefore, the strength of their solution is that users specify high-level goals, that are then translated into low-level system actions to take. Users can also define their own rules to handle specific use cases, using contextual information and specifying actions to take on sensor data. *LP-Guardian* [41] is a software running on Android proposed by Fawaz and Shin to protect location privacy of Android smartphones users. They designed a framework to protect privacy against different threats: tracking threat, identification threat and profiling threat. Their solution leverages a decision tree to decide which action to perform in a given situation by using the context (e.g., the application being used, the actual location). There is some manual input required from the user to bootstrap the application (i.e., define commonly visited places), and then each time the user uses a new application (he can set per-application rules) or uses an application from a new place (he can set per-place rules).

2.6 Related approaches

In this section, we present two other approaches for location privacy. We consider they are worth being mentioned in order to provide a thorough view of location privacy, but they are not direct solutions to our research problem. Therefore, we do not consider them as competitors because they do not fit in our workflow depicted in Figure 1.1.

2.6.1 Privacy-by-design architectures

Privacy-by-design has been theorized by the information and privacy commissioner of Ontario, Canada [73]. In a nutshell, it relies on seven core principles: proactivity, privacy as the default setting, privacy embedded in the design, full functionality, end-to-end security, visibility/transparency and user-centricity. In other words, it advocates for systems where privacy is integrated since the beginning as a requirement and by default, where the interests of the user come first, and without sacrificing the quality of service. Despite seeming utopian, this goal is actually reachable as soon as we throw away the LBS stack as we know it today. All LPPMs we surveyed so far in Section 2.5 rely on altering mobility data one way or another to protect sensitive information. With privacy by design architectures, there is no need anymore to alter mobility data, as the LBS itself integrates privacy as a first class citizen. The main drawback is that such solutions cannot be integrated with existing LBSs, they *are* LBSs by themselves.

A family of solutions relies on distributed computations and cryptographic methods to solve specific problems. For example, Popa et al. [123] introduced *PrivStats*, a system that is used to collect location-based aggregate statistics within defined geographic areas. Users collaborate to send pre-aggregated and encrypted data to the LBS, which allows to hide the number of tuples and the time at which they were collected. The LBS receives a constant number of (encrypted) values at fixed time intervals, combines them by using homomorphic encryption and asks a user to decrypt the final aggregate value. Authors also propose a privacy-preserving accountability protocol without any trusted party to prevent clients from cheating. Other works are interested in solving the problem of locating nearby friends, such as [97, 163].

Private information retrieval (PIR), first theorized by Chor et al. [27], is a cryptographic schema allowing someone to retrieve a record from a database without letting it know which record he wants to retrieve. Ghinita et al. [53] proposed to apply PIR to N-nearest neighbors spatial queries, that can be used for example to look for nearby venues (e.g., restaurants, monuments). They introduced a way to index spatial information in a PIR-compliant way by using Hilbert space-filling curves.

Garbled circuits where theorized by Yao [155] and allow two parties to privately evaluate the result of a generic function. Carter et al. [20] proposed a way to outsource the evaluation of such garbled circuits. Since they require a high computational power, outsourcing their evaluation in the cloud allows to speed up the processing, and eventually let mobile devices use them. The challenge is to preserve privacy guarantees even with an untrusted cloud. As an example, the authors implemented a privacy-preserving navigation application that mainly consists in a Dijkstra shortest-path algorithm used to privately get directions between two (private) points while taking into account (private) hazards that can occur along the path.

Finally, a last family of solutions is dedicated to proposing brand new architectures for LBSs, integrating privacy as a primary constraint. For example, *Koi* [62] is a platform proposed by Guha et al. It relies on two non-colluding servers, namely the matcher and the combiner. The matcher knows about entities (i.e., users and venues) and locations but nothing about links between them (i.e., which location belongs to which entity). The combiner knows the mappings between entities and locations but nothing about the

actual content of these entities and locations. A communication protocol between the matcher and the combiner allows to answer queries by performing a privacy-preserving matching. Instead of directly querying Koi, mobile devices set up triggers reacting to some events (e.g., getting notified when there is a restaurant at less than 500 meters). Application developers must hence create event-centric applications instead of location-centric applications. Other works working on privacy-by-design LBSs include [74, 119].

Some of these solutions are very specialized, solving one use case, e.g., *Louis, Lester and Pierre* [163] which addresses the problem of detecting nearby friends, while some other solutions are rather generic, e.g., *Koi* [62] which provides a platform on which to build LBS-like applications. Privacy-by-design is visionary, and we do hope that such architectures will prevail in the next decades, because they most certainly provide the best privacy/utility trade-off that is possible to achieve. However, we are still far from this point. These solutions address a different problem than ours, because they do not provide integrability with existing LBSs, which is one of our goals. Indeed, all of these solutions intend to either suppress (in the case of peer-to-peer protocols) or replace LBSs as we know them today, while our research problem is to add privacy in existing workflows, by protecting data before it is actually sent to an LBS or an analyst.

2.6.2 Privacy-preserving query engines

Instead of releasing the whole dataset during the publication phase, an alternative approach is to let analysts send queries over a dataset and only provide them aggregated results. A privacy-preserving query engine implements this pattern and additionally adds a privacy layer by ensuring that returned results do not breach privacy of individuals.

Privacy INtegrated Queries (abbreviated PINQ) [101] is a general-purpose analytics platform allowing to execute queries against a data source while preserving privacy through differential privacy. The data analyst writes his queries, specifies a privacy budget ϵ that can be consumed, and the platform automatically takes care of returning results satisfying ϵ-differentially privacy. One of the proposed examples illustrates geo-located queries and shows that PINQ can be successfully applied in this context. It is implemented as a C# library. Pelekis et al. proposed *Hermes++* [117], which is a privacy-preserving query engine for mobility data. It explicitly targets tracking attacks, in which an analyst may attempt to reconstruct the mobility trace of a specific user. It relies on the injection of dummies in results, these dummies being designed to have a similar behavior than actual users. This engine also has an auditing module that is able to detect if a sequence of queries can be harmful for the privacy of individuals.

Again, these solutions tackle a different problem than ours. They overcome the publication problem by not publishing at all the dataset. While this can be suitable for some use cases, publishing entire datasets gives more flexibility to analysts. Indeed, with a query engine an analyst is limited by the expressiveness of a query language at his disposal, while an entire dataset gives him the power to implement whatever task he wants to in whatever language is more suited.

2.7 Summary

Before us, location privacy had already been reviewed in several surveys. While both Krumm [82] and Shin et al. [137] published general surveys, Terrovitis [145] and Wernke et al. [151] followed an approach centred around location privacy attacks. However, only few of these papers address the evaluation of protection mechanisms. Moreover, previous surveys often focus either on the online or the offline scenario, although they all share similar properties and some LPPMs can fit in both cases. In this chapter, we surveyed the latest works about LPPMs. At the best of our knowledge, it is the first survey to propose a unified view on both online and offline LPPMs, and to highlight the way they are evaluated. Indeed, we performed an extensive work to analyze metrics commonly used when evaluating LPPMs, and showed in two synthetic tables which families of metrics were used for each LPPM under consideration. This shows that both kinds of LPPMs can be based on the same underlying primitives (e.g., differential privacy), while providing appropriate algorithms suited for the considered use case.

As seen in this survey, a prominent issue is the evaluation of LPPMs, because there is no standard metric that has ever been used for all protection mechanisms. It is hence very difficult to measure the practical efficiency of an LPPM, besides the particular use case and assumptions made by their authors. In the next chapter, we resume our state-of-the-art work by surveying and formally defining evaluation metrics across the categories presented in this chapter. We then use these metrics to conduct an experimental evaluation of a state-of-the-art LPPM, geo-indistinguishability [9], to assess its practical efficiency.

CHAPTER 3

Practically Evaluating Protection Mechanisms

3.1 Introduction

As presented in Chapter 2, literature is far from poor in terms of LPPMs. They are very diverse, accommodating with various use cases, providing different guarantees in terms of privacy, utility and performance, while requiring varying architectures. Metrics that are used to evaluate them are as diverse; even with our categorization (cf. Table 2.2 and Table 2.3 in Section 2.5) we could not extract any meaningful tendency.

Because of this large variety of disparate metrics, there is a need to propose an evaluation methodology, both to compare existing LPPMs and to evaluate our own subsequent propositions. The only work we are aware of interested in proposing a way to evaluate LPPMs is [139]. Shokri et al. proposed a privacy evaluation framework, exploiting privacy attacks to assess the efficiency of an LPPM. For that purpose, they proposed several attacks: two tracking attacks, whose goal is to reconstruct mobility traces of a particular user, a localization attack, whose goal is to find the location of a particular user at a given time, and a meeting disclosure attack, whose goal is to determine whether a pair of users met at given place and time. However, this framework is limited to only privacy evaluation, and does not consider utility or performance. We advocate that these two other families of metrics are complimentary to privacy metrics and should always be considered together to evaluate the efficiency of any LPPM. Furthermore, their framework is too restrictive compared to our research problem. Indeed, they consider a strict probabilistic framework in which LPPMs are defined as functions from $\mathcal{E}$ to $\mathcal{E}$ whose Probability Density Function is known. For our work, we need to consider any LPPM as defined in Section 2.2.3, i.e., a function from $\mathcal{D}$ to $\mathcal{D}$, without any other assumption about what happens inside that black box.

In this section we propose our own evaluation metrics, organized against the classification proposed in Section 2.4: two privacy metrics, four utility metrics and one performance metric. These metrics are either state-of-the-art metrics or original metrics. For state-of-the-art metrics, we refer to the paper they originate from and the modifications we possibly made. Indeed, metrics are not always clearly defined in papers, thus preventing to reproduce experiments. In this section, we define without any ambiguity all these metrics, going from their formal definition to implementation details. Furthermore, it is worth noting that the model and the metrics are implemented as a software tool, Accio, that we detail later in Chapter 6. Then, we evaluate a state-of-the-art LPPM, geo-indistinguishability [9], against our metrics. Indeed, despite it providing ϵ-differential privacy, we want to answer more practical questions such as its resiliency against privacy attacks. Through this case study, we draw two conclusions: (1) POIs are really sensitive and should be protected, because they open the way to practical and effective attacks such as inferring semantic knowledge or re-identifying users; (2) there is indeed a trade-off between privacy and utility, which means that choosing the right ϵ is of great importance.

The remaining of this chapter is structured as follows. We introduce formally a set of privacy metrics in Section 3.2, utility metrics in Section 3.3 and performance metrics in Section 3.4. We then present mobility datasets commonly used to evaluate LPPMs in Section 3.5. Section 3.6 is dedicated to the practical evaluation of geo-indistinguishability with respect to the metrics we defined previously. We finally conclude this chapter in Section 3.7.

In a nutshell. Our original contributions (related to contribution **C1**) in this chapter are the following:

- A formal definition of seven privacy, utility and performance LPPM evaluation metrics;
- A case study where these metrics are practically used to evaluate a state-of-the-art LPPM, geo-indistinguishability [9].

Associated publication: [127].

3.2 Privacy metrics

We present in this section two metrics that can be used to evaluate LPPMs in terms of privacy. Metrics defined here are elements of $\mathcal{M}$, as defined in Section 2.2.4.

3.2.1 Extracting POIs

POIs (defined in Section 3.2.1) are sensitive pieces of information, and as such can be used to quantify a privacy leakage. Because all of our privacy metrics rely on POIs, we first define how to extract them from a dataset. While there already exists several algorithms in the literature for this task, e.g., [65, 164], we propose our own algorithm, inspired by those previous works. The novelty of our algorithm is to take into account multiple appearances of a user inside a POI and allow to enforce a minimum frequency.

Our POIs extraction routine is depicted in Algorithm 1. It processes in two parts: stays are extracted and then aggregated into POIs. A stay corresponds to a passage inside one POI (as defined by [65]), and we only keep stays with multiple occurrences. The first part (lines 1-17) focuses on the extraction of stays. Identifying stays requires two parameters:

1. A time threshold Δt, which represents the minimum time that has to be spent in every stay. Its value should depend on the purpose of the extraction algorithm. Indeed, one might be interested in considering short stays (e.g., to identify visits to shopping malls) or longer stays (e.g., to identify holiday periods).
2. A distance threshold $\Delta \ell$ representing the maximal diameter[1] of the stay area. Once again, it should be set according to the granularity of information to capture. Moreover, it should be consistent with Δt. For example, there is (most likely) no interest in capturing stays of 1 day within a 50 meters diameter, which is not very likely to happen.

Stays are extracted by iterating over the mobility trace and building successive candidate stays. Our algorithm tests for each event if by adding it to the candidate stay, the diameter of the latter remains under the $\Delta \ell$ threshold (lines 6-7). By convention, if the candidate stay is empty, the latter test succeeds. If not satisfied (lines 10), we check if the elapsed time inside the candidate stay is above the Δt threshold. If so, the

[1] The diameter of a set of locations is the distance between the two farthest locations of this set.

Algorithm 1 Extracting POIs from a mobility trace.

function ExtractPois($u \in \mathcal{U}$, $t \in \mathcal{D}_u$, $\Delta t \in \mathbb{R}^+$, $\Delta\ell \in \mathbb{R}^+$, $minPts \in \mathbb{N}^+$)
  $stays \leftarrow \varnothing$ ▷ Stays extracted so far
  $events \leftarrow \varnothing$ ▷ Events candidate to form a stay
  $i \leftarrow 1$
  while $i \leq |t|$ **do**
    $d \leftarrow \max_{e \in events} d_{\mathcal{X}}(\mathbf{loc}(t_i), \mathbf{loc}(e))$
    if $d \leq \Delta\ell$ **then**
      $events \leftarrow events \cup \{t_i\}$
      $i \leftarrow i + 1$
    else
      if $\max(\vec{\mathbf{time}}(e)) - \min(\vec{\mathbf{time}}(e)) \geq \Delta t$ **then**
        $stays \leftarrow stays \cup \{$ Centroid($\vec{\mathbf{loc}}(e)$) $\}$
        $events \leftarrow \varnothing$
      else
        $events \leftarrow events \setminus \{e \in events \mid \mathbf{time}(e) = min(\tilde{\mathbf{time}}(e))\}$
  if $\max(\vec{\mathbf{time}}(e)) - \min(\vec{\mathbf{time}}(e)) \geq \Delta t$ **then**
    $stays \leftarrow stays \cup \{$ Centroid($\vec{\mathbf{loc}}(e)$) $\}$

  $clusters \leftarrow \varnothing$ ▷ Final clusters (i.e., POIs)
  for $stay$ in $stays$ **do**
    $neighborhood \leftarrow \{s \in stays \mid d_{\mathcal{X}}(s, stay) \leq 0.5 \times \Delta\ell\}$
    if $|neighborhood| \geq minPts$ **then**
      for $cluster$ in $clusters$ **do**
        if $neighborhood \cap cluster \neq \varnothing$ **then**
          $neighborhood \leftarrow neighborhood \cup cluster$
          $clusters \leftarrow clusters \setminus \{cluster\}$
      $clusters \leftarrow clusters \cup \{neighborhood\}$
  return $\bigcup_{c \in clusters}$ Centroid(c)

candidate stay is valid and added to the list of valid stays (line 12) and a new candidate stay is created (line 13). If not, we remove the first element of our candidate stay (line 15) and try again to add the current event at the next iteration.

In order to merge frequent and nearby stays, we use in the second part of our algorithm our own version of the DJ-clustering algorithm [164] (lines 19-27). This algorithm creates clusters with a maximal number of locations and at a minimal distance from other clusters. In its original version, this algorithm uses a preprocessing step to filter out static points (i.e., points where the speed of the user is zero). We skip this step because we are working on stays, which already represent locations where the user is almost not moving. This algorithm relies on two parameters:

1. A merge threshold which defines the maximum distance under which two distinct clusters are merged into a single one. It is defined as a function of $\Delta\ell$ in our algorithm. We fixed it as 50 % of the distance threshold, in order to merge nearby stays having half of their area in common.
2. A minimum number of stays $minPts$ necessary to create a POI. It gives us the notion of frequency of apparition of a stay and helps to eliminate "accidental" stays that occur only a few times. By default, we assume that $minPts = 1$ if its value is not explicitly specified, i.e., all stays are taken into account even if they occur only once.

Finally, the Centroid function gives return the centroid of a set of locations, i.e., the single location that represents the arithmetic mean of all those locations. Its actual implementation depends on the $d_{\mathcal{X}}$ distance function and the way locations are represented (e.g., euclidian points or latitude/longitude pairs).

3.2.2 Data distortion: POIs retrieval

We define a POIs retrieval metric which compares the amount of POIs that are retrieved from a dataset, before and after applying an LPPM. To do that, we use classical information retrieval metrics, traditionally used to evaluate the effectiveness of search engines: precision, recall and F-Score. To practically compute these metrics, we need to define the equality between POIs. As POIs are modeled as locations, we could simply check is they represent the same point on Earth; however this is not sufficient, as it is not likely that we find the exact same locations before and after applying an LPPM (e.g., the probability of retrieving the exact same decimals for a latitude/longitude pair is almost null in practice). Still, two close locations (e.g., a few meters apart) can designate the same POI. This is why we consider two POIs to be the same if the distance between them is less than a σ threshold.

Definition 4. *We consider there is a function used to extract POIs from a trace of any user $u \in \mathcal{U}$ such that:*

$$pois : \mathcal{D}_u \longrightarrow \mathcal{P}(\mathcal{L}).$$

Definition 5. *The remap operation associates to each location of a given set $L' \in \mathcal{P}(\mathcal{L})$ the closest location of another set $L \in \mathcal{P}(\mathcal{L})$, if and only if its distance is at most a threshold σ:*

$$remap_\sigma(L, L') = \bigcup_{\ell' \in L'} \left\{ \arg\min_{\ell \in L} \{ d_{\mathcal{X}}(\ell, \ell') \mid d_{\mathcal{X}}(\ell, \ell') \leq \sigma \} \right\}.$$

Definition 6. *POIs recall, with respect to a $\sigma \in \mathbb{R}^+$ threshold, is the ratio between the number of POIs from a protected trace $\hat{t} \in \mathcal{D}_u$ of any user $u \in \mathcal{U}$ that can be remapped to a POI from the actual trace $t \in \mathcal{D}_u$, and the number of POIs from the actual trace t:*

$$PoisRecall_\sigma(t, \hat{t}) = \frac{|pois(t) \cap remap_\sigma(pois(t), pois(\hat{t}))|}{|pois(t)|}.$$

POIs precision, with respect to a $\sigma \in \mathbb{R}^+$ threshold, is the ratio between the number of POIs from a protected trace $\hat{t} \in \mathcal{D}_u$ of any user $u \in mathcalU$ that can be remapped to a POI from the actual trace $t \in \mathcal{D}_u$, and the number of POIs from the protected trace $\hat{t}$:

$$PoisPrecision_\sigma(t, \hat{t}) = \frac{|pois(t) \cap remap_\sigma(pois(t), pois(\hat{t}))|}{|pois(\hat{t})|}.$$

POIs retrieval, with respect to a $\sigma \in \mathbb{R}^+$ threshold, is the vector of POIs F-Score, i.e., the harmonic mean of POIs precision and POIs recall, for each user, between POIs extracted from traces of a protected dataset $\hat{d} \in \mathcal{D}$ and POIs extracted from traces of the actual dataset $d \in \mathcal{D}$:

$$PoisRetrieval_\sigma(\hat{d}, d) = \left(\frac{2 \times PoisPrecision_\sigma(d_u, \hat{d}_u) \times PoisRecall_\sigma(d_u, \hat{d}_u)}{PoisPrecision_\sigma(d_u, \hat{d}_u) + PoisRecall_\sigma(d_u, \hat{d}_u)} \right)_{u \in \widehat{\mathbf{user}}(\hat{d})}.$$

Besides the σ parameter, POIs retrieval is also dependent on the way POIs are extracted. In practice, we use the ExtractPois function presented in Algorithm 1 to implement the *pois* function. It means that its parameters Δt, $\Delta \ell$ and $minPts$ (cf. Section 3.2.1 for more details on their meaning) impact POIs retrieval's results. Therefore, their values have to be considered when analyzing POIs retrieval's results.

The idea of such a metric was proposed in [49], where they evaluated the correctness of a POI by manually labelling some places (e.g., an airport, a mall) as ground truth POIs. Because this approach does not scale and is only limited to datasets where ground truth can be easily identified (though requiring extra human effort), we chose to use the POIs from the actual dataset as ground truth.

3.2.3 Attack correctness: Re-identification success

Re-identifying users has been the subject of several previous works (cf. Section 2.3.3). The goal is to associate a mobility trace without any personal identifier to the identity of an individual. To do so, we assume that attackers have access to background knowledge, modeled as a dataset composed of unprotected data previously collected by any mean. This allows an attacker to learn knowledge from an unprotected dataset (acting as a *training dataset*), before applying their hypothesis on a protected dataset (acting as a *testing dataset*). This model gives us a rather strong attacker, allowing us to evaluate worst cases scenarii. We propose a metric using the proportion of users an attacker is able to re-identify as a privacy quantifier. We count a protected trace as re-identified if it is possible to unambiguously associate it to a known user identifier. To achieve this goal, an attacker extracts POIs from each trace and attempts to match them with the

closest POIs from the dataset of known users, thanks to a similarity function that we detail just after. We assume that the set of users is fixed and known to the attacker, i.e., there is no new user entering the system.

The following scenario illustrates a practical situation in online use cases motivating these choices. Let us consider an LBS that has already collected actual mobility data from a set of users: it forms its training dataset. Later, these users start using an LPPM and send protected locations to this very same LBS. The LBS is now collecting protected locations from a set of known users: it forms its testing dataset. With help from the training dataset, the LBS may be able to learn enough knowledge to de-anonymize the testing dataset, and hence ultimately associate back users from the testing dataset to users of the training dataset. This is a particularly severe threat, as it could basically be useless for a user to suddenly start using an LPPM, if he was sending unprotected traces before.

Definition 7. *The similarity between two sets of POIs $L, L' \in \mathcal{P}(\mathcal{L})^2$ is defined as follows:*

$$sim(L, L') = median(\{\min_{\ell' \in L'} d_{\mathcal{X}}(\ell, \ell') \mid \ell \in L\} \cup \{\min_{\ell \in L} d_{\mathcal{X}}(\ell, \ell') \mid \ell' \in L'\}).$$

The re-identification function associates to a trace $\hat{t} \in \mathcal{D}_u$ of any user $u \in \mathcal{U}$ the most similar user from a training dataset $d \in \mathcal{D}$:

$$reident(\hat{t}, d) = \underset{u \in \mathbf{user}(d)}{\arg\min}\ sim(pois(\hat{t}), pois(d_u)).$$

The re-identification success is defined as the vector of booleans determining whether each user from the protected dataset $\hat{d} \in \mathcal{D}$ is correctly re-identified from the actual dataset $d \in \mathcal{D}$:

$$PoisReident(\hat{d}, d) = \begin{pmatrix} 1 \text{ if } reident(\hat{d}_u, d) = u \\ 0 \text{ otherwise} \end{pmatrix}_{u \in \mathbf{user}(\hat{d})}.$$

We remind that we made the assumption (cf. Section 2.2.4) that evaluated LPPMs did not change user identifiers. Moreover, we use here the actual dataset as a training dataset, containing the knowledge about how users usually move. That models an omniscient attacker who knows everything about the users, though if he was really omniscient he would not need to perform any privacy attack. We could define a more subtle metric differentiating between training and testing dataset, the training dataset being formed of knowledge (maybe partial) about past locations, while the testing dataset contains recent and protected locations. In the latter case, the training and testing dataset would not overlap temporally. However, we let this as future work and use the metric as defined above.

Using re-identification attacks as metrics has been already explored in literature, e.g., in [51, 139]. However, the attack we propose here is original, though inspired by these previous works.

3.3 Utility metrics

This section introduces four metrics that can be used to evaluate LPPMs in terms of utility. Once again, all metrics defined here are elements of $\mathcal{M}$, as defined in Section 2.2.4.

3.3.1 Data distortion: Spatial distortion

Spatial distortion is the quantification of spatial error between actual traces and protected traces. It quantifies the uncertainty coming from protected traces: every location that was not present in the actual dataset (whether it has been altered or created) degrades the expected quality of service. Spatial distortion is a distance, expressed in the same unit than the result of $d_{\mathcal{X}}$.

Definition 8. *Spatial distortion between traces of an actual dataset $d \in \mathcal{D}$ and traces of a protected dataset $\hat{d} \in \mathcal{D}$ is the vector of the average distance between locations of the protected traces and the closest location from the actual trace, for each user:*

$$SpatialDistortion(\hat{d}, d) = \left(\frac{\sum_{\hat{e} \in \hat{d}_u} \min_{e \in d_u} d_{\mathcal{X}}(\mathbf{loc}(e), \mathbf{loc}(\hat{e}))}{|\hat{d}_u|} \right)_{u \in \mathbf{user}(\hat{d})}.$$

Moreover, we define a modified version of spatial distortion that consider the projection of the protected trace onto the actual trace before computing the distance between locations. This allows to consider two traces having the same shape (i.e., following the same path but with events placed at different locations on this path) as identical. However, it works best with traces having a high sampling rate. If the sampling rate is too low (e.g., one event every 5 minutes), extrapolating the path between two consecutive events as a straight line will likely not match the actual trajectory.

