



**THÈSE DE DOCTORAT DE
L'UNIVERSITÉ PIERRE ET MARIE CURIE**

Spécialité Informatique

École Doctorale Informatique, Télécommunications et Électronique (Paris)

**ANALYSE EXPLORATOIRE DE FLOTS DE LIENS
POUR LA DÉTECTION D'ÉVÉNEMENTS
(EXPLORATORY LINK STREAM ANALYSIS
FOR EVENT DETECTION)**

Présentée par

Sébastien HEYMANN

Pour obtenir le grade de

DOCTEUR de L'UNIVERSITÉ PIERRE ET MARIE CURIE

Soutenue le 3 décembre 2013, devant le jury composé de :

<i>Rapporteurs :</i>	Mme Christine LARGERON	Professeur, Université Jean Monnet
	M. François POULET	MdC HDR, Université de Rennes 1 - IRISA
<i>Examineurs :</i>	M. Jean-Philippe COINTET	Chargé de Recherche, INRA
	M. Jean-Gabriel GANASCIA	Professeur, UPMC
	M. Franck GHITALLA	Professeur, UTC
<i>Directrice :</i>	Mme. Bénédicte LE GRAND	Professeur, Université Panthéon-Sorbonne

Acknowledgements

It is a privilege to have collaborated with Bénédicte Le Grand as a supervisor. Her constant happiness, excitement and commitment have literally saved me during the darkest days! I also thank Christian Queinnec for his wisdom in validating our collaboration as president of EDITE.

To Christine Largeron and François Poulet, a great thank you for having reviewed my thesis, as well as Jean-Gabriel Ganascia, Jean-Philippe Cointet and Franck Ghitalla for accepting to be part of my jury.

I gladly acknowledge the financial help I received from the Conseil Régional d'Ile-de-France through the Domaine d'Intérêt Majeur (DIM) program. This program provided me a grant but more importantly, it has given me academic freedom. I thank the Institut des Systèmes Complexes - Paris Ile-de-France (ISC-PIF), the Réseau National des Systèmes Complexes (RNSC), and especially David Chavalarias and Edith Perrier for their interest in graph visualization.

I received help from numerous people at UPMC LIP6 ComplexNetworks team, Sorbonne Centre de Recherche en Informatique (CRI) and other institutions. I had especially insightful discussions with Mathieu Latapy and Clémence Magnien; they had the patience to teach me the rules of academic jobs, and their advises helped me to get a foothold in statistics. I also had adorable colleagues at ComplexNetworks: I thank Jean-Loup and Alice (and their ugly ducklings), Daniel and Raphaël for the macro-economic goûtés, Thomas and Lamia for their nerves in sharing the office with me, Maximilien for the games, Amélie for the fashion exhibitions, and also Lionel, Sergey, Elie, Fabien, Oussama, Hamid, Massoud, and Romain. I also thank my colleagues at CRI, in particular Rebecca and Charlotte for the chocolates.

I would also like to thank the administrative assistants who were always kind and saved me hours of paperwork in Kafkaesque situations, in particular Véronique Varenne at LIP6, Marilyn Galopin and Antoinette Ouedrago at EDITE, and Geneviève Tual at ISC-PIF.

To my teaching collaborators who offered me the opportunity to share my knowledge in computer science, network analysis and controversy mapping, I thank Isabelle Wertel-Fournier (Université de Paris 8), Paul Girard and Tommaso Venturini (Sciences Po Paris), Olivier Marchetti and Emmanuel Chailloux (UPMC), Nicolas Auray (Telecom ParisTech), Charles Pahlawan (École de Guerre Économique), Lukas Zenk (Technische Universität Wien), Павел Браславский and Александр Гальперин (Уральский федеральный университет).

I have also been so lucky to get invited in many external collaborations, so I thank Nicole Coleman and Dan Edelstein (Stanford Humanities Center), Gior-

gio Uboldi and Giorgio Caviglia (Density Design Lab), Benjamin Loyauté, Andrea Scharnhorst (Koninklijke Nederlandse Akademie van Wetenschappen), Katy Börner and Luca Aiello (Indiana University), Amira Mouakher and Sadok Ben Yahia (Université de Tunis), Stéphane Ribas (Inria), and Simon McIntyre (University of New South Wales). I thank a lot Mathieu Bastian and Hana Zdravilova for hosting me so long in SF when I had no money left.

To the Gephi team making an awesome effort in engineering and community building, I thank Mathieu (again), Eduardo Ramos Ibañez, Guillaume Ceccarelli, Dmitry Paranyushkin, Clément Levallois, Elijah Meeks, André Panisson, Cezary Bartosiak, and all the other contributors.

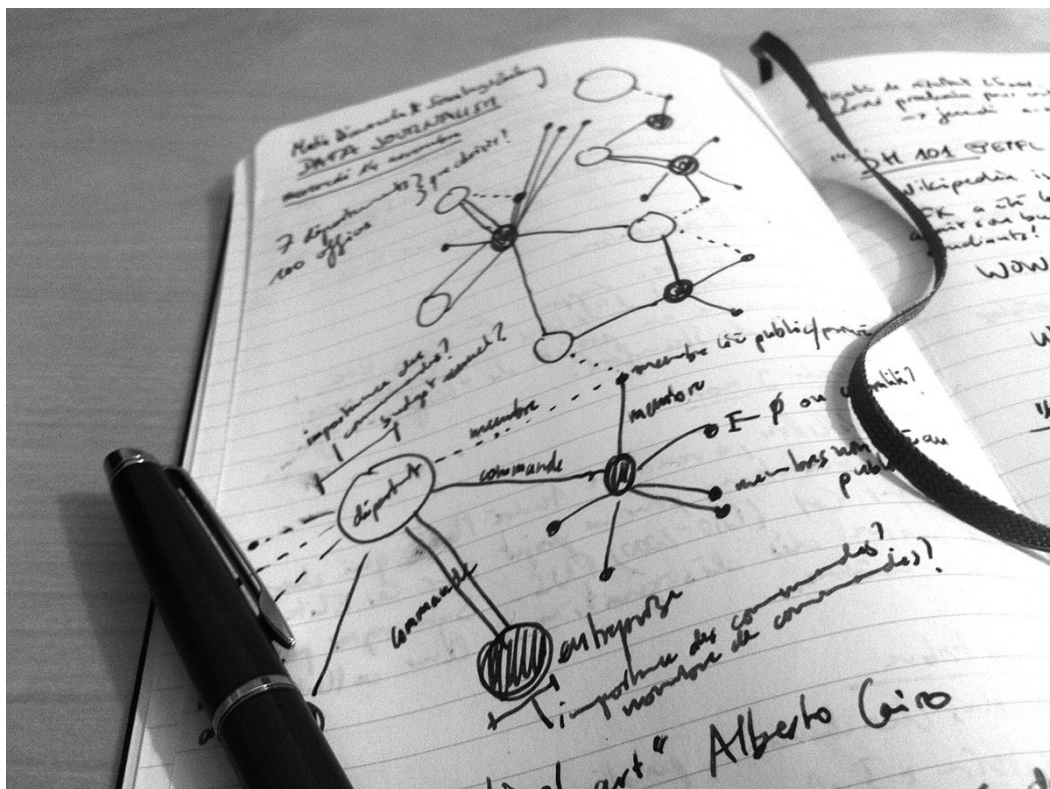
To my partners in crime at Linkurious SAS Jean Villedieu and Romain Yon, I hope to continue ruining your short nights and weekends! You are going to hate me for this, but I am sure that you will enjoy it a lot at the same time.

On a more personal note, thanks a million to my parents Chantal and Bernard, to my grand-mothers Marie-Madeleine and Stanislava, and to my little sister Estelle. Your love and unconditional support kept me up (and running)! To my girlfriend Vanessa and her family, thank you for illuminating my life (and for cooking). I would also like to mention my friends who kept me sane and happy during this time, especially Julien, Timothée, Laurine, Julian, Guillaume, Camille, Aurélien, Hassan, Jérémy, Elise, Anne, Thomas, and Sophia.

I also want to thank the people who put me in trouble, who leaved me or turned against me. They have shaped my path for better and for worse, but I would not change any bit of the past.

Finally, I would like to mention the WebAtlas crew, with whom this amazing journey has begun long before the Ph.D... Honor where honor is due, I thank Franck Ghitalla who has shared his passion and wonder of network science at Université de Technologie de Compiègne, and who convinced me to pursue a Ph.D. in this emerging field. I also thank Mathieu Bastian for the beer (aka Gephi inception), Marc Guichard and Raymon Duval for the INIST-CNRS deal, Guilhem Fouetillou and Alain Le Berre for the Linkfluence internship, Mathieu & Alexis Jacomy for the layouts and trolling coffees. I finally give my last words to Dana Diminescu: a special thank for trusting a crazy bunch of engineers and make the magic happens at FMSH TIC-Migrations program.

To nexis mirabiliæ!



A research notebook. Credits: Martin Grandjean, 2013.

Contents

1	Introduction	3
1.1	Network Dynamics & Link Streams	4
1.2	Problem Statement	6
1.3	Exploratory Framework	8
1.3.1	Exploratory Data Analysis	8
1.3.2	Non-Linear Processing Chain	10
1.3.3	Epistemological Perspective	11
1.3.4	Reaping Benefits from Data Mining Algorithms	12
1.4	Contributions & Organization of the Manuscript	14
2	Definitions & Datasets	17
2.1	Definitions	17
2.2	Datasets	21
2.2.1	French Population Size in the 20 th Century	21
2.2.2	Republic of Letters	21
2.2.3	Radar of Internet	22
2.2.4	Twitter network of COFA Online	23
2.2.5	Github Online Social Network	23
2.2.6	eDonkey P2P Network	24
3	Visual Event Detection	27
3.1	Related Work	27
3.1.1	Perceptual Support of Visualization	29
3.1.2	Emergence of Knowledge through Visualization	31
3.1.3	Visual Representation of Networks	33
3.1.4	Interaction	39
3.1.5	Global Approach for Network Visualization	40
3.1.6	Local Approach for Network Visualization	49
3.1.7	Visual Outlier Detection in Static Networks	53
3.1.8	Visual Event Detection in Temporal Networks	56
3.2	Our Experiments	64
3.2.1	Visual Events with the Global Approach	64
3.2.2	Interest Points on Static Networks with the Local Approach	65
3.3	Conclusion	67

4	Automatic Event Detection	69
4.1	Related Work	71
4.1.1	Outlier Detection in Static Networks	71
4.1.2	Event Detection in Dynamic Data	73
4.1.3	Event Detection in Temporal Networks	76
4.1.4	Conclusion	83
4.2	Outskewer	84
4.2.1	Skewness Signature	84
4.2.2	Our Method	85
4.2.3	Experimental Validation	87
4.2.4	Real-World Applications	94
4.3	Conclusion	99
5	Intrinsic Dynamics	101
5.1	Related Work	102
5.1.1	Notions of Topology Dynamics	102
5.1.2	Time Scale	103
5.1.3	Time Unit	107
5.2	Generalization of <i>Outskewer</i>	108
5.2.1	Use of a Sliding Time Window	108
5.2.2	Choice of the Time Unit	110
5.2.3	Sliding Window Width (Time Scale)	113
5.3	Application: Study of Github Internal Links	116
5.4	Conclusion	120
6	Event Detection & Visual Investigation	121
6.1	Exploratory Method for Event Detection in Link Streams	122
6.2	Description of the Prototype	124
6.3	Application: Study of Github Events	125
6.3.1	Automatic Event Detection	125
6.3.2	Visual Event Validation	127
6.4	Conclusion	130
7	Conclusion & Future Work	133
7.1	Conclusion	133
7.2	Future Work	135
7.3	Outro	136
	Bibliography	137

CHAPTER 1

Introduction

Contents

1.1	Network Dynamics & Link Streams	4
1.2	Problem Statement	6
1.3	Exploratory Framework	8
1.3.1	Exploratory Data Analysis	8
1.3.2	Non-Linear Processing Chain	10
1.3.3	Epistemological Perspective	11
1.3.4	Reaping Benefits from Data Mining Algorithms	12
1.4	Contributions & Organization of the Manuscript	14

Complex systems are made of interrelated elements with emergent features, i.e. which result from the interactions of the system's constituents and cannot be directly inferred from these individual constituents. Therefore a complex system cannot be reduced to the sum of its elements; this is precisely what makes it “complex” and raises difficult challenges.

Example of complex systems are very diverse, like the Internet topology where machines are connected through communication wires, the Web where hyperlinks allow users to navigate from page to page [AB02], peer-to-peer networks where users exchange files, but also online social networks, scientific collaboration and co-publication networks, biological networks of interactions between genes and proteins, linguistic networks, among others. They may involve millions, sometimes billions of entities.

These systems may be modeled as *networks* (also called *graphs*) where *nodes* represent the elements of the system and *links* materialize interactions between these elements. The way nodes are connected constitutes the *topology* of the network. Meta-data (i.e. attributes) can be associated with nodes and links as key-value pairs. For example, individuals of a social network may be characterized by their gender, language, and age, and their relationships may be of various types: friendship, work relation, etc. The analysis of complex networks has been the focus of many research works, and involves diverse tasks such as:

- understanding the statistical properties of their topology,

- identifying nodes and links of interest,
- detecting abnormal nodes, links, and structures.

1.1 Network Dynamics & Link Streams

Until recently, real-world networks have mostly been studied as static objects. However most of these networks are not static but dynamic, as elements and connections appear and disappear over time. This is called *topology dynamics* (see Section 5.1.1 for details). Another kind of dynamics is at stake: *flow dynamics* of processes that happen on networks (like information diffusion and virus propagation), which may be observed with nodes and links attributes. **In this thesis we focus on topology dynamics** because studying this dynamics is essential to characterize networks and to efficiently react to these changes. For instance, investigating Web dynamics helps reveal user behaviors [LMF]; studying the Internet dynamics helps detecting reliability issues [LMO08]. However, capturing such dynamics is difficult and, even when appropriate datasets are available, the dynamics is not easy to describe and to analyze. Revealing the underlying phenomena which lead to their evolution, and understanding **how and why these networks change over time**, is of high importance.

Dynamic networks may be classified into three different categories. The first one consists of graph snapshots (called ***time-aggregated networks*** after the decomposition of datasets of finer granularity, see Figure 1.1) representing the state of the network at different moments of time. An example is the topology of the Internet through the Autonomous Systems graph captured each year [EHSF12]. The second kind of dynamic network consists of a network where nodes and links existence is bound to time intervals (i.e. selection of time points), sometimes called ***time-ordered networks***. For instance, face-to-face human interactions may have different durations, from a few seconds to many hours [BCC⁺13]; airline timetables are bound to flight trajectories between airports and are subject to changes in case of flight delays. A formalism has been proposed in [KKK00] to study network connectivity, which has been used in several studies [SBF⁺08, KKT03, EP06, BHKL06].

