

MEMOIRE

En vue de l'obtention de l'

Habilitation à diriger les recherches

Délivré par : *l'Institut National Polytechnique de Toulouse (INP Toulouse)*

Présenté et soutenu le *21 Juin 2019* **par :**

GILLES TREDAN

Capturing Binary Graphs

JURY

VALÉRIE ISSARNY	Directrice de Recherches, Inria Paris	Rapporteur
MATTHIEU LATAPY	Directeur de Recherches, Lip6	Rapporteur
PASCAL FELBER	Professeur, Université de Neuchâtel	Rapporteur
FRANÇOIS TAÏANI	Directeur de Recherches, Irisa	Examinateur
MOHAMED KAÂNICHE	Directeur de Recherches, LAAS	Directeur

Unité de Recherche :
LAAS - UPR 8001
Correspondant :
Mohamed Kaâniche

The economic anarchy of capitalist society as it exists today is, in my opinion, the real source of the evil. (...) We see before us a huge community of producers the members of which are unceasingly striving to deprive each other of the fruits of their collective labor.

Unlimited competition leads to a huge waste of labor, and to that crippling of the social consciousness of individuals (...).This crippling of individuals I consider the worst evil of capitalism.

—Albert Einstein, *Monthly Review*, 1949

Remerciements

Je tiens à remercier l'ensemble des personnes ayant contribué à l'élaboration de ce document et à la conduite des recherches présentées ici.

En premier, je remercie mes rapporteurs Valérie Issarny, Pascal Felber et Matthieu Latapy pour avoir accepté d'évaluer mes travaux et permis la reconnaissance de ces pages en tant que diplôme universitaire. Avec François Taiani, examinateur mobilisé et pertinent, ils ont formé un jury avec lequel j'ai appris et pris plaisir à échanger.

Je remercie chaleureusement Mohamed Kaâniche, qui m'a soutenu dans ces travaux à la fois scientifiquement en tant que directeur de mémoire, et économiquement en tant que directeur d'équipe. Sans lui je n'aurais probablement jamais vu le bout de cette fastidieuse quête de qualification administrative.

Bien sûr, tous les travaux présentés dans ce document sont le fruit de collaborations. Le parcours transformant une idée en papier et un papier en une ligne sur dblp est escarpé. Je suis heureux d'avoir partagé ces aventures avec mes co-auteurs, et les remercie pour les joies, les déceptions, les raisons et les passions que nous avons rencontrées sur le chemin. Mersi da Erwan ma ankouezet ar wech diwezhan.

Parce que je ne travaille pas qu'avec des co-auteurs, et qu'ils façonnent l'environnement dont je suis le produit, je tiens à remercier tous ceux avec qui je travaille mais ne publie pas. Je pense à l'équipe TSF, et plus généralement aux collègues du LAAS. Je pense aussi au CRCA et au LPT (avec qui j'espère quand même publier un jour), à Maria Barthélémy et René Sultra.

Merci enfin à ceux avec qui je vis mais ne travaille pas. Ma famille étendue, Anne-Isa, Bébéchat et Mifou. Ils méritent une gratitude que je ne saurais exprimer dans ce document.

D

Contents

I Introduction **1**

II Is network tomography a good idea ? **11**

1 Modeling the impact of traceroute stars **13**
1.1 Context: Complex Networks arising from networking . . . 13
1.1.1 Related Work . . . 13
1.1.2 Contributions . . . 14
1.2 Model . . . 15
1.3 Inferrable Topologies . . . 18
1.3.1 Basic Observations . . . 18
1.4 Properties and Full Exploration . . . 21
1.5 Conclusion . . . 22

2 Mapping virtual clouds **23**
2.1 Context: Virtualized Networks . . . 23
2.2 Model . . . 25
2.2.1 VNet Embedding . . . 25
2.2.2 Request Complexity . . . 26
2.2.3 Additional Complexities . . . 27
2.2.4 Terminology and Notation . . . 27
2.3 Adversarial Topology Discovery . . . 28
2.3.1 Trees . . . 28
2.3.2 Arbitrary Graphs . . . 29
2.4 Motif Framework . . . 30
2.5 Simulations . . . 31
2.6 Related Work . . . 33
2.7 Conclusion . . . 34

III Data-oriented approaches **35**

1 Souk and Loca: inferring social networks from position information **39**
1.1 Context . . . 39
1.1.1 Typical Mobility and Interaction Analyses . . . 40
1.2 Souk: the Experimental Platform . . . 40
1.2.1 Position capture platform . . . 41
1.2.2 Mobility model framework . . . 42
1.2.3 Analysis pipeline . . . 42
1.3 Experimental Deployments . . . 43
1.3.1 Experimental setup . . . 43
1.3.2 General characteristics . . . 44

1.3.3 Constructing Interaction Graphs . . . 44
1.4 Reality Against Models . . . 46
1.5 Conclusion . . . 47

2 Loca: Inferring Social Networks from Sparse Traces **49**
2.1 Context . . . 49
2.2 Model and Algorithms . . . 50
2.2.1 Attacker Model . . . 50
2.2.2 Transformation Algorithms . . . 51
2.3 Experimental Results . . . 53
2.3.1 Datasets . . . 53
2.3.2 Metrics and Methods . . . 54
2.3.3 Sensors Deployment Strategies . . . 55
2.3.4 Quality of the Virtual Mapping . . . 55
2.3.5 Results . . . 56
2.4 Related Works . . . 59
2.5 Mitigation strategies . . . 60
2.6 Conclusion . . . 60

IV A model of $\mathcal{G}$ for dynamic graphs **63**
1.1 Context . . . 64
1.2 Preliminaries . . . 66
1.3 Centrality Distance . . . 67
1.4 Methodology . . . 69
1.5 Experimental Case Studies . . . 69
1.5.1 Centrality Distances over Time . . . 71
1.5.2 Dynamics Signature . . . 72
1.6 Related Work . . . 73
1.7 Conclusion . . . 74

V Conclusion and Future Works **77**
1.1 Some shortcomings of this document . . . 78
1.2 Future works . . . 79
1.2.1 Theory . . . 79
1.2.2 Applications . . . 80

List of Figures

1 Euler's first Graph (1736): abstracting islands as vertices, and bridges as edges. 2
2 Ernst-Reuter-Platz pedestrian domain abstracted as a graph. Vertices represent pedestrians waiting spots, sidewalks connect those as black edges, while inbound (resp. outbound) traffic lane pedestrian crossing are represented by orange (resp. green) edges. 3
3 Link activation pattern due to traffic lights: while it is always possible to walk on sidewalks, it is only possible to cross inbound traffic lanes (green links) two thirds of the time, and outbound traffic (orange links) one third of the time. Those possibilities are regulated by traffic lights that cycle through 3 distinct states. 4
4 A recurring illustration of $\mathcal{G}$. 6
5 Illustration of Chapter II.1 approach. 7
6 Illustration of Chapter II.2 approach. Arrows symbolise the $\mapsto$ embedding relation. 7
7 Illustration of Part III approaches: models and reality are compared by sampling. 8
8 Illustration of Part IV approach. 9
9 An example of Internet map produced by Matt Britt and based on traceroute measurements. 12

10 Model parameters in currently available trace sets. 16
11 Two non-isomorphic inferred topologies 17
12 Summary of bounds on inferrable topologies 22

13 Embedding process illustration. 24
14 Algorithm Dict overview 30
15 Results of Dict when run on different Internet and power grid topologies 32
16 The AS-4755 network 33
17 One of the first social networks, representing here the affinities in a kindergarten. Triangular and circular nodes correspond to girls and boys, respectively. The criteria for a link is "Studying and playing in proximity, actually sitting beside" [93] 36
18 Illustrations extracted from Milgram's small world experiment (1967) [128] 37
19 Different souk deployments 38

20 Each participant carries two lightweight tags, clamped on clothes on her/his shoulders. . . 41
21 Overview of the Souk processing pipeline 42
22 Layout of the experimental areas 44
23 Number of users in each trace. 44
24 Movement patterns for the `souk` dataset. 45
25 Temporal movement characteristics 45
26 Broadcast time on datasets and synthetic traces. 47
27 Digging into social interactions 47

28 Possible attack surfaces represented by red arrows. 50
29 (a) Deployment of 15 sensors in SOUK cocktail-room according to the Density strategy. (b) Representation of the *Transition Matrix* as a weighted graph laid out using the Fruchterman and Reingold algorithm. 54
30 AUC results of the (a) SOUK and (b) MILANO datasets for different deployment strategies. 56

31 Impact of $\mathsf{K_{in}}$ on SOUK (a) and MILANO (b) datasets, considering a deployment of 15 sensors with a 1 m range. 57
32 Detailed ROC for the SOUK dataset using 15 sensors: True Positive Rate (TPR) as a function of False Positive Rate (FRP). 58
33 Impact of sensors range on inference accuracy for 15 sensors evenly distributed in space. 58
34 Probability to have exactly x out of 10 best friends correctly identified by LOCA using 15 sensors of 1 m range. 59
35 Local/Global scenario: G_1 is a line graph, G_2 and G_3 differ from G_1 by only one link. . . 65
36 Two evolutionary paths from a line graph G_L to a shell graph G_S. 66
37 The graph-of-graph $\mathcal{G}$ connects named networks (represented as stars). Two networks are neighboring iff they differ by a graph edit distance of one. The centrality distance defines a distance for each pair of neighboring graphs. 67
38 Number of edges and graph edit distance *GED* in the network traces. 70
39 Example centrality distances against null model 72
40 Histogram representation of the different facets of graph dynamics 73

Part I
Introduction

Summary of the thesis

This thesis regroups a set of algorithmical approaches related to the problem of graph metrology. Graphs are an abstraction representing a set of entities along with their interactions. As such, it is widely used as a tool to model real-life homogeneous systems. However, when using graphs to represent actual systems, a central question is "How to accurately represent these systems using graphs ?".

Indeed, when using Graphs as an abstraction to represent a real interaction system, accurately obtaining the set of interactions is sometimes difficult. This is often due to standard phenomena like dynamism (entities and interactions evolve over time), or difficulties in accurately measuring pairwise interactions among entities that will result in noisy or erroneous measurements and missing information. Consider for instance social networks: while those networks have largely been studied, and assuming "social interaction" would find a consensual definition among the community, measuring the presence or the absence of such interaction remains an error-prone process.

More precisely, fix a set of entities V: each set of possible symmetric binary interactions E among the elements of V defines a graph G. Let $\mathcal{G}$ the set of possible such graphs. Assume the system of interest is perfectly represented by an ideal graph G that has to be captured through an imperfect process. As a result a graph G' is obtained. This thesis presents different perspectives on the relation between G and G'.

While physics have developed a strong metrological framework to harness such uncertainty in continuous models, the discrete nature of graphs along with the huge size of the possible graph space is problematic. To illustrate this, we show in the first part of this manuscript that some theoretical analyses of graph capture approaches provide bad pessimistic answers. In particular, we show in a first study on Traceroute that missing information has a very bad impact on the accuracy of graph capture. We also show in the context of virtual networks that capture algorithms are by nature expensive.

Nevertheless, while capturing graphs is in theory inacurrate and expensive, it is often efficient in practice. There is a gap between the worst case on any possible graph, and the average case on real graphs. Such a gap between theoretical analyses and practical usefulness provides a hint. Somehow, the approach of characterising for a given function f the worst case difference $\max_{G,G' \in \mathcal{G}} |f(G) - f(G')|$ does not match the classical use context of those topologies that could be in a nutshell represented by $\mathbb{E}_{G,G' \in \mathcal{G}_{real}} |f(G) - f(G')|$. This formulation highlights two complementary approaches to fill this gap: *i*) focus on the relation between $\mathcal{G}$ and $\mathcal{G}_{real}$ in order to reduce the gap between $\max_{G,G' \in \mathcal{G}}$ and $\max_{G,G' \in \mathcal{G}_{real}}$; *ii*) characterise the distance between G and G' and seek for properties of f allowing to relate $d(G, G')$ and f (e.g. continuity transposed to the discrete set of graphs).

The second part of this manuscript focuses on modeling real graphs to provide an approach to this question in the context of human mobility. By extensively collecting data on interacting humans in a crowd, we collect instances of $G' \in \mathcal{G}_{real}$ that highlight the importance of the link creation mechanism: there is some correlation between G and G'. We also show that current mobility models fail to reproduce important properties of real graphs. However, we show that such interaction structures can still prove very usefull even when constructed from very sparse information.

The last part of this manuscript takes a step back on these analyses by providing some tools to chart the space of possible graphs. While theory tells that graphs are extremely noise-sensitive, practice shows that in general the noise does not matter. The angle pursued in this part is to provide useful distances on $\mathcal{G}$, that more precisely allow to differentiate errors that matter and errors that don't. We define a new family of distances on $\mathcal{G}$, and illustrate their use in the context of dynamic graphs.

The selection of works presented in this thesis has for objective to shed different lights to the problem of graph capture. However, the big picture still has many holes. The final part provides some intuition about possible directions to fill them.

From Königsberg to Berlin

The first system modeled as a graph was probably the islands of the city of Königsberg and its bridges. It was introduced in 1736 by Euler. Since then, this abstraction has found vast application arrays, ranging from sociology to computer science through physics, mathematics or epidemiology.

Although the original article has for subject what is now known as "Euler's 7 bridges problem", its deepest contribution perhaps does not lie in the solution to the problem, but rather in the introduction of a novel abstraction that will both support deep theory (e.g. graph minors well-quasi-ordering [112]) and experimental approaches ([128]). Such a versatility is probably due to the simplicity of the abstraction: it only requires the definition of what is an *entity* or a vertex and of what is an *interaction* or an edge between these entities.

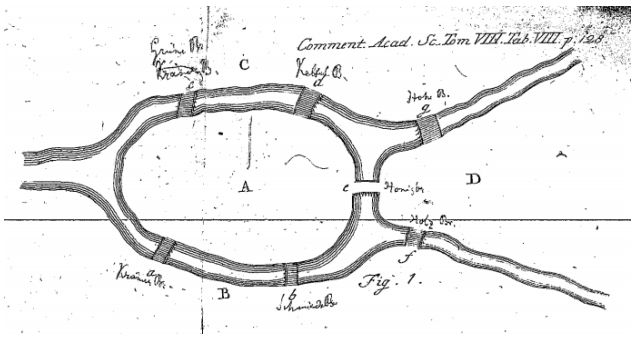


Figure 1: Euler's first Graph (1736): abstracting islands as vertices, and bridges as edges.

In this first graph abstraction ever defined, entities were islands defined by the river Pregel (that will later inspire the name of google's large graph mining system[88]). The interactions modeled between the islands were the bridges. The illustration of his original article is reproduced Figure 1: it contained 4 vertices, and 7 edges. In the remainder of this document, we will refer to such graphs as *binary graphs*: graphs composed of symmetric and unvalued edges that could be represented by symmetric binary adjacency matrices.

The Binary graph representation is a good tool to abstract a set of islands connected by bridges. Many reasons, often kept implicit, can justify this choice:

- it is impossible to change island without using bridge for a normal citizen having a walk in the city;
- once on a island it is possible to reach any bridge of that island;
- bridges do not change often; it is easy to decide whether there is a bridge between two given islands;
- bridges always connect two distinct islands and can be crossed both ways;
- no hidden or fake bridge exists.

In other words, binary graphs were the right tool to strip down the real city from all its complexity (road network, shops, individuals, districts) to only keep the structure of the island network. Let us however observe that such approach is only useful if the problem of interest relates to this island network.

Euler's original question is "is there a walk that would cross each bridge exactly once ?". This problem only involves bridges, and as such, a binary graph modeling the island and bridges network is sufficient to provide an answer to this question. That is, abstracting the city of Königsber by a binary graph removed lots of information but not what is required to answer the Euler's question. To state the obvious, such model would not necessarily be sufficient to answer other questions, for instance: "where is the best bar of the city ?".

In short, the binary graph abstraction fits very well this use-case. Since defining entities and interactions is a rather easy task, this graph model quickly grew in popularity. However, abstracting a real life problem by a binary graph is not always so simple or fruitful. I like to illustrate this point using an apparently very close problem that arised while I was working as a PostDoc in the FG-Inet group of TU-Berlin. The group was located in the Tel building, on the Ernst-Reuter-Platz roundabout. This roundabout is a major intersection of Berlin, connecting 5 major 4-lane streets. The roundabout has a diameter of roughly $150m$, and circumventing it as a pedestrian takes a long time.

The limits of binary graphs

If we want to model (pedestrian) walks along this roundabout, we can model it using Euler's approach: like the bridges connecting islands on the Pregel river, pedestrian crossings would be the edges connecting the different sidewalks. The Ernst-Reuter place and its abstraction as a graph are represented Figure 2. However, to cope with the intense vehicle traffic of the German capital, the flow of cars and pedestrians is orchestrated by a set of traffic lights. Pedestrians cannot cross the links at arbitrary times. More precisely, these lights follow a particular 3-stage pattern: during first stage, all lights are red for pedestrians and it is impossible to cross any road. During the second stage, traffic lights stop the cars *entering* the roundabout, allowing the pedestrian to cross the green links. During the last stage, traffic lights also stop the cars *leaving* the roundabout, allowing pedestrian to cross both green and orange links. This link activation pattern is represented in Figure 3.

Since the 4-lane streets are rather wide, crossing a street as a pedestrian takes a handful of seconds. It is therefore only possible to cross one link during each stage. The intriguing consequence of this timing is the following: as a pedestrian it is a lot easier to circle the roundabout anti-clockwise rather than clockwise. Consider for instance the path from i to g: regardless of the stage at which I reach i, I can reach g in 2 to 4 stages. However, on the other direction I need at most 5 stages (and at least 3) to reach i. Asymmetry, the consequence of this timing, interestingly arises even despite the very symmetric nature of link crossing: it takes exactly the same time to cross a link in both directions.

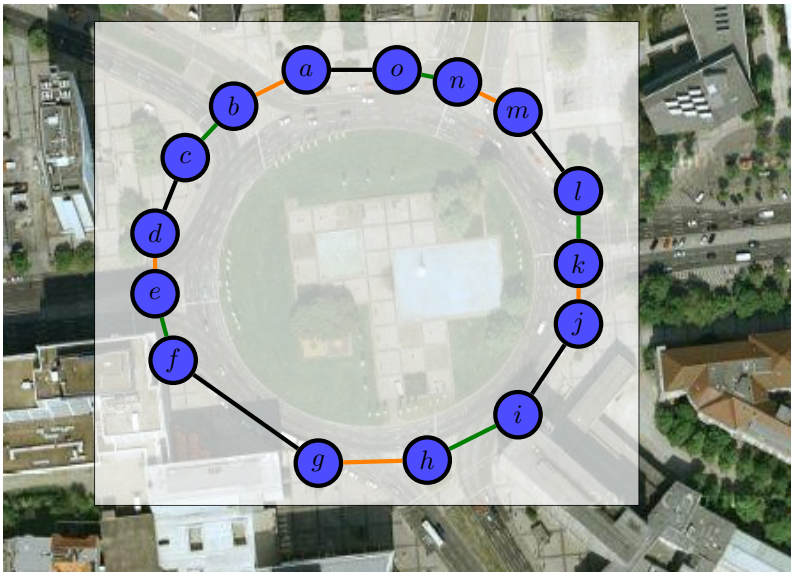


Figure 2: Ernst-Reuter-Platz pedestrian domain abstracted as a graph. Vertices represent pedestrians waiting spots, sidewalks connect those as black edges, while inbound (resp. outbound) traffic lane pedestrian crossing are represented by orange (resp. green) edges.

Can I use a static and binary graph model to think about the experience of a pedestrian on this roundabout ? The answer is in general no, as symmetric networks are by nature completely unable to capture the asymmetries of Ernst Reuter Platz. So despite the similarities of my problem compared to Königsberg 7 bridges, and despite that the edges (pedestrian crossings) are symmetric by nature, the activation patterns of the links combined with their topological arrangements have generated an asymmetric situation that I cannot capture with my symmetric graph model.

Of course, depending on the nature of the question, this abstraction will sometimes be useful. A binary representation will for example be enough to represent where should pedestrian crossings be painted. However, the topological shortest path from a to g will not represent the shortest *journey* [28].

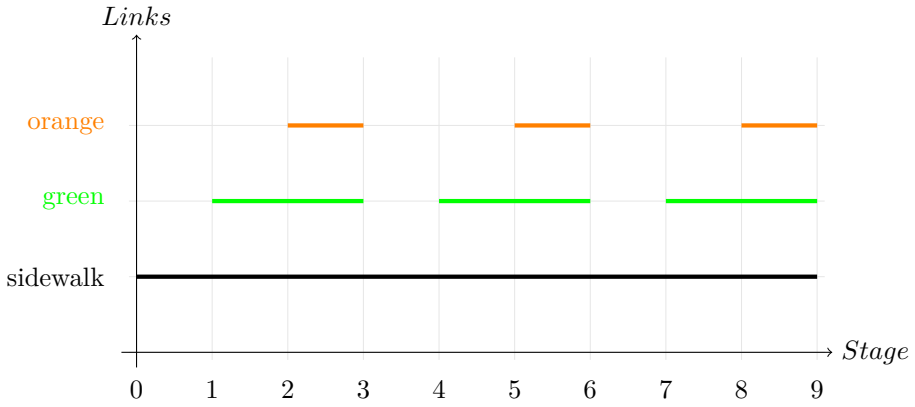


Figure 3: Link activation pattern due to traffic lights: while it is always possible to walk on sidewalks, it is only possible to cross inbound traffic lanes (green links) two thirds of the time, and outbound traffic (orange links) one third of the time. Those possibilities are regulated by traffic lights that cycle through 3 distinct states.

Sampling and modeling

Imagine now one wants to create a map of Berlin walking times: given the roundabout problem, we now understand it is a difficult problem. Since the walking times on the roundabout change depending on the direction travelled and the departure time, establishing an accurate map would require many measurements. Since Berlin is a big city, its architecture constantly changes for instance due to construction and road works, therefore such a map would need to be constantly updated to produce accurate answers. Another, more realistic approach, would be to measure the distance on the ground, that is unlikely to change a lot over time, and to account for delays at the crossings while simply ignoring the synchronized roundabouts like Ernst-Reuter-Platz. This map would be less accurate, but far easier to establish and maintain.

Does it matter ? This central question implies that inaccurate maps might be useful provided they are accurate *enough* for a purpose. An intuitive reasoning for the Ernst-Reuter-Platz case would probably lead to the conclusion that such configurations are relatively rare in a city, and that the impact of clockwise vs. anti-clockwise travel is not important enough to justify using a more complex model, such as asymmetric graphs.

To provide a theoretical answer to this problem would however require some work. For instance to capture that such configurations are relatively rare, one would probably require a model of urban pedestrian navigability graphs in general, and show that roundabouts rarely serve more than a fixed amount of streets, and that few other situations can create this problem, and so on.

The objective of these two introductory examples is to illustrate the necessity of a reasoning on the accuracy of our modeling operations and their impact. As we abstract the reality with models, we omit part of the reality. The consequences of these omissions are that the considered graph does not accurately describe the real situation. In short, these examples illustrate the necessity of a metrological reasoning about the graph objects as well.

This observation is not new. As we will see throughout this document, scientists worry since long ago about the potential biases introduced by their observation methodology of a given network. In 2005, [60] called for a metrological perspective on the problem of sampling complex networks. It is to our knowledge the earliest call for a more systematic approach on the problem introduced by representing real systems with networks. This thesis attemps to provide a perspective on the authors' works that concurs this objective.

Table I synthetises the different works presented in this thesis under the metrology perspective. Errors originate from two sources: the inaccuracies of the link observation process, and the unstability of the system over time. While these both sources have different nature, they both have for consequence a mismatch between the real system and its graph abstraction. One might observe that this approach is

Section	Entity	Edge/Interaction	Measurement method
Intro	Königsberg Islands	Bridges	City Map
II.1	WAN Routers	IP (Layer 3) communication	Traceroute Tomography
II.2	Hosts	Communication	Embedding requests
III.*	Individuals	Social contact	Position information Proximity
IV	Any fixed set of nodes	Any binary pairwise relation	Any method

../..

Section	Error Sources	Algorithmic Perspective	Quality Measure
Intro	$\emptyset$	$\emptyset$	$\emptyset$
II.1	Router Anonymity Time	Worst case analysis of graph properties Model Assumption	Property distribution range None
II.2	Only binary answers Time	Algorithms to greedily span a topology Model Assumption	Strict Equality None
III.*	Inference process, Elusive edge definition	Filtering, dry-runs Extrapolation Algorithm	Comparison against models "Arbitrary" ground truth
IV	Dynamism Link Sampling errors	Definition of distance functions	Comparison against a null model

Table 1: This table summarises the different graphs and metrology difficulties encountered throughout this document.

consistent with "traditional" metrology. Standards are for instance chosen to allow a *precise* measurement (e.g. the original definition of the second (s) depended on the length standard earth day has been abandonned because this earth day length is impacted by water tides) that is *stable* over time (e.g. the Kilogram reference is being abandoned because of its unstability over time despite all precautions). Interestingly, the concept of *failures*, particularly relevant when studying the resilience of man-made networks, can also be considered from this metrology perspective as a specific type of dynamism.

In summary, modeling a system by a graph is an operation that abstracts a real and complex set of intertwined elements into a pure structural representation of entities and interactions. Doing so is attractive if the seeked answers lie within the structure of the system, but in practice, finding the abstract graph that corresponds to a real system is often problematic: interactions often evolve in reality, and capturing those interactions is often not trivial. As a result, the observed/captured graph does not always accurately represent the real system. Understanding and mitigating the effect of these inaccuracies is the recurring angle of this thesis. The next chapter will focus on providing an overview of these approaches.

A visual perspective on this thesis

Throughout this thesis, a useful perspective will be the one of the set of all possible graphs over V named nodes, noted $\mathcal{G}^V$:

Definition:($\mathcal{G}$) Let V a finite set of elements. Let $\mathcal{G}^V$ the set of all possible symmetric binary graphs between these $n = |V|$ elements. Let $P(X)$ the power set of a set X. Then for each possible $E \in P(V \times V)$, there exists a unique graph $G \in \mathcal{G}^V$ such that $G = (V, E)$.

When the context is clear, we will write $\mathcal{G}$. One first observation is that we choose here to explicitely refer to "named" elements in V. This means that in particular that isometric graphs are considered different: $G_1 \in isom(G_2) \not\Rightarrow G_1 = G_2$. This distinction is important because it allows to test for equality at a small computational cost. However, this dooms us to only compare graphs among identical sets of nodes. For instance, the distances we will develop hereafter can only compute distances within a given set $\mathcal{G}^V$, and it is correct to write $\forall G_1, G_2 \in \mathcal{G}^V, d(G_1, G_2)$ but not $d(G_1, H_1)$ with $G_1 \in \mathcal{G}^V$ and $H_1 \in \mathcal{G}^{V'}$.

This approach however carries the natural meaning we usually have when graphs are applied to everyday objects ("Alice plays with Carl while Denise stays alone" does not mean the same as "Alice plays with Denise while Carl stays alone"). Moreover, in most of the presented works, V is easily known.

A second observation concerns the enormous size of this set. Its size is $2^{\binom{n}{2}}$. There are 1024 different possible graphs of 5 elements, and more than 68 billion possibilities with 9 elements. A most tragic consequence of this size is to forbid any enumerative approach. Yet it reamins a nice thought experiment, which I believe is very useful to apprehend (and spatialize) many of the problems presented in this thesis.

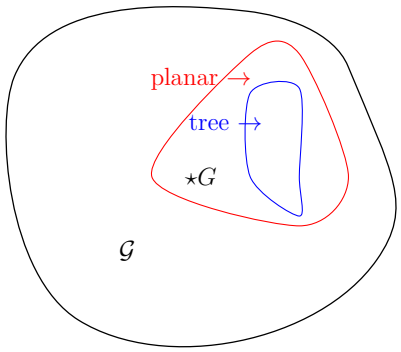


Figure 4: A recurring illustration of $\mathcal{G}$.

In the following I will represent $\mathcal{G}$ in two dimensions by a shapeless circuit set like in Figure 4. Each element in this shape is a binary symmetric graph, like G. Although the positions of the points representing the graphs within the shape are completely arbitrary, the representation of a set is useful to represent partitions and subsets of $\mathcal{G}$. In this context, Figure 4 would represent G as a planar graph that is not a tree.

In the first chapter, we study the properties of the set of graphs reconstructed from a given trace τ (using Algorithm 1). More precisely, since each trace τ defines a set of possible graphs on which τ can be captured, we try to estimate the possible range of some properties of G (abstractly noted $prop(G)$ on Figure 5). This approach is done by worst case, by exhibiting couples of graphs G_1, G_2 that present the most different values of $prop(G)$. We believe this information is more important than the mere number of members of $\mathcal{G}_\tau$: if the distribution of values is narrow (possibly even a point), then this means that traceroute tomography allows to precisely estimate the value of this property. An (obvious) example of such property would be the number of connected components (under already good covering hypotheses for τ, see Chapter 1 for more details).

In Chapter 2, we use another tomographic approach. In the context of Virtual Networks (VNETs), we can request to embed a graph G in a target host H. Informally, the maximal graph G (in terms of size and density) that we can embed in H is reached when $G = H$, and reaching this limit in the fewest

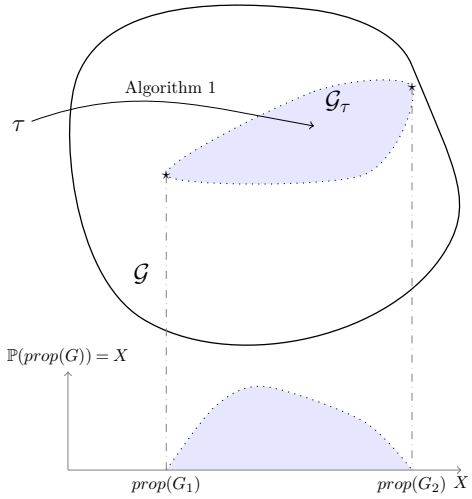


Figure 5: Illustration of Chapter II.1 approach.

possible attempts is the objective of our approach. To achieve this, we first observe that (under some mild conditions), the embedding relation (written $\mapsto$) defines a partial order relation on the set of graphs that we could embed in H: $(H, \mapsto)$ is a poset (partially ordered set). We then exploit this relation by defining greedy algorithms that, starting from the empty graph $\emptyset$, iteratively grow the structure G until H is reached, as illustrated in Figure 6. Note that only part of the poset structure is represented. In addition, we try to reduce the considerable cost of exploring this poset structure by reducing the problem to standard graph classes, such as Trees or Cacti where they become much more tractable.

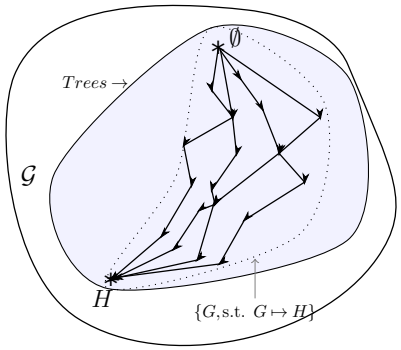


Figure 6: Illustration of Chapter II.2 approach. Arrows symbolise the $\mapsto$ embedding relation.

The conclusion of Chapter 2 is that reducing the search space to subsets of $\mathcal{G}$ (like the space of graphs constructed from few motifs $\mathcal{H}$) is an efficient approach. The popularity of traceroute-based measurement studies hint that the worst cases identified in Chapter 1 probably never happen in practice. Together, those conclusions illustrate the need for a more precise modeling of the graphs encountered "in practice". In other words, a better modeling of the reality is needed.

The notion of *graph model* is rather standard in the complex network community. Famous models such as Erdos-Renyi, Barabasi-Albert or Watts-Strogatz have driven the general understanding of processes on graphs. From $\mathcal{G}$'s perspective, a graph model is a probability distribution: let m be a model, then implicitely m defines a probability distribution $\mathbb{M} : \mathcal{G} \mapsto [0, 1]$ that associates with each graph its probability to be generated by model m.

While *graph classes* are subsets of $\mathcal{G}$, here the boundaries of a model are relatively soft: for instance, whatever the parameter $p > 0$ of an Erdos-Renyi (aka uniform) random graph model, all events of $\mathcal{G}$ have a non-negative probability of occurrence. That is, even K_n the densest graph has a small probability ($p^{n(n-1)/2} > 0$) to appear. For practical reasons however, we will represent models in the same way as classes, by delimiting a subregion of the original $\mathcal{G}$ shape.

In Part III we focus our approach on observing such "practical" graphs. More precisely, we focus our study on graphs that are generated from humans moving in the two dimensional plane. How to estimate whether our synthetic mobility models do a good job in reproducing likely observable graphs ? One possibility is to collect many observations, and generate many samples, and use the approach of Chapter 1: compare and average the natural and artificial samples along some standard graph properties. Although this approach has been presented in other publications (see e.g. [70, 99]) it is not the approach presented in this manuscript.

Indeed, doing so would neglect an important relation that binds all measured graphs: *time*. The captured and generated graphs are not static, they evolve as the individuals (or agents simulating individuals) move in the experience space. When I capture the evolving social network of a crowd, I obtain a set of graphs $\{G_i\}_{0<i<end}$. However, it is important to realise that the order matters (that is $G_1, G_2, \ldots, G_{end-1}$ in this specific order). For instance, the connectivity of vertices is bound by the physical location of the individuals they represent. Since those individuals cannot teleport, graphs evolve rather smoothly. Using standard graph properties has this limitation: it considers each single graph in isolation.

To illustrate this idea, consider a captured dynamic graph $DG = \{G_i\}_{0<i<end}$. One can compute its average diameter $E[diam(DG)] = \frac{1}{(end-1)} \sum_i diam(G_i)$. Imagine an arbitrary permutation of order of the graphs σ: let $DG' = \{G_{\sigma(i)}\}$. Of course reordering the snapshots will not change the average diameter $E[diam(DG)] = E[diam(DG')]$. From the diameter perspective, DG and DG' are strictly identical, and since DG is a real graph, this would push us to conclude that DG' is a realistic trace. Imagine however the real social event corresponding to DG'. It is probably not a very likely dynamic graph: people would leave and return to the event continuously, would not stay within a social circle for long, and so on.

This observation also holds for comparing models and real traces: while static graph measures already provide a hint on the quality of the model, the temporal dimension shall not be overlooked. This is why we chose to focus on an application primitive (being more interested in the capacity of synthetic models to be useful to the distributed application programmer), the broadcast. In the case of a broadcast time computed directly by simulation on dynamic graphs, it is easy to observe that re-ordering the snapshots will likely change the result.

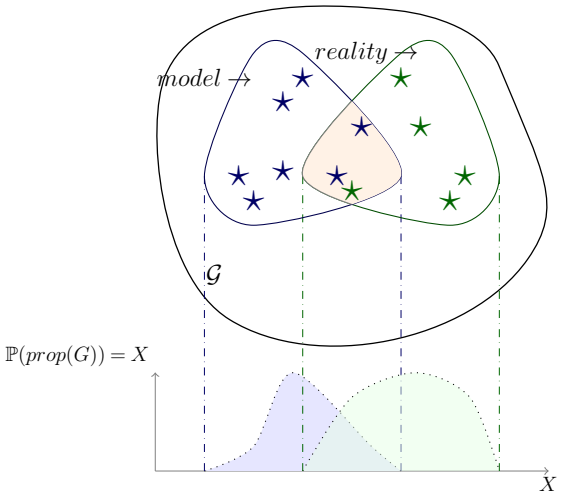


Figure 7: Illustration of Part III approaches: models and reality are compared by sampling.