Definition 9. *We define the path of a trace $t \in \mathcal{D}_u$ of any user $u \in \mathcal{U}$ as all locations belonging to a segment between two consecutive events:*

$$path(t) = \{\ell \in \mathcal{L} \mid \exists i \in \mathbb{N}, i \leq |t| \wedge d_{\mathcal{X}}(\mathbf{loc}(t_i), \ell) + d_{\mathcal{X}}(\ell, \mathbf{loc}(t_{i+1})) = d_{\mathcal{X}}(\mathbf{loc}(t_i), \mathbf{loc}(t_{i+1}))\}.$$

The projected spatial distortion between traces of an actual dataset $d \in \mathcal{D}$ and traces of a protected dataset $\hat{d} \in \mathcal{D}$ is the vector of the average distance between locations of the protected trace and their projections on the actual trace, for each user:

$$SpatialDistortion'(\hat{d}, d) = \left(\frac{\sum_{\hat{e} \in \hat{d}_u} \min_{\ell \in path(d_u)} d_{\mathcal{X}}(\ell, \mathbf{loc}(\hat{e}))}{|\hat{d}_u|} \right)_{u \in \mathbf{user}(\hat{d})}.$$

The idea of using spatial distortion as a metric is widely spread in the literature, e.g., in [6], we formalized it here.

3.3.2 Data distortion: Compression degree

Applying an LPPM can change the number of events a dataset contains. It can be smaller, if events have been deleted, or larger, if dummy events have been added. Producing datasets that are orders of magnitude larger than the actual one greatly decreases their usability, because the time needed to load and query them increases accordingly, while much smaller ones can introduce information losses (that could be quantified with the previous utility metrics).

Definition 10. *Compression degree is the singleton vector*[2] *containing the ratio between the size of the actual dataset $d \in \mathcal{D}$ and the size of a protected dataset $\hat{d} \in \mathcal{D}$:*

$$CompressionDegree(\hat{d}, d) = \left(\frac{|d|}{|\hat{d}|} \right).$$

The idea of such a metric was proposed in [6].

3.3.3 Task distortion: Count query distortion

A classical operation performed on a dataset is to count how many unique users cross a specific area during a given period of time. Despite being simple, this operation can be employed by many useful applications (e.g., traffic prediction, finding popular places). More specifically, to measure the utility related to count queries, we use the *count query distortion*, which computes the dissimilarity between results of a count query on the actual dataset and on its protected counterpart.

Definition 11. *Let $\mathcal{Q}$ be the set of all possible count queries. A count query $q \in \mathcal{Q}$ is a function $\mathcal{D} \longrightarrow \mathbb{N}$ that returns the number of distinct users that were present inside a specific area during a given period of time.*

Definition 12. *The distortion associated with a vector of n count queries $Q \in \mathcal{Q}^n, n \in \mathbb{N}$ is the vector of the relative error between its result over the actual dataset $d \in \mathcal{D}$ and over a protected dataset $\hat{d} \in \mathcal{D}$, for each query:*

$$QueryDistortion_Q(\hat{d}, d) = \left(\frac{|q(d) - q(\hat{d})|}{q(d)} \right)_{q \in Q}.$$

A count query is specified with two parameters: a time window and a geographical area. To compute a meaningful distortion, we can either use a set of statically defined queries, or generate them randomly. To minimize the evaluation work, we chose the latter. Our random query generator comes with two parameters: a range for time window widths and a range for geographical area sizes. It takes care of generating only relevant queries for which the result on the actual dataset is strictly positive.. Our smart count query generator follows a simple algorithm:

1. Draw a random event from the actual dataset.

[2] We define the result of this metric as a singleton to match the definition of a metric that was given in Section 2.2.4, stating that the result of a metric is a vector of reals.

2. Draw a random time window width and a random area size.
3. Generate a count query centered around this event and with this time window width and area size.

This ensures that for any generated query q applied on an actual dataset $d \in \mathcal{D}$, $q(d) \neq 0$, and guarantees that our distortion is always defined. Moreover, because this process in inherently random and hence can cause results to vary largely from one evaluation to another, we usually generate many queries and aggregate the results. In practice, we generate 1000 random queries in our experiments and report the average distortion.

The idea of such a metric has been first proposed in [1], though details provided in the paper are not sufficient to re-implement it as-is. Our contribution is to define precisely how these queries are generated. Indeed, there is an infinite number of possible queries, a large part of which have a result of 0, which makes totally randomly generated queries impracticable.

3.3.4 Task distortion: Area coverage

LPPMs modify mobility data to protect sensitive information. Consequently, they may remove or strongly alter locations considered as too sensitive for the user, or report fake events at locations a user never went to. This ultimately results in an alteration of the utility of protected data and may reduce the resulting quality of service. To take into account this side effect on utility, we introduce the notion of *area coverage*. Roughly speaking, considering a discrete division of the world into cells, it quantifies the overlap between cells for which there is data in the protected traces and cells for which there is data in the actual traces. Similarly to previous metrics, we use an F-Score to take into account both the proportion of cells for which there is still data in the protected traces and the proportion of cells from which we wrongfully receive data in the protected traces.

Definition 13. *Let $\mathcal{C}$ be the set of all possible cells. Each event belongs to one and only one cell. We consider there is a function that associates to an event the cell it belongs to, such that:*

$$cell : \mathcal{E} \longrightarrow \mathcal{C}.$$

Definition 14. *Area recall is the ratio between the number of cells of a protected trace $\hat{t} \in \mathcal{D}_u$ of any user $u \in \mathcal{U}$ corresponding to a cell of the actual trace $t \in \mathcal{D}_u$, and the number of cells of the actual trace t.*

$$AreaRecall(t, \hat{t}) = \frac{|\vec{cell}(\hat{t}) \cap \vec{cell}(t)|}{|\vec{cell}(t)|}.$$

Area precision is the ratio between the number of cells of a protected trace $\hat{t} \in \mathcal{D}_u$ of any user $u \in \mathcal{U}$ corresponding to a cell of the actual trace $t \in \mathcal{D}_u$, and the number of cells of the protected trace $\hat{t}$.

$$AreaPrecision(t, \hat{t}) = \frac{|\vec{cell}(\hat{t}) \cap \vec{cell}(t)|}{|\vec{cell}(\hat{t})|}.$$

Area coverage is the vector of area F-Score, i.e., the harmonic mean of area precision and area recall, between cells coming from traces of a protected dataset $\hat{d} \in \mathcal{D}$ and cells coming from traces of the actual dataset $d \in \mathcal{D}$, for each user:

$$AreaCoverage(\hat{d}, d) = \left(\frac{2 \times AreaPrecision(d_u, \hat{d}_u) \times AreaRecall(d_u, \hat{d}_u)}{AreaPrecision(d, \hat{d}_u) + AreaRecall(d, \hat{d}_u)} \right)_{u \in \mathbf{user}(\hat{d})}.$$

This definition can accommodate of various *cell* functions. In practice, we use Google's S2 geometry library [133] to implement it and generate cells of various sizes. This library is able to generate cells from a latitude and longitude at different levels, with the interesting property of cells having a similar area wherever they are on the globe. Levels range from 0 (the whole world) to 30 (a few squared millimeters). It means that when discussing of the area coverage metric, we have to specify the level at which it is taken.

We took inspiration from a metric proposed in [1], whose goal was to perform sequential patterns data mining task. We only kept the first part of their metric, consisting in discretizing the space through the usage of a grid. A similar area coverage metric was also proposed in [6]. Our main contribution is to formalize the way that grids are generated, making them adaptable to several datasets. We also use again well-known information retrieval metrics to take into account not only the recall but also the precision.

3.4 Performance metrics

This section defines one metric used to evaluate LPPMs in terms of performance.

3.4.1 Execution time: Wall time

The wall time is the duration taken by an LPPM to generate a protected dataset from the actual dataset, as measured by a clock. We assume the actual dataset is already stored somewhere accessible (e.g., a database or a filesystem) and measure the time it takes to read it, protect it and write back its protected version. Depending on the use case, the meaning and importance of this metric is largely different. In the online use cases, it directly impacts the request latency and hence the user experience, because the LPPM is applied before actually sending any request to an LBS. In the offline use cases, it is the time taken by an asynchronous process to complete. In these cases, the execution time is less crucial, although it still gives a hint about the performance of an algorithm (e.g., it should not take several days to protect one day of mobility traces), and more importantly impacts the company's business (because of the cost in terms of computational power occurred by this processing).

Obviously, this metric only gives a rough estimate of the efficiency of an algorithm. It is highly impacted by the performance of the storage holding the dataset, as well as computational resources at disposal. Consequently, we take care of using the same setup to allow a fair comparison between LPPMs, and precise the hardware on which experiments were executed.

3.5 Mobility datasets

Once that we got metrics, the last missing piece towards a successful evaluation is mobility data. Because we do not have access to any production LBS, we need already collected and realistic datasets. Several initiatives have been conducted to publicly provide datasets coming from real-life data collections. Table 3.1 lists some of these datasets with their characterizing features.

Table 3.1: Datasets of mobility traces.

Dataset	**Region**	**Time span**	**#users**	**#events**
Cabspotting	San Francisco, USA	1 month	536	11 million
MDC	Geneva, Switzerland	3 years	185	11 million
Geolife	Beijing, China	5,5 years	178	25 million
T-Drive	Beijing, China	1 week	10,357	15 million
Priva'Mov	Lyon, France	15 months	100	156 million
Brightkite	World	1,5 years	58,228	4 million
Gowalla	World	1,5 years	196,591	6 million

The Cabspotting dataset [121] contains GPS traces of taxi cabs in San Francisco (USA), collected in May 2008. The Geolife dataset [162] gathers GPS trajectories collected from April 2007 to August 2012 in Beijing (China). The MDC dataset [79, 85] involves 182 volunteers equipped with smartphones running a data collection software around Geneva, Switzerland, between 2009 and 2011. A privacy protection scheme based on k-anonymity has been performed on the actual data before releasing the MDC dataset. As described in [85], this privacy preserving operation includes many manual operations which have obviously an impact on the outcome of LPPMs, but these impacts are difficult to fully understand. It includes not only locations coming from the GPS sensor, but also data from various other sensors (e.g., accelerometer, battery). T-Drive [158,159] is another dataset collected in Beijing and featuring taxi drivers. It features a high number of users (more than 10,000) over a very short period of time (one week).

This thesis was funded by a projet named Priva'Mov [130], whose goal was to offer a collection platform of mobility traces for researchers. As a result, we also have our own dataset, available upon request after signing an NDA. Unfortunately, it was only made available at the end of the timespan dedicated to this thesis, which explains why we could not use it in experiments we present. This dataset followed 100 users during 15 months and contains a total of 156 million events [13]. Users were recruited mainly among students and staff of INSA and other universities at Lyon, France. Besides GPS locations, it also information coming from cellular networks, Wi-Fi routers, the accelerometer and the battery.

Other datasets come from geolocated social networks, rather than from a custom data collection campaign ran by academics, and as such provide check-ins events that allow to build sparse mobility traces for these users. Two datasets are available in this category, coming from the (now closed) Brightkite and Gowalla [26, 87] social networks. They contain 4 million and 6 million check-ins collected between February 2009 and October 2010. These datasets also come with a graph modeling relationships between users in the social network.

In the remaining of this thesis, Cabspotting, Geolife and MDC datasets will be used in our experimental evaluations.

3.6 Case study: Practical assessment of an LPPM

In this section, we study the behavior of a representative state-of-the-art LPPM, geo-indistinguishability [9] (abbreviated Geo-I). As a reminder, this a perturbation-based LPPM offering differential privacy guarantees, but specifically tailored for location privacy. We choose this one because it is a recent one, with a good traction, and based on differential privacy which appears more and more as the *de-facto* standard in privacy. Moreover, it will be used several times as a competitor or baseline in the next chapters; it is interesting to start by analyzing it thoroughly now.

We do not aim at analyzing the theoretical guarantees of Geo-I as the latter have already been formally proven by its authors. Instead, we aim at practically evaluating the degree of protection offered by Geo-I of used to protect mobility data. Towards this purpose, we use some of the evaluation metrics we introduced in previous sections. Because those were not considered by the authors of Geo-I in their paper, it is interesting to see how their LPPM behaves in a new situation. We experiment with all the metrics presented in this chapter: privacy (Section 3.6.2), utility (Section 3.6.3) and performance (Section 3.6.4).

3.6.1 Experimental setup

Dataset

We use the Cabspotting real-life dataset (cf. Section 3.5) to evaluate Geo-I, which followed 536 taxi drivers during a month, while Geolife contains data for 178 users over three years. This dataset was preprocessed to enforce a minimum duration of 5 minutes between consecutive points (to have a lower the sampling rate), and traces were split in two new traces, belonging to new virtual users, when there was a pause (i.e., no activity) of more than 6 hours. Furthermore, we only kept traces with at least 15 minutes of activity. This results in 11,951 smaller traces, and a higher granularity in the results that having only 536 traces, and in 273,063 events.

Parametrization

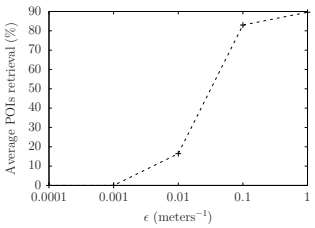
We parametrize Geo-I with $\epsilon \in \{0.0001, 0.001, 0.01, 0.1, 1\}$, to follow a logarithmic progression. Those values are in the same range of values that are typically used by the authors of this LPPM [9]. We remind that the lower ϵ, the higher the noise.

Implementation

Geo-I and the evaluation metrics are implemented on the Java Virtual Machine in Scala. Experiments were executed on a machine running Ubuntu 14.04, having access to 16 cores and 50 Gb of memory. The prototype will be presented in more details in Chapter 6.

Figure 3.1: Results of the privacy evaluation of Geo-I.

(a) POIs retrieval (lower is better)



(b) Re-identification success (lower is better)

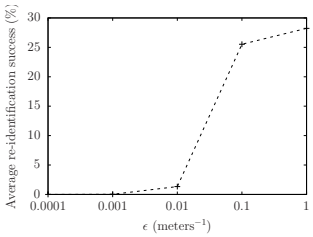
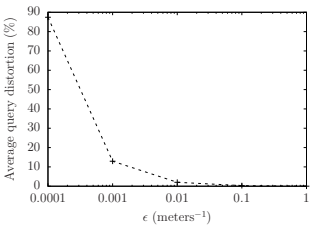
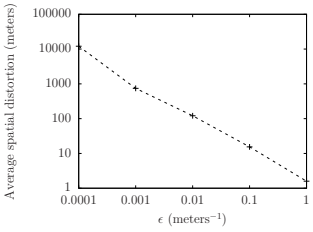


Figure 3.2: Results of the utility evaluation of Geo-I.

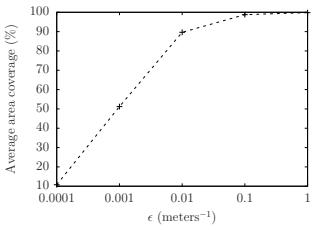
(a) Count query distortion (lower is better)



(b) Spatial distortion (lower is better)



(c) Area coverage (higher is better)



3.6.2 Privacy evaluation

We parametrize the POIs extraction (cf. Section 3.2.1), used by both the POIs retrieval and the re-identification success metric, with $\Delta t = 15$ minutes and $\Delta \ell = 200$ meters. It allows to capture typical activities occurring in areas of the size of a small neighborhood in an urban environment. We furthermore parametrize the POIs retrieval metric with $\sigma = \Delta \ell / 2 = 100$ meters, while the re-identification success metric does not need furthermore parametrization.

Results of the privacy evaluation are shown in Figure 3.1. Those graphs show the average value of the metrics, across all traces, varying for different values of ϵ. As expected, lower values of ϵ (and thus adding more noise) manage to protect privacy perfectly, in terms of POIs retrieval and re-identification success, while higher values of ϵ fail to protect POIs effectively. Still, the average re-identification success is no more than 28 % with $\epsilon = 1$, highlighting that even without distortion, re-identification is a not-so-easy task with the Cabspotting dataset. Indeed, as all taxi drivers share a large number of POIs (e.g., the airport, hotels), it becomes difficult to distinguish them with the heuristic used by our metric. However, it still does show that POIs can be a practical and efficient way to breach privacy, by allowing to re-identify up to 28 % of the users of a protected dataset.

3.6.3 Utility evaluation

We parametrize the count query distortion metric to generate queries with time windows ranging from 2 hours to 8 hours and with squared areas whose half-diagonals range from 500 to 5,000 meters (similarly to what was done in [2]). The area coverage metric is

parametrized to extract cells at the 13th level (i.e., areas of the size of a neighborhood in a urban environment). The spatial distortion metric requires no parameter, and we do not experiment with the compression degree metric as Geo-I does not change the size of the dataset.

Results of the utility evaluation are shown in Figure 3.2. Conversely to the privacy, lower values of ϵ result in a very degraded utility, with a high count query distortion, a high spatial distortion (up to 12 kilometers) and a low area coverage, while higher values of ϵ provide an almost perfect utility. The trade-off between privacy and utility appears very clearly when comparing Figure 3.1 and Figure 3.2. Overall, at $\epsilon = 0.01$, Geo-I behaves rather well with a low POIs retrieval (16 %) and a low re-identification rate (1.3 %) on the privacy side, associated with a high area coverage (89.7 %) and a low count query distortion (2.04 %) on the utility side. Only the spatial distortion is not ideal, with 121 meters, though not catastrophic. This shows that, with respect to some metrics, it may be possible to find (at least empirically, as we did here) a configuration that provides a satisfactory trade-off between privacy and utility, though it did not seem obvious *a priori* that $\epsilon = 0.01$ would be a good candidate.

3.6.4 Performance evaluation

Finally, we measured the execution time of Geo-I. With the Cabspotting dataset pre-processed as previously described, it took 16 seconds to protect it entirely. As already outlined, these results are particularly sensitive to the actual implementation of the algorithm (e.g., the degree of parallelization, the mathematical libraries used) and the hardware on which it runs (e.g., the number of cores at disposal). However, given that the 273,063 events contained in the Cabspotting dataset were processed in 16 seconds, and given that Geo-I scales linearly in the number of events[3], the latter seems reasonably practicable and efficient in protecting even large datasets, at a rate of 59 μs/event with our implementation.

3.7 Summary

In this chapter, we introduced a set of metrics aimed at quantifying privacy, utility and performance of LPPMs. We also presented six mobility datasets that are often used to experimentally evaluate LPPMs. To demonstrate the usefulness of those building blocks, we completed this chapter with a practical evaluation of a differentially private LPPM, Geo-I [9]. We leveraged all our metrics to assess its practical efficiency, going further than its strong theoretical guarantees. We believe that the two approaches (theoretical and practical) are complimentary, and provide different points of view over the same LPPM. The main takeaways from this chapter are: (1) POIs are sensitive information that has to be protected with care; (2) there is a fundamental trade-off between privacy and utility, in which the ϵ parameter, which controls the amount of noise, is of great importance.

These first results define our roadmap for the next chapters. As a follow-up to (1), Chapter 4 will explore the design of an LPPM specifically designed to hide POIs from

[3]Because Geo-I processes each event independently.

a mobility trace while enforcing a low spatial distortion. Our goal is to specifically take into account the threats that POIs may pose, while making it easy to configure it and obtain a fair trade-off with utility. Moreover, as a follow-up to (2), Chapter 5 will propose a solution to help users configuring their LPPMs in an efficient and friendly way. Because some configurations behave better than other when considering the privacy/utility trade-off, our goal is to automatically generate those configurations, without the need for users to understand how their LPPM works behind the scenes.

CHAPTER 4

PROMESSE: Protecting Points of Interest

4.1 Introduction

In this chapter, we are interested in the offline use cases, in which researchers and industrials are working with entire mobility datasets (cf. Section 1.1). Indeed, publishing fine-grained datasets allows analysts to implement data mining tasks with the tools and languages they want and run them on the published data. Moreover, there are use cases where such datasets are required. For example, researchers working on delay-tolerant networks test their algorithms with real-life datasets (e.g., [71]). Another example is the case of transportation mode detection. In this context, state-of-the-art algorithms need to extract a number of information from mobility traces such as speed, acceleration [161] or proximity to rail lines/bus stops [142], which is not possible using alternative solutions such as interactive querying, where data analysts are restricted to a pre-defined query language provided by the data owner (e.g., [117]), or the publication of pre-aggregated datasets (e.g., [5]). However, such data is highly sensitive because of the many attacks that can be ran (cf. Section 2.3). Of course, we cannot afford to publish datasets that would allow an attacker to infer sensitive information that could be threatening users' private life.

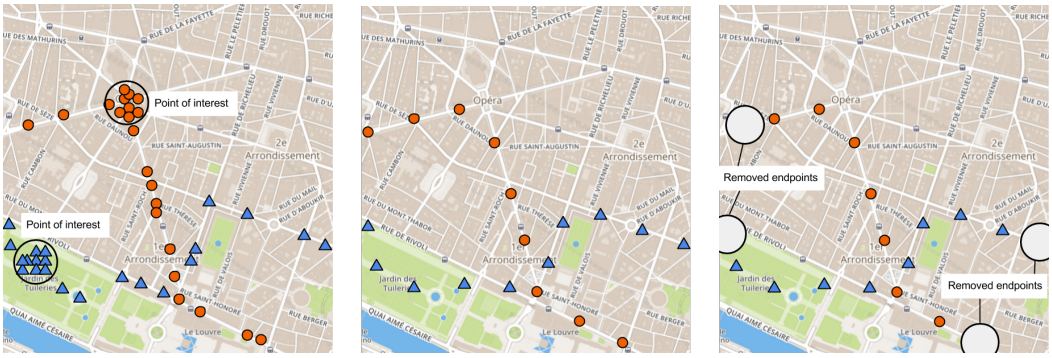
To address this issue, many LPPMs have been proposed (cf. Section 2.5). A classical solution that has been implemented is to alter locations in some way (e.g., [1, 9]), thus hiding the exact place where a user went and improving his privacy. However, according to the amount of added noise, this may also alter the utility of the published data, as highlighted in the previous chapter (cf. Section 3.6). Furthermore, the exemple of Geo-I [9] (always in Section 3.6) showed that such mechanisms may not be efficient at protecting POIs of users. We could not find a good configuration of Geo-I offering a good balance between privacy and utility.

To fulfill our objectives in terms of privacy and utility, we propose an alternative solution that leverages time distortion (i.e., altering the temporal component of events) instead of spatial distortion, as usually done in state-of-the-art LPPMs. In this chapter, we investigate this alternative and introduce our contribution, PROMESSE, which is the first LPPM aiming at hiding users' POIs by distorting time. Specifically, PROMESSE hides users' POIs by: (1) smoothing the users' speed along their trajectories and (2) removing the start and end points of these trajectories to make them less easily identifiable. We practically study the effectiveness of PROMESSE compared to two representative mechanisms relying on spatial distortion, namely *Wait for Me* [2], which enforces k-anonymity, and Geo-I [9], which guarantees differential privacy. Our evaluation, performed using three real-life datasets, shows that the number of retrieved POIs with PROMESSE is under 3 %, which is comparable to what the other mechanisms can achieve. In the same time, PROMESSE provides no spatial error (i.e., locations are not distorted in the protected dataset), while the other mechanisms' error ranges from 24 to 70,000 meters.

The remaining of this chapter is organized as follows. In Section 4.2, we give the intuition of how PROMESSE works, before presenting PROMESSE's algorithm in Section 4.3. We experimentally evaluate our solution in Section 4.4 and conclude this chapter in Section 4.5.

Figure 4.1: Overview of PROMESSE.

(a) Actual dataset. (b) After speed smoothing. (c) After removing endpoints.



In a nutshell. Our original contributions (related to contribution **C2**) in this chapter are the following:

- An algorithm leveraging speed smoothing to hide POIs;
- An experimental evaluation of this algorithm against two representative state-of-the-art competitors (Geo-I [9] and $\mathcal{W}4\mathcal{M}$ [2]) and three real-life datasets (Geolife, Cabspotting and MDC).

Associated publications: [63, 126, 128].

4.2 Overview

Our objective in terms of privacy protection is to hide users' POIs. These correspond to places where users stop and spend some time, before starting again to move to another place (cf. Section 2.2.5). Every trace can be viewed as a list of POIs that appear, when visualizing traces, as clusters of locations (as shown on Figure 4.1a), with transitions in between. Our counter-measure to hide POIs is thus to enforce a constant speed in the whole trace of a user, i.e., with *speed smoothing*. If we can guarantee that the speed is constant throughout the trace, it becomes difficult for an adversary to spot where a user stopped because there is no point at which he appears to be stationary. Clues can still be obtained from background knowledge (e.g. the probability is higher to stop in a park that in the middle of a highway) but there will be no certainty for an attacker (e.g. a user can either have just crossed a park or had a picnic there). Moreover, we guarantee that there is a constant duration and distance between two successive events in a trace. This prevents an attacker from inferring information by studying spatio-temporal intervals at which traces have been sampled.

