HAL
open science

Designing Location Privacy Mechanisms for flexibility over time and space

Marco Stronati

To cite this version:

Marco Stronati. Designing Location Privacy Mechanisms for flexibility over time and space. Cryptography and Security [cs.CR]. Ecole Polytechnique, 2015. English. NNT : . tel-01243295

HAL Id: tel-01243295

https://pastel.hal.science/tel-01243295

Submitted on 14 Dec 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

DOCTORAL THESIS

Designing Location Privacy Mechanisms for flexibility over time and space.

Author:
Marco Stronati
marco@stronati.org
http://www.stronati.org/

Reviewers:
Sébastien Gambs
Matteo Maffei

Supervisors:
Catuscia Palamidessi
Konstantinos Chatzikokolakis

Examiners:
Michalis Vazirgiannis
Kévin Huguenin
Markulf Kohlweiss
Santiago Zanella

September 2015

Abstract

With the increasing popularity of GPS-enabled handheld devices, location based applications and services have access to accurate and real-time location information, raising serious privacy concerns for end users. Trying to address these issues, the notion of *geo-indistinguishability* was recently introduced, adapting the well-known concept of Differential Privacy to the area of location-based systems. This is actually an instance of a more general property called $d_{\mathcal{X}}$-privacy, defined on arbitrary metric spaces: the notion of geo-indistinguishability is obtained by considering the Euclidean metric.

A Laplace-based obfuscation mechanism satisfying this privacy notion works well in the case of a *sporadic* use; Under repeated use, however, *independently* applying noise leads to a quick loss of privacy due to the correlation between the locations in the trace. In the first part of this thesis we show that correlation in the trace can be in fact exploited through a *prediction function* that tries to guess the new location based on the previously reported locations. However the inclusion of the prediction function in a privacy mechanism has to be private itself, leading to additional costs for the privacy budget of the user. If there is considerable correlation in the input trace, the extra cost of the test is small compared to the savings in budget, leading to a more efficient mechanism. Our carefully designed *budget managers* handle this balance between the costs of testing and sanitizing, producing a more efficient *predictive mechanism*.

The mechanism is evaluated using the Geolife and T-Drive datasets, containing traces of thousands of users in the Beijing area. The users are modeled as accessing a location-based service while moving around the city. Using a simple prediction function and two budget spending strategies, one optimizing the number of queries while the other their accuracy, we show that the predictive mechanism can offer substantial improvements over the independently applied noise.

Another problem (common to all location privacy approaches based on random noise) is that the repeated application of the mechanism will eventually dis-

close the *highly recurrent* locations, such as work or home. Even with the budget savings of the predictive mechanism, the user's privacy is bound to be breached in the long run. We propose a simple metric construction to model "geographic fences": Areas around highly recurrent locations where the mechanism reports uniformly, effectively stopping the privacy erosion. On one side the user has to release publicly the position of her fences but on the other the budget cost when reporting from inside them is zero, leading to a practical solution that can be used in combination with other metrics of the $d_{\mathcal{X}}$-privacy framework.

Although geo-indistinguishability presents various appealing aspects, it has the problem of treating space in a uniform way, imposing the addition of the same amount of noise everywhere on the map. This results in a non-optimal trade-off between privacy and utility, because low-density areas require more obfuscation than high-density areas to achieve the same degree of privacy. In the second part of this thesis we propose a novel elastic distinguishability metric that warps the geometrical distance, capturing the different degrees of density of each area. As a consequence, the obtained mechanism adapts the level of noise while achieving the same degree of privacy everywhere. The elastic metric can be efficiently computed and, in combination with the $d_{\mathcal{X}}$-privacy framework, achieves the desired degree of privacy everywhere with the minimal amount of noise locally required for "hiding-in-the-crowd".

We perform an extensive evaluation of our technique by building an elastic metric for Paris' wide metropolitan area, using semantic information from the OpenStreetMap database. We compare the resulting mechanism against the Planar Laplace mechanism satisfying standard geo-indistinguishability, using two real-world datasets from the Gowalla and Brightkite location-based social networks. The results show that the elastic mechanism adapts well to the semantics of each area, adjusting the noise as we move outside the city center, hence offering better overall privacy.

Lastly we propose a lighter version of the *elastic mechanism*, that requires no pre-computation of the metric, and is thus suitable for lower end devices and for an easier inclusion in existing systems. Of course this *tiled mechanism* provides less flexibility: Instead of adapting the noise differently in locations tens of meters apart, it can only adapt to large areas of a city, covering tens of square kilometers.

In conclusion the work developed in this thesis covers practical challenges faced in the development of privacy enhancing technologies, extending them to scenarios previously inaccessible. The $d_{\mathcal{X}}$-privacy framework provided a robust formal foundation of our privacy guarantees and thanks to its parametric nature

with respect to metrics, most of our results and techniques can be easily ported to domains other than location privacy. In order to test the practicality of our methods, all the mechanisms developed were thoroughly tested with real datasets and compared with the state of the art.

Résumé

Avec la croissante popularité des dispositifs équipés de GPS, les applications basées sur la géolocalisation ont accès à des informations précises et en temps réel sur la position des utilisateurs, posant des risques pour leur vie privée. Pour faire face à ce problème, la notion de géo-indiscernabilité a été introduite récemment, adaptant la célèbre Privacy Differentielle à le cadre de la géolocalisation. Il s'agit en fait d'une instance de $d_{\mathcal{X}}$-privacy, une notion plus générale de protection de la vie privée définie sur des espaces métriques arbitraires: géo-indiscernabilité est obtenue par l'espace métrique Euclideen.

Un mécanisme d'obfuscation basé sur du bruit Laplacien et satisfaisant cette notion de protection de la vie privée est efficace dans le cas d'un usage sporadique dans le temps, par contre, l'application indépendante de bruit amène à une perte rapide de protection à cause de la corrélation entre locations dans la trace. Dans la première partie de cette thèse on montre que la corrélation présente dans les traces peut, en fait, être exploitée à travers une fonction de prédiction, qui essaie à deviner la prochaine position à partir des celles déjà relâchées. Cependant l'inclusion de la fonction de prédiction dans le mécanisme doit être faite de façon privée elle même, apportant à des coûts additionnels pour le budget de vie privée de l'utilisateur. Dans le cas où il y a une forte corrélation dans la trace, le coût extra du test privé est petit par rapport au gagne en terme de budget. Nos algorithmes de gestion du budget se prennent charge de gérer l'équilibre entre coût de test et d'assainissement, réalisant un mécanisme prédictif plus flexible.

Le mécanisme est évalué utilisant les données des projets Geolife et T-Drive, contenant les traces des milliers d'utilisateurs dans la région de Beijing. Utilisant une simple fonction de prédiction et deux algorithmes de gestion du budget, un qui optimise la longueur des traces et l'autre leur précision, on montre que le mécanisme prédictif peut offrir des avantages importants par rapport à l'utilisation de bruit indépendant.

Malgré la géo-indiscernabilité présente plusieurs aspects positifs, elle présent le problème du traitement uniforme de l'espace, exigeant la même quantité de bruit dans les différentes zones géographiques. Pas conséquence on a un compromis insatisfaisant entre protection et utilité, parce que les zones à baisse densité nécessitent une offuscation majeure par rapport à celles plus denses pour obtenir le même niveau de protection. Dans la deuxième partie de cette thèse on propose une nouvelle métrique d'indiscernabilité qui peut déformer la distance géométrique,

capturant les différents degrés de densité pour chaque zone. Par conséquence, le mécanisme obtenu s'adapte au niveau de bruit et au même temps atteint le même degré de protection partout.

On a effectué des évaluations étendues de notre technique en réalisant un mécanisme élastique pour la région Parisienne, utilisant des donnés sémantiques du projet OpenStreetMap. On a comparé les résultats du mécanisme avec le mécanisme Laplacien Planaire qui garantit la géo-indiscernabilité, utilisant deux bases des donnés réelles obtenues à partir des réseaux sociaux Gowalla et Brightkite. Les résultats montrent que le mécanisme élastique s'adapte très bien à la sémantique de chaque région, augmentant le bruit en allant vers la banlieue.

En conclusion le travail développé dans cette thèse couvre des difficultés pratiques rencontrées dans la conception des mécanismes pour la protection de la vie privée, permettant leur utilisation dans des cadres nouveaux. La $d_{\mathcal{X}}$-privacy nous a fourni des fondations solides pour nos garanties des protection et grâce à sa nature paramétrique, la majorité des nos résultats sont réutilisables dans des domaines différents de la géolocalisation.

Acknowledgments

I would like to thank my advisors for guiding me during these three years. Catuscia for her positive and supportive attitude, wise suggestions and for showing me how to genuinely care for the people around you. Kostas for his endless patience and optimism, for showing me how research is just as much about creativity as it is about rigorous science and finally for countless climbing days. I want to thank my reviewers, Sébastien Gambs and Matteo Maffei, for the attention they dedicate to my work and to the jury at large, Michalis Vazirgiannis, Kévin Huguenin, Markulf Kohlweiss and Santiago Zanella, for evaluating it and for their precious feedback. I will be ever grateful to École Polytechnique and the french research community for welcoming and supporting me despite my terrible french. I can't but thank my team Comète for being such a great working environment (and ever changing!) and especially Luis Pino and Nicolas Bordenabe with whom I shared offices, movies, worries and joys that are part of the PhD student life. And in the hope of not forgetting anybody and getting all the accents right: Frank Valencia for his passion for research, Mário Alvim for his humor, Miguel Andrés for his friendliness, Salim Perchy for his movies reviews, Michell Guzmán for the vegetarianism, Joris Lamare for being french, Matteo Cimini for his excitement, Lili Xu for her kindness, Sophia Knight for her honesty, Andrés Artistizabal for his warmth, Ehab El-Salamouny for his tranquility, Sardouna Hamadou for his laugh, Yusuke Kawamoto for his Christmas song, Thomas Given-Wilson for his colorfulness. Finally I couldn't have done all this without the support of my dear wife Giulia, always at my side, and of my loving family.

Contents

1 Introduction **1**
1.1 They key role of privacy research 2
1.2 Location Privacy 3
1.3 State of the Art 5
1.4 Contributions 6
1.5 Publications 8
1.6 Plan of the thesis 8

2 State of the Art **9**
2.1 k-anonymity 9
2.1.1 l-diversity and t-closeness 10
2.1.2 Background knowledge 10
2.2 Differential Privacy 11
2.2.1 Compositionality and Budget 11
2.2.2 Background Knowledge 12
2.3 Bayesian Approach 13
2.3.1 min-entropy 14
2.3.2 g-leakage 14
2.3.3 Relation with Differential Privacy 16
2.4 Location Privacy 16
2.4.1 Location Data 16
2.4.2 Attacks 17
2.4.3 k-anonymity 18
2.4.4 Differential Privacy 19
2.4.5 Bayesian metrics for Location Privacy 19
2.4.6 Others metrics 21

3 Preliminaries **22**
3.1 Location Privacy through reduced Accuracy 22
3.2 Privacy Definitions 23
3.2.1 $d_{\mathcal{X}}$-privacy. 23
3.2.2 Geo-indistinguishability 24
3.2.3 Distinguishability level and ϵ 25
3.2.4 Traces 26
3.3 Privacy Mechanisms 27
3.3.1 Laplace Mechanism 27
3.3.2 Planar Laplace Mechanism 28
3.3.3 Exponential Mechanism 29
3.3.4 Independent Mechanism 29
3.4 Utility 30
3.4.1 Expected Error 31
3.4.2 $\alpha(\delta)$-accuracy 32

4 Repeated Use over Time **33**
4.1 Introduction 33
4.2 A Predictive $d_{\mathcal{X}}$-private Mechanism 36
4.2.1 Budget management 38
4.2.2 The mechanism 38
4.2.3 Privacy 39
4.2.4 Utility 43
4.2.5 Skipping the test 45
4.3 Predictive mechanism for location privacy 47
4.3.1 Prediction Function. 47
4.3.2 Budget Managers 48
4.3.3 Configuration of the mechanism 49
4.3.4 Evaluation 52
4.3.5 Future Work. 58
4.4 Incorporating fences in the metric 60
4.4.1 Fences 60
4.4.2 Mechanism 61
4.4.3 Future work 62
4.5 Conclusion 63

5 Flexible Use over Space **64**
5.1 Introduction 64
5.2 An elastic distinguishability metric 67
5.2.1 Privacy mass 68
5.2.2 Requirement 70
5.2.3 Extracting location quality 70
5.2.4 An efficient algorithm to build elastic metrics 72
5.3 Evaluation 76
5.3.1 Metric construction for Paris' metropolitan area 77
5.3.2 Evaluation using the Gowalla and Brightkite datasets . . . 78
5.4 Tiled Mechanism 83
5.5 Conclusion 85

6 Location Guard **87**
6.1 A web browser extension 87
6.2 Desktop and mobile 88
6.3 Operation 89
6.4 Adoption 90
6.5 Future Work 91

7 Conclusion **94**

Chapter 1

Introduction

The large increase in Internet connectivity during the past years has led to unprecedented amount of personal information online that poses new challenges for the privacy of individuals, in moral, juridic and technical terms. Despite the fact that many more people are connected, the Internet, instead of diversifying, has witnessed a large centralization in the hand of a few companies ("data silos"), due to their offer of free services. The services usually monetize on the data provided by the users ("data is the new oil"), creating rich collections of personal profiles that were unimaginable until a few years ago. On the other side we see public institutions more and more often forcing users to fill in overdetailed forms with personal information in order to access basic services e.g. online tax declarations or health insurance. Furthermore smart-phones have populated the Internet with new ubiquitous computers equipped with the "old" microphone and camera and a large array of new sensors (Proximity sensor, Gyroscope, Compass, Barometer, Accelerometer, Hall effect sensor, Ambient Light sensor, GPS/GLONASS/Beidou, step counter and detector, . . .). Numerous data inputs packaged together with a considerable computational capacity, high speed connectivity, manufactured on a large scale created new challenges for the protection of users's privacy.

Indeed several studies show that users are concerned for the handling of their personal data. Out of mistrust for service providers and lack of guidance on how to protect their privacy, users choose a range of behaviors, from the "quitters" [SBBV13], that completely abstain from a service, to "sell-outs", who, resigned to the death of privacy, are ready to exchange their data for a compensation [LLMS14, SOL$^+$14]. In the majority of cases however, users are mislead to reveal more that they are willing to or they are aware of [THD15].

When we look at the public sector, the double nature of the problem is even

more evident. The open-data and big-data movements are of central interest to foster scientific and technical innovation, improving government transparency and lowering administrative costs. On the other hand, while the news are filled with surveillance scandals (e.g. Snowden 2013), institutions are rushing to fill the legislative gap opened by new technologies (e.g. EU Privacy Reform[1]).

In this new data era we wonder what is the place of privacy research.

1.1 They key role of privacy research

In the early 2000s it became apparent after many credit card scams, that digital commerce, today a standard practice, was doomed to failure if the private and public parties were not able to cooperate and gain the trust of consumers.

Today, new data-driven technologies such as Big Data, Internet Of Things, machine learning, computational biology have to potential to solve the great challenges of our time but are at risk of being held back by the fear of the repercussion they might have over our private lives. Privacy research is again the key to restore the confidence of citizens in the new technologies and allow them to benefit from this untapped potential. A report from the European Commission DG Freedom, Security and Justice [Eco10] denotes the economic benefits of Privacy Enhancing Technologies (PETs) and future challenges. The US Defense Research Agency (DARPA) recently launched project Brandeis which "aims to enable individuals, enterprises, and U.S. government agencies to keep private and/or proprietary information private." These efforts prove that there is an awareness that the adoption of new ground-braking technologies will only be possible if regulations, guarantees and transparency are put in place.

The modern privacy scenario is extremely complex, a crossroad of technology, law, economy and social sciences. The contribution of information science can be twofold: finding formal privacy definitions (and measures) and designing mechanisms to enforce them.

The main characteristic that separates privacy form the other aspects of security is the *partial* disclosure of information. A classical problem of information security when handling a sensitive piece of information is keeping it confidential e.g. encrypting a document. When studying the information flow in a system, an analogous property is *non interference*, stating that from the observable outputs nothing should be learned of the secret inputs. These notions characterize a

[1] http://ec.europa.eu/justice/data-protection-reform

quality of the system, being safe or not.

On the contrary in privacy we typically *want* to extract part of the information of the secret in order to obtain some *utility* while maintaining safe its sensitive nature. For example a user may be willing to reveal her country of residence in order to have a website localized in the right language while keeping safe her home address. Because of the nature of the problem, we don't usually denote a system as being private or not, but rather *how much* private. In other words, in systems that leak some information by design, we are interested in quantitative *measures* of privacy to characterize this leakage. Finding the right balance between privacy and utility in one of the key challenges of the field.

Moreover privacy is an extremely diverse and personal topic, that may be interpreted differently from person to person. Going back to the example of the address, for some people revealing even the city of residence could still be considered harmless while for others revealing the country is already too much. Ultimately it is important to include the user in the discussion and leave room for personalization.

In summary the approach that we take in protecting privacy is to first define clearly what the *user* wishes to protect and *how much*, thus focusing on a formal quantitative *definition* of privacy. We then develop a *mechanism*, also called Privacy Enhancing Technology (PET), that enforces the definition and we evaluate it measuring both its privacy and utility.

1.2 Location Privacy

In recent years, the popularity of devices capable of providing an individual's position with a range of accuracies (e.g. through wifi access points, GPS satellites, etc) has led to a growing use of Location Based Services (LBSs) that record and process location data. A typical example of such systems are mapping applications, Points of Interest retrieval, coupon providers, GPS navigation, and location-aware social networks – providing a service related to the user's location. Although users are often willing to disclose their location in order to obtain a service, there are serious concerns about the privacy implications of the constant disclosure of location information. This concern was demonstrated, for instance, when in 2011 Allan and Warden [2] discovered that iOS4 was constantly logging the user's location in the device together with time-stamps for each location. The

[2] http://radar.oreilly.com/2011/04/apple-location-tracking.html

discovery made the news and Apple had to release a press statement claiming that the information was stored for the sole purpose of helping localizing the device when requested. Nonetheless the amount of information stored and especially the number of devices affected were enough to create alarm. More recently, during the Snowden scandal, The Guardian reported that NSA and GCHQ were targeting leaky mobile applications to collect user's private information, location being among them [Bal14]. Indeed location can be easily linked to a variety of other information that individuals wish to protect, such as sexual preferences, political views, religious inclinations, etc. Again from the Snowden documents we know that NSA was inferring personal relationships from phones tracking [Pos13].

On the private sector, building (or accessing) high quality maps, in the past 5 years, has been a foundational step of any IT company. Consider the economical effort of Apple and Microsoft to build their own in-house solutions to catch up with Google Maps. Incredibly large sums have been proposed by the automotive industry for the acquisition of Nokia Here. More recently Mapbox, a young startup, enjoyed extremely successful funding drives to build maps on top of OpenStreetMap data.

This interest for maps is not surprising if we consider that access to rich geolocated information is the fundamental starting point for all the other data-driven companies. These companies typically dispose of a large user base, whose private data is collected, analyzed and *joined* with the public information from the map. Companies generate most of their revenu from advertising - e.g. Foursquare, Facebook, Google, ... - and personalized ads, tuned to the profile of the user, are the most profitable. More recently however data licensing has become a source of revenue too. The data extracted from these huge datasets are sold to other companies or public administrations e.g. Strava Metro provides statistics for roads and city planning among other things. The size of the players in the sector of geolocated services should serve as indication of the strategic importance of this field. From the user perspective, location information is often considered the most valuable among personal informations [SOL$^+$14] and at the same time there is little awareness of the how and when this information is collected. For example, the location of a wifi connected device through a geolocated database of wifi SSIDs is an extremely opaque mechanism for the average user despite being by far the most often used in mobile devices.

1.3 State of the Art

In this thesis we consider the problem of a user accessing a LBS while wishing to hide her location from the service provider. We should emphasize that, in contrast to several works in the literature [GG03, MCA06], we are interested not in hiding the user's *identity*, but instead her *location*. In fact, the user might be authenticated to the provider, in order to obtain a personalized service (personalized recommendations, friend information from a social network, etc); still she wishes to keep her location hidden.

Several techniques to address this problem have been proposed in the literature, satisfying a variety of location privacy definitions. A widely-used such notion is k-anonymity (often called l-diversity in this context), requiring that the user's location is indistinguishable among a set of k points. A different approach is to report an *obfuscated* location z to the service provider, typically obtained by adding random noise to the real one. Shokri et al. [STT+12] propose a method to construct an obfuscation mechanism of optimal privacy for a given quality loss constraint, where privacy is measured as the expected error of a Bayesian adversary trying to guess the user's location [STBH11].

The main drawback of the aforementioned location privacy definitions is that they depend on the adversary's background knowledge, typically modeled as a prior distribution on the set of possible locations. If the adversary can rule out some locations based on his prior knowledge, then k-anonymity will be trivially violated. Similarly, the adversary's expected error directly depends on his prior. As a consequence, these definitions give no precise guarantees in the case when the adversary's prior is different.

Differential privacy [Dwo06] was introduced for statistical databases exactly to cope with the issue of prior knowledge. The goal in this context is to answer aggregate queries about a group of individuals without disclosing any individual's value. This is achieved by adding random noise to the query, and requiring that, when executed on two databases x, x' differing on a single individual, a mechanism should produce the same answer z with similar probabilities. Differential privacy has been successfully used in the context of location-based systems [MKA+08, HR11, CAC12] when *aggregate* location information about a large number of individuals is published. However, in the case of a single individual accessing a LBS, this property is too strong, as it would require the information sent to the provider to be independent from the user's location.

Our work is based on *geo-indistinguishability*, a variant of differential pri-

vacy adapted to location-based systems, introduced recently in [ABCP13]. Based on the idea that the user should enjoy strong privacy within a small radius, and weaker as we move away from his real location, geo-indistinguishability requires that the closer (geographically) two locations are, the more indistinguishable they should be. This means that when locations x, x' are close they should produce the same reported location z with similar probabilities; however the probabilities can become substantially different as the distance between x and x' increases. The result is that the mechanism protects the accuracy of the real location but reveals that the user is, say, in Paris instead than London, which is appropriate for the kind of applications (LBSs) we are targeting. Together with the privacy definition the authors propose a mechanism to enforce it, the Planar Laplace mechanism, which adds noise to the user's location drawn from a 2-dimensional Laplace distribution.

Despite the simplicity and effectiveness of geo-indistinguishability together with its Planar Laplace mechanism, it presents two major drawbacks for its applicability in practice. First, the repeated use of the mechanism causes a rapid erosion of the privacy guarantee due to the correlation between locations in a trace. This is common to all obfuscation techniques that in general treat privacy as a budget that eventually is depleted, limiting greatly their use over time. The mechanism needs to be more flexible over *time*, adapting to different budget saving strategies and privacy goals of the user. Second, geo-indistinguishability needs to be tuned to a fixed degree of privacy, while in practice the protection should adapt depending on the characteristics of the user's location. The privacy need in a dense city is different from a empty countryside and the mechanism needs to be flexible with the user's movement in *space*.

1.4 Contributions

We show that the correlation in the trace can be exploited in terms of a *prediction function* that tries to guess the new location based on the previously reported locations. The proposed mechanism tests the quality of the predicted location using a private test; in case of success the prediction is reported otherwise the location is sanitized with new noise. If there is considerable correlation in the input trace, the extra cost of the test is small compared to the savings in budget, leading to a more efficient *predictive* mechanism.

For an accurate use of the budget we also introduced a *budget manager*, a component that given a global privacy budget can tune the mechanism to a differ-

ent level of privacy at each location release. The two strategies that we developed allow, from a fixed privacy budget to prolong the use the mechanism over time, degrading the accuracy of the reported location, or viceversa to limit the noise addition at the cost of a limit number of reported locations. We evaluate the mechanism in the case of a user accessing a location-based service while moving around the city of Beijing thanks to two large datasets of real GPS traces, Geolife and TDrive. Using a simple prediction function and two budget spending strategies, optimizing either the utility or the budget consumption rate, we show that the predictive mechanism offers substantial improvements over independently applied noise.

Additionally, for highly recurrent locations such as work and home, we propose the use of geographic fences that on one side are publicly known to the attacker while on the other have no cost in terms of privacy, no matter how many times the mechanism is used (while in these locations). This technique can be used in combination with any other $d_{\mathcal{X}}$-private mechanism to form a *fenced* mechanism effectively stopping the privacy erosion.

The predictive and fenced mechanisms extends geo-indistinguishability with great flexibility of the budget usage over a prolonged *time*.

In the second part of this thesis we use semantic information extracted from the OpenStreetMap geographical database to build an *elastic* mechanism that adapts the privacy protection to the features of the user's location. We develop an efficient graph-based algorithm to compute the distinguishability metric and test its scalability by building an elastic mechanism for the Paris' wide metropolitan area. The mechanism is evaluated over the datasets of two location-based social networks, Gowalla and Brightkite, and shows a flexible behavior over *space*, providing adequate protection in the city as well as in less dense areas.

Moreover we propose an idea for a more efficient, although less flexible, version of the elastic mechanism. This mechanism provides a different protection degree, but for larger areas, like tiles covering a map. The *tiled* mechanism despite being more coarse than the elastic, can be computed for a fraction of the resources and this allows for easy integration with existing tools such as Location Guard, a popular browser extension for geo-indistinguishability.

Finally we describe in detail Location Guard and its central role for experimenting with the techniques developed in this thesis.

1.5 Publications

The work presented in this thesis has been partially published in the following conferences. The predictive mechanism, in Section 4.2, has first appeared in the paper *A Predictive Differentially-Private Mechanism for Mobility Traces* in the proceedings of Privacy Enhancing Technologies Symposium 2014 [CPS14]. The elastic and fenced mechanism appeared in the paper *Constructing elastic distinguishability metrics for location privacy* in the proceedings of Privacy Enhancing Technologies Symposium 2015 [CPS15]. Additionally all the mechanisms have been implemented and, together with their evaluation, they are available with an open source license on github [ela]. Finally Location Guard has been available in the different browsers websites for download and its source code is available at [loc] since November 2013.

1.6 Plan of the thesis

In Chapter 3.1 we introduce the concepts developed in the literature that are necessary to build our work upon. In Chapter 4 we present the predictive mechanism, explaining in details the challenges posed by repetitive use, the various components of the mechanism and its proofs of privacy and utility. A large section is devoted to the experimental evaluation of the predictive mechanism with two large datasets of location traces from real users and the comparison with standard geo-indistinguishability. The chapter concludes with the presentation of geographical fences and their integration in the other mechanisms. In Chapter 5 we explain in detail the rigidity of standard geo-indistinguishability and propose our elastic mechanism. We propose an efficient and scalable graph-based algorithm to compute the mechanism and perform an extensive evaluation using two datasets of location based social networks. We then introduce the tiled mechanism as a lighter alternative, explaining how we can obtain a approximate elasticity with modest computing resources. We conclude outlining a practical implementation of the tiled mechanism for inclusion in Location Guard. More details on Location Guard can be found in Chapter 6, including motivation and adoption.