Finally, Chapter IV seeks to provide a more general structuring framework to $\mathcal{G}$ in the context of dynamic graphs. Indeed, as illustrated through Figures 5- 7, a common feature of these approaches is to impose some structure to the "flat" set $\mathcal{G}$. The structures built are however highly application-dependent: different axioms would lead to different results in Chapter 1, a different embedding model would change the results of Chapter 2. A recurring question of these works is "how different/similar are these two graphs" ?

Chapter IV observes $\mathcal{G}$ as a graph, and shows it is possible to exploit this graph structure to construct distances on the space $\mathcal{G}$. The literature offers surprisingly few distance functions to compare two graphs. To alleviate this, we exploit centralities, a well known tool to capture the importance of nodes in graphs. We show that most standard centralities can be transformed into distances on $\mathcal{G}$. Such distance would capture the impact of a change in the topology on the roles of the nodes constituting this topology – where *role* is implicitely defined by the notion of *importance* each centrality exploits.

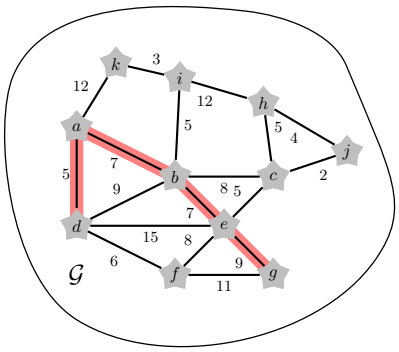


Figure 8: Illustration of Part IV approach.

This approach is illustrated Figure 8: each star $a-k$ is one topological configuration, one possible graph over the elements V. The edges of $\mathcal{G}$ are established by connecting each graph with the other graphs from which it only differs by one edge. The benefits of this approach is to allow representing dynamic graphs: they define trajectories in $\mathcal{G}$ provided the sampling resolution of the dynamic graph allows to observe each individual edge evolution (symbolized in red on the drawing). This approach also provides a very particular structure to the graph $\mathcal{G}$ (coined graph-of-graphs): it is bipartite (as opposed to what is represented) and $n(n-1)/2$ regular. By measuring the individual nodes centrality variations, we are able to decorate the constructed edges with weights. Each centrality defines a different weight distribution, and therefore a different perspective on the cost of the evolutionary path defined by the dynamic graph.

Interestingly, by exploiting the introduced distances, we also show in Chapter IV that "real" dynamic graphs (like those generated from human mobility) tend to minimise the variation (they define trajectories of low weight) compared to different randomized null models. In other words, the captured dynamic networks evolve less than expected (given the raw number of topological updates); roles of the nodes in those dynamic networks are more stable than expected. Sadly we do not know yet how to exploit these observations, for instance by leveraging the stability of information flows to improve the coordination capacities of a dynamic distributed system.

Part II
Is network tomography a good idea ?

In this part, I will present two rather theoretical approaches to network tomography. The central question that will structure these approaches is how to "capture" a topology ? The first study focuses on the Internet IP level routing, while the second focuses on Virtualized Network Infrastructures (VNETs). More precisely, the complete version of the presented results spans the following two papers :

- **Misleading Stars: What Cannot Be Measured in the Internet?** In *Journal Distributed Computing 26(4): 209-222 (2013).* with Yvonne-Anne Pignolet and Stefan Schmid
- **Adversarial Topology Discovery in Network Virtualization Environments: A Threat for ISPs?** In *Journal Distributed Computing (DIST), Springer, 2014.* with Yvonne-Anne Pignolet and Stefan Schmid

Both follow the same objective: study the performance of algorithms that produce a graph representing the target topology. The first chapter presents an offline algorithm (that works on a given input trace set), while the second chapter studies an "online" algorithm (generating requests depending on the answers of an online service). Hence these works have also different optimisation objectives: while the online algo optimises the number of issued requests, the first one studies the impact of missing information (namely, some node names) on the fidelity of reconstructed topologies.

In a nutshell, both theoretical studies bring bad news. In the context of VNETs, inferring dense topologies requires a large amount of requests. For instance, an optimal algorithm requires $o(n^2)$ requests to infer an n node graph in general (see more details in section 2.3.2). In the context of traceroute, missing node names can have a drastic impact on the accuracy of the inferred topologies. For instance, we show that even with the maximum of possible information available, the presence of s anonymous nodes can generate a quadratic uncertainty on the number of triangles (see more details in section 1.5). These results are however not very surprising, given the pessimistic nature of worst case approaches considered, and given the enormous size of $\mathcal{G}$.

Despite these theoretical barriers, the other interesting aspect of both studies is their relative pratical relevance. In section 2.5 we present simulations conducted on real internet topologies: due to their sparsity and relative regularity, they can be efficiently explored. Regarding the traceroute study, the large popularity of this approach to map the Internet such as the one on Figure 9 is also an argument of practical relevance: if people trust these maps despite their probable inaccuracy, it is probably because they find some use to it. In this case however, other factors such as the absence of more accurate and efficient alternatives to traceroute is probably also a main explanation parameter of traceroute popularity.

This common gap between the bad theoretical results and the good practical performance of the presented methods nicely introduces the need for models. The whole graph space $\mathcal{G}$ is too big for most explorations, and worst case approaches rather inaccurately reflect the practical performance of the methods. Models allow to circumvent both limitations: they reduce the graph space to a subset that more closely captures recurring aspects of the reality. Models also temper the worst case effect by again focusing the search on subsets of the graph space that are of practical relevance.

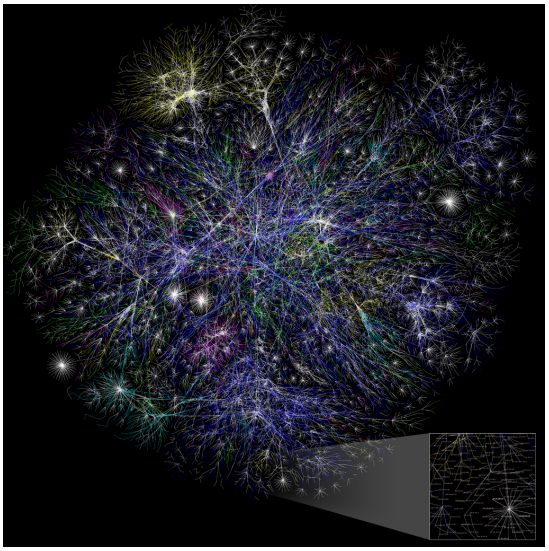


Figure 9: An example of Internet map produced by Matt Britt and based on traceroute measurements.

Chapter 1

Modeling the impact of traceroute stars

This chapter explores the relations between $\mathcal{T}$, the set of collected traces, and $\mathcal{G}_{\mathcal{T}}$, the set of topologies on which it is possible to observe $\mathcal{T}$. It is known that even if $\mathcal{T}$ is of high quality (this notion of quality will be described hereafter), the presence of some anonymous routers in the trace can trigger the exponential growth of the possible candidates sets $\mathcal{G}_{\mathcal{T}}$. This work builds on the observation that the mere number of inferrable topologies alone does not contradict the usefulness or feasibility of topology inference; if the set of inferrable topologies $\mathcal{G}_{\mathcal{T}}$ is homogeneous in the sense that the different topologies share many important properties, the generation of all possible graphs can be avoided: an arbitrary representative may characterize the underlying network accurately. Therefore, we identify important topological metrics such as diameter or maximal node degree and examine how "close" the possible inferred topologies are with respect to these metrics.

1.1 Context: Complex Networks arising from networking

The classic tool to study topological properties of the Internet (at the IP level) is *traceroute*. Traceroute allows us to collect traces from a given source node to a set of specified destination nodes. A trace between two nodes contains a sequence of identifiers describing a route between source and destination.

However, using traceroute to tomography IP-level topologies is error-prone. One of the major problems of traceroute based sampling is the presence of stars: not every node along a sampled path is configured to answer with its identifier. Rather, some nodes may be *anonymous* in the sense that they appear as stars ('$*$') in a trace. Anonymous nodes exacerbate the exploration of a topology because already a small number of anonymous nodes may increase the spectrum of inferrable topologies that correspond to a trace set $\mathcal{T}$.

1.1.1 Related Work

Arguably one of the most influential measurement studies on the Internet topology was conducted by the Faloutsos brothers [50] who show that the Internet exhibits a skewed structure: the nodes' out-degree follows a power-law distribution. Moreover, this property seems to be invariant over time. These results complement discoveries of similar distributions of communication traffic which is often self-similar, and of the topologies of natural networks such as human respiratory systems. The power-law property allows us to give good predictions not only on node degree distributions but also, e.g., on the expected number of nodes at a given hop-distance. Since [50] was published, many additional results have been obtained, e.g., [4, 5, 42, 80], also by conducting a distributed computing approach to increase the number of measurement points [27]. However, our understanding of the Internet's structure remains preliminary, and the topic continues to attract much attention from the scientific communities. In contrast to these

measurement studies, we pursue a more formal approach, and a complete review of the empirical results obtained over the last years is beyond the scope of this chapter.

The tool traceroute has been developed to investigate the routing behavior of the internet. [27, 57, 79, 102, 127] Paxson [102] used traceroute to analyze pathological conditions, routing stability, and routing symmetry. Another study by Gill et al. [57] demonstrates that large content providers (e.g., Google, Microsoft, Yahoo!) are deploying their own wide-area networks, bringing their networks closer to users, and bypassing Tier-1 ISPs on many routing paths.

Traceroute is also used to discover Internet topologies [32]. Unfortunately, there are several problems with this approach that render topology inference difficult, such as *aliasing* (a node appears with different identifiers on different interfaces) or *load-balancing*, which has motivated researchers to develop new tools such as *Paris Traceroute* [11, 68].

There are several results on traceroute sampling and its limitations. In their famous papers, Lakhina et al. [80] (by simulation) and Achlioptas et al. [4] (mathematically) have shown that the degree distribution under traceroute sampling exhibits a power law even if the underlying degree distribution is Poisson. Dall'Asta et al. [42] show that the edge and vertex detection probability depends on the betweenness centrality of each element and they propose improved mapping strategies. Another interesting result is due to Barford et al. [15] who experimentally show that when performing traceroute-like methods to infer network topologies, it is more useful to increase the number of destinations than the number of trace sources.

A drawback of using traceroute to determine the network topology stems from the fact that routers may appear as stars (i.e., anonymous nodes) in the trace since they are overloaded or since they are configured not to send out any ICMP responses. [134] The lack of complete information in the trace set renders the accurate characterization of Internet topologies difficult. This chapter attends to the problem of what information on the underlying topology can be inferred despite anonymous nodes and assumes a conservative, "worst-case" perspective that does not rely on any assumptions on the underlying network. Other work on this subject includes e.g., [134] by Yao et al. studying possible candidate topologies for a given trace set and computing the *minimal topology*, that is, the topology with the minimal number of anonymous nodes. Answering this question turns out to be NP-hard. Subsequently, different heuristics addressing this problem have been proposed [61, 68].

Our work is motivated by a series of papers by Acharya and Gouda. In [3], a network tracing theory model is introduced where nodes are "irregular" in the sense that each node appears in at least one trace with its real identifier. In [1], hardness results on finding the minimal topology are derived for this model. However, as pointed out by the authors themselves, the irregular node model—where nodes are anonymous due to high loads—is less relevant in practice and hence they consider strictly anonymous nodes in their follow-up studies [2]. As proved in [2], the problem is still hard (in the sense that there are many minimal networks corresponding to a trace set), even for networks with only two anonymous nodes, symmetric routing and without aliasing.

In contrast to the line of research on cardinalities of minimal networks, we are interested in the *network properties* of inferrable topologies. If the inferred topologies share the most important characteristics, the negative results on cardinalities in [1, 2] may be of little concern. Moreover, we believe that a study limited to minimal topologies only may miss important redundancy aspects of the Internet. Unlike [1, 2], our work is constructive in the sense that algorithms can be derived to compute inferred topologies.

Our work is also related to the field of end-to-end *network tomography*, where topologies are explored using pairwise measurements, without the cooperation of nodes along these paths. For a good discussion of this approach as well as results for a routing model along shortest and second shortest paths see [9]. For example, [9] shows that for sparse random graphs, a relatively small number of cooperating participants is sufficient to determine the network topology fairly well.

1.1.2 Contributions

This chapter reports on the study and characterization of topologies that can be inferred from a given trace set computed with the traceroute tool. While existing literature assuming a worst-case perspective has mainly focused on the cardinality of minimal topologies, we go one step further and examine specific topological graph properties.

We introduce a formal theory of topology inference by proposing basic axioms (i.e., assumptions on the trace set) that are used to guide the inference process. We present a novel definition for the isomorphism of inferred topologies which is aware of traffic paths; it is motivated by the observation that although two topologies look equivalent up to a renaming of anonymous nodes, the same trace set may result in different paths. Moreover, we propose the study of two extremes: in the first scenario, we only require that each link appears at least once in the trace set; interestingly, however, it turns out that this is often not sufficient, and we propose a "best case" scenario where the trace set is, in some sense, *complete*: it contains paths between all pairs of non-anonymous nodes.

The main result of this chapter is a negative one. It is shown that already a small number of anonymous nodes in the network renders topology inference difficult. In particular, we prove that in general, the possible inferrable topologies differ in many crucial aspects, e.g., the maximal node degree, the diameter, the stretch, the number of triangles and the number of connected components.

We introduce the concept of the *star graph* of a trace set that is useful for the characterization of inferred topologies. In particular, colorings of the star graphs allow us to constructively derive inferred topologies. (Although the general problem of computing the set of inferrable topologies is related to NP-hard problems such as *minimal graph coloring* and *graph isomorphism*, some important instances of inferrable topologies can be computed efficiently.) The chromatic number (i.e., the number of colors in the minimal proper coloring) of the star graph defines a lower bound on the number of anonymous nodes from which the stars in the traces could originate from. And the number of possible colorings of the star graph—a function of the *chromatic polynomial* of the star graph—gives an upper bound on the number of inferrable topologies. We show that this bound is tight in the sense that trace sets with that many inferrable topologies indeed exist. In particular, there are problem instances where the cardinality of the set of inferrable topologies equals the *Bell number*. This insight complements existing cardinality results and generalizes topology inference to arbitrary, not only minimal, inferrable topologies.

Finally, we examine the scenario of *fully explored networks* for which "complete" trace sets are available. As expected, the inferrable topologies are more homogeneous in this case and can be characterized well with respect to many properties such as the distances between nodes. However, we also find that some other properties are inherently difficult to estimate. Interestingly, our results indicate that full exploration is often useful to derive bounds on global properties (such as connectivity) while it does not help much for bounds on more local properties (such as node degrees).

1.2 Model

Let $\mathcal{T}$ denote the set of traces obtained from probing (e.g., by traceroute) a network $G_0 = (V_0, E_0)$ with *nodes* or *vertices* V_0 (the set of routers) and undirected *links* or *edges* E_0. We assume that G_0 is static during the probing time (or that probing is instantaneous), but we do not require that G_0 is connected. Each trace $T(u,v) \in \mathcal{T}$ describes a path connecting two nodes $u, v \in V_0$; when u and v do not matter or are clear from the context, we simply write T. Moreover, let $d_T(u,v)$ denote the distance (number of hops) between two nodes u and v in trace T. We define $d_{G_0}(u,v)$ to be the corresponding shortest path distance in G_0. Note that a trace between two nodes u and v may not describe the shortest path between u and v in G_0.

The nodes in V_0 fall into two categories: *anonymous* nodes and *non-anonymous* (or shorter: *named*) nodes. As it is the case in most related literature, we assume these categories to be permanent over time and traces: a node is either consistently anonymous or consistently non-anonymous. This is motivated by the fact that while sometimes nodes can be anonymous due to temporary events such as high loads, most of the time the cause are static configurations, see [134]: Some routers are configured to not send out ICMP responses while others use the destination addresses of traceroute packets instead of their own addresses as source addresses for outgoing ICMPv6 packets.

Therefore, each trace $T \in \mathcal{T}$ describes a sequence of symbols representing anonymous and non-anonymous nodes. We make the natural assumption that the first and the last node in each trace T are non-anonymous. Moreover, we assume that traces are given in a form where non-anonymous nodes appear with a unique, anti-aliased identifier (i.e., the multiple IP addresses corresponding to different interfaces of a node are resolved to one identifier); an anonymous node is represented as $*$ ("star")

data	# traces	n	s	# edges	# named edges	# src-dst pairs	avg trace length
routes1 [21]	6219	746	513	2372	1576	893	14.81
routes2 [21]	27510	1077	4571	10011	2243	1417	15.57
xprobes [21]	6407	909	631	2688	1512	905	15.69
xroutes.1 [21]	615	673	62	1026	906	273	15.83
PAM [14]	372	1511	279	2685	2153	372	12.25

Figure 10: This table gives an overview of the order of magnitudes of our model parameters in real life data. The number of edges refers to the number of edges in the trace set (the underlying graphs are unknown). The number of named edges counts the number of edges between named nodes. Some trace sets contain multiple queries for the same source-destination pairs, whereas others consist of one trace per pair.

in the traces. For our formal analysis, we assign to each star in a trace set $\mathcal{T}$ a unique identifier i: $*_i$. (Note that except for the numbering of the stars, we allow identical copies of T in $\mathcal{T}$, and we do not make any assumptions on the implications of identical traces: they may or may not describe the same paths.) Thus, a trace $T \in \mathcal{T}$ is a sequence of symbols taken from an alphabet $\Sigma = \mathcal{ID} \cup (\bigcup_i \{*_i\})$, where $\mathcal{ID}$ is the set of non-anonymous node identifiers (IDs): Σ is the union of the (anti-aliased) non-anonymous nodes and the set of all stars (with their unique identifiers) appearing in a trace set. The main challenge in topology inference is to determine which stars in the traces may originate from which anonymous nodes.

Henceforth, let $n = |\mathcal{ID}|$ denote the number of non-anonymous nodes and let $s = |\bigcup_i *_i|$ be the number of stars in $\mathcal{T}$; similarly, let a denote the number of anonymous nodes in a topology. Let $N = n + s = |\Sigma|$ be the total number of symbols occurring in $\mathcal{T}$.

Table 10 gives some statistics for our variables from other studies [102, 57]: the number of traces $|\mathcal{T}|$ is between 372 and 27510, the number of named nodes n is between 615 and 1077, the number of stars s is between 62 and 4571, and the number of edges in the trace set varies between 1026 and 10011. (See Section 1.1.1 for a short summary on the studies collecting these trace sets.)

Clearly, the process of topology inference depends on the assumptions on the measurements. In the following, we postulate the fundamental axioms that guide the reconstruction. First, we make the assumption that each link of G_0 is visited by the measurement process, i.e., it appears as a transition in the trace set $\mathcal{T}$. In other words, we are only interested in inferring the (sub-)graph for which measurement data is available.

Axiom 0 (*Complete Cover*): Each edge of G_0 appears at least once in some trace in $\mathcal{T}$.

The next fundamental axiom assumes that traces always represent paths on G_0.

Axiom 1 (*Reality Sampling*): For every trace $T \in \mathcal{T}$, if the distance between two symbols $\sigma_1, \sigma_2 \in T$ is $d_T(\sigma_1, \sigma_2) = k$, then there exists a path (i.e., a walk without cycles) of length k connecting two (named or anonymous) nodes σ_1 and σ_2 in G_0.

The following axiom captures the consistency of the routing protocol on which the traceroute probing relies. In the current Internet, policy routing is known to have an impact both on the route length [127] and on the convergence time [79].

Axiom 2 (α-*(Routing) Consistency*): There exists an $\alpha \in (0,1]$ such that, for every trace $T \in \mathcal{T}$, if $d_T(\sigma_1, \sigma_2) = k$ for two entries σ_1, σ_2 in trace T, then the shortest path connecting the two (named or anonymous) nodes corresponding to σ_1 and σ_2 in G_0 has distance at least $\lceil \alpha k \rceil$.

Note that if $\alpha = 1$, the routing is a shortest path routing. Moreover, note that if $\alpha = 0$, there can be loops in the paths, and there are hardly any topological constraints, rendering almost any topology inferrable. (For example, the complete graph with one anonymous router is always a solution.)

Any topology G which is consistent with these axioms (when applied to $\mathcal{T}$) is called *inferrable* from $\mathcal{T}$.

Definition 1.2.1 (Inferrable Topologies)**.** A topology G is (α-consistently) *inferrable* from a trace set $\mathcal{T}$ if axioms Axiom 0, Axiom 1, and Axiom 2 (with parameter α) are fulfilled.

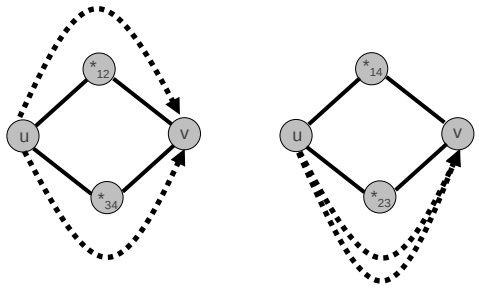


Figure 11: Two non-isomorphic inferred topologies, i.e., different mapping functions lead to these topologies.

We will refer by $\mathcal{G}_\mathcal{T}$ to the set of topologies inferrable from $\mathcal{T}$. Please note the following important observation.

Remark 1.2.2. In the absence of anonymous nodes, it holds that $G_0 \in \mathcal{G}_\mathcal{T}$, since $\mathcal{T}$ was generated from G_0 and AXIOM 0, AXIOM 1, and AXIOM 2 are fulfilled by definition. However, there are instances where an α-consistent trace set for G_0 contradicts AXIOM 0: as trace needs to start and end with a named node, some edges cannot appear in an α-consistent trace set $\mathcal{T}$. In the following, we will only consider settings where $G_0 \in \mathcal{G}_\mathcal{T}$.

The main objective of a topology inference algorithm ALG is to compute topologies which are consistent with these axioms. Concretely, ALG's input is the trace set $\mathcal{T}$ together with the parameter α specifying the assumed routing consistency. Essentially, the goal of any topology inference algorithm ALG is to compute a mapping of the symbols Σ (appearing in $\mathcal{T}$) to nodes in an inferred topology G; or, in case the input parameters α and $\mathcal{T}$ are contradictory, reject the input. This mapping of symbols to nodes implicitly describes the edge set of G as well: the edge set is unique as all the transitions of the traces in $\mathcal{T}$ are now unambiguously tied to two nodes.

So far, we have ignored an important and non-trivial question: When are two topologies $G_1, G_2 \in \mathcal{G}_\mathcal{T}$ different ? In this chapter, we pursue the following approach: We are not interested in purely topological isomorphisms, but we care about the identifiers of the non-anonymous nodes, i.e., we are interested in the locations of the non-anonymous nodes and their distance to other nodes. For anonymous nodes, the situation is slightly more complicated: one might think that as the nodes are anonymous, their "names" do not matter. Consider however the example in Figure 11: the two inferrable topologies have two anonymous nodes, one where $\{*_1, *_2\}$ plus $\{*_3, *_4\}$ are merged into one node each in the inferrable topology and one where $\{*_1, *_4\}$ plus $\{*_2, *_3\}$ are merged into one node each in the inferrable topology. In this chapter, we regard the two topologies as different, for the following reason: Assume that there are two paths in the network, one $u \rightsquigarrow *_2 \rightsquigarrow v$ (e.g., during day time) and one $u \rightsquigarrow *_3 \rightsquigarrow v$ (e.g., at night); clearly, this traffic has different consequences and hence we want to be able to distinguish between the two topologies described above. In other words, our notion of isomorphism of inferred topologies is *path-aware*.

It is convenient to introduce the following MAP function. Essentially, an inference algorithm computes such a mapping.

Definition 1.2.3 (Mapping Function MAP)**.** Let $G = (V, E) \in \mathcal{G}_\mathcal{T}$ be a topology inferrable from $\mathcal{T}$. A topology inference algorithm describes a surjective mapping function $\text{MAP} : \Sigma \to V$. For the set of non-anonymous nodes in Σ, the mapping function is bijective; and each star is mapped to exactly one node in V, but multiple stars may be assigned to the same node. Note that for any $\sigma \in \Sigma$, $\text{MAP}(\sigma)$ uniquely identifies a node $v \in V$. More specifically, we assume that MAP assigns labels to the nodes in V: in case of a named node, the label is simply the node's identifier; in case of anonymous nodes, the label is $*_\beta$, where β is the concatenation of the *sorted* indices of the stars which are merged into node $*_\beta$.

With this definition, two topologies $G_1, G_2 \in \mathcal{G}_\mathcal{T}$ differ if and only if they do not describe the identical (MAP-) labeled topology. We will use this MAP function also for G_0, i.e., we will write $\text{MAP}(\sigma)$ to refer to a symbol σ's corresponding node in G_0.

AXIOM 1 implies a natural way to merge traces to derive additional bounds on path lengths.

Lemma 1.2.4. For two traces $T_1, T_2 \in \mathcal{T}$ for which $\exists \sigma_1, \sigma_2, \sigma_3$, where σ_2 refers to a named node, such that $d_{T_1}(\sigma_1, \sigma_2) = i$ and $d_{T_2}(\sigma_2, \sigma_3) = j$, it holds that the distance between two nodes u and v corresponding to σ_1 and σ_3, respectively, in G_0, is at most $d_{G_0}(\sigma_1, \sigma_3) \leq i + j$.

Proof. Let $\mathcal{T}$ be a trace set, and $G \in \mathcal{G}_{\mathcal{T}}$. Let $\sigma_1, \sigma_2, \sigma_3$ s.t. $\exists T_1, T_2 \in \mathcal{T}$ with $\sigma_1 \in T_1, \sigma_3 \in T_2$ and $\sigma_2 \in T_1 \cap T_2$. Let $i = d_{T_1}(\sigma_1, \sigma_2)$ and $j = d_{T_2}(\sigma_1, \sigma_3)$. Since any inferrable topology G fulfills AXIOM 1, there is a path π_1 of length at most i between the nodes corresponding to σ_1 and σ_2 in G and a path π_2 of length at most j between the nodes corresponding to σ_2 and σ_3 in G. The combined path can only be shorter, and hence the claim follows. □ □

In this work we will often assume that AXIOM 0 is given. Thus, to prove that a topology is inferrable from a trace set it is sufficient to show that AXIOM 1 and AXIOM 2 are satisfied.

1.3 Inferrable Topologies

What insights can be obtained from topology inference with minimal assumptions, i.e., with our axioms? Or what is the structure of the inferrable topology set $\mathcal{G}_{\mathcal{T}}$? We first make some general observations and then examine different graph metrics in more detail.

1.3.1 Basic Observations

Although the generation of the entire topology set $\mathcal{G}_{\mathcal{T}}$ may be computationally hard, some instances of $\mathcal{G}_{\mathcal{T}}$ can be computed efficiently. The simplest possible inferrable topology is the so-called *canonic graph* G_C: the topology which assumes that all stars in the traces refer to different anonymous nodes. In other words, if a trace set $\mathcal{T}$ contains $n = |\mathcal{ID}|$ named nodes and s stars, G_C will contain $|V(G_C)| = N = n + s$ nodes.

Definition 1.3.1 (Canonic Graph G_C)**.** The *canonic graph* is defined by $G_C(V_C, E_C)$ where $V_C = \Sigma$ is the set of (anti-aliased) nodes appearing in $\mathcal{T}$ (where each star is considered a unique anonymous node) and where $\{\sigma_1, \sigma_2\} \in E_C \Leftrightarrow \exists T \in \mathcal{T}, T = (\ldots, \sigma_1, \sigma_2, \ldots)$, i.e., σ_1 follows after σ_2 in some trace T ($\sigma_1, \sigma_2 \in T$ can be either non-anonymous nodes or stars). Let $d_C(\sigma_1, \sigma_2)$ denote the *canonic distance* between two nodes, i.e., the length of a shortest path in G_C between the nodes σ_1 and σ_2.

Note that G_C is indeed one inferrable topology. In this case, $\textsc{Map} : \Sigma \to \Sigma$ is the identity function.

Theorem 1.3.2. G_C is inferrable from $\mathcal{T}$.

Proof. Fix $\mathcal{T}$. We have to prove that G_C fulfills AXIOM 0, AXIOM 1 and AXIOM 2.

AXIOM 0: The axiom holds trivially: only edges from the traces are used in G_C.

AXIOM 1: Let $T \in \mathcal{T}$ and $\sigma_1, \sigma_2 \in T$. Let $k = d_T(\sigma_1, \sigma_2)$. We show that G_C fulfills AXIOM 1, namely, there exists a path of length k in G_C. Induction on k: ($k = 1$:) By the definition of G_C, $\{\sigma_1, \sigma_2\} \in E_C$ thus there exists a path of length one between σ_1 and σ_2. ($k > 1$:) Suppose AXIOM 1 holds up to $k-1$. Let $\sigma'_1, \ldots, \sigma'_{k-1}$ be the intermediary nodes between σ_1 and σ_2 in T: $T = (\ldots, \sigma_1, \sigma'_1, \ldots, \sigma'_{k-1}, \sigma_2, \ldots)$. By the induction hypothesis, in G_C there is a path of length $k-1$ between σ_1 and σ'_{k-1}. Let π be this path. By definition of G_C, $\{\sigma'_{k-1}, \sigma_2\} \in E_C$. Thus appending $(\sigma'_{k-1}, \sigma_2)$ to π yields the desired path of length k linking σ_1 and σ_2: AXIOM 1 thus holds up to k.

AXIOM 2: We have to show that $d_T(\sigma_1, \sigma_2) = k \Rightarrow d_C(\sigma_1, \sigma_2) \geq \lceil \alpha \cdot k \rceil$. By contradiction, suppose that G_C does not fulfill AXIOM 2 with respect to α. So there exists $k' < \lceil \alpha \cdot k \rceil$ and $\sigma_1, \sigma_2 \in V_C$ such that $d_C(\sigma_1, \sigma_2) = k'$. Let π be a shortest path between σ_1 and σ_2 in G_C. Let $(T_1, \ldots, T_\ell)$ be the corresponding (maybe repeating) traces covering this path π in G_C. Let $T_i \in (T_1, \ldots, T_\ell)$, and let s_i and e_i be the corresponding start and end nodes of π in T_i. We will show that this path π implies the existence of a path in G_0 which violates α-consistency. Since G_0 is inferrable, G_0 fulfills AXIOM 2, thus we have: $d_C(\sigma_1, \sigma_2) = \sum_{i=1}^{\ell} d_{T_i}(s_i, e_i) = k' < \lceil \alpha \cdot k \rceil \leq d_{G_0}(\sigma_1, \sigma_2)$ since G_0 is α-consistent. However, G_0 also fulfills AXIOM 1, thus $d_{T_i}(s_i, e_i) \geq d_{G_0}(s_i, e_i)$. Thus $\sum_{i=1}^{\ell} d_{G_0}(s_i, e_i) \leq \sum_{i=1}^{\ell} d_{T_i}(s_i, e_i) < d_{G_0}(\sigma_1, \sigma_2)$: we

have constructed a path from σ_1 to σ_2 in G_0 whose length is shorter than the distance between σ_1 and σ_2 in G_0, leading to the desired contradiction. □ □

Theorem 1 implies that with our axioms the canonic graph is one of the possible topologies that could lead to a given trace set. However, it does not imply that the stars of traces from a given topology represent different nodes.

G_C can be computed efficiently from $\mathcal{T}$: represent each non-anonymous node and star as a separate node, and for any pair of consecutive entries (i.e., nodes) in a trace, add the corresponding link. The time complexity of this construction is linear in the size of $\mathcal{T}$. Also note that there is no inferrable graph in $\mathcal{G}_\mathcal{T}$ having a larger diameter than G_C.

With the definition of the canonic graph, we can derive the following lemma which establishes a necessary condition when two stars cannot represent the same node in G_0 from constraints on the routing paths. This is useful for the characterization of inferred topologies.

Lemma 1.3.3. Let $*_1, *_2$ be two stars occurring in some traces in $\mathcal{T}$. $*_1, *_2$ cannot be mapped to the same node, i.e., $\textsc{Map}(*_1) \neq \textsc{Map}(*_2)$, without violating the axioms in the following conflict situations:

(i) if $*_1 \in T_1$ and $*_2 \in T_2$, and T_1 describes a path that is too long between anonymous node $\textsc{Map}(*_1)$ and non-anonymous node u, i.e., $\lceil \alpha \cdot d_{T_1}(*_1, u) \rceil > d_C(u, *_2)$.

(ii) if $*_1 \in T_1$ and $*_2 \in T_2$, and there exists a trace T that contains a path between two non-anonymous nodes u and v and $\lceil \alpha \cdot d_T(u,v) \rceil > d_C(u,*_1) + d_C(v,*_2)$.

Lemma 1.3.3 can be applied to show that a topology is not inferrable from a given trace set because it merges (i.e., maps to the same node) two stars in a manner that violates the axioms. Let us introduce a useful concept for our analysis: the *star graph* that describes the conflicts between stars.

Definition 1.3.4 (Star Graph G_*)**.** The *star graph* $G_*(V_*, E_*)$ consists of vertices V_* representing stars in traces, i.e., $V_* = \bigcup_i \{*_i\}$. Two vertices are connected if and only if they must differ according to Lemma 1.3.3, i.e., $\{*_1, *_2\} \in E_*$ if and only if at least one of the conditions of Lemma 1.3.3 hold for $*_1, *_2$.

Note that the star graph G_* is unique and can be computed efficiently for a given trace set $\mathcal{T}$: Conditions (i) and (ii) can be checked by computing G_C. However, note that while G_* specifies some stars which cannot be merged, the construction is not sufficient: as Lemma 1.3.3 is based on G_C, additional links might be needed to characterize the set of inferrable and α-consistent topologies $\mathcal{G}_\mathcal{T}$ exactly. In other words, a topology G obtained by merging stars that are adjacent in G_* is never inferrable ($G \notin \mathcal{G}_\mathcal{T}$); however, merging non-adjacent stars does not guarantee that the resulting topology is inferrable.

What do star graphs look like? The answer is *arbitrarily*: the following lemma states that the set of possible star graphs is equivalent to the class of general graphs. This claim holds for any α.

Lemma 1.3.5. For any graph $G = (V, E)$, there exists a trace set $\mathcal{T}$ such that G is the star graph for $\mathcal{T}$.

The problem of computing inferrable topologies is related to the vertex colorings of the star graphs. We will use the following definition which relates a vertex coloring of G_* to an inferrable topology G by contracting independent stars in G_* to become one anonymous node in G. For example, observe that a maximum coloring treating every star in the trace as a separate anonymous node describes the inferrable topology G_C.

Definition 1.3.6 (Coloring-Induced Graph)**.** Let γ denote a coloring of G_* which assigns colors $1, \ldots, k$ to the vertices of G_*: $\gamma : V_* \to \{1, \ldots, k\}$. We require that γ is a proper coloring of G_*, i.e., that different anonymous nodes are assigned different colors: $\{u, v\} \in E_* \Rightarrow \gamma(u) \neq \gamma(v)$. G_γ is defined as the topology *induced* by γ. G_γ describes the graph G_C where nodes of the same color are contracted: two vertices u and v represent the same node in G_γ, i.e., $\textsc{Map}(*_i) = \textsc{Map}(*_j)$, if and only if $\gamma(*_i) = \gamma(*_j)$.