The last kind of dynamic network consists of a series of changes (usually called ***stream***), like the addition (or *appearance*) and removal (or *disappearance*) of nodes and links. ***Link streams*** represent dynamic networks as a stream of chronologically ordered links, and are simplified representations of time-ordered networks where interactions are considered to be instantaneous because duration is less important. Link streams are therefore a specific type of stream that consists only of link additions (i.e. observed). They are appropriate to the representation of interaction networks, which represent a

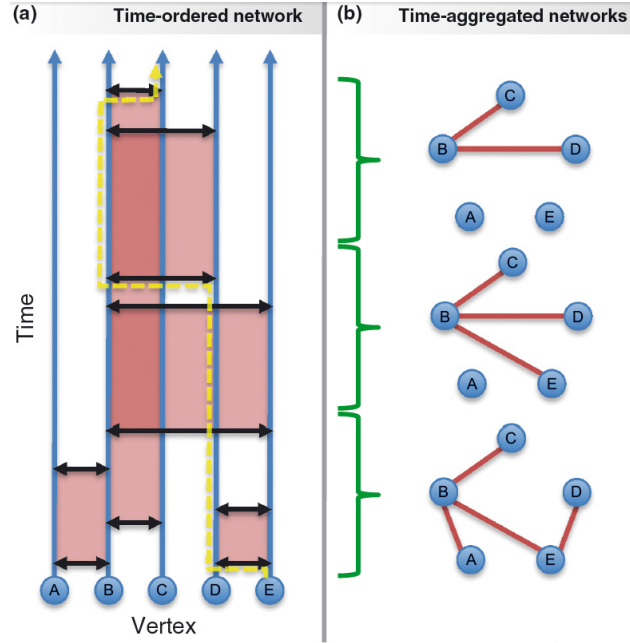


Figure 1.1: (a) Time-ordered networks capture all observed data for network dynamics. Nodes move forward in time and are connected at different times, enabling simultaneous visualization of network topology and diffusion processes. (b) Time-aggregated networks may be time-ordered networks after collapsing interactions that occur within certain time windows. [BWD⁺12]

large range of real-world networks. The activity of all online social networks, telecommunication networks, search engines, and even credit card payment systems may indeed be modeled by an interaction network. Each interaction may be represented as a link between two entities, which appears when the interaction occurs. Sexual networks and phone calls among people, email networks [DC05], *retweets* on the Twitter social network, and links between users who use the same keywords in search engine are just a few examples of link streams.

In spite of the diversity of complex systems which may be modeled as link streams, link streams dynamics has barely been explored for two reasons: network dynamics is itself a recent research topic, and the study of link streams has only become possible since recent years thanks to the release of large datasets of such precision. Most of dynamic networks studied so far are indeed series of graph snapshots captured at a specific frequency, e.g. a graph per month. The main issue¹ consists in characterizing the evolution of the underlying systems to better understand them and, in particular, differentiate normal dynamics from abnormal behaviors. From an analyst point of

¹See related works in Section 5.1

view who monitors a complex system and who has to raise alerts in case of anomalies (like a credit card fraud or an intrusion in an intranet), identifying and validating such events as fast as possible is critical. In other words, it is necessary to **determine *where* and *when* the topology of an interaction network abnormally changes**, ideally in a real-time fashion.

A significant body of research in interaction networks focuses on the detection of anomalies over time, called *events*. These methods rely on statistical analysis, visualization, data mining, etc. However the concept of event is usually related to specific applications, and a large part of these methods analyze sequences of graph snapshots. They thus miss a large amount of information available at a finer granularity [CE07] and using time scales that may be irrelevant to the scale of the real dynamics [KKB⁺12]. Finally, no general methodology of link streams exploratory analysis for event detection currently exists.

While the availability of large link stream datasets is rather new, we focus our study on them because they have the potential to describe the dynamics of large networks at a very precise granularity, contrary to graph snapshots [CE07, KKB⁺12, CPN⁺13]. Moreover, most studies on time-ordered networks focus on problems related to contact duration (e.g. temporal shortest paths or mean time delay between two nodes) [KKK00, BC13, HS12, KKK⁺11], whereas link stream datasets have no duration information associated to the links. We study events because they are tightly related to the characterization of link streams: nodes and links usually appear and disappear over time in such streams, so a core research question is to determine to what extent such dynamics is regular, and in which cases irregularities (i.e. events) occur. In this thesis we propose a methodology with no strong hypothesis on data nor on the underlying system’s behavior. **Our methodology provides an experimental framework for the exploration of real-world data modeled as link streams.**

1.2 Problem Statement

Events are a particular type of *outlier* related to the evolution of data over time. We will therefore use the term of outlier for anomalies in static data, and event in dynamic data. To give a general understanding of the challenges addressed in this thesis, we first introduce the problem of *outlier detection*, then we narrow the scope to event detection. We will discuss the methods and issues specific to *event detection* from a visualization perspective in Section 3.1 and from a statistical perspective in Section 4.1 respectively.

Outlier detection aims at finding data points very different from the others in a dataset. This field has received a large attention in the last decades because outliers often represent critical information about an abnormal be-

havior of the system described by the data. Outliers are also called: novelty, anomaly, noise, deviation, exception, or event (in a dynamic context) [HA04]. Outlier detection is a critical task in many domains. It helps for instance detect intrusions in computer networks, identify research fronts in scientific literature and patents, diagnose a fault while monitoring a critical system and fix it in real-time, evaluate the performance of a computer network, and detect auction or credit card frauds.

The diversity of applications has led to the introduction of various techniques for outlier detection [CBK09]. Research areas such as statistics, data mining, information theory and process control theory have produced diverse methods for spotting outliers in stochastic processes. More recently, anomaly detection is also addressed in networks like the Internet [GBBK11].

Existing methods may be divided into two categories: *univariate methods* (i.e. considering one variable), proposed in earlier works in statistics, and *multivariate methods* (i.e. considering multiple variables) which form the main part of the current body of research. We also distinguish parametric and non-parametric (model-free) procedures [Bg05]. Parametric procedures assume the values to be identically and independently distributed following a known probability distribution (generally a normal distribution), or at least a statistical estimate of the distribution parameters to fit the data. These methods flag as outliers the values that deviate from the model assumptions. They are often unsuitable for datasets without prior knowledge of the underlying distribution [PKGF03] because the hypotheses (e.g. the independence of observed values) are not satisfied. Statistical models are not reliable for real data and are hard to validate since many datasets do not fit one particular model.

Non-parametric procedures do not assume any knowledge of the data distribution, and thus learn to detect outliers. In some cases (*supervised learning*) labelled datasets are available, from which the program builds a model of normal behavior (and sometimes also a model of an outlying behavior). Otherwise (*unsupervised learning*), the procedure builds a probabilistic model of the dataset, and updates this model as new points appear. Non-parametric procedures classify as outliers the data points that deviate significantly from the model. These approaches are based on histogram analysis, kernel density, distance measures or clustering analysis. In the first (widely used) technique, the model estimation requires counting the frequency of occurrence of data points, thus inferring their probability.

The output of an outlier detector is a score of “outlierness” assigned to each data point, which represents its probability to be an outlier, or its distance from normal points. Data points are ultimately classified as outliers when their score is above a given threshold which is a parameter of the method.

Most of these problems and techniques are beyond the scope of our re-

search but they provide a general view of the diversity of existing approaches, as a result of the versatility of outlier definitions. Events are also an ambiguous notion and **one may consider as event any outlier related to the temporal evolution of data**. We see in Figure 1.2 that events can be found in many contexts. The classification shown in this figure and extracted from a survey published in 2013 is however incomplete: link stream analysis is missing. In this thesis we focus on event detection in link streams, but as little has been done so far on this topic we may look at related work in static and dynamic networks in general. We define our research question as follows:

“How to explore link streams to identify significant events, with no prior knowledge on the underlying complex systems under study?”

Our approach is *exploratory*: we know little about the studied system and we do not rely on *a priori* hypotheses to perform the analysis. We thus have to **detect statistically significant events in the evolution of a network without relying on a priori models of the dynamics**. By considering regular changes in the network, due to the appearance and disappearance of nodes and links at every moment, an event can be defined in relation to the normal dynamics of the network under study. This question depends directly on the characterization of what constitutes normal dynamics for the network. Such exploratory study requires multiples steps from data processing to event interpretation. In this thesis we propose a flexible methodology thanks to an appropriate exploratory framework.

1.3 Exploratory Framework

Exploratory approaches rely on a series of non-linear processes that eventually lead to new insights on data. These approaches usually comprise data acquisition, storage, mining, visualization systems, and communication of findings, wrapped up into an exploratory framework. The methodology we propose in this thesis to detect and validate events in link streams implies the creation of such an exploratory framework. We introduce its principles in this Section.

1.3.1 Exploratory Data Analysis

One of the biggest challenges encountered in network analysis is to get a good intuition of the network under study. Even when information like meta-data (e.g. age and gender in a social network) is available, extracting valuable knowledge and providing insights is challenging. Analysts may indeed deal with multiple dimensions such as social, topical, geographical, and temporal data, which may also be aggregated at different levels of detail.

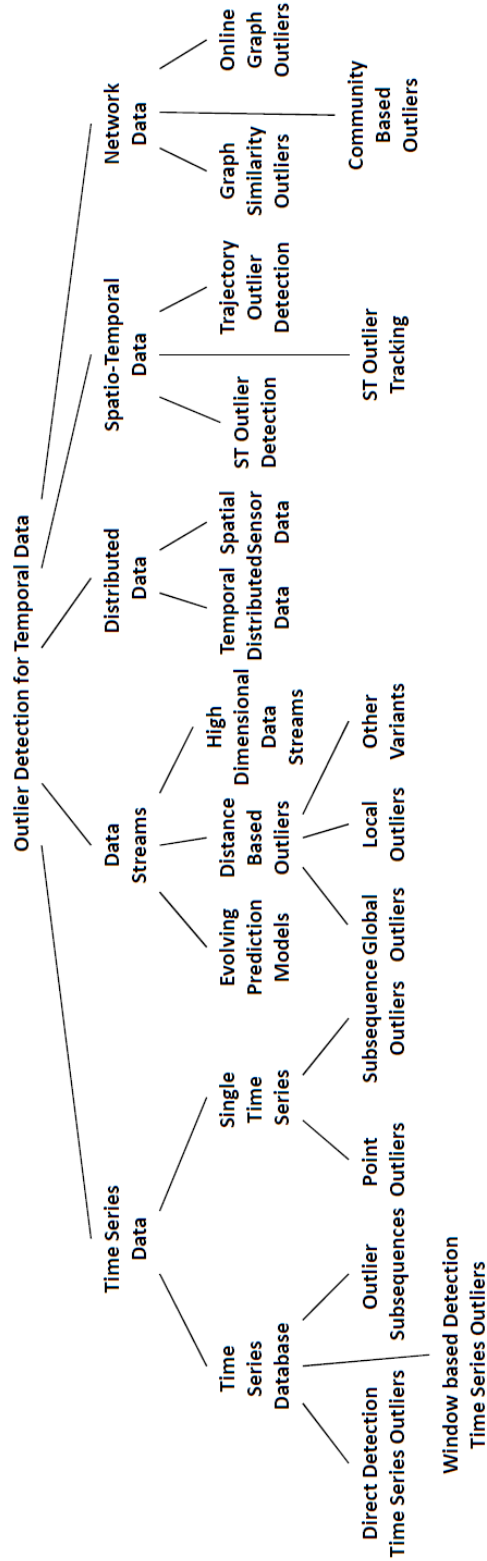


Figure 1.2: Classification of outlier detection approaches in temporal data [GGAH13]. This survey is recent (2013) but link stream analysis is missing.

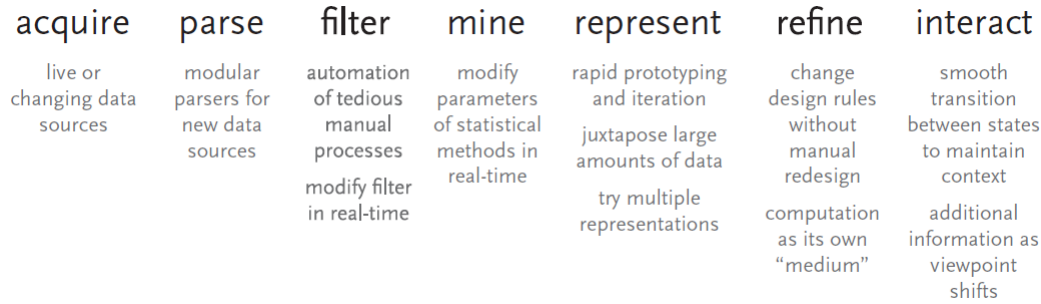


Figure 1.3: Steps of a processing chain [Fry04].

Faced with such diversity of data and the potentially unlimited number of analysis to perform at the first steps of a new project, analysts usually follow an exploratory approach to inspect data and outline interesting perspectives before drilling down to specific issues. When the datasets describe complex networks, this process is called **Exploratory Network Analysis (ENA)**; it is based on data visualization and manipulation to analyze complex networks. This framework takes its roots in the more general framework of **Exploratory Data Analysis (EDA)**, which consists in performing a preliminary analysis guided by visualization before proposing a model or doing a statistical analysis. Described by J. Tukey in 1962 [Tuk62], the philosophy of EDA can be summarized as follows:

“Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise.”

The main goal of EDA is to speed up the formulation of novel questions and relevant hypotheses on data [HSS⁺97, DHLZ13, RL05] through serendipitous findings (i.e. discoveries made while not necessarily looking for something in particular) and abductive reasoning². EDA’s process relies on visualization and interaction techniques embedded in a broader process, which includes data cleaning, storage, and mining. Related goals include error checking in data input, result validation, and faster finding of the facts we intuit.

1.3.2 Non-Linear Processing Chain

The process involved from data collection to information discovery requires a complete tool chain to acquire and parse data, filter, mine, then represent it and finally refine the visualizations interactively [Fry04], as illustrated in Figure 1.3. Nowadays, companies and research laboratories have access to a

²Coined by C.P.P. Peirce [Pei74], abduction is “a reasoning process invoked to explain a puzzling observation.” [AL97]

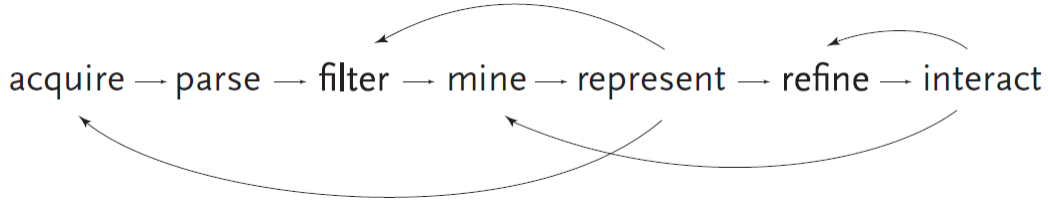


Figure 1.4: Illustration of a non-linear processing chain [Fry04].

large choice of methods and corresponding tools for each step. However their combination remains problematic because such variety makes the selection of the appropriate one difficult. Analysts must also learn how to use each new method properly, verify how to transfer data and intermediate results from one tool to another, check the integrity of the deployed processing chain to assess the validity of results, and eventually use the processing chain.

Moreover, this process is usually not static but dynamic: one rarely knows in advance which methods are relevant for the data under study. The freedom of disorderly processes was advocated by D. Engelbart in 1962 [Eng01]:

“When the course of action must respond to new comprehension, new insights and new intuitive flashes of possible explanations or solutions, it will not be an orderly process.”

Hence the process implies much trials and errors using various methods before discovering new information. As new questions and knowledge emerge during this process, each step must be modifiable at any time. Visualization may indeed reveal the need to acquire more data, or filter it in another way; interacting with it may require to change visual variables and aesthetics. Computing basic statistics and proceeding to an early visual exploration of data before performing more specific analysis can thus provide relevant hypotheses to start with. The interaction may also highlight new statistical patterns, hence requiring new visual refinements. This problem is clearly summarized by B. Fry on Figure 1.4. The validity of processing chains are questionable as we see in the following Section.