Figure 4.1b shows the result of speed smoothing applied to two mobility traces. From this figure, we can see that the POIs of the users have been removed and that events on each trace are regularly spaced. However, after this step, two events still remain unchanged in each trajectory: the first and the last ones. Because they are likely to be POIs (e.g., a home), they need to be protected too. Our solution is to remove endpoints to reduce the precision around them and hence to protect users' privacy around these places. Figure 4.1c shows the effect of removing endpoints.

4.3 Promesse: A utility-preserving protection mechanism for hiding POIs

In this section, we present the core algorithm behind Promesse in Section 4.3.1, before having a discussion about its parametrization in Section 4.3.2.

4.3.1 Algorithm

We now present an implementation of our speed smoothing LPPM, i.e., Promesse, depicted in Algorithm 2. The speed smoothing algorithm works on mobility traces and proceeds in two steps, by first working on locations (lines 4-20) and then working on timestamps (lines 22-29). It is parametrized by a single parameter, α, which is a distance expressed in the same unit than the result of $d_{\mathcal{X}}$. The first step of this algorithm is hence to extract regularly spaced locations, each being at a distance α from the previous one (lines 4-16). The larger α, the better the privacy guarantee, but the higher the quantity of information lost (because we are missing actual locations). To perform this sampling, locations are interpolated along segments joining known locations. This means that our method is more suited for traces with a high sampling rate (e.g., ten to thirty seconds between consecutive events). If the sampling rate is too low, the quality of the interpolation will be very degraded, because the algorithm will have to make up entirely fake locations. After this, we remove the first and last events from the list (lines 17-18), in order to help hiding endpoints that would otherwise be easily guessed. More precisely, because we have events spaced by α, it means that we reduce the precision by α around these endpoints. If after this step there is not enough remaining locations left to recreate a valid trace (i.e., two or less), we simply discard the trace (lines 19-20). The second step is then to assign to each of the previously sampled locations a timestamp by uniformly allocating the duration of the actual trace (lines 22-29). This is where the temporal distortion happens, because time is entirely re-allocated among sampled locations.

Finally, we define the Interpolate function. As its name suggests, the goal of this function is to interpolate a new point along a line between two points. Because its implementation actually depends on the way location data is represented (cf. Section 2.2), we provide it as a formal definition.

Definition 15. *The interpolation between two locations $(a, b) \in \mathcal{L} \times \mathcal{L}$ of a factor $\psi \in]0, 1]$ is defined as the unique location on the segment $[ab]$ at a distance $\psi \times d_{\mathcal{X}}(a, b)$ of a:*

$$Interpolate_{\psi}(a, b) = c \in \mathcal{L} \,|\, (d_{\mathcal{X}}(a, c) = \psi d_{\mathcal{X}}(a, b) \wedge d_{\mathcal{X}}(c, b) = (1 - \psi) d_{\mathcal{X}}(a, b))$$

Please note that, depending on the representation of locations, implementing correctly this function may be difficult, because of numerical instabilities and limited precision of floating point numbers in most programming languages.

The implementation of the full Promesse protection mechanism is shown in lines 1-2, where the speed smoothing algorithm is independently applied to each mobility trace. As every LPPM, our algorithm takes as input a dataset to protect, and requires

a single real-valued α parameter. From an implementation point of view, each trace can be protected independently from the others. In other words, we have an embarrassingly parallel problem: traces can be protected in parallel and then merged at the end. It allows us to protect large datasets in a very efficient manner, as we will outline it in the next section.

Algorithm 2 PROMESSE implementation.

Data: $\alpha \in \mathbb{R}^{+*}$ ▷ PROMESSE configuration parameter

function PROMESSE($d \in \mathcal{D}$)
  return $\bigcup_{u \in \overrightarrow{\mathbf{user}}(d)}$ SPEEDSMOOTHING(u, d_u, α)

function SPEEDSMOOTHING($u \in \mathcal{U}$, $t \in \mathcal{D}_u$, $\alpha \in \mathbb{R}^{+*}$)
  $sampled \leftarrow \emptyset$ ▷ Spatially sampled events
  $prev \leftarrow null$ ▷ Previous location in the trace
  for $e \in t$ **do**
    if $prev = null$ **then** ▷ First iteration
      $sampled \leftarrow sampled \cup \{e\}$
      $prev \leftarrow \mathbf{loc}(e)$
    else ▷ Second iteration and next
      $d \leftarrow d_{\mathcal{X}}(\mathbf{loc}(e), prev)$
      while $d \geq \alpha$ **do** ▷ Interpolate between previous and current locations
        $\ell \leftarrow Interpolate_{\alpha/d}(prev, \mathbf{loc}(e))$
        $sampled \leftarrow sampled \cup \{\langle \mathbf{user}(e), \ell, \mathbf{time}(e) \rangle\}$
        $prev \leftarrow \ell$
        $d \leftarrow d_{\mathcal{X}}(\mathbf{loc}(e), prev)$
  $sampled \leftarrow sampled \setminus \{e \in sampled \mid \mathbf{time}(e) = \min(\overrightarrow{\mathbf{time}}(e)\}$
  $sampled \leftarrow sampled \setminus \{e \in sampled \mid \mathbf{time}(e) = \max(\overrightarrow{\mathbf{time}}(e)\}$
  if $|sampled| \leq 2$ **then**
    return $\varnothing$ ▷ Non protectable trace, return an empty trace

  $t_{min} \leftarrow \min \overrightarrow{\mathbf{time}}(sampled)$
  $t_{max} \leftarrow \max \overrightarrow{\mathbf{time}}(sampled)$
  $\delta t \leftarrow (t_{max} - t_{min})/(|sampled| - 1)$ ▷ Duration between consecutive events
  $t_{curr} \leftarrow t_{min}$
  $events \leftarrow \varnothing$ ▷ Events in the resulting protected trace
  for $e \in sampled$ **do**
    $events \leftarrow events \cup \{\langle \mathbf{user}(e), \mathbf{loc}(e), t_{curr} \rangle\}$
    $t_{curr} \leftarrow t_{curr} + \delta t$
  return $events$

4.3.2 Parameters setting

From the above algorithm it appears that a variable that might have an important impact on the results of PROMESSE is α. This value should be chosen according to the granularity of POIs that the data owner wants to hide. Indeed, we can consider areas as large as entire cities as POIs, or increase the granularity and consider small neighborhoods or even individual buildings as POIs. This granularity, fixed in practice

by the $\Delta\ell$ parameter of our POIs extraction algorithm (cf. Section 3.2.1), depends on what kind of information the attacker is interested in extracting, or what kind of information the data owner wants to protect. The α parameter of PROMESSE is directly related to the diameter[1] of POIs that will be hidden. Intuitively, a value of α will tend to hide POIs that would be extracted with $\Delta\ell \leq \alpha$. A large value of α hence hides large POIs, while a small value of α hides small POIs. However, choosing an α too large also greatly reduces the expected utility of the protected dataset, because of the induced loss of precision (events will be at an α distance of the previous one). This means α has a great impact on the enforced privacy and provided utility and must be chosen carefully, while keeping in mind its meaning.

This parametrization question is not unique to our LPPM. Indeed, ϵ-differentially-private or k-anonymous LPPMs must also be parametrized according to the level of privacy to achieve. We discuss this question thoroughly and propose an innovative solution to help with LPPM parametrization in Chapter 5.

4.4 Experimental results

We start by describing in Section 4.4.1 our experimental settings. Then, we describe the evaluation of PROMESSE in terms of privacy (Section 4.4.2) utility (Section 4.4.3) and performance (Section 4.4.4). Eventually, we put in perspective the results in terms of both privacy and utility in Section 4.4.5.

4.4.1 Experimental setup

Datasets

We study PROMESSE using three real-life datasets: Cabspotting, Geolife and MDC (cf. Section 3.5). We pre-process our datasets to remove entire days with no data. Then we only kept the first 20 days of data, to have a dataset with a similar duration for traces of all users. Those steps were done to enforce the homogeneity on the three datasets, and allow a fair comparison.

We further performed another type of pre-processing on the three datasets. We divided each trace into individual parts, each one being a set of events with no temporal gap between two consecutive events. Specifically, a trace is divided into two parts when no event is logged during four consecutive hours. Each part is then considered as an independent trace associated with a new virtual user identifier, no matter to which logical user it really belongs. This pre-processing helps to preserve privacy, as it breaks the correlation between multiple journeys of a same logical user. We applied this pre-processing for all studied mechanisms to allow a fair comparison.

Parametrization

We test PROMESSE with various values of the α parameter: 50 (only for Geolife and MDC), 100, 200 and 500 meters. Indeed, $\alpha = 50$ meters was impracticable with Cab-

[1] We remind that the diameter of a POI refers to the diameter of the circular area where all the events related to this POI fall.

spotting because of too much data being generated (cf. the data compression evaluation in Section 4.4.3 for more on this).

We compare PROMESSE with two representative state-of-the-art LPPMs. The first one is Geo-I (introduced in [9], and presented Section 2.5.4), which is one of the latest approaches offering differential privacy guarantees to the users. We configure Geo-I with various values of ϵ that are similar to the ones authors of the paper consider in their publications [9, 22]. The lower ϵ, the more noise is added and the stronger the privacy guarantee. We acknowledge that Geo-I is not exactly designed for the publication of entire datasets, because of the sequential composition theorem (cf. Section 2.4.1)[2], but it is a representative mechanism that adds noise to locations to protect them.

The other LPPM we consider is *Wait for Me* (introduced in [2], and presented in Section 2.5.2), which enforces k-anonymity (later abbreviated $\mathcal{W}4\mathcal{M}$). We configure $\mathcal{W}4\mathcal{M}$ to use the LSTD distance, described in their paper, that is shown to perform better with large datasets. Further, we configure the mechanism with the following parameters: $\delta = 200$ meters, $k = 2$, $Max\_Trash = 10$ % of the dataset's size and $max\_radius = 5000$ meters. This means that at any time, any two traces of the protected dataset are in a cylinder with diameter of a 200 meters. Other parameters are default ones suggested by the authors of this paper. We only study one configuration of $\mathcal{W}4\mathcal{M}$ because $k = 2$ is the minimum value (results are worse when k increases [2]) and $\delta = 200m$ puts it in a similar situation than PROMESSE, in addition to being consistent with the value of $\Delta\ell$ we choose (cf. Section 4.4.2).

Implementation

PROMESSE, Geo-I and the evaluation metrics are implemented on the Java Virtual Machine in Scala. We use the implementation of $\mathcal{W}4\mathcal{M}$ as provided by their authors [3]. We ran our experiments on a single Debian virtual machine having access to 8 Gb of RAM and 8 cores clocked at 1.8 GHz each. The prototype will be presented in more details later in Chapter 6.

4.4.2 Privacy evaluation

We evaluate the privacy effectively guaranteed to users by running the POIs retrieval metric (cf. Section 3.2.2, Definition 6) on datasets protected by PROMESSE, Geo-I and $\mathcal{W}4\mathcal{M}$. We used a POIs maximum diameter of $\Delta\ell = 200$ meters, a POIs minimum duration of $\Delta t = 15$ minutes, and a POIs comparison threshold of $\sigma = \Delta\ell/2 = 100$ meters.

We report on the average POIs retrieval across all traces in each dataset in Table 4.1. From this table, we can see that in the Cabspotting dataset, POIs are always hidden with PROMESSE, no matter the value of α. With Geolife and MDC, we closely reach our goal with $\alpha = 200m$. This value of α was indeed expected to be its optimal parametrization, given that with this value we have $\alpha = \Delta\ell$. Still, the POIs retrieval is close but not equal to zero, most likely due to some edge cases. We remember that the

[2]For example, protecting each record of a one million events dataset with $\epsilon = 0.001$ results at the end in a global ϵ for the whole dataset equal to $0.001 \times 10^6 = 1000$, which is a very relaxed theoretical guarantee to say the least.

Table 4.1: POIs retrieval evaluation of PROMESSE (lower is better).

Protection mechanism	**Cabspotting**	**Geolife**	**MDC**
PROMESSE, $\alpha = 50m$	-	17 %	14 %
PROMESSE, $\alpha = 100m$	0 %	11 %	8 %
PROMESSE, $\alpha = 200m$	0 %	2 %	1 %
PROMESSE, $\alpha = 500m$	0 %	0 %	0 %
Geo-I, $\epsilon = ln(10)/100$	31 %	42 %	56 %
Geo-I, $\epsilon = ln(6)/200$	6 %	9 %	16 %
Geo-I, $\epsilon = ln(4)/200$	3 %	5 %	9 %
Geo-I, $\epsilon = ln(2)/200$	1 %	0.9 %	0.5 %
$\mathcal{W}4\mathcal{M}$, $k = 2$, $\delta = 200m$	0 %	0 %	0 %

POIs retrieval metric is computed as an F-Score; it is worth noting that we went deeper in our analysis and that behind this low F-Score we actually have both a low precision and a low recall. This means that very few POIs are retrieved, and that they are lost inside many false positives. Furthermore, we observe that $\mathcal{W}4\mathcal{M}$ hides all POIs in the three datasets, but this is a consequence of the great quantity of noise that has to be added to enforce k-anonymity. Here the privacy comes at the cost of a very degraded utility, as it will be shown in Section 4.4.3. Finally, retrieving POIs from the Geo-I protected datasets is very dependent on the quantity of noise that has been added. With lower values of ϵ, and hence more privacy, retrieving POIs becomes very challenging. With $\epsilon = ln(2)/200$, almost no POI is found, as with PROMESSE. Weaker values of ϵ allow many POIs to be retrieved, up to a POIs retrieval of 56 %.

4.4.3 Utility evaluation

We evaluate the utility of the protected datasets with three metrics: spatial distortion, count query distortion and compression degree.

Spatial distortion

We first use the spatial distortion metric in its projected version (cf. Section 3.3.1, Definition 9). It is indeed perfectly suited for our PROMESSE LPPM, because it was designed with the goal of guaranteeing a null distortion under that metric. If we do not take into account the error due to the numerical imprecision of the projections/interpolations, by construction, our LPPM is in measure to guarantee no spatial distortion.

We report about the average spatial error for all events in each dataset in Table 4.2. From this table, we observe that the spatial error of PROMESSE is equal to zero for the three datasets. Indeed, by construction, the only inaccuracy introduced by PROMESSE is due to the interpolation between sampled events, which shows to be negligible in this experiment. Geo-I instead adds noise to locations, depending on its ϵ parameter, which results in an average error ranging from 24 to 378 meters on the three datasets. This has to be compared with the average error due to GPS measurements which is about 5 to 15 meters. This means that at its weakest level of privacy, Geo-I is just a little bit less

Table 4.2: Spatial error evaluation of PROMESSE (lower is better).

Protection mechanism	**Cabspotting**	**Geolife**	**MDC**
PROMESSE, $\forall\alpha$	0 m	0 m	0 m
Geo-I, $\epsilon = ln(10)/100$	24 m	45 m	52 m
Geo-I, $\epsilon = ln(6)/200$	50 m	120 m	140 m
Geo-I, $\epsilon = ln(4)/200$	62 m	156 m	183 m
Geo-I, $\epsilon = ln(2)/200$	113 m	325 m	378 m
$\mathcal{W}4\mathcal{M}$, $k = 2$, $\delta = 200m$	13,046 m	69,676 m	19,222 m

precise than the error that could come from the normal usage of a GPS. However, when the level of privacy increases, the error can go as high as 378 meters, which is enough to disturb data mining tasks, especially in a dense urban environment. Finally, among the three tested mechanisms, $\mathcal{W}4\mathcal{M}$ is the one with the worst spatial error, which is at least equal to 13,046 meters in our experiments. This is due to the large amount of noise $\mathcal{W}4\mathcal{M}$ introduces to enforce k-anonymity. These results highlight the benefit of a time distortion LPPM for use cases where a high spatial accuracy is needed, which cannot be achieved with the other mechanisms building on spatial distortion.

Count query distortion

We use the count query distortion metric (cf. Section 3.3.3, Definition 12) to evaluate further the utility. Similarly to authors of $\mathcal{W}4\mathcal{M}$ [2], we choose time windows ranging from 2 hours to 8 hours and squared areas whose half-diagonals range from 500 to 5,000 meters.

Table 4.3: Count query distortion evaluation of PROMESSE (lower is better).

Protection mechanism	**Cabspotting**	**Geolife**	**MDC**
PROMESSE, $\alpha = 50m$	-	15 %	25 %
PROMESSE, $\alpha = 100m$	7 %	15 %	25 %
PROMESSE, $\alpha = 200m$	6 %	15 %	27 %
PROMESSE, $\alpha = 500m$	7 %	19 %	31 %
Geo-I, $\epsilon = ln(10)/100$	0.7 %	8 %	3 %
Geo-I, $\epsilon = ln(6)/200$	2 %	20 %	10 %
Geo-I, $\epsilon = ln(4)/200$	3 %	27 %	13 %
Geo-I, $\epsilon = ln(2)/200$	7 %	60 %	30 %
$\mathcal{W}4\mathcal{M}$, $k = 2$, $\delta = 200m$	102 %	102 %	94 %

We report about the average query distortion in Table 4.3, which is the average distortion over 1,000 randomly generated count queries. Results show that PROMESSE has a query distortion ranging from 6 % to 27 % for $\alpha = 200m$. This means that results of count queries have, on average, a relative error of at most 27 %. Further, results show that we perform at least 71 % better than $\mathcal{W}4\mathcal{M}$ with all the three datasets. Once again, the distortion with Geo-I is dependent on the value of ϵ. The weakest value features almost no distortion (but also does not protect POIs in a satisfactory way, cf.

previous section). Intermediary values of ϵ correspond to similar distortions than with the optimal PROMESSE. We notice that the count query distortion metric is more sensitive with Geolife and MDC because counts are way smaller than with Cabspotting, and therefore the effect of missing one user is more important on the relative error. This explains the larger difference of distortion between Geo-I and PROMESSE on the Geolife and MDC datasets.

Compression degree

We evaluate the compression degree that LPPMs provide (cf. Section 3.3.2, Definition 10).

Table 4.4: Compression degree evaluation of PROMESSE.

Protection mechanism	**Cabspotting**	**Geolife**	**MDC**
PROMESSE, $\alpha = 50m$	-	196 %	49 %
PROMESSE, $\alpha = 100m$	27 %	400 %	99 %
PROMESSE, $\alpha = 200m$	56 %	833 %	208 %
PROMESSE, $\alpha = 500m$	156 %	2500 %	555 %
Geo-I, $\forall\epsilon$	100 %		
$\mathcal{W}4\mathcal{M}$, $k = 2$, $\delta = 200m$	94 %	132 %	101 %

Experimental results for the compression are shown in Table 4.4. They highlight that because Cabspotting has a coarser sampling rate than Geolife, for small values of α the compression degree is very low (down to 27 %), which means that the dataset protected with PROMESSE is up to 369 % larger than the actual one. Conversely the Geolife dataset protected with PROMESSE is much smaller than the actual one, the latter having been collected with a very high sampling rate (average sampling rate is 7 seconds). $\mathcal{W}4\mathcal{M}$ and Geo-I both have almost no effect on the size of the produced dataset. This metric is interesting because it shows that for PROMESSE, some values of α are impracticable, resulting in too huge datasets (this is why we did not experiment with Cabspotting at $\alpha = 50m$). But it also shows that it is possible to reduce the size of a dataset with a high sampling rate without losing "too much" information (cf. the other utility metrics).

4.4.4 Performance evaluation

We evaluate the performance of PROMESSE with the wall time metric.

Wall time

We measure the wall time (cf. Section 3.4.1). We report about execution times in Table 4.5. Because execution times remain almost constant for the various configurations of Geo-I and PROMESSE, we do not report about subtle variations depending on parameters. Geo-I and PROMESSE are the fastest mechanisms because their algorithms are quite simple. Geo-I independently adds noise to each event and PROMESSE independently protects each trace, which enables it to be efficient in terms of computational

complexity. $\mathcal{W}4\mathcal{M}$ is the slowest mechanism because of the complexity coming from the clustering of similar traces that is the heart of this LPPM.

Table 4.5: Wall time evaluation of PROMESSE (lower is better).

Protection mechanism	**Cabspotting**	**Geolife**	**MDC**
PROMESSE	210 s	25 s	15 s
Geo-I	147 s	64 s	21 s
$\mathcal{W}4\mathcal{M}$	605 min	70 min	37 min

4.4.5 Discussion

From the results presented in this section, we conclude that time distortion is a promising alternative to spatial distortion for the privacy-preserving publication of mobility datasets. Indeed, on the three datasets we studied, our proposed PROMESSE mechanism parametrized with $\alpha = 200m$ hides almost all users' POIs, while keeping the spatial accuracy very high. Temporal distortion has though an impact on metrics for which time is important, e.g., count queries. Nevertheless, PROMESSE still offers a distortion varying from 6 % to 27 % according to the sparsity of dataset, which is comparable to Geo-I and way better than $\mathcal{W}4\mathcal{M}$. Furthermore, PROMESSE is simpler to parametrize because there is only one parameter α to set, whose meaning is clear: it represents the granularity of POIs to protect. Obviously, among the many use cases that data analysts may want to implement, there shall be some that require a high temporal accuracy, which PROMESSE cannot provide. In this case, Geo-I may still be a better candidate. Our goal in this chapter was to introduce another way to protect mobility data, that takes the opposite direction of actual state-of-the-art protection mechanisms and to practically study its effectiveness.

4.5 Summary

In this chapter we presented PROMESSE, a new offline LPPM to protect mobility datasets. Its novelty resides in the fact that it distorts timestamps instead of distorting locations, which allows it to have a better utility than representative state of the art mechanisms. We compared it to two state-of-the-art LPPMs, Geo-I, which provides differentially privacy, and $\mathcal{W}4\mathcal{M}$, which provides k-anonymity. Privacy evaluation showed that, when configured appropriately, PROMESSE resists POIs retrieval attacks similarly to Geo-I. $\mathcal{W}4\mathcal{M}$, still performs better, but at the cost of a very decreased utility. Finally, PROMESSE is fast, as it can protect a dataset of 9 million records in less than four minutes. From our study, we conclude that time distortion is a promising alternative to existing spatial ones, particularly for use cases where a high spatial accuracy is required.

As in Chapter 3, we outlined in this chapter that an LPPM is only as good as its configuration. Indeed, we found out that α should be chosen according to the diameter of POIs that the users wants to hide. However, if LPPM designers are aware of this fact and can produce optimal configurations, thanks to their knowledge of how their LPPM works, it is not always the case for final users. In the next chapter, we propose

a solution to tackle this problem and help ordinary users to configure their LPPMs efficiently.

CHAPTER 5

ALP: Configuring Protection Mechanisms

5.1 Introduction

To address the challenge of location privacy, many LPPMs have already been proposed. However, the effectiveness of these solutions largely rely on the tuning of a set of configuration parameters (possibly with a large range of possible values), which is a difficult task for non-expert users or data owners as these parameters have both an impact on the privacy offered to the users and on the utility of the protected data. As an example, $\mathcal{W}4\mathcal{M}$ takes at least five parameters, with some labeled as the "initial maximum radius used in clustering" or the "global maximum trash size" [2]. While there are useful to precisely tune the behavior of the algorithm, we do not expect final users to read the paper to understand what the trash is or how the clustering works. Even the single ϵ parameter of Geo-I is tricky to configure, because it is expressed in meters^{-1} and its impact is exponential. Similarly, it is difficult for a final user who knows (usually) nothing about differential privacy to set it appropriately. Moreover, most of the time these parameters are statically set up once and for all, and do not dynamically evolve according to the content of the data under analysis, especially in online use cases. Such a static LPPM parametrization may however lead to the over-protection of non sensitive data portions (e.g., a portion of the data without any POI) thus uselessly degrading its utility, and to the under protection of possibly sensitive data portions (e.g., the regular visit of a hospital), thus resulting in the leakage of sensitive information about the user.

A few adaptive LPPMs [6, 23] and LPPMs focusing on user experience [21, 41] have been presented in the literature. However, these works put the emphasis on privacy guarantees offered to users, but rarely put the utility of the resulting data on the same level. Consequently, utility is only provided on a best-effort basis.

In this chapter, we present ALP (which stands for Adaptive Location Privacy), a new framework for dynamically configuring LPPMs, that considers both privacy and utility as equally important objectives. Specifically, ALP contains a generic model enabling the specification of a set of privacy and utility *objectives* that the LPPM shall satisfy. Then, instead of testing static configuration parameters for each LPPM, ALP uses an *optimizer* that dynamically tunes the parameters of the LPPM under consideration according to the current data portion to which it is applied on in order to meet the privacy and utility objectives specified by the system designer. The generality of ALP allows its deployment either in offline uses cases, for comparing and tuning a set of LPPMs with the purpose of protecting a static dataset before releasing it, and in batch use cases, in the context of a crowd sensing application for dynamically configuring a given LPPM with respect to the given data portion under analysis. In both cases, the major contribution of ALP is its ability to automatically find LPPM configurations that fulfill a set of possibly conflicting privacy and utility objectives that it would be cumbersome to find manually otherwise. Unfortunately, due to its architecture, ALP is not suitable for real-time use cases.

We illustrate the capabilities of ALP by comparing two state-of-the-art LPPMs, i.e., Geo-I [9], which applies spatial distortion to the mobility data and PROMESSE (described in Chapter 4), which applies temporal distortion to the mobility data, on two real-life mobility datasets. We show in an offline use case that ALP eases the comparison of these LPPMs by relying on a set of metrics provided by the framework.