Chapter 2

State of the Art

In this chapter we introduce some popular notions of privacy and location privacy that are closely related to our work. We start by introducing two notions of privacy for statistical databases, k-anonymity and differential privacy, that preceded location privacy and largely influenced it. We then present a Bayesian approach from the area of Quantitative Information Flow, that can be used as alternative characterization of the leakage of a privacy mechanism. We then introduce location privacy, describing in more details the data produced and collected and the existing attacks. Finally we introduce the privacy notions and mechanism used for location privacy and explain their connection to the techniques introduced for statistical databases.

2.1 k-anonymity

In the context of statistical databases, we have a dataset populated with individuals and their personal (possibly sensitive) information. An analyst, or curator, wants to query the database to extract some statistical properties that are of interest for the general population. At the same time we would like to keep the sensitive information of a single participant safe, no matter what query or combination of queries are asked. For example in a hospital database we would like to learn if there is a larger incidence of cancer in a particular region but at the same time we want to hide the fact that a specific individual living in the region has cancer.

A widely used notion of privacy is k-anonymity, first introduced in 1998 by Samarati and Sweeny [SS98] and further developed over the years [Sam01, Swe02]. In order to protect the identities of the users present in the database, that

may contain sensitive information about them, the explicit identifiers such as name and surname are typically suppressed. However there are several other attributes that could help re-identify users with the help of background knowledge. For example birth date and address could be present in another dataset together with the name of the user. For this reason these attributes are called *quasi-identifiers* and need to be sanitized. K-anonymity proposes to generalize or suppress the quasi-identifiers in order to ensure that each entry in the database is part of a group of at least k individuals. In the example above, instead of providing the complete birth date, we could suppress the day of birth, assuming that there are at least k other users born in the same month. The intuition behind this technique is to increase the attacker's uncertainty by increasing the number of possible entries related to a certain user.

2.1.1 l-diversity and t-closeness

k-anonymity should provide a *measure* of privacy for the user, with a larger k corresponding to a higher privacy, however the size of the set where the user is hidden does not necessarily capture all that the set can reveal about the user. Imagine a group of k users where all have cancer, in this case, despite the fact that k-anonymity is satisfied, the attacker can learn a sensitive information that was uniform in the group. l-diversity [MKGV07] was introduced to overcome this issue by requiring that each group has a certain amount l of diversity among its *sensitive* attributes. A refinement of this technique was proposed in t-closeness [LLV07] where each group is required to be at a distance t from the distribution describing the general population.

2.1.2 Background knowledge

The major problem with k-anonymity and its derivatives is their weakness with respect to background knowledge, or conjunction attacks. The reason is that even if a specific database has been sanitized to guarantee a certain k, the attacker can use knowledge from another source to reduce his uncertainty and lower k. In other words, the measure of privacy k is valid only when posing strong assumptions on the background knowledge of the attacker.

2.2 Differential Privacy

Differential privacy is a property meant to guarantee that the participation in a database does not constitute a threat for the privacy of an individual. In was introduced in 2006 by Dwork et al. [Dwo06] in the context of statistical disclosure and found widespread acceptance in several fields. More precisely, the idea is that a (randomized) query satisfies differential privacy if two databases that differ only for the addition of one record are almost indistinguishable with respect to the results of the query. Two databases $D, D' \in \mathcal{D}$ that differ only for the addition of one record are called *adjacent*, denoted by $D \sim D'$.

Definition 1 (Differential privacy)**.** *A randomized function $\mathcal{K} : \mathcal{D} \to Z$ satisfies ϵ-differential privacy if for all pairs $D, D' \in \mathcal{D}$, with $D \sim D'$, and all $Y \subseteq Z$, we have that:*

$$Pr[\mathcal{K}(D) \in Y] \leq Pr[\mathcal{K}(D') \in Y] \cdot e^{\epsilon}$$

where $Pr[E]$ represents the probability of the event E.

The definition requires the distributions of reported values to be very "close" in the two scenario where the user participate or not. How close is measured by the parameter ϵ which indicates the *risk* of distinguishing the two cases.

Notice that the adjacency relation $\sim$ can also be expressed as the Hamming metric d_H over databases. In this case the definition should hold for all databases at distance 1.

The typical mechanism used for differential privacy is the Laplace mechanism, discussed in more details in the next Chapter 3.1.

2.2.1 Compositionality and Budget

One of the main advantages of differential privacy is its compositionality. It is easy to prove the privacy provided by the sequential application of several differentially private mechanisms.

Theorem 1 (Sequential Composition [McS09])**.** *Let K_i each provide ϵ_i-differential privacy. Performing all the K_i queries on the database D provides $(\sum_i \epsilon_i)$-differential privacy.*

This theorem shows that there is an accumulation of the ϵ_i with the number of accesses to the database. ϵ is a measure of the *risk* for the user's privacy and its growth due the continuous access to the database meets the intuition. In the design

of a mechanism to deploy in a realistic setting, where the analyst perform several queries for a period of time, this risk accumulation has to be taken into account. Usually we refer to the global sum of all ϵ_i as the privacy *budget*; every access has to be accounted for and a limit has to be set after which the system halts.

Having a finite amount of privacy to *spend* presents two main challenges to the design of a realistic system. First, the budget has to be managed following a strategy in order to achieve a goal. In Section 4.3.2 for example, we describe two possible *budget managers* to maximize either the number of queries or the accuracy of each query.

Second, the accumulation of budget described in Theorem 1 models a worst case scenario where only the number of accesses are considered, not the content of the queries. Imagine for example that several analysts independently perform the same query, if we naively run them, Theorem 1 applies. However if we run the query only once, cache the result and then return it to the other analysts, only one access needs to be accounted for. A large body of work has been developed to optimize the set of queries so to minimize the direct accesses to the database. The median mechanism [RR10] is based on the idea of exploiting the correlation on the queries to improve the budget usage. The mechanism uses a concept similar to our *prediction* to determine the answer to the next query using only past answers. An analogous work is the multiplicative weights mechanism [HR10]. The mechanism keeps a parallel version of the database, built from past queries, which is used to predict the next answer and in case of failure it is updated with a multiplicative weights technique.

In Section 4.2 we equip a privacy mechanism with a prediction function that from past reported values tries to guess future answers, that comes for free on the privacy budget. A key difference of statistical databases from our context is that in the former, several queries are performed against the *same database*. In our setting, however, the secret (the position of the user) is always changing and the query is just the identity query asking the exact location. A similar scenario is explored also in [DNPR10] were the authors consider the case of an evolving secret and develop a differentially private counter.

2.2.2 Background Knowledge

Among the advantages of this definition is the independence from the prior distribution on the secrets. There are no assumptions on the database or on the capabilities of the attacker, making differential privacy a very robust property

with respect to background knowledge. This is also due to the fact that contrary to k-anonymity, where the uncertainty of the attacker comes from the size of the groups, in differential privacy the uncertainty is introduced through random noise. Indeed the probabilistic nature of the definition is the key difference with k-anonymity and it is common with another line of work developed by [STBH11] in the context of location privacy that we present in details in the next chapter 2.3.

2.3 Bayesian Approach

A alternative point of view on privacy can be provided by the field of Quantitative Information Flow. The system under analysis is considered as to be an information theoretic channel, the input of the channel represents the secret and the output the observable. This model is adapted from the work of Shannon [Sha48] on communication channels and indeed a number of works use Shannon's *entropy* as a measure of the attacker's uncertainty of the secret [Mal07, CHM07, CPP08a].

Imagine for example a password-based authentication system, in this case the system is the channel, the password is the secret input and the observable is either `success` or `fail`. At first the entropy of the secret depends only on the size of the passwords space, however each time the attacker tries an incorrect password, this entropy decreases slowly.

The *mutual information* between the input and the output of the channel is the information leakage caused by the system and the maximum possible leakage is represented by the *capacity* of the channel. More recently a number of other measures have been introduced, more specific to information hiding: guessing entropy [Mas94], Bayes risk [CPP08b], min-entropy leakage [Smi09], g-leakage [ACPS12].

In general we are interested in measuring the uncertainty that the attacker has over the secret $\mathcal{X}$ and especially if, and *how much*, it is reduced after the attacker sees the output of the system $\mathcal{Z}$. Independently from the measure of choice, the general principle proposed by Smith in [Smi09] is:

$$\text{Information Leakage} = \text{Initial Uncertainty} - \text{Final Uncertainty}$$

The idea is that following a Bayesian approach, the attacker, after the observation of the output of the system, updates his information on the secret, possibly refining it.

2.3.1 min-entropy

We define two finite sets $\mathcal{X}$ and $\mathcal{Z}$ respectively of secret input values and observable output values. A channel is a triple $(\mathcal{X}, \mathcal{Z}, C)$, where C is a channel matrix, an $|\mathcal{X}| \times |\mathcal{Z}|$ matrix with entries between 0 and 1 and rows that sum to 1. Each element $C[x, z]$ is the conditional probability of obtaining an output z from an input x. We define π the *prior distribution* on $\mathcal{X}$, that represents the knowledge of the attacker on the secret *before* observing the system.

In min-entropy, rather that information content of a random variable like in Shannon's entropy, we measure the uncertainty as the maximum probability that the attacker as of guessing the secret *in one try*. We call this the *vulnerability* of the secret. For example the vulnerability measurable before to running the system is simply the most likely value in the prior distribution; its most vulnerable.

$$V(\pi) = \max_{x \in \mathcal{X}} \pi(x)$$

We can then obtain min-entropy with a simple change of scale to bits:

$$H_\infty(\pi) = -\log_2 V(\pi)$$

However we are interested in enriching this measure with a notion of gain, in order to model different attackers.

2.3.2 g-leakage

For the purposes of this thesis we focus on g-leakage [ACPS12], that extends min-entropy leakage [Smi09] with gain functions, allowing us to model different types of attacker. We could imagine for example that for the authentication system, an attacker may be interested in learning the credential of a specific user or alternatively it may be enough to recover the password of any user. We introduce the concept of *gain function* to represent the goal of the attacker, note that in the literature sometime this is achieved using a *loss* function, which is simply the complement of a gain function.

Definition 2 (Gain function)**.** *Given a set of secrets $\mathcal{X}$ and finite set of* guesses $\mathcal{W}$ *a* gain function *is a function $g : \mathcal{W} \times \mathcal{X} \to [0, 1]$.*

We can now extend vulnerability.

Definition 3 (g-vulnerability)**.** *Given gain function g and prior π, the prior g-vulnerability is*

$$V_g(\pi) = \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] g(w, x)$$

In this case the attacker does not necessarily pick the most likely value, but the one that maximizes his gain. Once the attacker observes the output and obtains an updated *posterior distribution*, we are interested in re-measuring the vulnerability.

Definition 4 (Posterior g-vulnerability)**.** *Given gain function g, a prior π and a channel C, the posterior g-vulnerability is*

$$V_g(\pi, C) = \sum_{z \in \mathcal{Z}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] C[x, z] g(w, x)$$

In the next chapter we introduce a measure of privacy used in location privacy, ADVERROR2.3, that is indeed equivalent to posterior g-vulnerability.

We can now compute the information leakage of the channel and its capacity, that is its maximum leakage over all possible priors.

Definition 5 (g-leakage and g-capacity)**.** *Given gain function g, a prior π and a channel C, we define*

$$\mathcal{L}_g(\pi, C) = H_g(\pi) - H_g(\pi, C) = \log_2 \tfrac{V_g(\pi, C)}{V_g(\pi)}$$

$$\mathcal{ML}_g(C) = \max_\pi \ \mathcal{L}_g(\pi, C)$$

The importance of leakage lies in the fact that it characterizes a property of the channel, not of the prior. Intuitively instead of measuring the vulnerability of the secret before and after the use of the system, it gives a measure of how much vulnerability is introduce by the system itself. In a case where posterior vulnerability is high for example, it could be the case that the mechanism is leaking information. However it could also be the case that the prior vulnerability was high to begin with and the mechanism is not actually leaking at all. Leakage represents a quality of the system and not of the secret, like vulnerability. Capacity represents the worst possible leakage that the system may cause with any prior.

In order to recover min-vulnerability (min-leakage and min-capacity) it is sufficient to use the identity gain function. A function that rewards only a guess that is exactly the secret.

Definition 6 (Identity gain function)**.** *The identity gain function:* $g_{id} : \mathcal{X} \times \mathcal{Z} \to [0, 1]$ *is given by*

$$g_{id}(w, x) = \begin{cases} 1 & \textit{if}\, w = x \\ 0 & \textit{if}\, w \neq x \end{cases}$$

2.3.3 Relation with Differential Privacy

Quantitative Information Flow is an alternative model to describe information hiding and leakage. The field of privacy inherently describes a leakage of information, given that to obtain some utility from the system we need to extract some information, and QIF offers a new way to measure it. At the same time our goal is to bound this leakage so to avoid learning sensitive information and techniques have been developed to achieve this, including differential privacy. However differential privacy, doesn't consider any prior information, but simply imposes a property of indistinguishability over the observables. We might wonder if and what is the relationship between these two fields which provide two different quantitative measures of leakage. Indeed in a number of cases it is possible to prove that a differentially private mechanism provides an upper bound on the min-capacity [AACP11, ECP14]. More on the advantages of considering both approaches has been recently explored by Shokri in [Sho15].

2.4 Location Privacy

We now present the main topic of this thesis, location privacy, we start by introducing the location data that is typically collected and then the attacks that have been developed in the literature on this data. We discuss only the attacks that are relevant to our work, for a complete overview there are excellent surveys [Ter11, Kru09, SJCH12] that present the threats, methods, and guarantees. We then discuss how the definitions introduced for databases can be applied in this setting.

2.4.1 Location Data

The data that are usually considered in this context consist in a pair of location coordinates and a time stamp. While the temporal information produced by the source devices are very accurate, the location can vary considerably in precision.

Civilian *GPS* sensors provide a precision in the order of tens of meters and in presence of trees or buildings the accuracy rapidly decreases.

Alternatively mobile phones can fall back to the *cellular network* cell id in order to be localized. The precision in this case depends on the cell size which varies with the density of the population. Usually denser areas are covered by a larger number of cell towers, each responsible for a smaller geographical area (around 100 meters), while in sparsely populated areas cells can be several hundreds of meters in radius.

In modern, internet connected devices, an alternative high precision source of location data are *wifi networks*. A device sends a list of visible wifi SSID to a location service such as Google Location Services that matches it against a database of geolocated wifi access points and returns the user's location. In this case the location, especially indoor, can be very accurate i.e. less than 10 meters. When examining this scenario we consider the service location provider to be trusted, the threat comes from subsequent usage of the obtained location.

As a last resort the user is located from his IP address, the accuracy in this case varies greatly, much like for the cellular network, but it is generally possible to establish the city of origin or even the neighborhood.

Given the continuous and inaccurate nature of location data, often a level of *accuracy* is indicated together with the location coordinates. For example in the Web Location API the `coordinates` object contains an `accuracy` attribute. In the rest of the text when referring to an "exact" location, we intend a location with an error of less than 10 meters.

Together with the coordinates and time stamp, the data can contain a user identifier and can be collected as a continuous trace. A *location trace* is sometimes referred to as a mobility trace or trajectory.

2.4.2 Attacks

There are a number of inference attacks that can be considered in this context, depending on the attacker goals and knowledge.

The most basic attack is user *localization*, where the attacker tries to find the user's exact location at a specific time. Despite its simplicity a large number of sensitive information can be inferred from one's location and this remains the main focus of this thesis.

Instead of any position of the user, we might be interested in discovering only his *important places*, such as home, work, the gym he attends every week or the

shop where he buys groceries. Even from a single location the attacker may try to re-map it (if it is not already accurate enough) to a relevant nearby Point Of Interest. However more often the attack is performed on several locations through which a *mobility model* of the user is built.

By far the most studied scenario is *de-anonymization* [DMDBP08, MYYR10, ZB11, GKdPC13a, GG03, MCA06]. If the mobility model of the user is accurate enough, very often, the attacker is able to recover the user's identity even though the personal identifiers have been removed from the traces. In [dMHVB13] the authors show how as few as 3 points are sufficient for the attack and with coarse location data from the cellular network.

In the presence of multiple users a number of *co-location* or meeting attacks can be performed to find locations where two or more users have met [OHSH14]. This reveals both that there is a link between the users and possibly the nature of the relationship thanks to the location information.

Several of these attacks are evaluated in [GKdPC10, GKdPC11, PMLB14] over large data sets of users' traces, showing how effective these techniques can be in extracting private information.

In this thesis we don't try to hide the user's identity, we are interested in protecting what can be derived directly from the location data. For this reason we focus mostly on the localization and important places attacks. It should be noted however that the obfuscation techniques that we employ have an averse effects on the other attacks as well, although the measurement of these effects are left as future work.

We now revisit the techniques presented for privacy protection in databases in the context of location privacy.

2.4.3 k-anonymity

In the context of location privacy k-anonymity was adapted as definition of privacy [GG03] and started a large body of work to improve the effectiveness of the sanitization mechanisms [GL08, MCA06, TLM09, XC09], with also an extension to work on traces [BWJ05]. In order to protect a user's location privacy, each of her queries must be indistinguishable from those of at least k other users. The generalization in this context consist in approximating locations and time stamps to their location-area and time-window. This could be achieved either by adding *dummy locations* to the query [KYS05, SGI09], or by creating a *cloaking region* including k locations with some semantic property, and querying the service

provider for that cloaking region [BLPW08, DK05, XKP09].

Again the definition suffers from the background knowledge of the attacker, that may rule out some of the k location lowering the privacy provided. We refer to [STD+10] for a more detailed description of the shortcomings of k-anonymity in the context of location privacy.

For the sake of comparison with the work developed in this thesis it should be noted that we focus solely on location privacy, while k-anonymity tries also to achieve query anonymity. Ideas inspired by the field of k-anonymity are incorporated in the our elastic mechanism presented in Section 5.2.1 and in future work we plan to extend them to l-diversity.

2.4.4 Differential Privacy

Notions that abstract from the attacker's knowledge based on differential privacy can be found in [MKA+08] and [HR11] although only for *aggregate* information. In order to make it suitable for applications in which only a single individual is involved [Dew12] proposes a mix of differential privacy and k-anonymity, by fixing an anonymity set of k locations and requiring that the probability to report the same obfuscated location z from any of these k locations should be similar.

The notion we based our work on, geo-indistinguishability [ABCP13], was largely influenced by differential privacy or rather by its generalization $d_{\mathcal{X}}$-privacy [CABP13]. Both notions are described in detail in the next Chapter 3.1 and offer the desired property of abstracting from the attacker's prior knowledge. For this reason they are more robust and are suitable for scenarios where the prior is unknown, or the same mechanism must be used for multiple users. Privacy is achieved though the addition of random noise, thus *obfuscating* the location of the user.

2.4.5 Bayesian metrics for Location Privacy

A prolific branch of research in location privacy has been developed at EPFL [STD+11, STT+12, STBH11, OHSH14, TST+14] where a Bayesian metric to define and evaluate location privacy was developed, together with a technique to compute optimal mechanisms. This metric considers a Bayesian adversary that has a prior knowledge π of the possible location of the user and obsers the output of the mechanism K. After seeing a reported location z the attacker applies a strategy $h : \mathcal{Z} \to \mathcal{X}$ to remap z to the real secret location where he believes the

user could be, e.g. if z is in a river, it is likely that the user is actually in a nearby location x on the banks.

Much like g-vulnerability the different attacks are modeled using different *loss* functions, measuring the loss of the attacker in this case. The ADVERROR metric measures the expected loss of an attacker trying to infer the user's location. It is defined as:

$$\text{ADVERROR}(K, \pi, h, d_A) = \sum_{x,z} \pi(x)\ K(x)(z)\ d_A(x, h(z))$$

where d_A is a metric modeling the adversary's loss in case he fails to identify the user's real location. Note how the loss function is not applied directly to the reported location z, but to the remapped location $h(z)$. A rational adversary will use the strategy h^* minimizing his error, hence [STBH11] proposes to use $\text{ADVERROR}(K, \pi, h^*, d_A)$ as a privacy metric, where

$$h^* = \arg\min_h\ \text{ADVERROR}(K, \pi, h, d_A)$$

Note that h^* can be computed efficiently using the techniques described in [STT+12].

Although this metric was independently developed, it is indeed equivalent to g-vulnerability, making it an ideal candidate to test our experimental result as shown in Section 5.3. More on the relation between adversarial error (or g-vulnerability) and differential privacy for location privacy can be found in [Sho15].

For our purposes adversarial error is used as independent measure of privacy when comparing our elastic mechanism with respect to the Planar Laplace in Chapter 5.

Optimal Mechanism

Shokri et al. [STT+12] also define a technique that, given the attacker prior knowledge and a utility constraint, is able to compute the mechanism that provides optimal privacy: a mechanism that has the highest ADVERROR among those achieving the required utility.

Despite the numerous advantages of this technique, it should be noted that using ADVERROR as a design principle for the mechanism leads to a privacy protection that is dependent on a particular prior. Like for k-anonymity we have a privacy guarantee that is valid only for a specific background knowledge of the attacker.

A similar technique that overcomes this limitation was developed by Bordenabe et al. in [BCP14], where they develop a mechanism that optimizes utility while respecting the constraints of differential privacy.

2.4.6 Others metrics

A popular technique to reduce the accuracy of reported location is the use of *cloaking* mechanism [CZBP06]. The degree of privacy is measured by the size of the cloak (also called uncertainty region), and by the coverage of regions that the user consider sensitive. [ACD+07] proposes different obfuscation techniques to increase the level of inaccuracy already present in the data, due to the specific sensing technology. This level of privacy is defined as the ratio between the accuracy before and after the application of the obfuscation techniques. Both these techniques are dependent on the adversary's side information.

Chapter 3

Preliminaries

3.1 Location Privacy through reduced Accuracy

Rarely we are interested in hiding our geographical location per se, more commonly we consider our location sensitive because of the many personal details it can indirectly reveal about us. Reporting that we are in a hospital may induce the attacker to think that we are either sick or visiting somebody that is. Visiting a church may reveal our religious belief, a shop our shopping habits and possibly something about our wealth.

However the confidence in the inference highly depends on the accuracy of the observed location. Modern GPS sensors have an accuracy of around 10 meters for civilian purposes, even less if we consider obstacles such as trees or buildings, while cell towers have an even coarser coverage. From these considerations an intuitive notion of privacy arise, that is reducing the accuracy of our reported locations to an area that allows us to *hide in a crowd*. If the area is rich in people and points of interest, the attacker inference will be more limited. Furthermore given that location data has historically been inaccurate to some degree, LBSs take accuracy into consideration e.g. the Web Geolocation API expects the user latitude longitude and accuracy. For this reason reducing the *accuracy*, through perturbation or cloaking, is an effective technique for reporting a location that is at the same time meaningful and decoupled from its sensitive semantic value.

However in the case of multiple releases over time, the correlation and timestamp offer more insight to the attacker. If a user reports every day from the same shop, she is probably working there. If she reports the same restaurant everyday, maybe she works nearby and so on. In this case with many sporadic releases it is possible to build an accurate profile of the user with her home and work places,

favorite shops and restaurants, sport activities and social behavior. This is the main setting where we place the work developed in this thesis: a user moving in an urban environment and accessing a LBS several times a day. The user can be identified with the service but wishes to protect her location, for example from behavioral advertising.

Many other attacks are possible on location data [GKdPC13b, dMHVB13], such as de-anonymization, co-location with other individuals, prediction of future movements. Despite our privacy protection is not directly aimed at these attacks, the obfuscation we apply can help to reduce their effectiveness. However an evaluation for this broader class of attacks is left as future work.

3.2 Privacy Definitions

Probabilistic model. We first introduce a simple model used in the rest of the thesis. We start with a set $\mathcal{X}$ of *secrets* and $\mathcal{Z}$ of possible *reported values*. For the needs of this work we consider $\mathcal{X}$ to be finite. The selection of a reported value $z \in \mathcal{Z}$ is *probabilistic* and the set of probability distributions over $\mathcal{Z}$ is denoted by $\mathcal{P}(\mathcal{Z})$. A *mechanism* is a probabilistic function $K : \mathcal{X} \to \mathcal{P}(\mathcal{Z})$ assigning to each secret $x \in \mathcal{X}$ a probability distribution on $\mathcal{Z}$, where $K(x)(z)$ is the probability to report $z \subseteq \mathcal{Z}$, when the user's secret is x.

3.2.1 $d_{\mathcal{X}}$-privacy.

The privacy definitions used in this thesis are based on a generalized variant of differential privacy that can be defined on an arbitrary set of secrets $\mathcal{X}$ (not necessarily on databases), equipped with a metric $d_{\mathcal{X}}$ [RP10, CABP13]. The distance $d_{\mathcal{X}}(x, x')$ expresses the *distinguishability level* between the secrets x and x', modeling the privacy notion that we want to achieve. A small value denotes that the secrets should remain indistinguishable, while a large value means that we allow the adversary to distinguish them.

Given that the mechanisms we are interested in are probabilistic, in order to measure the distinguishability of their outputs, we introduce a metric on probability distributions.

Definition 7 (Multiplicative distance)**.** *Given two probability distributions* $\mu_1, \mu_2 \in \mathcal{P}(\mathcal{Z})$*, their* multiplicative distance *is*

$$d_{\mathcal{P}}(\mu_1, \mu_2) = \sup_{Z \subseteq \mathcal{Z}} |\ln \tfrac{\mu_1(Z)}{\mu_2(Z)}|$$

with the convention that $|\ln \frac{\mu_1(Z)}{\mu_2(Z)}| = 0$ *if both* $\mu_1(Z), \mu_2(Z)$ *are zero and* ∞ *if only one of them is zero.*

Intuitively $d_{\mathcal{P}}(\mu_1, \mu_2)$ is small if μ_1, μ_2 assign similar probabilities to each reported value.

The generalized variant of differential privacy, called $d_{\mathcal{X}}$-privacy, is defined as follows:

Definition 8 ($d_{\mathcal{X}}$-privacy)**.** *A mechanism* $K : \mathcal{X} \to \mathcal{P}(\mathcal{Z})$ *satisfies* $d_{\mathcal{X}}$*-privacy iff:*

$$d_{\mathcal{P}}(K(x), K(x')) \leq d_{\mathcal{X}}(x, x') \qquad \forall x, x' \in \mathcal{X}$$

The main idea behind this notion is that it forces the output of the mechanism applied on x, x', i.e. the distributions $K(x), K(x')$, to be similar when x and x' are close with respect to $d_{\mathcal{X}}$, preventing an adversary from distinguishing them, while it relaxes the constraint when x, x' are far away from each other. In the case of location privacy for example the service provider is allowed to distinguish points in Paris from those in London, but not among points in Paris.