1.3.3 Epistemological Perspective

How can the data processing chain generate valid information on the objects of study? With the multiple steps involved from “raw” data (which are already constructed from such objects) to final representations, it is surprising that analysts’ discourse on objects of study can still be related to the objects themselves. An important theory to solve this epistemological problem was coined in [Lat95] with the “chains of circulating reference”. By observing how scientists transform the soil of Boa Vista forest into scientific facts, B. Latour

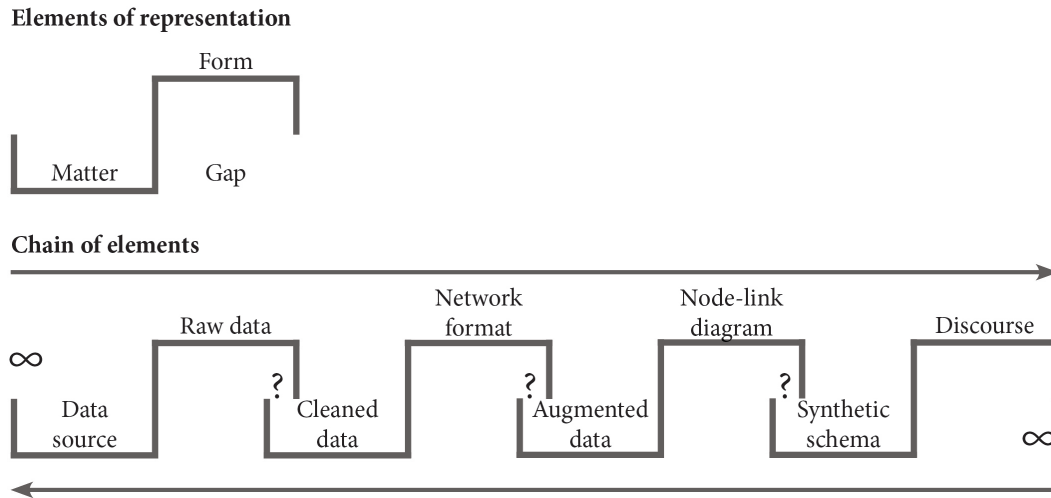


Figure 1.5: Circulating reference in a common processing chain of complex networks data. Data source may be the studied object, or may be an intermediary between the object and raw data. At some point data must be encoded in a network format (either in a file or a database) to be studied as such, and augmented with data mining results or third-party data. The schema is revamped from the more general one in [Lat95].

has remarked that scientific studies follow a series of transformations, each one going from matter to forms by creating a gap: forms lose material properties, but gain semiotic properties related to that matter. In this perspective, reference is a property of transformation chains which depends on the quality of transformations. Such chains can conduct truth (like copper wire conducts electricity) only if they remain reversible, i.e. changes can be traced back and forth so that valid reference circulates along chains without interruption. The circulating reference was originally illustrated by Latour. We revamp his schema on Figure 1.5, in an attempt to apply it to the processing chain of complex network data. We see in the next Section how augmented data (which is part of this chain) generated by data mining algorithms may speed up visual analysis.

1.3.4 Reaping Benefits from Data Mining Algorithms

Complementary data are indeed sometimes required to quickly perform visual analytic tasks, such as identifying the shortest path between two nodes. A solution to this problem is to augment data with the results of data mining algorithms, then to integrate them into visualizations.

For instance, consider the identification of all communities of the network, i.e. the groups of nodes with dense connections within groups and sparser connections between groups. The Louvain community detection al-

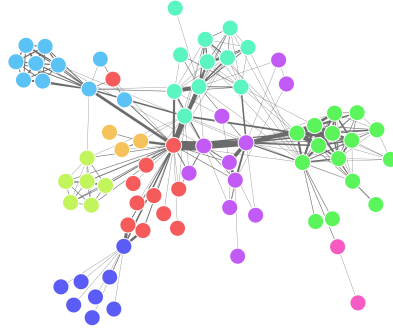


Figure 1.6: Sample of network visualization where node colors correspond to their community (computed with the Louvain algorithm with resolution=1).

gorithm [BGLL08] may be used to find one partition of the graph which maximizes a given quality function (modularity) of the community structure. Because this algorithm detects non-overlapping communities, each node is assigned to exactly one community. Analysts can color nodes according to their community, as illustrated in Figure 1.6.

In another example, we consider the identification of the most central links, where a central link is defined as a link traversed by the greatest number of shortest paths. One may compute all shortest paths using the algorithm of [Bra08] and map the results to lines thickness in the node-link diagram.

In a last example, consider the detection of someone’s “influential social circles” (where the influence is let to be defined by the analyst) in a social network. A possible method consists in filtering [AS94] the network to highlight the nodes surrounding a selected node. But too many nodes are displayed if the node (or its direct neighbors) has a high number of connections. A solution is to define a function usually called “degree of interest” [Fur86], which computes a score indicating to what extent each node is related to the selected node, then to prune the visualization by keeping only the nodes with high scores. This method has been used in [vHP09] in another context.

Data mining algorithms may also be executed by interacting with the representation, like computing the shortest path after having selected the path endpoints. Integrating these algorithms into the visualization and making them available at any time of the exploration is thus a solution to include them in the non-linear processing chain.

In this Section we have introduced the principles of exploratory frameworks to generate new insights. In this thesis we define a specific framework in order to detect, validate and interpret events in link streams.

1.4 Contributions & Organization of the Manuscript

The work conducted during this thesis contributes to the investigation of events in link streams in many aspects:

1) Definitions: as link streams have not been thoroughly studied before to our knowledge, we propose a *rigorous definition* of link streams and define the notions related to the study of their dynamics, in particular the concept of statistically significant event.

2) Visual approaches: we first experiment two approaches for event detection based on visualization. In the continuity of the large majority of methods that display an overview before focusing on particular data points, we propose an **augmented timeline** (implemented in Gephi software³) which integrates the chart of a time series statistic in a dynamic graph (e.g. the evolution of the number of nodes). This technique helps users select a sub-graph corresponding to an event observed on the chart, but it remains limited by the graph size. Then we propose an approach based on the **incremental exploration of a graph**, that allows experts to reveal points of interest in static graphs. This experiment encourages further studies of local approaches to visual exploration, which nonetheless have to be preceded by an automatic detection of abnormal time periods (i.e. events) in graph evolution.

3) Automatic event detection: most current methods of event detection need a priori knowledge on the observed system, but we would like to detect events with no prior information on it. We thus propose a novel method, called *Outskewer*, which allows to detect statistically significant anomalies both in samples and time series, with no parameter but the size of the sliding window on time series (which is a required parameter for multi-scale analysis). When applied to link streams, this method enable us to characterize their dynamics by distinguishing a regular dynamics (i.e. when nodes and links appear and disappear regularly over time) from abnormal dynamics. It can eventually be applied in an on-line fashion. The algorithm is available in open source, coded in R⁴.

4) Concept of time adapted to link streams: while studying the data of GITHUB online social network, we observe that the time series of graph statistic present day-night and weekly patterns due to the users activity. This phenomenon prevents us to observe the network's own dynamics and to detect related events. We then introduce the concept of *intrinsic time*, which provides new time units based on the appearance of links in the stream. This operation enables us to **generalize the use of Outskewer** to the precise observation of link stream dynamics. It especially allows detecting events

³Presented in Section 3.1.5.3

⁴<http://sheymann.github.io/outskewer/>

unseen before. However its impact on event detection is not trivial, and we study it using various parameters (like the size of the sliding time window).

5) Visual methodology of event validation: the previously detected events must be interpretable in order to be confirmed. We hence propose a **visual investigative method** on the sub-graphs that correspond to these events; it dramatically reduces the number of displayed items on screen. If abnormal patterns exist in the node-link diagram, and if they only appear at this particular moment, then we can reasonably interpret the events as correlated to these patterns, and use the associated meta-data to postulate an explanation. We successfully apply our methodology to the GITHUB dataset.

6) Exploratory framework: in the age of data intensive science, scientific contributions may have a scope not just covering one or two links in the data processing chain, but actually think through the integration in the whole chain [Tuk80]. This is not trivial as the different steps and their integration draw from different skill sets, usually cultivated in different academic disciplines. Our complete methodology combines automatic event detection with visual validation. We thereby propose an ***exploratory framework for link stream analysis*** which may be specified with regards to specific use cases, both in terms of the statistic used for event detection and of the kinds of event being validated. It is even possible to make this process entirely automatic once the events are characterized.

This manuscript is organized as follows. In the following Chapter we provide the definitions related to link streams, and present the datasets that will be used throughout the thesis. The four following chapters then detail the contributions mentioned above. In Chapter 3 we provide a state of the art on network visualization, then we study visual approaches for outlier detection in static networks, and event detection in dynamic networks. In Chapter 4 we propose *Outskewer*, an automatic method for outlier and event detection using the skewness of distribution of values, and we validate it experimentally on both synthetic and real-world data. In Chapter 5 we generalize the use of *Outskewer* on link streams by introducing the concept of *intrinsic time*. In Chapter 6 we illustrate our unified framework for automatic event detection and visual investigation. We apply it to the real-world dataset of the online social network GITHUB.

Definitions & Datasets

Contents

2.1	Definitions	17
2.2	Datasets	21
2.2.1	French Population Size in the 20 th Century	21
2.2.2	Republic of Letters	21
2.2.3	Radar of Internet	22
2.2.4	Twitter network of COFA Online	23
2.2.5	Github Online Social Network	23
2.2.6	eDonkey P2P Network	24

2.1 Definitions

A unifying formalism for time-ordered networks exists [KKK00]: it associates a departure time and an arrival time to each link, and model each link $e = (u, v)$ of the original graph by constructing two undirected links $e^- = (u, w_e)$ and $e^+ = (w_e, v)$, where w_e is a new node. This formalism allows computing time-respecting paths, however it implies unnecessary complications for link streams, and the time order of links is not explicit. It is thus difficult to express notions related to link streams with it. The formalism of data streams seems more suitable to link streams [AF07]: “a data stream is a possible infinite series of objects $\dots, obj_{t-2}, obj_{t-1}, obj_t, \dots$, where obj_t denotes the object observed at time t .” We thus specify it to link streams, and provide related definitions:

Definition 1 *Link stream*: let $F = \{f_0, f_1, \dots, f_m\}$ be the ordered multiset of triples $f_i = \langle n_x, n_y, t_i \rangle$, with the order relation $f_i \leq f_{i+1}$ if and only if $t_i \leq t_{i+1}$, where $i \in \mathbb{N}, t_i \in \mathbb{R}$. We call F a link stream. Each triple denotes a link between two nodes n_x, n_y observed at the date t_i and at the position i in F .

Definition 2 *Link stream graph*: let $G = (V, F)$ be the graph corresponding to such a link stream, with $V = \{n_0, \dots, n_n\}$ and $F \subseteq V \times V \times T$ where $T = \{t_0, \dots, t_m\}$.

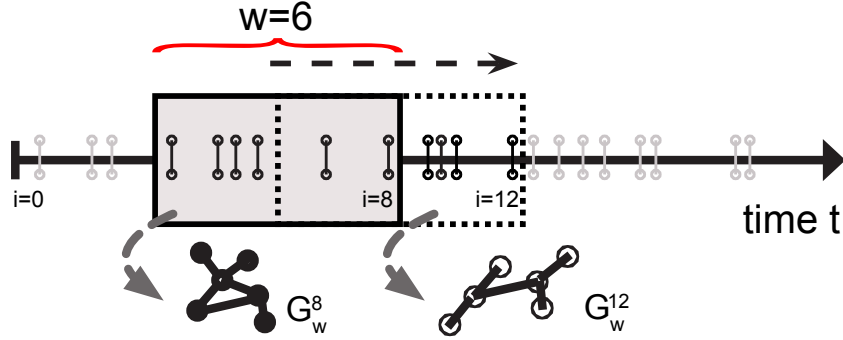


Figure 2.1: Illustration of a sliding time window (of size w) on a link stream. A sub-graph G_w^i is associated to each position i of the window.

We note that G has n nodes and m links. It is a dynamic graph because each link f_i has a time of appearance t_i .

Definition 3 *Link stream sub-graph:* $G_w^i = (V_w^i, F_w^i)$ is the corresponding sub-graph of the multiset of w triples $F_w^i = \{f_{i-w+1}, \dots, f_i\}$.

We note that sub-graphs have w links, see Figure 2.1.

Definition 4 *Statistic on a link stream sub-graph:* let S_w^i be a statistic of $G_w^i, \forall i \in [w-1, m]$ and $i \in \mathbb{N}$. The time series $S_w = \{S_w^{w-1}, \dots, S_w^m\}$ thereby represents the evolution of S_w^i over time, computed on each sub-graph of F with a sliding time window of size w .

Definition 5 *Outlier:* there is no formal definition of outliers because this intuitive notion varies with the context and the expected properties of outliers. From a statistical perspective, Grubbs [Gru69] states that “an outlying observation, or outlier, is one that deviates markedly from other members of the sample in which it occurs”. Hawkins [Haw80] defines an outlier as “an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism”, while Barnett and Lewis [BL94] call an outlier “an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data”.

Definition 6 *Event:* consecutive set of values $\{x_i, x_{i+1}, \dots, x_j\}, i+1 \leq j$ classified as outlier in a time series $X = \{x_0, x_1, \dots, x_n\}$.

For convenience, we refer to the events using their position i in X .

We now extend our definitions to the important notion of **bipartite graph**. Interaction systems may indeed be represented as bipartite graphs when interactions occur between two types of nodes. Also called *two-mode networks*, bipartite graphs are made of nodes (i.e. elements) which belong to two sets

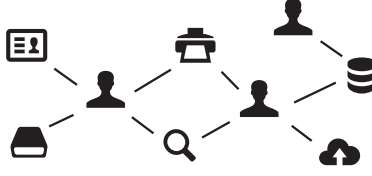
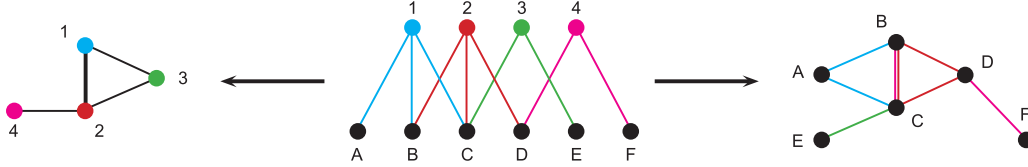


Figure 2.2: Example of user-services interactions.

Figure 2.3: Example of bipartite graph (center), together with its \top -projection (left) and its \perp -projection (right).

usually called *top* and *bottom*, and in which links exist only between nodes of different sets. Applications in computer science include client-server architectures where machines connected as clients make use of resources provided by machines connected as servers, and file-provider graphs where each file is connected to the individuals providing it, e.g. in peer-to-peer architectures. The invocations of various services of an information system by a set of users typically correspond to such a bipartite graph, as illustrated in Figure 2.2.

Definition 7 *Bipartite graph*: let $G = (\top, \perp, E)$ a triplet where \top is the set of top nodes, \perp is the set of bottom nodes, and $E \subseteq \top \times \perp$ is the set of links.

Bipartite graphs do not form a particular type of networks as one could think. On the contrary, all complex networks have an underlying bipartite structure [GL04]. Graphs which do not display a bipartite structure are indeed projections of bipartite graphs. Projecting bipartite graphs on the set of top or bottom nodes is a classical approach for studying such graphs, despite several drawbacks [LMDV08]. For instance, one can build the graph of clients which is a projection of the bipartite client-server graph, where two clients are linked together if they use the same server.

Definition 8 *Projection of bipartite graphs*: as defined in [LMDV08], the \perp -projection of G is the graph $G_{\perp} = (\perp, E_{\perp})$ in which two nodes of \perp are linked together if they have at least one neighbor in common in G : $E_{\perp} = \{(u, v), \exists x \in \top : (u, x) \in E \text{ and } (v, x) \in E\}$. The \top -projection G_{\top} is defined dually, as shown in Figure 2.3.