We further show in a batch use case that ALP is able to dynamically find configurations of these LPPMs that outperform a set representative static configurations of used LPPMs. For instance, we show that ALP is able to tune Geo-I on a per-trace basis enabling to perfectly hide POIs for at least 75 % of the traces while having a spatial distortion lower than 150 meters on the two datasets. The results for PROMESSE are even better as ALP is able to find for each trace a configuration of the LPPM enabling to globally outperform all the representative static configurations, both on the considered privacy and utility metrics, thus reaching the best of the two worlds. To assess the performance of ALP on mobile devices, we measured the execution time on an emulated smartphone. Results show that the execution time on a single batch of data is highly dependent on the LPPM under consideration, with an average execution time of 9 seconds with Geo-I and 500 milliseconds with PROMESSE.

The remaining of this chapter is structured as follows. We first review the related works in Section 5.2. We present an overview of ALP in Section 5.3, before going into the details of the algorithms in Section 5.4. Finally, we present our experimental evaluation in Section 5.5 and conclude this chapter in Section 5.6.

In a nutshell. Our original contributions (related to contribution **C3**) in this chapter are the following:

- A method to convert user-centric privacy/utility objectives into a set of parameters, applicable to different LPPMs;
- A framework leveraging this method to produce adaptive parametrizations.

Associated publication: [129].

5.2 Related work

Because we already presented a thorough state of the art about LPPMs in Section 2.5, we only give in this section complementary information about the specific problem we are interested in this chapter, i.e., the adaptive configuration of LPPMs.

The only works interested in user experience are rules-based LPPMs *ipShield* [21] and *LP-Guardian* [41] (already described in Section 2.5.5). These two LPPMs have been successfully implemented on Android and effectively used to protect users. Both papers put a strong focus on usability, for example by evaluating the energy overhead occurred by using their solution. Despite relying on state-of-the-art solutions to protect privacy (*LP-Guardian* notably integrates Geo-I), they do not require users to configure esoteric parameters. This is because they rely on user-defined rules that in turn smartly configure the underlying algorithms. *ipShield* also feature the notion of user-defined objective, where users define the relative importance of attacks against which they want to be protected. However, these two solutions put the emphasis on privacy and do not give users explicit control over the expected utility.

Furthermore, few initiatives have been proposed to dynamically tune the LPPMs according to the underlying data. We highlight here two such LPPMs (already described in Section 2.5). Chatzikokolakis et al. [23] proposed an extension of Geo-I, which leverages contextual information (i.e., if the user is located in an urban environment or a

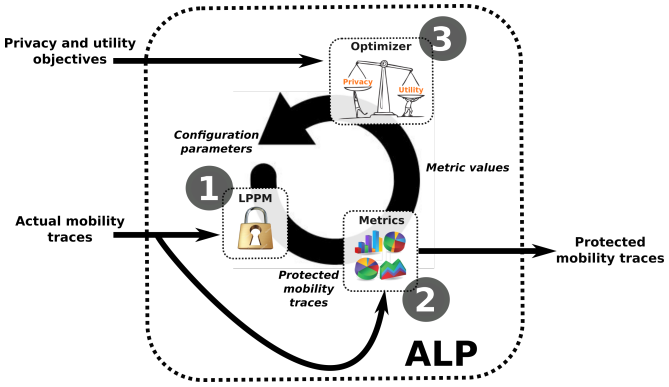


Figure 5.1: Components forming the ALP framework

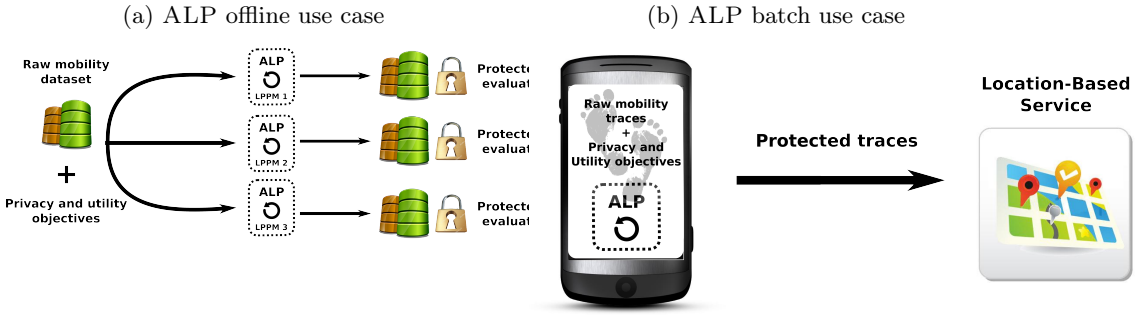
countryside area) to calibrate the amount of noise applied to disturb the mobility traces. This makes the actual noise level to be adaptive, depending on the context. Agir et al. [6], in turn, introduced an adaptive mechanism to dynamically change the size of an obfuscated area hiding the exact location of users. More precisely, the proposed solution locally evaluates the privacy level and enlarges the cloaking area accordingly until a targeted privacy level is reached. Again, this solution adapts the perturbation level to the actual location being protected. However, these two approaches are designed with a single privacy goal in mind and do not give utility the same level of importance.

5.3 Overview

We present in this section an overview of ALP, a framework for the dynamic configuration of LPPMs. As depicted in Figure 5.1, ALP takes as input actual mobility traces and outputs protected mobility traces . These traces can contain multiple days/-months of data (in offline use cases), or can be small batches containing only a few minutes/hours of data (in batch use cases). However, contrary to existing LPPMs, ALP does so by also considering a set of privacy and utility objectives specified by the user (in batch use cases) or the data owner (in offline use cases). To protect actual traces, ALP proceeds as follows. First, the actual traces get protected using a given LPPM applied with an initial (random) configuration (step 1 in the figure). The protected traces are then evaluated with respect to the specified privacy and utility objectives (step 2 in the figure). Then an optimization process uses the result of this evaluation to iteratively propose a better configuration for the LPPM (step 3 in the figure). In ALP, this optimization process, which is further presented in Section 5.4, is based on the simulated annealing algorithm [78]. This step outputs new values for the LPPM configuration parameters, which are re-used in another round of step 1. The three steps are repeated until a satisfactory configuration is found.

As depicted in Figure 5.2, ALP can be used in two major use cases. First, in offline use cases (Figure 5.2a), a data owner wants to protect a dataset of mobility traces before releasing it. Towards this purpose, he uses ALP to automatically tune different LPPMs according to a set of privacy and utility objectives he would like to achieve. As a result, the data owner gets the result of a set of evaluation metrics for each configured LPPM, which allows him to decide which corresponding protected dataset to release. Second, in batch use cases (Figure 5.2b), a user periodically sends protected data to an LBS

Figure 5.2: ALP in action: the offline protection of a complete dataset before releasing it (left) and the batch configuration of an LPPM for individual users periodically interacting with an LBS (right).



(typically a crowd-sensing application). In this scenario, ALP is deployed on the mobile device of a user to protect his mobility data. To achieve that, ALP dynamically tunes an LPPM according both to a set of privacy and utility objectives set by the user and to the current data under analysis. In both scenarios, the key feature of ALP is its ability to dynamically optimize an LPPM with respect to a set of privacy and utility objectives.

Available LPPMs were already presented in Section 2.5 (step 1), available privacy metrics in Section 3.2 and available utility metrics in Section 3.3 (step 2). Note that we do not consider performance metrics here, as they are not part of the privacy/utility trade-off, and because their evaluation is too strongly impacted by external factors such as the computational resources at disposal. What remains to explain is the optimizer (step 3), which is detailed in Section 5.4.

5.4 Optimizing with simulated annealing

By combining metrics with an optimizer, ALP is able to tune LPPMs to achieve a set of privacy and utility objectives. More precisely, the optimizer receives as input the values of privacy and utility metrics associated with the current parameters proposal, and automatically tunes these parameters of the LPPM for the next try. In other words, its goal is to solve an optimization problem with the objective to maximize/minimize metrics' values. Optimizing a mathematical function is a subject that has already been well-studied in the literature. Many methods exist such as *hill climbing* or *gradient descent* [141]. In this section, we propose a practical solution relying on the simulated annealing algorithm. Indeed, besides choosing an appropriate optimization algorithm, the challenge lies in correctly instantiating this algorithm, notably with a correct function to optimize (i.e., user-specified objectives have to be translated into a real-valued function). The approach we present here should be adaptable for other optimization algorithms.

This section starts by presenting how the user defines its objectives (Section 5.4.1). We then provides a background on the simulated annealing algorithm (Section 5.4.2) followed by the various adaptations necessary for using this algorithm in the context of ALP, i.e., the definition of a cost function, acceptance probability function, the randomization of the explored space and the cooling schedule described in Sections 5.4.3, 5.4.4, 5.4.5 and 5.4.6, respectively.

Algorithm 3 Simulated annealing algorithm.

```
1:  function SimulatedAnnealing(t_0 ∈ ℝ^+, t_min ∈ ℝ^+, δt ∈ ℝ^+*)
2:      s ← Initial()
3:      c ← Cost(s)
4:      t ← t_0
5:      while t ≥ t_min do
6:          s' ← Neighbor(s)
7:          c' ← Cost(s')
8:          ap ← Probability(c, c', t)
9:          if ap ≥ Random(0, 1) then
10:             s ← s'
11:             c ← c'
12:         t ← t × δt
13:     return s
```

5.4.1 Objectives

Objectives are set by the user and control the expected outcome of a parametrization in terms of privacy and utility. Normally, the user sets at least two conflicting objectives, i.e., a privacy objective and a utility objective, though he may set more. ALP relies on the library of privacy (resp. utility) evaluation metrics we presented in Section 3.2 (resp. Section 3.3). Two kinds of objectives are supported: maximizing or minimizing a metric. Practically, the user chooses a metric and whether it has to be maximized or minimized. Of course, most metrics have to be parametrized themselves. To alleviate the burden of this task and avoid a chicken-and-egg problem, we propose to use pre-defined values for those.

More formally, we define an objective as a triplet $\langle dir, m, f \rangle \in \mathcal{O}$, where $dir \in \mathbb{B}$ is a boolean indicating whether the metric should be minimized (true) or maximized (false), $m \in \mathcal{M}$ is a metric and $f \in \mathbb{R}^{+*}$ is a scaling factor (described below, it is not set manually by the user but rather pre-defined per metric). Let $\mathbf{minimize} : \mathcal{O} \longrightarrow \mathbb{B}$, $\mathbf{metric} : \mathcal{O} \longrightarrow \mathbb{M}$ and $\mathbf{scale} : \mathcal{O} \longrightarrow \mathbb{R}^{+*}$ be functions to access attributes of an objective, i.e., $\forall o = \langle dir, m, f \rangle \in \mathcal{O}, \mathbf{minimize}(o) = dir, \mathbf{metric}(o) = m, \mathbf{scale}(o) = f$.

5.4.2 Simulated annealing

Simulated annealing [78] is a well-known probabilistic optimization technique useful to find an approximation of the global optimum of a function. Finding the exact global optimum is not guaranteed, but this optimization technique ensures an acceptable local optimum in a reasonable amount of time compared to a brute-force method exploring all possible solutions. It is especially useful for large (or infinite) search spaces. It follows the physical analogy of cooling down a metal, where the temperature is gradually decreasing until the state is frozen. If the cooling takes enough time, atoms can find an optimal placement, i.e., a state associated with minimal energy. The algorithm is depicted in Algorithm 3. The underlying idea is, from an initial state $s \in \mathcal{S}$ (line 2), to probabilistically decide whether to move to a neighbor state s' (lines 9-11) depending on the current temperature and the cost associated with these states (line 8). This cost corresponds to the energy of a state in the physical analogy. This process is re-

Algorithm 4 ALP cost function.

Data: $O \in \mathcal{P}(\mathcal{O})$ a set of objectives
Data: $\pi \in \Pi$ the LPPM being configured
Data: $t \in \mathcal{D}_u$ the actual trace of user $u \in \mathcal{U}$

```
1:  function Cost(s ∈ S)
2:      t̂ ← π_s(t)                          ▷ Protected trace, w.r.t., current state
3:      c ← 0                                ▷ Total cost of all objectives
4:      for o ∈ O do
5:          v ← average(metric(o)(t̂, t))     ▷ Raw metric value
6:          v' ← min(|v|, scale(o))/scale(o) ▷ Rescaled metric value
7:          if minimize(o) then
8:              c ← c + v'
9:          else
10:             c ← c + (1 − v')
11:     return c
```

peated several times, with a decreasing temperature until the system reaches a minimal temperature (line 5).

As shown in the algorithm, a simulated annealing system needs several functions to be defined: an initial state function, producing an initial state $s \in \mathcal{S}$ (line 2); a neighbor function $\mathcal{S} \to \mathcal{S}$ associating each state to a neighboring state (line 6); a cost function $\mathcal{S} \to \mathbb{R}$ associating a cost to each state (lines 3 and 7); an acceptance probability function $\mathbb{R} \times \mathbb{R} \times \mathbb{R}^+ \to [0, 1]$ giving the probability to accept the new solution given the cost of the current solution, the cost of the new solution and the current temperature (line 8); a cooling schedule, controlling the values taken by the temperature (lines 4, 5 and 12). These functions must be defined according to the particular usage that is being done of the simulated annealing algorithm. We propose implementations for them in the next sections.

In our context, a state corresponds to a set of values for all parameters considered for a given LPPM. Indeed, each LPPM is configured with a set of parameters, which can be either real-valued or categorical. Not all of them have to be considered when optimizing to reduce the exploration space; for example some parameters may be almost constant or determined as a function of others. Each parameter has an associated domain, which fixes the discrete or continuous set of values that can be taken by values of this parameter. For example, PROMESSE has a single α parameter, which is real and restricted to strictly positive numbers, i.e., $\alpha \in \mathbb{R}^{+*}$.

5.4.3 Cost function

A challenge is to convert a set of objectives into a cost (i.e., a single real number) in such a way that the higher the cost, the worst the solution. Each objective contributes to a part of the cost. Our cost function is depicted in Algorithm 4. The cost is computed on a per-objective basis, each one being evaluated separately and then aggregated. Because our metrics produce a vector of real values (cf. Section 2.2.4), we first aggregate this vector into a single real by taking the average (line 5). As evaluation metrics can be defined in very different ranges (e.g., a distance will be expressed in meters and take

values in $\mathbb{R}^+$, whereas a percentage is restricted to $[0, 1]$), we normalize them in a value belonging to $[0, 1]$ in order to give to each metric a similar weight (line 6). To achieve that, we impose to each metric a maximum value which bounds the associated cost, and we scale the metric value accordingly: it is our scaling factor.

For now, we only support metrics working at the trace level, i.e., taking an actual and a protected trace and producing a real-valued quantification. We also only support maximizing or minimizing a metric. Supporting the whole diverse set of metrics presented in Chapter 3, as well as comparison operators (e.g., having a metric less than some value) remains future work.

5.4.4 Acceptance probability function

Let $c \in \mathbb{R}$ be the cost of the old solution, $c' \in \mathbb{R}$ be the cost of the new solution, $t \in \mathbb{R}^+$ be the current annealing temperature, $O \in \mathcal{P}(\mathcal{O})$ be the current set of objectives. We define the acceptance probability function as:

$$Probability(c, c', t) = \begin{cases} 1 & \text{if } c < c' \\ 1/(1 + e^{\frac{c'-c}{0.5 \times t \times |O|}}) & \text{otherwise} \end{cases}$$

It states that the probability to accept a solution with a higher cost decreases with the temperature, although we always accept a solution with a smaller cost. The $0.5 \times |O|$ expression is a normalization factor, that takes into account varying numbers of objectives. This is a standard form of the acceptance probability function, that is used for example by Matlab [98].

5.4.5 Randomizing solutions

Another challenge of simulated annealing is the way to explore the space of solutions. In ALP, solutions (or states) are configuration parameters for the considered LPPM. Each LPPM can be parametrized by several parameters, defined in different ranges of values, possibly infinite. For example, a k-anonymous LPPM should at least have a $k \in \mathbb{R}^+$ parameter defining the level of anonymity, or a basic LPPM randomly dropping events should have a probability $p \in [0, 1]$ to keep each event. In our framework, we consider parameters as having a domain formed of a (possibly infinite) set of possible values (e.g., $[0, 1]$ or $\mathbb{R}^+$).

The INITIAL function is used to provide the initial state, i.e., the initial value for each parameter of the LPPM being configured. We do this by defining randomly the value of each parameter, i.e., by picking it from its domain of definition. The NEIGHBOR function is used to compute the next state to explore, with respect to the actual one. More precisely, the new state corresponds to the previous state with a single parameter, randomly chosen, being changed. This parameter is modified by restricting its domain by half and shifted to be centered around the previous value (excluded). For example, if the domain of definition of a parameter is $\{1, 2, 3, 4, 5\}$ and its current value is 2, the domain when choosing the new value will be restricted to $\{1, 3\}$.

5.4.6 Cooling schedule

Finally, a cooling schedule determines the size of the parameter space effectively explored, and affects the acceptance probability. In ALP, we choose a simple solution which is a static cooling schedule, where temperatures range from $t_0 = 1$ to $t_{min} = 10^{-5}$, with a cooling rate of $\delta t = 0.9$. Consequently, 110 solutions are explored each time the algorithm is ran.

5.5 Experimental results

This section starts with the presentation of the experimental setup of our evaluation (Section 5.5.1). We then illustrate the capabilities of our framework by evaluating the optimization of two state-of-the-art LPPMs under two different use cases: (1) an offline use case where ALP helps a data owner to tune and compare the two LPPMs on a whole dataset (Section 5.5.2); (2) a batch use case where ALP is used by mobile users to fine tune a given LPPM on batches of geo-located data before sending them to an LBS (Sections 5.5.3 and 5.5.4). We finally evaluate the latency of running ALP in a mobile device (Section 5.5.5).

In a nutshell, our evaluation draws the following conclusions: first, in the offline use case, the generality of ALP eases the tuning and comparison of state-of-the-art LPPMs. Further, in the batch use case, ALP allows to find LPPM configurations reaching trade-offs between privacy and utility metrics that outperform representative static configurations of the latter. Finally, the latency of running ALP on a mobile device is reasonable and highly depends on the underlying LPPM.

5.5.1 Experimental setup

Datasets

We use two real-life datasets to evaluate ALP: Geolife and MDC (cf. Section 3.5).

Parametrization

Geo-I [9] takes an ϵ parameter (expressed in meters^{-1}) determining the amount of noise to add (the smaller ϵ, the higher the amount of noise added to the actual data). In ALP, ϵ has been configured to take values in $[0.001, 0.1]$. Moreover, we use a logarithmic space (in base 10) to draw values for ϵ, because the smallest its value is, the more impact it has on privacy (and therefore utility). To compare our adaptive solution with statically configured mechanisms, we take as baselines $\epsilon \in \{0.001, 0.01, 0.1\}$; 0.001 and 0.1 and the extreme values that are considered by ALP and 0.01 gives us a logarithmic progression. We set as objectives for the optimizer to minimize the POIs retrieval (privacy metric, described in Section 3.2.2) and to minimize the spatial distortion (utility metric, described in Section 3.3.1). We configure the POIs retrieval metric to extract POIs with a maximum diameter $\Delta\ell = 200$ meters and a minimum stay time $\Delta t = 15$ minutes. We use a threshold $\sigma = \Delta\ell/2 = 100$ meters to determine whether POIs are correctly retrieved. Because Geo-I is a non-deterministic LPPM, each metric is evaluated three times, and we consider the median value as the final metric value.

PROMESSE (cf. Chapter 4) takes an α parameter (expressed in meters) specifying the distance to enforce between two consecutive locations (the larger α, the higher the spatial distortion of the actual trace). In ALP, α takes values in $[0, 500]$ (meters), compared to an $\alpha \in \{100, 200, 300, 500\}$ for the static baselines. Baselines allow to explore different values regularly spaced, including 500 meters, the maximum α considered by ALP and 200 meters, that should be the globally optimal value, according to Section 4.4.5. Similarly to Geo-I, we also set as objectives for the optimizer to minimize the POIs retrieval with the same setting, but we set to maximize the area coverage[1] (utility metric, described in Section 3.3.4). For the latter, we consider cells at the 15th level, areas at this level typically covering a few blocks inside a city.

Implementation

ALP is implemented on the Java Virtual Machine in Scala. It is mainly split in two parts. The first one is a library of common data structures to represent and manipulate mobility data and implementation of state-of-the-art protection mechanisms. The second part is the glue assembling pieces together and creating the framework. ALP includes a configuration layer, an optimizer and an execution engine scheduling and running the different operations. ALP is designed to be extensible and allows researchers as well as practitioners to easily implement their own LPPMs and metrics. It has been published as open source and is thus publicly available [125], though it has largely been superseded by ACCIO for most aspects (detailed later in Chapter 6).

5.5.2 Offline: LPPM comparison

We evaluate an offline use case by using ALP to optimize both Geo-I and PROMESSE in order to protect the Geolife dataset. In this use case, the data owner configures ALP to provide a single value of ϵ and α (for Geo-I and PROMESSE, respectively) for each user, and to evaluate these LPPMs through three metrics: the POIs retrieval, the spatial distortion and the area coverage. Nevertheless, our framework also allows the data owner to perform pre-processing on the dataset, for instance to split it into smaller data portions and to choose to tune the LPPM configuration for each data portion.

Figure 5.3 reports the Cumulative Distribution Function (CDF) of the POIs retrieval, the spatial distortion and the area coverage for both LPPMs. For all these metrics, the configuration found by ALP for PROMESSE provides better results than the configuration found by ALP for Geo-I. Indeed, more than 95 % of users using PROMESSE have a POI retrieval of 0 (i.e., all of their POIs are hidden), a median spatial distortion of 25 meters (respectively 75 meters for Geo-I) and a median area coverage of 0.75 (respectively 0.55 for Geo-I). Ultimately, the decision is left to the data owner to select which of the resulting protected dataset he would use. ALP only provides all the necessary material to easily evaluate LPPMs according to privacy and utility objectives.

5.5.3 Batch: Privacy and utility trade-off

We now illustrate a batch use case. We consider a crowd-sensing application that collects the user location every 30 seconds through his mobile device, and sends this data once

[1] We did not consider the spatial distortion because it is always null or almost null with PROMESSE.

Figure 5.3: Cumulative distribution of privacy & utility metrics with Geolife in the offline use case.

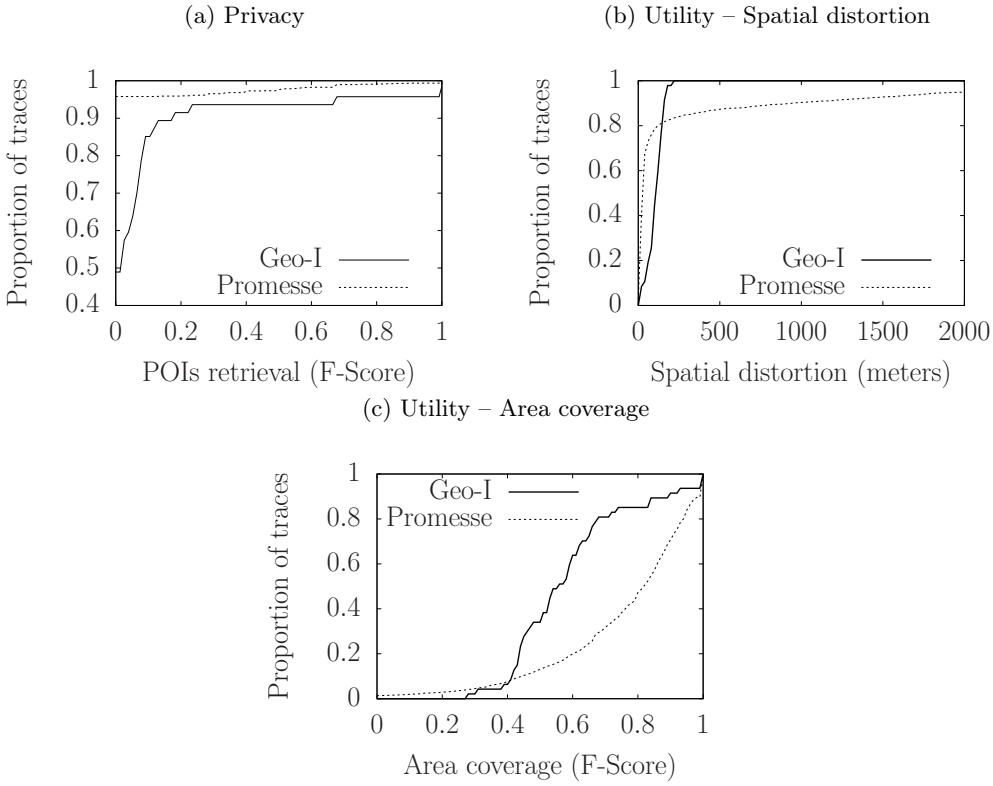


Figure 5.4: Cumulative distribution of privacy & utility metrics under Geo-I in the batch use case.