Different choices of $d_{\mathcal{X}}$ give rise to different privacy notions; it is also common to scale our metric of interest by a privacy parameter ϵ (note that $\epsilon d_{\mathcal{X}}$ is itself a metric).

$d_{\mathcal{X}}$-privacy can be equivalently expressed in a form more similar to standard differential privacy Def. 1:

$$K(x)(Z) \leq e^{d_{\mathcal{X}}(x,x')} K(x')(Z) \quad \forall x, x' \in \mathcal{X}, Z \subseteq \mathcal{Z}$$

Furthermore standard differential privacy simply corresponds to $\epsilon d_h(x, x')$-privacy, where d_h is the Hamming distance between databases x, x', i.e. the number of individuals in which they differ. If x, x' are *adjacent*, i.e. they differ in a single individual, then $d_h(x, x') = 1$ and the distinguishability level between such databases is exactly ϵ.

3.2.2 Geo-indistinguishability

In the case of location privacy, the secrets $\mathcal{X}$ as well as the reported values $\mathcal{Z}$ are sets of locations (i.e. subsets of $\mathbb{R}^2$), while K is an obfuscation mechanism.

Using the Euclidean metric d_2, we obtain ϵd_2-privacy, a natural notion of location privacy called geo-indistinguishability and first introduced in [ABCP13]. This privacy definition requires that the closer (geographically) two locations are, the more similar the probability of producing the same reported location z should be. As a consequence, the service provider is not allowed to infer the user's location with accuracy, but he can get approximate information required to provide the service.

Let $d_2(\cdot, \cdot)$ denote the Euclidean metric; a mechanism K satisfies ϵ-geo-indistinguishability iff for all x, x':

$$d_{\mathcal{P}}(K(x), K(x')) \leq \epsilon d_2(x, x')$$

The quantity $\epsilon d_2(x, x')$ is the *distinguishability level* between the secrets x and x'. The use of the Euclidean metric d_2 is natural for location privacy: the *closer* (geographically) two points are, the *less distinguishable* we would like them to be. Two characterization results are also given in [ABCP13], providing intuitive interpretations of geo-indistinguishability.

Seeing it from a slightly different viewpoint, this notion offers privacy *within any radius* r from the user, with a level of distinguishability ϵr, proportional to r. Hence, within a small radius the user enjoys strong privacy, while his privacy decreases as r gets larger. This gives us the flexibility to adjust the definition to a particular application: typically we start with a radius r^* for which we want strong privacy, which can range from a few meters to several kilometers (of course a larger radius will lead to more noise). For this radius we pick a relatively small ϵ^* (for instance in the range from $\ln 2$ to $\ln 10$), and set $\epsilon = \epsilon^*/r^*$.

3.2.3 Distinguishability level and ϵ

A point worth emphasizing is the role of the distinguishability level, and its relationship to ϵ in each definition. The distinguishability level between two secrets x, x' is their *distance* in the privacy metric employed, that is $\epsilon d_h(x, x')$ for differential privacy, $\epsilon d_2(x, x')$ for geo-indistinguishability, and $d_{\mathcal{X}}(x, x')$ for $d_{\mathcal{X}}$-privacy. Secrets that are assigned a "small" distinguishability level will remain indistinguishable, providing privacy, while secrets with a large distance are allowed to be distinguished in order to learn something from the system and provide utility. Typical values that are considered "small" range from 0.01 to $\ln 4$; we denote a small distinguishability level by l^*.

In the case of differential privacy, ϵ is exactly the distinguishability level between adjacent databases (since $d_h(x, x') = 1$ for such databases), hence we directly use our "small" level l^* for ϵ. For geo-indistinguishability, however, ϵ represents the distinguishability level for points such that $d_2(x, x') = 1$; however, depending on the unit of measurement as well as on the application at hand, we might or might not want to distinguish points at a unit of distance. To choose ϵ in this case, we start by defining a radius r^* of *high protection*, say $r^* = 300$ meters for an LBS application within a big city, and we set $\epsilon = l^*/r^*$. As a consequence, points within r^* from each other will have distinguishability level at most l^*, hence an adversary will be unable to distinguish them, while points will become increasingly distinguishable as the geographic distance between them increases.

Note the difference between $d_2(x, x')$, the geographic distance between x, x', and $\epsilon d_2(x, x')$, the distinguishability level between x, x'. In other words, ϵ compresses the Euclidean distance turning it into a distinguishability metric. To avoid confusion, throughout the text we use r to denote geographical distances and radii, and l to denote distinguishability levels.

If we want to *hide in a crowd* in a large urban environment, using $r^* = 300$ m is enough to contain a large number of shops, services and people so to limit the power of inference of the attacker. In the same way if we are in a less dense environment, like small town with residential areas and countryside, we need to increase r^*. However it should be noted that we cannot simply *change* r^* during the life of the system depending on the location of the user, as we would be changing our observable based on the secret. As a consequence, if the mechanism is used in both setting we are faced with the hard choice of picking one of the two configurations, one that will be always private and lead to bad utility in the city, or one that will not be private enough in the suburb. We address this issue more closely with the elastic mechanism in Chapter 5.

3.2.4 Traces

Having established a privacy notion for single locations, it is natural to extend it to location *traces* (sometimes called *trajectories* in the literature). Although location privacy is our main interest, this can be done for traces having any secrets with a corresponding metric as elements. We denote by $\mathbf{x} = [x_1, \ldots, x_n]$ a trace, by $\mathbf{x}[i]$ the i-th element of $\mathbf{x}$, by $[\,]$ the empty trace and by $x :: \mathbf{x}$ the trace obtained by adding x to the head of $\mathbf{x}$. We also define $\texttt{tail}(x :: \mathbf{x}) = \mathbf{x}$.

To obtain a privacy notion, we need to define an appropriate metric between

traces and a natural choice is the maximum metric.

Definition 9 (Maximum metric)**.** *Given a distinguishability metric $d_{\mathcal{X}}$ on $\mathcal{X}$, we can define on $\mathcal{X}^n$ its maximal metric as*

$$d_\infty(\mathbf{x}, \mathbf{x}') = \max_i d_{\mathcal{X}}(\mathbf{x}[i], \mathbf{x}'[i])$$

This captures the idea that two traces are as distinguishable as their most distinguishable points. In terms of protection within a radius, if $\mathbf{x}$ is within a radius r from $\mathbf{x}'$ it means that $\mathbf{x}[i]$ is within a radius r from $\mathbf{x}'[i]$. Hence, ϵd_∞-privacy ensures that all secrets are protected within a radius r with the same distinguishability level ϵr.

3.3 Privacy Mechanisms

3.3.1 Laplace Mechanism

The classical mechanism used to provide differential privacy, introduced in [Dwo06], is the Laplace Mechanism. The Laplace mechanism $Lap(\epsilon)$ adds to the result of a numerical query noise sampled from a Laplace distribution scaled by ϵ and centered on the real answer.

Typically the noise needs to be also adapted to the *sensitivity* of the query, that is a measure of how much the query can increase distances between results. In other words how much the query increases the distinguishability of two secret databases.

Definition 10 (Sensitivity)**.** *Given a query $Q : \mathcal{D} \to \mathbb{R}$, the sensitivity of Q, denoted by Δ, is defined as:*

$$\Delta = \sup_{D \sim D' \in \mathcal{D}} |\, Q(D) - Q(D') \,|.$$

The probability density function of the resulting mechanism on a real answer y is:

$$Lap(\frac{\epsilon}{\Delta})(z) = \frac{\epsilon}{2\Delta} e^{-|y-z| \cdot \frac{\epsilon}{\Delta}}$$

This mechanism can be used in the context of $d_{\mathcal{X}}$-privacy by extending the notion of sensitivity to arbitrary metrics.

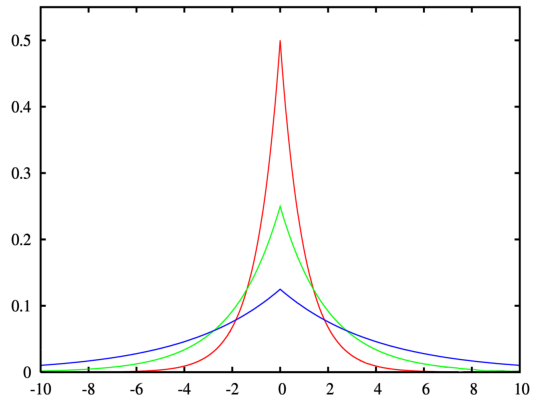


Figure 3.1: PDF of Lap machanism centered on a real answer 0, for different values of ϵ

3.3.2 Planar Laplace Mechanism

In the seminal paper on geo-indistinguishability [ABCP13], the authors propose also an efficient mechanism, called *Planar Laplace (PL)*, which is an extension for the Euclidean metric of Lap. When applied on location x, this mechanism draws a location z from the continuous plane with probability density function:

$$\frac{\epsilon}{2\pi} e^{-\epsilon d_2(x,z)}$$

In [ABCP13] a method to efficiently draw from this distribution is given, which uses polar coordinates and involves drawing an angle and a radius from a uniform and a gamma distribution respectively. The mechanism can be further discretized and truncated, and can be shown to satisfy ϵ-geo-indistinguishability.

The mechanism presents the advantages of being efficient to compute, making it suitable for low-end devices, and easy configurable with a single parameter ϵ which means that the same mechanism can be used by different users. For a fixed privacy parameter ϵ, utility is immediate to compute as the expected distance as it is independent from x (due to the symmetry of the continuous plane) and is given by $E_{\mathrm{PL}}[d_2] = 2/\epsilon$.

For these reasons the Planar Laplace was a prefect basis to develop Location Guard, a popular browser extension that provides geo-indistinguishability on top of the Geolocation API [1]. Location Guard is described in more detail in Chapter 6.

Furthermore the Planar Laplace is used as a component by the predictive

[1] http://www.w3.org/TR/geolocation-API/

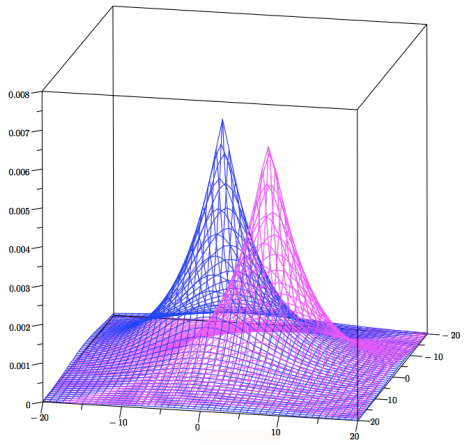


Figure 3.2: The pdf of two planar Laplacians, centered in $(-2, -4)$ and in $(5, 3)$, with $\epsilon = 0.2$.

mechanism introduced in Section 4.2.

3.3.3 Exponential Mechanism

In the case of an arbitrary distinguishability metric $d_{\mathcal{X}}$, a variant of the Exponential mechanism [MT07] can be employed. When applied at location x, this mechanism reports z with probability:

$$c_x e^{-\frac{1}{2} d_{\mathcal{X}}(x,z)} \quad \text{with} \quad c_x = \left(\sum_{z'} e^{-\frac{1}{2} d_{\mathcal{X}}(x,z')}\right)^{-1}$$

where c_x is a normalization factor. This mechanism can be shown to satisfy $d_{\mathcal{X}}$-privacy. Note the difference in the exponent between the two mechanisms: the Exponential mechanism has a factor $\frac{1}{2}$ missing from the Planar Laplace; in the proof of $d_{\mathcal{X}}$-privacy, this factor compensates for the fact that the normalization factor c_x is different for every x, in contrast to the Planar Laplace while the normalization factor $\frac{\epsilon}{2\pi}$ is independent from x. The advantage of this technique is the possibility of obtaining a privacy mechanism independently of the metric used, allowing us to focus solely on the metric design in Chapter 5.

3.3.4 Independent Mechanism

In the case of traces, in order to sanitize $\mathbf{x}$ we can simply apply a *noise mechanism* independently to each secret x_i. We assume that a family of noise mechanisms $N(\epsilon_N) : \mathcal{X} \to \mathcal{P}(\mathcal{Z})$ are available (such as the Planar Laplace or the exponential),

```
mechanism IM(x)
  z := []
  for i := 1 to |x|
    z := N(ε_N)(x[i])
    z := z :: z
  return z
```

Figure 3.3: Independent Mechanism

parametrized by $\epsilon_N > 0$, where each mechanism $N(\epsilon_N)$ satisfies ϵ_N-privacy. The resulting mechanism, called the *independent mechanism* $\mathrm{IM} : \mathcal{X}^n \to \mathcal{P}(\mathcal{Z}^n)$, is shown in Figure 3.3.

However, any obfuscation mechanism is bound to cause privacy loss when used repeatedly. In the case of a $d_{\mathcal{X}}$-private mechanism, applying it n times will satisfy $nd_{\mathcal{X}}$-privacy. This means that the distinguishability level between x, x' after n applications is $nd_{\mathcal{X}}(x, x')$; if $d_{\mathcal{X}}(x, x') > 0$ then as n grows x and x' are bound to become completely distinguishable. This is typical in the area of differential privacy, in which the global guarantee is $n\epsilon$ and it is thought as a budget which is consumed with every query (Section 2.2.1).

In Section 4.2 we go beyond the independent composition and we alleviate some of the budget accumulation thanks to a prediction function. Additionally, if we use a pseudo-metric such that $d_{\mathcal{X}}(x, x') = 0$, then x, x' are completely indistinguishable, and will remain so under any number of repetitions n. This property will be exploited by the "fence" technique of Section 4.4.

3.4 Utility

The goal of a privacy mechanism is not to hide completely the secret but to disclose enough information to be useful for some service while hiding the rest to protect the user's privacy. Typically these two requirements go in opposite directions: a stronger privacy level requires more noise which results in a lower utility.

Given the probabilistic nature of our mechanisms, one measure of their utility is the expected loss of a specific utility metric, chosen for the application. Additionally, given that the noise we use is unbounded, another useful measure of utility is the "worst" error, the maximum noise added with probability 0.9.

3.4.1 Expected Error

In order to evaluate and compare *general-purpose* location obfuscation mechanisms, the general principle is to report locations as close as possible to the original ones. However given that utility is very dependent on the specific application we are targeting, in the following we propose three metrics to cover the most common scenarios: euclidean, threshold and traces. Once we have a suitable utility metric d_u for our application, we measure utility as the *expected error* of mechanism K on location x as:

$$E[d_u] = \sum_x \pi(x) \sum_z K(x)(z)\ d_u(x, z)$$

Note that in general the expected error depends on the prior distribution π.

Euclidean d_2

In some applications, service quality degrades linearly as the reported location moves away from the real one; in such cases a natural and widely used choice [STBH11, STT+12, BCP14] is to define utility as the *geographical distance* d_2 between the actual and the reported locations.

Threshold metric d_r

Other applications tolerate a noise up to a certain threshold r with almost no effect on the service, but the quality drops sharply after this threshold. Imagine for example a weather forecast services, where utility remains unchanged even if the reported location is kilometers away from the real one. On the contrary a POI search application can tolerate lower noise addition in order to report meaningful results. In order to model these cases we also evaluate utility using a *threshold* metric that assign zero utility loss within a certain radius r and 1 outside.

$$d_r(x, z) = \begin{cases} 0 & d_2(x, z) < r \\ 1 & ow. \end{cases}$$

To continue the previous example we could use $r = 50$ km for the weather forecast application and $r = 0.5$ km for the POI search.

Trace metric $d_2(\mathbf{x}, \mathbf{z})$

In the case of traces we first need to define a notion of distance between a trace $\mathbf{x}$ and a sanitized trace $\mathbf{z}$. We do so by averaging the d_2 distance between each location in the trace and we overload the notation by using the same symbol but with bold arguments.

$$d_2(\mathbf{x}, \mathbf{z}) = \frac{1}{|\mathbf{x}|} \sum_i d_2(\mathbf{x}[i], \mathbf{z}[i]) \tag{3.1}$$

3.4.2 $\alpha(\delta)$-accuracy

Since typical noise mechanisms (such as Lap or PL) can return values at arbitrary distance from the original one, we have that the worst-case error is unbounded. Hence, we are usually interested in the 90-th percentile of the error, commonly expressed in the form of $\alpha(\delta)$-*accuracy* [RR10].

Definition 11 ($\alpha(\delta)$-accuracy)**.** *A mechanism K is $\alpha(\delta)$-accurate iff for all δ:*

$$Pr[d_2(x, z) \leq \alpha(\delta)] \geq \delta$$

In the rest of the text we will refer to $\alpha(0.9)$ (or simply α) as the "worst-case" error.

Chapter 4

Repeated Use over Time

4.1 Introduction

In the previous chapter we showed that geo-indistinguishability provides both a strong and intuitive concept of location privacy, together with an efficient and simple obfuscation mechanism to enforce it. A PLmechanism, configured to provide ϵ-geo-indistinguishability, will do so only for a *sporadic* use, in practice however, a user rarely performs a *single* location-based query. As a motivating example, we consider a user in a city performing different activities throughout the day: for instance he might have lunch, do some shopping, visit friends, etc. During these activities, the user performs several queries: searching for restaurants, getting driving directions, finding friends nearby, and so on. For each query, a new obfuscated location needs to be reported to the service provider, which can be easily obtained by independently adding noise at the moment when each query is executed. In order to independently apply noise we can use the *independent mechanism* 3.3.4. However, it is easy to see that privacy is degraded as the number of queries increases, due to the *correlation* between the locations. Intuitively, in the extreme case when the user never moves (i.e. there is perfect correlation), the reported locations are centered around the real one, completely revealing it as the number of queries increases. Technically, the independent mechanism applying ϵ-geo-indistinguishable noise to n location can be shown to satisfy $n\epsilon$-geo-indistinguishability [ABCP13]. This is typical in the area of differential privacy, in which ϵ is thought as a privacy *budget*, consumed by each query; this linear increase makes the mechanism applicable only when the number of queries remains small. Note that any obfuscation mechanism is bound to cause privacy loss when used repeatedly; geo-indistinguishability has the advantage of directly quantifying

this loss terms of the consumed budget.

The main idea behind the technique developed in this chapter is to actually use the correlation between locations in the trace to our advantage. Due to this correlation, we can often *predict* a point close to the user's actual location from information previously revealed. For instance, when the user performs multiple different queries from the same location - e.g. first asking for shops and later for restaurants - we could intuitively use the same reported location in all of them, instead of generating a new one each time. However, this implicitly reveals that the user is not moving, which violates geo-indistinguishability (nearby locations produce completely different observations); hence the decision to report the same location needs to be done in a private way.

Our main contribution is a *predictive mechanism* with three components: a *prediction function* Ω, a *noise mechanism* N and a *test mechanism* Θ. The mechanism behaves as follows: first, the list of previously reported locations (i.e. information which is already public) is given to the prediction function, which outputs a predicted location $\tilde{z}$. Then, it tests whether $\tilde{z}$ is within some threshold l from the user's current location using the test mechanism. The test itself should be private: nearby locations should pass the test with similar probabilities. If the test succeeds then $\tilde{z}$ is reported, otherwise a new reported location is generated using the noise mechanism.

The advantage of the predictive mechanism is that the budget is consumed only when the test or noise mechanisms are used. Hence, if the prediction rate is high, then we will only need to pay for the test, which can be substantially cheaper in terms of budget. The configuration of N and Θ is done via a *budget manager* which decides at each step how much budget to spend on each mechanism. The budget manager is also allowed to completely skip the test and blindly accept or reject the prediction, thus saving the corresponding budget. The flexibility of the budget manager allows for a dynamic behavior, constantly adapted to the mechanism's previous performance. We examine in detail two possible budget manager strategies, one maximizing utility under a fixed budget consumption rate and one doing the exact opposite, and explain in detail how they can be configured.

Note that, although we exploit correlation for efficiency, the predictive mechanism is shown to be private independently from the prior distribution on the set of traces. If the prior presents correlation, and the prediction function takes advantage of it, the mechanism can achieve a good budget consumption rate, which translates either to better utility or to a greater number of reported points than the independent mechanism. If there is no correlation, or the prediction does not take

advantage of it, then the budget consumption can be worse than the independent mechanism. Still, thanks to the arbitrary choice of the prediction function and the budget manager, the predictive mechanism is a powerful tool that can be adapted to a variety of practical scenarios.

We experimentally verify the effectiveness of the mechanism on our motivating example of a user performing various activities in a city, using two large data sets of GPS trajectories in the Beijing urban area ([ZXM10, YZZ$^+$10]). Geolife [ZXM10] collects the movements of several users, using a variety of transportation means, including walking, while in Tdrive [YZZ$^+$10] we find exclusively taxi drivers trajectories. The results for both budget managers, with and without the skip strategy, show considerable improvements with respect to independently applied noise. More specifically, we are able to decrease average error up to 40% and budget consumption rate up to 64%. The improvements are significative enough to broaden the applicability of geo-indistinguishability to cases impossible before: in our experiments we cover 30 queries with reasonable error which is enough for a full day of usage; alternatively we can drive the error down from 5 km to 3 km, which make it acceptable for a variety of application.

Despite the improvement in budget consumption that the predictive mechanism provides, in frequently disclosed locations such as home or work, the reiterate use of the mechanism over months is bound to reveal their exact position. A solution commonly employed in practice consists in building a "fence" around sensitive locations so that all points inside are completely indistinguishable from each other. In this way the attacker will be able, after many iterations, to identify the fence but not the exact location inside the fence. In a sense, instead of having our sensitive locations exactly determined during the use of the mechanism, we rather declare publicly beforehand their approximate position. We show that such a solution can be elegantly expressed in the distinguishability metric $d_{\mathcal{X}}$, and can be easily incorporated in other metrics we may decide to use outside the fence. The fenced metric effectively stops the linear growth of the privacy budget in frequently recurrent locations, remaining compatible with the predictive mechanism that cover the other movements of the user.

Note that both techniques can be efficiently implemented on the user's phone, and do not require any modification on the side of the provider, hence they can be seamlessly integrated with existing LBSs.

Contributions. This Chapter contributions are the following:

- We propose a predictive mechanism that exploits correlations on the input

by means of a prediction function.
- We show that the proposed mechanism is private and provide a bound on its utility.
- We instantiate the predictive mechanism for location privacy, defining a prediction function and two budget managers, optimizing utility and budget consumption rate.
- We evaluate the mechanism on two large sets of GPS trajectories and confirm our design goals, showing substantial improvements compared to independent noise.
- We show that the technique of geo-fences can be expressed as a distinguishability metric.

Plan of the chapter. In the next Section 4.2 we present in detail the components of the predictive mechanism, including budget managers and skip strategies, together with the main results of privacy and utility. In Section 4.3 we apply the predictive mechanism to location privacy, defining a prediction function, skip strategies and detailed configurations of the budget manager. In Section 4.3.4 we describe the experiments and their results. Finally in Section 4.4 we show how to model geographical fences with a metric.

A final note on the generality of out method, the work presented in this chapter started with the objective to extend the use of geo-indistinguishability to location traces with a more efficient use of the budget but thanks to the generality of the approach it developed into a viable mechanism for other domains in the same family of metric based $d_{\mathcal{X}}$-privacy. It is indeed the major focus of our future work to apply this technique in new fields such as smart meters and back in the standard domain of differential privacy, statistical databases. Although our main motivation is location privacy, the mechanism can work for traces of any secrets $\mathcal{X}$, equipped with a metric $d_{\mathcal{X}}$.

4.2 A Predictive $d_{\mathcal{X}}$-private Mechanism

The fundamental intuition of our work is that the presence of correlation on the secret can be exploited to the advantage of the mechanism. A simple way of doing this is to try to predict new secrets from past information; if the secret can be predicted with enough accuracy it is called *easy*; in this case the prediction can

be reported without adding new noise. One the other hand, *hard* secrets, that is those that cannot be predicted, are sanitized with new noise. Note the difference with the independent mechanism where each secret is treated independently from the others.

Let $\mathcal{B} = \{0, 1\}$. A boolean $b \in \mathcal{B}$ denotes whether a point is easy (0) or hard (1). A sequence $\mathbf{r} = [z_1, b_1, \ldots, z_n, b_n]$ of reported values and booleans is called a *run*; the set of all runs is denoted by $\mathcal{R} = (\mathcal{Z} \times \mathcal{B})^*$. A run will be the output of our predictive mechanism; note that the booleans b_i are considered public and will be reported by the mechanism.

Main components

The predictive mechanism has three main components: first, the *prediction* is a deterministic function $\Omega : \mathcal{R} \to \mathcal{Z}$, taking as input the run reported up to this moment and trying to predict the next *reported value*. The output of the prediction function is denoted by $\tilde{z} = \Omega(\mathbf{r})$. Note that, although it is natural to think of Ω as trying to predict the secret, in fact what we are trying to predict is the reported value. In the case of location privacy, for instance, we want to predict a reported location at acceptable distance from the actual one. Thus, the possibility of a successful prediction should not be viewed as a privacy violation.

Second, a *test* is a family of mechanisms $\Theta(\epsilon_\theta, l, \tilde{z}) : \mathcal{X} \to \mathcal{P}(\mathcal{B})$, parametrized by $\epsilon_\theta, l, \tilde{z}$. The test takes as input the secret x and reports whether the prediction $\tilde{z}$ is acceptable or not for this secret. If the test is successful then the prediction will be used instead of generating new noise. The purpose of the test is to guarantee a certain level of utility: predictions that are farther than the threshold l should be rejected. Since the test is accessing the secret, it should be private itself, where ϵ_θ is the budget that is allowed to be spent for testing.

The test mechanism that will be used throughout the chapter is the one below, which is based on adding Laplace noise to the threshold l:

$$\Theta(\epsilon_\theta, l, \tilde{z})(x) = \begin{cases} 0 & \text{if } d_{\mathcal{X}}(x, \tilde{z}) \leq l + Lap(\epsilon_\theta) \\ 1 & \text{ow.} \end{cases} \tag{4.1}$$

The test is defined for all $\epsilon_\theta > 0, l \in [0, +\infty), \tilde{z} \in \mathcal{Z}$, and can be used for any metric $d_{\mathcal{X}}$, as long as the domain of reported values is the same as the one of the secrets (which is the case for location obfuscation) so that $d_{\mathcal{X}}(x, \tilde{z})$ is well defined.

Finally, a *noise mechanism* is a family of mechanisms $N(\epsilon_N) : \mathcal{X} \to \mathcal{P}(\mathcal{Z})$,

parametrized by the available budget ϵ_N. The noise mechanism is used for hard secrets that cannot be predicted.