The following two lemmas establish an intriguing relationship between colorings of G_* and inferrable topologies. Also note that Definition 1.3.6 implies that two different colorings of G_* define two non-isomorphic inferrable topologies.

We first show that while a coloring-induced topology always fulfills AXIOM 1, the routing consistency is sacrificed.

Lemma 1.3.7. Let γ be a proper coloring of G_*. The coloring-induced topology G_γ is a topology fulfilling AXIOM 2 with a routing consistency of $\alpha' > 0$, for an arbitrarily small α'.

Proof available in the original paper [104].

An inferrable topology always defines a proper coloring on G_*.

Lemma 1.3.8. Let $\mathcal{T}$ be a trace set and G_* its corresponding star graph. If a topology G is inferrable from $\mathcal{T}$, then G induces a proper coloring on G_*.

Proof. For any α-consistent inferrable topology G there exists some mapping function MAP that assigns each symbol of $\mathcal{T}$ to a corresponding node in G (cf Definition 1.2.3), and this mapping function gives a coloring on G_* (i.e., merged stars appear as nodes of the same color in G_*). The coloring must be proper: due to Lemma 1.3.3, an inferrable topology can never merge adjacent nodes of G_*. □ □

The colorings of G_* allow us to derive an upper bound on the cardinality of $\mathcal{G}_\mathcal{T}$.

Theorem 1.3.9. Given a trace set $\mathcal{T}$ sampled from a network G_0 and $\mathcal{G}_\mathcal{T}$, the set of topologies inferrable from $\mathcal{T}$, it holds that:

$$\sum_{k=\gamma(G_*)}^{|V_*|} P(G_*,k)/k! \geq |\mathcal{G}_\mathcal{T}|,$$

where $\gamma(G_*)$ is the chromatic number of G_* and $P(G_*,k)$ is the number of colorings of G_* with k colors (known as the *chromatic polynomial* of G_*).

Proof. The proof follows directly from Lemma 1.3.8 which shows that each inferred topology has proper colorings, and the fact that a coloring of G_* cannot result in two different inferred topologies, as the coloring uniquely describes which stars to merge (Lemma 1.3.7). In order to account for isomorphic colorings, we need to divide by the number of color permutations. □ □

Note that the fact that G_* can be an arbitrary graph (Lemma 1.3.5) implies that we cannot exploit some special properties of G_* to compute colorings of G_* and $\gamma(G_*)$. Also note that the exact computation of the upper bound is hard, since the minimal coloring as well as the chromatic polynomial of G_* (in P♯) is needed. To complement the upper bound, we note that star graphs with a small number of conflict edges can indeed result in a large number of inferred topologies.

Theorem 1.3.10. Regardless of $\alpha > 0$, there is a trace set for which the number of non-isomorphic colorings of G_* equals $|\mathcal{G}_\mathcal{T}| \leq B_s$, where $\mathcal{G}_\mathcal{T}$ is the set of inferrable and α-consistent topologies, s is the number of stars in $\mathcal{T}$, and B_s is the *Bell number* of s. Such a trace set can originate from a G_0 network with one anonymous node only.

Proof is available in the original publication [103]

Using these observations it is now possible to design an algorithm extracting $\mathcal{G}_\mathcal{T}$: first, *i*) construct G_c and G_* using $\mathcal{T}$ and the parameter α. Then, *ii*) compute all the non isomorphic proper colorings of G_*. Each such coloring γ defines which vertices to merge in G_c to obtain a color-induced topology G_γ. Finally, *iii*) for each color-induced graph G_γ, test whether it is α-consistent with respect to $\mathcal{T}$. Note that each color of a proper coloring γ yields an anonymous router in G_γ. Thus, if one is only interested in minimal topologies, it is possible to compute only the minimal colorings of G_* on step *ii*). More formally, Algorithm 1 summarizes the steps required to produce the set of all inferrable topologies $\mathcal{G}_\mathcal{T}$.

Algorithm 1 Given traces $\mathcal{T}$:

```
Compute G_* and G_C
G_T ← ∅
for all proper colorings γ of G_* do
    G_γ ← G_C
    for all pairs {*_i, *_j} do
        if γ(*_i) = γ(*_i) then
            merge *_i and *_j in G_γ
    G_T ← G_T ∪ G_γ
return G_T
```

1.4 Properties and Full Exploration

Even if the number of inferrable topologies is large, studying trace sets can still be useful if one is mainly interested in some properties of G_0 and if the ensemble $\mathcal{G}_\mathcal{T}$ is homogeneous with respect to these properties; for example, if "most" of the instances in $\mathcal{G}_\mathcal{T}$ the properties are close to G_0, it may be an option to conduct an efficient sampling analysis on random representatives. Therefore, in the following, we will take a closer look on how much the members of $\mathcal{G}_\mathcal{T}$ differ in various aspects.

Important metrics to characterize inferrable topologies are, for instance, the graph size, the diameter $\textsc{Diam}(\cdot)$, the number of triangles $C_3(\cdot)$ of G. In the following, let $G_1 = (V_1, E_1), G_2 = (V_2, E_2) \in \mathcal{G}_\mathcal{T}$ be two arbitrary representatives of $\mathcal{G}_\mathcal{T}$. Inferrable topologies can also differ in the number of connected components. This implies that the shortest distance between two named nodes can differ arbitrarily between two representatives in $\mathcal{G}_\mathcal{T}$.

Another aspect of the usefulness of topology inference depends on the distortion of shortest paths.

Definition 1.4.1 (Stretch)**.** The maximal ratio of the distance of two non-anonymous nodes in G_0 and a connected topology G is called the *stretch* ρ:

$$\rho = \max_{u,v \in \mathcal{ID}(G_0)} \max \left\{ \frac{d_{G_0}(u,v)}{d_G(u,v)}, \frac{d_G(u,v)}{d_{G_0}(u,v)} \right\}.$$

So far, we assumed that the trace set $\mathcal{T}$ contains each node and link of G_0 at least once. At first sight, this seems to be the best we can hope for. However, sometimes traces exploring the vicinity of anonymous nodes in more than one trace yield additional information that help to characterize $\mathcal{G}_\mathcal{T}$ better.

This section introduces the concept of *fully explored networks*: A trace set $\mathcal{T}$ fully explores a network if it contains sufficiently many traces such that the distances between non-anonymous nodes can be estimated accurately.

Definition 1.4.2 (Fully Explored Topologies)**.** A topology G_0 is fully explored by a trace set $\mathcal{T}$ if it contains all nodes and links of G_0 and for each pair $\{u, v\}$ of non-anonymous nodes in the same component of G_0 there exists a trace $T \in \mathcal{T}$ containing both nodes $u \in T$ and $v \in T$.

A trace set for a fully explored network is the optimal input for generic topology inference in the sense that aspects that cannot be inferred well in a fully explored topology model are infeasible to infer without additional assumptions on G_0. Put differently, a fully exploring set of traces has the property that adding any additional traces does not change the set of topologies which are consistent with the given traces. Thus, the full exploration hypothesis provides "upper bounds" on what can be learned from topology inference.

Using these definitions and Algorithm 1, it is possible to compute worst case scenarios in which we maximise the difference between a property of interest on G_0 and its value obtained on a topology inferred from G_0. Figure 12 presents these results for standard properties on both fully explored and arbitrary traces. While full exploration provide guarantees on the global accuracy of the inference (number of components, stretch), it does not prevent arbitrarily bad reconstruction of local properties of G_0 (e.g. triangles).

Property/Scenario	Arbitrary		Fully Explored ($\alpha = 1$)	
	$G_1 - G_2$	G_1/G_2	$G_1 - G_2$	G_1/G_2
# of nodes	$\leq s - \gamma(G_*)$	$\leq (n+s)/(n+\gamma(G_*))$	$\leq s - \gamma(G_*)$	$\leq (n+s)/(n+\gamma(G_*))$
# of links	$\leq 2(s - \gamma(G_*))$	$\leq (\nu + 2s)/(\nu + 2)$	$\leq 2(s - \gamma(G_*))$	$\leq (\nu + 2s)/(\nu + 2)$
# of connected components	$\leq n/2$	$\leq n/2$	$= 0$	$= 1$
Stretch	-	$\leq (N-1)/2$	-	$= 1$
Diameter	$\leq (s-1)/s \cdot (N-1)$	$\leq s$	$\leq s/2$	≤ 2
Max. Deg.	$\leq 2(s - \gamma(G_*))$	$\leq s - \gamma(G_*) + 1$	$\leq 2(s - \gamma(G_*))$	$\leq s - \gamma(G_*) + 1$
Triangles	$\leq 2s(s-1)$	∞	$\leq 2s(s-1)/2$	∞

Figure 12: Summary of our bounds on the properties of inferrable topologies. s denotes the number of stars in the traces, n is the number of named nodes, $N = n + s$, and ν denotes the number of links between named nodes. Note that trace sets meeting these bounds exist for all properties for which we have upper bounds.

1.5 Conclusion

Our work is to be viewed as a first step to shed light onto the similarity of inferrable topologies if the trace sets are based on very basic axioms and no assumptions on the underlying network are taken (e.g., assumptions on power-law properties of the degree distribution). In other words, we consider the worst case: arbitrary networks. Using our formal framework we show that the topologies for a given trace set may differ significantly. Thus, it is impossible to accurately characterize topological properties of complex networks. Note that while this is a negative result from the perspective of application designers who may want to exploit the topology, certain players in the Internet (e.g., ISPs) are interested in keeping their topology as a business secret. [107]

To complement the general analysis, we propose the notion of fully explored networks or trace sets, as a "best possible scenario". As expected, we find that fully exploring traces allow us to determine several properties of the network more accurately; however, it also turns out that even in this scenario, other topological properties are inherently hard to compute. Our results are summarized in Figure 12.

Our work opens several directions for future research. So far we have only investigated fully explored networks with short path routing ($\alpha = 1$), and a scenario with suboptimal routes might lead to different results. One may also study whether the minimal inferrable topologies considered in, e.g., [1, 2], share more similarities than the whole set of inferrable graphs. More importantly, while this work merely presented bounds for the general worst-case, it is of great interest to devise (efficient) algorithms that compute, for a *given trace set*, worst-case bounds for the properties under consideration. For example, such approximate bounds would be helpful to decide whether additional measurements are needed. Moreover, such algorithms may even give advice on the locations at which such measurements would be most useful. Finally, it would also be interesting to study whether the additional knowledge that a network must belong to a certain graph family (e.g., types of backbone networks) may render inference more efficient and accurate.

Chapter 2

Mapping virtual clouds

2.1 Context: Virtualized Networks

Virtualization has arguably been one of the main innovation motors of today's Internet. Already a decade ago, node virtualization revamped the server business, and today's datacenters host thousands of virtual machines. But also links become more virtualized, e.g., through the introduction of Software-Defined Networking [89] technology.

After the virtualization of *nodes* and *links*, the industry as well as the academia discusses the virtualization of entire *networks*: *network virtualization* [34, 62] envisions an Internet where customers (e.g., a startup company or a content distribution provider) can request *virtual networks (VNets)* on short notice and with arbitrary specifications on the required node resources and their connectivity. Indeed, first prototypes have emerged (e.g., in the European CHANGE and UNIFY projects, the American GENI project or the Asian AKARI project). Network virtualization is particularly attractive for Internet Service Providers (ISPs): ISPs benefit from the improved resource utilization as well as from the possibility to introduce new services. [116]

While the last chapter relied on a rather standard link tomography technique (namely traceroute) to capture the IP-level networking infrastructure, this chapter explores a more exotic technique for tomography. Interestingly here, we follow the original paper angle which takes the ISP's perspective and argues that the network virtualization trend also comes with certain threats. In particular, we show that critical information about the ISP's network infrastructure and its properties can be learned from answers to VNet embedding requests. Given that the resource network constitutes a competitive advantage and is a business secret, this is problematic; moreover, the discovery of, e.g., bottlenecks, may be exploited for attacks or bad publicity. Hence, providers around the world are often reluctant to open the infrastructure to novel technologies and applications that might lead to information leaks.

Background on VNets and Embeddings. Basically, a VNet defines a *graph* (henceforth: the *guest graph*): a set of virtual nodes (e.g., virtual machines) which need to be interconnected via virtual links according to the specified VNet topology. This graph is often realized over a physical resource network (also called the *substrate network* or *host graph*): the virtual nodes must be mapped to physical nodes and the virtual links must be mapped to physical *paths* (e.g. realized by OpenFlow [89]).

We consider VNet requests which do not impose any location constraints on where the virtual nodes are mapped to: for example, a computational VNet to process large amounts of scientific data, a datacenter VNet, or a testbed VNet such as Planetlab to experiment with novel networking protocols may not require specific locations. This flexibility in the VNet specification can be exploited to optimize the VNet embedding.

VNets can come in different flavors and with different specifications, and can also provide QoS guarantees such as a minimal bandwidth (given that corresponding traffic shaping and reservation policies are in place). Although VNets appear as dedicated and "real" networks to their users, several VNets can be *embedded* (i.e., realized) over the same infrastructure network (referred to as the *substrate network*); network virtualization technology therefore enables resource reuse. We will prefer the term *substrate* to

the term *infrastructure* as the substrate network may not necessarily be a physical network, but a virtual network itself. [116]

Figure 13 illustrates the VNet embedding problem.

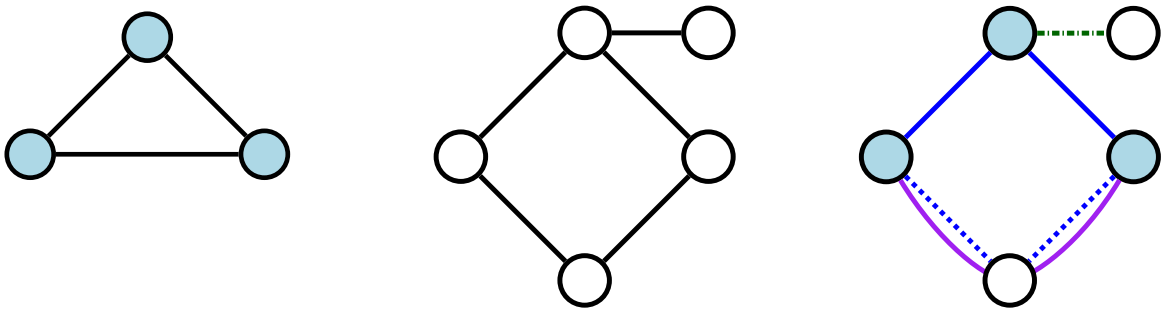


Figure 13: In order to embed a VNet (i.e., a *guest graph* G) with unit link and node demands forming a triangle (*left*) in the substrate network (i.e., a *host graph* H, in the *middle*), resources along four links are allocated in the substrate network (*right*). In the right graph, solid lines represent virtual links mapped on single substrate links, solid curves are virtual links mapped on multiple substrate links, dotted lines are substrate links implementing a multihop virtual link, and dashed lines are unused substrate links.

Contributions.

This chapter reports on the following main contributions:

- We introduce a formal model to study the problem of VNet topology inference. This model allows a customer or attacker to issue the following types of requests: `Is graph` G `(the VNet) embeddable in graph` H `(the substrate)?`

  To measure how quickly a topology can be disclosed, we define the notion of *request complexity*: the number of VNet requests needed to disclose a substrate topology in the worst-case. Our model differs from existing topology discovery problems (such as, e.g., *graph tomography*, see Section 2.6) in that the requests come in the form of entire *graphs* instead of paths only.

  An important assumption used in our work is the honesty of the provider, i.e., we assume that the provider will always accept a VNet request if it has sufficient resources for the embedding.

  We show that we can profit from the *poset* (partially ordered set) properties of the VNet embedding relation, and present algorithms achieving an optimal request complexity for different settings and substrate topologies.

- We derive request complexity bounds for different graph classes. This constitutes a different approach compared to Chapter 1, where we did not exploit particular graph classes. In particular, we show that the request complexity of trees is $O(n)$, while arbitrary graphs have a request complexity of $O(n^2)$, where n is the number of nodes in the substrate. In addition we provide matching lower bounds for these complexities.

- Moreover, we introduce a generalized graph inference framework based on the concept of so-called *motifs* that describes a graph based on components that are glued together. This framework serves as a transition towards the practical approaches presented in the next chapter: although it provides no theoretical guarantees, it performs greatly on the real topologies as those exhibit regularities (motifs) that can be efficiently leverage to speed up the discovery process.

In general, we understand our work as a first step towards a better understanding of the security aspects of VNet embeddings, and believe that our work opens interesting directions for future research.

2.2 Model

We first introduce the VNet topology discovery problem and subsequently describe our algorithmic approach.

2.2.1 VNet Embedding

Our formal setting consists of two entities: a *customer* (the "adversary") that issues virtual network (VNet) requests and a *provider* (a virtual network or infrastructure provider [116]) that performs the access control and the embedding of VNets. We model the virtual network requests as simple, undirected graphs $G = (V, E, w)$ (the so-called *guest graphs*) where V denotes the virtual nodes, E denotes the virtual edges connecting nodes in V, and w denotes the resource requirements of the node resp. link: the weight $w(v)$ describes the *demand* of node $v \in V$ for, e.g., computation or storage, and $w(e)$ describes the bandwidth demand of edge $e \in E$. Similarly, the infrastructure network is given as an undirected graph $H = (V, E, w)$ (the so-called *host graph* or *substrate*) as well, where V denotes the set of substrate nodes and E is the set of substrate links. Again, a weight function w can describe the available resources on a given node or edge.

Without loss of generality, we assume that neither the VNet nor the substrate topologies contain parallel edges or self-loops, and that H is connected. (As we will see, if H consists of multiple connected components, these components can be processed sequentially with our algorithms by keeping the VNet embedding in an already discovered component in order to focus on the next component.) Moreover in this chapter, in order to focus on the topological aspects, we will typically consider host graphs with unit capacities only, i.e., we will assume that $w \equiv 1$. As we will see, for substrate networks with unit capacities, we can also make the guest graph unweighted: all graph classes considered in this chapter can be inferred with an asymptotically optimal complexity using unweighted VNets only.

We also assume that besides the resource demands, the VNet requests do not impose any mapping restrictions, i.e., a virtual node can be mapped to *any* substrate node, and we assume that a virtual link connecting two substrate nodes can be mapped to an entire (but single) *path* on the substrate as long as the demanded capacity is available. These assumptions are common for virtual networks. [34]

A virtual link which is mapped to more than one substrate link however can entail certain costs at the *relay nodes*, the substrate nodes which do not constitute endpoints of the virtual link and merely serve for forwarding. For example, this cost may represent a header lookup cost and may be a function of the packet rate. However, depending on the application, the cost can also be more complex, e.g., in case of a VNet which requires additional functionality at the backbone routers, e.g., to implement an intrusion detection system. We model these kinds of costs with a parameter $\epsilon > 0$ (per link). Moreover, we also allow multiple virtual nodes to be mapped to the same substrate node if the node capacity allows it; we assume that if two virtual nodes are mapped to the same substrate node, the cost of a virtual link between them is zero.

Armed with these definitions, we can formalize our VNet embedding problem.

Definition 2.2.1 (Embedding π, Validity Properties, Embedding Relation $\mapsto$)**.** An *embedding* of a graph $A = (V_A, E_A, w_A)$ to a graph $B = (V_B, E_B, w_B)$ is a mapping $\pi : A \to B$ where every node of A is mapped to exactly one node of B, and every edge of A is mapped to a path of B. That is, π consists of a node mapping $\pi_V : V_A \to V_B$ and an edge mapping $\pi_E : E_A \to P_B$, where P_B denotes the set of paths. We will refer to the set of virtual nodes embedded on a node $v_B \in V_B$ by $\pi_V^{-1}(v_B)$; similarly, $\pi_E^{-1}(e_B)$ describes the set of virtual links passing through $e_B \in E_B$ and $\pi_E^{-1}(v_B)$ describes the virtual links passing through $v_B \in V_B$ with v_B serving as a relay node.

To be *valid*, the embedding π has to fulfill the following four properties: (i) Each node $v_A \in V_A$ is mapped to exactly one node $v_B \in V_B$ (but given sufficient capacities, v_B can host multiple nodes from V_A). (ii) Links are mapped consistently, i.e., for two nodes $v_A, v'_A \in V_A$, if $e_A = \{v_A, v'_A\} \in E_A$ then e_A is mapped to a single (possibly empty and undirected) path in B connecting nodes $\pi(v_A)$ and $\pi(v'_A)$. A link e_A cannot be split into multiple paths. (iii) The capacities of substrate nodes are not exceeded: for all $v_B \in V_B$: $\sum_{u \in \pi_V^{-1}(v_B)} w(u) + \epsilon \cdot |\pi_E^{-1}(v_B)| \leq w(v_B)$. (iv) The capacities in E_B are respected as well, i.e., for all $e_B \in E_B$: $\sum_{e \in \pi_E^{-1}(e_B)} w(e) \leq w(e_B)$.

If there exists such a valid embedding π, we say that graph A can be embedded in B, indicated by $A \mapsto B$. Hence, $\mapsto$ denotes the VNet *embedding relation*.

Definition 2.2.2 (Embedding Cost)**.** The cost associated with an embedding π is denoted by

$$\text{Cost}(\pi) = \sum_{v_A \in V_A} w(v_A) + \sum_{e_A \in E_A} w(e_A) \cdot |\pi^{-1}(e_A)| \tag{2.1}$$

$$+ \epsilon \cdot \sum_{v_B \in V_B} |\pi_E^{-1}(v_B)| \tag{2.2}$$

The provider has a flexible choice where to embed a VNet as long as a valid mapping is chosen. In order to design topology discovery algorithms, we can exploit the *poset* property of the embedding relation. The proof of the following lemma appears in the original paper.

Lemma 2.2.3. The embedding relation $\mapsto$ applied to any graph family $\mathcal{G}$ (short: $(\mathcal{G}, \mapsto)$), forms a partially ordered set (a *poset*), i.e., it satisfies reflexivity, antisymmetry and transitivity.

Observe that even though the VNet embedding is similar to the minor graph relation, there are some important differences. The following lemma shows that the minor graph relation and the embedding relation do not imply each other. The proof appears in the appendix.

Lemma 2.2.4. Given two graphs A, $B \in \mathcal{G}$ with unit capacity it holds that (i) $A \mapsto B$ implies that A is a minor of B for $\epsilon > 0.5$. (ii) For smaller ϵ it holds that A may be embeddable in B even if A is not a minor of B. (iii) Not every minor A of a graph B can be embedded in A.

2.2.2 Request Complexity

We assume the perspective of a customer (an "adversary") that seeks to disclose the (fixed) infrastructure topology of a provider with a minimal number of requests. These requests (and the answers to them) are the only means of obtaining information. As a performance measure, we introduce the notion of *request complexity*, i.e., the number of VNet requests which have to be issued until a given network is fully discovered, i.e., all vertices, edges and capacities are known to the adversary. The motivation for focusing on a request measure is based on the fact that a request constitutes the natural unit of negotiation between customer and provider, and issuing a request entails a certain overhead for both sides. In particular a situation where after the provider's answers, the customer repeatedly revokes her request, can be perceived as a denial-of-service attack. Algorithms achieving a low request complexity hence also ensure that adversarial information discovery activities remain hidden.

We are interested in algorithms that "guess" the target topology H (the host graph) among the set $\mathcal{H}$ of possible substrate topologies allowed by the model. Concretely, we assume that given a VNet request G (a guest graph), the substrate provider always responds with *an honest (binary) reply* R informing the customer whether the requested VNet G is embeddedable on the substrate H. In the following, we will use the notation $\texttt{request}(G, H)$ to denote such an embedding request of G to H, and the provider will answer with the binary information whether G is embeddable in H (short: $G \mapsto H$). Based on this reply, the customer may then decide to ask the provider to embed the corresponding VNet G on H, or it may not embed it and continue asking for other VNets.

Let ALG be an algorithm that issues a series of requests $G_1, \ldots, G_t$ each consisting of a request graph to reveal H.

Definition 2.2.5 (Request Complexity)**.** The *request complexity* to infer the topology is measured in the number of requests t (in the worst case) until ALG issues a request G_t which is isomorphic to H and *terminates* (i.e., ALG knows that $H = G_t$ and does not issue any further requests).

2.2.3 Additional Complexities

Depending on the perspective, our model can be regarded as overly conservative or overly optimistic, in the sense that in reality, the request complexity may likely be lower or higher. In the following, we elaborate on some of our assumptions and discuss their implications.

Recall that we assume that the attacker only gets a *binary* response, whether the VNet can be embedded or not: thus, minimal information about the provider network is revealed, entailing a high request complexity. Alternative models may reduce the number of requests needed to infer the topology. For example, a plausible alternative may be a "valued reply model" where the provider returns the embedding *cost* $\text{Cost}(\texttt{request}(G,H))$. (The binary reply model can be emulated with this model by using the variable $(\text{Cost}(\texttt{request}(G,H)) < \infty)$.)

Moreover, we rely on the assumption that the provider always gives an *honest answer*. While there are algorithms to compute such honest answers [115], the problem is related to graph isomorphism testing and is generally NP-hard (see e.g., [8]). Thus, in practice, a provider may use approximation algorithms and heuristics to embed VNets (see e.g. [34]), and may not accept a VNet although it is theoretically embeddable. In addition, a provider being aware of potential attacks may proactively randomize its responses.

In other words, by focussing on the idealized case where providers answer embedding requests correctly and honestly, we investigate a worst case situation for the provider. Hence the results of this work determine how quickly and how much information on the host network can be revealed in circumstances that are ideal for attackers. These findings can help providers to decide on the tradeoff of a comprehensive client offering and the cost/benefit of attack countermeasures.

Besides the *request complexity* and the *embedding complexity*, there is a third relevant complexity which has not been discussed so far: the complexity of generating the attack sequence, i.e., generating the request G_{i+1} out of G_i given the provider answer. We will refer to this complexity as the *generation complexity*. While the generation complexity plays a secondary role in this approach, all presented algorithms are essentially greedy and efficient; an exception is the dictionary framework which relies on a pre-computed directed acyclic graph.

2.2.4 Terminology and Notation

The algorithms presented in this chapter are often based on simple graphs (so-called *motifs*) such as a *chain*, a virtual link connecting two virtual nodes, or a *cycle*, three virtual nodes connected as a triangle.

Definition 2.2.6 (Chain C, Cycle Y, Diamond D, Bipartite Motif $K_{x,y}$, Cliques K_x)**.** The motif C denotes a Chain graph, a virtual edge that maps to a path on the substrate: $C = (V = \{u,v\}, E = \{\{u,v\}\})$. Motif Y is a cYcle, i.e., a virtual triangle that maps to a cycle in the substrate: $Y = (V = \{u,v,w\}, E = \{\{u,v\},\{u,w\},\{w,v\}\})$. The Diamond motif D is a complete graph over four nodes with one missing link. Finally, $K_{x,y}$ is a complete bipartite graph with $x+y$ nodes, and K_x is a clique with x nodes.

These basic motifs are then attached to each other to grow more complex VNet requests. We will often make use of a *graph grammars* notation [29] in order to describe the graph exploration iteratively.

When two motifs are attached to each other we use the term *concatenation* to describe that two motifs are glued to each other selecting one of the nodes of each of them as *attachment points* and merging them.

Definition 2.2.7 (Concatenation, $prefix$, $postfix$)**.** We will use the notation M^j to denote the *concatenation* of j motifs M (where the attachment points must be clear from the context). Moreover, given two graphs G_1 and G_2 containing nodes u,v, the notation $G_1 v G_2(u)$ denotes a graph where the node $v \in G_1$ is merged with the node $u \in G_2$, i.e., the edges of G_2 are added to the set of G_1's edges and the corresponding nodes to the set of G_1's nodes; the node $u \in G_1$ is optional if clear from the context. Given a sequence S of motifs attached to each other, and a particular motif M belonging to this sequence, then $prefix(M,S)$ and $postfix(M,S)$ denote the subsequences of S before and after this motif M.

2.3 Adversarial Topology Discovery

This section presents tight bounds for the discovery of *trees* and *general graphs*.

2.3.1 Trees

We start with a simple observation: it is easy to decide whether the host graph forms a tree or not. A topology discovery algorithm ALG can test if the substrate $H \in \mathcal{H}$ is tree-like using a single request: ALG simply asks for a triangle network (i.e., a complete graph K_3 consisting of three virtual nodes, with unit virtual node and link capacities). The triangle can be embedded if and only if H contains a cycle. Once it is known that the set of possible infrastructure topologies (or host graphs) $\mathcal{H}$ is restricted to trees, the algorithm described in this section can be used to discover them. Moreover, as we will see, the algorithm presented in this section has the property that if $H \in \mathcal{H}$ does contain cycles, it automatically computes a *spanning tree* of H.

The tree discovery algorithm TREE (see Algorithm 2 for the formal listing) described in the following is based on the idea of incrementally growing the request graph by adding *longest chains* (i.e., "branches" of the tree). Intuitively, such a longest chain of virtual nodes will serve as an "anchor" for extending further branches in future requests: since the chain is maximal and no more nodes can be embedded, the number of virtual nodes along the chain must equal the number of substrate nodes on the corresponding substrate path. The endpoints of the chain thus cannot have any additional neighbors and must be tree leafs (we will call these nodes *explored*), and we can recursively explore the longest branches of the so-called *pending nodes* discovered along the chain.

More concretely, TREE first discovers the overall longest (cycle-free) chain of nodes in the substrate tree by performing binary search on the length of the maximal embeddable path. This is achieved by requesting, in request R_i, a VNet of 2^i linearly connected virtual nodes (of unit node and link capacities); in Algorithm 2, we refer to a single virtual link connecting two virtual nodes by a *chain* C, and a sequence of j chains by C^j. The first time a path of the double length 2^i is not embeddable, TREE asks for the longest embeddable chain with 2^{i-1} to $2^i - 1$ virtual nodes; and so on. Once the longest chain is found, its end nodes are considered *explored* (they cannot have any additional neighbors due to the longest chain property), and all remaining virtual nodes along the longest chain are considered *pending* (set $\mathcal{P}$): their tree branches still need to be explored. TREE then picks an arbitrary pending node v and seeks to attach a maximal chain ("branch") analogously to the procedure above, except for that the node at the chain's origin is left pending until no more branches can be added. The scheme is repeated recursively until there are no pending nodes left. Formally, in Algorithm 2, we write GvC to denote that a chain C is added to an already discovered graph G at the virtual node v.

Algorithm 2 Tree Discovery: TREE

```
1: G := {{v},∅}   /* current request graph */
2: P := {v}   /* pending set of unexplored nodes*/
3: while P ≠ ∅ do
4:    choose v ∈ P, S :=exploreSequence(v)
5:    if S ≠ ∅ then
6:       G := GvS, add all nodes of S to P
7:    else
8:       remove v from P
```

exploreSequence(v)

```
1: S := ∅
2: if request(GvC, H) then
3:    find max j s.t. GvC^j ↦ H (binary search)
4:    S := C^j
5: return S
```

Theorem 2.3.1. Algorithm TREE is correct and has a request complexity of $O(n)$, where n is the number of substrate nodes. This is asymptotically optimal in the sense that any algorithm needs $\Omega(n)$ requests in the worst case.

Proof. Correctness: Since the substrate network is connected, each node can be reached by a path from any other node. As the algorithm explores each path attached to a discovered node until no more nodes can be added, every node is eventually found. Since a tree is cycle-free, this also implies that the set of discovered edges is complete.

Complexity (upper bound): We observe that our algorithm has the property that at time t, it always asks for a VNet which is a strict super graph of any embeddable graph asked at time $t' < t$ (positive answer from the provider). Moreover, due to the exponential binary search construction, TREE issues $O(\log \ell)$ requests to discover a chain consisting of ℓ links. The cost of exploring a path can be distributed among its constituting links, thus we have an accounting scheme which shows that the amortized cost per link is constant: As there are at most $n-1$ links in a tree, the total number of requests due to the link discovery is linear in n as well. In order to account for requests at nodes that do not have any unexplored neighbors and lead to marking a node explored (at most one request per node), $O(n)$ requests need to be added.

Complexity (lower bound): The lower bound follows from the cardinality of the set of non-isomorphic trees, which is in the order of $2.96^n/n^{5/2}$ [106]. Since any discovery algorithm can only obtain a binary information for each request issued, a request cuts the remaining search space in (at most) half. Therefore, the request complexity of any algorithm is at least $\Omega(\log(2.96^n/n^{5/2})) = \Omega(n)$. □

Although TREE looks simple, one has to be careful to avoid pitfalls when trying to design tree discovery algorithms. For instance, consider a scheme where instead of searching for longest chains, we want to find the nodes of largest degree, e.g., by performing binary search on the neighborhood size of a given virtual node, and then recursively explore the discovered pending nodes by adding newly found nodes to the pending set: Since pending nodes are connected by virtual links to each other, they may physically be separated by many hops, and one has to ensure that the substrate nodes along these paths are not forgotten. Although these problems can be handled within the same asymptotical complexity, we do not elaborate on these directions more here.

Observe that TREE has the nice property that if H is not a tree, TREE simply computes a spanning tree of H by extending maximal (cycle-free) branches from the nodes.

Corollary 2.3.2. TREE determines a spanning tree of any graph with request complexity $O(n)$.

2.3.2 Arbitrary Graphs

Let us now turn to the general problem of inferring arbitrary substrate topologies. First note that even if the total number of substrate nodes is known, the adversary cannot simply compute the substrate edges by testing each virtual link between the node pairs: the fact that the corresponding virtual link can be embedded does not imply that a corresponding substrate link exists, because the virtual link might be mapped across an entire substrate *path*. Nevertheless, we will show in the following that a request complexity of $O(n^2)$ can be achieved; this is asymptotically optimal.

The main idea of our algorithm GEN is to build upon the TREE algorithm to first find a spanning tree (see Corollary 2.3.2). This spanning tree (consisting of pending nodes only) "reserves" the resources on the substrate nodes, such that they cannot serve as relay nodes for virtual links passing through them. Subsequently, we try to extend the spanning tree with additional edges. An arbitrary pending node u is chosen, and we try to add an edge to any other pending node v in the spanning tree. After looping over all pending nodes and adding the corresponding links, u is marked *explored*. GEN terminates when no more pending nodes are left.

Theorem 2.3.3. A general graph can be discovered with request complexity $O(n^2)$. This is asymptotically optimal.

Proof. Upper bound: According to Corollary 2.3.1, computing a spanning tree G using TREE requires $O(n)$ requests. Subsequently, GEN asks for each pair of pending nodes if an edge between them can be

added without destroying embeddability. Since the spanning tree is used as a basis, and since $\epsilon > 0$, two nodes can be connected if and only if there is a direct substrate link between them; otherwise, the capacity constraints would be violated. Thus one request per possible substrate link is sufficient, which results in at most $O(n^2)$ requests.

Lower bound: The number a_n of non-isomorphic simple graphs on n vertices satisfies $2^{\binom{n}{2}}/n! \leq a_n \leq 2^{\binom{n}{2}}$. By the same argument as for the lower bound of the request complexity for trees, $\Omega(\log a_n) = \Omega(n^2)$ requests are necessary in the worst case. □

2.4 Motif Framework

We have shown that tree graphs can be discovered in time $O(n)$, while general graphs require time $O(n^2)$. This raises the question whether there exist additional graph families, which still allow for a sub-quadratic complexity. In the original paper, we answer this question affirmatively using a framework for arbitrary graph inference. The framework is based on three principles: (1) growing request graphs iteratively, (2) using simple motifs to reserve entire subgraphs while exploring the detailed structure of the subgraph only later, (3) requesting more highly connected motifs first.