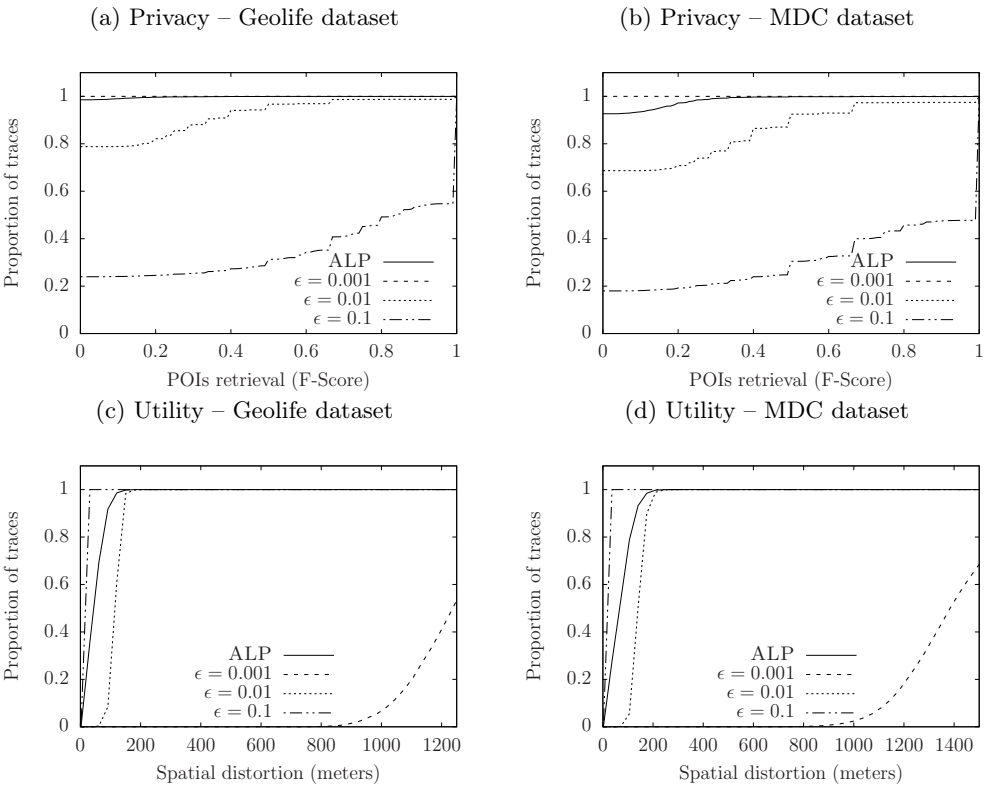
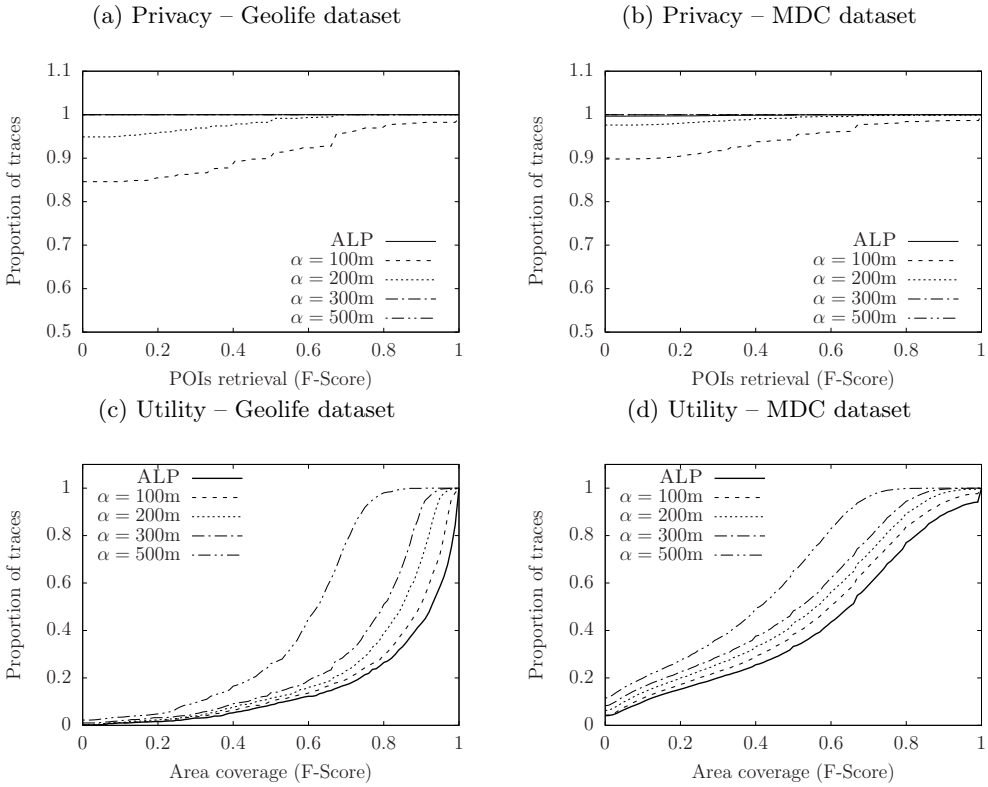


Figure 5.5: Cumulative distribution of privacy & utility metrics under PROMESSE in the batch use case.

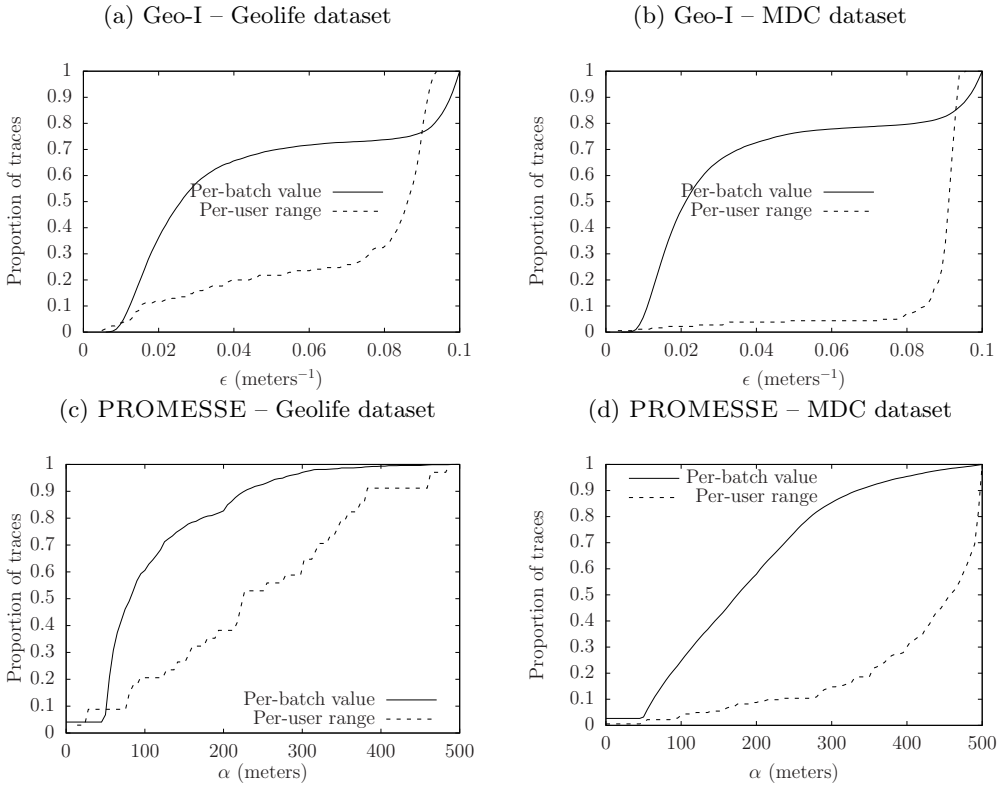


a day to an LBS.

Figure 5.4 reports for Geo-I and for the two considered datasets the CDF of the privacy and the utility objective metrics (i.e., POIs retrieval and spatial distortion, respectively) for both the dynamic configuration of ϵ found by ALP and several static configurations of ϵ. We show that ALP hides all POIs of at least 92 % of users (i.e., a null POIs retrieval) for both datasets (Figures 5.4a-5.4b) while maintaining a median spatial distortion of 40 and 70 meters with Geolife and MDC, respectively (Figures 5.4a-5.4b). Note that some static ϵ configurations outperform our dynamic solution either on privacy or on utility (e.g., the one with the lowest value of ϵ is better for hiding POIs but has a worse spatial distortion and the one with the highest value of ϵ has the opposite behavior). Nevertheless, there is no static configuration that outperforms the dynamic configurations found by ALP both on privacy and on utility. This means that the trade-off between privacy and utility provided by ALP is better than one found by the static baselines as the latter adjusts the amount of noise according to the underlying data to protect.

Figure 5.5, in turn, depicts for PROMESSE using the Geolife and MDC datasets, the CDF of the privacy and the utility objective metrics (i.e., POIs retrieval and area coverage, respectively) for both the dynamic configuration of ALP and static baselines. We show that the dynamic α configuration of ALP offers a nearly perfect protection with a null POIs retrieval for almost all users and on both datasets (Figures 5.5a-5.5b), while offering a better utility (i.e., the smaller area coverage) than the various static α configurations (Figures 5.5a-5.5b).

Figure 5.6: Cumulative distribution function of the value taken by ϵ and α for Geo-I and PROMESSE, respectively.



In the case of PROMESSE, these results show that ALP is able to provide the best of the two worlds by outperforming static configurations both on privacy and utility.

5.5.4 Adaptive configuration

We now focus our evaluation on the analysis of the adaptive capabilities of ALP. Specifically, we analyze the variation of the LPPM parametrization according to the evolution of the input batch under analysis. Figure 5.6 shows for both Geo-I and PROMESSE on the two considered datasets the CDFs of the different values of ϵ or α generated by ALP for each batch, and the range of parameter values taken for each user (i.e., max - min).

Interesting enough, results for Geo-I (Figures 5.6a-5.6b) show that 65 % with Geolife (respectively 72 % with MDC) of the chosen per-batch values for ϵ are smaller than 0.04, and 27 % (respectively 20 %) are greater than 0.09. Values of ϵ between 0.04 and 0.08 are rarely chosen by our algorithm, which could indicate that either a batch needs to be strongly protected or almost not. If we consider the range of ϵ values taken per-user, results show that for 77 % of users with Geolife (respectively 93 % with MDC) the range of values is greater than 0.08 (out of a maximum of 0.1). This large range indicates that ALP chooses very different values of ϵ for each user during their mobility activity. This variability across batches of a single user highlights the dynamic optimization that ALP performs to adapt the configuration parameter of the protection mechanism according to the data portion under analysis.

Results for PROMESSE (Figures 5.6c-5.6d) exhibit a different behaviour. The different values chosen by α per-batch are almost chosen uniformly distributed across the range of possible values, with a median value of 80 and 170 meters with Geolife and MDC, respectively. The range of α values taken for each user also reports a uniform distribution for Geolife. However, the per-user range for MDC exhibits a different distribution where 70 % of users have a range greater than 400 meters (out of 500 meters). For both datasets, the large range chosen for α supports once again the necessity to adapt configuration parameters of LPPMs according to the current mobility data.

5.5.5 Deployment on mobile devices

Finally, we evaluate the cost of running ALP on a mobile device. More precisely, we measure the wall time (cf. Section 3.4.1) taken by a mobile device to perform the optimization of the configuration parameters of an LPPM. This latency must be limited to avoid the device to be frozen while the optimizer is running. To achieve this measurement, we constrained this particular experiment to run on a single core, clocked to 1.2 GHz, and with 1 Go of RAM. The time taken by ALP to find a parametrization in this case is on average of 9 seconds with Geo-I and 500ms with PROMESSE. We remind that ALP is designed for offline and batch use cases, hence as a process running periodically. These values seem very reasonable when protecting data, for example, every hour. However, we acknowledge that this execution time is still non-negligible, and too high to consider using ALP in a real-time use case.

We found that the rate at which we collect records has a non-negligible impact on the performance. For instance, if we collect a record every 5 minutes (instead of 30 seconds in the current experiments), the execution time with Geo-I is on average of 7 seconds (22 % less) due to a smaller size of the batch of data to be processed.

5.6 Summary

In this chapter, we presented ALP, a solution for adaptive location privacy configuration of LPPMs. ALP makes the parametrization and the evaluation of LPPMs easier by shifting the process of protecting location privacy from a parameter-centric paradigm where users or data owners have to set obscure parameters, to an objective-centric paradigm where users only have to define their target privacy and utility objectives. Using these objectives, ALP automatically tunes the set of LPPM configuration parameters according to the data under analysis, which allows adding the right amount of noise and avoids unnecessarily degrading the quality of the data or under protecting sensitive data portions. We illustrated the capabilities of our framework through the optimization of two state-of-the-art LPPMs on two use case scenarii and with two real-life datasets. We showed that ALP enabled to find dynamic LPPM configurations that outperform representative static configurations, thus reaching the best of both worlds in terms of privacy and utility.

As future work, we wish to better define and enhance the user experience, with a prototype mobile application and a help in setting objectives without too much burden. We are also eager to experiment with other optimization algorithms, besides simulated annealing. For example, *Vizier* [57] is a black-box optimization framework, which means

it makes minimal assumptions on the system under consideration. It is an interface to well-known bayesian optimization algorithms [140], largely used in the machine learning community to help choosing parameters of their algorithms. Finally, we would want our optimizer to be able to automatically suggest the right LPPM to use, and then configure it as we proposed here. In the next chapter, we lift the veil on Accio, which is the software tool allowing to reproduce experiments that were executed in this chapter and the previous ones.

CHAPTER 6

ACCIO: Experimenting with Location Privacy

6.1 Introduction

In the past decade, researchers have been highly active on proposing LPPMs. The latter are evaluated by metrics in terms of privacy, utility and performance. As outlined in Table 2.2 (online LPPMs) and Table 2.3 (offline LPPMs), there is a large heterogeneity in metrics used to evaluate LPPMs, making rather difficult to fairly compare them. Practically, LPPMs are usually evaluated with monolithic code designed only towards this purpose, and furthermore not always made available by their authors. In a 2015 study [31], researchers attempted to reproduce results of 601 papers across 13 top computer science conferences. They considered 402 eligible papers and were able to obtain the code for 226 papers (56 %). Finally, they were able to successfully build the code for 48 % of these papers, which represents 27 % of the total amount of considered papers. They characterized this approach as "weak reproducibility", because they were only interested in building and running the code and not actually validating the results.

To deal with the difficulty of evaluating LPPMs, few works have been proposed in the literature. For instance, the Location Privacy Meter [139] is a framework designed to quantify location privacy. However, this framework has a strict underlying probabilistic model that does not accommodate the large variety of LPPMs and metrics that exist in the literature. Another work is GEPETO [48], a toolkit whose goal is to visualize the impact of LPPMs and attacks in a graphical user interface. However this tool only focuses on re-identification attacks and does not allow easily the automation of experiments, because of its UI-centric approach.

The challenge we attempt to solve is three-fold. First, we need to improve the *reproducibility* of research results, which cannot be achieved if source code is not available and if the evaluation methodology is not defined precisely enough. Second, we need to be able to *compare* LPPMs, to help choosing the right LPPM for the right task. Third, we need to *drive experimentation* by facilitating the creation, monitoring and exploitation of experiments. As a solution, we propose in this chapter Accio, a location privacy experimentation platform enabling researchers to quickly design and launch location privacy experiments. Indeed, Accio comes with a standard library of reusable operators. An operator in Accio can be a mobility data manipulation method, an LPPM implementation or an evaluation metric. At the time of writing, 25 such operators are implemented in Accio. Experiments are then executed using the Accio runtime, which can transparently adapt task deployment to the available computing resources, from a deployment on a single machine to a distributed deployment on a cluster of machines or a custom cloud infrastructure. Finally, our framework allows to quickly analyze results (via a Web interface) as well as to export results (in CSV) for a later analysis with custom tools (e.g., Excel, Matlab, Python). We evaluate our solution by demonstrating its effective usability to evaluate three very different state-of-the-art LPPMs (Geo-I [9], $\mathcal{W}4\mathcal{M}$ [2] and Promesse), various metrics and datasets. We show all of those algorithms can be unified under concepts provided by our framework, and expressed elegantly with a few lines of JSON.

The remaining of this chapter is organized as follows. We first review related works in Section 6.2. We then provide an overview of our solution in Section 6.3 before detailing our platform in Section 6.4. Section 6.5 provides an evaluation under the form of a case study showing how to use Accio to solve applied research questions. We finally

conclude this chapter in Section 6.6.

In a nutshell. Our original contributions (related to contribution **C4**) in this chapter are the following:

- An extensible experiment platform, and its client applications command-line and Web interfaces.
- A standard library of operators for spatio-temporal data manipulation, LPPMs and metrics.
- An implementation publicly released as an open-source tool.

6.2 Related work

Because we already presented a thorough state of the art about LPPMs in Section 2.5, we only give in this section complementary information about the specific problem we are interested in this chapter, i.e., experimenting and evaluating LPPMs.

Location privacy frameworks. Shokri et al. proposed a fully-fledged framework designed to evaluate location privacy [139], providing both a formalization of the problem and an implementation as a tool. The privacy offered by an LPPM is quantified by comparing the outcome of a privacy attack performed on an actual trace and on its protected counterpart. The whole evaluation process is divided in five steps: reading data, simulating an application, applying an LPPM, executing an attack and evaluating its efficiency with a metric. Each step can be replicated to compare different datasets, LPPMs, attacks or metrics. Towards this purpose, they propose several new attacks and formally define three metrics: accuracy, certainty and correctness. They actually implemented their framework as a tool and released it under an open source licence [93]. However, this solution only works for probabilistic LPPMs and is not adapted to more generic mechanisms (e.g., $\mathcal{W}4\mathcal{M}$ or PROMESSE). Furthermore, it only considers privacy when evaluating an LPPM, which means utility and the trade-off between privacy and utility is not considered.

GEPETO [48] is a tool for location privacy study proposed by Gambs et al. It allows to apply several LPPMs on mobility datasets, and launch privacy attacks. It focuses on visualization by providing a graphical user interface to display on a map results of algorithms. Because it did not scale to large datasets, they proposed a way to port two clustering algorithms as MapReduce tasks [50]. However, this is still preliminary work, and much more algorithms would need to be implemented to have a complete framework. Moreover, they do not give any detail about the programmatic API behind their work, which makes difficult to automate experiments.

Scientific workflow tools. Although our goal is to support location privacy research and not to provide a generic workflow management tool, there are some similarities between our work and such systems. Scientific workflow tools are used to model experiments with workflows, launch them on distributed architectures (e.g., a grid or a cloud), and provide access to results. They are especially used in disciplines such as bioinformatics and astronomy. Pegasus [36] reads workflows from XML files, in addition to providing programmatic APIs for generating these files. It also comes with a Web

interface to monitor and debug executions. Swift [152] provides a language roughly similar to C to describe computations. It is then compiled and automatically parallelized when possible. Kepler [8] comes with a desktop application to create and execute workflows. It allows to visually connect operations and see how they interact. These tools are more generic that location privacy study. They usually come with a set of operations targeted towards astronomy or chemistry, and not for spatio-temporal datasets. The survey of Liu et al. [90] gives and extensive view about scientific workflow management systems.

6.3 Overview

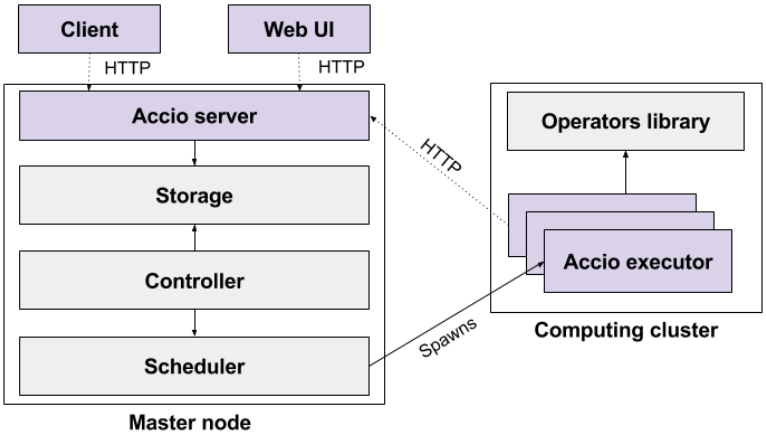


Figure 6.1: High-level architecture of Accio.

In this section, we give an overview of Accio, a framework designed to ease the evaluation of LPPMs. Accio is not yet another LPPM, but rather an open and extensible platform to evaluate and compare LPPMs, that encompasses state-of-the-art LPPMs and can be extended with new LPPMs. Accio comes with a library embedding basic spatio-temporal data manipulation operators as well as more advanced location privacy algorithms such as state-of-the-art LPPMs and evaluation metrics that we described in previous chapters. Operators are the most basic building block, and are defined as black-box functions producing some outputs given some inputs. The Accio operators library is open and designed to be easily extensible with new operators. Researchers then describe their location privacy experiments via a composition of operators, and submit them to Accio. These experiments, as well as all other entities (e.g., operators) are represented as plain JSON or YAML.

Figure 6.1 presents the overall architecture of Accio. Purple boxes correspond to software binaries belonging to Accio, while grey boxes correspond to internal components embedded inside these binaries. The central part of Accio is its server. It receives the requests coming both from users and other components, and responds accordingly. The only interface exposed to the outside world is a REST API that allows to interact with the various entities (e.g., operators, experiments) that Accio supports. The server can be decomposed in three layers:

1. The *storage layer* is in charge of retrieving or persisting entities in response to REST calls;

2. The *controller layer* is listening for changes in the storage layer and takes appropriate actions;
3. The *scheduler* is called by the controller to actually launch and monitor operators.

For example, if someone submits an experiment to the Accio server, it will go all along those three layers. First, it will be persisted by the storage layer into a persistent storage (e.g., a relational database or a key/value store). Second, this new experiment will be picked up by the controller layer and broken up into tasks. Indeed, experiments are not executed as a whole but rather at the operator level, by generating one task per operator in the experiment. Third the controller will call the scheduler to actually start the execution of those tasks and monitor them.

The executor is a binary in charge of actually executing a single task. An executor is dynamically spawned for each task to execute, and terminates once it is done with this task. Each task runs inside a sandboxed environment, isolated from the other running tasks. Executors communicate their progress and results back to the Accio server through its REST API. Because our goal in Accio is not to create yet another scheduling system (this has already been largely studied in the literature, and it is not our domain of interest in this thesis), we chose to rely on existing scheduler to do the actual scheduling, e.g., Mesos [67]. As a result, we only provide an interface to this scheduler, which is doing the actual work of finding a machine with enough resources to process the task, launch the executor, collect logs, etc.

Finally, Accio comes with two user interfaces, a command-line client that is mainly used to create experiments and get results back, and a Web interface that provides a read-only view on running experiments and allows to preview their results. Both interfaces communicate with the server via its REST API.

6.4 Accio architecture

Accio is a system formed of several components, detailed in Figure 6.1. It is made of almost 20,000 lines of code written mainly in Scala (a language running in the Java Virtual Machine) for backend services, and about 3,000 additional lines of Javascript for the Web interface. We detail in this section our platform and its implementation. We first introduce the obligatory concept of operator in Section 6.4.1. We then detail the lifecycle of an experiment: describing it in JSON Section 6.4.2, its storage in Section 6.4.3, decomposition in tasks in Section 6.4.4, and scheduling in Section 6.4.5. Furthermore, we explain how an experiment is monitored and its results exploited in Section 6.4.6. Finally, we present how Accio can be extended with new operators in Section 6.4.7.

6.4.1 Operators

An *operator* is the basic building block of Accio. It acts as a function in a program: given some inputs, it produces some outputs. Each operator comes with a very clearly defined interface: it defines the inputs it consumes and the outputs it produces, using a type system provided by Accio. Because input and outputs are strongly typed, values are checked for correctness before actually executing operators. Outputs generated by

the execution of an operator are automatically collected and ingested back into Accio. Operators are identified by a name, which is unique and used to reference them later in workflows. They need to be implemented by developers, but they can later be used even by non-developers thanks to our JSON syntax used to describe experiments (cf. Section 6.4.2). Operators are stateless, though they can access external storages such as databases or filesystems. Operators are the basic execution unit and are always executed on a single machine. They define resource constraints about the number of CPU cores, the quantity of RAM and disk space they need to execute properly.

Operators are assumed to be deterministic. It means that given some inputs, they should produce the exact same outputs at each execution. However, we support injecting some randomness through *unstable operators*. Unstable operators are given access to an initial seed, which can be randomly generated or manually provided when launching an experiment. It means that given a set of inputs and a seed, unstable operators are expected to produce the exact same outputs at each execution.

6.4.2 Describing experiments

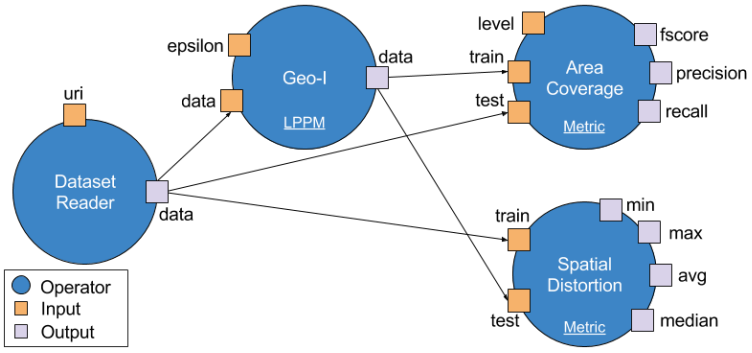


Figure 6.2: Example of a simple workflow with four nodes.