4.2.1 Budget management

The parameters of the mechanism's components need to be configured at each step. This can be done in a dynamic way using the concept of a *budget manager*. A budget manager β is a function that takes as input the run produced so far and returns the budget and the threshold to be used for the test at this step as well as the budget for the noise mechanism: $\beta(\mathbf{r}) = (\epsilon_\theta, \epsilon_N, l)$. We will also use β_θ and β_N as shorthands to get just the first or the second element of the result.

Of course the amount of budget used for the test should always be less than the amount devoted to the noise, otherwise it would be more convenient to just use the independent noise mechanism. Still, there is great flexibility in configuring the various parameters and several strategies can be implemented in terms of a budget manager. In this work we fix the level of privacy guaranteed, as it is our priority, and for predictable traces the budget manager will improve the utility, in terms of average error or budget consumption rate.

In the next section we will discuss two possible budget management policies, one maximizing utility under a fixed budget consumption rate and one doing the exact opposite.

All the components are defined here with the minimal information needed for their function, consider though that all of them could access additional public information, for example we may want to enrich the prediction function for a database with common statistics of a population or in geolocalization with maps of the territory.

4.2.2 The mechanism

We are now ready to fully describe our mechanism. A single step of the predictive mechanism, displayed in Figure 4.1b, is a family of mechanisms $\mathrm{Step}(\mathbf{r}) : \mathcal{X} \to \mathcal{P}(\mathcal{Z} \times \mathcal{B})$, parametrized by the run $\mathbf{r}$ reported up to this point. The mechanism takes a secret x and returns a reported value z, as well as a boolean b denoting whether the secret was easy or hard. First, the mechanism obtains the various configuration parameters from the budget manager as well as a prediction $\tilde{z}$. Then the prediction is tested using the test mechanism. If the test is successful the prediction is returned, otherwise a new reported value is generated using the noise

```
mechanism PM(x)
  r := []
  for i := 1 to |x|
    (z, b) := Step(r)(x[i])
    r := (z, b) :: r
  return r
```

(a) Predictive Mechanism

```
mechanism Step(r)(x)
  (ε_θ, ε_N, l) := β(r)
  z̃ := Ω(r)
  b := Θ(ε_θ, l, z̃)(x)
  if b == 0 then z := z̃
  else z := N(ε_N)(x)
  return (z, b)
```

(b) Single step of the Predictive Mechanism

mechanism.

Finally, the predictive mechanism, displayed in Figure 4.1a, is a mechanism $\mathrm{PM} : \mathcal{X}^n \to \mathcal{P}(\mathcal{R})$. It takes as input a trace $\mathbf{x}$, and applies $\mathrm{Step}(\mathbf{r})$ to each secret, while extending at each step the run $\mathbf{r}$ with the new reported values (z, b).

Note that an important advantage of the mechanism is that it is *online*, that is the sanitization of each secret does not depend on future secrets. This means that the user can query at any time during the life of the system, as opposed to *offline* mechanisms were all the queries need to be asked before the sanitization. Furthermore the mechanism is *dynamic*, in the sense that the secret can change over time (e.g. the position of the user) contrary to static mechanism where the secret is fixed (e.g. a static database).

It should be also noted that, 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 be 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. We are currently investigating proper assumptions under which the budget can be reset while satisfying a formal privacy guarantee. 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.

4.2.3 Privacy

We now proceed to show that the predictive mechanism described in the previous section is $d_{\mathcal{X}}$-private. The privacy of the predictive mechanism depends on that of its components. In the following, we assume that each member of the families of

test and noise mechanisms is $d_{\mathcal{X}}$-private for the corresponding privacy parameter:

$$\forall \epsilon_\theta, l, \tilde{z}.\ \Theta(\epsilon_\theta, l, \tilde{z}) \text{ is } \epsilon_\theta d_{\mathcal{X}}\text{-private} \tag{4.2}$$

$$\forall \epsilon_N. \qquad N(\epsilon_N) \qquad \text{is } \epsilon_N d_{\mathcal{X}}\text{-private} \tag{4.3}$$

In the case of the test $\Theta(\epsilon_\theta, l, \tilde{z})$ defined in (4.1), we can show that it is indeed $d_{\mathcal{X}}$-private, independently of the metric or threshold used.

Fact 1 (Privacy of Test function)**.** *The family of test mechanisms* $\Theta(\epsilon_\theta, l, \tilde{z})$ *defined by* (4.1) *satisfies assumption 4.2.*

Proof. We use the fact that Laplacian noise scaled by ϵ_θ is ϵ_θ-d.p. We assume that Lap is scaled with ϵ_θ so we omit it in the following.

$$\begin{array}{ll}
P[d(x,\tilde{z}) \leq l + Lap(0)] = & \text{translating the noise} \\
P[Lap(d(x,\tilde{z}) - l) \leq 0] = & l \text{ and } \tilde{z} \text{ are constants} \\
P[Lap(t) \leq 0] \leq & \text{assumption on Lap} \\
e^{\epsilon_\theta \cdot d(t,t')} P[Lap(t') \leq 0] \leq & \text{using } 4.4 \\
e^{\epsilon_\theta \cdot d(x,x')} P[Lap(t') \leq 0] = & \text{translating back} \\
e^{\epsilon_\theta d(x,x')} P[d(x',\tilde{z}) \leq l + Lap(0)] &
\end{array}$$

$$\begin{aligned}
d(t,t') = \ & |d(x,\tilde{z}) - l - d(x',\tilde{z}) + l| \leq \\
& |d(x,x') + d(x',\tilde{z}) - d(x',\tilde{z})| \ = d(x,x')
\end{aligned} \tag{4.4}$$

□

The global budget for a certain run $\mathbf{r}$ using a budget manager β is defined as:

$$\epsilon_\beta(\mathbf{r}) = \begin{cases} 0 & \text{if } |\mathbf{r}| = 0 \\ \beta_\theta(\mathbf{r}) + b(\mathbf{r}) \times \beta_N(\mathbf{r}) + \ \epsilon_\beta(\texttt{tail}(\mathbf{r})) & o.w. \end{cases} \tag{4.5}$$

As already discussed, a hard step is more expensive than an easy step because of the cost of the noise mechanism.

Building on the privacy properties of its components, we first show that the predictive mechanism satisfies a property similar to $d_{\mathcal{X}}$-privacy, with a parameter ϵ that depends on the run.

Lemma 1. *Under the assumptions* (4.2),(4.3)*, for the test and noise mechanisms, the predictive mechanism* PM*, using the budget manager* β*, satisfies*

$$\mathrm{PM}(\mathbf{x})(\mathbf{r}) \leq e^{\epsilon_\beta(\mathbf{r})\ d_\infty(\mathbf{x},\mathbf{x}')} \mathrm{PM}(\mathbf{x}')(\mathbf{r}) \qquad \forall \mathbf{r}, \mathbf{x}, \mathbf{x}' \tag{4.6}$$

Proof. We want to show that:

$$\forall \mathbf{x}, \mathbf{x}'. \quad P[\mathbf{r}|\mathbf{x}] \leq e^{\epsilon(\mathbf{r}) \cdot d(\mathbf{x}, \mathbf{x}')} P[\mathbf{r}|\mathbf{x}'] \tag{4.7}$$

In the following, we use the subscript i to indicate both a tuple from 0 to i such as $\mathbf{x}_i$ or the i-th element, such as x_i. Decomposing in *dependent* steps using the chain rule we obtain:

$$P[\mathbf{r}_i|\mathbf{x}_i] = P[(z_i, b_i)|\mathbf{x}_i, \mathbf{r}_{i-1}] \cdot P[\mathbf{r}_{i-1}|\mathbf{x}_{i-1}] \tag{4.8}$$

Analyzing the single step we have a binary choice between the easy case, which is deterministic, and the hard case, which is probabilistic. We introduce the random variable B_i to denote the outcome of the test at step i.

$$\begin{aligned}
&P[(z_i, b_i)|x_i, \mathbf{r}_{i-1}] = \\
&P[B_i = 1|x_i, \mathbf{r}_{i-1}] \cdot P[\Omega(\mathbf{r}_{i-1}) = z_i|\cancel{x_i}, \mathbf{r}_{i-1}] + \\
&P[B_i = 0|x_i, \mathbf{r}_{i-1}] \cdot P[N(x_i) = z_i|x_i, \cancel{\mathbf{r}_{i-1}}] = \\
&P[B_i = 1|x_i, \mathbf{r}_{i-1}] \cdot 1 + \\
&P[B_i = 0|x_i, \mathbf{r}_{i-1}] \cdot P[N(x_i) = z_i|x_i]
\end{aligned} \tag{4.9}$$

The composition of such steps forms a binary tree with all the possible runs that the test can produce; to treat this, we split the tree in traces $\bar{b}$ as they are disjoint events.

$$P[(\mathbf{z}, \mathbf{b})|\mathbf{x}] = P[(\mathbf{z}, \mathbf{b})|\mathbf{x}, \mathbf{b}] \cdot P[\mathbf{b}|\mathbf{x}]$$

Now that we know the trace, we reorganize the indexes of its steps in two groups, the easy $I_E = \{i \mid B_i = 1\}$ and hard steps $I_H = \{i \mid B_i = 0\}$. After having applied assumptions 4.2, 4.3 we can regroup the exponents and obtain a form close to 4.7. Here follows the complete proof:

$\forall n, \forall \mathbf{x}, \mathbf{x}'. \quad P[\mathbf{r}_n|\mathbf{x}_n] =$

$$
\begin{aligned}
\text{(chain rule)} \quad & = P[\mathbf{r}_n|\mathbf{r}_{n-1}, \mathbf{x}_n] \cdot P[\mathbf{r}_{n-1}|\mathbf{x}_n] \\
\text{(independence from } x_n\text{)} \quad & = P[\mathbf{r}_n|\mathbf{r}_{n-1}, \mathbf{x}_n] \cdot P[\mathbf{r}_{n-1}|\mathbf{x}_{n-1}] \\
\text{(iterating)} \quad & = \prod_{i=1}^{n} P[\mathbf{r}_i|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
\text{(chain rule)} \quad & = \prod_{i=1}^{n} P[z_i|\mathbf{z}_{i-1}, \mathbf{b}_i, \mathbf{x}_i] \cdot P[b_i|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
\text{(partitioning indexes)} \quad & = \prod_{i \in I_H(r)} P[z_i|\mathbf{z}_{i-1}, \mathbf{b}_i, \mathbf{x}_i] \cdot P[b_i|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
& \quad \prod_{i \in I_E(r)} P[z_i|\mathbf{z}_{i-1}, \mathbf{b}_i, \mathbf{x}_i] \cdot P[b_i|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
\text{(independences)} \quad & = \prod_{i \in I_H(r)} P[z_i|x_i] \cdot P[B_i = 0|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
& \quad \prod_{i \in I_E(r)} 1 \cdot P[B_i = 1|\mathbf{r}_{i-1}, \mathbf{x}_i] \\
\text{(assumptions 4.2, 4.3)} \quad & \leq \prod_{i \in I_H(r)} e^{\beta_N(\mathbf{r}_i) d(x,x')} \cdot P[z_i|x'_i] \cdot \\
& \quad e^{\beta_\theta(\mathbf{r}_i) d(x,x')} \cdot P[B_i = 0|\mathbf{r}_{i-1}, \mathbf{x}'_i] \\
& \quad \prod_{i \in I_E(r)} e^{\beta_\theta(\mathbf{r}_i) d(x,x')} \cdot P[B_i = 1|\mathbf{r}_{i-1}, \mathbf{x}'_i] \\
\text{(grouping exponents)} \quad & \leq e^{\epsilon(r)} \prod_{i \in I_H} P[z_i|x'_i] \cdot P[B_i = 0|\mathbf{r}_{i-1}, \mathbf{x}'_i] \\
& \quad \prod_{i \in I_E} P[B_i = 1|\mathbf{r}_{i-1}, \mathbf{x}'_i] \\
& = e^{\epsilon(\mathbf{r})} \cdot P[\mathbf{r}_n|\mathbf{x}'_n]
\end{aligned}
$$

With a global exponent for the run:

$$
\epsilon(\mathbf{r}) = \left(\sum_{i \in I_H(\mathbf{r})} \beta_N(\mathbf{r}_i) + \sum_{i \in I(\mathbf{r})} \beta_\theta(\mathbf{r}_i) \right) \cdot d_\infty(\mathbf{x}_n, \mathbf{x}'_n)
$$

□

This results shows that there is a difference between the budget spent on a "good" run, where the input has a considerable correlation, the prediction performs well and the majority of steps are easy, and a run with uncorrelated secrets, where any prediction is useless and all the steps are hard. In the latter case it is clear that our mechanism wastes part of its budget on tests that always fail,

performing worse than an independent mechanism.

Finally, the overall privacy of the mechanism will depend on the budget spent on the worst possible run.

Theorem 2 ($d_{\mathcal{X}}$-privacy)**.** *Under the assumptions* (4.2),(4.3)*, for the test and noise mechanisms, the predictive mechanism* PM*, using the budget manager* β*, satisfies* ϵd_∞*-privacy, with*

$$\epsilon = \sup_{\mathbf{r}} \ \epsilon_\beta(\mathbf{r})$$

Based on the above result, we will use ϵ*-bounded* budget managers, imposing an overall budget limit ϵ independently from the run. Such a budget manager provides a fixed privacy guarantee by sacrificing utility: in the case of a bad run it either needs to lower the budget spend per secret, leading to more noise, or to stop early, handling a smaller number of queries. In practice, however, using a prediction function tailored to a specific type of correlation we can achieve good efficiency. Moreover, we have the flexibility to use several prediction functions, each specialized on a specific set of correlated inputs, and to dynamically switch off the prediction in case it performs poorly (see Section 4.2.5).

4.2.4 Utility

We now turn our attention to the utility provided by the predictive mechanism. The property we want to prove is $\alpha(\delta)$*-accuracy*, introduced in Chapter 3.1. Similarly to the case of privacy, the accuracy of the predictive mechanism depends on that of its components, that is, on the accuracy of the noise mechanism, as well as the one of the Laplace mechanism employed by the test $\Theta(\epsilon_\theta, l, \tilde{z})$ (4.1). We can now state a result about the utility of a *single step* of the predictive mechanism.

Proposition 1 (accuracy)**.** *Let* $\mathbf{r}$ *be a run,* β *a budget manager, let* $(\epsilon_\theta, \epsilon_N, l) = \beta(\mathbf{r})$ *and let* $\alpha_N(\delta)$*,* $\alpha_\theta(\delta)$ *be the accuracy of* $N(\epsilon_N)$*,* $Lap(\epsilon_\theta)$ *respectively. Then the accuracy of* $\mathrm{Step}(\mathbf{r})$ *is*

$$\alpha(\delta) = \max(\alpha_N(\delta), l + \alpha_\theta(\delta))$$

Proof. Assumptions: the noise mechanism N is $\alpha_N(\delta)$-accurate and the laplacian noise Lap is $\alpha_\theta(\delta)$-accurate. The output depends on the outcome of the test function, and the possible cases are:

- $A \equiv d(x, \tilde{z}) \leq l - |Lap(\epsilon_\theta)|$
  returns the prediction, and we know its accuracy is within l

- $C \equiv l \leq d(x, \tilde{z}) \leq l + |Lap(\epsilon_\theta)|$
  despite it is not precise enough the prediction is returned
- $B \equiv l - |Lap(\epsilon_\theta)| \leq d(x, \tilde{z}) \leq l$
  despite the prediction was precise enough, a hard point is returned, which is $\alpha_N(\delta)$-accurate
- $D \equiv d(x, \tilde{o}) \geq l + |Lap(\epsilon_\theta)|$
  returns a hard point, which is $\alpha_N(\delta)$-accurate

In the following we denote the predicate with its letter, e.g. A, and the probability of it being true with P_A. In addition we denote with H and E the event of being in a hard or easy case.

We want to prove that for all δ, for each step i

$$P[d(z_i, x_i) \leq \alpha(\delta)] \geq \delta \tag{4.10}$$

For the hard cases, we use the assumption that N is $\alpha_N(\delta)$-accurate:

$$\begin{aligned} &P[d(z_i, x_i) \leq \alpha_N(\delta) \mid H] \cdot P_H = \\ &P[d(z_i, x_i) \leq \alpha_N(\delta) \mid B] \cdot P_B + \\ &P[d(z_i, x_i) \leq \alpha_N(\delta) \mid D] \cdot P_D \geq \\ &(P_B + P_D)\delta \end{aligned} \tag{4.11}$$

For the easy cases, we use the assumption that Lap is $\alpha_\theta(\delta)$-accurate and we define the shifted accuracy $\alpha'_\theta(\delta) = l + \alpha_\theta(\delta)$.

$$\begin{aligned} &P[d(z_i, x_i) \leq \alpha'_\theta(\delta) | E] \cdot P_E = \\ &P[d(z_i, x_i) \leq \alpha'_\theta(\delta) | A] \cdot P_A + \\ &P[d(z_i, x_i) \leq \alpha'_\theta(\delta) | C] \cdot P_C \geq \\ &1 \cdot P_A + \delta \cdot P_C \end{aligned} \tag{4.12}$$

We now join the two cases, choosing $\alpha(\delta) = \max(\alpha_N(\delta), \alpha'_\theta(\delta))$:

$$
\begin{aligned}
&P[d(z_i, x_i) \leq \alpha(\delta)] \geq \\
&P[d(z_i, x_i) \leq \alpha_N(\delta)|H] \cdot P_H + \\
&P[d(z_i, x_i) \leq \alpha'_\theta(\delta)|E] \cdot P_E \geq \qquad \text{using } 4.11, 4.12 \\
&(P_B + P_D)\delta + P_A + P_C\delta = \\
&P_A + (P_B + P_C + P_D)\delta = \\
&(1-\delta)P_A + \delta \geq \delta
\end{aligned}
$$

□

This result provides a bound for the accuracy of the predictive mechanism at each step. The bound depends on the triplet used $(\epsilon_\theta, \epsilon_N, l)$ to configure the test and noise mechanisms which may vary at each step depending on the budget manager used, thus the bound is step-wise and may change during the use of the system.

It should be noted that the bound is independent from the prediction function used, and assumes that the prediction gives the worst possible accuracy allowed by the test. Hence, under a prediction that always fails the bound is tight; however, under an accurate prediction function, the mechanism can achieve much better utility, as shown in the evaluation of Section 4.3.4.

As a consequence, when we configure the mechanism in Section 4.3.3, we scale down this bound to account for the improvement due to the prediction.

In the next section we will discuss the possibility to skip entirely the test in certain cases, of course our bound on accuracy cannot hold is such a case unless the mechanism designer can provide some safe assumptions on the accuracy of its skip-the-test strategy.

4.2.5 Skipping the test

The amount of budget devoted to the test is still linear in the number of steps and can amount to a considerable fraction; for this reason, given some particular conditions, we may want to skip it altogether using directly the prediction or the noise mechanism. The test mechanism we use (4.1) is defined for all $\epsilon_\theta > 0, l \in [0, +\infty)$. We can extend it to the case $\epsilon_\theta = 0, l \in \{-\infty, +\infty\}$ with the convention that $\Theta(0, +\infty, \tilde{z})$ always returns 1 and $\Theta(0, -\infty, \tilde{z})$ always

returns 0. This convention is based on the intuition that $d_{\mathcal{X}}(x, \tilde{z})$ is always greater than $-\infty$ and smaller than $+\infty$, and no budget is needed to test this.

The new test mechanisms are independent of the input x so they can be trivially shown to be private, with no budget being consumed.

Fact 2 (Privacy of Test function)**.** *The test functions* $\Theta(0, +\infty, \tilde{z})$ *and* $\Theta(0, -\infty, \tilde{z})$ *satisfy assumption 4.2.*

Now if β returns $(0, \epsilon_N, -\infty)$ we always fallback to the noise mechanism $N(\epsilon_N)$; this is especially useful when we know the prediction is not in conditions to perform well and testing would be a waste of budget. For instance, consider a prediction function that needs at least a certain number n of previous observables to be able to predict with enough accuracy; in this case we can save some budget if we directly use the noise mechanism for those n steps without testing. Note that the bound on utility is preserved in this case, as we can rely on the $\alpha_N(\delta)$-accuracy of $N(\epsilon_N)$.

On the other hand, the budget manager can return $(0, 0, +\infty)$ which causes the prediction to be reported without spending any budget. This decision could be based on any public information that gives high confidence to the prediction. Going back to the example of a user walking in a city, we can consider the time-stamp of each query, and the maximum speed that a user usually has when walking. If the test threshold we use is 5 km, and a new query arrives before the time that the user needs to walk 5 km, we can safely report the prediction, without checking. A good use of this case can be found in Section 4.3.4 where *timing information* is used to skip the test.

Note that the prediction is computed from public knowledge, so releasing it has no privacy cost. However in this case we loose any guarantee on the utility of the reported answer, at least in the general case; based on the criteria for skipping the test (as in the case of the user walking in the city), we could make assumptions about the quality of the prediction which would allow to restore the bound.

Note also that a purely predictive mechanism could be a viable alternative also when the mechanism runs out of budget and should normally stop. Reporting an untested prediction for free could provide some utility in this case.

On the prediction function

We provide here some remarks about the predictive component of our mechanism, which plays a crucial role in its behavior. The main idea of this work was to exploit

the correlation in the data, which in general is far from trivial to measure, analyze and characterize. Furthermore it is restrictive to consider *one* correlation as more often we refer informally about several *kinds* of correlation: among the input data itself, between different dataset or on the same dataset but at different times. The use of the prediction function allows to decouple the privacy mechanism from the correlation analysis, creating a family of modular mechanisms where we can *plug-in* different predictions. Moreover proving desirable security properties about the mechanism independently of the complex engineering aspects of the prediction is both easier and more reliable.

In the next section we explore the predictive mechanism in the context of location privacy, using a simple yet general prediction function. In Section 7 we discuss at length some more advanced predictions that we plan to consider in the near future.

4.3 Predictive mechanism for location privacy

The applicability of $d_{\mathcal{X}}$-privacy to location-based systems, called geo-indistinguishability in this context, was already discussed in Section 3.1. Having studied the general properties of our predictive mechanism, we are ready to apply it for location privacy.

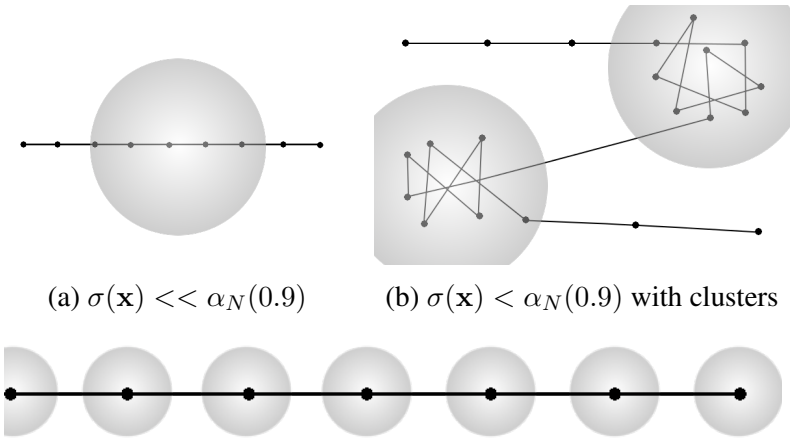
As already described in the preliminaries the sets of secret and observables are sets of geographical coordinates, the metric used is the euclidean distance and we will use $\Theta(\epsilon_\theta, l, \tilde{z})$ (4.1) as test function. We start with the description of a simple prediction function, followed by the design of two budget managers and finally some heuristics used to skip the test.

4.3.1 Prediction Function.

For the prediction function we use a simple strategy, the `parrot` prediction, that just returns the value of the last observable, which ultimately will be the last hard observable.

$$\texttt{parrot}((z, b) :: \mathbf{r}) = z \tag{4.13}$$

Despite its simplicity, this prediction gives excellent results in the case when the secrets are close to each other with respect to the utility required - e.g. suppose the user queries for restaurants and he is willing to accept reported points as far as 1 km from the secret point, if the next positions are tens of meters apart, then the



(a) $\sigma(\mathbf{x}) << \alpha_N(0.9)$ (b) $\sigma(\mathbf{x}) < \alpha_N(0.9)$ with clusters

(c) $\sigma(\mathbf{x}) >> \alpha_N(0.9)$

Figure 4.2: Example of traces with different σ and α_N.

same reported point will be a good prediction for several positions. Similarly, the prediction is quite effective when the user stays still for several queries, which is a typical case of a smartphone user accessing an LBS.

More concretely, we define the *step* of a trace as the average distance between its adjacent points $\sigma(\mathbf{x}) = \text{avg}_{0 \leq i < |\mathbf{x}|} \; d(x_i, x_{i+1})$ and we compare it with the $\alpha_N(0.9)$-accuracy of the noise mechanism. The intuition is that the parrot prediction works well on a trace $\mathbf{x}$ if $\sigma(\mathbf{x})$ is smaller than $\alpha_N(0.9)$ or in the presence of clusters because once we release a hard point we can use it as a good enough prediction for several other secret points close to it. Several cases are depicted in Fig. 4.2, where the black dots are secret points and the gray area represent $\alpha_N(0.9)$, where the reported points are likely to fall into.

Furthermore the parrot prediction can be trivially implemented on any system and it has the desirable property of being independent from the user; taking into account past traces of the user, for instance, would give a more effective prediction, but it would be restricted to that particular user.

4.3.2 Budget Managers

When configuring a mechanism we need to take into account 3 global parameters: the global privacy, the utility and the number of interactions, written (ϵ, α, n) for brevity. All three are interdependent and fixing one we obtain a relation between the other two. In our case we choose to be independent of the length of the

traces; to do so we introduce the *privacy consumption rate* (or just rate) which is the amount of budget spent at each step on average: $\rho(\mathbf{r}) = \frac{\epsilon(\mathbf{r})}{|\mathbf{r}|}$. This measure represent the privacy usage of the mechanism or how *fast* we run out of budget and given this value we can easily retrieve how many points we can cover given a certain initial budget. As already done for d_{err}, we also introduce the average-case rate for the mechanism as the expected value of ρ, given a prior distribution $\pi \in \mathcal{P}(\mathcal{X}^n)$ on traces:

$$E[\rho] = \sum_{\mathbf{x}} \pi(\mathbf{x}) \sum_{\mathbf{r}} \mathrm{PM}(\mathbf{x})(\mathbf{r})\ \rho(\mathbf{r})$$

Given that our main concern is privacy we restrict ourselves to ϵ-*bounded* budget managers, that guarantee that the total budget consumed by the mechanism will never exceed ϵ, and divide them in two categories:

Fixed Utility: In the independent mechanism if we want to guarantee a certain level of utility, we know that we need to use a certain amount of budget at each step, a fixed rate, thus being able to cover a certain number n of steps. However in our case, if the test is successful, we may save the cost of the noise and meet the fixed utility with a smaller rate per point; smaller rates translates in additional interactions possible after n. We fix the utility and minimize the rate.