We also describe the notion of *internal links* that we will use in Section 5.3. The intrinsic bipartite notion of internal links has been introduced

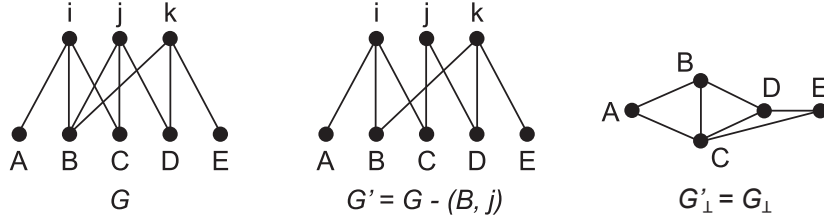


Figure 2.4: Example of \perp -internal link. Left to right: a bipartite graph G , the bipartite graph G' obtained by removing link (B, j) from G , and the \perp -projection of them. As $G'_\perp = G_\perp$, (B, j) is a \perp -internal link of G .

recently and studied for static networks [ATML12] to bring novel insights on the characterization of these networks. An internal link is such that its removal does not change the projection of the graph for a given set of nodes, either *top* or *bottom*. In the example of an information system of users (i.e. *top* nodes) interacting with services (i.e. *bottom* nodes), an internal link from the *top* (resp. *bottom*) point of view is a link which is not mandatory to connect two users (resp. services) in the corresponding projection. One can interpret internal links as a measure of links redundancy.

Definition 9 Internal link: \top -internal (resp. \perp -internal) links are links which may be removed from E without altering the \top -projection (resp. \perp -projection), as shown in Figure 2.4. Let $(u, v) \in \perp \times \top$ with $(u, v) \in E$ and let $G' = G - (u, v)$, (u, v) is a \perp -internal link if and only if $G_\perp = G'_\perp$ where G'_\perp is the \perp -projection of G' . \top -internal links are defined dually.

We naturally extend the definitions of link stream and link stream graph as follows:

Definition 10 Bipartite link stream: let $E = \{e_0, e_1, \dots, e_m\}$ be the ordered multiset of triples $e_i = \langle n_\top, n_\perp, t_i \rangle$, with the order relation $e_i \leq e_{i+1}$ if and only if $t_i \leq t_{i+1}$, where $i \in \mathbb{N}, t_i \in \mathbb{R}$. We call E a bipartite link stream. Each triple denotes a link between two nodes n_\top, n_\perp observed at the date t_i and at the position i in E , where $n_\top \in \top$ and $n_\perp \in \perp$.

Definition 11 Link stream bipartite graph: let $B = (\top, \perp, E)$ be the bipartite graph corresponding to such a link stream, where \top is the set of top nodes, \perp is the set of bottom nodes, and $E \subseteq \top \times \perp \times T$ where $T = \{t_0, \dots, t_m\}$.

The definitions of link stream sub-graph and associated statistics can be extended in a similar way.

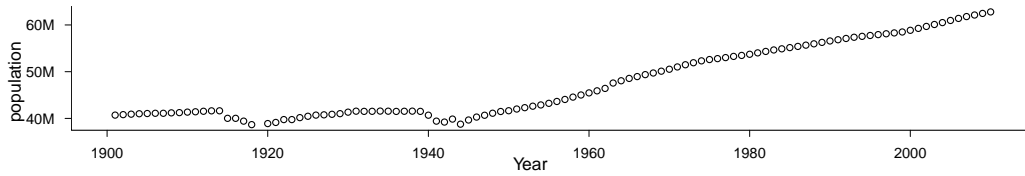


Figure 2.5: Number of inhabitants per year in France during the 20th century.

2.2 Datasets

We have used various datasets to test each step of our methodology. The experiments related to visualization have been conducted in the context of various research projects. The visualization experiments are conducted with the Twitter network of COFA Online and the dataset of Mapping the Republic of Letters project. The automatic detection of outliers in samples and time series (Chapter 4) has required the use of three different datasets: the evolution of the French population size in the 20th century, Radar of Internet, and search queries from the eDonkey P2P network. The link streams cover a wide range of volume, from a hundred links to hundreds of millions. We have finally performed an extensive study on one dataset (GITHUB online social network), combining all the techniques developed during our research into a unified methodology. This is not the largest dataset in terms of number of links, but the most appropriate thanks to the richness of its meta-data and the live system (Github.com) from which the dataset was extracted.

2.2.1 French Population Size in the 20th Century

This dataset is a time series of the evolution of the French population during the 20th century¹, see Figure 2.5. It is the estimation of the population size for each year, from 1900 to 2010. The dataset contains 111 data points. Experiments conducted on this dataset are described in Section 4.2.4.1.

2.2.2 Republic of Letters

This dataset is a static network of 60,000 letters and relationships among philosophers during the Enlightenment. The Mapping the Republic of Letters initiative (MRofL)² is a digital humanities project based at Stanford University exploring intellectual exchange in the early modern period through the analysis of correspondence, travel and intellectual network data. Bringing together an international network of scholars, researchers, graduate students, collaborators, and partners through several research and learning initiatives,

¹https://fr.wikipedia.org/wiki/Histoire_d%C3%A9mographique_de_la_France

²<http://republicofletters.stanford.edu>

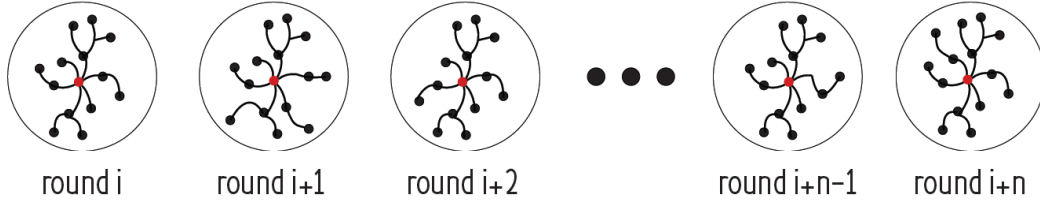


Figure 2.6: Radar measurements are iterated from a monitor to a set of destinations [HLM10].

the project is a collection of case studies in history and literary studies that makes use of information visualization to examine the scope and the dimension of heterogeneous datasets. In the last four years, the MRofL research group has engaged with several research labs and organizations from the design and computer science communities, in order to better understand roles, opportunities and synergies between these fields, in the development of digital humanities tools.

The data coming from MRofL, as from many other humanities projects, present a high level of uncertainty and incompleteness for a number of reasons, such as the nature of the data itself (roughly 60,000 letters from the 17th century using various sources), the process of acquisition and digitization (e.g. letters are handwritten making it difficult, if not impossible, to recognize and process the content) and the heterogeneity of different sources (e.g. each data collection provides different content and metadata). Experiments conducted on this dataset are described in Section 3.2.2.

2.2.3 Radar of Internet

This dataset is a series of 4993 graph snapshots, with 10,000 nodes per snapshot on average. It was collected with a method called the Radar of Internet [LMO11], which observes the dynamics of the internet's topology at the scale of a few minutes. It consists in focusing on the part of the internet's topology viewed from a single computer called the *monitor*, when IP packets are sent using the *tracetree* tool to a set of random targets. Periodical measurements of this map, called *ego-centered view*, have been performed every 15 minutes during several months, leading to a series of graphs, see Figure 2.6. The obtained map is obtained by merging all the links discovered between two machines during a round of measurement.

The dataset hence consists in a series of 4993 static graphs captured from a monitor of the PlanetLab project³ in Japan. The dataset is freely available⁴. Experiments conducted on this dataset are described in Section 4.2.4.2.

³<http://www.planet-lab.org/>

⁴<http://data.complexnetworks.fr/Radar/>

2.2.4 Twitter network of COFA Online

This dataset is a stream of 123 links. COFA Online Gateway⁵ is an Australian platform for teaching e-learning methods and techniques. The tweets talking about it have been manually collected using the Twitter Timeline and search engine from October 26, 2011 to January 11, 2012, and 508 users have been geolocated manually. We have built the network of these users, in which a link exists when a user mentions another user in a tweet during this period by either “retweeting” a message or mentioning another user without Twitter built-in features (e.g. when missing “@” at the beginning of a user name). Users appear the first time they tweet about COFA Online, or the first time they are mentioned. This network can be modeled as a link stream, as we observe appearing links at each interaction between Twitter users talking about COFA Online. The dataset is available on demand. Experiments conducted on this dataset are described in Section 3.2.1.

2.2.5 Github Online Social Network

This dataset is a stream of 2.2 million links. Github.com is an online platform created in 2008 to help developers share open source code and collaborate. Built on the Git decentralized versioning system, it facilitates contributions and discussions by providing a Web interface (see Figure 2.7). Github reached 3 million users on January 16, 2013, who collaborate on 5 million source code repositories. Our dataset describes the complete activity between users and repositories on the platform from March 11, 2012 to July 18, 2012. We have extracted the data from the Github Archive⁶, which is a record of every public event on Github. Then we have built the graph of “who contributes to which repository”, where nodes represent users and repositories, and where links represent any kind of activity users have on repositories.

Github data is modeled as a bipartite graph of link stream $G = (\mathbb{T}, \perp, E)$ where \mathbb{T} nodes represent users, \perp nodes represent repositories, and $E \subseteq \mathbb{T} \times \perp \times T$, where T is a set of time-stamped values, represents an activity between a user and a repository⁷.

The considered activities are the following: commit and push source code, open and close issues for bug reports, comment on issues, commits or pull request (i.e. asking for a patch to be merged), create or delete branches and tags, and edit the repository wiki. We ignore the other activities: fork (i.e. repository duplication), mark repositories as favorite, and follow of the timeline of another user or repository. There are slightly more than 336 000 nodes and 2.2 million links in the graph.

⁵<http://online.cofa.unsw.edu.au/>

⁶<http://www.githubarchive.org>

⁷Definitions related to bipartite graphs may be found in Section 2.1

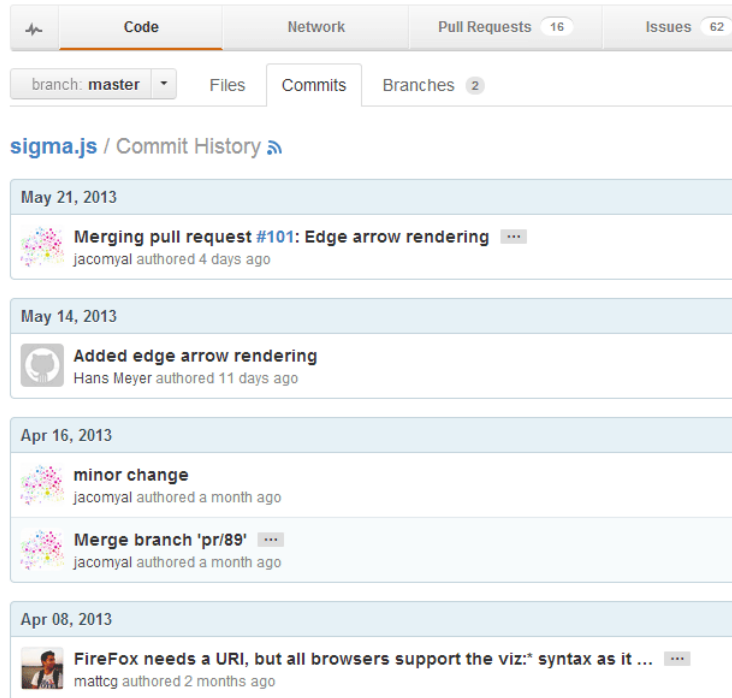


Figure 2.7: Screenshot from Github.com showing a history of contributions to the source code of a project.

We have collected all data necessary to monitor the evolution of the graph, as we have stored all nodes and links over time. Each link is associated with a timestamp indicating the moment when it has been observed. The data is thus a stream of observed links, ordered by their timestamp. A node is considered to appear in the graph when it is attached to an observed link for the first time. However there is no information in the data about the duration of nodes and links existence. A node may indeed be observed only once even if it exists during a long period. It means that we do not observe the nodes which appeared before the beginning of the measurement and for which no link is observed during the measurement, i.e. who do not contribute or for which there is no activity during the studied period. We thus miss the registered users who are not active in the social network during the measurement, and we also miss the existing repositories on which there is no activity. Experiments conducted on this dataset are described in Sections 5.2, 5.3, and 6.3.

2.2.6 eDonkey P2P Network

This dataset is a stream of 205 millions links. It consists in search queries captured from a eDonkey server for 28 weeks in 2009 [LMF]. The dataset contains textual queries made by users for lists of files matching certain key-

words. The dataset contains 205,228,820 queries entered from 24,413,195 IP addresses. Samples and procedure descriptions are publicly available⁸.

The data is stored as a link stream: a link exists when a user requests a file from a P2P server. A timestamp is associated to each link, and the links are ordered by timestamp. Experiments conducted on this dataset are described in Section 4.2.4.3.

We have finished to introduce the definitions and datasets used in our thesis. We propose in the following Chapter our experiments in visualization for outlier and event detection.

⁸<http://antipaedo.lip6.fr/>

CHAPTER 3

Visual Event Detection

Contents

3.1	Related Work	27
3.1.1	Perceptual Support of Visualization	29
3.1.2	Emergence of Knowledge through Visualization	31
3.1.3	Visual Representation of Networks	33
3.1.4	Interaction	39
3.1.5	Global Approach for Network Visualization	40
3.1.6	Local Approach for Network Visualization	49
3.1.7	Visual Outlier Detection in Static Networks	53
3.1.8	Visual Event Detection in Temporal Networks	56
3.2	Our Experiments	64
3.2.1	Visual Events with the Global Approach	64
3.2.2	Interest Points on Static Networks with the Local Approach	65
3.3	Conclusion	67

Visualization is an intuitive solution to explore networks but is far from trivial. In this Chapter we distinguish the global approach, which is so far the most common, to the local approach which has received a more recent attention. These two approaches combine different interactions techniques. We have experimented them for outlier detection in static networks, and for event detection in temporal networks. The knowledge gained from these experiments has provided ground to the rest of our research.

We review the state of the art in Section 3.1 and describe our experiments in Section 3.2, to conclude in Section 3.3.

3.1 Related Work

In this Section we review existing visualization methods for outlier detection in networks, and for event detection in dynamic networks. We also distinguish

the global approach, in which the complete network is displayed, and the local approach, where only a sub-graph is displayed and possibly modified interactively.

Information visualization has been used to support social network analysis since the 1930s with the “sociogram” of J. Moreno [Mor37], which is a graphic representation of social ties among a group of people. Despite the early beginning of network visuals, we had to wait until the 1990s and the democratization of computer graphics to see the development of interactive visualization software, which has made the interactive exploration of complex networks possible. Pajek [BM98] is the most noticeable tool, as it provides both statistical algorithms and visual representations of social networks. Its methodological book entitled “Exploratory Social Network Analysis with Pajek” was published in 2005.

The contribution of Information Visualization to science is stated by J-D. Fekete [FVWSN08]:

“Information Visualization is meant at generating new insights and ideas that are the seeds of theories by using human perception as a very fast filter: if vision perceives some pattern, there might be a pattern in the data that reveals a structure. [...] Therefore, it plays a special role in the sciences as an insight generating method.”

More generally, Information Visualization is a way to reveal data properties which would not be trivially detected otherwise, to shed light on breakthroughs, and to share the poignant experience of “Aha, I see!” [Few06] thanks to its intuitive aspect. This research field contributes to the emergence of novel scientific theories by improving the exploitation of human cognition. According to Card, Mackinlay and Shneiderman [CMS99], the main focus of visualization is indeed to amplify cognition. The authors listed a number of key ways to do so, showing the advantages of using visualization techniques during data exploration:

- Reducing time spent looking for information,
- Enhancing the recognition of patterns,
- Enabling perceptual inference operations,
- Using perceptual attention mechanisms for monitoring tasks,
- Encoding information in an actionable medium.



property	marks	ordinal/nominal mapping	quantitative mapping
shape	glyph	○ □ + △ S U	
size	rectangle, circle, glyph, text	● ● ● ●	● ● ● ● ● ● ● ● ● ●
orientation	rectangle, line, text	— / \ \ /	— / / / / / / / /
color	rectangle, circle, line, glyph, y-bar, x-bar, text, gantt bar		

Figure 3.1: Example of guidelines for mapping data variables to visual variables [STH02].