Experiments are described with two concepts: *workflows* and *runs*.

A workflow is a directed acyclic graph, whose nodes are instances of operators. Example workflow depicted in Figure 6.2 is formed of four nodes, each with its own inputs (in orange) and outputs (in purple). The `DatasetReader` node is the root node, meaning it has no input coming from another node. It accepts one input, `uri`, and produces one output, `data`. The goal of this operator is rather simple: read a dataset stored somewhere (e.g., local disk or Amazon S3), possibly in various formats, and convert it into a standardized format that other operators will understand. The latter output is then consumed as an input by node `Geo-I`, as well as by nodes `AreaCoverage` and `SpatialDistortion`. It becomes clear that some inputs are filled from the output of another node (e.g, the `data` input of `Geo-I`), while some other are directly specified through a constant value (e.g, the `epsilon` input of `Geo-I`).

When specifying a workflow, one essentially defines a list of nodes and how to connect them. Each node is an instance of a given operator, has a name (which by default is the operator's name) and some inputs. It specifies its inputs, either directly with a constant value or by connecting it to the output port of another node. Lastly, inputs can also be filled by workflow parameters, which are values specified by the user when launching a

workflow. They allow him to vary the value of one or several inputs depending on this parameter, as long as they have of the same type. This is how we allow re-usability of workflows: the same parametrized graph can be launched multiple times with different parameters combinations.

A run corresponds to one or several instantiation of a workflow, where all parameters are defined with a single value. Workflows can be seen as templates for creating runs, while a run is a single execution of a workflow. While operators need to be implemented by developers, workflows and runs are represented in JSON. A run can as well represent a single execution of a workflow, or a large parameter sweep involving tens or hundreds different combinations of parameters being tested.

To create an experiment, one has to first create a workflow and then a run associated with this workflow. The workflow only has to be created once. Workflow depicted in Figure 6.2 would be created with the JSON description featured in Listing 6.1 (YAML can also be used). We first provide metadata about the workflow (lines 2-4), then the parameters the workflow accepts (lines 5-15), and finally its graph of operators describing the execution flow (lines 16-45).

Listing 6.1: Description of our simple workflow in JSON.

```
1  {
2    "id": "geoind-workflow",
3    "name": "Geo-indistinguishability workflow",
4    "owner": "vprimault",
5    "params": [
6      {
7        "name": "epsilon",
8        "kind": "double",
9        "default_value": 0.01
10     },
11     {
12       "name": "uri",
13       "kind": "string"
14     }
15   ],
16   "graph": [
17     {
18       "op": "DatasetReader",
19       "inputs": {
20         "uri": {"param": "uri"}
21       }
22     },
23     {
24       "op": "Geo-I",
25       "inputs": {
26         "epsilon": {"param": "epsilon"},
27         "data": {"reference": "DatasetReader/data"}
28       }
29     },
30     {
31       "op": "AreaCoverage",
32       "inputs": {
33         "level": {"value": 15},
34         "train": {"reference": "DatasetReader/data"},
35         "test": {"reference": "Geo-I/data"}
36       }
37     },
```

```
38      {
39        "op": "SpatialDistortion",
40        "inputs": {
41          "train": {"reference": "DatasetReader/data"},
42          "test": {"reference": "Geo-I/data"}
43        }
44      }
45    ]
46  }
```

Then, our workflow can be instantiated several times through runs, that are much shorter to describe: they only need to specify the workflow to execute and values for its parameters. An example of a run description in JSON is provided in Listing 6.2. The `uri` parameter receives a single value, while the `epsilon` parameter receives four different values. It means that the entire workflow will actually be executed four times, once for each combination of values.

Listing 6.2: Description of a run in JSON.

```
1  {
2    "workflow": "geoind-workflow",
3    "params": {
4      "epsilon": {"values": [0.0001, 0.001, 0.01, 0.1]},
5      "uri": {"value": "/path/to/my/dataset"}
6    }
7  }
```

6.4.3 Persisting entities

Accio supports various entities, among which operators, workflows and runs. These entities are exposed to the outside via a REST API, allowing to perform generic actions on them (e.g., create, delete, list) as well as more specific actions (e.g., abort a run). The storage layer is in charge of handling incoming HTTP requests, validating them and interacting with a persistent storage to either retrieve or store entities. After an entity has been persisted or deleted, an event is propagated on an internal event bus allowing the controller to react appropriately. By design, we preferred orchestration over composition in the server.

The storage layer supports different backends, from relational databases to key/value stores. For now, our storage of choice is Zookeeper [72], a highly available and consistent key/value store.

6.4.4 Generating tasks

Every time an object is persisted through the REST API, the controller is informed and can react accordingly. Most of the time it has nothing to do (e.g., a workflow is a simple descriptive object), but sometimes its role is more crucial. Once a run has been created and persisted, it is then handled by the controller that will split it into a list of tasks, where each task corresponds to the execution of a single operator. If the workflow is seen as the logical execution plan, the set of tasks corresponds to the physical

execution plan. All tasks are created at once when a workflow is submitted; it means that tasks cannot be dynamically instantiated. For example, for the run described in Listing 6.2, the workflow will give birth to four tasks (one for each operator) and will be executed four times (once for each combination of parameters' values), for a total of 16 generated tasks. Tasks are assigned to a single physical machine on which they are executed, according to the resources requested by their operator. Indeed, each operator specifies how much CPU cores, amount of RAM and amount of disk it needs. These resources represent at the same time a request (i.e., specified resources are guaranteed to be available when the operator is executed) and a limit (i.e., specified resources cannot be exceeded, otherwise the task will fail).

A task can be in several different stages, obeying to a state machine depicted in Figure 6.3. Initially `WAITING`, a task enters the `SCHEDULED` state once all its dependent tasks have successfully completed. In the latter state, it is waiting for computational resources to be available. Then it enters the `RUNNING` state, where its execution actually begins. Once completed, a task can be either `SUCCESSFUL` or `FAILED`, if an exception was raised. Because of machine failures or communication problems, it can happen that a task gets `LOST`. This is a special failure status indicating that Accio lost contact with the executor of a running task and that it can be retried later, by contrast to the ordinary failed state that rather indicates a problem in the operator's implementation. Finally, a task may be `ABORTED`, either at the user's request or if a dependent task failed.

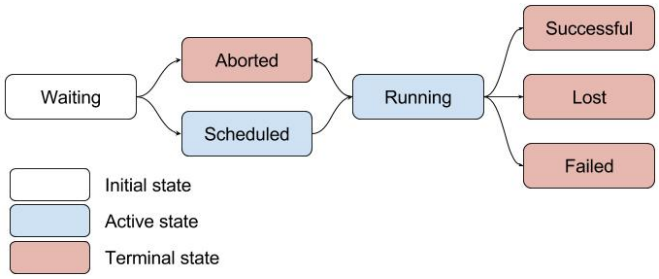


Figure 6.3: Task state machine.

6.4.5 Scheduling

Once a run has been transformed into a set of tasks, it is up to the scheduler to launch these tasks and monitor their execution. The actual execution of a task is delegated to an executor, which is a binary whose goal is to execute a single task and report its result back to the server. Each executor runs inside a separated (and preferably sandboxed) process on a computing cluster. Executors communicate with the scheduler to regularly confirm they are still alive (i.e., heartbeating) and report their results when they successfully complete.

Because each task declares how much resources it needs, several tasks can be executed in parallel, as long as computational resources are available in the cluster. In practice, tasks belonging to the same run as well as tasks from different runs can be all executed in parallel, thus resulting in maximizing the resource usage of the cluster. For example, from the 16 tasks generated by the run description presented in Figure 6.2, the scheduler will first schedule the four `DatasetReader` root tasks (one for each of the four runs,

as they have no dependent task). They will hence be executed in parallel (if there is enough computational resources available, otherwise remaining tasks will be queued waiting for resources to be available). If execution goes well, the `Geo-I` tasks of each run will next be scheduled. Finally, `AreaCoverage` and `SpatialDistortion` of each run will be scheduled at the same time, as there are no dependencies between them.

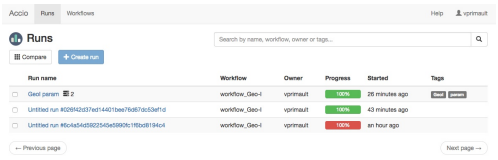
Our goal in this work is not to create yet another task scheduling system, as this has been already largely studied in the literature. Therefore, the scheduler can be implemented by relying on well-known resource managers such as Mesos [67] or HTCondor [146]. However, we also provide a simple local scheduler, where tasks are ran as sub-processes of the main server process.

6.4.6 Monitoring and analyzing results

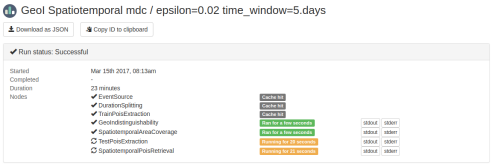
During the execution of a run, it is possible to get the progress from the command-line client or from the Web UI. Figure 6.4 shows what kind of information the latter provides.

Figure 6.4: Monitoring progress with Accio Web UI.

(a) List of runs.



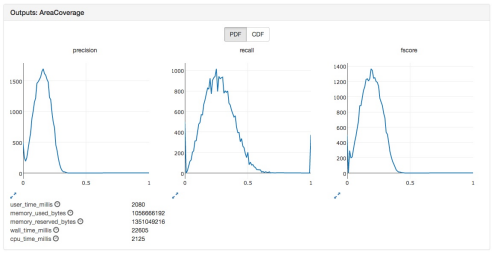
(b) Run under progress. Five tasks have completed and two tasks are running.



When a task completes successfully, its executor sends its result back to the server. The result of a task is formed of all outputs of the underlying operator. Moreover, additional execution metrics are also collected. These metrics are not directly generated by the operator but instead gathered by profilers. They are used to provide additional insights about the execution, such as its duration or its maximum memory consumption, and provide help when debugging failed operators. Finally, execution logs (i.e., standard output and standard error streams) are also made available by the scheduler. Task results are memoized to avoid recomputing again and again the same values. Before a task is actually launched, the controller checks whether there is already a result stored for the same task signature (i.e., operator name and values of inputs). If such a result is found, task execution is skipped and previous result is pulled back from the storage. This memoization feature also supports unstable operators by storing the seed that was used when launching the task and integrating it into the task's signature.

Results are then available through both the command-line client and the Web UI. The latter is more targeted towards visually previewing results and quickly taking a decision about the outcome of the experiment (cf. Figure 6.5), while the former is more suited for exporting the whole results as CSV, for a more detailed analysis. Accio does not intend to be a full-fledged analysis framework; we prefer to let the users have the control over the tools they want to leverage, whether it is Excel, Python or R.

Figure 6.5: Previewing results with Accio Web UI.



6.4.7 Extending with new operators

Accio has been designed to be extensible, which means that several components have pluggable implementations: storage, scheduler and operators. In this section, we give a hint about how custom operators can be implemented.

Operators implement a very simple interface, which is shown in Listing 6.3 (because Accio is developed in Scala, we present the equivalent Java interface for the sake of readability). It highlights what we said previously: an operator is really not much more than a function. The inputs of an operator are given via its constructor, and the outputs as the result of its `execute` method. The outputs can be as simple as a single real value or as complex as a Java class with several members. The `execute` method receives a single parameter which contains additional information, mainly the seed to be used by unstable operators to allow them to access a controlled source of randomness.

Listing 6.3: Accio operator interface.

```
interface Operator<T> {
  T execute(OpContext ctx);
}
```

Moreover, all operator classes have to be annotated with an `@Op` annotation that is also used to provide additional metadata, such as a human-readable description of what this operator does, or this amount of computational resources it requires.

The operator can then be implemented in any way the developper wants, with any library he needs, giving him maximum flexibility and the power to choose the right tools to achieve his goal. The code for the Geo-I operator is given as an example in Appendix A.

6.5 Case study: Experimenting with Accio

In this section, we evaluate Accio with a set of thorough use cases. The goal is to highlight how our platform actually unifies different LPPMs, datasets and metrics under a common model. More precisely, we reproduce our initial case study (in Section 3.6) and extend it to show what benefits Accio can provide. More precisely, we start with the evaluation of Geo-I [9] with a single privacy and a single utility metrics (Section 6.5.2). We then progressively enrich our analysis by adding more metrics (Section 6.5.3), more datasets (Section 6.5.4) and finally more LPPMs (Section 6.5.5). Table 6.1 summarizes

Table 6.1: Summary of the cases studies presented.

Use case	**#LPPMs**	**#Datasets**	**#Metrics**
Use case 1: Baseline	1	1	1
Use case 2: Metric diversity	1	1	5
Use case 3: Dataset diversity	1	2	5
Use case 4: LPPM diversity	3	1	5

the use cases presented in this section. We complete this evaluation with a discussion about the effort it took to write those experiments in Section 6.5.6.

6.5.1 Experimental setup

Datasets

We use two real-life datasets to evaluate Accio: Geolife and Cabspotting (cf. Section 3.5). They were all pre-processed to reduce the sampling rate to at most one event every five minutes, and to split a trace into two new traces belonging to new virtual users when there is an inactivity of at least 6 hours. Furthermore, we only kept the traces having at least 15 minutes of data.

Computing resources

Experiments were executed on a single machine "cluster" running Ubuntu 14.04, having access to 16 cores and 50 Gb of memory. In experiments presented in the following sections, we use the following subset of Accio operators.

Operators

We detail here the operators that are used in our case studies.

Pre-processing operators. The purpose of the pre-processing is either to clean a dataset to remove outliers (e.g., remove too short traces), to enforce some features for a fair comparison (e.g., sampling rate, duration) or to simulate an applicative use case (e.g., sending data by batches of six hours). We use three such operators in our experiments.

- `TemporalSampling(duration)`: samples traces to ensure a minimum duration between two consecutive points.
- `EnforceSize(minDuration,maxDuration)`: it rejects (resp. truncate) traces that do not respect a minimum (resp. maximum) duration threshold.
- `TemporalGapSplitting(duration)`: splits traces into two new traces each time the temporal gap between two consecutive points is greater than a specified duration.

LPPMs. These operators implement state-of-the-art protection mechanisms. They all take as inputs the actual dataset and some parameters, and produce as output a protected dataset. We use three such operators in our experiments.

- `Geo-I(epsilon)`: implements geo-indistinguishability (cf. Section 2.5.4 or [9]). It was reimplemented from the methodology described by the authors.
- `W4M(k,delta)`: implements $\mathcal{W}4\mathcal{M}$ (cf. Section 2.5.2 or [2]). We reused the binary that was made available by the authors [3] and simply wrote converters between our dataset format and theirs.
- `Promesse(alpha)`: implements PROMESSE (cf Chapter 4).

Privacy and utility metrics. These operators evaluate two different aspects: either the *privacy* gained from a dataset to another or the *utility* preserved from a dataset to another. We use five such operators in our experiments, plus an additional one to extract POIs.

- `PoisExtraction(maxDiameter, minTime)`: extracts POIs from a dataset using our algorithm (cf. Section 3.2.1) where $\Delta\ell = maxDiameter$ and $\Delta t = minTime$.
- `PoisRetrieval(actual, protected, threshold)`: implements the POIs retrieval metric (cf. Section 3.2.2) where $\sigma = threshold$, between an *actual* list of POIs and a *protected* list of POIs.
- `CountQueries(actual, protected, n, minSize, maxSize, minDuration, maxDuration)`: implements the count query distortion metric (cf. Section 3.3.3) by generating n random queries whose area size is between $minSize$ and $maxSize$ and whose time window in between $minDuration$ and $maxDuration$, between an *actual* dataset and a *protected* dataset.
- `AreaCoverage(actual, protected, level)`: implements the area coverage metric (cf. Section 3.3.4) with cells at a given *level*[1], between an *actual* dataset and a *protected* dataset.
- `SpatialDistortion(actual, protected)`: implements the spatial distortion metric (cf Section 3.3.1), between an *actual* dataset and a *protected* dataset.
- `PoisReident(actual, protected)`: implements the POIs-based reidentification metric (cf Section 3.2.3), between an *actual* list of POIs and a *protected* list of POIs.

A major difference between our definition of these metrics in Chapter 3 and their implementation in ACCIO is that the extraction of POIs is decoupled from the actual metric operator, which takes as input two lists of POIs (instead of two datasets). This allows for much more flexibility and less repetition in the implementation of the operators.

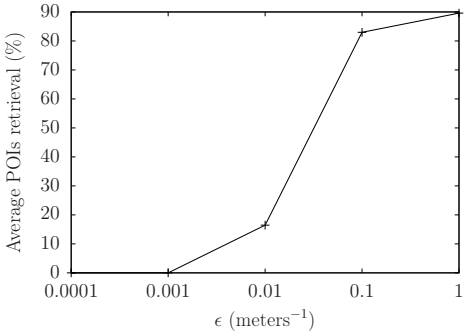
6.5.2 Use case 1: Baseline evaluation

In this experiment we evaluate the Geo-I LPPM with the POIs retrieval metric (cf. Section 3.2.2, Definition 6) for privacy and the count query distortion metric (cf. Section 3.3.3, Definition 12) for utility. Similarly to previous chapters, the POIs retrieval metric uses POIs extracted with $\Delta\ell = 200$ meters and $\Delta t = 15$ minutes, while the count query distortion metric is configured to generate 1,000 queries whose time windows range from 2 hours to 8 hours and whose half-diagonals range from 500 to 5,000 meters. We test multiple configurations of Geo-I, with $\epsilon \in \{0.0001, 0.001, 0.01, 0.1, 1\}$, on the Cabspotting dataset pre-processed as previously described.

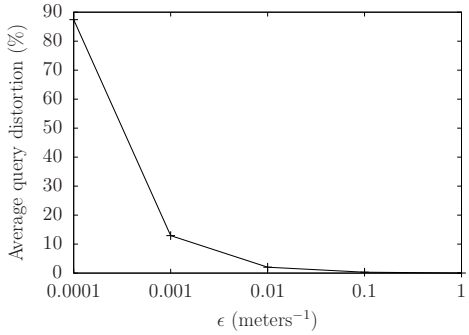
[1]We remind that we use Google's S2 geometry library [133] to generate cells on levels varying between 0 (the whole world) and 30.

Figure 6.6: Results of baseline evaluation of Geo-I.

(a) Privacy – POIs retrieval (lower is better)



(b) Utility – Count query distortion (lower is better)



This experiment translates into the workflow description shown in Listing B.1 (Appendix B), which represents 87 lines of JSON. Launching this workflow then only requires 14 lines of JSON to describe the run, as shown in Listing B.2 (Appendix B).

Results of this experiment are shown in Figure 6.6. We remind that the lower ϵ, the strongest the theoretical guarantee. Conversely, a high value of ϵ means a very relaxed theoretical privacy guarantee. Results clearly show a trade-off between privacy and utility, as it has been already highlighted in previous experiments. Until $\epsilon = 0.001$, privacy is perfectly preserved, with respect to chosen privacy metric, at the cost of a degraded utility (between 87 % and 12 %). Increasing ϵ results in weakening privacy while improving utility.

6.5.3 Use case 2: Metric diversity

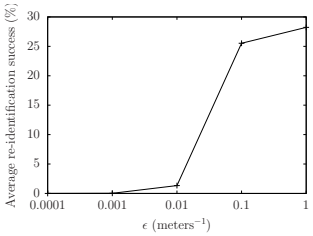
In this experiment, we complement the previous use case by adding another privacy metric and two other utility metrics. The re-identification success metric (cf. Section 3.2.3, Definition 7) uses the same POIs as the POIs retrieval metric, i.e., extracted with $\Delta\ell = 200$ meters and $\Delta t = 15$ minutes. The area coverage metric (cf. Section 3.3.4, Definition 14) is configured to extract cells at the 13th level, which corresponds approximately to a neighborhood inside a city, while the spatial distortion metric requires no parametrization.

Adding those three metrics requires 22 additional lines of JSON to the previous workflow description (cf. Listing B.1), and the run description is the same than in Listing B.2. This use case shows that Accio can accommodate with new metrics very easily. Moreover, it allows to experiment with metrics that were not specifically designed in the first place for the LPPM under consideration.

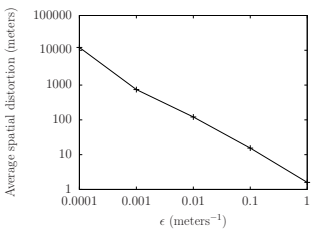
Figure 6.7 shows the results for our three complimentary metrics (POIs retrieval and count query distortion are already plotted in Figure 6.6, results are the same here). The privacy/utility trade-off is still visible: privacy and utility are evolving in opposite directions.

Figure 6.7: Results of metric diversity evaluation of Geo-I, featuring three new metrics.

(a) Privacy – Re-identification success (lower is better)



(b) Utility – Spatial distortion (lower is better)



(c) Utility – Area coverage (higher is better)

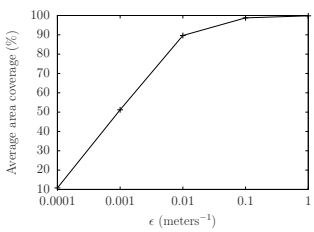
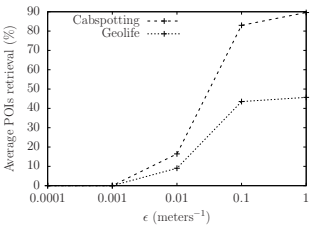
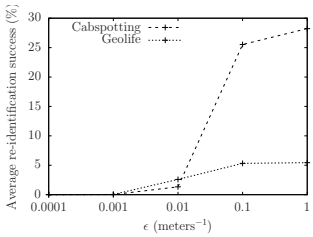


Figure 6.8: Results of dataset diversity evaluation of Geo-I, featuring two different datasets.

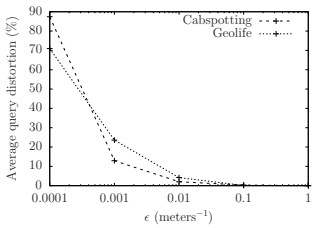
(a) Privacy – POIs retrieval (lower is better)



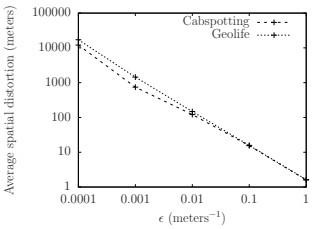
(b) Privacy – Re-identification success (lower is better)



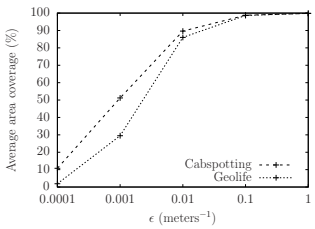
(c) Utility – Count query distortion (lower is better)



(d) Utility – Spatial distortion (lower is better)



(e) Utility – Area coverage (higher is better)

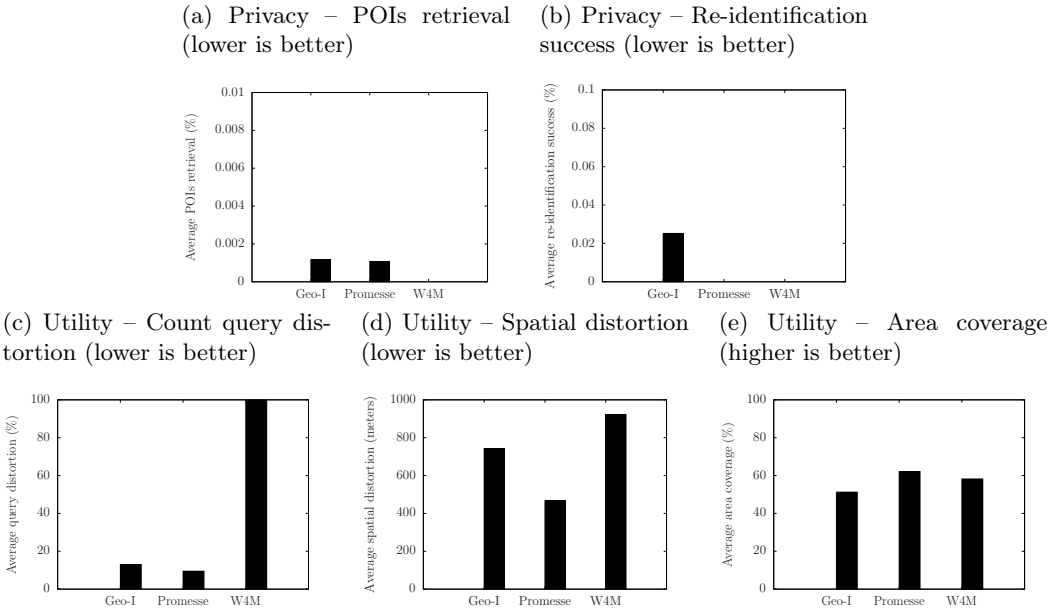


6.5.4 Use case 3: Dataset diversity

In this experiment, we evaluate Geo-I with the same metrics than in previous use case (cf. Section 6.5.3), but with another mobility dataset, Geolife (cf. Section 3.5). From the 2nd use case, only the run description changes by adding at line 5 the path to the Geolife dataset next to the path to the Cabspotting dataset. The same workflow definition is used.