Fixed Rate: Alternatively, if in the independent mechanism we want to cover just n steps, thus fixing the rate, we would obtain a certain fixed utility. On the contrary the predictive mechanism, in the steps where the test succeeds, spends less than the chosen rate, allowing the next steps to spend *more* than the rate. This alternance creates a positive behavior where hard points can use the saved budget to increase their accuracy that in turn makes predicting more accurate and likely to succeed, leading to more saving. Of course the average cost for all steps meets the expected rate. In this case we fix the rate and maximize the utility.

In both approaches (and all strategies in between), it is never easy to determine exactly the behavior of the mechanism, for this reason the budget manager should always be designed to respond dynamically over time.

4.3.3 Configuration of the mechanism

We now give an overview of the constraints that are present on the parameters of the predictive mechanism and a guideline to configure them to obtain the desired levels of privacy and utility. The only settings that the user needs to provide are ϵ and either α or ρ. The budget manager will define at each step the amount of

budget devoted to the test ϵ_θ, the noise mechanism ϵ_N and the test threshold l, starting from the global settings.

Budget usage. First we define the *prediction rate* PR as the percentage points predicted successfully; this property will be used to configure and to verify how effective is the predictive mechanism. We can then introduce a first equation which relates ϵ_θ and ϵ_N to the budget consumption rate: $\rho = \epsilon_\theta + (1 - PR)\epsilon_N$. This formula is derived from the budget usage of the mechanism (Lemma 1), with the two following approximations. First, ϵ_θ and ϵ_N in future steps are assumed constant. In practice they will be variable because this computation is re-done at each step with the actual remaining budget. Second, we assume the hard steps are evenly distributed along the run. This allows us to use PR, which is a global property of the trace, in a local computation.

Note that ρ is constant in the fixed rate case and is computed over the current run for the fixed utility case. We already knew that the budget available at each step had to be split between Θ and N, this result confirms the intuition that the more we manage to predict (higher PR) the less we'll need to spend for the noise generation (on average over the run).

Utility. From the utility result given by Proposition 1 we obtain an equation that relates all the parameters of the mechanism, ϵ_θ, ϵ_N and l. Given that the global utility will be the worst of the two, we decide to give both the noise and predictive components the same utility: $\alpha_N = l + \alpha_\theta$. Moreover, as discussed in the utility section, this result is a bound valid for every possible prediction function, even one that always fails, for this reason the bound may be too pessimistic for the practical cases where the prediction does work. In order to reduce the influence of the accuracy of the predictive component we introduce a parameter $0 \leq \eta \leq 1$ that can be set to 1 to retrieve the strict case or can safely go as low as 0.5 as shown in our experiments. Finally we obtain the following relation between the parameters: $\alpha = \alpha_N = \eta(l + \alpha_\theta)$.

Noise-threshold ratio. Now we have two equations for three parameters and to completely configure the mechanism we introduce an additional parameter $0 \leq \gamma \leq 1$ that is used to tune, in the predictive component, the ratio between the threshold l and the Laplacian noise added to it so that $\gamma = \frac{\alpha_\theta}{l}$. The intuition is that γ should not be bigger that 1, otherwise the noise could be more important

```
budget manager β(r)
  if ε(r) ≥ ε then STOP
  else
    ε_θ := η (c_θ/α)(1 + 1/γ)
    ε_N := c_N/α
    l := c_θ/(γ ε_θ)
    return (ε_θ, ε_N, l)
```

(a) Fixed Utility configured with ϵ and α

```
budget manager β(r)
  if ε(r) ≥ ε then STOP
  else
    ε_N := ρ / ((1−PR) + (c_θ/c_N) η (1+1/γ))
    ε_θ := ε_N η (c_θ/c_N)(1 + 1/γ)
    l := c_θ/(γ ε_θ)
    return (ε_θ, ε_N, l)
```

(b) Fixed Rate configured with ϵ, ρ and PR

than the threshold and we might as well use a random test. For our experiments we found good values of γ around 0.8.

Note that both η and γ are values that should be determined using a representative sample of the expected input, in a sort of tuning phase, and then fixed in the mechanism. The same goes for the expected prediction rate that is used to configure the budget managers, at least in the beginning this value is necessary to allocate some resource for Θ, after some iterations it is computed from the actual run.

Relation between accuracy and epsilon. The final simplification that we apply is when we compute the accuracy of the noisy components, for both the linear Laplacian and the polar Laplacian we can compute their maximum value up to a certain probability δ using their inverse cumulative probability distributions, that we denote `icll` and `icpl` respectively. Fixing δ to 0.9, both these functions can be expressed as the ratio of a constant and the epsilon used to scale the noise $\alpha_N(\delta) = \texttt{icpl}(\epsilon_N, \delta) = \frac{c_N(\delta)}{\epsilon_N}$ and $\alpha_\theta(\delta) = \texttt{icll}(\epsilon_\theta, \delta) = \frac{c_\theta(\delta)}{\epsilon_\theta}$.

Note that this characterization of α_N is valid only for the polar Laplacian used to achieve geo-indistinguishability. In fact, this is the only domain specific part of the configurations presented, that can otherwise be applied to a generic notion of $d_{\mathcal{X}}$-privacy.

Now that we have the equations that relate the various parameters, from the settings given by the user we can realize the two budget managers, shown in Figure 4.3a and 4.3b.

Furthermore we can compare the expected rate or accuracy of our mechanism with those of an independent mechanism and find the prediction rate that we need to meet to provide an improvement. We obtain in both cases a lower bound on the prediction rate: $PR \geq \eta \frac{c_\theta}{c_N}(1 + \frac{1}{\gamma})$. This gives an idea of the feasibility of a

configuration before actually running it, for example using the parameters of our experiments we find that it is necessary to predict at least 46% of points to make up for the cost of the test.

4.3.4 Evaluation

To evaluate our mechanism, we follow our motivating example stated in the introduction of a user performing several activities while moving around the city throughout a day, possibly using different means of transport. During these activities, the user performs queries to an LBS using his mobile device, while wishing to keep his location private.

We assume that the user queries the LBS only when being still or moving at a slow speed (less than 15 km/h); this reflect the semantic of a geo localized query: there is usually little value in asking information relative to one's current position if the position is changing quickly. We perform a comparison between the independent mechanism IM and our predictive mechanism PM, both using polar Laplace noise as the underlying noise mechanism. The mechanisms are evaluated on two data sets of real GPS trajectories, using both a fixed-utility and fixed-rate budget managers and a skip strategy.

Data sets

The first data set we tested our mechanism against, is the well known GeoLife [ZXM10] which collects 18.670 GPS trajectories from 182 users in Beijing during a period of over five years. In this set the users take a variety of means of transport, from walking and biking to car, train, metro, taxi and even airplane. Regarding the trajectories length we can roughly divide them on three equal groups, less than 5 km, between 5 and 20 km and more than 20 km. As for duration 58% are less than 1 hour, 26% between 1 and 6 hours and 16% more than 6 hours.

The second data set is Tdrive [YZZ$^+$10], a collections of about 9000 taxi trajectories, always in the city of Beijing. As opposed to the variety of Geolife in this set we have only cars movements and the trajectories tends to be longer in both time and distance. The interest of using this set, which does not exactly correspond to our target use case of a user walking in a city, is to test the flexibility of the mechanism.

In order to use this sets some preprocessing is needed in order to model our use case. GPS trajectories present the problem of having *all the movements* of the user, instead of just the points where the user actually *queried* the LBS, which

is a small subset of the trajectory. For this reason we perform a probabilistic "sampling" of the trajectories that, based on the speed and type of user, produces a trace of query points. First, we select the part of the trace where the speed is less than 15 km/h, and in these segments we sample points depending on the type of user, as explained below.

Users are classified based on the frequency of their use of the LBS, from occasional to frequent users. This is achieved by defining two intervals in time, one brief and the other long (a *jump*), that could occur between two subsequent queries. Then each class of users is generated by sampling with a different *probability of jumping* p, that is the probability that the next query will be after a long interval in time. Each value of p gives rise to a different prior distribution π on the produced traces, hence affecting the performance of our mechanism.

The interval that we used in our experiments are 1 and 60 minutes, both with addition of a small Gaussian noise; frequent users will query almost every minute while occasional users around every hour. In our experiments we generated 11 such priors, with probability of jumping ranging from 0 to 1 at steps of 0.1, where each trace was sampled 10 times.

Configuration

In order to configure the geo-indistinguishable application, first the user defines a radius r^* where she wishes to be protected, that we assume is 100 meters, and then the application sets ϵ^*, the global level of privacy, to be $\ln 10$. This means that taken two points on the radius of 100 meters their probability of being the observables of the same secret differ at most by 10, and even less the more we take them closer to the secret. We think this is a reasonable level of privacy in a dense urban environment. It follows that we have a budget $\epsilon = \epsilon^*/r^* = 0.023$ to manage. For what concerns the two budget managers, the fixed-rate was tested with a 3.3% rate, which corresponds to about 30 queries, which in a day seems a reasonable number even for an avid user. For the fixed-utility we set an accuracy limit 3 km, again reasonable if we consider a walking distance and that these are worst cases.

Skip-the-test strategy

The nature of location traces allows us to use a very safe skip-the-test strategy, which greatly improves the performance of the system for some configurations while keeping the error under control, which is not always the case when the test

is skipped. While the aim of the mechanism is to hide the user's position, the timestamp of a point is observable, hence we can use the elapsed time from the last reported point to estimate the distance that the user may have traveled. If this distance is less than the accuracy required, we can report the predicted value without testing it, we know that the user can't be too far from his last reported position. The risk of this approach lies in the speed that we use to link elapsed time and traveled distance, if the user is faster that expected (maybe he took a metro) we would report an inaccurate point. To be on the safe side it should be set to the maximum speed we expect our users to travel at, however with lower values we'll be able to skip more, it is a matter of how much we care about accuracy or how much we know about our users. In our experiments we assumed this speed to be 0.5 km/h. The speed could also be approximated from previous points.

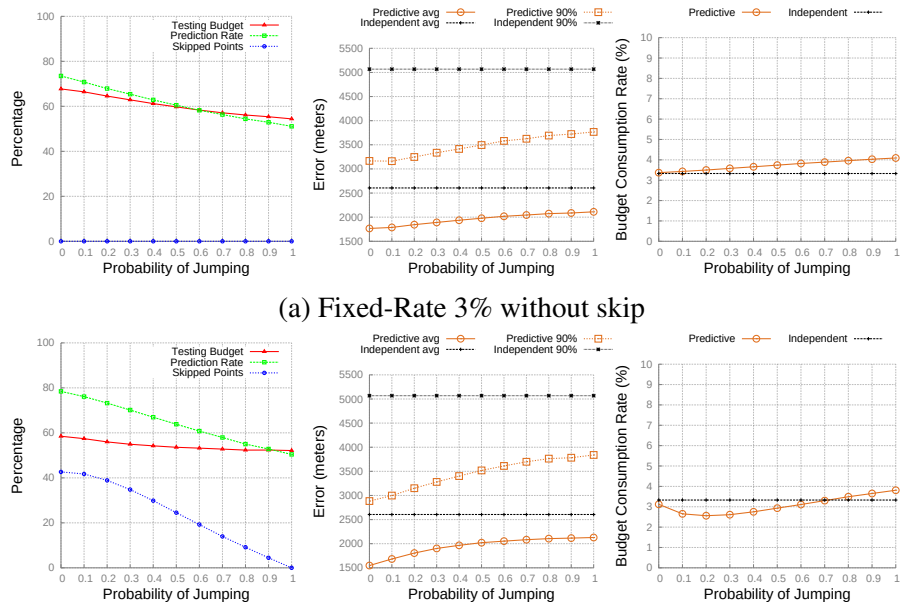
We would expect this approach to be more convenient in a context where accuracy is not the primary goal; indeed skipping the test will provide the greatest advantage for the fixed-utility case, where we just don't want to exceed a worst case limit.

Additionally we use another skip-the-test strategy to use directly with the noise mechanism when we are in the first step and thus there is no previous hard point for the parrot prediction to report. This is a trivial example of skip strategy, yet it can lead to some budget savings.

Results

It should be noted that both the preprocessing and the sanitization were performed with same configuration on both data sets. The results of running the mechanism on the samples traces from the Geolife data set, are reported in figures 4.4, 4.5. In the horizontal axis we have the probability p that was used during the sampling, to determine how often the user performs a *jump* in time: the smaller the value the more frequent the queries. For each budget manager we plot: In the first graph, some general statistics about the mechanism, such as the prediction rate achieved, the amount of budget devoted to Θ and the amount of skipped points; In the second column the average ($E[d_{\text{err}}]$) and 90-th percentile ($\alpha(0.9)$) of the error; In the third the average budget consumption rate $E[\rho]$. Furthermore we run the experiments with and without the skip the test strategy, for the sake of comparison.

The graphs present a smooth behavior, despite the use of real data, because of the sampling on each trace and the averaging over all traces of all users. As general remarks, we can see that the prediction rate degrades as the users become more



(a) Fixed-Rate 3% without skip

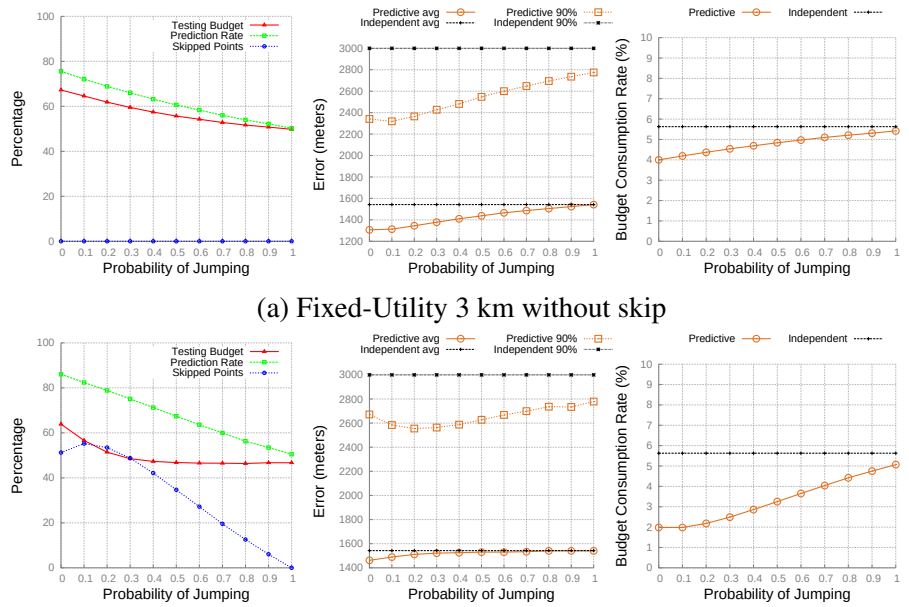
(b) Fixed-Rate 3% with skip

Figure 4.4: General statistics, Average Error and Rate for Fixed-Rate budget manager.

occasional, thus less predictable, and the same goes for the number of skipped points. Notice that the testing budget adapts with the prediction rate which is a sign that the budget managers reconfigure dynamically.

Fixed-rate (Fig. 4.4): fixing the rate to 3.3% to cover 30 points, we can devote the budget saved to improve the accuracy. In the right most graph we see that indeed the rate is very stable even in the unpredictable cases, and very close to the rate of the independent mechanism. The graph in the center shows great improvements in the average error, 500 m in the worst case and 700 m in the best, and even more remarkable is the improvement for the maximum error, 1.3km up to 1.9km. With the skip strategy we see a small improvement for $p \leq 0.5$, again both in average and maximum error, which correspond to a decrease in the testing budget in the left most graph: the budget saved skipping the test is invested in more accurate noise.

Fixed-utility (Fig. 4.5): fixing the maximum utility (or in-accuracy) to 3 km, our mechanism manages to save up to 1.5% of budget rate. If we want to compare the number of points covered, the independent mechanism can do around 17 points while the predictive 24. As expected the average and max errors are below the independent mechanism corresponding values which confirms that the budget manager is working correctly keeping the utility above a certain level. Despite this



(a) Fixed-Utility 3 km without skip

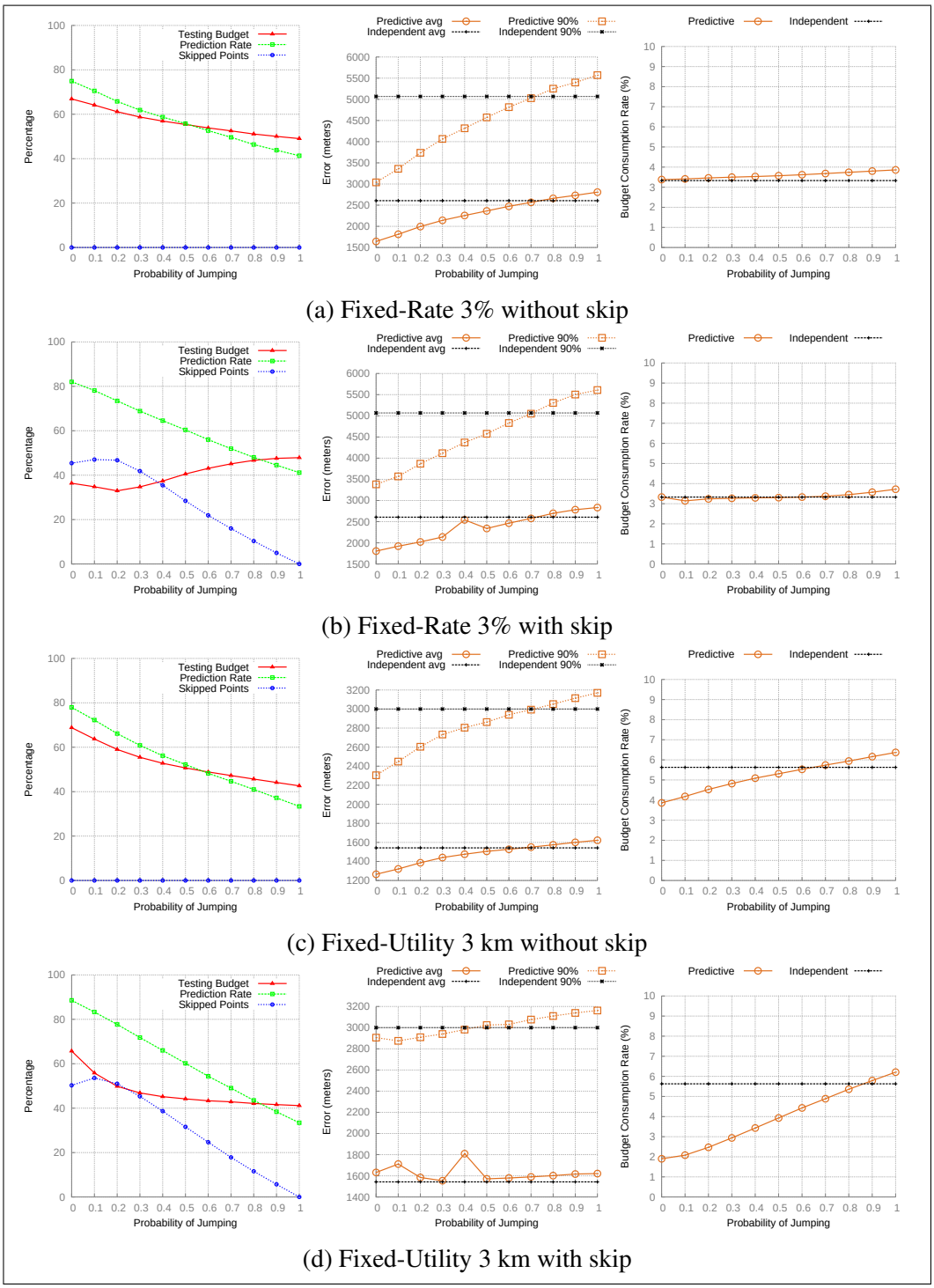
(b) Fixed-Utility 3 km with skip

Figure 4.5: General statistics, Average Error and Rate for Fixed-Utility budget manager.

they don't show a stable behavior like the rate in the fixed-rate case, this is due to the fact that while we can finely control the amount of budget that we spend, the error is less controllable, especially the one produced by the predictive component. With the skip strategy in this case we obtain a very noticeable improvement in this case, with rates as low as 2% in the best case which translates to 50 points covered. As already pointed out, in this case the skip strategy is more fruitful because we care less about accuracy.

Tdrive. In Figure 4.6 we see that this data set reports remarkably similar performance to Geolife when the probability of jumping p is less than 0.7. In this cases the predictive mechanism is consistently a better choice than the independent mechanism on both budget managers. On the contrary for higher values of p the independent mechanism performs better, it is interesting to notice that the prediction rate at $p = 0.7$ starts to be lower than 46%, as expected from Section 4.3.3. This difference between the best and worst case is more accentuated in Tdrive precisely because the prediction function was not designed for this scenario. The more sporadic users are even less predictable as they are moving at higher speeds and roaming larger areas. Also the skip strategy, again designed for walking users, shows some spikes in the average error, due to wrongly skipped

points where probably the taxi speeded up suddenly.



(a) Fixed-Rate 3% without skip

(b) Fixed-Rate 3% with skip

(c) Fixed-Utility 3 km without skip

(d) Fixed-Utility 3 km with skip

Figure 4.6: Tdrive: General statistics, Average Error and Rate.

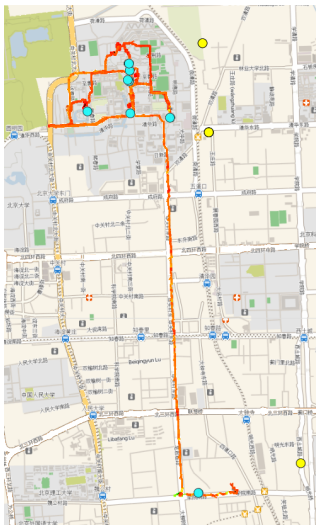


Figure 4.7: Original trace (red), sampled trace (light blue) and reported trace (yellow).

Example of sanitized trace. Figure 4.7 displays one of Geolife trajectories sanitized with fixed utility. The original trace, in red, starts south with low speed, moves north on a high speed road and then turns around Tsinghua University for some time, again at low speed, for a total of 18 km traveled in 10 hours. The sampled trace was obtained with a probability 0.5 of jumping and is plotted in light blue: as expected, 9 of the points are north, one south and the middle part was skipped. Finally in yellow we have the reported trace with 3 locations, which were used once for the point at the bottom, 7 times for the one in the middle and twice for point in the top.

4.3.5 Future Work.

As the experiments show the more efficient use of budget allows us to cover a day of usage, which was the goal we were aiming for in order to attack realistic applications. The intuition is that even if there is correlation between the traces of the same user on several days (for example we go to work and home every day) still it is not enough to accurately locate the user at a precise moment in time (we might go to work later, or follow a different road). It is not clear though if one day is enough time to break the correlation and possibly reset the budget, we leave to future work to investigate in which cases it is indeed possible to reset the system and when on the contrary the epsilon keeps increasing.

One other possibility to prolong even further the use of the system is to im-

prove the prediction. The experimental part of this thesis was carried on with a prediction simple enough to be effective yet not distracting with engineering details. An extension we plan to develop consist in using the mobility traces of a user, or of a group of users, to designate locations where the next position is likely to be. In [GKdPC11] the authors already developed inference attacks on the reported locations of users to discover points of interests and future locations, among other things; the idea is to use these attacks as a prediction. If we consider the use case of a mobile phone, the mechanism itself could collect the reported traces and train itself to predict more accurately.

We are also developing a *linearizing* prediction, that determines the direction using a linear regression method which additionally allows to detect *turns*, sharp changes in direction, thanks to the error reported. This kind of prediction targets cases where the system needs to frequently report its position with good accuracy, such as a navigation system, we think that a small amount of privacy could still be desirable, for example to hide the exact position along a road. Of course this prediction only works in cases where the secret trajectory is very linear, restricting its usage to cases such as trains, airplanes or possibly boats as means of transport. One possible improvement could be the use of a non-linear regression technique but it still has to be explored.

Alternatively we are considering the use of public geographic information to improve the prediction, which could simply translate to using already developed map-matching algorithms: typically in navigation systems an approximate location needs to be matched to an existing map, for example to place the user on a road. Map matching would make trivial predicting the direction of the user moving on a road for example, while in crossroads could be dealt with with the help of the mobility traces already discussed before: if on the left the is just countryside and on the right a mall, the user is more likely to turn right. Ultimately if more than one prediction function prove effective, we are interested in the possibility to merge them, for instance using multiplicative weights or related technique (e.g. Kalman filters): each prediction is assigned a weight, at each turn the prediction with the highest weight is interrogated, if we are in easy case its weight is raised otherwise is reduced, in the hope that when changing scenario the weights would adjust and the right prediction would be picked.

4.4 Incorporating fences in the metric

As discussed in the introduction one issue of geo-indistinguishability is that, repetitive use of a mechanism from the same location is bound to reveal that location as the number of reports increases. This is crucial for locations that the user frequently visits, such as his home or work location. Such locations cannot be expected to remain indistinguishable in the long run; repetitive use of the mechanism can reveal them with arbitrary accuracy.

Despite the fact that all privacy mechanism are susceptible to this privacy erosion over time, the compositionality property of ϵ-geo-indistinguishability quantifies exactly this privacy degradation: there is a linear accumulation of ϵ, lowering the privacy protection guaranteed. In the previous Section 4.2 we developed one technique, the predictive mechanism, to alleviate this effect and make a more efficient use of the budget. However for highly recurrent cases even the predictive mechanism will eventually exhaust its budget.

This problem, especially for the home-work locations, has been already studied in the literature [GP09], although in the context of anonymity i.e. how to match a user identity to a pair of home-work locations. In fact our interest is focused on reducing the accuracy of the reported location, because of the sensitive data the attacker can infer from it. Even if the user is authenticated with a LBS, for instance to notify her friends that she is home, it is still valuable to not disclose the precise address (that friends know anyway).

4.4.1 Fences

For highly recurrent locations we propose the use of geo *fences*, areas on the map where the user's movements are completely hidden and that are considered known to the attacker. This technique is not novel and indeed has been widely used by a large number of LBS. For example, personal rental services (e.g. Airbnb) allow the user to indicate an area where the good to be rented is located, so that other users can evaluate if it is at a convenient distance without compromising the privacy of the owner. Despite the vast use of fences in practice, to the best of our knowledge, there is a lack of works in the literature about their implementations or evaluating their effectiveness.