3.1.1 Perceptual Support of Visualization

Information Visualization relies on the properties and perception abilities of the human visual system. According to Information Theory, vision is the sense that has the largest bandwidth (100 Mbits/s), which makes it the best suited canal to convey information to the brain (in contrast, audition has only around 100 bits/s) [War00]. Visualization hence requires building and applying a visual language to encode information that can be read and interpreted correctly. This operation is called a mapping between data variables and visual variables. This language relies on visual features like geometric primitives, colors and sizes, and was theorized in [BB67, CM84], and extended in [Mac86]. However selecting visual features to convey information is not trivial. One would indeed like to select the most effective ones, but while avoiding misunderstandings and over-interpretations. Well-established guidelines distinct two kinds of data variables: quantitative and qualitative variables (see Figure 3.1). Visual features can be selected according to the type of data, but difficulties remain when mixing different visual variables in the same image.

Two main psychological theories explain how vision can be used efficiently to perceive features and shapes, according to [War00]: the preattentive processing theory, and the Gestalt theory.

Some visual features are particularly efficient as demonstrated in [Tre85, HBE95], an effect called preattentive processing. Visual saliences (i.e. elements and patterns which perceptually stand out from the remainder of the picture and grab the attention of the observer [IK01]) can be perceived very quickly (in an order of less than 250 milliseconds) and can be recognized "at a glance" without any cognitive effort, even if it has been found that the attention level plays a critical role. An example is illustrated in Figure 3.2, where we spot the red letters among several dark letters (left image), as well as we spot the T among lines very quickly (right image). But mixing colors and shapes forces us to pay a specific attention to each item, see Figure 3.3.

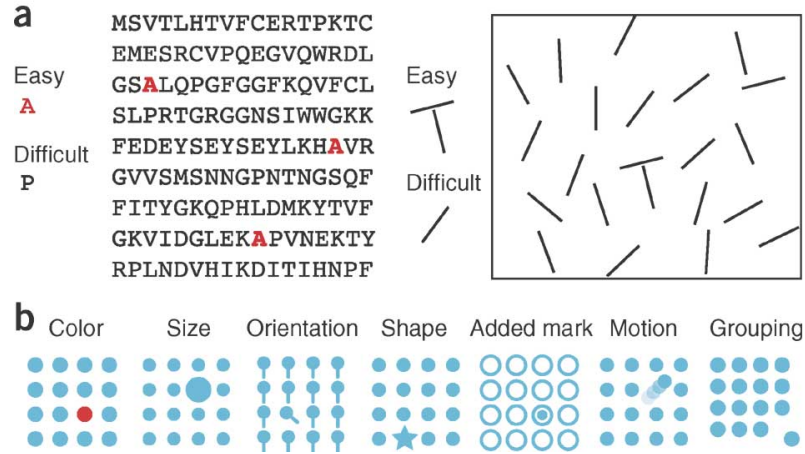


Figure 3.2: Illustration of the impact of preattentive processing on the detection of outlying elements [Won10]. (a) Certain elements can be seen in a single glance, whereas others are difficult to find. (b) Examples of visual features that make objects distinct.

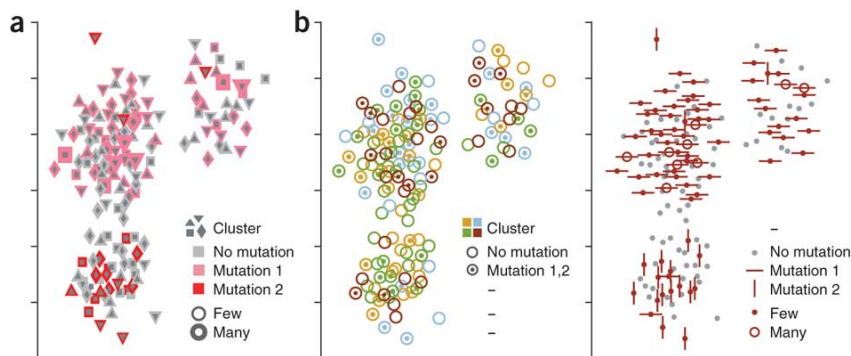






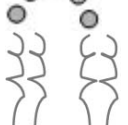
Figure 3.3: Illustration of the impact of mixing visual features on the preattentive processing effect [Won10]: (a) Simultaneous use of many graphical features can impede visual assembly of the data. (b) Multiple views of the same data with limited parameters plotted can better communicate specific relationships.

The Gestalt theory, established in [Kof35], explains the main principles that lead to images interpretation. [War00] summarizes them as follows:

- **Proximity:** Things that are close together are perceptually grouped together;
- **Similarity:** Similar elements tend to be grouped together;
- **Continuity:** Visual elements that are smoothly connected or continuous tend to be grouped;
- **Symmetry:** Two symmetrically arranged visual elements are more likely to be perceived as a whole;
- **Closure:** A closed contour tends to be seen as an object;
- **Relative Size:** Smaller components of a pattern tend to be perceived as objects whereas large ones as a background.”

We illustrate them in Table 3.1.

Table 3.1: Interactions among structures, from the Gestalt theory.

Laws of Grouping	Structure	Perception	Illustration
Proximity	2 close components	1 single component	
Similarity	Similar components	Grouped components	
Closure	Close boundaries	Unified boundaries	
Continuity	Neighboring items	Grouped items	
Symmetry	Symmetrical items	Global item	

3.1.2 Emergence of Knowledge through Visualization

The goal of Exploratory Data Analysis is to find the best hypothesis which supports the observation of data. The knowledge discovery process is thus considered to be abductive, i.e. given an observation, our explanation has a

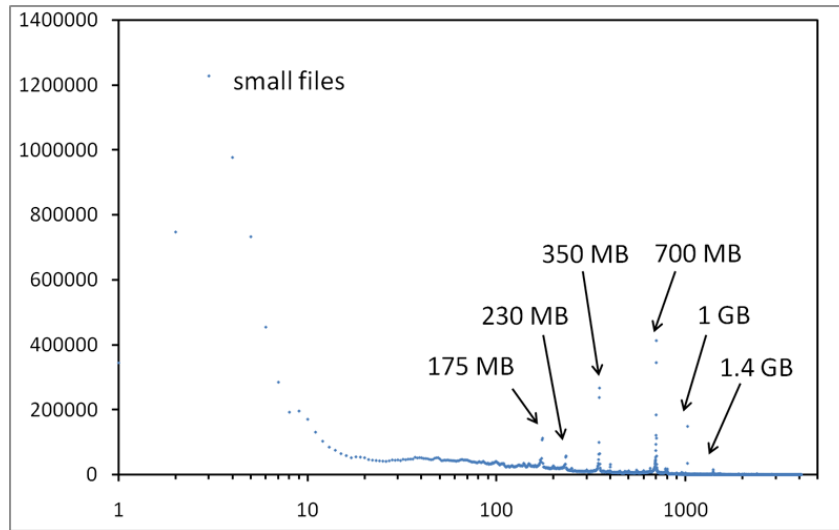


Figure 3.4: Distribution of file sizes in a P2P file exchange system, with the number of files as a function of file size [From Complexnetworks.fr].

reasonably good chance to be right according to our current results, knowledge, and intuition, but there might be an unknown number of explanations that can be at least as good as this one. Further studies through visualization and statistical analysis are then necessary to try disproving our explanation in favor of a better one. The explanation may finally be accepted after a couple of experiments that fail at invalidating it. The insights gained may be used to confirm already known results, as well as providing ideas of novel statistical indicators and data descriptors in general.

The data properties spotted by visual saliences may challenge current hypotheses and raise new questions. The analyst may want to modify the visualization accordingly, to eventually select a picture which clearly reveals an issue, or which supports a hypothesis. The key role of visualization in the emergence of knowledge is emphasized by J. Tukey [Tuk77]:

“The greatest value of a picture is when it forces us to notice what we never expected to see.”

We illustrate it on a simple example: in the distribution of file sizes in a P2P system (see Figure 3.4), we observe clear peaks on specific values, and we know that these values correspond to the most common sizes of films, depending on their formats. These values are thus interesting outliers, not anomalies in data. The authors of the study raise then the following hypothesis:

“Even though in principle files exchanged in P2P systems may have any size, their actual sizes are strongly related to the space capacity of classical exchange and storage supports.”

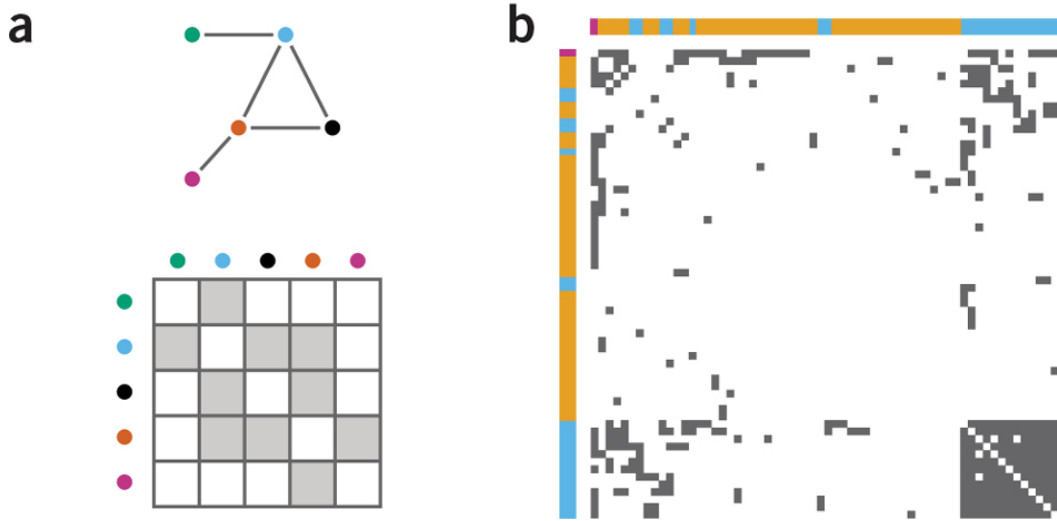


Figure 3.5: (a) Nodes are ordered as rows and columns; connections are indicated as filled cells. (b) A matrix representation of data from Figure 3.6 (b) [GW12].

The visual investigation of this P2P dataset helped the authors of the study to make a discovery, which however has to be confirmed by complementary analyses.

3.1.3 Visual Representation of Networks

What makes complex network data particular is the key importance of relationships. Observing and navigating in this context calls for the development of suitable visualization and interaction techniques in conjunction with storage and data mining solutions. Complex networks have therefore received a large attention from Information Visualization researchers, which has led to multiple methods and techniques for their representation and exploration. Usually, representations of networks are projections of the topology on two or three dimensional spaces using algorithms that calculate nodes coordinates. These algorithms are called layouts. We present here two classical representations: matrix-based representations, and representations with dots and lines on which we will more specifically focus in this chapter.

3.1.3.1 Matrix-Based Representations

Introduced in [BB67], matrix-based representations rely on the adjacency matrix, i.e. a Boolean matrix whose rows and columns represent the nodes of the network. For each link between two nodes, the cell at the intersection of the corresponding row and column contains the value “true”, see Fig. 3.

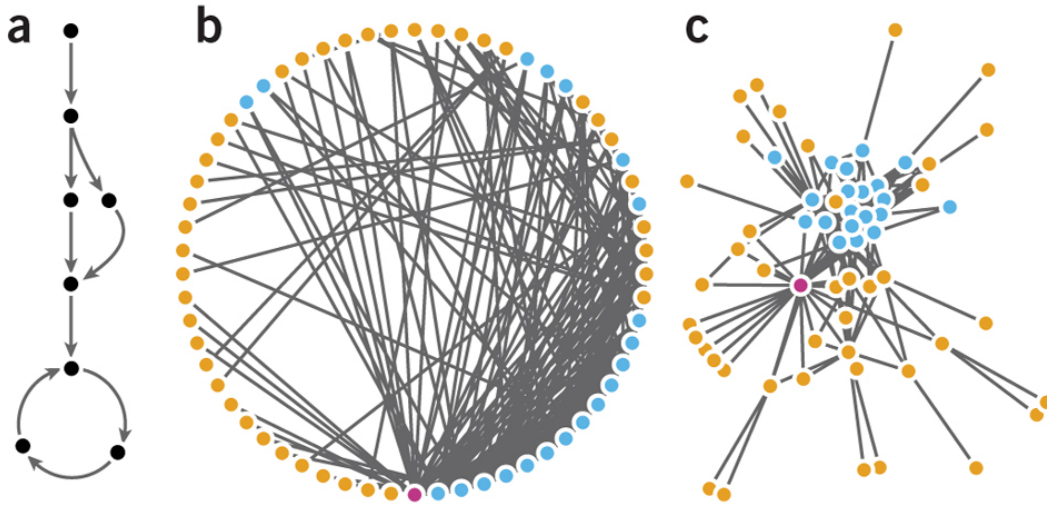


Figure 3.6: (a) A directed graph typical of a biological pathway. (b) An undirected graph with nodes arranged in a circle. (c) A spring-embedded layout of data from b [GW12].

Otherwise, it is set to “false”. It is possible to replace the Boolean values by those links’ attributes to add more information to the representation.

Matrix-based representations can be “reordered” through successive permutations of its rows and columns to reveal interesting patterns in the network structure. One of the main advantages of this representation is to avoid occlusion problems encountered using the representation with dots and lines, which we will see in the next Section. Matrices are efficient to perform basic tasks like identifying the most connected node, a link between two nodes, or a common neighbor of two nodes. However they perform poorly on more complex tasks such as finding a path between two nodes, even in small matrices [GFC04]. Such drawbacks may be the reason why they remain underused compared to representations with dots and lines.

3.1.3.2 Representations with Dots and Lines

These representations rely on “graph drawing”, which is the art and science of making this type of representation, also known as node-link diagrams, using layout algorithms. These diagrams represent nodes as dots and links as line segments (or curves). A significant majority of network visualization software implement such representations: in 2007 [HFM07] referenced 54 (out of 55) node-link based systems in the Social Network Analysis Repository, and 49 (out of 52) on the Visual Complexity website.

Force-directed algorithms are the most common layouts. They are usually described as spring embedders [Kob12] due to the way the forces are computed: roughly speaking, connected nodes tend to be closer, while discon-

nected nodes tend to be more distant. More precisely, force-directed layouts compute repulsive forces between all nodes, but also attractive forces among linked nodes. Forces are calculated and applied on each node at each layout iteration to update its position until the algorithm has converged to a stable position of nodes.

All force-directed algorithms rely on a formula for the attraction force and one another for the repulsion force. The “spring-electric” layout proposed in [Ead84], for instance, is a simulation inspired by real life. It uses the repulsion formula of electrically charged particles ($Fr = k/d^2$) and the attraction formula of springs ($Fa = -k.d$) involving the geometric distance d between two nodes. The pseudo-code is given as follows:

```
algorithm SPRING(G:graph):
  place vertices of G in random locations
  repeat M times:
    calculate the force on each vertex
    move the vertex  $c_4 * (\text{force on vertex})$ 
  draw graph
```

Fruchterman and Rheingold [FR91] have created an efficient algorithm using different forces (attraction $Fa = d^2/k$ and repulsion $Fr = -k^2/d$, with k adjusting the scaling of the network).