Results for all metrics under both datasets are shown in Figure 6.8. These results are interesting because they exhibit similar behaviors for both datasets on all metrics, except for the re-identification success. Indeed, Figure 6.8b shows a much lower re-identification success when $\epsilon > 0.01$. Similarly than with metrics, ACCIO permits to integrate datasets very easily, thus allowing to cross-validate results. Our observation with the re-identification success shows how important it is to confirm results with multiple datasets.

Figure 6.9: Results of LPPM diversity evaluation, featuring three different LPPMs.



6.5.5 Use case 4: LPPM diversity

In this experiment, we take yet another step by comparing three LPPMs: Geo-I, $\mathcal{W}4\mathcal{M}$ and PROMESSE (cf. Section 2.5). We use the same privacy and utility metrics than in previous use cases (cf. Section 6.5.3 and Section 6.5.4), and the Cabspotting dataset. For the sake of readability, we chose one parametrization of each LPPM that is expected to offer a "good" trade-off between privacy and utility. Geo-I is configured with $\epsilon = 0.001$ by analyzing results of previous use cases. Indeed, this value of ϵ gave an almost perfect privacy level with a minimal utility loss (compared to $\epsilon = 0.0001$). For $\mathcal{W}4\mathcal{M}$, we choose the parametrization by reading their respective research papers and extracting from experimental results and tips given by authors reasonable parameters. We configure it with $k = 10$ and $\delta = 600$ meters. Finally, PROMESSE is configured with $\alpha = 200$ meters, which should be the optimal value to hide POIs extracted with $\delta\ell = 200$ meters (cf. Section 4.3.2).

From the 2nd use case, two additional workflows are created by replacing Geo-I with either PROMESSE or $\mathcal{W}4\mathcal{M}$. The workflow with PROMESSE takes the same number of JSON lines, while the workflow with $\mathcal{W}4\mathcal{M}$ takes an additional line (because the LPPM has one more parameter). The run description remains the same. We could also run all three LPPMs in parallel inside the same workflow, but because we have for now no way to express a branch inside a workflow, it would require to duplicate all metric operators, which we would rather prefer no to do.

Figure 6.9 exposes the results of this experiment. Because of the particular parameterizations we chose for each mechanism, they all feature good privacy levels, with a perfect privacy for $\mathcal{W}4\mathcal{M}$ with both metrics and almost perfect privacy for Geo-I and PROMESSE (with an average POIs retrieval below 0.001 % and an average re-identification success below 0.003 %). However, large differences appear when evaluating utility. They all show similar area coverage, between 50 % and 60 %, but behave sig-

Table 6.2: Size of JSON description files.

Use case	Workflow size	Run size
Use case 1: Baseline	87	14
Use case 2: Metric diversity	109	14
Use case 3: Dataset diversity	109	14
Use case 4: LPPM diversity	109 + 109 + 110	14

nificantly differently in termes of count query distortion and spatial distortion, where $\mathcal{W}4\mathcal{M}$ obtains the worst results. Figure 6.9d differ from the previous comparison of PROMESSE and Geo-I (cf. Section 4.4.3), with PROMESSE featuring a non-null spatial distortion. This is due to the fact that we have two different versions of this metric, the one that does a projection before computing the distortion (Definition 9, used in Chapter 4.4.3), and the one that uses the raw locations (Definition 8, used here).

6.5.6 Discussion

We want to stress that our goal in this chapter is not to evaluate once again state-of-the-art LPPMs and choose which one is better, but to highlight the flexibility of ACCIO and how easy it is to test new scenarii, even ones not considered by the authors of the LPPMs under consideration. It appears the most in the 4th use case, where LPPMs are evaluated with a mix of metrics coming from their respective papers and new ones.

Table 6.2 summarizes the sizes of the workflows and runs description expressed in JSON[2] that we used in this section. Overall, an experiment involving an LPPM, two datasets and five metrics (use case 3) needs a total number of 123 lines of JSON to be written by a scientist, to be compared with the few thousands of lines of code that would produce the same features[3]. Moreover, as showcased during these case studies, ACCIO allows to very easily alter an experiment (e.g., adding a new metric, a new LPPM, changing a parameter), without having to recompiling anything.

We also observe that run definitions are much shorter than workflow definitions. Differentiating between the two allow non-expert researchers to very easily launch a new run of an existing workflow, for example to experiment with a different value for a parameter, without having the burden of defining manually the whole graph of operators. Therefore, workflows act mostly as template for runs, and fulfill our objective of encouraging researchers to test alternative scenarii.

[2]We acknowledge that the number of lines is a coarse indicator, because it depends on the indentation chosen when presenting the JSON.

[3]As a rough indicator, the code of only the operators used in this experiment take about 1,050 single lines of code (excluding blank lines and comments) in Scala, which does not take into account the code that would be needed to orchestrate them properly outside ACCIO, nor the code dealing with reading and writing datasets (which is a library integrated into ACCIO).

6.6 Conclusion

In this chapter, we presented Accio, a framework designed to enhance location privacy study. This is not yet-another-LPPM but rather a platform to compare and evaluate LPPMs. The research in location privacy suffers from a heterogeneity problem. Indeed, many LPPMs are proposed but few are compared and evaluated using the same criteria. This is where Accio comes into play by proposing a unified framework for experimenting with location privacy. It includes a library of operators for spatio-temporal data manipulation, and new ones can easily be created. Accio comes with a JSON-based description format to express heavy and complex experiments. The user interacts with Accio via a Web or CLI interface. Different uses cases were presented showing how easy it is to launch experiments with Accio going from the evaluation of one LPPM with one privacy metric and one utility metric using one dataset to comparing multiple LPPMs with multiple metrics using multiple datasets.

Accio is already being actively used of the context of this thesis by other PhD students. It has quickly become the platform of choice for launching location privacy experiments. Consequently, there are plenty of future planned improvements on Accio. A major improvement would be to add built-in visualization tools to visualize mobility data on a map. Indeed, visualization would give researchers an intuition about what actually happens inside operators. Another major feature would be to support temporally-evolving workflows. There is a need to account more for the temporal dimension and design both smart attacks and LPPMs that evolve and adapt as the time goes. This feature would also allow to integrate ALP, that is for now not fully integrated inside Accio. Finally, we would like to integrate a more powerful DSL, allowing to write even more concise workflows and runs, and express more powerful constructs such as branches.

CHAPTER 7

Conclusion & Future Work

7.1 Conclusion

In the last six chapters, we discussed about the importance of protecting location privacy and presented ways to do it. We summarize in this section our principal findings.

7.1.1 Understanding and evaluating LPPMs

The first important matter was to understand the domain of study. Towards that purpose, we conducted a thorough survey of existing methods to enhance location privacy (i.e., LPPMs) and classified them across five different families: mix-zones, generalization-based, dummies-based, perturbation-based and rules-based. We also highlighted the metrics used by their authors to evaluate their LPPMs across three different families: privacy, utility and performance. This outlined the large diversity of LPPMs appearing in the literature, and as diverse means of evaluating them. Finally, we also presented two related approaches: privacy-by-design architectures and privacy-preserving query engines, which are both closely related but not directly solving our problem of protecting mobility datasets.

Then, we went deeper in our understanding of the ways to evaluate those LPPMs, by formalizing seven metrics, either coming from the state of the art or designed by ourselves: two privacy metrics, four utility metrics and one performance metric. We also presented seven mobility datasets that can be used to evaluate LPPMs, as well as their characteristics. Finally, we used those metrics and datasets to practically evaluate a state-of-the-art LPPM, Geo-I. Our experimental results showed that (1) a trade-off between privacy and utility is indeed difficult to achieve; (2) POIs are of importance and should be protected.

7.1.2 Protecting POIs

Then, we proposed a new LPPM named PROMESSE whose goal is specifically to hide POIs, while avoiding degrading too much the utility. The approach taken was to smooth speed, in order to make users appear to be constantly moving. Therefore, if a user seems to be constantly moving, it becomes difficult to identify his POIs, because by definition they are places where users are (almost) static. This had the impact of guaranteeing a high spatial accuracy, while degrading the temporal accuracy. We compared PROMESSE with two other competitors, and showed that our LPPM hides at least 97 % of POIs, which is similar to the results of other LPPMs. However, our mechanism comes with no spatial distortion, by design, while its competitors added from 24 meters to 70 kilometers of spatial error with our experimental settings.

7.1.3 Configuring LPPMs

Third, we introduced ALP, a solution to help users with the configuration of their LPPMs. The idea was to alleviate the burden of manually setting LPPM parameters and thus removing the need for final users to understand how their LPPM works (which seems required if we want them to be adopted!). Indeed, we shifted from a parameter-oriented workflow to an objective-oriented workflow, where users specify objectives to

achieve in terms of privacy and utility. Those objectives will be driven by their needs (in terms of privacy) and the needs of the LBSs they want to use (in terms of utility). Then, ALP translates these objectives into a set of parameters. Another feature of ALP is to be adaptive: the configuration happens in real-time and takes into account the data to protect. It means that parameters are dynamically chosen accordingly to the actual user's behavior. Our experimental evaluation showed that ALP is able to tune Geo-I and allows to achieve hiding all POIs for at least 75 % of the traces, while having a spatial distortion lower than 150m. Furthermore, we demonstrated the benefits of having an adaptive system as opposed to static parametrizations. With Geo-I, the range of values taken by its ϵ parameter was large, with more than 75 % of the users having ϵ values covering 80 % of the entire range of possible values.

7.1.4 Driving location privacy experimentation

Finally, we introduced Accio, which is a location privacy experimentation framework. It proposes a unified framework under which to describe and launch experiments. Its goal is to allow reproducibility of past experiments and drive innovation for future experiments. It is actually implemented on the Java Virtual Machine and made available as open source [124]. Accio comes with a library of 25 operators for various spatio-temporal manipulation tasks, state-of-the-art LPPMs and evaluation metrics. We demonstrated Accio's flexibility be launching classical experiments with three different LPPMs, five different metrics and on two datasets, by simply specifying the experiment as JSON. It allows non-technical users to leverage the existing operators by writing and launching their own experiments on Accio.

7.2 Future work

There is a large number of areas that have still to be explored, following the completion of this thesis. We present four of them in this section, a mix of research- and engineering-oriented perspectives.

7.2.1 Quantifying privacy & utility

Evaluating the efficiency of an LPPM is not an easy task. Fairly comparing LPPMs requires a robust procedure to *quantitatively* evaluate them. As shown in Chapter 2, there is a large variety of metrics used to evaluate LPPMs, although it is possible to categorize them in a small number of categories. We formally defined seven such metrics in Chapter 3, but it only represents a small subset of all metrics. In particular, the substantial work of Shokri et al. [138,139], primarily only focused on privacy, could be leveraged and integrated into our evaluation framework. Besides, our model was inspired by and close to theirs, which would ease an integration. On the utility side literature is far poorer. We would like to develop new utility metrics relying on practical use cases such as transportation modes detection (e.g., [142, 161]).

7.2.2 Users awareness

Most of the users are not aware of the risks related to the exploitation of their mobility data, and there is a lack of tools to improve users' awareness on this. To give again this example, *Please Rob Me* [122] is a website whose goal is to "raise awareness about oversharing", by showing it is possible to infer from geo-located tweets whether users are at home. Moreover, people are not aware of the value of their mobility data, certainly because they do not know the amount of knowledge that can be derived from it. A study showed that people would share their mobility trace in exchange of a little amount of money (the median was £10 or £20 for a commercial usage in [34]) or a gift (1% of chances to win a US$200 MP3 player in [82]). Despite not being pure research (but rather related to the dissemination of research results), we advocate it is one of the mission of researchers to raise awareness on societal problems such as privacy. Besides talks targeted towards the general public, tools could be developed to highlight privacy issues and the benefits of using an LPPM. For example, a manner to support our discourse would be a visual tool demonstrating various privacy attacks and their harmful effects, and the impact of an LPPM.

7.2.3 Datasets

To conduct experimental evaluations of LPPMs, researchers need real-life mobility datasets. We surveyed the most widely used ones in Section 3.5. However, despite their wide usage, all these datasets remain rather small (the largest dataset has 156 million events), far from the volume of data handled by actual LBSs. Consequently, it is challenging to evaluate an LPPM with a large and unaltered mobility data collection. To overcome this limitation, some works have investigated the generation of synthetic datasets (e.g., [18,103]) mimicking real mobility patterns and characteristics. Providing large mobility data collections would definitely be very useful for all research around location privacy. It is worth noting that during the context of this thesis, we collected our own dataset, Priva'Mov (also presented in Section 3.5), and developed a tooling around (to manage devices, visualize data, etc.). There is a real need to share methodologies and tools around those collections, and make them available to the research community. Some efforts are already going into that direction, such as the Funf [7] and APISENSE [64] platforms, or the Crawdad [32] community.

7.2.4 Implementation effort

A considerable amount of time was dedicated to Accio, which encompasses a large part of the work presented in this thesis inside a common framework. We believe that such a platform is fundamental to make significant progresses in the field of location privacy. For example, in distributed systems, such frameworks are pretty common: BFT-Smart [16], to evaluate byzantine fault-tolerant state machine replication algorithms, PeerSim [106], to evaluate peer-to-peer protocols, or Splay [86], which facilitates the deployment of distributed systems on a testbed. When available, these platforms give a common framework under which to evaluate further propositions. Moreover, researchers still need to make more often their implementations available, to allow others to compare with them, and possibly practitioners to actually use them.

A notable work is *ipShield* [21], which is actually implemented on the Android platform (though not necessarily installable trivially by end-users, because it is tightly integrated in the Android kernel). Geo-indistinguishability [9] has also been implemented by its authors as a browser extension [92] working with several popular browsers. This extension easily allows users to benefit from some privacy when using geolocated services through their Web browser. Another example is Aircloak [55], a project that aims to propose a trusted sensitive data collection architecture with privacy-preserving querying capabilities. By using several layers of noise, as well as maintaining a history of previous queries, the application is able to detect combinations of queries that could result in a privacy leak and prevent this to happen.

APPENDIX A

Code for the Geo-I operator

We present the Scala implementation of the Geo-I operator. We closely followed the instructions given by their authors [9]. We do not show `LambertW.lambertWm1`, because it is a purely mathematical computation. It implements the Lambert-W mathematical function [33], more precisely its W_{-1} branch.

```
package fr.cnrs.liris.accio.ops.lppms

import fr.cnrs.liris.accio.ops.SparkleOperator
import fr.cnrs.liris.accio.ops.model.Trace
import fr.cnrs.liris.accio.platform.sdk._

import scala.util.Random

@Op(
  help = "Enforce geo-indistinguishability guarantees on traces.",
  unstable = true,
  cpu = 1,
  ram = "2G")
case class GeoIndistinguishabilityOp(
  @Arg(help = "Privacy budget")
  epsilon: Double = 0.001,
  @Arg(help = "Input dataset")
  data: Dataset[Trace])
  extends Operator[GeoIndistinguishabilityOut] {

  override def execute(ctx: OpContext): GeoIndistinguishabilityOut = {
    val rnd = new Random(ctx.seed)
    val output = data.map(trace => noise(rnd, trace))
    GeoIndistinguishabilityOut(output)
  }

  private def noise(rnd: Random, trace: Trace): Trace = {
    trace.map(event => event.copy(point = noise(rnd, event.point)))
  }

  private def noise(rnd: Random, point: Point): Point = {
    val azimuth = math.toDegrees(rnd.nextDouble() * 2 * math.Pi)
    val z = rnd.nextDouble()
    val distance = inverseCumulativeGamma(z)
    point.translate(S1Angle.degrees(azimuth), distance)
  }

  private def inverseCumulativeGamma(z: Double): Distance = {
    val x = (z - 1) / math.E
    val r = -(LambertW.lambertWm1(x) + 1) / epsilon
    Distance.meters(r)
  }
}

case class GeoIndistinguishabilityOut(
  @Arg(help = "Output dataset")
  data: Dataset[Trace])
```

APPENDIX B

Description of the ACCIO baseline workflow

We present the entire description of the baseline workflow used in Section 6.5, involving Geo-I as an LPPM and two evaluation metrics.

Listing B.1: Baseline – JSON workflow description.

```
{
  "id": "baseline_geoind",
  "params": [
    {
      "name": "url",
      "kind": "string"
    },
    {
      "name": "epsilon",
      "kind": "double"
    }
  ],
  "graph": [
    {
      "op": "DatasetReader",
      "inputs": {
        "url": {"param": "url"}
      }
    },
    {
      "op": "TemporalSampling",
      "inputs": {
        "data": {"reference": "DatasetReader/data"},
        "duration": {"value": "5.minutes"}
      }
    },
    {
      "op": "TemporalGapSplitting",
      "inputs": {
        "data": {"reference": "TemporalSampling/data"},
        "duration": {"value": "6.hours"}
      }
    },
    {
      "op": "EnforceDuration",
      "inputs": {
        "data": {"reference": "TemporalGapSplitting/data"},
        "minDuration": {"value": "15.minutes"}
      }
    },
    {
      "op": "Geo-I",
      "inputs": {
        "epsilon": {"param": "epsilon"},
        "data": {"reference": "EnforceDuration/data"}
      }
    },
    {
      "op": "PoisExtraction",
      "name": "TrainPoisExtraction",
      "inputs": {
        "diameter": {"value": "200.meters"},
        "duration": {"value": "15.minutes"},
        "data": {"reference": "EnforceDuration/data"}
      }
    },
```

```
    {
      "op": "PoisExtraction",
      "name": "TestPoisExtraction",
      "inputs": {
        "diameter": {"value": "200.meters"},
        "duration": {"value": "15.minutes"},
        "data": {"reference": "Geo-I/data"}
      }
    },
    {
      "op": "PoisRetrieval",
      "inputs": {
        "threshold": {"value": "100.meters"},
        "train": {"reference": "TrainPoisExtraction/data"},
        "test": {"reference": "TestPoisExtraction/data"}
      }
    },
    {
      "op": "CountQueriesDistortion",
      "inputs": {
        "train": {"reference": "EnforceDuration/data"},
        "test": {"reference": "Geo-I/data"},
        "n": {"value": 1000},
        "minSize": {"value": "500.meters"},
        "maxSize": {"value": "5000.meters"},
        "minDuration": {"value": "2.hours"},
        "maxDuration": {"value": "8.hours"}
      }
    }
  ]
}
```

Listing B.2: Baseline – JSON run description.

```
{
  "workflow": "baseline_geoind",
  "params": {
    "url": {
      "value": "/path/to/cabspotting",
    },
    "epsilon": {
      "from": 0.0001,
      "to": 1,
      "step": 10,
      "log10": true
    }
  }
}
```

Bibliography

[1] Osman Abul, Francesco Bonchi, and Mirco Nanni. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In *ICDE*, pages 376–385. IEEE Computer Society, 2008.

[2] Osman Abul, Francesco Bonchi, and Mirco Nanni. Anonymization of moving objects databases by clustering and perturbation. *Information Systems*, 35(8):884–910, 2010.

[3] Osman et al. Abul. Wait 4 Me: Time-tolerant Anonymization of Moving Objects Databases – Executable. Available online at `http://kdd.isti.cnr.it/W4M/`.

[4] Jagdish Prasad Achara, Franck Baudot, Claude Castelluccia, Geoffrey Delcroix, and Vincent Roca. Mobilitics: Analyzing Privacy Leaks in Smartphones. *ERCIM News*, 2013(93), 2013.

[5] Gergely Acs and Claude Castelluccia. A Case Study: Privacy Preserving Release of Spatio-temporal Density in Paris. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '14, pages 1679–1688, 2014.

[6] Berker Agir, ThanasisG. Papaioannou, Rammohan Narendula, Karl Aberer, and Jean-Pierre Hubaux. User-side adaptive protection of location privacy in participatory sensing. *GeoInformatica*, 18(1):165–191, 2014.

[7] Nadav Aharony, Wei Pan, Cory Ip, Inas Khayal, and Alex Pentland. Social fmri: Investigating and shaping social mechanisms in the real world. *Pervasive Mobile Computing*, 7(6):643–659, December 2011.

[8] Ilkay Altintas, Chad Berkley, Efrat Jaeger, Matthew Jones, Bertram Ludascher, and Steve Mock. Kepler: An extensible system for design and execution of scientific workflows. In *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, SSDBM '04, pages 423–, Washington, DC, USA, 2004. IEEE Computer Society.

[9] Miguel E. Andrès, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential Privacy for Location-based Systems. In *CCS*, pages 901–914, 2013.

[10] Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604):452—454, May 2016.

[11] Bhuvan Bamba, Ling Liu, Peter Pesti, and Ting Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *WWW*, pages 237–246, 2008.

[12] Michael Barbaro and Tom Zeller Jr. Available online at `http://www.nytimes.com/2006/08/09/technology/09aol.html`, August 2006.

[13] Sonia Ben Mokhtar, Antoine Boutet, Louafi Bouzouina, Patrick Bonnel, Olivier Brette, Lionel Brunie, Mathieu Cunche, Stéphane D'Alu, Vincent Primault, Patrice Raveneau, Hervé Rivano, and Razvan Stanica. PRIVA'MOV: Analysing Human Mobility Through Multi-Sensor Dataset. Research report, LIRIS UMR CNRS 5205, April 2017.

[14] Alastair R. Beresford and Frank Stajano. Mix Zones: User Privacy in Location-aware Services. In *Percom Workshops*, pages 127–. IEEE Computer Society, 2004.

[15] A.R. Beresford and F. Stajano. Location privacy in pervasive computing. *Pervasive Computing, IEEE*, 2(1):46–55, January 2003.

[16] Alysson Bessani, João Sousa, and Eduardo E. P. Alchieri. State machine replication for the masses with bft-smart. In *Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN '14, pages 355–362, Washington, DC, USA, 2014. IEEE Computer Society.

[17] Igor Bilogrevic, Kévin Huguenin, Murtuza Jadliwala, Florent Lopez, Jean-Pierre Hubaux, Philip Ginzboorg, and Valtteri Niemi. Inferring Social Ties in Academic Networks Using Short-Range Wireless Communications. In *WPES*, pages 179–188, 2013.

[18] Vincent Bindschaedler and Reza Shokri. Synthesizing Plausible Privacy-Preserving Location Traces. In *Proceedings of the 2016 IEEE Symposium on Security and Privacy*, SP '16, 2016.

[19] Antoine Boutet, Sonia Ben Mokhtar, and Vincent Primault. Uniqueness Assessment of Human Mobility on Multi-Sensor Datasets. Research report, LIRIS UMR CNRS 5205, October 2016.

[20] Henry Carter, Benjamin Mood, Patrick Traynor, and Kevin Butler. Secure Outsourced Garbled Circuit Evaluation for Mobile Devices. In *Proceedings of the 22nd USENIX Conference on Security*, pages 289–304. USENIX Association, 2013.

[21] Supriyo Chakraborty, Chenguang Shen, Kasturi Rangan Raghavan, Yasser Shoukry, Matt Millar, and Mani Srivastava. ipshield: A framework for enforcing context-aware privacy. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, NSDI'14, pages 143–156, Berkeley, CA, USA, 2014. USENIX Association.

[22] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. A predictive differentially-private mechanism for mobility traces. In *PETS*, volume 8555 of *Lecture Notes in Computer Science*, pages 21–41. Springer Berlin Heidelberg, 2014.

[23] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. Constructing elastic distinguishability metrics for location privacy. In *Proceedings on Privacy Enhancing Technologies*, volume 2015, pages 156–170, June 2015.

[24] David L. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–90, February 1981.

[25] Rui Chen, Benjamin C.M. Fung, Bipin C. Desai, and Nériah M. Sossou. Differentially Private Transit Data Publication: A Case Study on the Montreal Transportation System. In *KDD*, pages 213–221, 2012.

[26] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and Mobility: User Movement in Location-based Social Networks. In *KDD*, pages 1082–1090, 2011.

[27] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private Information Retrieval. In *FOCS*, pages 41–, 1995.

[28] Chi-Yin Chow, Mohamed F. Mokbel, and Xuan Liu. A Peer-to-peer Spatial Cloaking Algorithm for Anonymous Location-based Service. In *SIGSPATIAL*, pages 171–178, 2006.

[29] City Domination GmbH & Co. KG. City Domination website. Available online at `http://www.citydomination.games`.

[30] Google privacy policy: Wp29 proposes a compliance package. Available online at `https://www.cnil.fr/fr/node/15712`, September 2014.

[31] Christian Collberg, Todd Proebsting, and Alex M. Warren. Repeatability and benefaction in computer systems research. Technical report, University of Arizona, February 2015.

[32] Crawdad Community. Crawdad website. Available online at `http://crawdad.cs.dartmouth.edu`.

[33] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the lambertw function. *Advances in Computational Mathematics*, 5(1):329–359, Dec 1996.

[34] George Danezis, Stephen Lewis, and Ross Anderson. How much is location privacy worth. In *Proceedings of the 4th Workshop on the Economics of Information Security*, 2005.

[35] Yves-Alexandre de Montjoye, Cesar A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports*, 3(1376), 2013.