Our contribution consist in a simple formalization of fences in the framework of distinguishability metrics. This construction allows to hide completely sensitive locations within a fence, while permitting the use of any other $d_{\mathcal{X}}$-private

mechanism outside. Given a privacy metric $d_{\mathcal{X}}$, we define a new fenced metric d_F as:

$$d_F(x, x') = \begin{cases} 0 & x, x' \in F \\ d_{\mathcal{X}}(x, x') & x, x' \notin F \\ \infty & \text{otherwise} \end{cases}$$

Outside the fence the original metric $d_{\mathcal{X}}$ is in place while inside the fence all points are completely indistinguishable. The advantage is that being zero the distance *inside* the fence, any repeated use of the mechanism from the sensitive location comes for *free*, effectively stopping the linear growth of the budget. On the other hand being the fence completely distinguishable from the outside, the attaker is always able to tell if we are inside or not.

Regarding utility, in this case it simply depends on the size of the fence, in direct contrast with privacy.

4.4.2 Mechanism

We describe now two possible mechanisms to implement a fenced metric.

The first possibility is to take an existing mechanism K providing $d_{\mathcal{X}}$-privacy, and obtain a fenced mechanism K_F by extending K with a deterministic test to determine if we are in the fence or not. When outside the fence, we keep using the original mechanism K. When inside the fence we can report a location in the fence that can be choosen uniformly at random or fixed once and consistently reported.

The second possibility is, once defined our fenced metric d_F, to plug it in an Exponential mechanism. As discussed in the Preliminaries 3.1, the Exponential mechanism can take any distinguishability metric $d_{\mathcal{X}}$, and provide $d_{\mathcal{X}}$-privacy by applying exponential noise scaled according to the metric.

In Figure 4.8 we can see an example of fence introduced in an elastic metric (presented in the next Chapter 5) and implemented with an Exponential mechanism. On the left we have the distribution inside the fence, that as expected is perfectly uniform, covering a few blocks and proving an adequate level of privacy. On the right we can see the distribution of a point right outside, the fence is clearly visible and the mechanism reports right outside it.

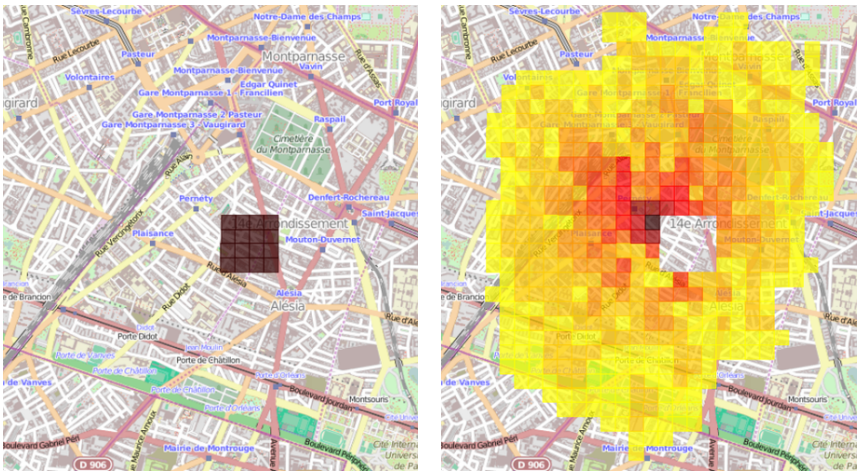


Figure 4.8: Probability distribution of reported location inside and outside the fence. Darker colors indicate more likely values.

4.4.3 Future work

Automatic configuration of position and size. In order to configure the position and size of the fences, the user input would be the best option (as shown in [BKS10]), however they could also be inferred and suggested automatically. In [GKdPC11] the authors developed an attack to identify POI of a specific user, from a set of mobility traces. A similar technique could be employed on the user's phone, over a training period, to collect and analyze her movements for a few days. The mechanism would then automatically detect recurrent locations and suggest the user to fence them, possibly detecting more than just home/work locations.

With the use of geolocated queries, such as those used to extract privacy points in Section 5.2.3, we could determine the size of the fence so to include a reasonable amount of buildings for home and other POIs for work.

Evaluation. The main problem in this context is to find a dataset with user traces *and* recurring locations such as home or work. Using again the techniques of [GKdPC11] it is possible to approximate some of this locations but it is challenging to confirm their correctness with ground knowledge. For this reason we are currently leaving as future work the evaluation of the fenced mechanism.

4.5 Conclusion

We designed a general framework for private predictive $d_{\mathcal{X}}$-private mechanisms able to manage the privacy budget more efficiently than the standard approach, in the cases where there is a considerable correlation on the data. The mechanism is modular and clearly separates the privacy protecting components from the predictive components, allowing ease of analysis and flexibility. We provide general configuration guidelines usable for any notion of $d_{\mathcal{X}}$-privacy and a detailed instantiation for geo-indistinguishability. We tested the geo private mechanism obtained with two large sets of GPS traces and confirmed the goals set in the design phase. Experimental results show that the correlation naturally present in a user data is enough for our mechanism to outperform the independent mechanism in the majority of prior tested. Additionally, for highly recurrent locations, we have discussed how the geo fencing technique can be elegantly expressed in any distinguishability metric and automatically configured to aid the user.

Chapter 5

Flexible Use over Space

5.1 Introduction

In the preliminaries (Chapter 3.1) we described our intuition of location privacy as *hiding in a crowd*, with the idea that the size and variety of the crowd makes the inference of the attacker less accurate. In geo-indistinguishability we obtain this property by properly setting the ϵ parameter, thus establishing an area of points (resident people, shops, recreational centers etc.) indistinguishable from our real location.

However, one problem with the geo-indistinguishability framework is that, being based on the Euclidean distance, its protection is uniform in space, while the density of the elements that constitute the "crowd" in general is not. This means that, once the privacy parameter is fixed, a mechanism providing geo-indistinguishability will generate the same amount of noise independently of the real location on the map, i.e., the same protection is applied in a dense city and in a sparse countryside. As a consequence, an unfortunate decision needs to be made: one could either tune the mechanism to the amount of noise needed in a dense urban environment, leaving less dense areas unprotected. Or, to ensure the desired level of privacy in low-density areas, we can tune the mechanism to produce a large amount of noise, which will result in an unnecessary degradation of utility in high-density areas.

In this chapter we propose a novel *elastic* privacy definition that warps the geometrical distance, capturing the different degrees of density of each area. This can be achieved while maintaining the main principles of geo-indistinguishability, by replacing the Euclidean distance with a constructed *distinguishability metric* $d_{\mathcal{X}}$. $d_{\mathcal{X}}$-privacy (Chapter 3.1) ensures that secrets which are close with respect to

$d_{\mathcal{X}}$ should remain indistinguishable, while secrets that are distant $d_{\mathcal{X}}$ are allowed to be distinguished. We then have the flexibility to adapt the distinguishability metric $d_{\mathcal{X}}$ to our privacy needs.

Going back to the intuition of privacy as being surrounded by a crowd, we can reinterpret it in the light of distinguishability metrics. On one hand people or points of interest can be abstracted as being a *privacy mass* that we can assign differently to every location. On the other hand the concept of being close to a location rich in privacy can be seen as the desire to be similar, or indistinguishable to such a location. Therefore we can express our intuitive privacy with a distinguishability metric that satisfies the following requirement: every location should have in proximity a certain amount of privacy mass. This can be better formalized with a requirement function $\mathrm{req}(l)$ that for every distinguishability level l, assigns a certain amount of privacy mass that must be present within a radius l in the metric $d_{\mathcal{X}}$. Contrary to geo-indistinguishability, that considers space uniform and assigns to every location the same privacy value, we build a metric that is flexible and adapts to a territory where each location has a different privacy importance. In comparison with the Euclidean metric, our metric stretches very private areas and compresses the privacy poor ones in order to satisfy the same requirement everywhere, for this reason we call it an *elastic metric*. By using $d_{\mathcal{X}}$-privacy with a metric that takes into account the semantics of each location, we preserve the strengths of the geo-indistinguishability framework while adding flexibility, borrowing ideas from the line of work of l-diversity [XKP09, MKGV07]. This flexible behavior reflects also on the utility of the resulting mechanism, areas poor in privacy will result in more noisy sanitization.

We then need a way to compute the actual metric $d_{\mathcal{X}}$ satisfying the requirement $\mathrm{req}(l)$. We propose a graph-based algorithm that can efficiently compute an elastic metric for a large number of locations. The algorithm requires a set of locations marked with privacy mass and a privacy requirement to satisfy. Starting from an empty graph, it iteratively adds weighted edges to satisfy the requirement. The resulting distance between two locations is the weight of the shortest path connecting them. Once obtained our metric we show how to use an exponential distribution to obtain automatically a $d_{\mathcal{X}}$-private mechanism that can also be efficiently implemented.

Finally, we show the applicability of our technique by evaluating it on two real-world datasets. We start by building an elastic metric for Paris' wide metropolitan area, in a grid of $562,500$ locations covering an area of 5600 km^2. Privacy mass is computed from semantic information extracted from the OpenStreetMap database

[1], and the whole computation is performed in under a day with modest computational capability, demonstrating the scalability of the proposed algorithm.

We then compare the elastic mechanism to the Planar Laplace mechanism satisfying geo-indistinguishability, on two large areas in the center of Paris as well as a nearby suburb. The evaluation is performed using the datasets of Gowalla and Brightkite[CML11], two popular location-based social networks, and the widely used Bayesian privacy and utility metrics of Shokri et al. [STBH11]. The results show that the dynamic behavior of the elastic mechanism, in contrast to Planar Laplace, provides adequate privacy in both high and low-density areas.

Finally, building on the experience of the elastic mechanism, we develop a lightweight adaptable mechanism, that requires little pre-computation. The idea is to use simple queries to establish the privacy mass of large areas of several squared kilometers, called *tiles*, and to adapt the level of noise when moving in each tile. We achieve this by employing a Planar Laplace mechanism tuned with different values of ϵ. Given that the change of ϵ is dependent on the current location of the user, which is sensitive, we need to test the current tile in a private way, similarly to the private test used in the predictive mechanism (Section 4.2). The flexibility of the tiled mechanism is not comparable to the fine-grained construction of the elastic mechanism, but its simplicity makes it suitable for inclusion in the browser extension Location guard that we describe in detail in Chapter 6.

Contributions.

- We propose the use of elastic metrics to solve the flexibility problem of geo-indistinguishability. We formalize a requirement of such metrics in terms of privacy mass, capturing properties such as space, population, points of interest, etc.
- We define an efficient and scalable graph-based algorithm to compute a metric $d_{\mathcal{X}}$ satisfying this requirement.
- We perform an extensive evaluation of our technique in a large metropolitan area using two real-world datasets, showing the advantages of the elastic metric compared to standard geo-indistinguishability.
- We propose a tiled mechanism that allows some degree of flexibility for a fraction of the computational cost of the elastic mechanism.

[1] `http://openstreetmap.org/`

Plan of the chapter. In Section 5.2 we present in detail the elastic metric. First how to extract meaningful privacy resource for each location, then the definition of a privacy requirement and finally the graph-based algorithm that generated the metric. In Section 5.3 an elastic mechanism is built and evaluated in comparison with a geo-indistinguishable mechanism. Finally in Section 5.4 we describe the tiled mechanism and some preliminary results on its implementation.

5.2 An elastic distinguishability metric

Rarely we are interested in hiding our geographical location per se, more commonly we consider our location sensitive because of the many personal details it can indirectly reveal about us. For this reason reducing the *accuracy*, through perturbation or cloaking, is considered an effective technique for reporting a location that is at the same time meaningful and decoupled from its sensitive semantic value. In the preliminaries we explained how in geo-indistinguishability the privacy level is configured for a specific radius r^*, that is perceived as private. Using $r^* = 300$ m for a large urban environment is based on the fact that a large number of shops, services and people can be found within that radius, limiting the power of inference of the attacker. This is an intuitive notion of location privacy that we call *hiding in a crowd*, where the crowd represents the richness and variety that a location provides to the user's privacy.

As explained in the introduction, the use of ϵd_2 as the distinguishability metric has a major drawback. The simple use of geographical distance to define privacy ignores the nature of the area in which distances are measured. In a big city, ϵ can be tuned so that strong privacy is provided within 300 meters from each location but in a rural environment, they are not perceived as sufficient privacy. And even inside a city, such a protection is not always adequate: within a big hospital an accuracy of 300 meters might be enough to infer that a user is visiting the hospital.

In this section we address this issue using a custom distinguishability metric that is adapted to the properties of each area, an *elastic metric*. More specifically we discuss properties that such a metric should satisfy, and in the next section we present an algorithm for efficiently computing such a metric.

Once obtained the elastic metric, we can plug it in the $d_{\mathcal{X}}$-privacy definition and obtain an *elastic privacy definition* for location privacy, much like was done for geo-indistinguishability. Furthermore we can use the Exponential mechanism

presented in Chapter 3.1 to obtain a *sanitization mechanism* that satisfies our elastic privacy definition.

5.2.1 Privacy mass

The main idea to overcome the rigidity of geo-indistinguishability is to construct a distinguishability metric $d_{\mathcal{X}}$ that adapts depending on the properties of each area. However in order to distinguish a city from its countryside, or on a finer scale, a crowded market place from a hospital, we first need to assign to each location how much it contributes to the privacy of the user. In other words we consider *privacy as a resource* scattered on the geographical space and each locations is characterized by a certain amount of privacy.

More precisely the privacy of *location* x depends on the *points that are indistinguishable from* x. Let $\mathrm{cover}(x)$ denote the set of points that are "highly" indistinguishable from x. For the moment we keep $\mathrm{cover}(x)$ informal, it is properly defined in the next section. Intuitively, the privacy of x depends:

- on the *number* of points in $\mathrm{cover}(x)$: an empty set clearly means that x can be inferred, while a set $\mathrm{cover}(x)$ containing a whole city provides high privacy. This corresponds to the idea that hiding within a large area provides privacy.
- on the semantic *quality* of points in $\mathrm{cover}(x)$: it is preferable for $\mathrm{cover}(x)$ to contain a variety of POIs and highly populated locations, than points in a desert or points all belonging to a hospital. This corresponds to the idea that hiding within a populated area with a variety of POIs provides privacy.

To capture this intuition in a flexible way we introduce the concept of *privacy mass*. The privacy mass of a location x, denoted by $m(x)$, is a number between 0 and 1, capturing the location's value at providing privacy. We also denote by $m(A) = \sum_{x \in A} m(x)$ the total mass of a set A. The function $m(\cdot)$ should be defined in a way such that a set of points containing a *unit of mass* provides sufficient cover for the user. Hence, the metric we construct needs to satisfy that

$$m(\mathrm{cover}(x)) \geq 1 \qquad \forall x \in \mathcal{X}$$

Following the idea that privacy comes by hiding within either a "large" or "rich" area, we define $m(x)$ as

$$m(x) = a + q(x)b \tag{5.1}$$

where a is a quantity assigned to each location simply for "occupying space", $q(x)$ is the "quality" of x and b is a normalization factor. The quality $q(x)$ can be measured in many ways; we measure it by querying the OpenStreetMap database for a variety of POIs around x, as explained in Section 5.2.3. Assuming $q(x)$ to be given, we can compute a and b as follows: we start with the intuition that even in empty space, a user feels private if he is indistinguishable within some large radius r_{large}, for instance 3000 m (r_{large} can be provided by the user himself). Let $B_r(x) = \{x' \mid d_2(x, x') \leq r\}$ denote the Euclidean ball of radius r centered at x. Letting x be a location in empty space, i.e. with $q(x) = 0$, intuitively we want that $\text{cover}(x) = B_{r_{\text{large}}}(x)$, and $m(\text{cover}(x)) = 1$, hence

$$a = \frac{1}{|B_{r_{\text{large}}}(x)|}$$

Similarly, in an "average" location in a more private place, like a city, a user feels private if he is indistinguishable within some smaller radius r_{small}, for instance 300 m (r_{small} can be also provided by the user himself). Let

$$avg_q = E_x q(B_{r_{\text{small}}}(x))$$

be the average quality of a r_{small} ball (where expectation is taken over all location in the city). On average we establish that such a ball contains one unit of privacy mass, thus we get:

$$1 = a \cdot |B_{r_{\text{small}}}(x)| + b \cdot avg_q \qquad \text{hence}$$
$$b = \frac{1}{avg_q}(1 - \frac{|B_{r_{\text{small}}}(x)|}{|B_{r_{\text{large}}}(x)|})$$

Note that the intuitive requirement of being indistinguishable from a set of entities with some semantic characteristics is widely used in the privacy literature. Most notably, k-anonymity 3.1 requires to be indistinguishable from group of at least k individuals, while l-diversity adds semantic diversity requirements: hiding among k hospitals is not acceptable since it still reveals that we are in a hospital. It should be emphasized, however, that although we follow this general intuition, we do so inside the geo-indistinguishability framework, leading to a privacy definition and an obfuscation mechanism more robust to background knowledge and that doesn't require any third party.

5.2.2 Requirement

Having fixed the function $m(x)$, we turn our attention to the requirement that our distinguishability metric $d_{\mathcal{X}}$ should satisfy in order to provide adequate privacy for all locations.

Let $B_l(x)$ denote the $d_{\mathcal{X}}$-ball of distinguishability level l. The $d_{\mathcal{X}}$-privacy property ensures that, the smaller l is, the harder it will be to distinguish x from any point in $B_l(x)$. Our requirement is that $B_l(x)$ should collect an appropriate amount of privacy mass:

$$m(B_l(x)) \geq \mathrm{req}(l) \qquad \forall l \geq 0, x \in \mathcal{X} \tag{5.2}$$

where $\mathrm{req}(l)$ is a function expressing the required privacy mass at each level. The algorithm of Section 5.2.4 ensures that the above property is satisfied by $d_{\mathcal{X}}$.

It remains to define the $\mathrm{req}(l)$ function. Let l^* denote a "small" distinguishability level (see Chapter 3.1 for a discussion on distinguishability levels and what small means. In this thesis we use $l^* = \ln 2$). Points in $B_{l^*}(x)$ will be "highly" indistinguishable from x, hence $B_{l^*}(x)$ plays the role of $\mathrm{cover}(x)$ used informally in the previous Section. Privacy mass was defined so that $m(\mathrm{cover}(x)) \geq 1$, hence we want $\mathrm{req}(l^*) = 1$.

Moreover, as the $d_{\mathcal{X}}$-distance l from x increases, we should collect even more mass, with the amount increasing quadratically with l (since the number of points increases quadratically). Hence we define $\mathrm{req}(l)$ as a quadratic function with $\mathrm{req}(0) = 0$ and $\mathrm{req}(l^*) = 1$, that is:

$$\mathrm{req}(l) = \left(\frac{l}{l^*}\right)^2$$

Defining the requirement in terms of privacy mass is a flexible way to adapt it to the properties we are interested in. Indeed we can re-obtain geo-indistinguishability as a special case of our new framework if all locations are considered just for their contribute in space, not quality, i.e. $q(x) = 0$. The requirement is then to be indistinguishable in certain area and in an Euclidean metric it is simply the area of the circle with radius l, a function that is indeed quadratic in l.

5.2.3 Extracting location quality

Our definition of privacy mass depends on the semantic quality $q(x)$ of a point x. To compute the quality in a meaningful way, we used the OpenStreetMap database

[2] to perform geo-localized queries. The open license ODbL of the database allows to download regional extracts that can then be loaded in a GIS database (Postgresql+PostGIS in our case) and queried for a variety of geo-located features. The data in many urban areas is extremely fine grained, to the level of buildings and trees. Furthermore there is a great variety of mapped objects produced by more than 2 millions users.

In order to extract the quality of a cell $q(x)$, we perform several queries reflecting different privacy properties and we combine them in one aggregate number using different weights. In our experiments we query for a variety of Points Of Interest in the tag class `amenity`, such as restaurants and shops, and for the number of buildings in a cell. The buildings are an indication of the population density, in fact despite the database provides a `population` tag, it is for large census areas and with a scarce global coverage, while the `building` tag can be found everywhere and with fine resolution. Considering the simple nature of the queries performed we believe the resulting grid captures very well the concept of hiding in the crowd that we wanted to achieve (a sample can be viewed in Figure 5.3a). We leave more complex query schemes as future work as the main focus of this work is on the metric construction, described in detail in Section 5.2.4.

Future directions. Among possible improvements three directions seem promising. First, one strength of $d_{\mathcal{X}}$-privacy is that it is independent of prior knowledge that an attacker might have about the user, making the definition suitable for a variety of users. However in some cases we might want to tailor our mechanism to a specific group of users, to increase the performance in terms of both privacy and utility. In this case, given a prior probability distribution over the grid of locations, we can use it to influence the privacy mass of each cell. For instance, if we know that our users never cross some locations or certain kind of POIs, we can reduce their privacy mass.

Second, we are interested in queries that reward variety other that richness e.g. a location with 50 restaurants should be considered less private than one with 25 restaurant and 25 shops.

Finally, different grids could be computed for certain periods of the day or of the year. For instance, our user could use the map described above during the day, feeling private in a road with shops, but in the evening only a subset of the tags should be used as many activities are closed, making a road with many restaurants a much better choice. The same could be applied to seasons, imagine for example

[2] http://www.openstreetmap.org

how snow affects human activities in many regions.

Once we have enriched every location x with a quality $q(x)$, we can compute the resource function $m(x)$ as described previously. In the next part we describe how to exploit this rich and customizable information to automatically build an elastic metric satisfying a requirement req.

5.2.4 An efficient algorithm to build elastic metrics

In this section we develop an efficient algorithm to compute a distinguishability metric $d_{\mathcal{X}}$ that satisfies the quadratic requirement defined before. The metric we produce is induced by an undirected graph $\mathcal{G} = (\mathcal{X}, E)$, that is the main structure manipulated by the algorithm, where vertices are locations and edges (x, d, x') are labeled with the distance between locations. The distance between two locations is the shortest path between them and thanks to this property instead of computing $|\mathcal{X}|^2$ edges, we can actually keep just a subset and derive all other distances as shortest paths.

Like shown in Fig 5.1, we start with a fully disconnected graph where all distances are infinite (thus each location is completely distinguishable) and we start adding edges guided by the requirement function. We work in *iterations* over the grid, where at each iteration we add only one edge per vertex, stopping when req is satisfied for all vertices. The reason to work in iterations is that even if at iteration i a vertex can only reach a certain number of cells, because of the other edges added during the same iteration, at $i + 1$ it will find itself connected to many more vertices. This approach distributes edges uniformly which provides two main advantages. First, it increases the locality of connections which in turn reduces the average error (or increases the utility) of the resulting mechanism. Second, it leads to a smaller number of edges, thus decreasing the size of the graph.

The requirement function $\mathrm{req}(l)$ is used as a guideline to define the edges. Let $\mathrm{req}^{-1}(m) = l^*\sqrt{m}$ be the inverse of req.[3] This function tells us at what distinguishability level $l = \mathrm{req}^{-1}(m)$ we should find m amount of privacy mass in order to satisfy the requirement. For each location x we keep a temporary level l_x that is updated at each iteration using req^{-1} and stops at a predefined maximum

[3]Note that our algorithm is not tied to the specific quadratic requirement function, it can work with an arbitrary function req. Even if $\mathrm{req}(l)$ is not invertible (e.g. for a step-like requirement), we could use $\mathrm{req}^*(m) = \inf\{l \mid \mathrm{req}(l) \geq m\}$ in place of req^{-1}.

value $l^\top$. At the beginning l_x is set using only the privacy mass provided by x alone but adding edges will take into account also the ball of points reachable within l_x. In other words the temporary level of each location indicates up to what level of distinguishability the requirement is satisfied.

We then start the iterations and for each vertex x that hasn't already reached $l^\top$ we recompute an updated l_x. The update is necessary to take into account other connections that may have been added for other vertices and that could increase the ball of x. In order to add a new edge we need a strategy to find a candidate vertex x' to connect to. The strategy we employ is `next-by-geodistance`, that returns the cell x' geographically closest to x, but still not visited. In the resulting metric locations are more indistinguishable to nearby locations, reducing the average error of reported points. Once we have a candidate location x' there are two possible situations. If the distance between x and x' is greater than l_x, we need to lower it to satisfy the requirement, so we add an edge (x, l_x, x'). Otherwise if the distance is shorter or equal than l_x, this means that x' is already in the l_x ball of x, so we ask `next-by-geodistance` for another candidate. For each vertex not completed an edge is added to the graph and the process is repeated in iterations until all locations reach $l^\top$.

Our experiments showed that completion of the last few tens of vertices can take extremely long and they are localized mostly on the border of the grid. This is due to the fact that points close to the border have fewer neighbors, making it harder for them to find a candidate to connect to. As a consequence they need to reach much further away, taking more iterations and resulting in a higher average error because of the long connections created. For this reason we use a stopping condition that checks, at the end of every iteration, if all the nodes remaining to complete are closer to the border than a certain *frame* constant. If they are, the algorithm stops without completing their requirement. All locations inside this frame of the grid can be reported as sanitized locations but cannot be used as secret locations. The *frame* value is a compromise between the algorithm running time and usable grid size, in our experiments we used 3% of the grid size.

It can be shown that the metric $d_{\mathcal{X}}$ constructed by this algorithm does satisfy the requirement req for all $l \leq l^\top$. The stopping level $l^\top$ can be set arbitrarily high, but in practice setting it to any value larger than 10 has no effect on the resulting metric. As shown in the evaluation of Section 5.3, the algorithm can scale to an area of half a million locations with modest computing resources. This is several orders of magnitude better that techniques computing optimal obfuscation mechanisms [STT+12, BCP14, Sho14] which can only handle a few hundred

```
foreach x ∈ 𝒳 do
  l_x := req^{-1}(m({x}))
while ∃x. l_x ≠ l^⊤ do
  foreach x ∈ 𝒳 do
    l := req^{-1}(m(B_{l_x}(x)))
    do
      x' := next-by-geodistance(x)
    while d_𝒳(x, x') ≤ l_x
    E := E ∪ {(x, l_x, x')}
```

Figure 5.1: Pseudo-code description of the algorithm core.

points within reasonable time constraints. Of course, our method gives no optimality guarantees, it only constructs one possible metric among those satisfying the requirement.

Future directions. We believe further improvements in performance are possible in three directions, that we leave as future work. When working in privacy poor areas, like in the country, the algorithm spends a considerable time compressing large areas, as expected. We believe that this work could be avoided by grouping together several locations already when laying down the grid. We would have a coarser resolution in the country, which is acceptable, and a large speed up in the metric construction. A second obvious improvement would come from running the algorithm in parallel on portions of the map and merging the results. The problem arises on the borders of the submaps, where on adjacent locations we have connections with sharply different shapes. We believe however that computing several submaps leaving a frame of uncompleted points and them running again the algorithm in the entire map could provide a reasonable result. Finally several strategies can be applied in the choice of the next candidate and in the way we perform the iterations that could have an impact on speed and utility of the mechanism by completing faster the requirement.