Moreover, recent software like Gephi (introduced in Chapter 3) draw the visualization at each iteration, thus providing real-time feedback to users. When layouts are implemented with no stopping condition, users can tweak the layout parameters in real-time until they decide to stop its execution. Interaction while calculating layout is usually made technically possible by using multi-threading processing, and by using the GPU for rendering the visualization. The goal is to avoid the layout algorithm being perceived as a “black box” by the analyst (although no scientific study has been performed yet to verify this belief), and to accelerate the testing of the layout parameters to obtain an aesthetically good visualization.

The targeted visualization of force-directed layouts is a rough correspondence between the distances in the projection space and the distances in the network topology. The goal is to enable a visual interpretation of the topology using the spatial positions of nodes. When a “good” layout is applied, the resulting image hastens the understanding of the network structure by revealing visual patterns. The readability of graphical representations can be defined by the relative ease with which users find the information they are looking for. Alternative definitions include the potential to make sense of the data, the familiarity to users, and aesthetic criteria; readability is subjective because the result should be visually appealing and depend on the analysis task. However, some metrics are available to compare layouts, such as the

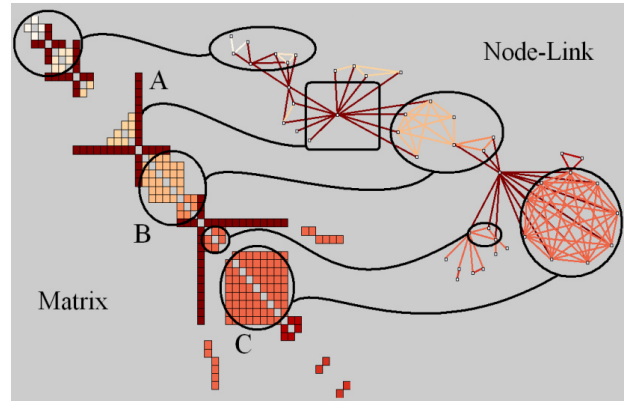


Figure 3.7: Visual patterns in Matrix and Node-link representations of social networks. A represents an actor connecting several communities, B a community and C a clique (complete sub-graph) [HFM07].

number of occlusions, the uniformity of link lengths, and the number of link crossings. A more detailed introduction to this topic can be found in [Tam07].

Other kinds of representation exist, but the readers should be able to cover most of their needs using matrix-based representations and node-link diagrams. As choosing a representation may also depend on the analysis task to perform, [HFM07] provides the following comparison guide, see Table 3.2. The correspondence between some visual patterns is illustrated in Figure 3.7.

Table 3.2: Pros and cons of matrix and node-link diagrams [HFM07].

	Matrix-based representations	Node-link diagrams
+	No node overlapping	Intuitive
+	No edge crossing	Compact
+	Readable for dense graphs	More readable for path following
+	Fast navigation	
+	Fast manipulation	More effective for small graphs
+	More readable for some tasks	More effective for sparse graphs
-	Less intuitive	Useless without layout
-	Use more space	Node overlapping
-	Weak for path following tasks	Edge crossing
-		Not readable for dense graphs
-		Manipulation requires layout computation

3.1.3.3 A Visual Language of Node-Link Diagrams

The visual language of node-link diagrams helps to observe global patterns of connectivity (e.g. disconnected groups, structural holes, aggregates of nodes

called communities, bridges between communities, cores and peripheries), to spot the presence of unexpected connections and central nodes through visual saliences, and to study trivial correlations between topology and properties of nodes and links through visual features like color and size. When information is added to node-link diagrams, one generally uses at most five data variables: nodes, node labels, links, a qualitative attribute, and a quantitative attribute. These data variables are usually mapped to visual variables in Table 3.3.

Table 3.3: Usual mapping between data variables and visual variables in node-link diagrams.

Data variable	Visual variable
Node	Dot
Node label	Text near the corresponding dot
Link	Line segments (or curves)
A qualitative attribute	Dot colors
A quantitative attribute	Dot size

When no qualitative attribute is available, the quantitative attribute can be mapped to dot color as well. Alternatively, one may encode information in the dot border (size and color), and in the node label (size and color). Figure 3.8 is a sample visualization of the network of sales representatives in a private firm. The legend is necessary to explain the visual language used, thus allowing the reading and interpretation of the underlying data.

Despite its wide usage among researchers, node-link diagrams are not subject to well-established graphical conventions like those found in geographical maps. One can easily misunderstand them, so the visualization should come with a cautionary text in the legend, stating that:

- Distances are not absolute but relative to local connections. In consequence, one should not compare two graphical distances.
- The representation may be rotated in every direction so the top, bottom, left and right positions have no particular meaning.
- Nodes at the center of the picture may not be central at all in the network.

Geographical conventions may nonetheless influence the design of node-link diagrams. When dealing with multiple data attributes, several authors (see [BKB05, KB09]) distinguish the visual topology made of dots and lines to other visual variables. Like for geographical maps, the topology is then considered as the “base map”, while other variables are added as layers of information. In such cases, these visualizations are called “network maps”.

Network of company X sales representatives

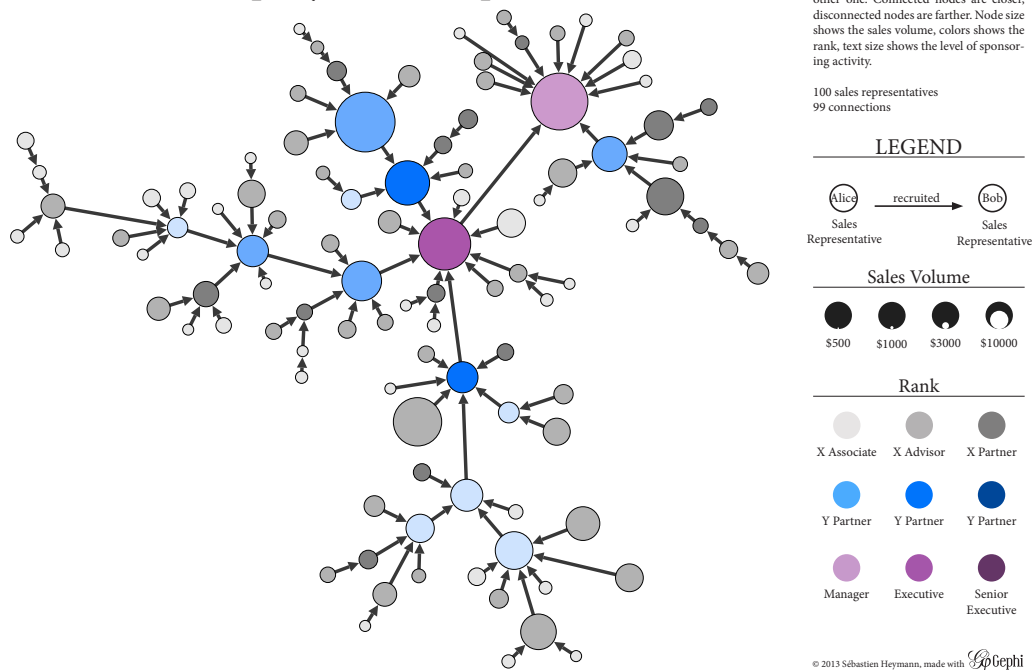


Figure 3.8: Visualization of a network sample representing which sales representative recruited which other in a company. Dot size corresponds to sales volume during the year. Dot color corresponds to the rank in the company. Private work from Sébastien Heymann, 2013.

Their comparison is facilitated because node and link coordinates are the same for all maps. This approach is remarkably used in scientometric studies (i.e. the study of science as a system), where maps of science represent the way scientific fields relate to each other through publications and co-authorship networks.

We have seen that the exploration of complex networks is greatly enhanced by visualization. However when dealing with large networks of hundred thousands of nodes and links, reading a static picture is difficult and provides limited insights due to the density of nodes and links. One may want to focus on a specific sub-graph, or to compare maps colored by different attributes, or to filter the network based on particular rules... Such tasks are supported by interactive features as we see in the next Section.

3.1.4 Interaction

Information Visualization is a research field of its own, but is only a part of a larger process to extract insights from data. A typical data exploration involves extracting, cleaning and sometimes merging various sources of data, then exploring data using various techniques, and finally rendering results for communication purposes. Visualizing data is embodied in these different stages:

1. One must look at the raw data to understand how to process it and to identify obvious errors like character encoding issues and exceptions such as missing data.
2. It is involved in the exploration process.
3. It can be used to communicate insights through static –final– renderings or dynamic –interactive– systems.

The quantity of information displayed by visual representations is naturally limited by properties of the medium, such as paper size and resolution of screen devices, i.e. the number of points that can be displayed in the two dimensions. When neither the size nor the resolution can be increased, a solution to overcome this issue on screen is to interact with the representation so that one can display information on demand. This approach helps to improve the readability of visualizations by reducing the quantity of displayed information at a given instant.

A set of interaction techniques using the mouse has become a standard: node selection on mouse click, node drag-and-drop to move its position, zoom and pan navigation features with the mouse wheel. These features are shared by noticeable software for the visual analysis of complex networks (introduced in Chapter 3), such as Cytoscape, Gephi, SocialAction and Tulip.

Moreover, advanced interaction techniques can enhance analysis tasks. For instance, Gephi proposes to follow the shortest path from a node to another by clicking on the source node and on the target node, then coloring links along the path. However, interaction techniques are bound to visual representations and are therefore difficult to generalize [AAB⁺12]. New technologies of Human-Computer Interfaces like multi-touch screen devices provide also new area of innovation [SNDC10].

Finally, interaction can be used not only to explore a dataset, but also to command the other steps of the processing chain. For example, one may filter the network according to a given query based on the properties of nodes and links, such as “display the nodes of degree greater than 10” [Ada06]. One may also acquire new data by interacting with the representation, as it is the case on visual Web crawlers: crawlers are programs which grab the content of Web pages by recursively visiting the hyperlinks of given Web pages. One can encode Web pages as nodes, and hyperlinks as links. The corresponding node-link diagram represents the Web explored by the crawler. One could then ask for the crawler to visit the hyperlinks of a Web page by double-clicking on its corresponding node. The crawler would therefore retrieve the new Web pages and scan the new hyperlinks available, to update the visualization.

Interaction techniques are therefore essential to explore large networks, to hasten analysis tasks, and to integrate visualization in the data processing chain (see Section 1.3).

3.1.5 Global Approach for Network Visualization

The global approach in the study of static networks consists in visualizing the whole network before possibly focusing on its parts. Such visualizations are called “synoptic views” or “overviews”, because they allow grasping the general properties of a complex system by seeing it entirely. For instance, a social science researcher may want to identify groups of individuals who interact frequently with one another, while a network architect may want to decompose network structures according to the paths taken by information going from one computer to another. This approach allows addressing different categories of questions such as the characterization of the global network topology, and the detection of outliers.

3.1.5.1 Guidelines

Most visual analyses studied in [Shn96] follow the same pattern of interaction with visual representations, which has led to Shneiderman’s well-established mantra of Visual Information Seeking:

“Overview first, zoom and filter, then details-on-demand”

These terms have been explained as follows:

Overview: get an overview of the entire data, for instance by zooming out the view.

Zoom: zoom in on items or groups of items of interest by controlling the zoom focus and the zoom factor. A good practice is to point to a location and trigger a zoom command.

Filter: filter out uninteresting items using dynamic queries through textual or widget-based interfaces (e.g. sliders, checkboxes and other buttons). A quick execution (less than 100 milliseconds) is desired.

Details-on-demand: get details on a selected item or group of items, usually by showing a pop-up window on click or by updating information on a sidebar.

This mantra should be considered as a recommendation which describes how data should be presented on screen [CC05]. It has been followed by numerous researches and implemented in the graphical user interfaces of notable software (see the tools in the following subsection), however it is surprising that few studies address the effectiveness of overviews (constructed artifacts) at “overviewing” [HH11], i.e. the ability to raise “qualitative awareness of one aspect of some data, preferably acquired rapidly and, even better, pre-attentively: that is, without cognitive effort” [Spe07]. Overviews are assumed to be an effective way to detect outliers in static networks, and events in dynamic networks because one can spot significantly different elements and groups among the whole through *visual saliencies* (see Appendix).

When applied to the study of complex networks, this mantra implies the creation of representations in 2-d or 3-d space to visualize the whole networks and interpret the data (see Section 3.1.3). In this perspective, quality representations are data projections which allow reading the network topology aided by a visual language. Grouping elements, filtering nodes and links, and using other interaction techniques (see Section 3.1.1) are keys to implement the mantra’s principles.

3.1.5.2 Tools

Since the release of Pajek [BM98], visualization and interaction features of scientific tools have been improved to support the global approach in a unified graphical user interface. We introduce the most noticeable ones below according to the number of times their original publication is cited by other research articles.

Pajek (1998): 1025 citations of [BM98]. Pajek is a closed source application which provides fast data mining algorithms for Social Network Analysis and node-link diagrams.

Cytoscape (2003): 3300 citations of [SMO⁺03]. Initially focused on visualizing molecular interaction networks, Cytoscape is an open source platform

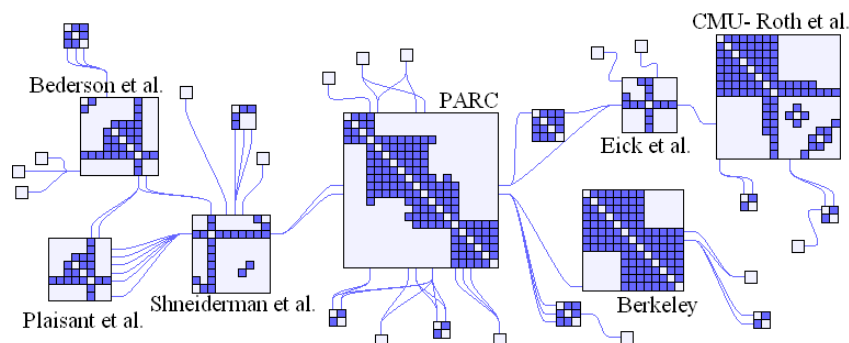


Figure 3.9: Visualization using NodeTriX.

suitable for any kind of networks. It combines a rich set of algorithms to create node-link diagrams with visual styles, filters and interaction tools. It is coded in Java and can be extended by plugins.

Tulip (2004): 240 citations of [Aub04]. Tulip is an information visualization framework dedicated to the analysis and visualization of relational data. The graphical user interface provides combined views using node-link diagrams and other kinds of representations like histograms and treemaps to support advanced analysis. It provides an open source library written in C++ to support the development of algorithms, visual encodings, interaction techniques, data models, and domain-specific visualizations. Tulip is particularly suitable for research prototyping of new kinds of visual representations and interaction techniques.

GUESS (2006): 171 citations of [Ada06]. This open source software enables the exploratory data analysis and visualization by combining node-link diagrams and a textual query language to edit data, filter networks and refine the representation.

SocialAction (2006): 159 citations of [PS06]. This closed source software integrates statistics and node-link diagrams in a step-by-step (yet flexible) process to get an overview, rank nodes and links according to their properties, and to find communities and outliers. A unique layout is maintained through the operations so users can make comparisons.

NodeTriX (2007): 176 citations of [HFM07]. When networks are globally sparse but locally dense, the global topology is readable using node-link diagrams but not the local groups of nodes. To solve that problem, NodeTriX provide a hybrid representation: node-link diagrams and matrices to visualize dense groups, see Figure 3.9. A set of interaction techniques based on direct manipulation of the nodes using drag-and-drop is available to smooth the exploration process.