[36] Ewa Deelman, Karan Vahi, Gideon Juve, Mats Rynge, Scott Callaghan, Philip J. Maechling, Rajiv Mayani, Weiwei Chen, Rafael Ferreira da Silva, Miron Livny, and Kent Wenger. Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*, 46(C):17–35, May 2015.

[37] Claudia Díaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In *Proceedings of the 2Nd International Conference on Privacy Enhancing Technologies*, PET'02, pages 54–68, Berlin, Heidelberg, 2003. Springer-Verlag.

[38] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[39] Cynthia Dwork. Differential Privacy. In *Automata, Languages and Programming*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2006.

[40] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *OSDI*, pages 1–6, 2010.

[41] Kassem Fawaz and Kang G. Shin. Location privacy protection for smartphone users. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 239–250, New York, NY, USA, 2014. ACM.

[42] Foursquare Labs, Inc. Foursquare location intelligence for enterprise. Available online at `https://enterprise.foursquare.com`.

[43] Foursquare Labs, Inc. Foursquare swarm website. Available online at `https://www.swarmapp.com`.

[44] Foursquare Labs, Inc. Foursquare website. Available online at `https://www.foursquare.com`.

[45] Lorenzo Franceschi-Bicchierai. Redditor cracks anonymous data trove to pinpoint muslim cab drivers. `http://mashable.com/2015/01/28/redditor-muslim-cab-drivers/`, January 2015.

[46] Julien Freudiger, Reza Shokri, and Jean-Pierre Hubaux. On the optimal placement of mix zones. In *PETS*, pages 216–234. Springer-Verlag, 2009.

[47] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next Place Prediction Using Mobility Markov Chains. In *MPM*, pages 3:1–3:6, 2012.

[48] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. GEPETO: A GEoPrivacy-Enhancing TOolkit. In *Proceedings of the 24th International Conference on Advanced Information Networking and Applications Workshops*, pages 1071–1076, April 2010.

[49] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Show Me How You Move and I Will Tell You Who You Are. *Transactions on Data Privacy*, 4(2):103–126, August 2011.

[50] Sebastien Gambs, Marc-Olivier Killijian, Izabela Moise, and Miguel Nunez del Prado Cortez. Mapreducing gepeto or towards conducting a privacy analysis on millions of mobility traces. In *Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing Workshops and PhD Forum*, IPDPSW '13, pages 1937–1946, Washington, DC, USA, 2013. IEEE Computer Society.

[51] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. De-anonymization attack on geolocated data. *Journal of Computer and System Sciences*, 80(8):1597–1614, 2014.

[52] Bugra Gedik and Ling Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. In *ICDCS*, pages 620–629, 2005.

[53] Gabriel Ghinita, Panos Kalnis, Ali Khoshgozaran, Cyrus Shahabi, and Kian-lee Tan. Private queries in location based services: anonymizers are not necessary. In *SIGMOD*, 2008.

[54] Gabriel Ghinita, Panos Kalnis, and Spiros Skiadopoulos. PRIVE: Anonymous Location-based Queries in Distributed Mobile Systems. In *WWW*, pages 371–380, 2007.

[55] Aircloak GmbH. Aircloak website. Available online at `https://www.aircloak.com`.

[56] Philippe Golle and Kurt Partridge. On the Anonymity of Home/Work Location Pairs. In *PerCom*, pages 390–397, Berlin, Heidelberg, 2009. Springer-Verlag.

[57] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1487–1495, New York, NY, USA, 2017. ACM.

[58] Google. Google maps geolocation api. Available online at `https://developers.google.com/maps/documentation/geolocation/intro`.

[59] Google Inc. Google Maps website. Available online at `https://maps.google.com`.

[60] Marco Gramaglia and Marco Fiore. Hiding Mobile Traffic Fingerprints with GLOVE. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '15, pages 26:1–26:13, New York, NY, USA, 2015. ACM.

[61] Marco Gruteser and Dirk Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *MobiSys*, pages 31–42. ACM, 2003.

[62] Saikat Guha, Mudit Jain, and Venkata N. Padmanabhan. Koi: A Location-Privacy Platform for Smartphone Apps. In *NSDI*, pages 183–196. USENIX, 2012.

[63] Nicolas Haderer, Vincent Primault, Patrice Raveneau, Christophe Ribeiro, Romain Rouvoy, and Sonia Ben Mokhtar. Towards a practical deployment of privacy-preserving crowd-sensing tasks. In *Proceedings of the Posters & Demos Session*, Middleware Posters and Demos '14, pages 43–44. ACM, 2014.

[64] Nicolas Haderer, Romain Rouvoy, and Lionel Seinturier. Dynamic Deployment of Sensing Experiments in the Wild Using Smartphones. In Jim Dowling and François Taïani, editors, *13th International IFIP Conference on Distributed Applications and Interoperable Systems*, volume LNCS-7891 of *DAIS '14*, pages 43–56, Florence, Italy, June 2013. Springer.

[65] Ramaswamy Hariharan and Kentaro Toyama. Project Lachesis: parsing and modeling location histories. In *Geographic Information Science*, pages 106–124, 2004.

[66] Heikki Henttu, Jean-Manuel Izaret, and David Potere. Geospatial Services: A $1.6 Trillion Growth Engine for the U.S. Economy. `http://www.bcg.com/documents/file109372.pdf`, 2012.

[67] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy Katz, Scott Shenker, and Ion Stoica. Mesos: A platform for fine-grained resource sharing in the data center. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, pages 295–308, Berkeley, CA, USA, 2011. USENIX Association.

[68] Baik Hoh and Marco Gruteser. Protecting location privacy through path confusion. In *SECURECOMM*, pages 194–205, 2005.

[69] Baik Hoh, Marco Gruteser, Hui Xiong, and Ansaf Alrabady. Preserving privacy in gps traces via uncertainty-aware path cloaking. In *CCS*, pages 161–171, 2007.

[70] Justin Hsu, Marco Gaboardi, Andreas Haeberlen, Sanjeev Khanna, Arjun Narayan, Benjamin C. Pierce, and Aaron Roth. Differential privacy: An economic method for choosing epsilon. In *Proceedings of the 2014 IEEE 27th Computer Security Foundations Symposium*, CSF '14, pages 398–410, Washington, DC, USA, 2014. IEEE Computer Society.

[71] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble Rap: Social-based Forwarding in Delay Tolerant Networks. In *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 241–250, New York, NY, USA, 2008. ACM.

[72] Patrick Hunt, Mahadev Konar, Flavio P. Junqueira, and Benjamin Reed. Zookeeper: Wait-free coordination for internet-scale systems. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, USENIXATC'10, pages 1–14, Berkeley, CA, USA, 2010. USENIX Association.

[73] Information and Privacy Commissioner of Ontario (Canada). Privacy by design. Available online at `https://www.ipc.on.ca/privacy/protecting-personal-information/privacy-by-design/`.

[74] Sharad Jaiswal and Animesh Nandi. Trust No One: A Decentralized Matching Service for Privacy in Location Based Services. In *MobiHeld*, 2010.

[75] Kaifeng Jiang, Dongxu Shao, Stéphane Bressan, Thomas Kister, and Kian-Lee Tan. Publishing Trajectories with Differential Privacy Guarantees. In *SSDBM*, pages 12:1–12:12, 2013.

[76] Ryo Kato, Mayu Iwata, Takahiro Hara, Akiyoshi Suzuki, Xing Xie, Yuki Arase, and Shojiro Nishio. A dummy-based anonymization method based on user trajectory with pauses. In *SIGSPATIAL*, pages 249–258, 2012.

[77] Hidetoshi Kido, Yutaka Yanagisawa, and Tetsuji Satoh. Protection of Location Privacy Using Dummies for Location-based Services. In *ICDE Workshops*, page 1248, 2005.

[78] Scott Kirkpatrick, C.D. Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[79] N. Kiukkonen, Blom J., O. Dousse, Daniel Gatica-Perez, and Laurila J. Towards rich mobile phone datasets: Lausanne data collection campaign. In *ICPS*, 2010.

[80] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011.

[81] John Krumm. Inference attacks on location tracks. In *Proceedings of the 5th International Conference on Pervasive Computing*, PERVASIVE'07, pages 127–143, Berlin, Heidelberg, 2007. Springer-Verlag.

[82] John Krumm. A Survey of Computational Location Privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, August 2009.

[83] John Krumm. Realistic driving trips for location privacy. In *PerCom*, volume 5538 of *Lecture Notes in Computer Science*, pages 25–41. Springer Berlin Heidelberg, 2009.

[84] John Krumm and Dany Rouhana. Placer: Semantic Place Labels from Diary Data. In *UbiComp*, pages 163–172. ACM, 2013.

[85] Juha K. Laurila, Daniel Gatica-Perez, Imad Aad, Jan Blom, Olivier Bornet, Trinh Minh Tri Do, Olivier Dousse, Julien Eberle, and Markus Miettinen. From big smartphone data to worldwide research: The mobile data challenge. *Pervasive Mob. Comput.*, 9(6):752–771, December 2013.

[86] Lorenzo Leonini, Étienne Rivière, and Pascal Felber. Splay: Distributed systems evaluation made simple (or how to turn ideas into live systems in a breeze). In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'09, pages 185–198, Berkeley, CA, USA, 2009. USENIX Association.

[87] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. Available online at `http://snap.stanford.edu/data`, June 2014.

[88] Haoran Li, Li Xiong, Lifan Zhang, and Xiaoqian Jiang. Dpsynthesizer: Differentially private data synthesizer for privacy preserving data sharing. *Proceeding of the VLDB Endowment*, 7(13):1677–1680, August 2014.

[89] Ninghui Li, Weining Yang, and Wahbeh Qardaji. Differentially private grids for geospatial data. In *Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE 2013)*, ICDE '13, pages 757–768, Washington, DC, USA, 2013. IEEE Computer Society.

[90] Ji Liu, Esther Pacitti, Patrick Valduriez, and Marta Mattoso. A Survey of Data-Intensive Scientific Workflow Management. *Journal of Grid Computing*, 13(4):457–493, 2015.

[91] Xinxin Liu, Han Zhao, Miao Pan, Hao Yue, Xiaolin Li, and Yuguang Fang. Traffic-aware multiple mix zone placement for protecting location privacy. In *INFOCOM*, pages 972–980, March 2012.

[92] Location guard. Available online at `https://github.com/chatziko/location-guard`.

[93] Location-privacy meter tool. Available online at `http://icapeople.epfl.ch/rshokri/lpm/doc/`.

[94] Wentian Lu, Gerome Miklau, and Vani Gupta. Generating private synthetic databases for untrusted system evaluation. In *Proceedings of the 2014 IEEE 30th International Conference on Data Engineering*, ICDE '14, pages 652–663, March 2014.

[95] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy Beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), March 2007.

[96] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, CA, USA, 1967. University of California Press.

[97] Sergio Mascetti, Dario Freni, Claudio Bettini, X.Sean Wang, and Sushil Jajodia. Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies. *The VLDB Journal*, 20(4):541–566, 2011.

[98] MathWorks. How simulated annealing works. Available online at `https://fr.mathworks.com/help/gads/how-simulated-annealing-works.html`.

[99] MaxMind. Maxmind website. Available online at `https://www.maxmind.com`.

[100] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 94–103, Washington, DC, USA, 2007. IEEE Computer Society.

[101] Frank D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, SIGMOD '09, pages 19–30, New York, NY, USA, 2009. ACM.

[102] Kristopher Micinski, Philip Phelps, and Jeffrey S. Foster. An Empirical Study of Location Truncation on Android. In *MOST*, pages 1–10, May 2013.

[103] Darakhshan J. Mir, Sibren Isaacman, Ramón Cáceres, Margaret Martonosi, and Rebecca N. Wright. DP-WHERE: Differentially private modeling of human mobility. In *Proceedings of the 2013 IEEE International Conference on Big Data*, BigData '13, pages 580–588, Oct 2013.

[104] Prashanth Mohan, Venkata N. Padmanabhan, and Ramachandran Ramjee. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 323–336, New York, NY, USA, 2008. ACM.

[105] Mohamed F. Mokbel, Chi-Yin Chow, and Walid G. Aref. The New Casper: Query Processing for Location Services Without Compromising Privacy. In *VLDB*, pages 763–774, 2006.

[106] Alberto Montresor and Mark Jelasity. Peersim: A scalable p2p simulator. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 99–100, Sept 2009.

[107] Hayam Mousa, Sonia Ben Mokhtar, Omar Hasan, Osama Younes, Mohiy Hadhoud, and Lionel Brunie. Trust management and reputation systems in mobile participatory sensing applications. *Computer Networks*, 90(C):49–73, October 2015.

[108] Min Mun, Sasank Reddy, Katie Shilton, Nathan Yau, Jeff Burke, Deborah Estrin, Mark Hansen, Eric Howard, Ruth West, and Péter Boda. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *Proceedings of the 7th International Conference on Mobile Systems,*

Applications, and Services, MobiSys '09, pages 55–68, New York, NY, USA, 2009. ACM.

[109] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 111–125. IEEE Computer Society, 2008.

[110] Mehmet Ercan Nergiz, Maurizio Atzori, and Yucel Saygin. Towards trajectory anonymization: A generalization-based approach. In *SIGSPATIAL Workshops*, pages 52–61, 2008.

[111] Hoa Ngo and Jong Kim. Location privacy via differential private perturbation of cloaking area. In *Proceedings of the 2015 IEEE 28th Computer Security Foundations Symposium*, CSF '15, pages 63–74, July 2015.

[112] Niantic, Inc. Pokemon GO website. Available online at `http://pokemongo.nianticlabs.com`.

[113] Anastasios Noulas, Salvatore Scellato, Neal Lathia, and Cecilia Mascolo. Mining User Mobility Features for Next Place Prediction in Location-Based Services. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining*, pages 1038–1043. IEEE Computer Society, 2012.

[114] Openstreetmap. Available online at `https://www.openstreetmap.org`.

[115] B. Palanisamy and Ling Liu. MobiMix: Protecting location privacy with mix-zones over road networks. In *ICDE*, pages 494–505, April 2011.

[116] Sai Teja Peddinti and Nitesh Saxena. On the Limitations of Query Obfuscation Techniques for Location Privacy. In *UbiComp*, pages 187–196, 2011.

[117] Nikos Pelekis, Aris Gkoulalas-Divanis, Marios Vodas, Despina Kopanaki, and Yannis Theodoridis. Privacy-aware Querying over Sensitive Trajectory Data. In *CIKM*, pages 895–904, 2011.

[118] Albin Petit, Thomas Cerqueus, Sonia Ben Mokhtar, Lionel Brunie, and Harald Kosch. Peas: Private, efficient and accurate web search. In *14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, TrustCom '15, pages 571–580, Washington, DC, USA, 2015. IEEE Computer Society.

[119] Sarah Pidcock and Urs Hengartner. Zerosquare: A Privacy-Friendly Location Hub for Geosocial Applications. In *MOST*, May 2013.

[120] A. Pingley, Wei Yu, Nan Zhang, Xinwen Fu, and Wei Zhao. CAP: A Context-Aware Privacy Protection System for Location-Based Services. In *ICDCS*, pages 49–57, June 2009.

[121] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24). Downloaded from `http://crawdad.cs.dartmouth.edu/epfl/mobility`.

[122] Please Rob Me. Available online at `http://pleaserobme.com`.

[123] Raluca Ada Popa, Andrew J. Blumberg, Hari Balakrishnan, and Frank H. Li. Privacy and accountability for location-based aggregate statistics. In *CCS*, pages 653–666, 2011.

[124] Vincent Primault. Accio: A location privacy framework – Source code. Available online at `https://github.com/privamov/accio`.

[125] Vincent Primault. Adaptive Location Privacy with ALP – Source code. Available online at `https://github.com/privamov/alp`.

[126] Vincent Primault, Sonia Ben Mokhtar, and Lionel Brunie. Privacy-preserving Publication of Mobility Data with High Utility. In *Proceedings of the 2015 35th IEEE International Conference on Distributed Computed Systems*, ICDCS '15, pages 802–803, June 2015.

[127] Vincent Primault, Sonia Ben Mokhtar, Cédric Lauradoux, and Lionel Brunie. Differentially Private Location Privacy in Practice. In *Proceedings of the 3rd Workshop on Mobile Security Technologies*, MOST '14, May 2014.

[128] Vincent Primault, Sonia Ben Mokhtar, Cédric Lauradoux, and Lionel Brunie. Time Distortion Anonymization for the Publication of Mobility Data with High Utility. In *Proceedings of the 4th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, TrustCom '15, pages 539–546, August 2015.

[129] Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, and Lionel Brunie. Adaptive Location Privacy with ALP. In *Proceedings of the 35th Symposium on Reliable Distributed Systems*, SRDS '16, pages 269–278, September 2016.

[130] Priva'Mov. Priva'Mov project's website. Available online at `http://privamov.liris.cnrs.fr`.

[131] Daniele Quercia, Ilias Leontiadis, Liam McNamara, Cecilia Mascolo, and Jon Crowcroft. SpotME If You Can: Randomized Responses for Location Obfuscation on Mobile Phones. In *ICDCS*, pages 363–372, 2011.

[132] Daniele Riboni and Claudio Bettini. Differentially-private release of check-in data for venue recommendation. In *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 190–198, March 2014.

[133] S2 geometry library. Available online at `https://github.com/google/s2-geometry-library-java`.

[134] Adam Sadilek and John Krumm. Far out: Predicting long-term human mobility. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 814–820. AAAI Press, 2012.

[135] Pravin Shankar, Vinod Ganapathy, and Liviu Iftode. Privately Querying Location-based Services with SybilQuery. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, UbiComp '09, pages 31–40, New York, NY, USA, 2009. ACM.

[136] Kumar Sharad and George Danezis. An automated social graph de-anonymization technique. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, WPES '14, pages 47–58, New York, NY, USA, 2014. ACM.

[137] Kang G. Shin, Xiaoen Ju, Zhigang Chen, and Xin Hu. Privacy protection for users of location-based services. *IEEE Wireless Communications*, 19(1):30–39, February 2012.

[138] Reza Shokri, Julien Freudiger, Murtuza Jadliwala, and Jean-Pierre Hubaux. A distortion-based metric for location privacy. In *Proceedings of the 8th ACM Workshop on Privacy in the Electronic Society*, WPES '09, pages 21–30, New York, NY, USA, 2009. ACM.

[139] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. Quantifying location privacy. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, SP '11, pages 247–262, Washington, DC, USA, 2011. IEEE Computer Society.

[140] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 2951–2959, USA, 2012. Curran Associates Inc.

[141] Jan Snyman. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Ba sed Algorithms*. Applied Optimization. Springer, 2005.

[142] Leon Stenneth, Ouri Wolfson, Philip S. Yu, and Bo Xu. Transportation Mode Detection Using Mobile Phones and GIS Information. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 54–63, New York, NY, USA, 2011. ACM.

[143] Leon Stenneth, Phillip S. Yu, and Ouri Wolfson. Mobile systems location privacy: "MobiPriv" a robust k anonymous system. In *2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 54–63, Oct 2010.

[144] Latanya Sweeney. k-Anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[145] Manolis Terrovitis. Privacy preservation in the dissemination of location data. *ACM SIGKDD Explorations Newsletter*, 13(1):6–18, 2011.

[146] Douglas Thain, Todd Tannenbaum, and Miron Livny. Distributed computing in practice: the condor experience. *Concurrency - Practice and Experience*, 17(2-4):323–356, 2005.

[147] Anthony Tockar. Riding with the stars: Passenger privacy in the nyc taxicab dataset. `https://research.neustar.biz/2014/09/15/riding-with-the-stars-passenger-privacy-in-the-nyc-taxicab-dataset/`, September 2014.

[148] European Union. EUR-Lex – directive 95/46/ec. Available online at `http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:31995L0046`, October 1995.

[149] European Union. EUR-Lex – regulation (eu) 2016/679. Available online at http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679, April 2016.

[150] Waze Mobile Ltd. Waze website. Available online at https://www.waze.com.

[151] Marius Wernke, Pavel Skvortsov, Frank Dürr, and Kurt Rothermel. A Classification of Location Privacy Attacks and Approaches. *Personal Ubiquitous Computing*, 18(1):163–175, January 2014.

[152] Michael Wilde, Mihael Hategan, Justin M. Wozniak, Ben Clifford, Daniel S. Katz, and Ian Foster. Swift: A language for distributed parallel scripting. *Parallel Computing*, 37(9):633–652, September 2011.

[153] Yonghui Xiao and Li Xiong. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1298–1309, New York, NY, USA, 2015. ACM.

[154] Yahoo!, Inc. Yahoo! Weather website. Available online at https://mobile.yahoo.com/weather/.

[155] Andrew C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, SFCS '82, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.

[156] Yelp, Inc. Yelp website. Available online at https://www.yelp.com.

[157] Tun-Hao You, Wen-Chih Peng, and Wang-Chien Lee. Protecting moving trajectories with dummies. In *Proceedings of the 2007 International Conference on Mobile Data Management*, MDM '07, pages 278–282, Washington, DC, USA, 2007. IEEE Computer Society.

[158] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 316–324, New York, NY, USA, 2011. ACM.

[159] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: Driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 99–108, New York, NY, USA, 2010. ACM.

[160] Hui Zang and Jean Bolot. Anonymization of location data does not work: A large-scale measurement study. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, pages 145–156, New York, NY, USA, 2011. ACM.

[161] Yu Zheng, Like Liu, Longhao Wang, and Xing Xie. Learning Transportation Mode from Raw Gps Data for Geographic Applications on the Web. In *Proceedings of the 17th International Conference on World Wide Web*, pages 247–256, New York, NY, USA, 2008. ACM.

[162] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, pages 791–800. ACM, 2009.

[163] Ge Zhong, Ian Goldberg, and Urs Hengartner. Louis, Lester and Pierre: Three Protocols for Location Privacy. In *Proceedings of the 7th International Conference on Privacy Enhancing Technologies*, PET'07, pages 62–76. Springer-Verlag, Berlin, Heidelberg, 2007.

[164] Changqing Zhou, Dan Frankowski, Pamela Ludford, Shashi Shekhar, and Loren Terveen. Discovering Personal Gazetteers: An Interactive Clustering Approach. In *Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems*, pages 266–273, 2004.

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : Primault
(avec précision du nom de jeune fille, le cas échéant)

DATE de SOUTENANCE : 01/03/2018

Prénoms : Vincent Guy Gabriel

TITRE : Practically Preserving and Evaluating Location Privacy

NATURE : Doctorat

Numéro d'ordre : 2018LYSEI

Ecole doctorale : InfoMaths

Spécialité : Informatique

RESUME :

Depuis quelques dizaines d'années, l'utilisation de téléphones contenant un capteur GPS a fortement augmenté, ouvrant la voie à de nouveaux usages tels que Google Maps ou Pokemon GO. Cependant, tous ces usages ne sont pas sans menace pour la vie privée des utilisateurs. En effet, les données de mobilité qu'ils envoient à ces services peuvent être utilisées pour inférer des informations sensibles telles que leur domicile, leur lieu de travail, leur bars préférés ou encore leurs amis. C'est à ce moment qu'entrent en action les mécanismes de protection, visant à redonner aux utilisateurs le contrôle sur leur vie privée. Ces mécanismes fonctionnent tous en altérant les données de localisation d'une façon ou d'une autre, ce qui donne naissance à un compromis entre vie privée (le niveau de protection) et utilité (la qualité de service).

Nous commençons par répertorier les mécanismes de protection existants et les métriques utilisées pour les évaluer. Cette première analyse met en avant une information particulièrement sensible : les points d'intérêt. Ces derniers représentent tous les lieux où les utilisateurs passent la majeure partie de leur temps, comme leur travail ou leur domicile. Cela nous conduit à proposer un nouveau mécanisme de protection, PROMESSE, dont le but principal est de cacher ces points d'intérêt.

Les mécanismes de protection sont en général configurés par des paramètres, qui ont un grand impact sur leur efficacité. Nous proposons ALP, une solution destinée à aider les utilisateurs à configurer leurs mécanismes de protection à partir d'objectifs qu'ils ont spécifié. Les configurations générées sont dynamiquement modifiées lorsque le comportement des utilisateurs change (par exemple s'ils déménagent).

Enfin, nous présentons Accio, un logiciel regroupant la majeure partie du travail de cette thèse. Il permet de lancer facilement des expériences destinées à étudier des mécanismes de protection, tout en renforçant leur reproductibilité.

MOTS-CLÉS : vie privée, données de mobilité, applications géolocalisées, mécanismes de protection, points d'intérêt

Laboratoire (s) de recherche : LIRIS

Directeur de thèse : Lionel Brunie

Président de jury :

Composition du jury :

Chbeir, Richard	Professeur, IUT de Bayonne et du Pays Basque	Rapporteur
Nguyen, Benjamin	Professeur, INSA-CVL	Rapporteur
Capra, Licia	Professeur, UCL	Examinatrice
Huguenin, Kévin	Professeur assistant, UNIL	Examinateur
Brunie, Lionel	Professeur, INSA-LYON	Directeur de thèse
Ben Mokhtar, Sonia	Chargée de recherche, INSA-LYON	Examinatrice
Lauradoux, Cédric	Chargé de recherche, INRIA	Examinateur