Practical considerations

We believe that the techniques presented are practical enough to deploy a location privacy mechanism in a real setting. The resources required both in terms of hardware and time are very limited, consider that the mechanism evaluated in the next section was built in a day on a medium Amazon E2C instance. As already

mentioned querying the database is easily parallelizable and the same grid can be reused to build several metrics. We could imagine having a choice of pre-computed grids, for different flavors of location quality and times (as explained in Sec 5.2.3), on top of which the user could tune the requirement.

The computation of the metric is the most demanding part of the process but proved reasonably fast for an area that can easily contain all the movements of the user and many optimization are still possible in the algorithm. We imagine these two computationally intensive activities to be performed by a remote server, with seldom updates from the OpenStreetMap database.

The user's phone needs to take care only of downloading an extract of the metric to use in the Exponential mechanism. For every request, the mechanism computes the exponential distribution of sanitized locations and draws from it, which amounts to a trivial computation, both in memory and time. In principle we could also avoid contacting a third party by saving the entire metric locally on the phone, as a reference our metric for Ilê de France is only 58MB.

From a user's perspective, the amount of configuration required to run the mechanism varies according to her needs. It can go from no configuration, in the case she downloads a pre-computed map, to scaling the privacy mass requirement, to full customization in the case the users wants to tailor the queries to her specific needs.

Compatibility with the fenced metric

It should be noted that fences are applicable to any distinguishability metric, including but not limited to the elastic metric presented in Sec 5.2. Not only we can incorporate fences in our elastic definition but also in our graph based algorithm, in a simple and efficient way. It amounts to connecting the locations inside the fence with zero labeled edges and to leave them disconnected from the nodes outside. When running the algorithm we should also take care to maintain this disconnection of the fence. In order to avoid adding edges from the inside we simply set the temporary radius r_x of all nodes inside the fence to $d^\top$, so that the algorithm considers them completed and skips them. On the other side, to avoid adding edges from the outside, we need to modify the function `next-by-geodistance` so to avoid considering a candidate any location inside a fence. Both alterations to the algorithm are trivial to implement and have no effect on performances. The only drawback of the presented method is that the fences need to be set before building the metric, which is inconvenient as for each

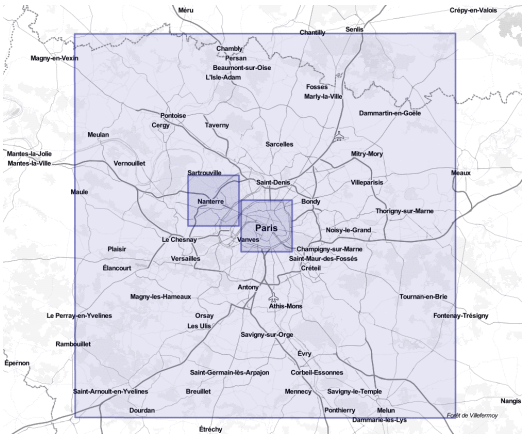


Figure 5.2: Coverage of EM with two subregions: Nanterre suburb on the left and Paris city on the right

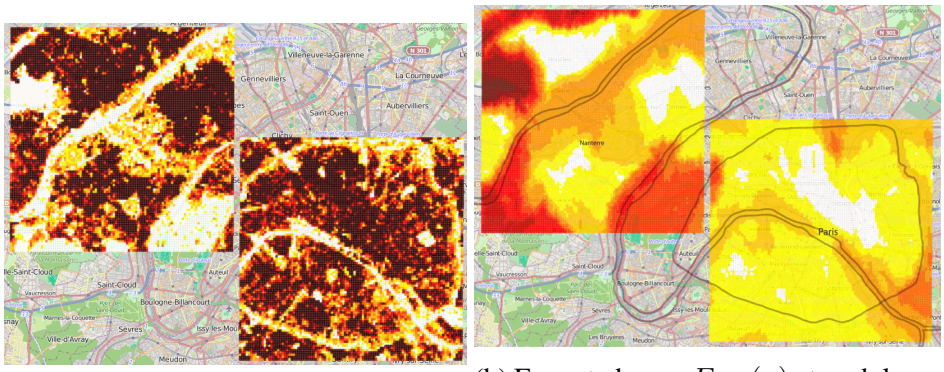
user we are obliged to recompute, for the most part, the same metric. Despite this our experiments show that in under a day is possible to generate the metric so this remains an effective technique for practical purposes, especially considering the very static nature of the fences.

Related work

Regarding the construction of finite mechanisms, [Sho14] proposes a linear programming technique to construct an optimal obfuscation mechanism with respect to either the expectation of distance error or geo-indistinguishability. In [BCP14] the authors propose again a linear programming technique to compute a geo-indistinguishable mechanism with optimal utility. Their approach uses a spanner graph to approximate the metric in a controlled way. Our algorithm does not provide optimality with respect to privacy nor utility, it guarantees the respect of a privacy requirement while achieving good utility. Moreover the state of the art in optimal mechanism construction is limited to few tens of locations while the purpose of our technique is to scale to several thousands of points.

5.3 Evaluation

In this section we perform an extensive evaluation of our technique in Paris' wide metropolitan area, using two real-world datasets. We start with a description of the metric-construction procedure, and we discuss the features of the resulting metric as well as the obfuscation mechanism obtained from it. Then, we compare the



(a) Privacy mass $m(x)$ of each location

(b) Expected error $E_{\mathrm{EM}}(x)$ at each location

Figure 5.3: Paris' center (right) and the nearby suburb of Nanterre (left)

elastic mechanism to the Planar Laplace mechanism satisfying geo-indistinguishability, using data from the Gowalla and Brightkite social networks. The comparison is done using the privacy and utility metrics of [STBH11]. It should be emphasized that the metric construction was completely independent from the two datasets, which were used only for the evaluation. All the code used to run the evaluation is publicly available [ela].

5.3.1 Metric construction for Paris' metropolitan area

We build an elastic metric $d_{\mathcal{X}}$ for a 75 km $\times$ 75 km grid centered in Paris, roughly covering its extended metropolitan area. Each cell is of size 100 m $\times$ 100 m, and the set of locations $\mathcal{X}$ contains the center of each cell, giving a total number of $562,500$ locations. The area covered is shown in Fig 5.2; note that the constructed metric covers the larger shown area, the two smaller ones are only used for the evaluation of the mechanism in the next section.

The semantic quality $q(x)$ of each location was extracted from OpenStreetMap as explained in Section 5.2.3, and the privacy mass $m(x)$ was computed from (5.1) using $r_{\text{small}} = 300$ m and $r_{\text{large}} = 3$ km. The resulting mass of each location is shown in Figure 5.3a, where white color indicates a small mass while yellow, red and black indicate increasingly greater mass. The figure is just a small extract of the whole grid depicting the two smaller areas used in the evaluation: central Paris and the nearby suburb of Nanterre. Note that the colors alone depict a fairly clear picture of the city: in white we can see the river traversing horizontally, the main ring-road and several spots mark parks and gardens. In yellow colors we find low density areas as well as roads and railways while red colors are present

in residential areas. Finally dark colors indicate densely populated areas with presence of POIs.

For this grid, we use the algorithm presented in Section 5.2.4 to compute an elastic metric $d_{\mathcal{X}}$ with the quadratic requirement of (5.2), configured with $l^* = \ln 2$. The whole computation took less than a day on an entry-level Amazon EC2 instance. This performance of the algorithm is already sufficient for real-world use: the metric only need to be computed once, while the computation can be done by a server and the result can be then transmitted to the user's device. Note that the algorithm can deal with sizes several orders of magnitude bigger that techniques computing optimal obfuscation mechanisms [STT+12, BCP14, Sho14], which makes it applicable to more realistic scenarios.

We then construct an Exponential mechanism (described in Section 3.1) using $d_{\mathcal{X}}$ as the underlying metric. We refer to the resulting obfuscation mechanism as the Elastic Mechanism (EM). The mechanism is highly adaptive to the properties of each location: high-density areas require less noise than low-density ones to achieve the same privacy requirement. Figure 5.3b shows our utility metric per location, computed as the expected distance $E_{\mathrm{EM}}(x)$ between the real and the reported location. Compared to Figure 5.3a it is clear that areas with higher privacy mass result to less noise. Populated areas present a good and uniform error that starts to increase on the river and ring-road. On the other hand, the large low-density areas, especially in the Nanterre suburb, have a higher error because they need to report over larger areas to reach the needed amount of privacy mass.

Finally, Figure 5.4 shows a boxplot of the expected error for each location in the two areas. It is clear that the amount of noise varies considerably, ranging from a few hundred meters to several kilometers. It is also clear that locations in central Paris need considerably less noise that those in the suburban area. For comparison, the Planar Laplace mechanism (compared against EM in the next section) has a constant expected error for all locations.

Note that the expected error will always be higher than the r_{small} used in the normalization. For example in a location that satisfies its requirement in 300 m it would be 870 m. This is expected and it is due to the nature of the exponential noise added.

5.3.2 Evaluation using the Gowalla and Brightkite datasets

In this section we compare the Elastic Mechanism (EM) constructed in the previous section with the Planar Laplace mechanism [ABCP13] satisfying standard

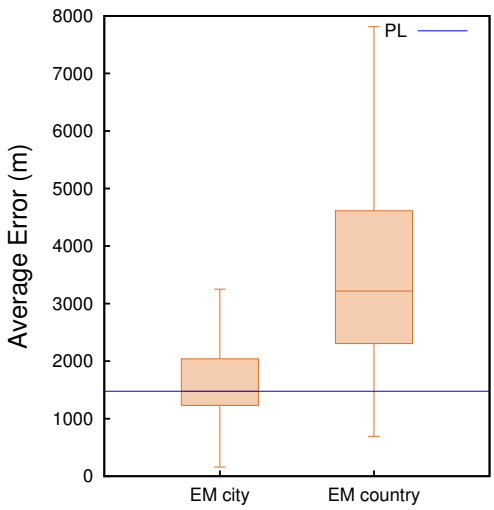


Figure 5.4: Expected error $E[d_2]$ per location

geo-indistinguishability. For the evaluation we use two real-world datasets from location-based social networks.

The Gowalla and Brightkite datasets. Gowalla was a location-based social network launched in 2007 and closed in 2012, after being acquired by Facebook. Users were able to "check-in" at locations in their vicinity, and their friends in the network could see their check-ins. The Gowalla dataset [CML11] contains $6,442,890$ public check-ins from $196,591$ users in the period from February 2009 to October 2010. Of those, $9,635$ check-ins were made in Paris' center area and 429 in the Nanterre area (displayed in Fig 5.2).

Brightkite was another location-based social network created in 2007 and discontinued in 2012. Similarly to Gowalla users could check-in in nearby locations and query who is nearby and who has been in that location before. The Brightkite dataset [CML11] contains $4,491,143$ check-ins from $58,228$ users. Of those, $4,014$ check-ins were made in Paris' center and 386 in Nanterre.

These datasets are particularly appealing for our evaluation since a check-in denotes a location of particular interest to the user, and in which the user decided to interact with an actual LBS. This is in sharp contrast to datasets containing simply mobility traces, which just contain user movements without any information about the actual use of an LBS.

Privacy metrics. Since the EM and PL mechanisms satisfy different privacy definitions, to perform a fair comparison we employ the widely used Bayesian

privacy metric presented in 2.4.5: *adversarial error*.

$$\textsc{AdvError}(K, \pi, h, d_A) = \sum_{x,z} \pi(x)\ K(x)(z)\ d_A(x, h(z))$$

where d_A is the loss function of the attacker and h his remapping strategy.

In our evaluation the secrets are POIs in each dataset. We use the two commonly employed loss functions presented in the Preliminaries 3.1 to model different attackers: d_{bin} and d_2.

For the binary function we have that $\textsc{AdvError}(K, \pi, h, d_{\text{bin}})$ expresses the adversary's *probability of error* in guessing the user's *exact* POI. On the other hand, the euclidean function models an adversary who is interested in guessing any POI *close* to the user's. In this case, $\textsc{AdvError}(K, \pi, h, d_2)$ gives the adversary's *expected error* in meters in guessing the user's POI.

We should emphasize an important difference between the two adversaries: d_{bin} tries to extract semantic information from the actual POI, and is less effective in dense areas where the number of POIs is high. On the other hand, d_2 is less sensitive to the number of POIs: if many POIs are close to each other, guessing any of them is equally good. This difference is clearly visible in the evaluation results.

We use each dataset to obtain a prior knowledge π^* of an "average" user of each social network, by considering all check-ins within the areas of interest. Note that the datasets do not have enough check-ins *per-user* to construct *individual* user profiles. Indeed, most users have checked-in in each location at most once, hence any profile built by $n - 1$ check-ins would be completely inadequate for inferring the remaining one. As a consequence, we assume that the adversary will compute his best strategy h^* by using the global profile π^*.

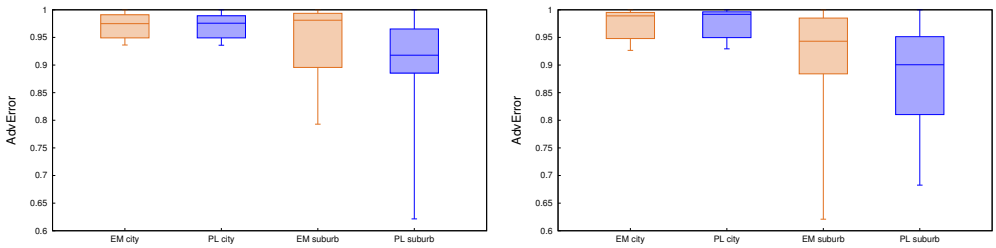
Finally, we use

$$\textsc{AdvError}(K, \pi_u, h^*, d_A) \qquad d_A \in \{d_{\text{bin}}, d_2\}$$

as our privacy metric, where π_u is the user's individual prior (computed only from the user's check-ins), and

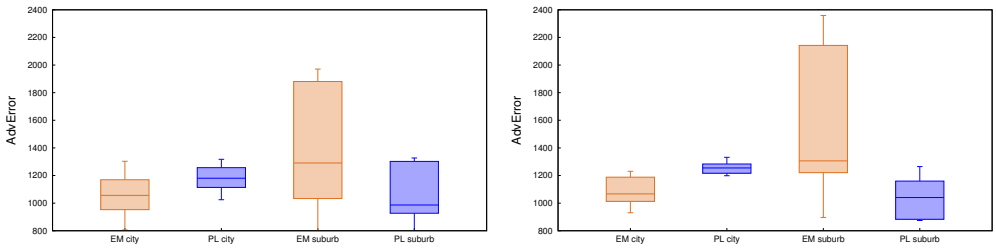
$$h^* = \arg\min_h\ \textsc{AdvError}(K, \pi^*, h, d_A)$$

is the adversary's strategy computed from the global profile. Hence, the individual priors π_u are only used for averaging and not for constructing the strategy.



(a) Gowalla (b) Brightkite

Figure 5.5: Per-user binary ADVERROR of the EM and PL mechanisms for each area



(a) Gowalla (b) Brightkite

Figure 5.6: Per-user Euclidean ADVERROR of the EM and PL mechanisms for each area

Utility metrics. The utility of an obfuscation mechanism is in general closely tied to the application at hand. In our evaluation we want to avoid restricting to a particular one; as a consequence we use two generic utility metrics that are reasonable for a variety of use cases. We measure utility as the *expected distance* between the real and the reported locations, using the two metric introduced in the Preliminaries 3.4. In some applications, service quality degrades linearly as the reported location moves away from the real one; in such cases the Euclidean distance d_2 provides a reasonable way of measuring utility. Other applications tolerate a noise up to a certain threshold r with almost no effect on the service, but the quality drops sharply after this threshold. In this cases we use the threshold metric d_r.

Results. We carry out our evaluation in two different areas of the Paris metropolitan area, with very different privacy profiles. The first area is the city of Paris intra-muros, very private on average, while the second is the adjacent suburb of Nanterre, where already the concentration of privacy mass is much lower.

To obtain a fair comparison, the Planar Laplace PL mechanism is configured

to have the same utility as the EM mechanism in Paris' center. We computed the utility of EM using the global profile π^* from both datasets, using four distance functions, namely d_2 and d_r with $r = 1200, 1500$ and 1800 meters. In each case, we computed the parameter ϵ of PL that gives the same utility; we found that in all 8 cases the ϵ we obtained was almost the same, ranging from 0.001319 to 0.001379. Since the difference between the values is small, we used a single configuration of PL with the average of these values, namely $\epsilon = 0.001353$. With this configuration, we then compare the two mechanisms' privacy in both areas.

Note that, although we configured the *expected* utility of both mechanisms to be the same in the city, the EM's behavior is highly dynamic: in the most private areas of Paris EM uses much less noise: in 10% of the city's locations PL adds 50% or more noise than EM.

The results for the binary adversarial error are shown in Figure 5.5. We can see that in the center of Paris both mechanisms provide similar privacy guarantees. The dynamic nature of PL does not affect its privacy: locations in which the noise is lower have more POIs in proximity, hence even with less noise it is still hard to guess the actual one. On the other hand, in the suburb there is a sharp degradation for PL. The reason is that the number of POIs in both datasets is much smaller in Nanterre than in Paris, the distance between them is greater, and the resulting priors are more "informed". Hence it is considerably easier for the adversary to distinguish such POIs, leading to a probability of error as low as 0.62. On the other hand, EM maintains a higher ADVERROR in the suburb, by introducing a higher amount of noise.

To match EM's privacy guarantees in the suburb, PL should be configured with a higher amount of noise. However, since Planar Laplace treats space in the same way everywhere, this would lead to a high degradation of utility in Paris' center, which is unfortunate since (i) the extra noise is unnecessary to provide good privacy in the center, and (ii) the extra noise could render the mechanism useless in a dense urban environment where accurate information is crucial. In short, the flexibility of the elastic mechanism allow it to add more noise when needed, while offering better utility in high-density areas.

The results for the Euclidean adversarial error are shown in Figure 5.6. Here, we see a sharp difference with respect to the binary adversary: the effectiveness of guessing a POI *close* to the real one is not affected much by the number of POIs (guessing any of them is equally good). As a consequence, EM, which adds less noise in dense areas with a great number of POIs, scores a lower adversarial error in the city (although the difference is moderate). The median error for EM in

Gowalla is 1056 meters while for PL it is 1180 meters. The motivation behind PL was that, in a dense urban area, it is harder for the adversary to extract semantic information from the reported location even if less noise is used. But the Euclidean adversary is not interested in semantic information, so he scores better with less noise.

In the suburb, on the contrary, the picture is reversed. Due to the fact that priors in Nanterre are more "informed" making remapping easier, the adversarial error for PL decreases compared to the city. On the other hand, EM adapts its noise to the less dense environment, leading to much higher adversarial error.

Note that the Nanterre area is still quite close to the city center and highly populated itself. The fact that we can already see a difference between the two mechanisms so close to the center is remarkable; clearly, the difference will be much more striking as we move away from the city center (unfortunately, the datasets do not contain enough data in these areas for a meaningful quantitative evaluation).

Finally, we should emphasize that the mechanism's construction and evaluation were completely independent: no information about Gowalla's or Brightkite's list of check-ins was used to construct the metric. The only information used to compute $d_{\mathcal{X}}$ was the semantic information extracted from OpenStreetMap.

5.4 Tiled Mechanism

The extreme flexibility of the elastic mechanism, which can change its behavior for locations just 100 meters apart, comes with the cost of a heavy phase of preprocessing to build its semantic map. However even for low end applications we would still like some degree of flexibility. One such application, and the main motivation behind this work, is Location Guard (see Chapter 6), a browser extension that implements geo-indistinguishability for the Geolocation API. In the current stable version, Location Guard implements a simple Planar Laplace mechanism where the user can configure the ϵ parameter. Our goal is to provide some of the flexibility of the elastic mechanism in the constrained environment of a browser extension, where the processing footprint should be minimal and especially disk space is extremely reduced.

What we would need is a simplified version of the elastic mechanism where the noise level is adapted to large areas, that we call *tiles*, small enough to distinguish a park from a residential area, but still easily computable. In order to build

these tiles, we can query a number of online geographical services, to obtain a set of polygons together with a quantitative description of the amount of privacy they provide, the privacy mass. This dataset should cover an area large enough to contain most of the user usual movements, a few tens of kilometers, while retaining a small size. Once this small dataset is built, we have a mapping from tiles to their privacy mass. We can now use it to define a function `find` that for each location, finds the containing tile i and returns a privacy level ϵ_i adapted to its privacy mass. We can then use a standard Planar Laplace mechanism to sanitize. The idea is that if the noise is scaled with the right ϵ_i, the user would be indistinguishable from a space containing enough privacy mass.

The mechanism described above, despite achieving the flexible behavior we needed, does not satisfy geo-indistinguishability. It is enough to note that the level of protection, a public information of the mechanism, depends on the current location of the user, which is sensitive: There is clearly a leakage of information. More precisely, if we pick two locations very close to the border of two tiles, despite the distance between the locations is small the distributions of their sanitized versions would be very far apart, namely from two mechanisms configured with different epsilons.

In order to solve this problem we need to make `find` itself differentially private; a simple way to do it is to first sanitize the current location with a fixed privacy level and then feed it to `find`. Post processing a sanitized location does not pose any threat to privacy and would allow the mechanism to reduce sharply the amount of noise added to locations in very private areas.

The mechanism

We are currently evaluating the use of two online services to build the dataset. In order to extract the geographic polygons of administrative areas and natural features (such as lakes and rivers) we are using OverPass Turbo [4], that provides a simple API to query the OpenStreetMap database. Once obtained the array of tiles `tiles` we use again Overpass Turbo to query for POIs densities and DBpedia [5] to obtain population densities from Wikipedia. These two values are combined in a privacy mass measure of each tile and converted in a suitable ϵ value, stored in the attribute `tile.epsilon`. Preliminary results are shown in Figure 5.8 for New York City and Paris.

[4] http://overpass-turbo.eu/
[5] http://dbpedia.org/

```
function tiled(x){
    var z = planarLaplace(epsilonFixed,x);
    var tile = tiles.find(z);
    return planarLaplace(tile.epsilon,x);
}
```

Figure 5.7: Pseudo javascript code for tiled mechanism.

In the pseudo code in figure 5.7, the secret location x is first sanitized by a Planar Laplace mechanism configured with a *fixed* value of epsilon. The sanitized location is used to locate the user's current tile and its correspondent ϵ. Finally the latter is used to sanitize again the user's real location and report it.

Privacy and Utility

We are still in the process of understanding how the privacy mass should be estimated for a tile and what form of distinguishability metric the mechanism would provide exactly. Regarding budget usage, at each usage we have to pay for the fixed epsilon plus the local epsilon of the tile. The location sanitized with the fixed epsilon can be very noisy as we are only trying to establish the tile where the user is located. The cost of this small epsilon can be worth if there is great variety among tiles, namely in private tile we would be able to safely reduce the amount of noise added. If however the tiles extracted are very uniform it would be more convenient to revert to a simple Planar Laplace with a uniform epsilon.

We are currently in the process of evaluating more precisely the tiled mechanism to establish when it is a viable choice. Furthermore we are working on the engineering aspects of the tiles extraction from public online services, finding a balance between correct characterization and performance of the queries.

5.5 Conclusion

In this chapter, we have developed a novel elastic privacy metric that allows to adapt the privacy requirement, and hence the amount of applied noise, to the properties of each location. We have formalized a requirement for such metrics based on the concept of privacy mass, and using semantic information extracted from OpenStreetMap we were able to model locations with variable density and protection. We have developed a graph-based algorithm to efficiently construct

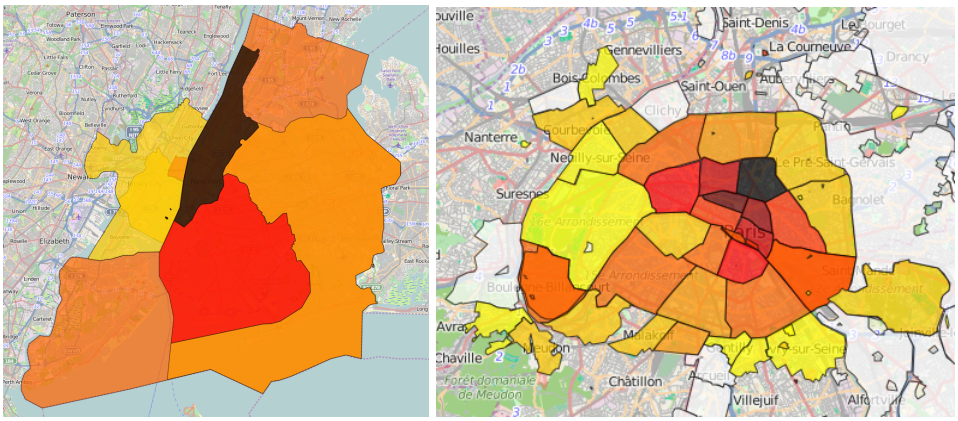


Figure 5.8: Tiles computed for New York and Paris

a metric satisfying this requirement for large geographical areas. The resulting mechanisms offer a more consistent protection for the user along his movement across different areas. Finally, we have performed an extensive evaluation of our technique in Paris' wide metropolitan area, using two real-world datasets from the Gowalla and Brightkite social networks. The results show that the adaptive behavior of the elastic mechanism can offer better privacy in low-density areas by adjusting the amount of applied noise. Lastly we proposed the tiled mechanism, that offers some degree of adaptability with the advantage of negligible computational cost making it an ideal candidate for inclusion in the browser extension Location Guard.

Chapter 6

Location Guard

Despite the many theoretical advancements in the field of location privacy, there is still a lack of practical options for the average user to protect her privacy while using a Location Based Service. Geo-indistinguishability has both the advantage of providing a formal and user independent privacy guarantee, and at the same time have a simple and efficient mechanism to achieve it. The Planar Laplace can be configured with one single parameter ϵ that gives a simple intuition of the trade off between privacy and utility. Because of these good properties it was a perfect candidate to try to implement a Privacy Enhancing Technology that could be used by non-technical users, *Location Guard*. The aim was on one side to test the practical *impact* of our research on formal methods for privacy and on the other to understand the reaction of the users.

Location Guard is an open source [loc] web browser extension that provides location privacy when using the HTML5 geolocation API. The privacy notion enforced by Location Guard is geo-indistinguishability, provided through a Planar Laplace mechanism (see Chapter 3.1). The extension has reached considerable popularity since its release, covering Chrome, Firefox and Opera browsers, and more recently moving to mobile devices with Firefox for Android. The next step for the project is to incorporate ideas from the techniques developed in this thesis and to reach for new mobile platforms.