Gephi (2009): 408 citations of [BHJ09]. Inspired by GUESS and SocialAction, Gephi is an open source software for the visual exploration of any kind

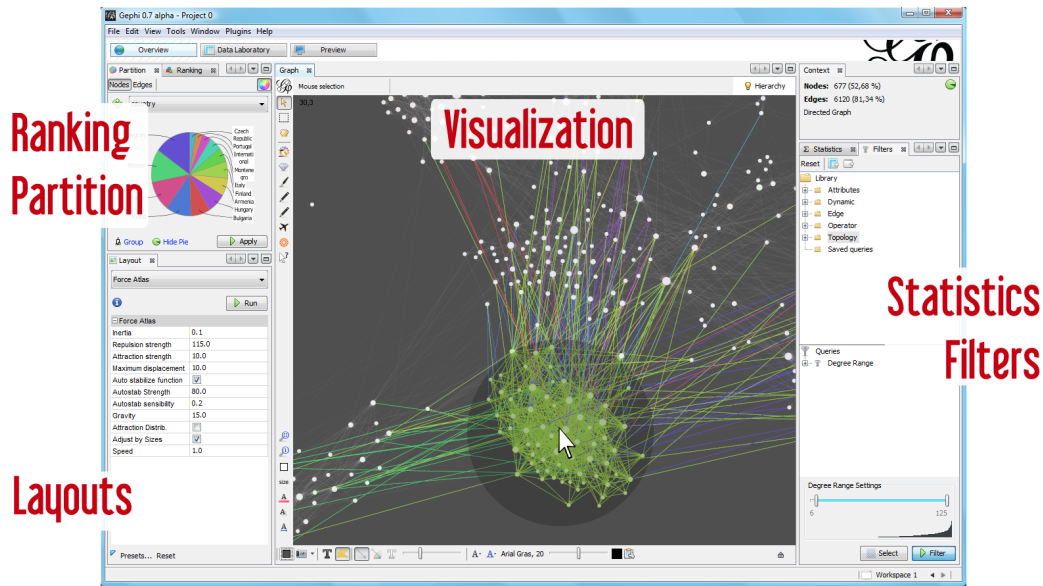


Figure 3.10: Overview of Gephi 0.8.

of networks. While various software exist to visualize and analyze networks, Gephi is particularly suited for networks with node properties like gender and age in social networks. Designed to facilitate the non-linear process of information discovery, it is focused on the visualization of the network using node-link diagrams, real-time interaction, and the use of a visual language. Gephi is coded in Java and can be extended by plugins.

Treeplus (2006) [LPP⁺06], **Vizster (2005)** [HB05], and the degree-of-interest approach proposed in [vHP09]. These approaches generally support the idea of starting with a small subgraph and expanding nodes to show their neighborhoods (and in the case of [vHP09], help identify useful neighborhoods to expand).

3.1.5.3 Special Notes on Gephi

I have co-founded Gephi 3.10, an open source software for the visual exploration of networks. As stated above, Gephi is particularly suited for networks with node properties. Properties are key-value pairs associated to each node or each link. For example, members of a social network may have attributes such as gender, language, and age.

Gephi software is generic. Any kind of network can be analyzed, like communication (e.g. email) and financial networks, online social networks (e.g. Twitter, Facebook), data center networks (i.e. connections between machines), document networks, among others.

Gephi has been designed to facilitate the non-linear processing of information discovery. In particular, it is focused on the visualization of the net-

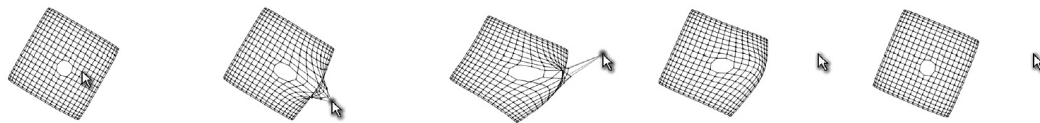


Figure 3.11: A sample network manually stretched and released while ForceAtlas2 is running [JHVB11].

work using *node-link diagrams* (in which nodes are represented by discs and links by lines), real-time interaction with data (e.g. node grouping, filtering, use of statistical results in the visualization), and the building of a visual language (the mapping of data variables to visual variables was theorized in [Ber83], [CM84]). This language makes use of colors and sizes to create informative visuals, which aim at being the network equivalent of geographical maps [BKB05].

A typical visual analysis with Gephi follows the well established mantra of Visual Information Seeking: “Overview First, Zoom and Filter, Details-on-Demand” [Shn96] detailed previously. The objective is to reveal visual saliencies of interest for the analyst, i.e. elements which perceptually stand out from the remainder of the elements and grab the attention of the observer [HBE96]. Such saliencies may challenge current hypotheses and raise new questions. The analyst then changes the visualization accordingly, to eventually select a picture which clearly reveals an issue, or which supports an hypothesis.

In Gephi, users interact with the visualization in real-time to position nodes in a two or three dimensional space using layout algorithms, or by manually moving nodes (see Figure 3.11). They use node properties to change their colors and sizes, in order to find groups and detect significant nodes (i.e. individuals in the case of social networks). The goal is to study the correlation of node properties and network structure by using visual patterns. Classic data mining algorithms of Social Network Analysis, such as the Louvain community detection algorithm [BGLL08], or the betweenness centrality measure [Bra01], can be computed at any time and their results integrated in the visualization through visual features. The network can also be filtered according to nodes and links properties.

The strengths of Gephi are its real-time visual feedback, performance, code modularity, and community of developers and users. The Gephi user interface is focused on the creation of network visuals in real-time. The key innovation is to ease the interaction with the network, as users can literally *play* with its visual representation. By playing, we mean experimenting various visual configurations to see the outcome of any action instantaneously, for instance by testing different force-directed layouts to shape the network structure. Such algorithms are usually described as spring embedders [Kob12] due to the way

the forces are computed. These layouts rely on a physical metaphor to position the nodes according to the position of the others. Roughly speaking, connected nodes tend to be closer, while disconnected nodes tend to be more distant. More precisely, they compute repulsive forces between all nodes, but also attractive forces among adjacent nodes. Each layout iteration calculates the forces applied on each node, and updates its position. The visualization is refreshed at each iteration, thus providing real-time feedback to users. Some layouts are implemented with no stopping condition. Users can thus tweak the layout parameters in real-time until they decide to stop its execution. Interacting with the visualization while calculating layout is made technically possible by using multi-threading processing, and by using the GPU for rendering the visualization.

These features enable the visual exploratory analysis of networks as explained in the Appendix. The approach of Exploratory Data Analysis [Tuk77] emphasizes the importance of curiosity and serendipity (i.e. discoveries made while searching for something else) to data analysis. The main benefit is to generate novel questions and research hypotheses. Gephi is used worldwide and supports a large user community; the last version (0.8.2-beta) has been downloaded 194,000 times¹ from January 1, 2013 to October 13, 2013.

Finally, Gephi is an ideal platform to implement dynamic network visualization features: [FQ11] states that despite the interest and advances from the visualization community in dynamic social networks, the number of available systems used by social network scientists that handle dynamic networks is still lagging behind the developments reported from visualization. The authors call for implementation of such mechanisms into comprehensive general purpose systems that are widely available, not only in prototype systems. We have worked on the addition of specific features for dynamic network analysis during this thesis, which can be used to detect events.

3.1.5.4 Problems with Very Large Datasets

Existing tools are faced with critical issues with very large networks.

Large Static Networks: The rapid increase of memory and processing resources, associated with improvements on the algorithms that generate visual patterns, has enabled us to process and display larger and larger networks on screen. However we face both cognitive and technical limits:

- Representations are cluttered when visual items overlap. This situation can be due to limited screen sizes, or to the use of layout algorithms that do not take dot size nor node labels into account.

¹<https://launchpad.net/gephi/+download>

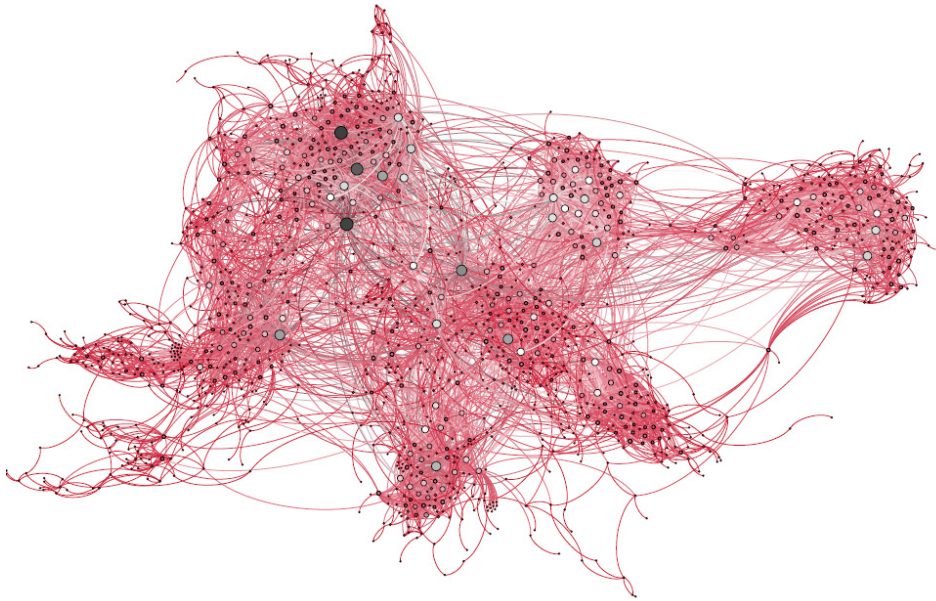


Figure 3.12: Node-link diagram of a few thousand node network, using the *ForceAtlas 2* layout.

- We may lose preattentive perception effects by mixing different visual features, which makes it difficult to combine various data variables in a single representation (see Section 3.1.1).
- The readability of graph layouts (see Section 3.1.3) may vary with the analyst's knowledge and with the performed tasks. Making layouts reasonably good for the largest number of situations is therefore challenging; so is the dissemination of graphical conventions.
- Some analytical tasks remain difficult with large numbers of displayed items, like following the path from one node to another.
- Real-time interaction is desired to facilitate trials and errors on data, and the processing chain should be flexible enough to handle various network structures like the evolution over time, but a tradeoff must be found between flexibility, development costs and performance when implementing data structures and algorithms.

The global approach has therefore many limits when displaying a large amount of data like in Figure 3.12. Increasing the size and resolution of computer screens is however not a solution, because it implies increasing the requirements of computing power, and this factor is not always controlled as people generally keep their computers for a couple of years. With more processing time required between every interaction with representations, the

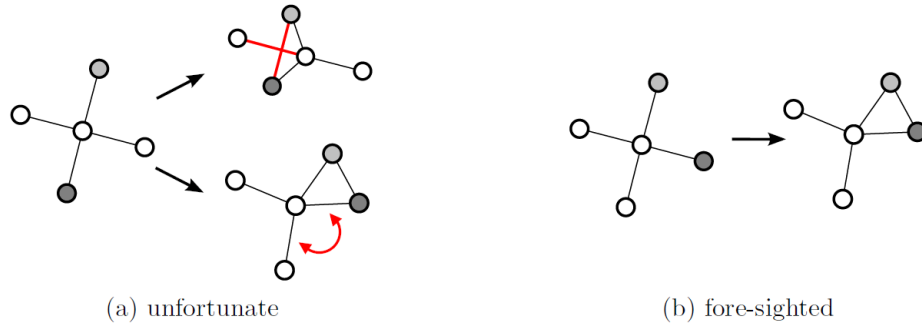


Figure 3.13: Knowledge of future changes can guide the choice between otherwise equally good layouts [BIM12].

process of visual exploratory analysis may become so painful that analysts may completely abandon visualization.

Temporal Networks: Traditional approaches use a time-to-time mapping and show the time-varying graph data as animated sequences of node-link diagrams. Although this visualization strategy is very intuitive it also has the following drawbacks:

- If the graph is very dense, i.e. contains many links, visual clutter occurs because of many link crossings.
- Animation requires cognitive efforts from a viewer to preserve his mental map.
- Sophisticated layout algorithms needed to circumvent these problems have a high run time complexity, see Figure 3.13.

Some approaches use a time-to-space mapping and show the stream of interactions over time entirely. Large networks are however unreadable, see Figure 3.14. A solution may be to aggregate data, but aggregation techniques are questionable when the exploratory task is unclear. Bias introduced by sampling methods should be completely understood to avoid misinterpretation of visual results. Network segmentation is also hardly applicable in case of small-world networks (where the average shortest path between nodes increases much slower than the number of nodes), or loosely speaking when the network topology is not a grid, which has been shown to be a shared property of many real-world networks encountered so far by researchers [WS98].

However, visualization of the whole network is not absolutely necessary. It is even sometimes not feasible nor desirable. Instead, one may look at a sub-part of the network with carefully chosen strategies to retrieve the sub-graphs of interest. We discuss the alternative to the global approach in the following Section.

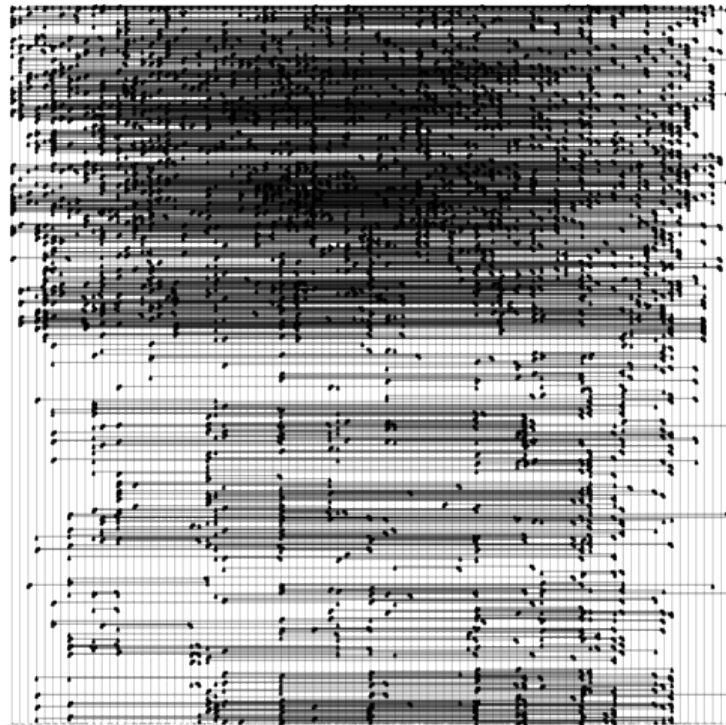


Figure 3.14: An interaction network visualized using the timeordered package for R. Time is on the vertical axis, nodes are listed on the horizontal axis [BD11].

3.1.6 Local Approach for Network Visualization

The global approach has become very popular as we have seen in the previous Section, but the observation of global patterns is not always relevant. This is the case in particular when we need to investigate a particular node and its connections (a task called “lookup”). More concretely, a local approach may be successfully performed in the following (non-exhaustive list of) activities:

- Data cleaning: scientists sometimes have to find and delete duplicate nodes due to measurement errors.
- Network monitoring: network administrators try to identify security holes after the detection of a suspicious pattern of activities from a visitor.
- Impact analysis: programmers need to understand the dependencies of a specific piece of code to prevent the impact of potential changes.

Moreover, many datasets such as the ones available from online social networks (e.g. Twitter, Facebook, Github) are simply too large to be fully displayed by average computers, as they are made of millions of nodes and links. Common graph databases like Neo4j², OrientDB³, DEX⁴ and TitanDB⁵ are designed to scale and hence allow the storage of dozens millions, even billions of nodes and links. However, even for much smaller networks, less powerful devices like tablets do not have the required resources to compute these overviews, and to interact smoothly with representations. When it comes to navigating in large networks, researchers [LPP⁺06, vHP09] (see Figure 3.15) and some commercial products such as Palantir⁶ or Linkurious⁷ propose solutions that deviate from the “Overview first, zoom and filter, details on demand” visualization strategy. This idea is not new in the context of graph drawing [ECH97, PvH12] but the recent availability of large networks makes local visualization an attractive approach to skirt the technical and cognitive burden of overviews.