Concretely, our algorithm Dict is based on the observation that given a spanning tree, it is very costly to discover additional edges between nodes in 2-connected components: essentially, finding a single such edge requires testing all possible node pair combinations, which is quadratic in the component size. Thus, our algorithm iteratively first explores the basic "knitting" of the topology, i.e., it discovers first a sequence of maximal minors which are at least 2-connected (the *motifs*). Dict maintains the invariant that there are never two nodes u, v which are not k-connected in the currently requested graph H' while they are k-connected in H; no path relevant for the connectivity of the current component is overlooked and needs to be found later. Subsequently, nodes and edges which are not contributing to the connectivity of the current component can then be discovered by (1) *edge expansion* where additional nodes of degree two are added along a motif edge, and by (2) adding sequences of motifs to the nodes iteratively with the same process.

Figure 2.4 gives a simplified overview of our discovery strategy: in a first phase, we seek to discover the highly connected parts of the physical network (and reserve the corresponding resources); this is achieved by trying to embed sequences of so-called "motifs" of the highest connectivity and subsequently shifting toward less connected motifs. Nodes of degree two are found using *edge expansion*, that iteratively tries to reserve as many node as possible along each identified edge.

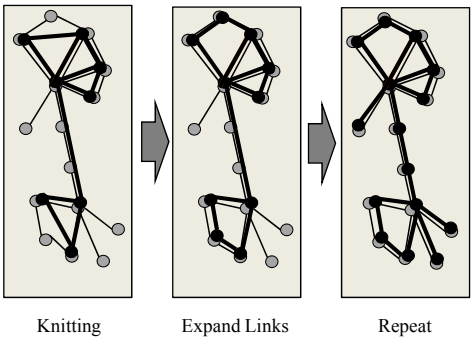


Figure 14: Algorithm Dict first discovers and reserves sequences of highly connected parts of the host graph. Then, less connected parts are discovered by edge expansion and then the process is repeated until no additional nodes and edges can be found.

With the above intuition in mind, we give a short overview of this approach and emphasize its relevant

aspects with respect to graph tomography. The interested reader is referred to the original paper for a more detailed presentation of the DICT algorithm.

Some sparse graph classes can be described using only motifs. For example, for a tree graph we only need a single motif, the *chain* C (two nodes connected by a link); since there are no 2-connected motifs in a tree, it can be discovered by attaching chains C to each other.

Another interesting graph family are *cactus graphs*: connected graphs in which any two simple cycles have at most one vertex in common. Cactus graphs are interesting in the sense that many backbone or core networks are reminiscent of cacti (see, e.g., the topologies collected in the Rocketfuel project [122]). In terms of motifs, since a cactus graph may contain cycles, besides the chain C we also need a cycle motif Y: three nodes connected in a triangle manner; however, C and Y are also sufficient to discover all cactus graphs.

More complicated graphs which include higher connected parts may require, e.g., motifs describing diamonds D or cliques K_4. A method extracting motifs for a given graph family is provided in the original paper.

The first idea behind our algorithm DICT is that once a motif is embedded, it occupies resources on the entire host subgraph, and the discovery of the remaining parts of the subgraph where the motif is embedded can be postponed to some later point: For example, Figure 2.4 (*left*) shows a motif occupying resources on the subgraph at the top; additional nodes (Figure 2.4 (*middle*)) and edges (Figure 2.4 (*right*)) can be discovered later.

However, in order to discover entire graphs, multiple motifs are needed: motifs must be attached to each other; accordingly, we introduce the concept of *motif sequences* and *attachment points*. Moreover, for DICT to be correct, it is important that more highly connected motifs are embedded first: a less connected motif may be embeddable on a subgraph but may "overlook" (i.e., not allocate resources on) some links. The situation is complicated by the fact that a graph A may not be embeddable on graph B and vice versa, but A may be embeddable on *two* graphs B which are attached to each other. This motivates the dictionary concept where motifs are organized in an acyclic directed graph.

Our main result shows that DICT correctly discovers a given topology by respecting the order of the dictionary of the corresponding graph family.

2.5 Simulations

To complement our theoretical results and to validate our framework on real network topologies, we dissected the *ISP topologies* provided by the Rocketfuel [122] mapping engine. In particular, given the high execution times of DICT in the worst-case and for complex motif sets, we were interested in (1) the structure and complexity of the motif sets of these topologies, and in (2) the question whether an efficient and approximate variant of DICT could be used in practice to infer large parts of the network in linear time.

For this purpose, we implemented our motif extraction algorithm. Concretely, for each Rocketfuel topology, we first isolate the subgraphs that are trees (i.e., the 1-connected subgraphs), by recursively removing leaf-nodes. The remainder of the topology is then simplified by contracting degree-2 nodes that would be detected during the algorithms' edge expansion phases. The result is a topology only containing motifs connected by "articulation vertices", which provide us with a mean to distinguish motifs. Finally, the extracted motifs are compared pairwise to distinguish between isomorphic and non-isomorphic ones.

Figure 15 *a)* provides some statistics about the considered AS router-level topologies: their overall size, the number of nodes belonging to a 1-connected subgraph, the number of nodes belonging to a motif, and the number of nodes belonging to the largest motif of each topology. For instance, the figure shows that the AS 1221 topology contains 2669 nodes, but only 310 nodes are part of a bi-connected component. Among those nodes, 47 are expanded degree-2 nodes, and 48 are part of motifs containing only 3 or 4 nodes. The remaining 215 nodes compose the "network heart" (containing 716 edges out of the topologies' 3175 edges): such a single highly-connected motif is typical and appears for many AS topologies. One takeaway from this plot is that, since most Rocketfuel networks are built of simple motifs and since DICT discovers both tree and relay nodes easily, most of each topology could be discovered

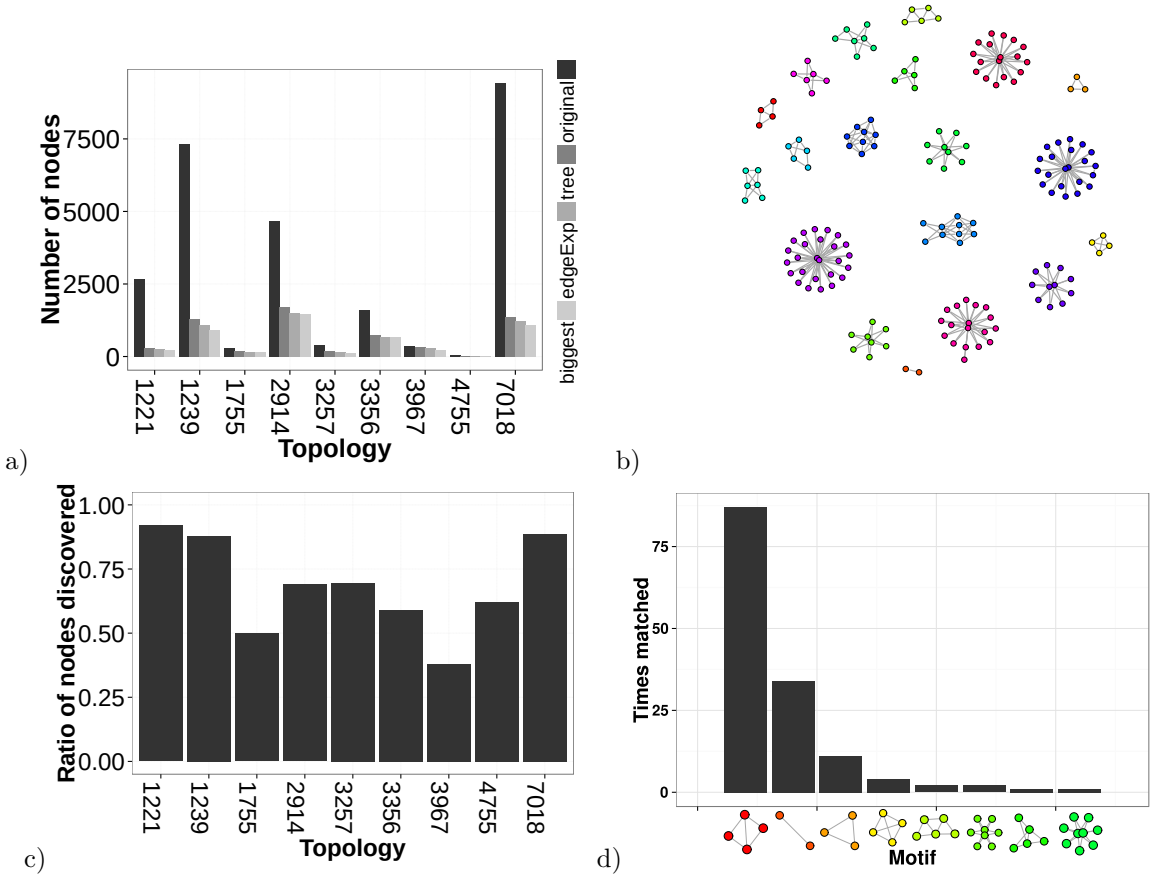


Figure 15: Results of DICT when run on different Internet and power grid topologies. **a)** Number of nodes in different autonomous systems (AS). We computed the set of motifs of these graphs and counted the number of remaining nodes after removing the nodes that : (i) belong to a tree structure at the "fringe" of the network, (ii) have degree 2 and belong to two-connected motifs, and finally (iii) are part of the largest motif. **b)** The 19-motif dictionary built from the dataset. **c)** The fraction of nodes that can be discovered with the dictionary. **d)** Most frequent motifs encountered in the dataset.

quickly with an *approximate* DICT algorithm containing merely the basic motifs: only the few more complex motifs in the network heart require an exhaustive link exploration. In other words, the vast majority of the topology can be discovered in linear time.

Figures 15 *b)* and *c)* further explore the fraction of nodes that can be discovered by DICT restricted to a small motif set. For this purpose, we built a dictionary containing all the motifs identified in our dataset, and removed the biggest motif of each topology (the "network heart"). The remaining 19 motifs are depicted Figure 15 *b)*: they are surprisingly simple and symmetric. Figure 15 *c)* shows the fraction of nodes in perfectly inferred subgraphs: we see that a 19-motifs dictionary is sufficient to explore from 37 to 92% of the nine considered AS topologies. Figure 15 d) represents the frequency distribution of the most common motifs in the nine topologies. Note that interestingly, motif Y only comes third, with 11 occurrences across the dataset.

To conclude our glimpse into ISP network motifs, Figure 16 shows an example how the motifs cover a specific AS topology (namely AS-4755).

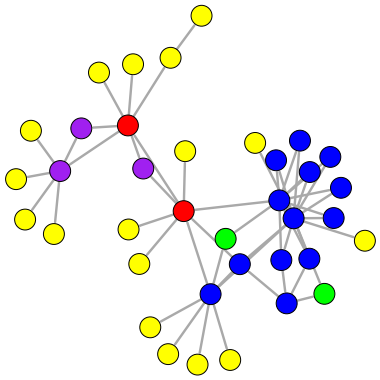


Figure 16: Representation of the AS-4755 network where tree nodes are colored yellow, relay-nodes are green, attachment point nodes are red, cYcle motifs nodes are purple and the maximal motif nodes are blue.

2.6 Related Work

Our work is motivated by the virtualization trend in today's Internet and especially network virtualization. For a general introduction and a good survey, we refer the reader to [34]. Our model differentiates between a customer (e.g., a service provider requesting VNets) and a substrate provider (e.g., a physical infrastructure provider or a virtual network provider). In the terminology of [116], our customer is the SP, and the provider may either be the PIP or the VNP.

VNet Embedding. The embeddings of VNets is an intensively studied problem and there exists a large body of literature (e.g., [51, 86, 135]), and there also exist distributed computing approaches [64] and online algorithms [13, 48]. Our work is orthogonal to this line of literature in the sense that we assume that an (arbitrary and not necessarily resource-optimal) embedding algorithm is given. Rather, we focus on the question of how the feedback obtained through these algorithms can be exploited, and we study the implications on the information which can be obtained about a provider's infrastructure.

Topology Inference. Our work studies a new kind of topology inference problems. Traditionally, much graph discovery research has been conducted in the context of today's complex networks such as the Internet which have fascinated scientists for many years, and there exists a wealth of results on the topic. One of the most influential measurement studies on the Internet topology was conducted by the Faloutsos brothers [50], and their work has subsequently been intensively discussed both in practical [80] and theoretical [4] papers. The classic instrument to discover Internet topologies is *traceroute* [32], but as we've seen in Chapter 1, stars (i.e., anonymous nodes) renders the accurate characterization of Internet topologies difficult [2, 104, 134]. *Network tomography* is another important field of topology discovery. In network tomography, topologies are explored using pairwise end-to-end measurements, without the cooperation of nodes along these paths. This approach is quite flexible and applicable in various contexts, e.g., in social networks. For a good discussion of this approach as well as results for a routing model along shortest and second shortest paths see [9]. For example, [9] shows that for sparse random graphs, a relatively small number of cooperating participants is sufficient to discover a network fairly well.

Both the traceroute and the network tomography problems differ from our virtual network topology discovery problem in that the exploration there is inherently path-based while we can ask for entire virtual graphs.

Virtualization and Security. The benefits and threats of virtualization are extensively studied but still not well-understood. A complete review of the research is beyond the scope of this document,

and we refer the reader to the recent literature, e.g., on virtual machine collocation attacks [111].

Graph Grammars and Mining. We decided to describe our algorithms using a graph grammar formalism, and indeed, some graph grammar problems share commonalities with our work. Graph grammars are a powerful tool to generate and characterize topologies, and we refer the interested reader to, e.g., [29]. More remotely, our work also has connections with *graph data mining* [131]. For instance, in [37], an algorithm is presented to search subsequences which can best compress an input graph based on a minimum description length principle. Although these algorithms pursue a different goal, the computationally-constrained beam search is reminiscent of some of our techniques as nodes are incrementally (and greedily) expanded.

2.7 Conclusion

This chapter has provided an algorithmic perspective on topology discovery in network virtualization environments. We presented tight bounds for three important graph classes. We understand our results as a first step to shed light on possible security threats of the virtualization technology.

We find that the topology of typical sparse backbone networks such as tree or cactus graphs can be discovered relatively fast (request complexity $O(n)$). However, the motif-based topology discovery framework we sketched suggests that the request complexity for denser graphs is higher, as the number of graph knittings can grow combinatorially, and one has to resort to testing edges individually (after computing a spanning tree) which yields a quadratic request complexity.

These results open several interesting directions for future research. First, more work is needed to understand the implications of the framework on other important graph classes, such as different types of *planar graphs*: in order to beat the trivial $O(n^2)$ upper bound on the request complexity, additional dictionary properties must be exploited.

Moreover, in the context of Tree we have seen that if the host graph is not a tree, running Tree will simply result in a spanning tree. This raises the question whether similar spanning structures can be computed with incomplete motif sets, resulting in the "densest spanning structures" *given these motif sets*. For example, when applying Cactus to a general graph, which fraction of edges will be discovered?

While our work so far has focused on discovering the *entire* topology, specific graph properties may be inferred much more efficiently. For instance, identifying a minimal cut efficiently would have strong implications in the security domain.

Finally, a benefit of these algorithms is to allow the precise targeting of nodes, which could be exploited to infer non-topological properties of the network such as latency, throughput, etc.

Part III
Data-oriented approaches

In the introduction, we presented graph as a fruitful abstraction to represent real systems. However, the first part of this thesis, dedicated to a theoretical modeling of some graph observation processes, mostly brought bad news. Indeed, we've shown both that tomographying a graph is difficult, and that measurement errors can have a strong influence on the properties of the considered graph.

In other words it is, at least from a worst case perspective, very easy to draw wrong conclusions from a graph that inadequately represents the (real) system of interest. On the other hand, the widespread use of graphs by the scientific community suggests that sampling errors are in practice either very rare (that is, the studied fault model is too strong compared to reality), or that those errors do not impede the practical relevance of these models (that is, the resulting error bounds are too pessimistic).

Bottom-up versus Top-down in graph metrology

To understand a bit more the relation between systems, their abstraction as graphs, and the application of those, I believe it is important to see a concrete example of modeling of a system by graphs, and their applications. Such approach can be seen as complementing the "top-down" theoretical study of the first chapters, with a "bottom-up" practical aspect, that aims at producing graphs based on data collected about a system.

This data-based approach has some other advantages beyond its complementarity with a theoretical analysis. First, I believe practice is an important dimension of metrology: if one wants to measure the impact of errors in the process of measuring something, understanding how this measurement is concretely carried out and what are its objectives is essential.

Second, the landscape of the eternal struggle between theory-based and data-based approaches in computer science has evolved. The growing abundance of data impacts science both by increasing the demands of the scientific community in terms of experimental justifications – therefore impacting the tools one has to use to produce accepted science, but also by changing the scientific methodology itself. Indeed data is not a clue one has to collect to support a predefined theory anymore: the wealth of data readily available allows more explorative approaches.

Capturing social networks

While providing a different perspective on graph metrology, this part also tackles the construction and the inference of a particular type of graphs: social networks. Indeed, while the introduction and the first chapter exploited graphs to represent man-made systems, graphs are also commonly used to capture natural systems. Arguably one of the most studied of such systems is the social structure of humans.

The notion of social network as we know it today seems to originate from J.L. Moreno in his 1934 book "Who shall survive" [93], where he represented the social relations between childen of a public school (see Figure 17) as a graph. Since then, many sociological studies focused on the structure of human interaction networks.

One of the most notable success of this thread of research is probably Milgram's letter experiment, that led to the notion of small world effect and the world's "six degrees of separation". Figures 18a and 18b are extracted for the original publication in 1967 [128].

Social networks are an interesting subject in the context of graph metrology due to the elusive and plastic definition of *interaction*. Indeed, in most of these studies, the individuals (vertices of the graph) are easily identified. However, characterizing the existence of an edge within the graph that corresponds to a social interaction between the individuals is a difficult task. First, one has to agree on a definition of social interaction and second, one has to be able to measure it. These difficulties directly impact the existence and availability of a ground truth, which would be in our context a consensual and provable reference social network against which we can compare our inferred networks to measure the quality of the inference.

In the context of computer networks, due to the declarative nature of those networks, such ground truth often exists even if it cannot be accessed in practice: for instance, in the context of Chapter 1,

the ground truth can be extracted from the union of each individual router's configuration. Note that in cases where the ground truth is easily available, the interest of developing inference algorithms is minored: one just has to assess the ground truth.

Regarding social networks, first there is no consensual way to project the hole complexity of interpersonnal relationships on a binary notion (link/no link). There is for instance a large thread of research on the notion of (social) "tie" strength [59]. Second, assuming one settles on a consensual notion of social link, there is generally no easy way to assess it. While interviews are considered the gold standard by some sociologists, those still suffer a number of shortcomings (see e.g. [114]). Moreover, those networks are dynamic, and one has to design an experimental setup where the action of measuring the network does not influence the network itself.

Interestingly, in [128] Travers and Milgram apply a tomography technique that is very similar to the one studied in Chapter 1: by capturing only some paths of the global social structure of the USA, he collected enough data to support a general theory about the organisation of networks that has found applications way beyond social networks.

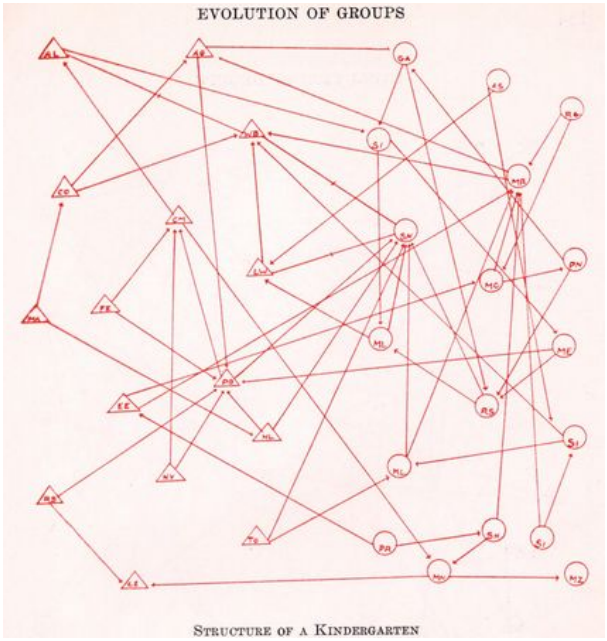


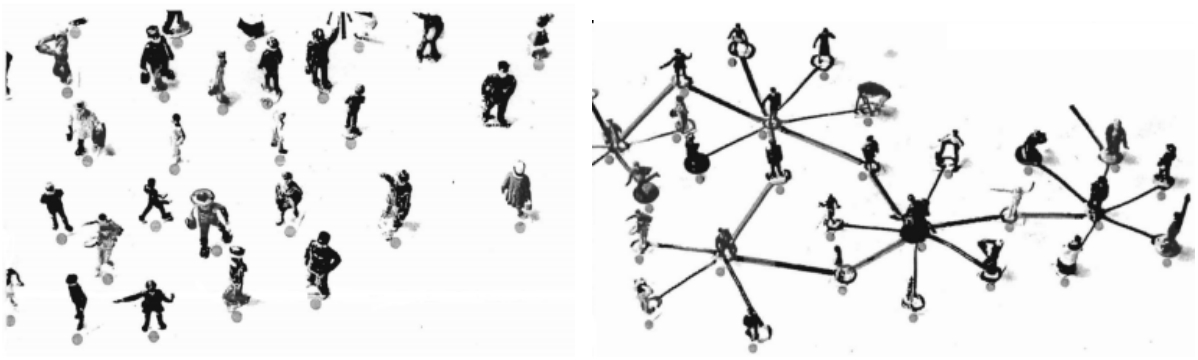
Figure 17: One of the first social networks, representing here the affinities in a kindergarten. Triangular and circular nodes correspond to girls and boys, respectively. The criteria for a link is "Studying and playing in proximity, actually sitting beside" [93]

Social Networks and graph metrology

In the context of this thesis, this part focuses on exploring parts of $\mathcal{G}$ that represent human interaction graphs extracted from proximity information.

One first observation is that those graphs are tied to a geography. More precisely, in all the graphs considered in this part, the nodes also have a physical location, and the existence of an interaction between two nodes is related to their respective location: these are *geometric graphs*. As a result, all these graphs only span a small subpart of $\mathcal{G}$: for instance in practice, they are all sparse and usually have a high clustering coefficient. Moreover, theory shows that if the existence of an interaction is decided by a fixed distance interval between the corresponding nodes (that is, there exists two bounds d_{min} and d_{max} s.t. $(i,j) \in E$ iff $d_{min} \leq d_2(i,j) \leq d_{max}$), we can derive an upper bound on the chromatic number of the resulting graphs (for instance if $d_{min} = d_{max}$, we fall in the Hadwiger–Nelson problem, resulting graphs are 7-colorable at most [120]).

However, these properties concerning geometric graphs are probably not enough to precisely define



(a) "Random dispersement of people in the small world" (b) "The network spreads, with complicated inter-connections"

Figure 18: Illustrations extracted from Milgram's small world experiment (1967) [128]

the subspace of $\mathcal{G}$ containing human interaction graphs. Using such analytical approach, we can in the good cases come up with a list of properties those graphs have to satisfy, but it is difficult to evaluate the coverage of such approach: namely, how precise is my characterisation. In addition, entanglement of geometry and graphs quickly renders analytical approaches difficult. To complement this approach, a common methodology is to *i*) collect many such graphs, and study their common properties to then *ii*) devise a model that recreates the observed behaviors. Along the process, comparing the properties distribution of the synthetic (*i.e.* from a model) and real (*i.e.* from measurements) graphs allow to evaluate the accuracy of the approach.

The first chapter shows that the observed graphs are radically different from what one could create using synthetic mobility models – therefore indicating that human interaction graphs are more than simple objects randomly moving on the euclidean plane. The second chapter on the other hand shows that with sparse information about the position of the individuals, it is possible to infer the social network of interactions that happened during an event. The combination of these observations in my humble opinion suggest a strong entanglement between social and physical proximity –that may be illustrated by the expression "close friends".This is not new: for instance Moreno itself introduced the notion of psychogeography to refer to the relation between physical distance and social relations. This also suggests that using mobility models to create synthetic interaction graphs is not the most efficient approach, and that directly creating dynamic interaction models would be more efficient.

The Souk experimental platform

This part presents the works conducted around the Souk experimental platform. Souk is sought as a tool to capture interactions in crowds. More precisely, Souk captures the positions of individuals in a room. Based on those positions, it is possible to construct a certain number of graphs. In the following chapter, we will briefly present the physical platform and present two use-cases.

I believe it is important to emphasize the considerable amount of time needed for those measurement campaigns. The radically different approach (compared to theoretical approaches), and the necessity to manipulate large amounts of data in rather explorative approaches, were the opportunity to learn a large array of new techniques and methods. Moreover, Souk, thought initially as a demonstrator to allow the public to more concretely visualise graphs, has generated a wide thread of interedisciplinary research.

Although the results of these interdisciplinary experiences still need to be published, it led us to construct projects with physicists, ethologists, biologists, herdsmen and dancers. I believe the most profitable outcome of Souk is this generation of data in interdisciplinary fields.

Nevertheless, Souk raised some interest inside the computer science community, and this part is based on the following papers:

- **Souk: Spatial Observation of hUman Kinetics** In *Computer Networks, Elsevier, 2016.* with



(a) Measuring the interactions of a herd and a dog during sheep-herd trials



(b) Art and Science project Stromo, an attempt to measure harmony

Figure 19: Different souk deployments

Marc-Olivier Killijian, Roberto Pasqua, Matthieu Roy and Christophe Zanon

- **Loca: A Location-Oblivious Co-location Attack in Crowds** In the *2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, Ubicomp 2016* with Roberto Pasqua and Matthieu Roy.

Contributions We introduce Souk, a platform to capture the behavior of a crowd at an unprecedented resolution. Traces captured using this platform allow access to both the position and the social contacts of individuals, allowing precise simulation of any Short Range Communication topology.

We showcase our approach by comparing standard mobility models against data collected during experimental deployments of the platform during multiple social events. Although the shortcomings of models such as the random waypoint model have already been pinpointed before, the comparison is only presented to demonstrate how the proposed approach allows to easily compare a given model against reality.

In the second chapter, we exploit Souk traces to show that co-location attacks can be performed without any information on the location of users or sniffing devices. More precisely, we show that it is indeed possible to perform location-oblivious co-location attacks, putting emphasis on the possible risks involved by the deployment of cheap and ubiquitous objects, as advocated by the envisioned Internet of Things. We provide an algorithm named Loca that implements this attack, identify the key parameters influencing the attack accuracy and evaluate our approach on different datasets. This evaluation shows that our approach is efficient on data captured using different technologies and on synthetic datasets. We also compare Loca with existing work.

Chapter 1

Souk and Loca: inferring social networks from position information

1.1 Context

During the past two decades, the problem of understanding human mobility has received a growing attention from the research community. Thanks to the widespread use of mobile handheld devices, large scale datasets have been produced and successfully exploited to provide a wealth of results characterizing human mobility patterns. Applications are multiple: from street planning to epidemics modeling, every hint about how humans move is a powerful ally for designing tomorrow's information society.

Indeed, one of the most crucial parameters of a ubiquitous system that relies on the network formed by users's devices —often referred to as Mobile Opportunistic Networking, or Mobile Networking in Proximity (MNP)— is its users contact model. Available mobility traces are usually coarse grained and do not allow to precisely emulate short range communication topologies. Contact traces exist, but these are usually established using wireless technologies themselves. The problem of such approaches is genericity: how to simulate a Bluetooth communication topology using an RFID contact trace, and vice-versa? Due to the wide variety of wireless technologies and their rapid evolution, it is of prime importance to establish datasets that are technology independent.

The idea that users' mobility and social ties (e.g. friends, foes, acquaintances) are connected has recently given rise to the development of mobility models taking these two dynamics in consideration. But in the absence of traces capturing both interactions and movements, such models remain only partially validated.

One of the fundamental question that is left unanswered is "*what is a good analytical model for crowd connectivity, required to implement the MNP paradigm?*", and, as a corollary, "*how to validate models?*".

To that end, we present Souk— Social Observation of hUman Kinetics. This platform allows the precise capture, in real time, of both the *position* and the *orientation* of individuals in a dense region. To achieve this, each individual is equipped with two lightweight wireless tags that are localized with a 15 cm accuracy using a network of sensors. More precisely, we present Souk as a means to test the realism of existing and future human mobility models. Moreover, by assessing directly positions independently of a given NMP communication protocol, the same dataset can be used reproduced communication networks produced by virtually any NMP communication technology.

The level of accuracy attained when capturing localization and orientation data allows Souk to infer interactions between users in a crowd. The dynamic network of social interactions arising during the social event can thus be computed, exposed in real-time and logged for off-line analysis. To the best of our knowledge, this is the first time that both social interactions and movements are assessed at such a fine granularity and at such a scale.

1.1.1 Typical Mobility and Interaction Analyses

The CRAWDAD project[1] regroups several mobility data collection campaigns. Table 1.1 presents a selection of datasets that include some flavor of mobility in captured data. These campaigns use Off-The-Shelf hardware, such as smartphones, to capture information, thus their localization source is either a GPS system or based on wireless interfaces.

Compared to the datasets we capture, the major difference lies in the scale: SOUK's dataset has a smaller scale (i.e., building-wise vs. town-wise, and short term vs. long-term) but provides a higher accuracy (i.e., in the order of 10cm vs. $10-100$m) and includes users orientation, thus enabling a precise capture of social interactions between users.

Since SOUK aims at capturing micro-mobility whereas all other datasets in Table 1.1 are interested in macro-mobility, these two types of datasets are complementary: on the one hand, understanding micro-mobility and fine-grained interactions between users requires the accuracy offered by a localization system similar to the one we used, and allows a better understanding of connectivity patterns within a pedestrian-carried mobile system. On the other hand, long-term evolution of systems and recurrent behaviors can only be captured on platforms that are large scale both in terms of time and space [12].

Dataset	**Users**	**Duration**	**Resolution**
Yellow cabs	100	1 month	GPS: 1/min
Reality Mining	100	9 months	BlueTooth: 12/h
UIUC-UIM	28	3 weeks	BlueTooth: 1/min, WiFi: 1/h
Nokia	200	1 year	GPS: user-defined frequency
Yonsei/Lifemap	8	2 months	GPS, WiFi $12-30$/h
SOUK	50	1.5h	Ultra-Wide Band: 10cm at 1Hz

Table 1.1: Typical available mobility/interaction datasets

Indoor positioning has been a very active area of research in ubiquitous and pervasive computing. Much effort has been spent on developing indoor localization technologies: from the original Cricket system [108] that used both radio and ultrasonic signals, to more recent systems using power lines [101], Ultra-Wide Band signals [6], digital cameras and SLAMs [74], CDMA mobile phone fingerprinting [130], resonant magnetic coupling [105], etc.

The study, and modeling, of the relationship between human mobility and social aspects of human behavior has recently gained a lot of attention. In particular, much effort is spent in developing socially inspired mobility or propagation models [63, 94, 23, 21, 100, 124]. In these works, positioning is not necessarily of primary interest but, rather, access to data concerning contacts or proximity between the individuals is necessary. Several technologies and methods have been used to collect or infer social contacts. Bluetooth and WiFi networks were used in various environments : in offices [81, 31], conferences [65], during a roller trip [16], on a campus [45] or in shopping malls [54]. The main limitation of these experiments lies in the fact that contacts are inferred when two devices are co-located or in communication range. Accuracy of this inference can be questioned. In the same manner, dead reckoning can be used to estimate relative positions between individuals [69] but has the same drawbacks. RFIDs may be used to record contacts when individuals are engaged in face-to-face interaction [124], without knowing users positions — notice that tags have to really face each other for a long enough period of time and thus, some interactions can be missed.

Recent works have focused on analyzing interactions in crowds [26, 25]. Yet, to the best of our knowledge, this is the first time that such an accurate and precise framework for capturing *both positions* and *contacts* is produced in the context of dense populations.

1.2 SOUK: the Experimental Platform

SOUK consists of three parts: an experimental platform to capture the position and orientation of mobile

[1]The CRAWDAD project: `crawdad.cs.dartmouth.edu`

individuals through wireless *tags*, a framework to develop mobility models, and a software system that exploits the output of either the capture process or model-generated traces. In a nutshell, both mobility models and the experimental platform can feed a database that is then accessed by the software pipeline. The use of a database between the production and the exploitation of positions ensures the repeatability of the experiments, and a certain degree of genericity: any model or positioning system can be used, for real-time exploitation of data or for later off-line analysis.

1.2.1 Position capture platform

The experimental platform relies on two types of elements: a sensor network, and a set of tags. Currently, SOUK relies on a Ultra Wide Band (*UWB*) based localization system developed by Ubisense [129]. The whole system can be deployed within a day, and the *UWB* technology allows SOUK to work seamlessly in crowded environments. The precise whereabouts concerning the hardware system are of little interest to this thesis: any system providing a similar localization accuracy would fit. However, to allow a general understanding of the process we hereafter briefly present the system.

Tags

A tag is localized by the system in a three dimensional space. Each participant is equipped with two tags, so that along with the position of the participant, we are able to compute the orientation of his body.

Each tag dimension is approximately $4 \times 4 \times 1.5\,\text{cm}$ and weighs 25 g, hence users tend to forget about them and the impact of wearing the tags on users behaviors is rather limited.

Tags communicate with sensors using UWB wave trains, and although these wave trains can traverse the human body to a certain extent, we chose to locate the tags on participants' shoulders to limit obstacles, as shown on Figure 20.

Moreover, since tag localization accuracy is about 15 cm (symbolized by the red halos on Figure 20), it is important to put sensors as far apart as possible from each other, in order to attain a sufficiently reliable orientation.

Sensors

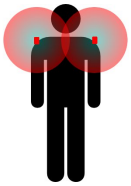


Figure 20: Each participant carries two lightweight tags, clamped on clothes on her/his shoulders.

Each tag periodically beacons a UWB pulse train. Sensors, which have fixed positions in the environment, locate the tag using both the angle of arrival and the time difference of arrival principles. Therefore, sensors need to be tightly synchronized together, and precisely positioned and oriented in space. The practical range of a sensor is a cone of 90° and 25 m depth. At least two sensors are required to provide an accurate position, but, in practice, the more sensors that compute the localization, the more precise and robust the position is (in our current setting, we use a set of 6 sensors).

Sensors use a Time Division Multiple Access (*TDMA*) channel access method. Indeed, both UWB and traditional WiFi signals are used to provide synchronization in the system. WiFi is used to negotiate the TDMA slots allocated to tags: every tag communicates using 802.11 signals with a *master* sensor that provides it with a unique TDMA slot. As soon as a tag gets allocated a TDMA slot, it starts to beacon pulse trains in its slot in the UWB range.