6.1 A web browser extension

When building a privacy tool, we are faced with a choice of platform. Including location privacy in a single application has the advantage of allowing fine tuning

of both privacy and utility and in general great flexibility. For example imagine a search application that retrieves close by Point Of Interest by querying a LBS. If it is the application itself to perform the sanitization, then when receiving the search results from the LBS, it can filter out the results too far away from the user's real location because of the noise. However we are forced to re-implement every service, with a *privacy-by-design* approach, and only privacy-aware users will care to run them. On the other side if we target the operating system, we can enforce location privacy across the user's whole range of applications and provide her with stronger guarantees. The latter approach however limits us to those users willing to run a custom, privacy friendly, version of their operating system, which in general is even harder than the first approach.

The Web browser is by far the most used interface between users and privacy sensitive services nowadays; its growth in popularity as a *platform* to develop applications makes it an ideal target for a privacy preserving tool. Browser extensions, allowed to run code with limited privileges in order to offer new functionalities – e.g. ad-blockers, search-engines – are becoming increasingly popular. For our purposes it is possible to intercept calls to the Geolocation API, sanitize the original location provided by the browser, then return a private version to the calling application in a transparent way. A browser extensions allows to incorporate privacy in a great number of services, in a way that is familiar and easy to install.

6.2 Desktop and mobile

Despite the fact that location privacy is considered especially important in mobile devices, the accuracy of modern wifi-based location providers is detrimental also to desktop users: the position of a wifi connected laptop can often be determined more accurately than a mobile phone with GPS. Users awareness of the problem is shown by the popularity of the first versions of Location Guard, that were limited to desktop browsers.

Since release 1.2.0 (February 2015) Location Guard runs on Firefox [1] for Android, currently the only mobile browser supporting extensions, and its mobile user base has been growing rapidly. Supporting mobile devices is crucial since they typically follow all users' movements, while mobile-optimized websites ask for user's location increasingly often.

Although on smartphones native apps are the most popular way to interact

[1] https://addons.mozilla.org/en-US/firefox/addon/location-guard/

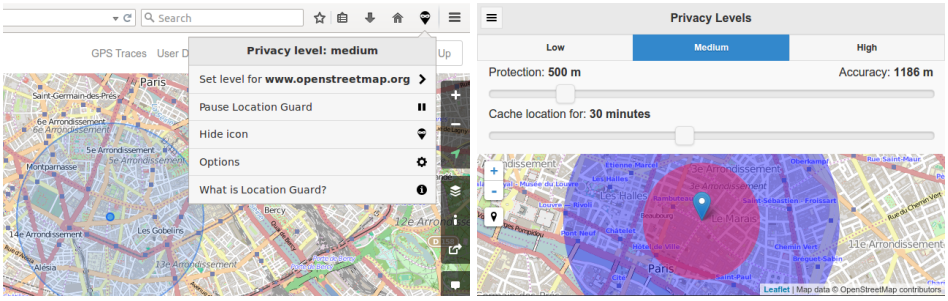
with online services, the growing popularity of web applications, to contrast the rampant fragmentation of mobile development, promise a larger coverage of services for Location Guard in the near future. Furthermore Mozilla announced a new port of Firefox to iOS in the next year thus covering the other half of the mobile space.

6.3 Operation

Every web application runs in a separate environment and can access privileged information, such as the user's location, through a JavaScript API provided by the browser. A browser extension has the ability to run JavaScript code with higher privileges than a normal page and, among other things, to modify the content of any web page. When a page is loaded and before any other code is executed, Location Guard injects a small snippet of JavaScript that redefines `geolocation.getCurrentPosition`, the main function provided by the Geolocation API to retrieve the current position. When the rest of the page code runs and tries to access this function, it gets intercepted by Location Guard, which in turn obtains the real location from the browser, sanitizes it and returns it to the page.

The location is sanitized through the use of random noise drawn from a Planar Laplace distribution. The amount of noise added can be configured easily with a single parameter, the privacy *level* ϵ. Location guard provides three predefined levels {high,medium,low} and the user is also free to pick any other value. Additionally the privacy level can be adjusted per domain, so that different protection can be applied to different services: a larger amount of noise can be added to a weather service as opposed to a point of interest search engine.

An advantage of geo-indistinguishability is that it is relatively intuitive to explain to the user the effect of changing the levels on privacy and utility. For a certain privacy level we can compute two radiuses r_p and r_u, respectively the radius of privacy protection and of utility. r_p is the area of locations highly indistinguishable from the actual one, i.e. all locations producing the same sanitized one with similar probabilities. r_u is the area in which the reported location lies with high probability, thus giving an idea of the utility that the user can expect. Both these radiuses can be easily plotted on a map to give the user a direct impression of privacy and utility, according to the level of protection chosen. Figure 6.1a displays the main menu of the extension to select one of the predefined



(a) Main menu. (b) Level configuration.

Figure 6.1: Location Guard running on Firefox.

levels for a specific domain, in this particular instance the level `medium` is set for `www.openstreetmap.org`. Additionally the user can configure its own level (Figure 6.1b) and inspect the effect on an interactive map. In pink circle represents the area where the user is highly indistinguishable while the purple area shows the area where the reported location will fall with high probability (as the $\alpha(0.9)$-accuracy presented in Chapter 3.1).

Apart from sanitizing the real location, Location Guard supports reporting a *fixed* predefined location, which offers perfect privacy at the cost of very low utility.

6.4 Adoption

Location Guard has witnessed wide adoption with close to zero publicity. For the first year and half there was a steady and slow growth due only to users searching explicitly for location privacy extensions in the Firefox and Chrome websites. Only in the past 6 months, with a few independent blog posts and other online reviews, enough momentum was gathered to generate a true boom of installations. Particularly the extension has been selected as Pick of the Month of June 2015 in the Firefox Addons website, meaning a constant display of the addon in the website main page. This publicity is the cause of the spike of installation displayed in Figure 6.3.

Additionally the extension has enjoyed a regular growth in Chrome and Opera as well. In Table 6.2 we can see how the user base is spread across browsers, note that all the numbers are active users except for Opera where only the number of downloads is available.

Browser	Users
Chrome	10,397
Firefox	30,980
Firefox for Android	1,367
Opera	5,173*

Figure 6.2: User adoption per browser.

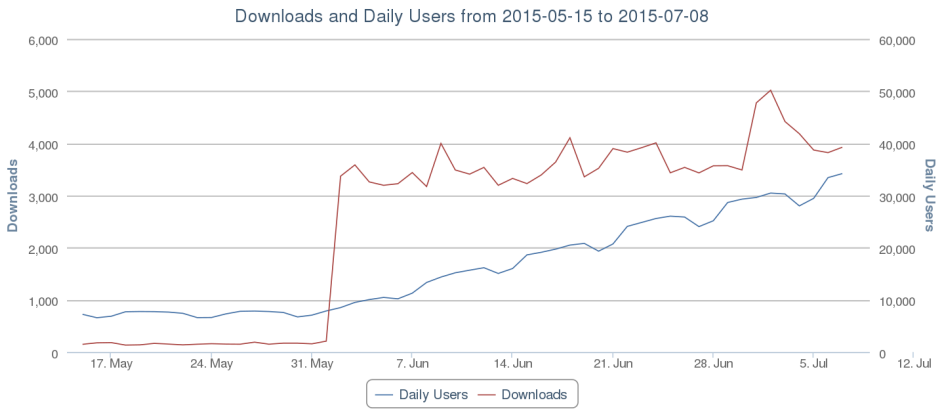


Figure 6.3: Firefox Daily Users (blue) and Downloads (red).

6.5 Future Work

In the immediate future three main developments are planned for Location Guard.

Data Analysis. One first addition will be the possibility for the user to log locally all the activity of Location Guard, which means all reported locations for each domain. This information will be used in two ways. First we user will be able to inspect on a map all his real and reported locations, thus having an insight of his mobility model and how it is affected by the extension. Secondly we will run some simple attacks (see Chapter 3.1) in order to show the users what an LBS might learn from his reported locations. Note that every manipulation of the data will be done locally to protect the privacy of the user. Some of the computed values will be anonymized and collected with the user consent, in order to obtain more information on our user base, such as how frequently the extension is used or if the users change often the levels. None of the original logged data will be collected.

Richer mechanisms. The second planned extension is to evaluate the inclusion of more mechanisms presented in this thesis. Other than the technical implemen-

tation it is a priority of the extension to be understandable for the user and the addition of any feature needs to be presented in a clear and user friendly way.

Particularly easy to present to the user would be the use of fences. However there are the engineering challenges described in Section 4.4, regarding the automatic suggestion of position and size. This would go in the same direction as the data analysis, we could run a clustering attack on the local data to raise the user awareness of his location privacy and then suggest the fences as a solution.

The parrot prediction doesn't present any technical difficulty, already in Location Guard there is a simple caching in place to avoid reporting too often the same location. In this case the issue is how to manage the budget, how much should be allocated, if and when it should be reset. In order to address this issue we are planning to gather more insight from the data analysis, especially on the frequency with which the users report their location.

Finally the tiled mechanism is the more technically involved. Most of the complexity comes from the huge variety of the data that we get from OverpassTurbo and DBpedia. The variety comes from the large differences of each country in the way they map features and provide privacy mass as well as mapping quality of different areas. We need to establish a small and efficient set of queries that provide a consistent result in locations where the data quality is good or detect if the quality is not sufficient to continue.

New platforms. Like explained in the introduction the web browser is a good compromise between ease of use and impact on applications. However being able to work at the operating system level would allow us to ensure a more consistent protection to the user. Especially on mobile, Location Guard can sanitize only locations released though Firefox Mobile, while the user may leak his location though a variety of native apps.

The first mobile platform that we investigating to port Location Guard is Firefox OS. Being based on web technologies, Firefox OS presents the least amount of technical work and given that Mozilla has made of privacy one of its pillars there should be interest for inclusion of an extension that is proving popular among Firefox user base.

However Firefox OS is still in an early stage of adoption and Android still remains the most wide spread mobile platform. The problem of porting Location Guard to Android would be the need to *root* the device and install a custom version of the operating system, a *mod*. The operation is complex for the average user and it is huge engineering effort to realize and maintain. However there is already

a popular mod for Android phones called CyanogenMod[2] that targets more technical users, interested in additional privacy protections. In this case we are hoping to develop the port of Location Guard and involve the Cyanogenmod community in help maintaining it. The advantage of this solution would be to cover a large user base and to observe the behavior of actual mobile applications in use today.

[2]http://cyanogenmod.com/

Chapter 7

Conclusion

Geo-indistinguishability provides a solid foundation for the design of robust location privacy mechanism. Thanks to the metric notions of d_x-privacy we were able to extend geo-indistinguishability in order to tackle some shortcomings that limited its applicability in practice. We are able to greatly extend the use of the mechanism over *time* and to better tune it for our purposes. Furthermore by finely crafting our own distinguishability metric we are able to capture semantic features of locations and flexibly adapt the behavior of the mechanism over *space*. The efficacy of our techniques are demonstrated through experimental evaluation using real user dataset and realistic scenarios. The interest and practicality of our methods are also confirmed by the wide adoption of Location Guard, where our theoretical results are able to find a intuitive presentation of location privacy.

We are currently working on new prediction functions for the *predictive mechanism* to address new settings and improve budget consumption. Another priority is the correct and automatic configuration of *fences*, in order to perform an adequate and thorough evaluation of their effectiveness. Additionally we plan to study a more formal definition of *privacy mass* including ideas from l-diversity and find globally viable queries to extract this mass from OpenStreetMap. At the same time we plan to parallelize the graph-based algorithm to compute the *elastic* metric to scale it to larger areas. For the *tiled mechanism* we are working in a complete formalization of privacy and utility as well as the technical computation of the tiles. Regarding Location Guard, the main goal is to keep it a valuable tool for the users and at the same time an experimentation platform for our techniques. The compromise between the two provides interesting challenges to present our work in a clear and useful way for non-technical users. Finally we are interested in evaluating the robustness of our techniques to established attacks in the literature

and possibly obtain an alternative approach to determine the privacy parameter ϵ (more generally the indistinguishability level l^*).

It should be noted that even if our focus is mainly location privacy, thanks to the parametric nature of $d_{\mathcal{X}}$-privacy, most of the results and techniques we developed can be ported to different privacy fields.

Bibliography

[AACP11] Mário S. Alvim, Miguel E. Andrés, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. On the relation between Differential Privacy and Quantitative Information Flow. In Jiri Sgall Luca Aceto, Monika Henzinger, editor, *38th International Colloquium on Automata, Languages and Programming (ICALP 2011)*, volume 6756 of *Lecture Notes in Computer Science*, pages 60–76, Zurich, Switzerland, 2011. Springer.

[ABCP13] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS 2013)*, pages 901–914. ACM, 2013.

[ACD+07] Claudio Agostino Ardagna, Marco Cremonini, Ernesto Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati. Location privacy protection through obfuscation-based techniques. In Steve Barker and Gail-Joon Ahn, editors, *Proc. of the 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security (DAS)*, volume 4602 of *Lecture Notes in Computer Science*, pages 47–60. Springer, 2007.

[ACPS12] Mário S. Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Geoffrey Smith. Measuring information leakage using generalized gain functions. In *Proceedings of the 25th IEEE Computer Security Foundations Symposium (CSF)*, pages 265–279, 2012.

[Bal14] J. Ball. Angry birds and 'leaky' phone apps targeted by nsa and gchq for user data. The Guardian, January 2014. http://www.theguardian.com/world/2014/jan/27/nsa-gchq-smartphone-app-angry-birds-personal-data.

[BCP14] Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proceedings of the 21th ACM Conference on Computer and Communications Security (CCS 2014)*, 2014.

[BKS10] A. J. Bernheim Brush, John Krumm, and James Scott. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Proceedings of the 12th International Conference on Ubiquitous Computing (UbiComp 2010)*. ACM, 2010.

[BLPW08] Bhuvan Bamba, Ling Liu, Péter Pesti, and Ting Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *Proc. of the 17th International Conference on World Wide Web (WWW)*, pages 237–246. ACM, 2008.

[BWJ05] Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. Protecting privacy against location-based personal identification. In *Proceeding of the 2nd Workshop on Secure Data Management (SDM 2005)*. Springer, 2005.

[CABP13] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás E. Bordenabe, and Catuscia Palamidessi. Broadening the scope of Differential Privacy using metrics. In Emiliano De Cristofaro and Matthew Wright, editors, *Proceedings of the 13th International Symposium on Privacy Enhancing Technologies (PETS 2013)*, volume 7981 of *Lecture Notes in Computer Science*, pages 82–102. Springer, 2013.

[CAC12] Rui Chen, Gergely Ács, and Claude Castelluccia. Differentially private sequential data publication via variable-length n-grams. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS 2012)*, pages 638–649. ACM, 2012.

[CHM07] David Clark, Sebastian Hunt, and Pasquale Malacaria. A static analysis for quantifying information flow in a simple imperative language. *Journal of Computer Security*, 2007.

[CML11] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2011.

[CPP08a] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Prakash Panangaden. Anonymity protocols as noisy channels. *Information and Computation*, 206(2–4):378–401, 2008.

[CPP08b] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Prakash Panangaden. On the Bayes risk in information-hiding protocols. *Journal of Computer Security*, 16(5):531–571, 2008.

[CPS14] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. A predictive differentially-private mechanism for mobility traces. In E. De Cristofaro and S.J. Murdoch, editors, *Proceedings of the 14th International Symposium on Privacy Enhancing Technologies (PETS 2014)*, volume 8555 of *Lecture Notes in Computer Science*, pages 21–41. Springer, 2014.

[CPS15] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. Constructing elastic distinguishability metrics for location privacy. *CoRR*, abs/1503.00756, 2015.

[CZBP06] Reynold Cheng, Yu Zhang, Elisa Bertino, and Sunil Prabhakar. Preserving user location privacy in mobile data management infrastructures. In George Danezis and Philippe Golle, editors, *Proceedings of the 6th International Workshop on Privacy Enhancing Technologies (PET)*, volume 4258 of *Lecture Notes in Computer Science*, pages 393–412. Springer, 2006.

[Dew12] Rinku Dewri. Local differential perturbations: Location privacy under approximate knowledge attackers. *IEEE Transactions on Mobile Computing*, 99(PrePrints):1, 2012.

[DK05] Matt Duckham and Lars Kulik. A formal model of obfuscation and negotiation for location privacy. In *Proc. of the Third International Conference on Pervasive Computing (PERVASIVE)*, volume 3468 of *Lecture Notes in Computer Science*, pages 152–170. Springer, 2005.

[DMDBP08] Yoni De Mulder, George Danezis, Lejla Batina, and Bart Preneel. Identification via location-profiling in gsm networks. In *Proceedings of the 7th ACM Workshop on Privacy in the Electronic Society (WPES 2008)*, pages 23–32. ACM, 2008.

[dMHVB13] Y.A. de Montjoye, C.A. Hidalgo, M. Verleysen, and V.D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific Reports*, 2013.

[DNPR10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724. ACM, 2010.

[Dwo06] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *33rd International Colloquium on Automata, Languages and Programming (ICALP 2006)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

[Eco10] London Economics. Study on the economic benefits of privacy-enhancing technologies, 2010. European Commission DG Freedom, Security and Justice.

[ECP14] Ehab ElSalamouny, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Generalized differential privacy: Regions of priors that admit robust optimal mechanisms. In Franck van Breugel, Elham Kashefi, Catuscia Palamidessi, and Jan Rutten, editors, *Horizons of the Mind. A Tribute to Prakash Panangaden*, volume 8464 of *Lecture Notes in Computer Science*, pages 292–318. Springer International Publishing, 2014.

[ela] https://github.com/paracetamolo/elastic-mechanism.

[GG03] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of the First International Conference on Mobile Systems, Applications, and Services (MobiSys)*. USENIX, 2003.

[GKdPC10] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. GEPETO: A GEoprivacy-enhancing TOolkit. In *IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, pages 1071–1076. IEEE Computer Society, 2010.

[GKdPC11] 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, 2011.

[GKdPC13a] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. De-anonymization attack on geolocated data. In *12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2013)*, pages 789–797, 2013.

[GKdPC13b] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. De-anonymization attack on geolocated data. In *Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2013)*, pages 789–797. IEEE, 2013.

[GL08] Bugra Gedik and Ling Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Trans. Mob. Comput.*, 2008.

[GP09] Philippe Golle and Kurt Partridge. On the anonymity of home/work location pairs. In *Proceedings of the 7th International Conference on Pervasive Computing (PerCom)*. IEEE, 2009.

[HR10] Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70. IEEE Computer Society, 2010.

[HR11] Shen-Shyang Ho and Shuhua Ruan. Differential privacy for location pattern mining. In *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS (SPRINGL)*, pages 17–24. ACM, 2011.

[Kru09] John Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.

[KYS05] Hidetoshi Kido, Yutaka Yanagisawa, and Tetsuji Satoh. Protection of location privacy using dummies for location-based services. In *Proc. of ICDE Workshops*, page 1248, 2005.

[LLMS14] Chao Li, Daniel Yang Li, Gerome Miklau, and Dan Suciu. A theory of pricing private data. *ACM Trans. Database Syst.*, 39(4):34:1–34:28, 2014.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, volume 7, pages 106–115, 2007.

[loc] Location guard. https://github.com/chatziko/location-guard.

[Mal07] Pasquale Malacaria. Assessing security threats of looping constructs. In Martin Hofmann and Matthias Felleisen, editors, *Proceedings of the 34th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2007)*, pages 225–235. ACM, 2007.

[Mas94] Massey. Guessing and entropy. In *Proceedings of the IEEE International Symposium on Information Theory*, page 204. IEEE, 1994.

[MCA06] Mohamed F. Mokbel, Chi-Yin Chow, and Walid G. Aref. The new casper: Query processing for location services without compromising privacy. In Umeshwar Dayal, Kyu-Young Whang, David B. Lomet, Gustavo Alonso, Guy M. Lohman, Martin L. Kersten, Sang Kyun Cha, and Young-Kuk Kim, editors, *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB)*, pages 763–774. ACM, 2006.

[McS09] Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In Ugur Çetintemel, Stanley B. Zdonik, Donald Kossmann, and Nesime Tatbul, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 19–30. ACM, 2009.

[MKA+08] Ashwin Machanavajjhala, Daniel Kifer, John M. Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory meets practice on the map. In Gustavo Alonso, José A. Blakeley, and Arbee L. P. Chen, editors, *Proceedings of the 24th International Conference on Data Engineering, ICDE 2008, April 7-12, 2008, Cancún, México*, pages 277–286. IEEE, 2008.

[MKGV07] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[MT07] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007)*, pages 94–103. IEEE Computer Society, 2007.

[MYYR10] Chris Y. T. Ma, David K. Y. Yau, Nung Kwan Yip, and Nageswara S. V. Rao. Privacy vulnerability of published anonymous mobility traces. In *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking, (MOBICOM 2010)*, pages 185–196, 2010.

[OHSH14] Alexandra-Mihaela Olteanu, Kévin Huguenin, Reza Shokri, and Jean-Pierre Hubaux. Quantifying the effect of co-location information on location privacy. In E. De Cristofaro and S.J. Murdoch, editors, *Proceedings of the 14th International Symposium on Privacy Enhancing Technologies (PETS 2014)*, Lecture Notes in Computer Science, pages 184–203. Springer, 2014.

[PMLB14] Vincent Primault, Sonia Ben Mokhtar, Cédric Lauradoux, and Lionel Brunie. Differentially private location privacy in practice. In *Proceedings of the Third Workshop on Mobile Security Technologies (MoST 2014)*. IEEE, 2014.

[Pos13] The Washington Post. Nsa tracking cellphone locations worldwide, snowden documents show, 2013.

[RP10] Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 157–168, Baltimore, Maryland, USA, September 27-29 2010. ACM.

[RR10] Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proc. of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 765–774, 2010.

[Sam01] Pierangela Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng*, 13(6):1010–1027, 2001.

[SBBV13] Stefan Stieger, Christoph Burger, Manuel Bohn, and Martin Voracek. Who commits virtual identity suicide? differences in privacy concerns, internet addiction, and personality between facebook users and quitters. *Cyberpsy., Behavior, and Soc. Networking*, 16(9):629–634, 2013.

[SGI09] Pravin Shankar, Vinod Ganapathy, and Liviu Iftode. Privately querying location-based services with SybilQuery. In Sumi Helal, Hans Gellersen, and Sunny Consolvo, editors, *Proc. of the 11th International Conference on Ubiquitous Computing (UbiComp)*, pages 31–40. ACM, 2009.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 625–56, Jul, Oct 1948.

[Sho14] Reza Shokri. Optimal user-centric data obfuscation. Technical report, ETH Zurich, 2014. `http://arxiv.org/abs/1402.3426`.

[Sho15] Reza Shokri. Privacy games: Optimal user-centric data obfuscation. In *Proceedings of the 15th International Symposium on Privacy Enhancing Technologies (PETS 2015)*, 2015.

[SJCH12] Kang G. Shin, Xiaoen Ju, Zhigang Chen, and Xin Hu. Privacy protection for users of location-based services. *IEEE Wireless Commun*, 19(2):30–39, 2012.

[Smi09] Geoffrey Smith. On the foundations of quantitative information flow. In Luca de Alfaro, editor, *Proceedings of the 12th International Conference on Foundations of Software Science and Computation Structures (FOSSACS 2009)*, volume 5504 of *LNCS*, pages 288–302, York, UK, 2009. Springer.

[SOL+14] Jacopo Staiano, Nuria Oliver, Bruno Lepri, Rodrigo de Oliveira, Michele Caraviello, and Nicu Sebe. Money walks: a human-centric study on the economics of personal mobile data. In *The 2014 ACM Conference on Ubiquitous Computing (UbiComp 2014)*, pages 583–594, 2014.

[SS98] Pierangela Samarati and Latanya Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In ACM, editor, *PODS '98. Proceedings of the ACM SIGACT–SIGMOD–SIGART Symposium on Principles of Database Systems, June 1–3, 1998, Seattle, Washington*, pages 188–188, pub-ACM:adr, 1998. ACM Press.

[STBH11] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. Quantifying location privacy. In *IEEE Symposium on Security and Privacy*, pages 247–262. IEEE Computer Society, 2011.

[STD+10] Reza Shokri, Carmela Troncoso, Claudia Diaz, Julien Freudiger, and Jean-Pierre Hubaux. Unraveling an old cloak: k-anonymity for location privacy. In *Proceedings of the 9th annual ACM Workshop on Privacy in the Electronic Society (WPES 2010)*, pages 115–118 115–118 115–118, 2010.

[STD+11] Reza Shokri, George Theodorakopoulos, George Danezis, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Quantifying location privacy: The case of sporadic location exposure. In *Proceedings of the 11th International Privacy Enhancing Technologies Symposium (PETS 2011)*, volume 6794 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[STT+12] Reza Shokri, George Theodorakopoulos, Carmela Troncoso, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Protecting location privacy: optimal strategy against localization attacks. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS 2012)*, pages 617–627. ACM, 2012.

[Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[Ter11] Manolis Terrovitis. Privacy preservation in the dissemination of location data. *SIGKDD Explorations*, 13(1):6–18, 2011.

[THD15] Joseph Turow, Michael Hennessy, and Nora Draper. The tradeoff fallacy: How marketers are misrepresenting american consumers and opening them up to exploitation. 2015.

[TLM09] Kar Way Tan, Yimin Lin, and Kyriakos Mouratidis. Spatial cloaking revisited: Distinguishing information leakage from anonymity. In *Proceedins of the 11th International Symposium on Advances in Spatial and Temporal Databases (SSTD 2009)*. Springer, 2009.

[TST+14] George Theodorakopoulos, Reza Shokri, Carmela Troncoso, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Prolonging the hide-and-seek game: Optimal trajectory privacy for location-based services. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society (WPES 2014)*. ACM, 2014.

[XC09] Toby Xu and Ying Cai. Feeling-based location privacy protection for location-based services. In *Proceedings of the 2009 ACM Conference on Computer and Communications Security (CCS 2009)*. ACM, 2009.

[XKP09] Mingqiang Xue, Panos Kalnis, and Hung Pung. Location diversity: Enhanced privacy protection in location based services. In *Proc. of the 4th International Symposium on Location and Context Awareness (LoCA)*, volume 5561 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2009.

[YZZ+10] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: driving directions based on taxi trajectories. In *GIS*, pages 99–108, 2010.

[ZB11] 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 2011)*, pages 145–156. ACM, 2011.

[ZXM10] Yu Zheng, Xing Xie, and Wei-Ying Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.