The number of TDMA slots per second is a fixed parameter given a set of sensors: let c be the number of communication slots per second, n the number of attendees in the monitored event (requiring $2 \times n$ tags), and f the position refresh frequency, in Hertz. All these parameters are bound by the relation: $f \leq \frac{c}{2n}$. In our setup, the highest rate available on the system is $c = 128$. As a consequence, to obtain a position refresh rate of 1 Hz, we limit to at most 64 participants. Notice that breaking this 64 participants barrier is just a matter of resources: duplication of sensors doubles the systems capacity; more generally, the *cost of the system is linear* with respect to the number of equipped participants. As of 2016, the price of the position capturing system for n users is roughly $\lceil n/64 \rceil \times 20\text{k\$}$.

1.2.2 Mobility model framework

When devising a mobility model for humans, it is fundamental to question its accuracy. To that aim, we provide researchers with a large generic toolbox of metrics that can characterize both models and experiments. SOUK allows using this toolbox regardless of the model-generated or experimental nature of the data, allowing a fair comparison between models and experiments.

In the framework, it is simple to develop a mobility model to use it as an input of the analysis process. It is thus possible to check generated traces against real, captured ones. More interestingly, it is possible to use model-generated traces as an input of the analysis pipeline, and make comparisons on higher level metrics, as we show in the next subsection, rather on low-level traces that cannot be easily compared.

Practically, developing a new mobility model is as simple as extending the provided random way-point model.

1.2.3 Analysis pipeline

This part constitutes the heart of SOUK. It consists of a set of software bricks that dynamically retrieve data from the database, filter results, infer contacts, and maintain various statistics. Figure 21 illustrates the information processing pipeline used in SOUK. The layered approach used allows partitioning the analysis in different bricks that are detailed below.

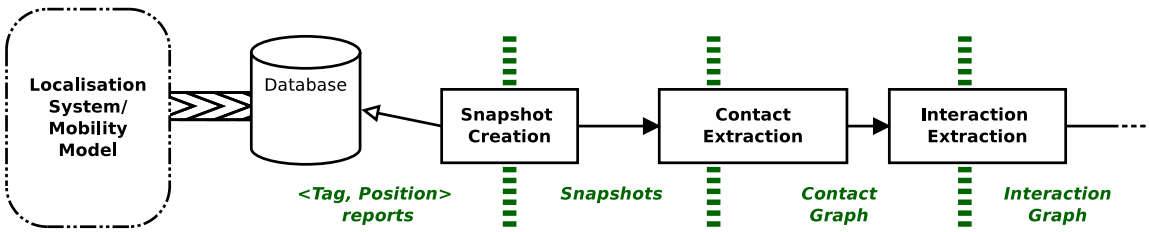


Figure 21: Overview of the SOUK processing pipeline

Snapshot creation

The localization system produces a flow of tag positions (more precisely, `(tagId, x, y, z, timestamp)` tuples) that need to be transformed into participants position/orientation couples. This is the role of the snapshot creation module, that periodically polls the database for new data, associates sensor pairs to each participant, and computes participants orientation.

Snapshot is an important abstraction in SOUK. Intuitively, it provides a series of "pictures" of the system as if it were periodically frozen. Such a discretization of system states dramatically simplifies data processing.

An important feature of this mechanism is to handle possible missed position reports. Indeed, due to the high contention on the wireless medium, or to possible obstructions, some position reports might be missed. As a result, one of the two tags locating a participant may have no position known at a given round. The snapshot creation module implements a synchronization and filtering policy to handle such cases: periodically, when a snapshot is created, the last known position is used for every tag, mitigated by a classical aging policy — when a tag has no position known for Δ time units, it is considered to be out of the system. From our first experiments, we found that setting $\Delta = 3$s provides good filtering properties while still guaranteeing sufficient temporal and spatial accuracy.

Contact extraction

Every snapshot is then transferred to the contact extraction module. Its role is to decide whether two given participants (or the devices they carry) a and b are potentially engaged in a contact. This decision is based on the analysis of participants respective positions and orientations, and depends on the nature of the contact that one seeks to capture, and therefore on the finality of the study.

In this paper, we present two different contract extractions, corresponding to two different use cases for SOUK. One extracts the communication topology of *devices* carried by users and communicating wirelessly at short range. The other extracts social contact between *users*. Although defining what is a social contact is far from being trivial —several definitions can coexist— SOUK modular architecture allows to implement and test heuristics corresponding to those different definitions. In the next section, we showcase different techniques: a device-based unit-disc graph wireless contact extraction, and a two user-based social contact extraction techniques.

Interaction extraction

Once contacts have been extracted, they are exploited by the interaction module to decide whether a and b are engaged in an interaction. The interaction module keeps an history of the previous contact graphs, and uses information filtering techniques to transform contact graphs into an interaction graph.

This step is very important for two reasons. First, recall that information about tags positioning is noisy, and therefore contact graphs might be biased. Therefore this information filtering step allows to remove spurious interactions caused, for instance, by inaccurate participant orientation. Second, a contact is not necessarily the beginning of an interaction: two people crossing each other because they travel opposite directions in the same corridor will appear for a short period of time as being in contact, but this scenario cannot be considered as an interaction.

Finally, each pipeline step can be decorated with observers that perform additional functions. We used this feature to export topologies to a database and to the ns-2 simulator, to stream the graph to a graph rendering software in real-time for visualization purposes, and to compute statistics.

1.3 Experimental Deployments

1.3.1 Experimental setup

We present here three measurement studies, named `souk`, `milano` and `cap2`. All experiments were conducted in roughly the same context: a social event where attendees are free to move and socialize. `Souk` and `cap2` were conducted in Toulouse while `milano` was conducted in Italy. The first deployment, `souk`, was conducted during the social event consecutive to the inauguration of a new building in the LAAS laboratory. The attendance mixed lab members, officials and representatives of the local and national science communities. The second deployment, `milano`, took place after a science and art project in which SOUK was used to measure the movements of a group of amateur dancers. We recorded participants positions during the closing cocktail. Similarly in `Cap2`, the recording took place during a closing cocktail after a measurement study where SOUK was deployed for an experimental project with biologists.

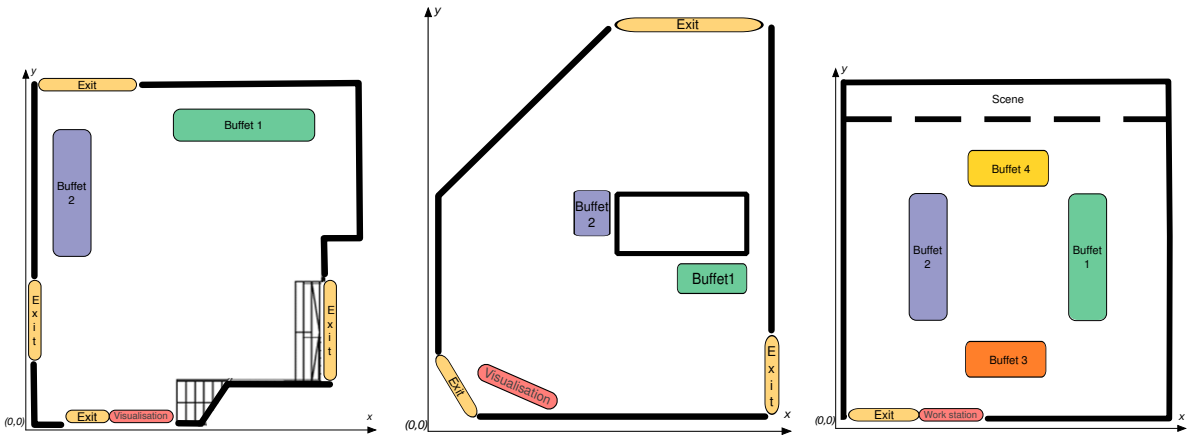
Table 1.2 reports on the physical setup of the campaigns.

Date	Name	Experimental room size (m^2)	Maximal number of participants	Trace length (minutes)
05-07-2012	souk	110	45	44
03-05-2015	milano	225	64	117
18-09-2015	cap2	100	56	101

Table 1.2: General statistics regarding the reported deployments.

The physical layout of the rooms is presented Figure) 22.

In the 3 deployments, we equipped the vast majority of participants. However, the open nature of these events prevented us from ensuring that 100% of the social event attendees had tags. In `souk` one participant refused to equip for privacy reasons. Moreover, the number of users in the dataset also declines through time as attendees leave the cocktail. Figure 23 plots the evolution of the number of captured users over time.



(a) `Souk` room map (b) `Milano` room map (c) `Cap2` room map

Figure 22: Layout of the experimental areas

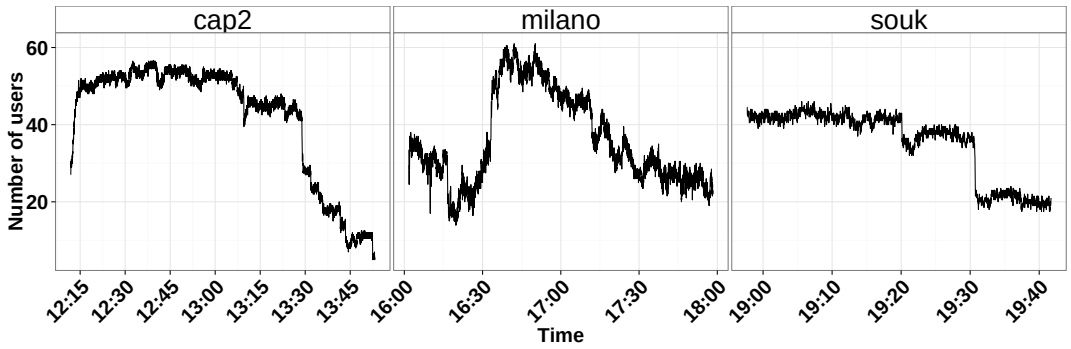


Figure 23: Number of users in each trace.

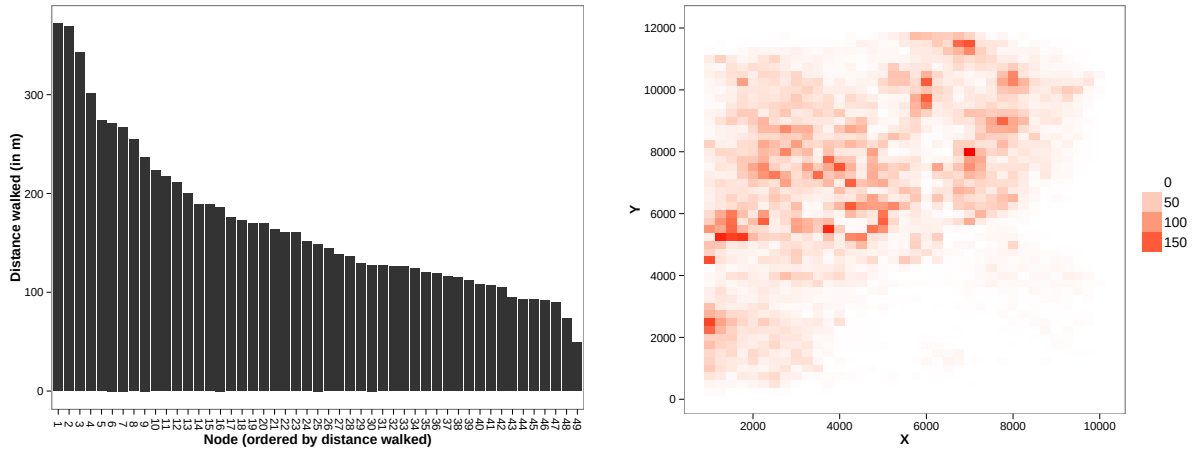
1.3.2 General characteristics

While the goal of this section is not to get in details through each of the datasets, we here reproduce some notable characteristics. First, all datasets exhibited very irregular movement patterns. For instance, the density of reports is not homogeneous in the experimental area as attendees like to flock and to group around points of interest such as bars. Attendees also exhibited a wide range of total distance walked. Figure 24 plots those distributions for the `souk` deployments.

Our framework also enables to monitor attendee's movement patterns through time. Figure 25 plots some results: since it is possible to detect attendee's immobility with a good accuracy, we can partition their mobility pattern in 3 broad classes (moving, stop, out –that is, they are outside, out of range of the system). One can observe a median walking speed of $0.5m.s^{-1}$ ($1.8km.h^{-1}$), a fairly low value that however fits the idea of range at which and individual can evole in a crowded place. The `milano` deployment has a relatively lower density, which might explain the large fract of speeds above $0.8m.s^{-1}$.

1.3.3 Constructing Interaction Graphs

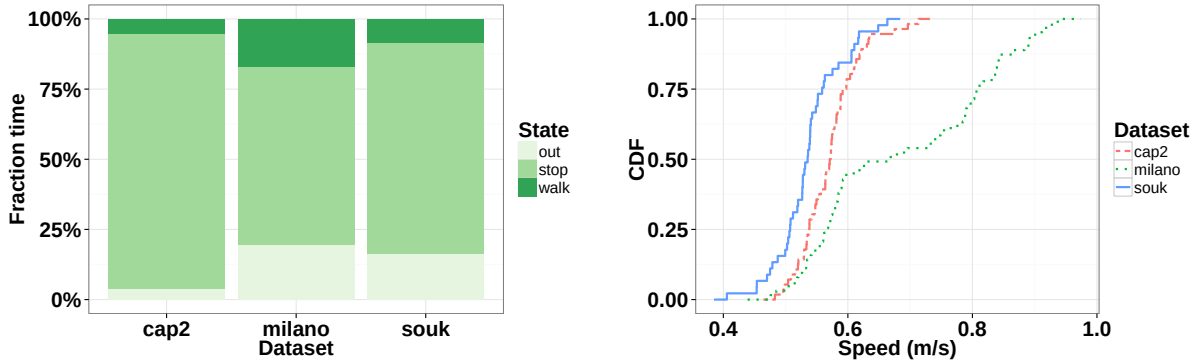
We here explore three simple models of what a contact is. Defining social contacts is far beyond the scope of this thesis, but we provide these examples in order to illustrate the potential uses of the platform. Thanks to the raw nature of the data – that is, it contains positions rather than contacts directly– incorporating a new contact model to Souk is a matter of minutes, the only limitation being that such a contact model has to be solely based on position and orientation of individuals.



(a) Distance walked by each attendee

(b) Number of position reports for each area

Figure 24: Movement patterns for the `souk` dataset.



(a) Fraction of time attendees are walking/stopped/out

(b) Cumulative Distribution Function of walking speeds

Figure 25: Temporal movement characteristics

The first two models are based on the simple distance between the users. More precisely, we consider two individuals are in contact iff the euclidean distance between them is less than x meters. This contact model mimics the traditional Unit Disk Graph model well known to the distributed sensor network community (see e.g. [76]). A use case for this model is to model the communication network of user carried devices. In the following we consider contact ranges of 1 and 2 meters.

The third model carries a sociological dimension. Based on an agglomerative clustering, it tries to identify clusters of individuals with the objective to capture social groups gathered to exchange. In a nutshell, the algorithm works as follows: each individual initially belongs to his own cluster. At each step, the algorithm merges the two clusters that are closest to each other, until the minimal distance between two clusters reaches a defined distance. The maximal distance we considered is $1.5m$ between two clusters, where the distance between two clusters is defined as the euclidean distance between the barycenters of each cluster. All the individuals belonging to the same cluster are then considered in contact with this model named *face-to-face* (ftf) hereafter.

To construct our graphs, we first construct snapshots of individuals positions at each second of each dataset. We then transform each of these snapshots in a graph using the desired link creation model, yielding for each dataset a dynamic graph $\{G_t\}_{0<t<end}$.

1.4 Reality Against Models

Thanks to the collected experimental data, we can now inspect dynamic interactions graphs. We here only illustrate one potential usecases of such data: we will focus on comparing the constructed "real" interaction graphs with those generated using state-of-the-art mobility models.

To achieve this, we relied on André Panisson's Pymobility framework[2] that can generate mobility traces from 7 different mobility models. To ensure a fair comparison, we parametrized each of these models as close as possible to our traces (same number of nodes, room size, flight lenght, pause time, etc.).

Obviously, the generated networks will be different, but how to measure such differences ? One approach would be, like in Chapter 1, to compare the distributions of synthetic graph properties like diameter, connected components, and so on.

However, since the orginal objective of these works was to enable novel MNP approaches, we here take the MNP application developer viewpoint. If this developer relies on models to simulate its application deployment, a central question he faces is: "will my application behave the same on synthetic traces and in reality" ? While it is difficult to test all possible applications, we chose to implement a central primitive of MNP (and, in a larger extent, distributed) system: the broadcast time.

We argue this broadcast time is a good proxy to capture the efficiency of information diffusion in those networks. More precisely, we measure the time needed for a node to send a (multi-hop) network to reach a majority (i.e. $\lfloor n/2 \rfloor + 1$) of other nodes for two reasons:

- It is the time needed to reach a *standard* quorum, on which many distributed agreement problems are based
- As some attendees can get out, measuring the time needed to reach the hole population (n) is noisy: the absence of a single node will delay the completion of all broadcast, and the resulting measure is not stable enough to bring information.

The implemented broadcast algorithm is a simple flooding: for each new encounter, send all previously received messages. Since we actually only simulate such flooding, the classical problems of collisions and saturation of the network can easily be avoided. In practice, a more subtle algorithm should be implemented (like Lightweight Probabilistic Broadcast [47]). However, flooding provides a lower bound on the time (as any congestion control mechanism or any collision would only increase the broadcast time).

Figure 26 summarises the obtained results. In short, all models systematically underestimate the time needed to reach half of the population. While the difference is small with a $2m$ range because the diameter of the communication graph is low anyway, this difference is considerable in the two other models where the median time is sometimes twice lower in the synthetic traces compared to the collected data. Moreover, it is also much more uniform in the models.

Why such a difference ? Figure 27a provides hints about the answer. It represents for the `souk` deployment the weight distribution of the cumulated interaction graph. That is, $\forall i,j \in V^2, w_{ij} = \sum_t 1_{V_i^t}(j)$, where $1_{V_i^t}(j) = 1$ iff $(i,j) \in E^t$. This not surprisingly skewed distribution in the `souk` case translates a known behaviour: we are not uniformly random in the time we spend with other, we spend much time with few people, and few or no time with many people. For comparison, the same weight distribution is represented for the random waypoint (`rwp`) model: it exhibits a standard binomial distribution, translating the fact that our random agents have no social preferences and "spend time" with anyone with the same probability.

It is possible to imagine the impact of these differences on a broadcast time: since random agents meet equally likely, they quickly propagate the info. On the other hand, the social preferences of real individuals lead them to spend most of their time with the same few individuals, from which they do not learn as much. This effect is further reinforced by the strong clustering effect of real networks (the friends of my friends are my friends): Figure 27b presents a layout of the weighted graph; colors correspond to communities detected using the standard Louvain Method [20]. This relatively strong community

[2] `https://github.com/panisson/pymobility`

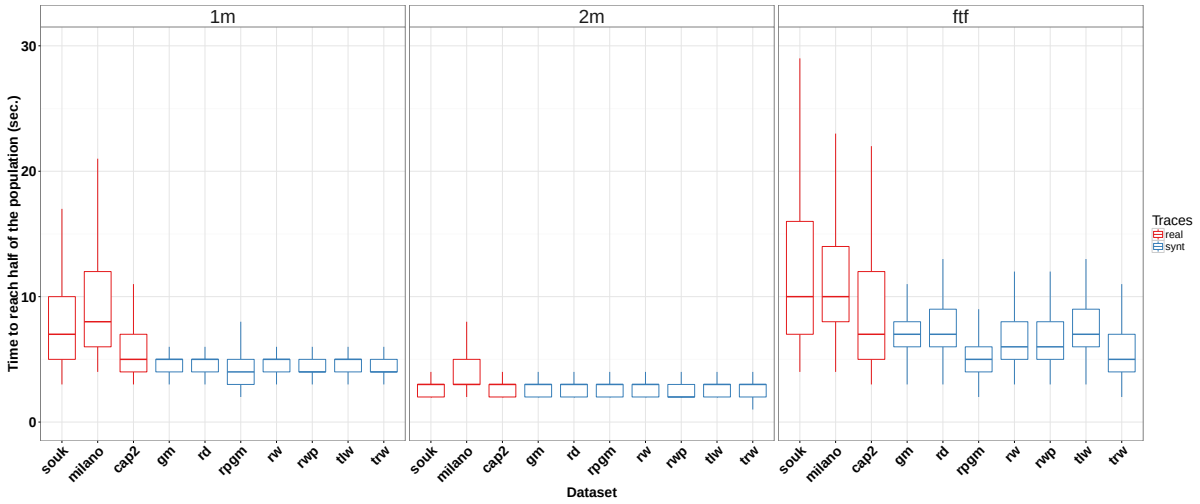
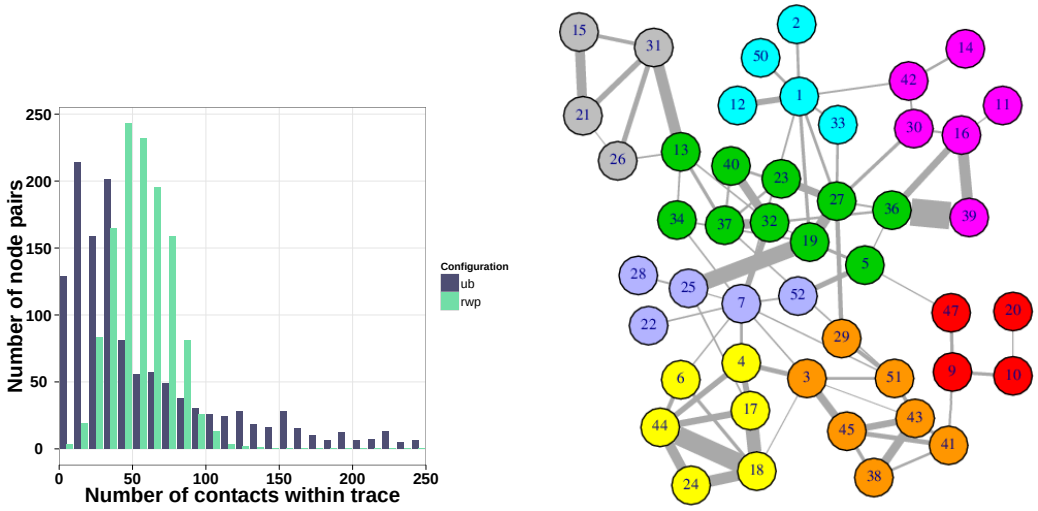


Figure 26: Broadcast time on datasets and synthetic traces.

structure of the interaction network allows a broadcasts to quickly spread within a community, but extends the time needed to reach the majority of nodes.



(a) Weight distributions of the wireless contact graphs generated from the (user-based, ub) collected data, and from the (random waypoint, rwp) model.

(b) Layout of the final interaction graph.

Figure 27: Digging into social interactions

1.5 Conclusion

In this chapter, we introduced the Souk experimental platform, designed to capture the position of a crowd in order to infer its connectivity. As a validation process, we then compared our results with state of the art mobility models, both from an application perspective (based on the broadcast primitive), and from the infered graph properties.

The conclusion we can draw from this comparison of Souk against a palette of mobility models is that they fail to produce realistic graphs at least from an information diffusion perspective. Interestingly, this

process fails despite the fact that mobility models and collected data both generate geometric graphs that all share the desired properties (like sparsity and clustering) listed in the introduction of this part (section III). While the limitations of mobility models are well known, such failure brings to the conclusion that mobility models fail to capture a part of reality that plays an important role in the establishment of those contact graphs.

The last observations led us to suspect the social preferences as such missing element: somehow, mobility traces embed social preferences while most models ignore this dimension. The next chapter will focus on exploiting this relation between social preferences and position information. More precisely it focuses on exploiting mobility traces to extract social preferences, a problem that currently finds applications in privacy.

Chapter 2

Loca: Inferring Social Networks from Sparse Traces

2.1 Context

The massive diffusion of GPS-equipped smart phones and the tremendous growth of the data market raised concerns about the privacy of users. The balance between the benefits in terms of service, and the drawbacks in terms of exposed private information is now better understood. To achieve this understanding, a decisive step was to assess the amount of private information that was enclosed in mobility traces of users. We know that the location of individuals can reveal much about their activities, lifestyle, job satisfaction [44] or even social contacts. As such, the recognition that mobility traces represent sensitive information has widened.

In this context, co-location attacks have demonstrated the danger of implicitly revealing social networks by exploiting the geographical proximity of located users and translating them in terms of "social proximity" [40]. The work carried out in [40] exploits a set of GPS-geo-tagged pictures. It divides Earth in a roughly 80km sided grid and correlates users' co-locations on this grid. Due to the huge time and space span of the dataset, accidental co-locations are extremely rare, and users being repeatedly co-located are often socially tied, as they explain: "two people have almost a 60% chance —nearly 5,000 times the baseline probability— of having a social tie on Flickr when they have five co-occurrences at a temporal range of a day in distinct cells".

Their work highlights the following principle: if users can be repeatedly located (either directly using a GPS device, or indirectly by revealing their proximity to a located device), an attacker can isolate recurring physical proximities between users, and derive localized social contacts, that are in turn correlated to general social interactions. Hence, to be effective, the attacker needs to have access to precise location information of equipment, be they user-carried or fixed.

The requirement of a good location accuracy in order to infer social ties as been relaxed in [19] where the authors assume an attacker able to trilaterate mobile phone users. To that end, they assume the attacker knows the position of 37 Access Points (APs) in a Wifi mesh over a $130\text{m} \times 250\text{m}$ campus region. They also assume the attacker is able to sniff some of the packets exchanged between the APs. This allows them to trilaterate mobile users based on APs positions and the sniffed RSSI values (used as a proxy for AP-user distance). Then, they exploit the obtained positions using a probabilistic framework to infer relationships among the targeted users. Finally, they swiftly exploit the specificities of the academic structure (*i.e.* students divided in classes) to improve the attack results using a community detection algorithm.

Thanks to the awareness raised by the location-privacy research thread, location is now generally considered as sensitive information. The corresponding attack surface is shrinking as users are more aware of the threat and take actions to hide their location information. Yet, the expected massive deployment of small connected objects that are not located due to cost reasons gives attackers access to huge amount of data collected on users possibly without their consent. Consider the following two

scenarios:

Sniffing Smart Dust UHF RFID readers can read passive tags at a range of 1m [95] while cheap micro-controllers with Bluetooth Low Energy client capabilities and low power consumption cost around $30 (*e.g.* RFduino). An attacker can deploy many such sensors for a relatively low price, and this deployment can be done quickly as no positioning of the sensor is required. Such sensors have a very low power consumption and need only to be fitted with an RTC clock and some storage capacity to log the encountered MAC addresses.

Network-based Eavesdropper An attacker spies on the network activity generated by users' smart phones reacting to the proximity of IoT objects such as iBeacons from Apple, Eddystone from Google, BLE tags, or reactive advertising platforms.

Consider for instance iBeacons scattered in the environment (say, a shop) that interact with users' smart phones to provide users with an enriched experience [66]. This is typically done by "binding" a tag to some digital content on the cloud. Therefore, these interactions typically involve a network fetch from the smart phone of the user (e.g., a web page, a price). The tuple (smart phone IP, tag url, time-stamp) can be monitored at various levels of the network infrastructure.

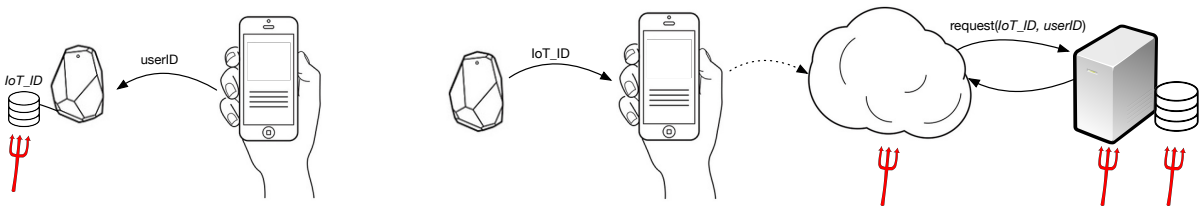
These scenarios show that a huge number of day-to-day activities of users, including users' colocations, are or will soon be available to potential attackers. Although such personal information on users does not include location data, the main threat of these scenarios is the impossibility for users to even notice the data collection, hence to take action against this collection, and the ubiquitous nature of such data collection campaigns.

In this chapter, we go beyond location-based co-location attacks by providing a method to interpolate social interactions from a co-location attack performed on location-oblivious activity traces gathered in one of the scenarios exemplified above. Removing the dependence on the knowledge of location to conduct co-location attacks clearly extends their threat and scope. Indeed, this implies that cheap *IoT* objects represent a potential support of such attacks. Moreover, such an attack can be conducted without knowing the geographical layout of the targeted area, or even knowing where the targeted area is: it can be based on monitoring co-locations, independently of where they happen, provided that co-occurrences of users detected by a device represent a real co-location of users.

The principle of this attack is as follows: we assume we are able to collect the proximity of each user to a fixed set of K sensors for which we do not know any information apart from that they have unique identifiers. We use proximity data of users in two ways: first we exploit the recorded proximities of all users as a whole, and use them to construct a virtual map of the sensed space. Then we exploit "trajectories" of users in this virtual space to conduct a standard co-location attack. Through simulation on both real and synthetic datasets, we show that in dense enough environments, such a virtual cartography is accurate enough to reveal social contacts of users and preferences.

2.2 Model and Algorithms

2.2.1 Attacker Model



(a) IoT device under control

(b) Compromised infrastructure or service provider

Figure 28: Possible attack surfaces represented by red arrows.

Algorithm 3 ComputeSimilarityMatrixhist$_k$

Require K sensor logs hist$_k$ /*For brevity, the case of overlapping sensors/virtual sensors creation is omitted*/

`int` $P[1\dots N,1\dots T]=\{\{0\}\}$ /*Proximity logs for users, initially null*/
`int` $R[1\dots K,1\dots K]=\{\{0\}\}$ /*Transition matrix on sensors, initially null*/
`int` $M[1\dots N,1\dots N]=\{\{0\}\}$ /*Interaction matrix between users, initially null*/

for $t=1\dots T$ /*_First step_: compute proximity logs*/ **do**
  for $k=1\dots K$ **do**
    for all $i\in \text{hist}_k(t)$ **do**
      $P[i,t]\leftarrow k$

for $i=1\dots N$ /*_Second step_: compute transitions*/ **do**
  for $t=1\dots T-1$ **do**
    if $P[i,t]\neq 0$ /*User i is detected at time t:*/ **then**
      if $\{t'>t, P[i,t']\neq 0\}\neq\emptyset$ /*If user i is detected after t,*/ **then**
        let $t_i=\min\{t'>t, P[i,t']\neq 0\}$
        $R[P[i,t],P[i,t_i]]\leftarrow R[P[i,t],P[i,t_i]]+1$ /*then update transition matrix accordingly.*/
for $k,k'=1\dots K$ **do**
  $R[k,k']\leftarrow R[k,k']/\sum_{t\in T}|\text{hist}_k(t)|$ /*Normalize transitions*/

for $i=1\dots N$ **do**
  for $j=i+1\dots N$ **do**
    $score\leftarrow 0$
    for $t=1\dots T$ **do**
      $score\leftarrow score+R[P[i,t]],P[j,t]]$ /*_Third step_: compute interactions based on similarity*/
    $M[i,j]\leftarrow score$
    $M[j,i]\leftarrow score$
Return M /*Output interaction matrix*/

We assume that an attacker has access to a set of *proximity logs*, captured by wireless equipped objects. More precisely, information available to the attacker has the following properties:

- Every user sensed by the objects has a unique identifier, e.g., a physical address or a cookie,
- Proximity logs contain timestamped detection of users' identifiers.

We do not assume a particular model for the attacker to get such information. As depicted in Figure 28, proximity logs may come from the objects themselves, if the attacker has physical access to IoT objects. It may also be eavesdropped in the network infrastructure, if the attacker controls a set of networks links. Additionally, proximity logs may come from the IoT service provider if the attacker is an insider —or if the service provider has been compromised.

From these logs, the attacker computes the number of users N, and the time span of the attack T, transforming proximity logs into a sequence snapshots for all time instants $t\in\{1,\dots,T\}$. We consider that the attacker has K distinct proximity logs, coming from *immobile* sensors deployed in a given area. Each sensor is unaware of its position and is able to detect any subset of the N users that pass in its vicinity at each discrete instant $t\in\{1,\dots,T\}$. Consequently, the total amount of information that is available is a set of K logs $(\mathsf{hist}_k)_{k=1..K}$ output by sensors, each one providing a set of detected users at every time instant $(\mathsf{hist}_k:[1,T]\rightarrow 2^{[1,N]})$.

2.2.2 Transformation Algorithms

In order to exploit low-level information hist_k, LOCA proceeds in three steps: 1) we compute aggregated proximity logs for each user, that provide the sequence of encountered sensors for each user during an experiment, 2) we construct a transition matrix that evaluates the flow of users between sensors

during the experiment, and 3) we compute an interaction matrix that quantifies interactions between pairs of users, i.e., we generate the co-location events for each pair of users in the dataset.

Physical sensors: proximity logs The following definitions formalize these steps, and the LOCA algorithm is described in Algorithm 3.

Definition 2.2.1 (Proximity Log)**.** Given a set of histories hist_k, the proximity log p_i for user i ($i \in 1 \dots N$) is defined as:

$$\begin{array}{rl} p_i: & [1,T] \to [1,K] \\ t & \mapsto \left\{ \begin{array}{ll} k & if\ i \in \mathsf{hist}_k(t) \\ 0 & otherwise \end{array} \right. \end{array}$$

In this definition, we assume that sensors have non-overlapping range of detection. Such an assumption greatly simplifies presentation of later steps of the transformation algorithm.

Virtual sensors Yet, in real systems, assuming that sensors shall not overlap is not realistic and too restrictive. As we will see later, our real attack algorithm does take into account sensors overlapping by means of *virtual sensors*. When two sensors overlap effectively, i.e. when two (or more) sensors detect the same user simultaneously, we assign this detection to a virtual sensor defined as the intersection of sensors that detected a user simultaneously. Notice that such a virtual sensors creation scheme is oblivious to the position of sensors, and is solely based on simultaneous detection of a user by different physical sensors.

Transition matrix: users flows Starting from proximity logs of users, we can compute the *Transition Matrix* R that evaluates the transition of users between different sensors. Its elements $r_{k,k'}$ measure the flow of users that move from sensor k to sensor k'.

Definition 2.2.2 (Transition Matrix)**.** For every $(k,k') \in [1,K]^2$, the element $r_{k,k'}$ of the Transition Matrix R is computed as follows:

$$r_{k,k'} = \frac{1}{\sigma_k} \sum_{i \in [1,N]} |\{(i,t,t') \text{ s.t. } p_i(t) = k \wedge p_i(t') = k' \wedge p_i(t'') = 0, \forall t'' \in]t,t'[\}|$$

with $\sigma_k = \sum_{i \in [1,N], t \in [1,T]} \delta_{k,p_i(t)}$, where δ is the Kronecker delta: $\delta_{i,j} = 1$ iff $i = j$, 0 otherwise.

Due to physical constraints, the Transition Matrix is a non symmetric sparse matrix that contains an indirect estimation of real sensors distance, corresponding to "virtual distance".

Users' similarity score To quantify interactions between two users and evaluate their co-location profile, we compute the similarity between their trajectories. We define the similarity index as

$$\forall (i,j) \in [1,N]^2, score(i,j) = \sum_{t \in [1,T]} r_{p_i(t),p_j(t)}.$$

The definition above shows that similarity index is based on the virtual distance between sensors defined through the Transition Matrix.

Interaction matrix: co-locations Finally, in order to study interactions between each pair of users, we define the *Interaction Matrix* M, a symmetric matrix that measures how two given users interact. This matrix is the result of the co-location inference attack and contains the whole amount of co-location events detected during the experiment.

Definition 2.2.3 (Interaction Matrix)**.** Element $m_{i,j}$ of the Interaction Matrix M is defined as:

$$m_{i,j} = m_{j,i} = \left\{ \begin{array}{ll} score\,(i,j) & \text{if } i \neq j. \\ 0 & \text{if } i = j. \end{array} \right. \tag{2.1}$$

Algorithm 3 describes the complete code used to compute the interaction matrix.

To sum up, LOCA exploits proximity logs in two complementary ways. First, to compute similarity between users, in a quite classical way. Second, to establish a virtual "map" between sensors: as we will see in the next Section, this map is an estimation of sensors distance that is sufficient to conduct an efficient colocation attack.

2.3 Experimental Results

In this section, we present simulation results quantifying the accuracy and sensitivity of LOCA, as presented in Section 2.2.2.

2.3.1 Datasets

The LOCA algorithm has been tested on three real-world datasets and a set of synthetic ones, described hereafter. Each dataset consists of two parts: a) a set of users location traces and b) a ground truth representing for the real number of social interactions of each pair of users. The difficulty of acquiring datasets that contain both position information and social contacts severely constrained the number of datasets that could be exploited.

	Space $[m]$		Nodes			Step
Name	Width	Length	N	v_{min}	v_{max}	T
HD	10	10	40	0.01 m/s	0.5 m/s	1000
LD	30	30	40	0.01 m/s	0.5 m/s	1000

Table 2.1: Input parameters.

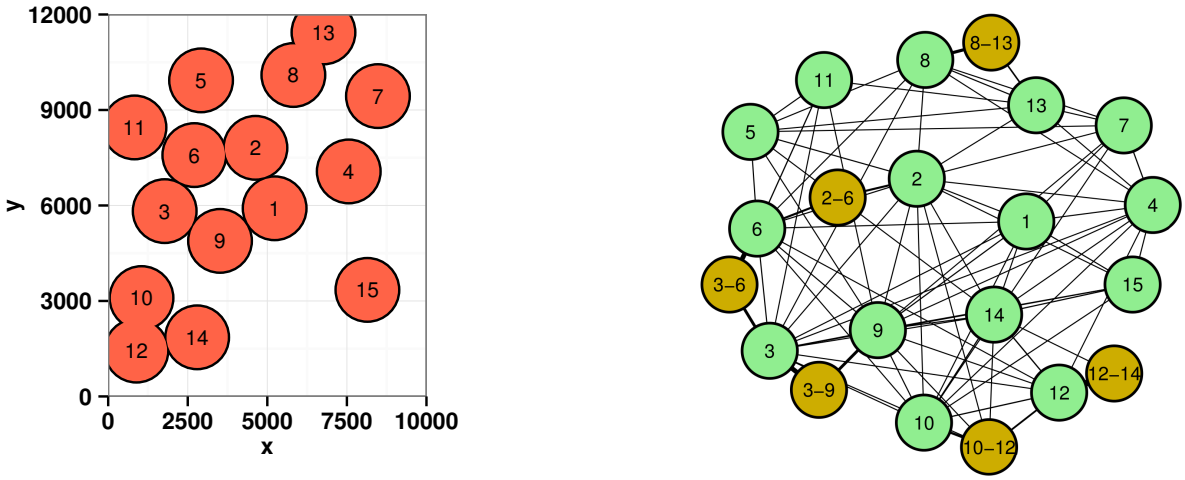
Synthetic We first test the accuracy of the inference on a broad range of synthetic datasets generated thanks to the `pymobility` generator [10]. Table 2.1 contains the settings used to configure the different models to generate the datasets. For mobility models featuring a varying node speed, we chose $0.01m.s^{-1}$ as the minimal speed (some models require a strictly positive value), and $0.5m.s^{-1}$ as the maximum speed (picked according to our measurements). We configure `pymobility` to simulate two types of gatherings, one corresponding to a *High Density* situation (hereafter, HD) with 40 individuals in a 100m^2 area, and one representing a *Low Density* situation (hereafter, LD) with again 40 individuals in a 900m^2 area. For both gatherings, we generate 10 independent mobility traces of 1000 seconds using Random Walk (rw), Random Waypoint (rwp), Random Direction (rd), Truncated Levy Walk (tlw), Gauss-Markov (gm), Reference Point Group Mobility model (rpgm), Time-variant Community (trw). For each of these traces, we also generate the corresponding contact traces using a distance-based interaction model as our ground truth, adapted from [72].

Real/SOUK The SOUK data set contains precise trajectories of 45 users during a 45 minutes cocktail party [71], captured using ultra-wide band real-time location technology in a 100m^2 (10m $\times$ 10m) closed space. The ground truth consists of detected social interactions using the algorithm provided in [71]. Both dataset and algorithms can be freely downloaded [113].

Real/MILANO The MILANO data set is similar to the SOUK dataset. It was captured using a similar technology and contains trajectories of 64 users during a 45 minutes cocktail party in a 225m^2 (15m $\times$ 15m) closed space. The ground truth consists of detected social interactions using the algorithm provided in [71, 113].

Real/Badges The Badge dataset [43] contains trajectories of 39 employees equipped with active sociometric badges in an office environment during approximately one month. The considered ground truth consists of the cumulated face-to-face interaction time as recorded by the RFID badges. Even though in total 39 employees were equipped with badges, not all employees were present everyday. We only retained days where strictly more than 20 employees were present.

2.3.2 Metrics and Methods



(a) Physical sensors deployment: Density

(b) Computed virtual map —virtual sensors indicated in yellow are labelled with intersecting physical sensors label set.

Figure 29: (a) Deployment of 15 sensors in SOUK cocktail-room according to the Density strategy. (b) Representation of the *Transition Matrix* as a weighted graph laid out using the Fruchterman and Reingold algorithm.

For each dataset, we virtually "deployed" a set of K co-location sensors in the experimental space. The set of K proximity traces extracted from each sensor is then the only input of the inference algorithm: we exploit the sensor traces without relying at any point on their location. To measure the accuracy of the attack, we compare the results obtained from the attack represented by an interaction matrix M with the ground truth matrix G obtained from the original datasets. The ground truth matrix is similar to the interaction matrix M in the sense that it contains the real interaction behavior for each pair of users.

To measure the accuracy of the inference, we rely on two metrics:

1. **Global Social Network Inference:** We measure the accuracy at which the attack identifies the strongest social ties of the ground truth social network G. To do so, we only target the K_{in} pairs of G having the highest number of interactions.

   We use Receiver Operating Characteristic (hereafter, ROC curves) analysis to compare different models by representing the evolution of the true positive rate (in our case: the predicted user pair is indeed one of the K_{in} strongest social ties) as a function of the false positive rate (in our case: the predicted user pair is not one of the K_{in} strongest ties).

   The Area Under the Receiver Operating Characteristic curve (or Area Under Curve, hereafter denoted as AUC) summarizes the obtained inference results. It can be interpreted as the probability of LOCA to assign a higher score to a randomly chosen user pair in K_{in} than to a randomly chosen user pair not in K_{in}. This methodology is commonly used in classification analysis to determine which of the studied models has the best prediction capabilities.

We typically set $\mathsf{K_{in}} = 3N$ to obtain a connected social network with an average degree equal to 3. Yet, due to the different levels of social activity of the nodes, we often observe a huge bias towards more social nodes.

2. **Local Best Friend Inference:** For each node $i \in [1, N]$, we measure the size of the intersection between lists of i's 10 best friends (i.e. strongest social ties) in G and in M: this is the recall@10 for each user. This is a tougher test for LOCA inference since we remove the bias toward more social individuals to also target weakly socialized users for which data is mechanically sparser.

To provide a comparison baseline, we also assessed the accuracy of the co-location attack presented in [19]. As this work is, to the best of our knowledge, the closest to our proposal to date, we implemented their approach to compare it against our algorithm. In the following, we refer to this approach as the State of the Art (SoA), obtained using the full trajectories. The gap between SoA and LOCA inference accuracy therefore measures the added cost of having no position information.

2.3.3 Sensors Deployment Strategies

Two parameters greatly impact the inference accuracy: the number of sensors K, and their range. These two parameters indirectly define the area covered by sensors. Intuitively, and as experiments will confirm, the higher the coverage achieved by sensors, the more accurate the attack is.

But another important parameter influencing the attack success is the density of users: sensors should be placed in dense areas in order to gather as many records as possible. Yet, adversary's control over this parameter is limited as he might have no control over sensors placement, or might not know a priori where users will densely regroup. We therefore study the following strategies:

- *Grid*: sensors are deployed along a regular grid starting from the center of the area.
- *Density*: sensors are placed using a priori knowledge of high-density areas (i.e. where people will gather and where social interactions are more likely to happen). This deployment represents the best possible situation and provides an upper bound of the sensor placement impact. Such information could have been obtained through observation of previous similar gatherings or exterior knowledge (e.g. metro platforms rather that metro corridors), or by targeting Points of Interest.
- *Spiral*: sensors are placed starting from the center of the area, following a counter-clockwise spiral extending up to the periphery of the area.
- *Random*: in this strategy, sensors are randomly located in the experimental area.

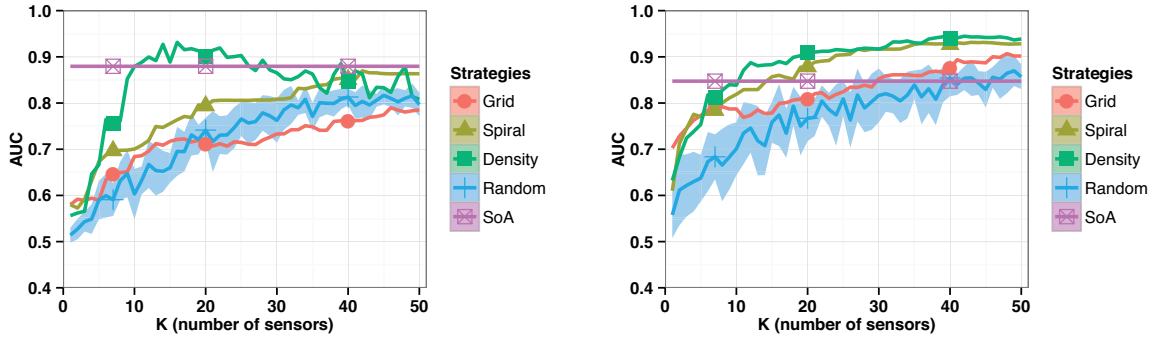
Note that the impact of poor sensor placement on LOCA is twofold: first, badly located sensors will provide very few co-locations, hindering the score computation, and second badly located sensors will degrade the relation between the virtual distances matrix of sensors (the Transition matrix) and the actual physical distances of sensors.

The last parameter conditioning the covered area is the sensors detection radius. A 100 cm range could for instance represent the immediate range of an iBeacon device or a very conservative read range for an UHF RFID passive tag reader [95], whereas BLE range can be set by the programmer for up to 50 cm in practical contexts. The impact of range is detailed in Section 2.3.5.

2.3.4 Quality of the Virtual Mapping

At the heart of our approach is the exploitation of users' sensor to sensor flows to estimate the distance between sensors through the construction of the Transition Matrix. Implicitly, we assume that the number of individuals going from one sensor to another is related to the distance between sensors.

Figure 29 illustrates the correlation between the real positions of sensors (Figure 29(a) – *left*) and the virtual positions obtained using the *Transition Matrix* values, represented using a Fruchterman-Reingold layout algorithm (Figure 29(b) – *right*). One can globally observe a good match: close sensors have high



(a) SOUK (b) MILANO

Figure 30: AUC results of the (a) SOUK and (b) MILANO datasets for different deployment strategies.

values in the transition matrix (symbolized by bold lines on the right representation), and therefore end up with a low "virtual distance" when computing the LOCA interaction matrix. Figure 29(b) depicts an example of virtual sensors as created by the LOCA algorithm in the case two sensors overlap (the same strategy is applied for overlapping of more than two sensors). The virtual sensors are named after the concatenation of identifiers of intersecting sensors (*e.g.*, $3-9$ corresponds to the overlapping of sensors 3 and 9).

2.3.5 Results

		Strategy			
Model	Config	Grid	Spiral	Density	Random
gm	HD	0.651	0.719	0.790	0.703
gm	LD	0.523	0.540	0.553	0.551
rd	HD	0.663	0.738	0.840	0.752
rd	LD	0.529	0.553	0.555	0.557
rpgm	HD	0.834	0.872	0.922	0.870
rpgm	LD	0.696	0.700	0.798	0.762
rw	HD	0.696	0.766	0.916	0.820
rw	LD	0.524	0.545	0.722	0.658
rwp	HD	0.685	0.735	0.785	0.698
rwp	LD	0.557	0.604	0.580	0.547
tlw	HD	0.665	0.723	0.850	0.753
tlw	LD	0.520	0.534	0.600	0.561
trw	HD	0.837	0.874	0.898	0.860
trw	LD	0.650	0.685	0.770	0.748

Table 2.2: Global network inference using LOCA on synthetic dataset: AUC of the inference accuracy using 15 sensors of 1 m range.

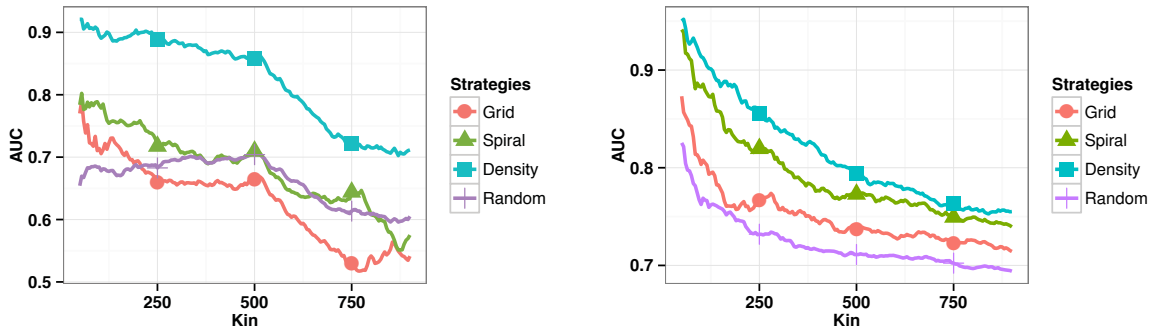
Synthetic/global Table 2.2 summarizes the inference accuracy AUC for the different considered synthetic models. High density setups perform consistently better. This is explained by the relative area covered with sensors: the HD setup area is 9 times smaller than the LD area, resulting in a higher coverage with the same number of sensors. A high coverage provides a wealth of data to accurately estimate relative distance of sensors and ensures that most of the interactions are captured.

Not surprisingly, models where social interactions have an impact on the physical proximity of the users (namely *trw* and *rpgm*) are more accurately inferred than purely random ones, even in the worst case of a random deployment strategy.

Real/global Figure 30 presents the AUC results for SOUK and MILANO datasets global inference using the different strategies and a varying number of sensors. It reads the following: for $K = 15$ sensors deployed during the SOUK cocktail, the AUC of LOCA inference is 66% for the Random strategy, 72% for the Grid strategy, 76% for the Spiral strategy and 89% for the Density strategy. Colored areas represent the standard deviation of the corresponding randomized strategies assessed over 10 independent runs. Since the SoA attack relies on full trajectories, it is not impacted by K, and yields 88% and 84% accuracies on SOUK and MILANO respectively.

On both datasets, the impact of K is clear: more sensors improve the inference accuracy. The only exception to this trend is the Density strategy on SOUK: it peaks quickly with surprisingly few sensors: 16 sensors (AUC is 93.14%) only cover about 12% of the 100 m^2 of the room, yet allow to identify the quasi-totality of the 150 most important social links. The slight decrease in the accuracy when more sensors are added is explained by the virtual sensor creation mechanism: as more sensors are only added in the densest areas, many sensing radii tend to overlap, leading to the creation of many "virtual sensors" that each contain little information. This globally degrades the quality of the Transition Matrix and therefore the quality of the inference.

The Spiral strategy follows the Density results: this translates the fact that most of the social interactions happened in the center of the room on both datasets. Covering the SOUK area with 36 sensors (28% of the room covered) yields a 83% accuracy. To completely cover the SOUK area, 120 sensors are required for 89.2% accuracy (point not shown on picture), slightly less than Density strategy optimum. This again illustrates the fact that too many sensors add much noise to the process that can impede the detection with spurious proximities. The Random strategy is only efficient when many sensors are available, but yet yields accuracies close to the SoA on MILANO.



(a) SOUK (b) MILANO

Figure 31: Impact of K_{in} on SOUK (a) and MILANO (b) datasets, considering a deployment of 15 sensors with a 1 m range.

Figure 32 zooms on the accuracy results by presenting the ROC curves of the LOCA global network inference for $K = 15$ on the SOUK dataset. The Density strategy quickly identifies 50% of the 150 targeted links with a low false positive rate, similarly to the SoA approach. Both Grid and Random strategies provide the same detection performance with 12% false positives. The random strategy identifies some of the targeted links early on, but then performs no better than a purely random classifier.

Sensitivity analysis/global Figure 31 illustrates the sensitivity of LOCA to the targeted social tie strength. The left-hand side of the picture, SOUK, considers 45 participants, therefore the number of possible social links is almost 1000. In Figure 31(a), $\mathsf{K}_{\mathsf{in}} = 250$ corresponds to the detection of the 25% strongest social ties. Similarly, on the right-hand side of the figure, MILANO contains 64 users, resulting

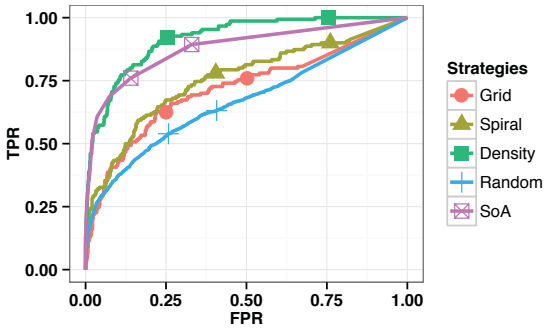


Figure 32: Detailed ROC for the SOUK dataset using 15 sensors: True Positive Rate (TPR) as a function of False Positive Rate (FRP).

in slightly more than 2000 possible links. Hence, in Figure 31(b), $\mathsf{K}_{\mathsf{in}} = 250$ corresponds to the detection of the first eighth of social ties.

As K_{in} grows, we measure the ability of LOCA to detect weaker social ties. As stronger ties are easier to detect than weaker ones, the global inference accuracy decreases when K_{in} grows. The hierarchy of strategies (*e.g.* density is more efficient than random) is globally unchanged, whatever K_{in}.

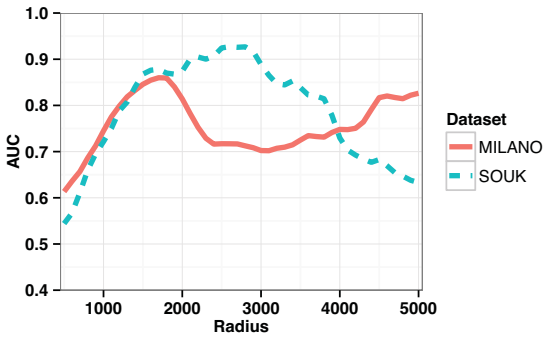


Figure 33: Impact of sensors range on inference accuracy for 15 sensors evenly distributed in space.

Figure 33 evaluates the impact of range on detection accuracy. Due to the different sizes of SOUK and MILANO experimental spaces, the peak accuracy is achieved at different ranges: 2.6 m for SOUK and 1.7 m for MILANO. One can observe that 3 m range sensors perform poorly on the MILANO dataset.

This fact is partly explained by the combined effect of our virtual sensors creation scheme and the different density of users. In the SOUK experiment, the spatial density is higher than in the MILANO experiment, reducing the coverage of the assumption that co-locations and interactions are correlated. From this observation, one may intuitively assume the optimal range would be smaller for SOUK than for MILANO. Let us now consider our virtual sensor definition strategy: if (at least) two sensors overlap and (at least) one co-location happens in an intersection, then we create a virtual sensor whose sensing area is the intersection of the overlapping sensors. If sensors are densely deployed, many ranges will overlap, and therefore many virtual sensors will be created. If too many of these virtual sensors are created, each will end up with a little fraction of the (finite) localisation events, leading to a relatively poor distance estimation. Hence, the optimal range for a scattered social event is finally lower than the optimal range we measured in a dense social event.

Badges/global Table 2.3 presents the AUC results for the Badges dataset. In this dataset, due to the specificities of the office environment, we chose to place sensors near points of interests only: coffee and

Day:	4	5	6	7	8	11
Population:	24	28	31	32	29	23
AUC:	0.776	0.740	0.683	0.665	0.620	0.588

Table 2.3: Badges Dataset results.

meeting rooms, printers, and managers' offices. In total, 30 sensors of 50 cm range are virtually deployed, covering approximately 7.5% of the office space. Results present a huge variability although each day is taken as an independent experiment. The low mobility of the users combined with the low number of social interactions happening per day could explain the lower accuracy of days 8 and 11.

SOUK and MILANO/local Figure 34 offers a local perspective on LOCA accuracy. It represents for each Dataset the probability of having x of each individual's 10 best friends correctly inferred. Surprisingly, SOUK outperforms MILANO from this local perspective. Note that this does not contradict the global results as the objectives are rather different: detecting the strongest social ties among a whole network is different than detecting each participant's strongest social ties. The difference between SOUK and MILANO can be explained by the participants behaviors: in SOUK, most of the 45 participants stayed for the whole experimentation, whereas many of the 64 participants of the MILANO cocktail left quickly, providing LOCA with very few data to infer their local ties.

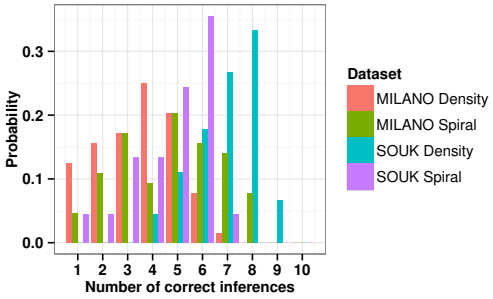


Figure 34: Probability to have exactly x out of 10 best friends correctly identified by LOCA using 15 sensors of 1 m range.

Together, these experimental results show the relatively good accuracy of LOCA inference considering the sparse information sources on which it relies. Moreover, they allow us to identify the key factors driving the inference accuracy: the number of sensors and their range, and above all the sensors deployment strategy. Indeed, having even a slight hint about the likely locus of future interactions (e.g. the Spiral strategy) considerably improves the inference accuracy over a Random sensor placement. Surprisingly enough, LOCA not only allows us to derive an accurate picture of the social network's most salient contacts, but also provides a fairly precise estimation of each users' individual contacts, as it correctly infers on average more than 6.5 out of the 10 strongest ties on the SOUK dataset for the Density strategy.

2.4 Related Works

The thread of co-location research started recently along with our ability to explore rich datasets containing both location and social information. One paradigmatic such dataset is the Reality Mining dataset and its analysis in [44]. Similarly, [40] exploited Flickr geo-tagged pictures at the global scale to infer Flickr declared friendship network, and [33] relied on the exploitation of both Location Based Systems and cell phone CDRs. Those works more generally question the interplay between physical proximity and social proximity, and typically rely on data having large geographical and temporal scales. Our work

focuses on local observation of users' behaviors (as opposed to, say, day and country-wide CDR-based user tracking) in order to stick to the IoT-based sensor use case.

More recent studies exploit the connection of proximity and social ties with a privacy angle [19, 96]. These landmark works evaluate the privacy implications of wireless communications in office environments. Having equipped for 3 months a set of 80 university users with tracked mobile phones, the authors use the campus Wifi map to locate users and infer, from an eavesdropper perspective, the social contacts of the users as captured by the tracked mobile phone. Their attack assumes the adversary is able to "trilaterate" mobile phones by using both Access Point positions and exfiltered RSSI values. In [96], the authors study the impact of obfuscated location information on co-location attack accuracy. Our approach is both more localized in time and space, and assumes that no location information is available at all.

A closely related study [41] exploits another wireless-related information dataset to infer social contacts: the list of known SSIDs often broadcasted by smart phones to speedup the link establishment to APs. The authors exploit the similarity of SSID lists to infer social proximity. In this approach, the vast variety of encountered SSIDs allows them to efficiently fingerprint users. In contrast, our approach is generic as it does not depend on the specificities of one wireless protocol, and exploits only proximities.

2.5 Mitigation strategies

We here present some suggestions to mitigate the success of Loca-like attacks. Those strategies are known: in that respect Loca simply extends the scope of co-location attacks against which strategies have already been elaborated.

A first step is to prevent Loca from following the same individual when he/she moves across different sensors. This can be achieved for instance by mix-zones [17], by extending the scope of traditional Location Privacy Protection Mechanisms [119, 55], or by simply relying on the use of many pseudonyms —this last solution applies only if the traceability of a user across different sensors is not an important feature for the provided services.

Another mitigation is to limit the attack surface of such systems, by avoiding network operations, especially if they contain identifying information. One strategy is to design "silent protocols" such that the carried device cannot be inadvertently detected by an eavesdropping device. Since RFIDs can be passively read at distance, the users can store them in shielded wallets that are available on the market.

2.6 Conclusion

In this part, we presented a framework to capture and analyze mobility data as a tool to extract interaction networks. We show that this tool enables to assess the realism and generality of models and traces, allowing us to better understand and simulate human-centered Short Range Communication-based systems for instance.

To the best of our knowledge the dynamics of such a dense crowd had never been assessed that precisely before. Analysis reveals that crowd behavior is all but random, demonstrating the need of a better model toolbox to design and test mobility-resilient software systems.

On the practical side, the Souk platform, which is mostly open source, has been designed as a scalable solution towards analysis of large crowds: although the results presented here are illustrated on an experiment involving 50 persons, the hardware cost and the complexity of software analysis (using Voronoï-based approach) are both linear with respect to the number of tracked individuals.

One of the applications of Souk is Loca, a simple co-location inference algorithm that has the particularity of not relying on positioning information. Instead, it uses sensed data to estimate the physical distance between sensors. The so-constructed *virtual distances* are then used to conduct the co-location inference attack. Experimental results show that Loca inference remains accurate even if the monitored space is sparsely covered, and highlights the importance of the sensor placement strategy in such contexts.

LOCA shows that location is not required to conduct co-location attacks. This conclusion strikes the potential privacy issue of trending IoT devices that could become potential attack surfaces for co-location attacks. Such attacks could be conducted either by directly monitoring users' proximity (i.e., altering the device's firmware to report nearby users), or for instance by monitoring the web page requests generated by close users' smart phones as LOCA requires no information about the sensors themselves.

Going further users' privacy, LOCA demonstrates a potential risk for service providers in the Internet of Things ecosystem: even though service providers may not store positioning information of deployed communicating objects, performing LOCA on the logs of users-objects interaction will provide lots of information on objects physical repartition. As such, by raising this issue over exploitation of users-objects interaction logs, we advocate the deployment of more decentralized architectures for sustainable ubiquitous systems deployment.

This part only spans the first results obtained around the Souk platform. Many research directions opened around these original works. Among those directions, one is to design models that reflect more closely crowd connectivity dynamics. Another direction is to study this evolving connectivity following a graph-theoretic perspective: like most observed *static* interaction networks that exhibit scale-free properties, we conjecture that the dynamics of interactions in human-carried mobile networks should have common properties tied to the geographical and social distribution of users.

Part IV
A model of $\mathcal{G}$ for dynamic graphs

This part is based on the following article:

- **The many faces of graph dynamics** In *Journal of Statistical Mechanics: Theory and Experiment, Volume 2017, June 2017* with Yvonne Anne Pignolet, Matthieu Roy, Stefan Schmid.

While the previous chapters provided $\mathcal{G}$with some structure depending on some application objectives, this part designs a general structure for $\mathcal{G}$. The goal is to provide a framework in which to represent and imagine many different problems related to a graph $G \in \mathcal{G}$ and its many different variations (be they due to noisy measurements or to the intrinsic dynamism of G).

It is interestingly the paper we had most troubles to publish, and what I paradoxically consider so far as my most interesting production. This is probably due to the type of contribution, as it is an article that provides many questions and a few answers. Moreover, it is a paper that brings together practical and theoretical approaches, and that also brought together co-authors from Part II and Part III for the most fruitful interactions.

In the previous part of this document, we focused on the process of measuring graphs based on experimental data. More precisely, Souk by essence produces dynamic graphs, rather than unique graphs. And although the first part provides bad news on the exploitability of observed graphs, the last chapter provided us with interesting applications around the graph observation that seem to resist this firsthand analysis.

One way to explain the quality of the obtained results lies perhaps in the dynamic nature of the graph observation which is implicitely exploited by `Loca` :all the graphs composing the observation of one dynamic graph contain noise but the repeated observation allows to average it out for instance when extracting a social network from these observations.

Another, maybe more pessimistic analysis of the Souk related results would also reveal that when justifying the accuracy of the observations, we never directly compare to some ground truth, either because it is not available, or because the accuracy of the process is measured using other methods: for instance, in `Loca` we only compare local neighborhoods, a measure that does not reflect the quality of the global inference. This last point illustrates the importance of an application-related perspective to metrology.

In this last part, we take a more general perspective on both graph metrology and the dynamic graphs. We argue that while both may seem unrelated, the theoretical toolbox to study them both involve the same essential tool: a distance function. Indeed, both require quantifying the difference between one graph and another, either because they represent the same process at different instants, or because one is the noisy observation of the other.

More precisely, this part is dedicated to presenting a family of such distances for the space of graphs. This is achieved by establishing a link between (inter-graph) distances, and centralities, a family of measures that aim at capturing the importance of a given node in a graph. Because the notion of importance is not consensual –and is application dependent–, by allowing the transposition of centralities to graph distances we allow the distances to also transport application-dependent flavours into the dynamic graph analysis.

1.1 Context

How do real-world networks evolve with time? While empirical studies provide many intuitions and expectations, many questions remain open. In particular, we lack tools to characterize and quantitatively compare temporal graph dynamics. In turn, such tools require good observables to quantify the (temporal) relationships between networks.

A natural prerequisite to measure evolutionary distances are good metrics to *compare* graphs. The classic similarity measure for graphs is the *Graph Edit Distance (*GED*)* [56]: the graph edit distance $d_{GED}(G_1,G_2)$ between two graphs G_1 and G_2 is defined as the minimal number of *graph edit operations* that are needed to transform G_1 into G_2. The specific set of allowed graph edit operations depends on the context, but typically includes node and link insertions and deletions. While graph edit distance metrics play an important role in computer graphics and are widely applied to pattern analysis and recognition, *GED* is not well-suited for measuring similarities of networks in other contexts [50]: the set of graphs at a certain graph edit distance d from a given graph G exhibit very diverse characteristics and seem unrelated.

A similarity measure that takes into account the inherent structure of a graph may however have many important applications. A large body of work on graph similarities focusing on a variety of use cases have been developed in the past (see our discussion in Section 1.6). Depending on the context in which they are to be used, one or another is more suitable. In particular, we argue that graph similarities and graph distance measures are also an excellent tool for the analysis, comparison and prediction of temporal network traces, allowing us to answer questions such as: *Do these two networks have a common ancestor? Are two evolution patterns similar?* or *What is a likely successor network for a given network?* However, we argue that in terms of graph similarity measures, there is no panacea: rather, graphs and their temporal patterns, come with many faces. Accordingly, we propose to use a parametric, *centrality-based approach* to measure graph similarities and distances, which in turn can be used to study the evolution of networks.

More than one century ago, Camille Jordan introduced the first graph *centrality* measure in his attempt to capture "the center of a graph". Since then the family of centrality measures has grown larger and is commonly employed in many graph-related studies. All major graph-processing libraries commonly export functionality for degree, closeness, betweenness, clustering, pagerank and eigenvector centralities. In the context of static graphs, *centralities* have proven to be a powerful tool to extract meaningful information on the structure of the networks, and more precisely on the *role* every participant (node) has in the network. In social network analysis, centralities are widely used to measure the importance of nodes, e.g., to determine key players in social networks, or main actors in the propagation of diseases, etc.

Today, there is no consensus on "good" and "bad" centralities: each centrality captures a particular angle of a node's topological role, some of which can be either crucial or insignificant, depending on the application. *Am I important because I have many friends, because I have important friends, or because without me, my friends could not communicate together?* The answer to this question is clearly context-dependent.

We argue that the perceived quality of network similarities or distances measuring the difference between two networks depends on the focus and application just as much. Instead of debating the advantages and disadvantages of a set of similarities and distances, we provide a framework to apply them to characterize network evolution from different perspectives. In particular, we leverage centralities to provide a powerful tool to quantify network changes. The intuition is simple: to measure how a network evolves, we measure the change of the nodes' roles and importance in the network, by leaving the responsibility to quantify node importance to centralities.

Contributions This work is motivated by the observation that centralities can be useful to study the dynamics of networks over time, taking into account the individual roles of nodes (in contrast to, e.g., isomorphism-based measures, as they are used in the context of anonymous graphs), as well as the context and semantics (in contrast to, e.g., graph edit distances). In particular, we introduce the notion of *centrality distance* $d_C(G_1,G_2)$ for two graphs G_1,G_2, a graph similarity measure based on a *node centrality* C.

We demonstrate the usefulness of our approach to identify and characterize the different faces of graph dynamics. To this end, we study five generative graph models and seven dynamic real world networks in more details. Our evaluation methodology comparing the quality of different similarity measures to a random baseline using data from actual graph evolutions, may be of independent interest.

In particular, we demonstrate how centrality distances provide interesting insights into the structural evolution of these networks and show that actual evolutionary paths are far from being random. Moreover, we build upon the centrality distance concept to construct dynamic graph signatures. The intuition is simple: we measure the probability of an update to be considered as an outlier compared to a uniformly random evolution. This allows us to quantify the deviation of a given dynamic network from a purely random evolution (our null-model) of the same structure for a set of centrality distances. The signature consisting of the resulting deviation values enables the comparison of different dynamisms on a fair basis, independently from scale and sampling considerations. Indeed, each signature is established by comparing a dynamic graph against its corresponding null model generated with same scale, same sampling rate: signatures hide those scale and sampling differences by only capturing deviations from a reference model.

Examples To motivate the need for tools to analyse network evolution, we consider two simple examples.

Example 1. [Local/Global Scenario] Consider the three graphs G_1, G_2, G_3 of Figure 35 over five nodes $\{v_1, v_2, \ldots, v_5\}$: G_1 is a line, where v_i and v_{i+1} are connected; G_2 is a cycle, i.e., G_1 with an additional link $\{v_1, v_5\}$; and G_3 is G_1 with an additional link $\{v_2, v_4\}$. In this example, we first observe that G_2 and G_3 have the same graph edit distance to G_1: $d_{GED}(G_1, G_2) = d_{GED}(G_1, G_3) = 1$, as they contain one additional edge. However, in a social network context, one would intuitively expect G_3 to be closer to G_1 than G_2. For example, in a friendship network a short-range "*triadic closure*" [75] link may be more likely to emerge than a long-range link: friends of friends may be more likely to become friends themselves in the future. Moreover, more local changes are also expected in mobile environments (e.g., under bounded human mobility and speed). As we will see, the centrality distance concept of this chapter can capture such differences.

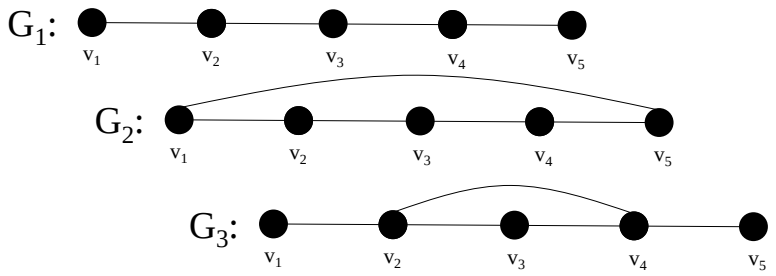


Figure 35: Local/Global scenario: G_1 is a line graph, G_2 and G_3 differ from G_1 by only one link.

Example 2. [Evolution Scenario] As a second synthetic example, consider two graphs G_L and G_S, where G_L is a line topology and G_S is a "shell network" (see also Figure 36). How can we characterize evolutionary paths leading from the G_L topology to G_S? Note that the graph edit distance does not provide us with any information about the likelihood or the role changes of *evolutionary paths* from G_L to G_S, i.e., on the order of edge insertions: there are many possible orders in which the missing links can be added to G_L, and these orders do not differ in any way when comparing them with the graph edit distance. In reality, however, we often have some expectations on how a graph may have evolved between two given snapshots G_L and G_S. For example, applying the triadic closure principle to our example, we would expect that the missing links are introduced one-by-one, from left to right.

The situation may look different in technological, man-made networks. Adding links from left to right only slowly improves the "routing efficiency" of the network: after the addition of t edges from left

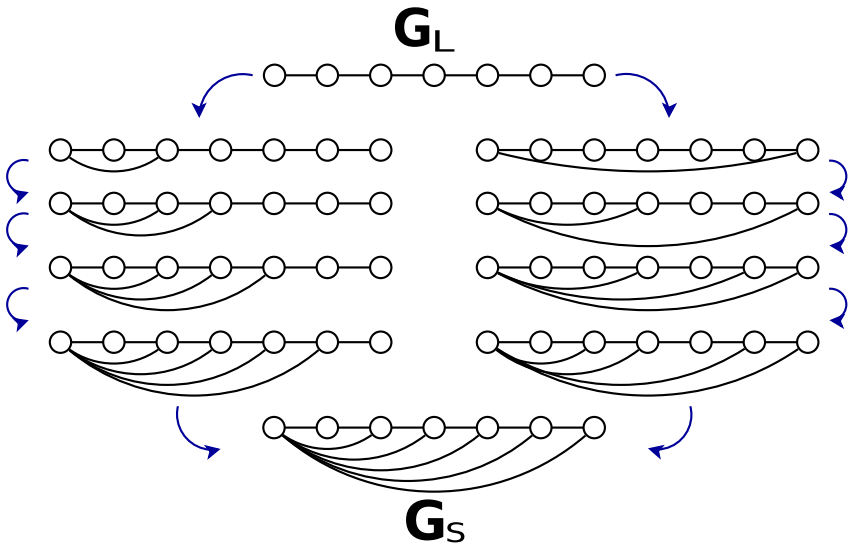


Figure 36: Two evolutionary paths from a line graph G_L to a shell graph G_S.

to right, the longest shortest path is $n-t$ hops, for $t < n-1$. A more efficient evolution of the network is obtained by connecting v_1 to the furthest node, adding links to the middle of the network, resulting in a faster distance reduction: after t edge insertions, the distance is roughly reduced by a factor t.

Thus, different network evolution patterns can be observed in real networks. Instead of defining application-dependent similarities with design choices focusing on which evolution patterns are more expected from a certain network, we provide a framework that allows the joint characterization of graph dynamics along different axes.

1.2 Preliminaries

We here focus on *labeled* graphs $G = (V, E)$, where vertices $v \in V$ have unique identifiers and are connected via undirected edges $e \in E$. In the following, we denote as $\Gamma(v)$ the set of neighbors of node v: $\Gamma(v) = \{w \in V \text{ s.t. } \{v, w\} \in E\}$. A temporal network trace is a sequence $T = [G_0, G_1, \ldots, G_l]$, where $G_i(V, E_i)$ represents the network at the i^{th} snapshot.

We focus on *node centralities*, a centrality being a real-valued function assigning "importance values" to nodes. Obviously, the notion of importance is context-dependent, which has led to many different definitions of centralities. We refer to [22] for a thorough and formal discussion on centralities.

Definition 1.2.1 (Centrality)**.** A *centrality* C is a function $C\colon (G, v) \to \mathbb{R}^+$ that, given a graph $G = (V, E)$ and a vertex $v \in V(G)$, returns a non-negative value $C(G, v)$. The centrality function is defined over all vertices $V(G)$ of a given graph G.

By convention, we define the centrality of a node without edges to be 0. We write $C(G)$ to refer to the vector in $(\mathbb{R}^+)^n$ where the i^{th} element is $C(G, v_i)$ for a given order of the identifiers.

Centralities are a common way to characterize networks and their vertices. Frequently studied centralities include the *degree centrality* (*DC*), the *betweenness centrality* (*BC*), the *closeness centrality* (*CC*), and the *pagerank centrality* (*PC*) among many more. A node is *DC*-central if it has many edges: the degree centrality is simply the node degree; a node is *BC*-central if it is on many shortest paths: the betweenness centrality is the number of shortest paths going through the node; a node is *CC*-central if it is close to many other nodes: the closeness centrality measures the inverse of the distances to all other nodes; and a node is *PC*-central if the probability that a random walk on G visits this node is high. We use the classical definitions for centralities, and the exact formulas are presented in Section 1.5.1 for the sake of completeness.

Finally, we define the graph edit distance *GED* between two graphs G_1 and G_2 as the minimum number of operations to transform G_1 into G_2 (or vice versa), where an *operation* is one of the following: link insertion and link removal.

1.3 Centrality Distance

The canonical distance measure is the graph edit distance, *GED*. However, *GED* often provides limited insights into the graph dynamics in practice. Figure 36 shows an example with two evolutionary paths: an incremental (*left*) and a binary (*right*) path that go from G_L to G_S. With respect to *GED*, there are many equivalent shortest paths for moving from G_L to G_S. However, intuitively, not all traces are equally likely for dynamic networks, as the structural roles that nodes in networks have are often preserved and do not change arbitrarily. Clearly, studying graph evolution with GED thus cannot help us to understand how structural properties of graphs evolve.

We now introduce our centrality-based graph similarity measure. We will refer to the set of all possible topologies by $\mathcal{G}$, and we will sometimes think of $\mathcal{G}$ being a graph itself: the "graph-of-graphs" which connects graphs with graph edit distance 1. Figure 36 illustrates the concept.

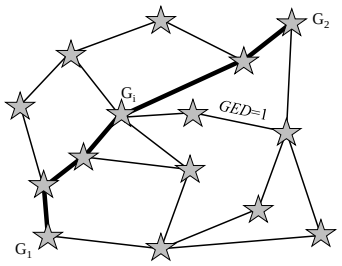


Figure 37: The graph-of-graph $\mathcal{G}$ connects named networks (represented as stars). Two networks are neighboring iff they differ by a graph edit distance of one. The centrality distance defines a distance for each pair of neighboring graphs.

Definition 1.3.1 (Centrality Distance)**.** Given a centrality C, we define the centrality distance $d_C(G_1, G_2)$ between two *neighboring* graphs as the component-wise difference:

$$\forall (G_1, G_2) \in E(\mathcal{G}), d_C(G_1, G_2) = \sum_{v \in V} |C(G_1, v) - C(G_2, v)|.$$

This definition extends naturally for *non-neighboring* graph couples: the distance $d_C(G_1, G_2)$ between G_1 and G_2 is simply the graph-induced distance.

$$\forall G_1, G_2 \in \mathcal{G}^2, d_C(G_1, G_2) = \min_{P \in \mathcal{P}_{G_1,G_2}} \sum_{i=1}^{|P|-1} d_C(P[i], P[i+1]),$$

where $\mathcal{P}_{G_1,G_2}$ represents the set of all possible paths between G_1 and G_2 in $\mathcal{G}$ and $P[i]$ represents the i^{th} graph on path P.

As we will see, the *distance axioms* are indeed fulfilled for the major centralities. The resulting structure supports the formal study with existing algorithmic tools. Let us first define the notion of *sensitivity*.

Definition 1.3.2 (Sensitive Centrality)**.** A centrality C is *sensitive* if any single edge modification of any graph G changes the centrality value of at least one node of G. Formally, a centrality C is *sensitive* iff

$$\forall G \in \mathcal{G}, \forall e \in E(G), \exists v \in V(G) \text{ s.t. } C(G, v) \neq C(G \setminus \{e\}, v),$$

where $G \setminus \{e\}$ is the result of removing edge e from G.

Lemma 1.3.3. *DC*, *BC* and *CC* are sensitive centralities.

Proof. Let $G \in \mathcal{G}$, and $e = (u,v) \in E(G)$.

Degree centrality: $DC(G,u) = DC(G \setminus \{e\}, u) + 1$: *DC* is sensitive.

Betweenness centrality: Recall our slightly changed definition of betweenness. Now, $BC(G,u) = \sum_{x,w \in V} \sigma_u(x,w)/\sigma(x,w)$. In $G \setminus \{e\}$, all shortest paths are at least as long as in G, and the shortest path between u and v has increased at least one unit: *BC* is sensitive.

Closeness centrality: $CC(G,u) = \sum_{w \in V \setminus \{u\}} 2^{-d(u,w)}$. In $G \setminus \{e\}$, all distances are greater or equal than in G, and strictly greater for the couple (u,v): *CC* is sensitive.

□

It is easy to see that also other centralities, such as cluster centralities and Page Rank centralities are sensitive. The distance axioms now follow directly from the graph-induced distance.

Theorem 1.3.4. For any centrality C, d_C is a distance on $\mathcal{G}$ iff C is sensitive.

Proof. We show that d_C is a metric on $\mathcal{G}$:

Separation $\forall G_1, G_2 \in \mathcal{G}, d_C(G_1,G_2) \geq 0$ since all summands are non-negative.

Coincidence If $G_1 = G_2$, we have $\sum_{v \in V} |C_1(v) - C_1(v)| = 0$. If $G_1 \neq G_2$ and C is sensitive, since $\forall G \in N(G_1), d_C(G_1,G) > 0$, necessarily $d_C(G_1,G_2) > 0$. For the sake of contradiction, assume C is not sensitive: $\exists G \in \mathcal{G}, e \in E(G)$ s.t. $\forall v \in G, C(G,v) = C(G \setminus \{e\}, v)$, and therefore $d_C(G, G \setminus \{e\}) = 0$ with $G \neq G \setminus \{e\}$: d_C is not a metric.

Symmetry Straightforward since $|C(G_2,v) - C(G_1,v)| = |C(G_1,v) - C(G_2,v)|$.

Triangle inequality Observe that the neighbor-based d_C definition associates each edge of $\mathcal{G}$ with a strictly positive weight. The multi-hop distance d_C is the weighted shortest path in $\mathcal{G}$ given those weights. Since the weighted shortest path obeys the triangle inequality for strictly positive weights, d_C does as well.

□

The centrality distance metric of Definition 1.3.1 however comes with a major drawback: it is impossible to compute in practice. Thus, we propose the following approximate version:

Definition 1.3.5 (Approximate Centrality Distance)**.** Given a centrality C, we define the approximate centrality distance $\widetilde{d_C}(G_1,G_2)$ between any two graphs as the component-wise difference:

$$\forall (G_1,G_2), \widetilde{d_C}(G_1,G_2) = \sum_{v \in V} |C(G_1,v) - C_2(G_2,v)|.$$

Note that $\widetilde{d_C} \leq d_C$ always holds. While the approximate distance can be far from the exact one in the worst case, it features some interesting properties.

We conclude by noting that given two centralities C_1 and C_2 and two arbitrary graphs G_1 and G_2 with n nodes, the respective distances are typically different, i.e., $d_{C_1}(G_1,G_2) \neq d_{C_2}(G_1,G_2)$. Hence, using a set of different centrality distances, we can explore the variation of the graph dynamics in more than one "dimension".

1.4 Methodology

In order to characterize the different faces of graph dynamics and to study the benefits of centrality-based measures, we propose a simple methodology. Intuitively, given a centrality capturing well the roles of different nodes in a real-world setting, we expect the centrality distance between two consecutive graph snapshots G_t and G_{t+1} to be smaller than the typical distance from G_t to other graphs that have the same GED.

To verify this intuition, we define a *null model* for evolution. A null model generates networks using patterns and randomization, i.e., certain elements are held constant and others are allowed to vary stochastically. Ideally, the randomization is designed to mimic the outcome of a random process that would be expected in the absence of a particular mechanism [58]. Applied to our case, this means that starting from a given snapshot G_t that represents the fixed part of the null model, if the evolution follows a null model, then any graph randomly generated from G_t at the given *GED* is evenly likely to appear.

Concretely, for all consecutive graph pairs G_t and G_{t+1} of a network trace, we determine the graph edit distance (or "radius") $R = d_{GED}(G_t, G_{t+1})$. Then, we generate a set S_{t+1} of $k = 100$ sample graphs $(H_i)_{i=1..k}$ at the same *GED* R from G_t *uniformly at random*. That is, to create H_i, we first start from a copy of G_t and select R node pairs, $(u_l, w_l) \in V^2, 1 \leq l \leq R$, uniformly at random. For each of these pairs (u_l, w_l) we add the edge (u_l, w_l) to H_i if it does not exist in G_t or we remove it if it was in G_t originally. Such randomly built sample graphs at the same graph edit distance allow us to assess the impact of a uniformly random evolution of the same magnitude from the same starting graph G_t: $\forall H_i \in S_{t+1}, d_{GED}(G_t, H_i) = d_{GED}(G_t, G_{t+1})$. In other words, G_t is the pattern and the evolution to H_i at graph edit distance R is the randomized part of the null model[1].

As a next step, given a centrality C, we compare G_{t+1} with the set S_{t+1} that samples the evolution following the null model. We consider that G_{t+1} does not follow the null model if it is an *outlier* in the set S_{t+1} for the centrality C. Practically, G_{t+1} is considered an outlier if the absolute value of its distance from G_t minus the mean distance of S_{t+1} to G_t is at least twice the standard deviation, i.e., if

$$|d_C(G_t, G_{t+1}) - \ \mu(\{d_C(G_t, x), x \in S_{t+1}\})| \ > 2\sigma(\{d_C(G_t, x), x \in S_{t+1}\}).$$

Given a temporal trace T, we define $p_{C,T}$ as the fraction of outliers in the trace for centrality C. An ensemble of such values $p_{C_i,T}$ for a set of centralities $\mathcal{C} = \{C_1, \ldots, C_k\}$ is called a *dynamic signature* of T.

1.5 Experimental Case Studies

Based on our centrality framework and methodology, we can now shed some light on the different faces of graph dynamics, using real world data sets.

- **Caida (AS):** This data captures the Autonomous Systems relationships as captured by the Caida project. Each of the 400 snapshots represents the daily interactions of the 1000 first AS identifiers from August 1997 until December 1998 [78].
- **ICDCS (ICDCS):** We extracted the most prolific authors in the ICDCS conference (IEEE International Conference on Distributed Computing Systems) and the co-author graph they form from the DBLP publication database (`http://dblp.uni-trier.de`). This trace contains 33 snapshots of 691 nodes and 1076 collaboration edges. The timestamp assigned to an edge corresponds to the first ICDCS paper the authors wrote together. Clearly, the co-authorship graph is characterized by a strictly monotonic densification over time.
- **UCI Social network (UCI):** The third case study is based on a publicly available dataset [97], capturing all the messages exchanges realized on an online Facebook-like social network between 1882 students at University of California, Irvine over 7 months. We discretized the data into a dynamic graph of 187 time steps representing the daily message exchanges among users.

[1] This is the least constrained randomization of network evolution w.r.t. the graph edit distance. More refined null models may preserve other structural graph properties in the sample graphs, e.g., their densities. The original publication also describes results obtained for a null model that guarantees the average degree of G_{t+1} in the sample graphs.

- **Hypertext (HT):** Face-to-face interactions of the ACM Hypertext 2009 conference attendees. 113 participants were equipped with RFID tags. Each snapshot represents one hour of interactions [78].
- **Infectious (IN):** Face-to-face interactions of the "Infectious: Stay away" exhibition held in 2009. 410 Participants were equipped with RFID tags. Each snapshot represents 5 minutes of the busiest exhibition day [78].
- **Manufacture (MA):** Daily internal email exchange network of a medium-size manufacturing company (167 nodes) over 9 months of 2010 [78].
- **Souk (SK):** This dataset captures the social interactions of 45 individuals during a cocktail, see [71] for more details. The dataset consists of 300 snapshots, describing the dynamic interaction graph between the participants, one time step every 3 seconds [71].

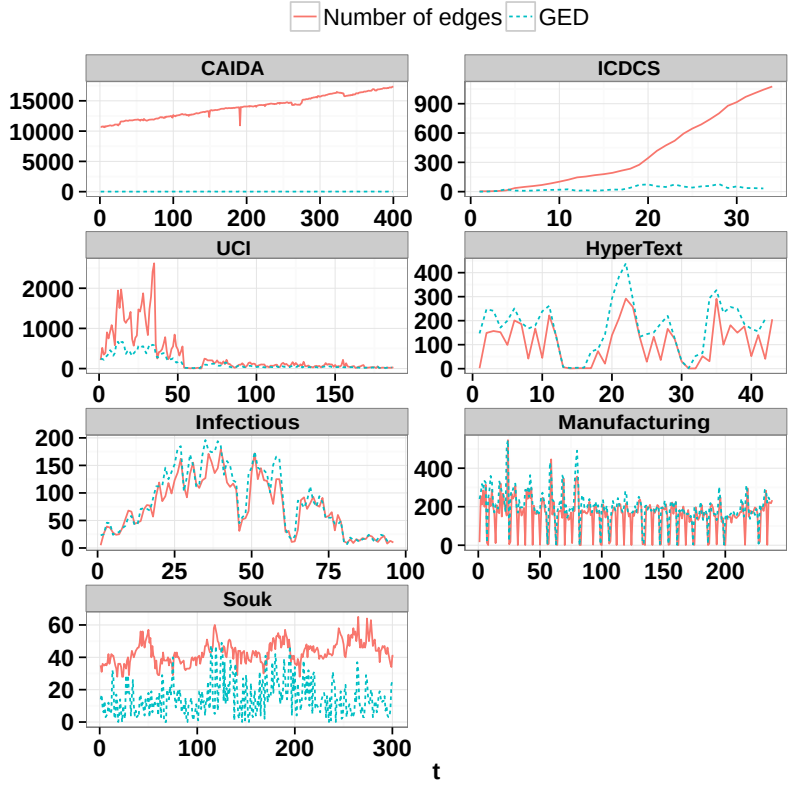


Figure 38: Number of edges and graph edit distance *GED* in the network traces.

Figure 38 provides a temporal overview on the evolution of the number of edges in the network and the *GED* between consecutive snapshots. Some of the seven datasets exhibit very different dynamics: one can observe the time-of-day effect of attendees interactions on *Hypertext*, and the day-of-week effect on *Manufacture*. *UCI*, *Hypertext*, *Infectious* and *Manufacture* all exhibit a high level of dynamics with respect to their number of links. This is expected for *Infectious*, as visitors come and leave regularly and rarely stay for long, but rather surprising for *Manufacture*.

The density of *Caida* slowly increases, and with a steady *GED*. Similarly, the number of co-author edges of *ICDCS* steadily increases over the years, while the number of new edges per year is relatively stable. The number of days of the conference *Hypertext* and the fact that conference participants sleep during the night and do not engage in social activity is evident in the second trace. The dynamic pattern of the online social network *UCI* has two regimes: it has a high dynamics for the first 50 timestamps, and is then relatively stable, whereas *Souk* exhibits a more regular dynamics. Generally, note that *GED* can be at most twice as high as the maximal edge count of two consecutive snapshots.

1.5.1 Centrality Distances over Time

To instanciate our framework, we picked a standard set of centralities:
Degree Centrality *DC*: Recall that $\Gamma(v)$ is the set of neighbors of a node v. The *degree centrality* is defined as:

$$DC(G,v) = |\Gamma(v)|.$$

Betweenness Centrality *BC*: Given a pair $(v,w) \in V(G)^2$, let $\sigma(v,w)$ be the number of shortest paths between v and w, and $\sigma_x(v,w)$ be the number of shortest paths between v and w that pass through $x \in V$. The *betweenness centrality* is:

$$BC(G,v) = \sum_{x,w \in V} \sigma_v(x,w)/\sigma(x,w).$$

For consistency reasons, we consider that a node is on its own shortest path, i.e., $\sigma_v(v,w)/\sigma(v,w) = 1$, and, by convention, $\sigma_v(v,v)/\sigma(v,v) = 0$. If G is not connected, each connected component is treated independently ($\sigma(x,w) = 0 \Rightarrow \forall v, \sigma_v(x,w)/\sigma(x,w) = 0$).
Ego Centrality *EC*: Let G_v be the subgraph of G induced by $(\Gamma(v) \cup \{v\})$. The *ego centrality* is:

$$EC(G,v) = BC(G_v, v).$$

Closeness Centrality *CC*: Let $\delta_G(a,b)$ be the length of a shortest path between vertices a and b in G. The *closeness centrality* is defined as:

$$CC(G,v) = \sum_{w \in V \setminus v} 2^{-\delta_G(v,w)}.$$

Pagerank Centrality *PC*: Let $0 < \alpha < 1$ be a damping factor (i.e. the probability for the random walk to stop [98]). The *pagerank centrality* of G is defined as:

$$PC(G,v) = \frac{1-\alpha}{n} + \alpha \sum_{w \in V \setminus v} \frac{PC(G,w)}{|\Gamma(w)|}.$$

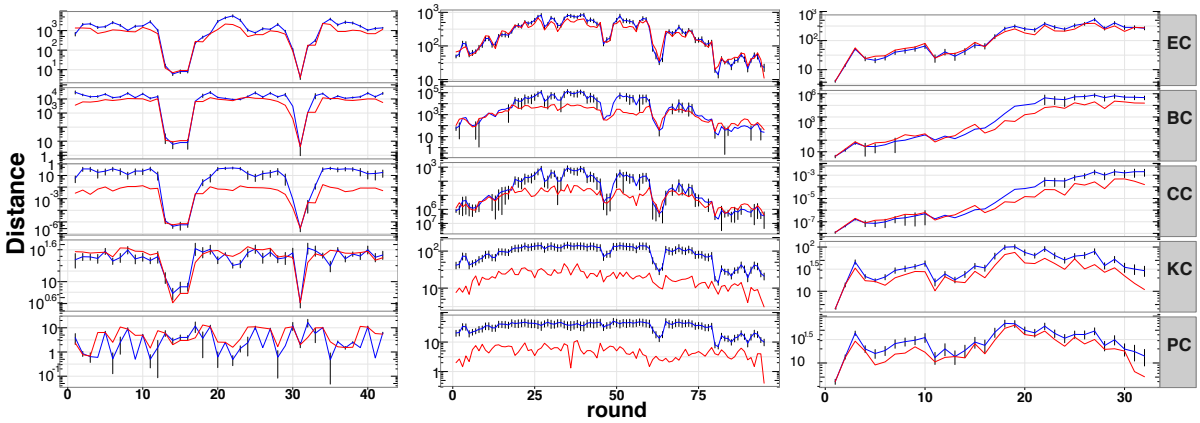
Cluster Centrality *KC*: The *cluster centrality* of a node v is the cluster coefficient of v, i.e., the number of triangles in which v is involved divided by all possible triangles in v's neighborhood. By convention, $KC(G,v) = 0$ for $|\Gamma(v)| = 0$, and $KC(G,v) = 1$ for $|\Gamma(v)| = 1$. For higher degrees:

$$KC(G,v) = \frac{2|\{\{j,k\} \text{ s.t. } (j,k) \in \Gamma(v)^2, (j,k) \in E\}|}{|\Gamma(v)|(|\Gamma(v)|-1)}.$$

Figure 39 presents examples of the results of our comparison of random graphs with the same graph edit distance GED as real-world network traces. The red dashed lines represent the centrality distances of G_t and G_{t+1}. The distribution of d_C values from G_t to the 100 randomly sampled graphs of S_{t+1} is represented as follows: the blue line is the median, while the gray lines represent the 2σ outlier detection window.

For most graphs under investigation and for most centralities it holds that the induced centrality distance between G_t and G_{t+1} is often lower than between G_t and an arbitrary other graph with the same distance. There are however a few noteworthy details.

Hypertext and *Infectious* exhibit very similar dynamics compared from a GED perspective as shown in Figure 38. Yet from the other centralities' perspective, their dynamism is very different. Consider for instance *Infectious* for *PC*, where the measured distance is consistently an order of magnitude less than the sampled one. This can be understood from the link creation mechanics: in *Infectious*, visitors at different time periods never meet. By connecting these in principle very remote visitors, the null model dynamics creates highly important links. This does not happen in *Hypertext* where the same group of researchers meet repeatedly. In the monotonically growing co-authorship network of ICDCS, we can observe that closeness and (ego) betweenness distances grow over time, which is not the case for the other networks in Figure 39.



Hypertext Infectious ICDCS

Figure 39: Centrality distance between G_t and G_{t+1} in dashed red lines and between G_t and 100 graphs with the same GED as G_t and G_{t+1} in solid blue lines representing the median, 2σ bars in grey. EC: Ego centrality, BC: Betweenness centrality, CC: Closeness centrality, KC: Cluster centrality, PC: Pagerank centrality.

When looking at other centrality distances, we observe that even though the local structure changes, a different set of properties remains mostly unaltered across different networks. Moreover, for some *(graph, distance)* pairs, like *KC* on *ICDCS*, *CC* on *Hypertext*, or *PC* on *Infectious*, the measured distance is orders of magnitude lower than the median of the sampled ones. This underlines a clear difference between random evolution and the observed one from this centrality perspective: the link update dynamics is biased.

1.5.2 Dynamics Signature

Figure 40 summarizes the $p_{C,G}$ signatures for $C \in \{CC, EC, BC, PC, KC\}$ applied to 7 real and 5 synthetic graphs in the form of a histogram chart—for synthetic graphs, each point is the average of 50 independent realizations of the model, and $|S_t| = 100$. That is, each chart represents the probability of having graph evolutions being outliers with respect to the null model for the corresponding centralities. Interestingly, this "distinction ratio" is not uniform among datasets. On *Caida*, *Infectious* and *UCI*, the ratio is high for local centralities such as *PageRank* and *Clustering*, and low for global centralities such as *Closeness* or *Betweenness*. On the contrary, *Hypertext* and *Manufacture* exhibit large ratios for global centralities and small ratios for local centralities. Both local and global centralities perform well on Souk. The difference of these behaviors show that these graphs adhere to different types of dynamics.

To complement our observations on real networks with graph snapshots produced according to a model, we investigated graph traces generated by some of the most well-known models: *Erdős-Rényi* ER [46], *random regular* RR [123], *Barabasi-Albert* BA [14] and *preferential attachment* [35] graphs with an equal number of node and edge events (CMHALF) and with the number of node events depending logarithmically on the time (CMLOG). Perhaps the most striking observation is that all tested dynamic network models have low $p_{C,T}$ values for all C. This is partly due to the fact that the graph edit distance between two subsequent snapshots is one and thus the centrality vectors do not vary as much as between the snapshots and the sampled graphs of the same graph edit distance for the real networks. Moreover, these randomized synthetic models are closer to the null model, and lack some of the characteristics (like link locality) of real world networks. Furthermore, we observe that each random network model exhibits distinct dynamics signatures, with ER being closest to the null model.

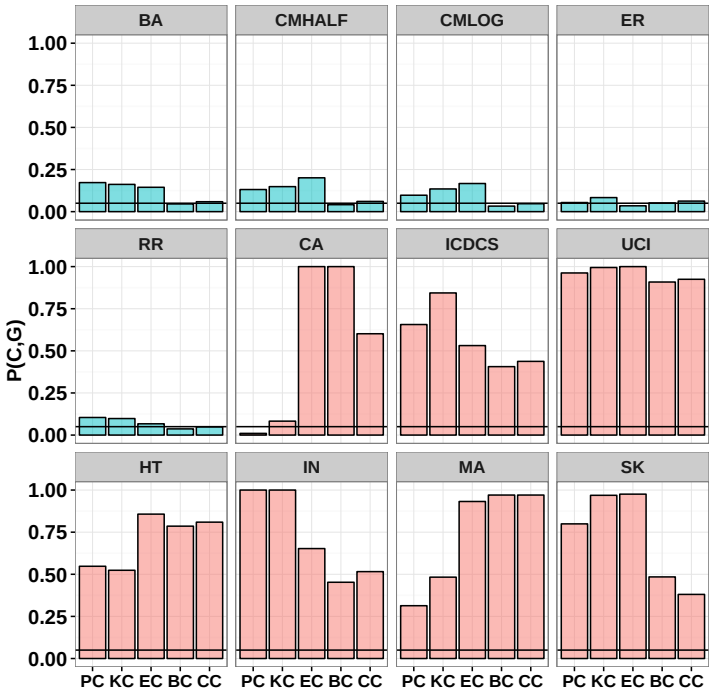


Figure 40: Histogram representation of the different facets of graph dynamics, the *dynamics signatures*. For each data set, a histogram chart for the different centralities is depicted, showing $p_{C,T}$, the probability that G_{t+1} is an outlier w.r.t. the null model for the corresponding centrality. Synthetic scenarios are depicted in *blue*, real scenarios in *red*. The black line at 5% represents the null model, i.e., the fraction of graphs that are at distance at least 2σ from the mean in a normal distribution. Synthetic datasets: BA: Barabasi-Albert, CMHALF: Preferential attachment—equiprobable nodes and edge events CMLOG: Preferential attachement—node events decay in log, ER: Erdös-Rényi, RR: Random Regular. Real-life datasets: CA: Caida, ICDCS: ICDCS co-authors, UCI: Online social network of UCI, HT: Hypertext conference, IN: Infectious MA: Manufacture mails, SK: Souk cocktail.

1.6 Related Work

To the best of our knowledge, this approach is the first to combine the concepts of centralities and graph distances. In the following, we review related work in the two fields in turn, and subsequently discuss additional literature on dynamic graphs.

Graph characterizations and centralities. Graph structures are often characterized by the frequency of small patterns called *motifs* [92, 117, 132], also known as *graphlets* [109], or *structural signatures* [36]. Another important graph characterization, which is studied in this paper, are *centralities* [24]. Dozens of different centrality indices have been defined over the last years, and their study is still ongoing, with no unified theory yet. We believe that our centrality distance framework can provide new inputs for this discussion.

Indeed, one of the recurring critics about centralities is that they lack a formal specification. For instance [22] says: "While many measures of centrality have been proposed, the category itself is not well defined (..). The one thing that all agree on is that centrality is a node-level construct. But what specifically defines the category?". Our approach, by connecting both concepts, could allow for a different take on the problem and structure the space of centralities by exploring the properties of the distances they define.

Graph similarities and distances. Graph edit distances have been used extensively in the context of inexact graph matchings in the field of pattern analysis. We refer the reader to the good survey by Gao et al. [56]. Soundarajan et al [121] compare twenty network similarities for anonymous networks.

They distinguish between comparison levels (node, community, network level) and identify vector-based, classifier-based, and matching-based methods. Surprisingly they are able to show that the results of many methods are highly correlated. NetSimile [18] allows to assess the similarity between k networks, possibly with different sizes and no overlaps in nodes or links. NetSimile uses different social theories to compute similarity scores that are size-invariant, enabling mining tasks such as clustering, visualization, discontinuity detection, network transfer learning, and re-identification across networks. The Deltacon method [49] is based on the normed difference of node-to-node affinity according to a Belief Propagation method. More precisely, the similarity between two graphs is the Root Euclidean Distance of their two affinity matrices or an approximation thereof. The authors provide three axioms that similarities should satisfy and demonstrate using examples and simulations that their similarity features the desired properties of graph similarity functions. Our work can be understood as an attempt to generalize the interesting approach by Faloutsos et al. in [49], which derives a distance from a normed matrix difference, where each element depends on the relationships among the nodes. In particular, we argue that there is no one-size-fits-it-all measure, and propose an approach parametrized by centralities.

Dynamic graphs. Among the most well-known evolutionary patterns are the shrinking diameter and densification [83]. A lot of recent work studies link prediction algorithms [7, 84, 133]. Others focus on methods for finding frequent, coherent or dense temporal structures [67, 118, 125], or the evolution of communities and user behavior [52, 136].

Another line of research attempts to extend the concept of centralities to dynamic graphs [38, 73, 82, 126]. Some researchers study how the importance of nodes changes over time in dynamic networks [126]. Others define temporal centralities to rank nodes in dynamic networks and study their distribution over time [73, 82]. Time centralities which describe the relative importance of time instants in dynamic networks are proposed in [38]. In contrast to this existing body of work, our goal is to facilitate the direct comparison of entire networks and their dynamics, not only parts thereof.

A closely related work but using a different approach is by Kunegis [77]. Kunegis studies the evolution of networks from a spectral graph theory perspective. He argues that the graph spectrum describes a network on the global level, whereas eigenvectors describe a network at the local level, and uses these results to devise link prediction algorithms.

1.7 Conclusion

This work proposes a centrality-based distance measure, and introduces a simple methodology to study the different faces of graph dynamics. Motivated by the observation that in terms of graph similarity measures, there is no "one size fits it all", our experiments confirm that the evolution patterns of dynamic networks are not universal, and different networks need different centrality distances to describe their behavior.

We observe that the edges in networks represent structural characteristics that are inherently connected to the roles of the nodes in these networks. These structures are maintained under changes, which explains the inertia of centrality distance which capture these properties. This behavior can be used to distinguish between natural and random network evolution.

We believe that this work opens a rich field for future research. We focused on five well-known centralities and their induced distances, and showed that they feature interesting properties when applied to the use case of dynamic social networks. However, we regard our approach as a *distance framework*, which can be configured with various additional centralities and metrics, which may not even be restricted by distance metrics, but can be based on the angles between centrality vectors or use existing correlation metrics (e.g., Pearson correlation, Tanimoto coefficient, log likelihood). Finally, exploiting the properties of centrality distances, especially their ability to distinguish and quantify between similar evolutionary traces, also opens the door to new applications, such as graph interpolation (what is a likely graph sequence between two given snapshots of a trace) and extrapolation, i.e., for link prediction algorithms based on centralities.

While these works were mostly presented under the perspective of dynamic networks, recent works [53] expose their transposability to the problem of network sampling. Indeed the authors exploit the same model of dynamic graphs containing labelled nodes, and exploit distances on $\mathcal{G}$ to study the convergence

of sampling processes before exploring higher-level operations like Principal Component Analysis. The beauty of the distance abstraction is indeed its capacity to provide tools for analysing graph variations that apply both to metrology and dynamism by considering both as processes that will *slightly* change a graph.

Part V
Conclusion and Future Works

It is now time to conclude this document. I will here first recall the different sections and their relation to our tour on graphs metrology, before higlighting some of the shortcomings of this thesis. Lastly, I will present some research directions that might provide interesting perspectives on the subject.

In the introduction, we presented the binary graph notion and its application to model real-world interacting systems. We have seen that this modeling operation is error-prone, and advocated for a metrological perspective on this operation. Moreover, in the introduction we mostly focused on the general approach: how to represent my system using a binary graph, and what are the impacts of such approach ? While this abstraction has originally been applied to systems where interactions and entities are reasonably well defined, we have seen that the graph abstraction is today applied to models where interactions might be hard to capture, or evolve over time, or both. This introduction presented graph capture (i.e. matching a real system with its graph abstraction) as a central problem of this abstraction process.

The first chapter presents a theoretical perspective on one method to capture graphs (that is, to match real systems with their binary graph abstraction): tomography. We studied algorithmic properties of two such tomography approaches yielding mostly negative results: in theory tomography is costly and error-prone. This raised a structuring contradiction: why are graphs so successful even though the means of observing these is in theory inaccurate and expensive ?

In the second part, we focused on one family of graphs: the social interaction networks. We presented a platform named Souk to experimentally capture such graphs, and analysed the obtained results. Then, in the context of privacy, we presented another method to capture such graphs, namely inference based on co-location information. This part introduced the important notion of models, that can from a high-level perspective be seen as particular sub-regions of the space of possible graphs $\mathcal{G}$ that concentrate the observables of one particular process. We have also illustrated through the privacy part that the need for accuracy can be relaxed depending on the context: identifying 60% of the links of a graph can be seen as successful in some contexts. Finally, this part led us to consider the temporal dimension of the problem: by repeatedly capturing the interactions of a real system, it is possible to extract meaningful global trends and average out some of the noise introduced by the graph-level sampling. As a result, it is possible to extract useful information out of rather noisy measurements.

These results point a source of this contraction between the practical use and the theoretical analysis of graph abstractions: real systems exhibit some internal and temporal coherence that is not accounted for in the theoretical studies. The necessity of developing theoretical tools to assess this coherence serves as a transition towards the next part.

In the last part, we provide a framework that could host observations regarding noise, graph dynamism, and more generally the coherence of graph objects over time. By considering dynamic graphs as trajectories in the space of all possible graphs, and by providing this space with adequate distance measures, we observe interesting results. Indeed, the inspected dynamic graphs evolve very differently and in a more deterministic fashion than our random models. The consequences of such observations are twofold: first, these bring some hope to circumvent sensitivity of individual graphs to errors of the sampling process by exploiting the temporal coherence of these observations. Second, by allowing such reflexions, it suggests that this broader perspective of dynamic graphs and trajectories in the space of graphs is a good abstraction that could prove fruitful.

1.1 Some shortcomings of this document

In this section, we elaborate on the limits of this thesis.

Delimiting the boundaries of these works

The ambition of this document is to concatenate different publications produced for different audiences into a single document coherently adressing one scientific question. While the diversity of these documents helps to broaden the picture around the problem of graph capture, focusing on specific problems mostly provides detailed perspectives that are arguably weakly inter-related.

Circumventing this effect could be achieved using two concurrent approaches: on one hand, finding a common system model (that would for instance precisely define what is meant by "real interacting system"), while on the other hand convince that the proposed conclusions hold at this common system model.

While the last part of this manuscript achieves so by considering "dynamic graphs" at large, such approach would require an extensive additionnal work for the two first parts.

Graph Theory

Moreover, by focusing right away our study on graphs representing "real" systems, we implicitely elude a huge branch of graph theory. Indeed, not all graphs are meant to be the direct abstraction of some real interaction network (consider for instance infinite graphs and grammars [30]).

This frontier is however arbitrary, and while the question of "what is real" is beyond the horizon of this document, precisely delimiting the scope of the presented results (i.e. clarifying where those results do not apply) would have improved the contribution. An important observation to nuance this aspect is to observe that generally we manipulate labelled graphs, a practical difference that mechanically excludes a large body of theoretic research on isomorphic graphs.

In addition, this document mixes conclusions driven by theoretical studies (Part 1) and experimental observations (Part 2,3) without much distinction. The goal is to provide arguments from both approaches in hope to find a middle ground between the top-down (Theory to practice) and the bottom-up (Practice to theory) methodologies. However, by doing so we implicitly put correctness (theory's quality criteria) and representativity (experience quality criteria) at the same level, which is misleading.

A more precise modelling of use cases

The introduction clearly presented the final application as one of the pillars of a metrology study. Beyond pointing inaccuracies, the goal of a metrological reasoning is to assess whether the phenomena are measured precisely enough for the considered application, hence the necessity of having the target use case correctly defined. There is no such attempt to chart the space of possible applications in this document. Indeed, these works usually pursue another approach by abstracting the use case. For instance, instead of trying to model all distributed applications in Part 2, we abstract the notion of application arbitrarily by a broadcast, arguing that most applications rely at some point on broadcast.

First, this opinion might not be consensual. Another approach that could for instance be based on some paradigmatic applications, but again establishing a consensus on the set of paradigmatic applications is difficult.

A more subtle downside of the "abstractive approach" (i.e. most applications rely on broadcast) is to make the contributions more abstract, which in turn reduces their impact on potentially interested audiences. For instance, the distributed application developer would probably have an easier time transposing his problem onto a couple of concrete applications.

Pipes and flows

When capturing interaction structures, the goal is often to *infine* provide information about the flows (be they information, contagion, data packet). However, all the thinking and modelling is done at the link (pipe)-level, exploiting the intuition that flows need to be carried along pipes. While this seems to be an interesting distinction, it is not carried throughout the document. For instance, tomography in Part 1 relies on flows to establish the pipe network, while Souk somewhat directly infers links at the pipe level.

While this confusion is common in other academic works, I believe it is problematic to maintain it throughout the document. Lifting such confusion would however require a broad exploration of the large body of graph-related works trying to identify patterns that belong predominantly to either flow-based or link-based approaches.

Machine Learning
This manuscript tackles networks, models and notions like inference, accuracy without ever looking at machine learning. While some machine learning techniques (especially neural networks) could have been either used as tools, either analysed as research objects in these works, it is also important to observe that neural networks have little to do with the networks of this document.

Neural networks typically represent a corner case of "real" networks – as these networks arguably do not represent reality, but are built as simple regressions of one phenomenon of the real world that can hypothetically be represented as a function. Moreover, they represent a very particular type of network (they are weighted, directed, and most importantly cycle-free), which would not fit the scope of this document.

This family of networks however represent a very interesting research subject, as those networks are at the focal point of an ever growing attention from the scientific and industrial community. One particularly interesting aspect of neural networks is their use as recommenders. Take for instance Youtube: the recommendations produced by its neural network [39] translate into hyperlinks on webpages that can be seen as a graph that one could call a recommendation network. Investigating on the connections between the nature of the recommender, the input data and the resulting recommendation network ends up (in my humble opinion) being very close to for instance the problem of monitoring network infrastructures. Both networks are the product of artifacts (in its sociological meaning) that roughly involve algorithms and distributed input, and end up being complex objects for which no easy description exists. Finally, recommendation networks are also difficult to capture because of their scale (which typically only allow partial exploration), their dynamism, so that their capture and exploitation raises problems that could fall in the scope of this document.

1.2 Future works

1.2.1 Theory

Dynamic graphs
This approach of dynamics and tomography on $\mathcal{G}$ allows to develop some interesting perspectives. Perhaps the most pressing need is for models of dynamic graphs. Such models would allow for instance to directly generate connection patterns without the intermediary step of generating node positions, but also maybe capture the effects of slow sampling, or alleviate problems of iid link measurement errors.

We've seen that each static graph model m defines a probability distribution on $\mathcal{G}$: they associate each $G \in \mathcal{G}$ with a probability $p_m(G)$. Chapter IV interprets $\mathcal{G}$ as a graph, considering each G as a vertex of $\mathcal{G}$, each link of $\mathcal{G}$ capturing an atomic topology evolution. This view exhibited a link between the simplest graph distance (the graph edit distance) and the simplest centrality (the degree centrality).

Since dynamic graphs define trajectories in $\mathcal{G}$, let us consider trajectories using a standard tool: random walks. The simplest form of random walk, the uniform random walk u: since $\mathcal{G}$ is regular, the stationary distribution of G is uniform (see e.g. [87]): $\forall G \in \mathcal{G}, p_u(G) = 1/|\mathcal{G}|$. Consider now the simplest static graph model, the Erdos-Renyi (uniform) graph model er. It has one parameter, p, the probability of each individual link. If $p = 1/2$, it is relatively easy to see that $p_{er} = p_u$: pick a graph G, fix one edge e, the probabilty to generate a graph having e is $1/2$ (same with non-edges). Therefore the probability to obtain exactly G is $(1/2)^{n(n-1)/2}$. In other words, there is a direct connection between the weights in $\mathcal{G}$, and random graph models. I believe it is important to explore this intriguing yet natural relation and try to transpose it to other models.

Indeed since our centrality-induced distances decorate the edges of $\mathcal{G}$ with weights, and since those weights could be used to bias random walks, each centrality induced distance would then be translated in a static graph model. This connection would work both ways: the standard, from the edge weights (centralities) to stationary distribution (static graph models), and the inverse: from the static graph models to graph distances and centralities. Note however that due to the enormous dimensions of $\mathcal{G}$, this question is mostly of theoretical interest. The very particular structure of $\mathcal{G}$ however probably enables many optimisations and simplifications that might help.

In addition to connecting static graph models and distances, this random walk approach also brings dynamic models as a bonus: each sequence of random walk steps is a dynamic graph. Interestingly, dynamic graphs generated by this model would in addition comply with the experimental observations of Chapter IV: random walks would statistically minimise the induced distance compared to uniform walks. Moreover, this connects with the intuition: all we know about real graphs is through the capture of such objects. But these objects (consider for instance a social network) existed long before the measurement and continue existing after: as if we captured them at a random time of their existence. In that sense it is exactly the definition of a stationary distribution.

Finally, I also believe the quality of a model also resides in its acceptance by the community. Constructing new models by bringing a temporal depth to static models (static graph models and dynamic graph models) like this seems more promising than producing new models in the hope they encounter success in the scientific community. Such approach would indeed have the benefit of not re-inventing new models –which would have to be sold to the community, as it would instead allow the transpositions of old models already accepted by the community.

Link creation mechanisms

Another common pattern of our approaches is the discrete yet important role of the link creation mechanism: the function responsible of the knitting of our topology. Usually, it is part of the model, like the set of axioms for traceroute, the vnet embedding model, or the hypothesis that individuals are in the plane. I believe an important approach would be to formalise the role of this link creation mechanism, as it implicitely regroups many distinct phenomena that do not necessarily share the same characteristics. For instance, a computer network, a predator/prey network or an author collaboration graph can all be abstracted by a graph, but the underlying dynamics (what leads two nodes to be connected or not) seem radically different. It is surprising that these differences do not translate into topological properties.

Chapter 1 provides interesting observations of the link creation mechanism in the case of individuals in the 2d plane. Indeed, we have seen that simulating random agents moving on the plane provides not so convincing results, hinting that geographical positions play a secondary role in this link creation. However, the Loca approach constructs a (fairly) good inference graph by reconstructing an approximate geographic map of the physical layout of the problem. In other words, to conduct co-location attacks without location information, the Loca first reconstructs (virtual) location information. The ambiguous role of geometry in this example should be investigated. Moreover, while this relation between links and geometric positions was explored (in the context of Unit Disk Graphs, see e.g. [76]), the dynamics of an evolving Unit Disk Graph (UDG) have not been studied. That is, given a dynamic graph $DG = \{G_1, \dots G_T\}$, I cannot precisely answer to the question "can DG be generated from moving individuals in the plane ?". I can verify whether a particular G_i represents a UDG, but not whether DG can be generated by points moving all along *the same* UDG.

More generally on the relation between graphs and euclidean spaces, Linial, London and Rabinovich have this very interesting sentence [85]: "Many combinatorial and algorithmic problems concern either directly or implicitly the distance, or metric on the vertices of a possibly weighted graph. It is, therefore, natural to look for connections between such questions and classical geometric structures. " I believe our capacity to alternate between discrete representations of networks (graphs) and continuous ones (euclidean embeddings) is a key to fast approximations algorithms, and beyond this, a key to modelling the dynamism of interaction structures: euclidean spaces enriched with a temporal dimension are a central tool of physics. There is hence a large toolbox available (notions of speed, acceleration, differential equations, etc.) that would probably greatly benefit the dynamic graphs analysis.

To summarise, we have seen in this document there is a gap between the practical relevance of graph, and the theoretical analysis of its accuracy. This observation calls for new theoretical tools to grasp this paradox, either from a global perpective based on the study of $\mathcal{G}$, either from a local perspective exploring the properties of link creation mechanisms.

1.2.2 Applications

Transparency

Another attractive field for understanding link creation dynamics is the *algorithmic transparency*. Indeed, the Web as we users explore it is in great part constructed by algorithms: filtering and recommendation algorithms that choose which information to present, ranking algorithms that select which information to present first, etc. Since the Web is a graph, observing the dynamics of such network can provide important informations on the internals of these algorithms [90, 91]. Moreover, while modelling social networks is intrinsically difficult because of the many parameters that influence each individual's own decisions, modelling algorithm-produced graphs can be easily done in-vitro with a complete control over simulation parameters. The interest I see in algorithmic transparency is therefore to combine a certain experimental ease (for collecting information and devising models) with an extreme societal importance.

Indeed, the position of some algorithms in the society's information flows is now very central. We cannot leave the full control over these algorithms to some firms led by objectives that will necessarily at some point conflict with a notion of fair treatment of the information.

Networking and shortest paths

Another interesting research direction relates to the distinction between flow and pipes. A class of systems in which this study could start are the graphs used as abstraction of (computer) networking systems.

This abstraction is very commonly used to represent datacenter flows, virtual networks, infrastructures, and so on. In some of these approaches, we model the infrastructure as a proxy to model the information flows. Information flows are then interpolated as travelling the shortest paths of the infrastructure. However, multiple shortest paths are very common (in non-tree topologies). Therefore, when problems like load balancing or network tomography –that directly relate to information flows– come with the input of a graph, they are underspecified: the choice of the actual shortest path is not defined.

It seems that refining this view (by formalising some properties on the set of shortest paths of a network) can provide considerable improvements in the complexity of the related problems. The relatively recent trend of Software Defined Networks, that continue to blur the (artificial) frontier between systems and networks, constitutes an exciting use-case for such graph-related approaches on networking.

Data collection

Finally, I'd like to stress the importance of data collection approaches. Theory can provide a powerful framework to abstract and manipulate reality, but in the context of graphs we have seen the prime importance of models: $\mathcal{G}$ is in general a too broad model of reality. Identifying and focusing on subparts of $\mathcal{G}$ is the key to accurate and efficient algorithmic approaches. To achieve this, we need to capture and constitute databases of dynamic interaction networks.

This task has largely been empowered by the digitalisation of our daily lives. Many of our activities are logged, recorded, analysed, generating an ever growing amount of data on our behaviours, ideas and opinions. However, it is important to realise that a large fraction of this data does not meet the standards of scientific data: it is only accessible to a small part of the community (through collaboration with firms), and it is collected through rather opaque means (e.g. smartphone applications). It is very difficult to estimate the biases introduced by the collection (who has a smartphone, which mechanical Turk users were directed to my ad) and by the collaboration (will I still get data from company X if I say this about their users). For these reasons, I believe we ought to design open data collection campaigns, and construct datasets allowing researchers to explore in complete freedom and produce reproductible research.

Of course, the constitution of such open data sets raises a number of privacy challenges. Those are important challenges that must be addressed. However, there is a huge gap between how privacy is officially considered in our society and the large scale tracking and personal data open bar organised by the major data brokers: this question is also a political one [110].

Bibliography

[1] H. Acharya and M. Gouda. The weak network tracing problem. In *Proc. Int. Conference on Distributed Computing and Networking (ICDCN)*, pages 184–194, 2010.

[2] H. Acharya and M. Gouda. On the hardness of topology inference. In *Proc. Int. Conference on Distributed Computing and Networking (ICDCN)*, pages 251–262, 2011.

[3] M. Acharya, H. a nd Gouda. A theory of network tracing. In *Proc. 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 62–74, 2009.

[4] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. In *Proc. 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 694–703, 2005.

[5] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger. Anatomy of a large european ixp. In *Proc. ACM SIGCOMM*, 2012.

[6] R. Aipperspach, T. Rattenbury, A. Woodruff, and J. Canny. A quantitative method for revealing and comparing places in the home. In *Procs. of UbiComp'06*, pages 1–18. Springer, 2006.

[7] O. Allali, C. Magnien, and M. Latapy. Internal link prediction: A new approach for predicting links in bipartite graphs. *Intelligent Data Analysis*, 17(1):5–25, 2013.

[8] C. Ambühl, M. Mastrolilli, and O. Svensson. Inapproximability results for maximum edge biclique, minimum linear arrangement, and sparsest cut. *SIAM J. Comput.*, 40(2):567–596, Apr. 2011.

[9] A. Anandkumar, A. Hassidim, and J. Kelner. Topology discovery of sparse random graphs with few participants. In *Proc. SIGMETRICS*, 2011.

[10] André Panisson. `pymobility`: python implementation of mobility models. `https://github.com/panisson/pymobility`, 2015.

[11] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proc. 6th ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 153–158, 2006.

[12] A. J. Aviv, M. Sherr, M. Blaze, and J. M. Smith. Evading cellular data monitoring with human movement networks. In *Proc. 5th USENIX Workshop Hot Topics on Security*, 2010.

[13] N. Bansal, K.-W. Lee, V. Nagarajan, and M. Zafer. Minimum congestion mapping in a cloud. In *Proc. 30th PODC*, pages 267–276, 2011.

[14] A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286, 1999.

[15] P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement (IMW)*, pages 5–17, 2001.

[16] F. Benbadis, J. Leguay, V. Borrel, M. Amorim, and T. Friedman. Millipede: a rollerblade positioning system. In *Procs. of ACM WiNTECH'06*, pages 117–118. ACM, 2006.

[17] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive computing*, 1:46–55, 2003.

[18] M. Berlingerio, D. Koutra, T. Eliassi-Rad, and C. Faloutsos. Network similarity via multiple social theories. In *Proc. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1439–1440, 2013.

[19] I. Bilogrevic, K. Huguenin, M. Jadliwala, F. Lopez, J.-P. Hubaux, P. Ginzboorg, and V. Niemi. Inferring social ties in academic networks using short-range wireless communications. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, pages 179–188. ACM, 2013.

[20] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[21] C. Boldrini, M. Conti, and A. Passarella. The sociable traveller: human travelling patterns in social-based mobility. In *Procs. of ACM MobiWAC'09*, pages 34–41. ACM, 2009.

[22] S. Borgatti and M. Everett. A graph-theoretic perspective on centrality. *Social Networks*, 28(4):466–484, 2006.

[23] V. Borrel, F. Legendre, M. D. De Amorim, and S. Fdida. Simps: using sociology for personal mobility. *IEEE/ACM Trans. Netw.*, 17(3):831–842, June 2009.

[24] U. Brandes and T. Erlebach. *Network Analysis: Methodological Foundations*. LNCS 3418, Springer-Verlag New York, Inc., 2005.

[25] C. Brown, C. Efstratiou, I. Leontiadis, D. Quercia, and C. Mascolo. Tracking serendipitous interactions: How individual cultures shape the office. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW 2014)*, 2014.

[26] C. Brown, C. Efstratiou, I. Leontiadis, D. Quercia, C. Mascolo, J. Scott, and P. Key. The architecture of innovation: Tracking face-to-face interactions with ubicomp technologies. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp 2014)*, 2014.

[27] M. Buchanan. Data-bots chart the internet. *Science*, 813(3), 2005.

[28] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012.

[29] D. Caucal. Deterministic graph grammars. *Logic and Automata*, pages 169–250, 2008.

[30] D. Caucal. On cayley graphs of algebraic structures. *arXiv preprint arXiv:1803.08518*, 2018.

[31] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, June 2007.

[32] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the internet. In *Proc. USENIX Annual Technical Conference (ATEC)*, 2000.

[33] E. Cho, S. A. Myers, and J. 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*, pages 1082–1090. ACM, 2011.

[34] M. K. Chowdhury and R. Boutaba. A survey of network virtualization. *Elsevier Computer Networks*, 54(5), 2010.

[35] F. R. Chung and L. Lu. *Complex graphs and networks*, volume 107. American mathematical society Providence, 2006.

[36] N. S. Contractor, S. Wasserman, and K. Faust. Testing multitheoretical organizational networks: An analytic framework and empirical example. *Academy of Management Review*, 2006.

[37] D. J. Cook and L. B. Holder. Substructure discovery using minimum description length and background knowledge. *J. Artif. Int. Res.*, 1:231–255, 1994.

[38] E. C. Costa, A. B. Vieira, K. Wehmuth, A. Ziviani, and A. P. C. da Silva. Time centrality in dynamic complex networks. *arXiv preprint arXiv:1504.00241*, 2015.

[39] P. Covington, J. Adams, and E. Sargin. Deep neural networks for youtube recommendations. In *RecSys*, 2016.

[40] D. J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg. Inferring social ties from geographic coincidences. *Proceedings of the National Academy of Sciences*, 107(52):22436–22441, 2010.

[41] M. Cunche, M.-A. Kaafar, and R. Boreli. Linking wireless devices using information contained in wi-fi probe requests. *Pervasive and Mobile Computing*, 11:56–69, 2014.

[42] L. Dall'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani. Exploring networks with traceroute-like probes: Theory and simulations. *Theoretical Computer Science*, 355(1):6–24, 2006.

[43] W. Dong, D. Olguin-Olguin, B. Waber, T. Kim, and A. S. Pentland. Mapping organizational dynamics with body sensor networks. In *International Workshop on Wearable and Implantable Body Sensor Networks*, volume 0, pages 130–135, Los Alamitos, CA, USA, 2012. IEEE Computer Society.

[44] N. Eagle, A. S. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences*, 106(36):15274–15278, 2009.

[45] N. Eagle and A. (Sandy) Pentland. Reality mining: sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268, Mar. 2006.

[46] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5:17–61, 1960.

[47] P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems (TOCS)*, 21(4):341–374, 2003.

[48] G. Even, M. Medina, G. Schaffrath, and S. Schmid. Competitive and deterministic embeddings of virtual networks. In *Proc. ICDCN*, 2012.

[49] C. Faloutsos, D. Koutra, and J. T. Vogelstein. DELTACON: A principled massive-graph similarity function. In *Proc. 13th SIAM International Conference on Data Mining (SDM)*, pages 162–170, 2013.

[50] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proc. SIGCOMM*, pages 251–262, 1999.

[51] J. Fan and M. H. Ammar. Dynamic topology configuration in service overlay networks: A study of reconfiguration policies. In *Proc. IEEE INFOCOM*, 2006.

[52] J. Ferlez, C. Faloutsos, J. Leskovec, D. Mladenic, and M. Grobelnik. Monitoring network evolution using mdl. In *International Conference on Data Engineering (ICDE)*, pages 1328–1330, 2008.

[53] D. Fraiman, N. Fraiman, and R. Fraiman. Nonparametric statistics of dynamic networks with distinguishable nodes. *Test*, 26(3):546–573, 2017.

[54] A. Galati and C. Greenhalgh. Human mobility in shopping mall environments. In *Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, MobiOpp '10, pages 1–7. ACM, 2010.

[55] S. Gao, J. Ma, W. Shi, and G. Zhan. Ltppm: a location and trajectory privacy protection mechanism in participatory sensing. *Wireless Communications and Mobile Computing*, 15(1):155–169, 2015.

[56] X. Gao, B. Xiao, D. Tao, and X. Li. A survey of graph edit distance. *Pattern Anal. Appl.*, 13(1):113–129, 2010.

[57] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. The flattening internet topology: Natural evolution, unsightly barnacles or contrived collapse? *Passive and Active Network Measurement*, pages 1–10, 2008.

[58] N. J. Gotelli and G. R. Graves. Null models in ecology. *online resources*, 1996.

[59] M. S. Granovetter. The strength of weak ties. In *Social networks*, pages 347–367. Elsevier, 1977.

[60] J.-L. Guillaume and M. Latapy. Complex network metrology. *Complex systems*, 16(1):83, 2005.

[61] M. Gunes and K. Sarac. Resolving anonymous routers in internet topology measurement studies. In *Proc. INFOCOM*, 2008.

[62] A. Haider, R. Potter, and A. Nakao. Challenges in resource allocation in network virtualization. In *Proc. ITC Specialist Seminar on Network Virtualization*, 2009.

[63] K. Herrmann. Modeling the sociological aspects of mobility in ad hoc networks. In *Procs. of ACM MSWIM'03*, pages 128–129. ACM, 2003.

[64] I. Houidi, W. Louati, and D. Zeghlache. A distributed virtual network mapping algorithm. In *Proc. IEEE ICC*, 2008.

[65] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *Procs. of WDTN'05*, pages 244–251. ACM, 2005.

[66] E. Inc. Estimote beacons. `http://www.estimote.com`, 2016.

[67] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *Proc. ACM SIGKDD*, pages 606–611. ACM, 2005.

[68] X. Jin, W.-P. Yiu, S.-H. Chan, and Y. Wang. Network topology inference based on end-to-end measurements. *IEEE Journal on Selected Areas in Communications*, 24(12):2182 –2195, 2006.

[69] D. Kamisaka, T. Watanabe, S. Muramatsu, A. Kobayashi, and H. Yokoyama. Estimating position relation between two pedestrians using mobile phones. In *Procs. of Pervasive Computing'12*, pages 307–324. Springer Berlin Heidelberg, 2012.

[70] M.-O. Killijian, R. Pasqua, M. Roy, G. Trédan, and C. Zanon. Souk: Spatial observation of human kinetics. *Computer Networks*, 111:109–119, 2016.

[71] M.-O. Killijian, M. Roy, G. Trédan, and C. Zanon. Souk: social observation of human kinetics. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 193–196. ACM, 2013.

[72] B. Kim, J.-Y. Ha, S. Lee, S. Kang, Y. Lee, Y. Rhee, L. Nachman, and J. Song. Adnext: a visit-pattern-aware mobile advertising system for urban commercial complexes. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, pages 7–12. ACM, 2011.

[73] H. Kim and R. Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107, 2012.

[74] M. Köhler, S. N. Patel, J. W. Summet, E. P. Stuntebeck, and G. D. Abowd. Tracksense: infrastructure free precise indoor positioning using projected patterns. In *Procs. of Pervasive computing'07*, pages 334–350. Springer-Verlag, 2007.

[75] G. Kossinets and D. J. Watts. Empirical analysis of an evolving social network. *Science*, 311(5757):88–90, 2006.

[76] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Unit disk graph approximation. In *Proceedings of the 2004 Joint Workshop on Foundations of Mobile Computing*, DIALM-POMC '04, pages 17–23, New York, NY, USA, 2004. ACM.

[77] J. Kunegis. On the spectral evolution of large networks. *PhD thesis*, 2011.

[78] J. Kunegis. KONECT – The Koblenz Network Collection. In *Proc. Int. Conf. on World Wide Web Companion*, pages 1343–1350, 2013.

[79] C. Labovitz, A. Ahuja, S. Venkatachary, and R. Wattenhofer. The impact of internet policy and topology on delayed routing convergence. In *Proc. 20th IEEE INFOCOM*, volume 1, pages 537–546, 2001.

[80] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in ip topology measurements. In *Proc. 22nd IEEE INFOCOM*, volume 1, pages 332 – 341, 2003.

[81] V. Lenders, J. Wagner, and M. May. Analyzing the impact of mobility in ad hoc networks. In *Procs. of ACM RealMan'06*, pages 39–46. ACM, 2006.

[82] K. Lerman, R. Ghosh, and J. H. Kang. Centrality metric for dynamic networks. In *Mining and Learning with Graphs*, 2010.

[83] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007.

[84] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proc. 12th International Conference on Information and Knowledge Management (CIKM)*, 2003.

[85] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[86] J. Lischka and H. Karl. A virtual network mapping algorithm based on subgraph isomorphism detection. In *Proc. ACM SIGCOMM VISA*, 2009.

[87] L. Lovasz. Random walks on graphs: A survey. *Combinatorics, Paul Erdos in Eighty*, 2, 1993.

[88] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 135–146. ACM, 2010.

[89] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38:69–74, 2008.

[90] E. L. Merrer and G. Trédan. The topological face of recommendation: models and application to bias detection. *CoRR*, abs/1704.08991, 2017.

[91] E. L. Merrer and G. Trédan. Uncovering influence cookbooks: Reverse engineering the topological impact in peer ranking services. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, CSCW 2017, Portland, OR, USA, February 25 - March 1, 2017*, pages 1413–1418, 2017.

[92] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. In *SCIENCE*, 2001.

[93] J. L. Moreno, H. H. Jennings, et al. Who shall survive? *Nervous and mental disease publishing co.*, 1934.

[94] M. Musolesi and C. Mascolo. Designing mobility models based on social network theory. *SIGMOBILE Mob. Comput. Commun. Rev.*, 11(3):59–70, July 2007.

[95] P. V. Nikitin and K. Rao. Performance limitations of passive uhf rfid systems. In *IEEE Antennas and Propagation Society International Symposium*, volume 1011, 2006.

[96] A.-M. Olteanu, K. Huguenin, R. Shokri, and J.-P. Hubaux. Quantifying the effect of co-location information on location privacy. In *Privacy Enhancing Technologies*, pages 184–203. Springer, 2014.

[97] T. Opsahl and P. Panzarasa. Clustering in weighted networks. *Social networks*, 31(2):155–163, 2009.

[98] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[99] R. Pasqua. *Inference and models of personal data : social mobility, spatial proximity.* Theses, Université Paul Sabatier - Toulouse III, Nov. 2016.

[100] A. Passarella, M. Conti, C. Boldrini, and R. I. Dunbar. Modelling inter-contact times in social pervasive networks. In *Proc. ACM MSWiM '11*, pages 333–340. ACM, 2011.

[101] S. Patel, K. Truong, and G. Abowd. Powerline positioning: A practical sub-room-level indoor location system for domestic use. In *Procs. of UbiComp'06*, pages 441–458. Springer Berlin Heidelberg, 2006.

[102] V. Paxson. End-to-end routing behavior in the internet. In *ACM SIGCOMM Computer Communication Review*, volume 26, pages 25–38. ACM, 1996.

[103] Y.-A. Pignolet, S. Schmid, and G. Tredan. Misleading stars: what cannot be measured in the internet? *Distributed computing*, 26(4):209–222, 2013.

[104] Y. A. Pignolet, G. Tredan, and S. Schmid. Misleading Stars: What Cannot Be Measured in the Internet? In *Proc. DISC*, 2011.

[105] G. Pirkl and P. Lukowicz. Robust, low cost indoor positioning using magnetic resonant coupling. In *Procs. of ACM UbiComp'12*, pages 431–440. ACM, 2012.

[106] J. M. Plotkin and J. W. Rosenthal. How to obtain an asymptotic expansion of a sequence from an analytic identity satisfied by its generating function. *J. Austral. Math. Soc. Ser. A*, 1994.

[107] I. Poese, B. Frank, B. Ager, G. Smaragdakis, and A. Feldmann. Improving content delivery using provider-aided distance information. In *Proc. ACM IMC*, 2010.

[108] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Procs. of ACM Mobicom'00*, pages 32–43. ACM, 2000.

[109] N. Przulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 2007.

[110] P. Rimbert. Données personnelles, une affaire politique (in french). In *Le Monde diplomatique – Septembre*, page 3, 2016.

[111] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *Proc. 16th ACM CCS*, pages 199–212, 2009.

[112] N. Robertson and P. Seymour. Graph minors. xx. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325 – 357, 2004. Special Issue Dedicated to Professor W.T. Tutte.

[113] M. Roy, G. Tredan, and C. Zanon. SOUK: Social & spatial observation of human kinetics. `http://projects.laas.fr/souk/software`, 2016.

[114] L. Ryan, J. Mulholland, and A. Agoston. Talking ties: Reflecting on network visualisation and qualitative interviewing. *Sociological Research Online*, 19(2):1–12, 2014.

[115] G. Schaffrath, S. Schmid, and A. Feldmann. Optimizing long-lived cloudnets with migrations. In *Proc. IEEE/ACM UCC*, 2012.

[116] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy. Network virtualization architecture: Proposal and initial prototype. In *Proc. ACM SIGCOMM VISA*, 2009.

[117] F. Schreiber and H. Schwöbbermeyer. Frequency concepts and pattern detection for the analysis of motifs in networks. *Transactions on Computational Systems Biology*, 3:89–104, 2005.

[118] N. Shah, D. Koutra, T. Zou, B. Gallagher, and C. Faloutsos. Timecrunch: Interpretable dynamic graph summarization. In *Proc. ACM SIGKDD*, pages 1055–1064. ACM, 2015.

[119] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying location privacy. In *2011 IEEE Symposium on Security and Privacy*, pages 247–262. IEEE, 2011.

[120] A. Soifer, B. Grünbaum, P. Johnson, and C. Rousseau. *The Mathematical Coloring Book: Mathematics of Coloring and the Colorful Life of its Creators*. Springer New York, 2008.

[121] S. Soundarajan, T. Eliassi-Rad, and B. Gallagher. A guide to selecting a network similarity method. In *Proc. SIAM International Conference on Data Mining*, 2014.

[122] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring isp topologies with rocketfuel. *IEEE/ACM Trans. Netw.*, 12(1):2–16, 2004.

[123] A. Steger and N. C. Wormald. Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8(04), 1999.

[124] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina, and P. Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*, 6(8), 08 2011.

[125] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proc. ACM SIGKDD*, pages 687–696. ACM, 2007.

[126] T. M. Tabirca, K. N. Brown, and C. J. Sreenan. Snapshot centrality indices in dynamic fifo networks. *Journal of Mathematical Modelling and Algorithms*, 10(4):371–391, 2011.

[127] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin. The impact of routing policy on internet paths. In *Proc. 21st IEEE INFOCOM*, volume 2, pages 736–742, 2002.

[128] J. Travers and S. Milgram. The small world problem. *Psychology Today*, 1(1):61–67, 1967.

[129] Ubisense. Series 7000 ip sensors. `ubisense.net/en/media/pdfs/factsheets_pdf/83188_series_7000_ip_sensors.pdf`.

[130] W. ur Rehman, E. de Lara, and S. Saroiu. Cilos: a cdma indoor localization system. In *Procs. of ACM UbiComp'08*, pages 104–113. ACM, 2008.

[131] T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explor. Newsl.*, 5:59–68, 2003.

[132] S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 3(4):347–359, Oct. 2006.

[133] S. H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike: joint friendship and interest propagation in social networks. In *Conference on World Wide Web (WWW)*, 2011.

[134] B. Yao, R. Viswanathan, F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *Proc. 22nd IEEE INFOCOM*, pages 353–363, 2003.

[135] S. Zhang, Z. Qian, J. Wu, and S. Lu. An opportunistic resource sharing and topology-aware mapping framework for virtual networks. In *Proc. IEEE INFOCOM*, 2012.

[136] X. Zhao, A. Sala, C. Wilson, X. Wang, S. Gaito, H. Zheng, and B. Y. Zhao. Multi-scale dynamics in a massive online social network. In *Proc. ACM IMC*, 2012.