3.1.6.1 Benefits

The local approach is an alternative capable of overcoming (to some extent) the limitations of the global approach. It takes its roots in ego-centered views, i.e. nodes connected to an *ego* node at a limited distance, with connections between these nodes. For instance, your friends and the friends of your friends

²<http://www.neo4j.org>

³<http://www.orientdb.org/>

⁴<http://www.sparsity-technologies.com/dex>

⁵<http://thinkaurelius.github.io/titan/>

⁶<https://www.palantir.com/>

⁷<http://linkurio.us/>

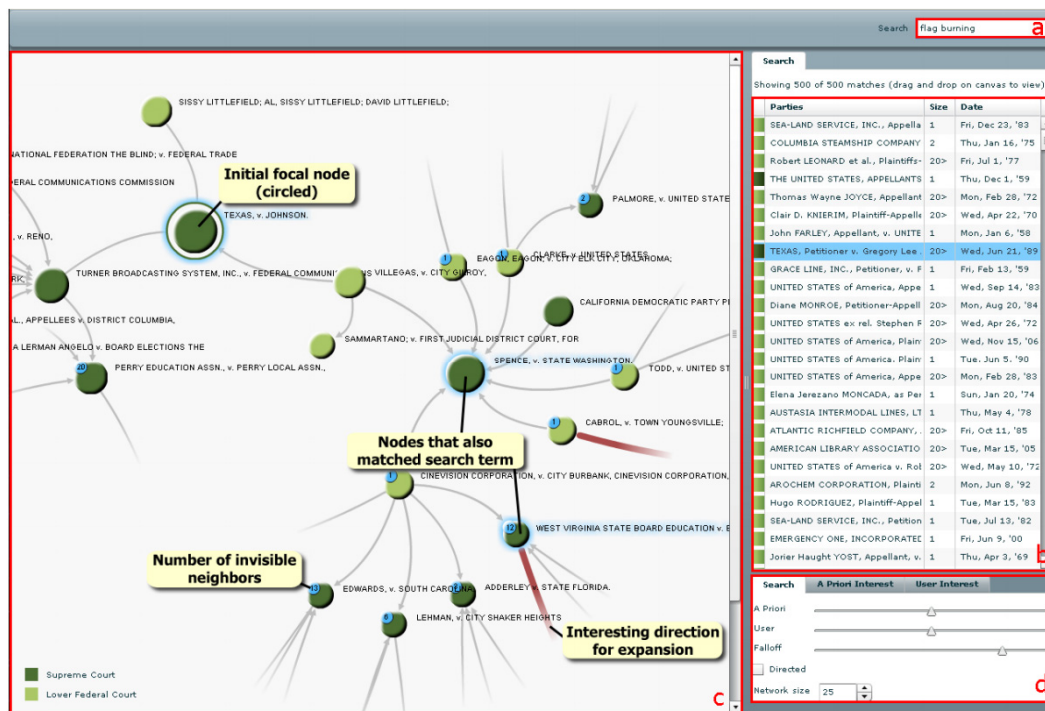


Figure 3.15: Screenshot of [vHP09] showing a local exploration of a graph. “A user types a query in the searchbox (a) which yields a number of hits presented in tabular form (b). One of these hits can then be dragged to the main screen (c) which shows the sub-graph centered on that node. Other nodes that match the user’s search are highlighted in blue. Users can adapt the balance between different components of the Degree Of Interest function and the size of the subgraph in a separate panel (d).”

can be represented by an ego-centered network of distance 2 from the ego (you). Such views are traditionally found in Social Network Analysis studies [Was94].

The goal of local approaches is to ease the navigation from node to node and to help focus on nodes of interest without being distracted by the rest of the network. The key point is to visualize nodes and links surrounding a given node or a given group of nodes, then to expand this local view with additional neighbors according to the analyst's interest. The initial set of nodes may be the result of a search query or a pre-computed view provided by the system which defines an "optimal" context. The nodes to be expanded may also be suggested by the system based on topological features and properties of nodes and links. The main benefit of this approach is the reduction of the number of simultaneously displayed items so that a large variety of devices may be able to display the representations.

3.1.6.2 Drawbacks

The main drawback of the local approach is the loss of a complete overview. It can lead to a false perception of global properties of the network under study as shown in [New03]. Such views are indeed biased samples centered on specific nodes.

Moreover, users get easily lost in the graph because layouts change when nodes are added in or removed from the local view. A potential solution is the integration of a "mini-map" (i.e. miniature map typically placed at a screen corner to aid orienting in the visual space) displaying a stable yet simplified representation of the whole network. Mini-maps are sometimes provided in overviews as well as they help when the camera is zoomed in.

Another problem also appears when nodes with a high number of connections are displayed. Sometimes called super-nodes or hubs, they may have thousands of connections while average nodes have less than a dozen. They distort representations which become hard to read and to navigate. [vHP09] proposes a solution against the "super-node problem". The node's local view is computed based on current browsing activity, network topology and nodes with statistically interesting properties. Only nodes with the highest interest scores are displayed. Analysts can then expand the local view on directions suggested by the system.

In the global approach, the entire network is a single entry point. On the contrary in the local approach, analysts can enter in the network in multiple ways, for instance using a search query, a meta-graph, or a pre-computed view. Other techniques must hence be provided to help dig into the network data, i.e. to provide an entry point into the network.

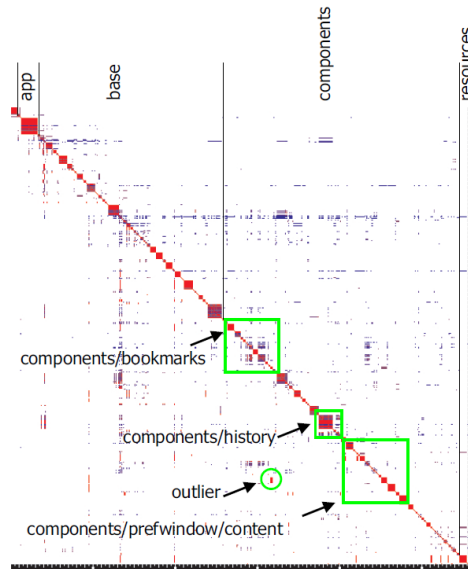


Figure 3.16: Example of outlier detection in matrix [BDW05]: “associations of the files in the /browser subdirectory of the software archive of the MOZILLA project. As the files are ordered hierarchically one can see that files which are next to each other, i.e. those that are in the same part of the hierarchy, are more strongly related than others. Thus clusters typically extend along the diagonal of the matrix. Outliers are these pixels representing couplings between files in different directories. One outlier is highlighted by a circle.”

3.1.6.3 Entry Points Techniques

The first entry point is inspired by information retrieval techniques, as it implies searching for and selecting a focal node, then displaying its neighborhood. Van Ham and Perer introduced this technique in [vHP09] to help reveal points of interest in a large citation network of legal documents. They coined the following mantra to sum up the approach:

“Search, Show Context, Expand on Demand”

This approach allows analysts to navigate in the network by following the links with a minimal visual complexity, even in large networks. The main drawback is the necessity for the analyst to have an initial idea of the things to look for in order to formulate a relevant search query. To guide the analyst in the search process, the graphical user interface should provide enough affordances (i.e. “the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used.” [Nor88]), like the auto-completion of the search field results.

The second entry point relies on the computation of a meta-graph, i.e. a graph computed from the original network where meta-nodes represent node aggregates, and where meta-links represent aggregates of links that connect the nodes of two meta-nodes. The meta-graph is much smaller than the original network in order to use it as a “summary” graph. The meta-graph is the result of an automatic analysis performed before visualization. Once displayed, the analyst can expand meta-nodes of interest to dig into sub-parts of the network. This approach is proposed for instance by [MHYLG13] to visually explore large folksonomies, i.e. triplets of $\langle user, document, tag \rangle$, which result from social bookmarking tools such as Flickr, Delicio.us and Bibsonomy. This approach is related to the Visual Analytics mantra [KMSZ06], which is an established guideline for iterative visual analysis of large datasets:

“Analyze First, Show the Important, Zoom and Filter, Analyze Further, Detail on Demand”

The difference with the Visual Information Seeking mantra is the emphasis on the automatic discovery of points of interest before any visualization.

Finally, the last entry point relies on the extraction of a sub-graph prior to visualization, using data mining algorithms. This approach is particularly appealing in the case of temporal networks, where a slice of the network evolution may be extracted and projected as a static network for further visual analysis [LJRH11, MFKN09, LSH⁺11].

3.1.7 Visual Outlier Detection in Static Networks

Visualization has inherent capabilities to help reveal outliers in static networks. As we have seen in this Chapter, the visual language of node-link diagrams and adjacency matrices (see Figure 3.16) represent structural features such as connected components, paths between nodes, density of connections, and communities. They turn out to be efficient in spotting structural outliers like isolated nodes and small components, bridges between groups, surprising links [RJTT⁺06], and unusual communities. Researches in human perception provide guidelines to augment representations with visual features for reaching *preattentive processing* effects, where outliers can be identified without cognitive effort and almost instantaneously. The Gestalt theory also helps explain how we interpret various arrangements of elements to better understand biases (e.g. false outlier detection and missed outliers), to improve existing layout algorithms and to create new visual metaphors that are efficient in terms of outlier detection. The use of combined views is reported in [NSGS07] to make outliers more noticeable.

However sheer visual analysis is not sufficient: the readability of graph layouts generally decreases with an increase of network size, and matrices

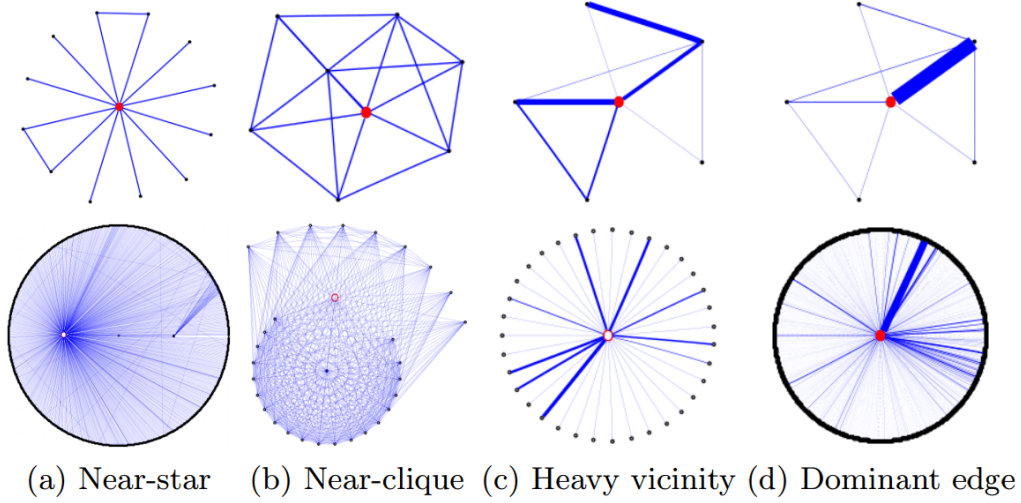


Figure 3.17: Node outliers detected by OddBall [AMF10a].

are limited by the number of rows (or pixels) that can be displayed at a single glance. Integrating data mining techniques into interactive exploration systems is a solution to assist the analyst and to speed up the detection of relevant outliers. For instance, terrorists heads have been identified using a network analysis method in [PS08].

Large graphs of billion nodes and links pose the fundamental problem that there are simply not enough pixels on the screen to show the entire graph. Finding a good starting point to investigate them becomes a difficult task, as users can no longer visually distill points of interest. A method based on anomaly detection techniques in data mining (e.g. OddBall [AMF10a], see Figure 3.17), called *attention routing*, is proposed in [Cha12] to channel users' attention through massive networks on interesting nodes or subgraphs which do not conform to normal behavior.

Able to spot outlying ego-networks, EgoNav [HACH12] is a visual analytics system that characterizes egos based on the relationship structure of their egocentric networks and presents the results as a spatialization. An ego, or individual node in a network, is most closely related to its neighbors, and to a low degree, to its neighbor's neighbors. For example, in social networks, people are closely related to their friends and family. Using network motif analysis and dimensionality reduction techniques, the system places egos in similar areas of a spatialization if their egocentric networks are structurally similar. This view of a network discriminates between the various classes of typical and exceptional egos.

To illustrate outlier detection in a non-trivial case where the expert has no prior idea of what would constitute an outlier, we take the example of a Web

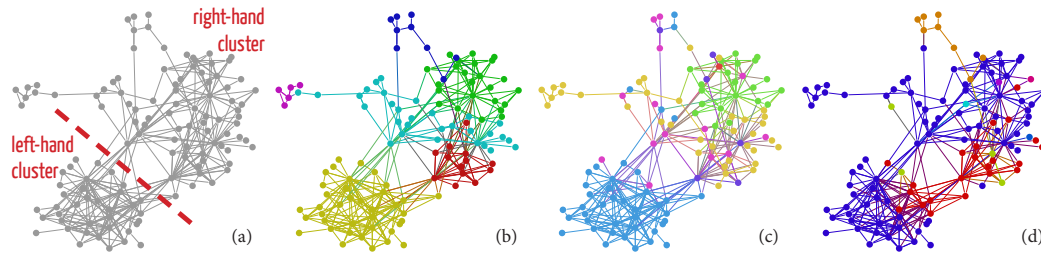


Figure 3.18: Example of outlier detection in a node-link diagram [HLG13a]: Giant component (i.e. connected component of the largest number of nodes) of the network of moroccan migrants websites (e-Diaspora): (a) graph laid out using the ForceAtlas algorithm; (b) colors mapped to Louvain modularity communities (resolution=1); (c) colors mapped to website categories (blue=blog, green=institutional, orange=NGO, ...); (d) colors mapped to languages (blue=French, red=English, orange=Spanish, ...). Links colors correspond to source node colors.

data analysis from the e-Diaspora research project [Dim12]. This project aims at studying the usages of the Web by migrants communities. A migrant site is a website created or managed by migrants or that is related to migrations or diasporas. This may be for example a personal site or blog, the site of an association, a portal / forum, an institutional site. After the collection of the initial corpus of websites, researchers annotate each website using properties defined manually, and an automatic detection of the website's main language is performed. The dataset contains both websites (nodes), hyperlinks between them (links), and properties of the websites (i.e. node properties). Then the network is visually analyzed using Gephi. The following analysis is performed on the network of websites of the Moroccan diaspora.

The Gephi ForceAtlas layout is applied to this network to get an overview of the network structure, see Figure 3.18 (a). We observe that it is clearly divided into two main clusters⁸ of nodes (on the bottom-left and on the top-right) with a few nodes connecting these clusters.

To validate this observation, the Louvain modularity algorithm (resolution=1) is applied, which automatically detects non-overlapping communities that are finally represented with different colors. Intuitively, it shows how the network is divided naturally into groups of nodes with dense connections within each group and sparser connections between different groups. We see in Figure 3.18 (b) that the left-hand cluster is clearly detected. Sub-clusters are also detected in the right-hand cluster (the resolution parameter may be modified to find different sub-clusters), however Louvain algorithm provides no justification on the existence of these clusters. The algorithm may indeed

⁸The equivalence of the problem of finding visual clusters and statistical clusters is demonstrated in [Noa09].

detect communities in networks with no community structure, which is one of its limits.

One would like to explain why these clusters exist, and why some nodes (visual outliers) act as bridges between them. The correlation between node properties and visual patterns is thus studied. The property called *website category* is mapped to node colors, see Figure 3.18 (c). We observe that the left-hand cluster corresponds very clearly to websites classified as *blogs* (in blue). This trivial grouping supports the hypothesis that blogs tend to be more connected to other blogs than to the remainder of the websites. However there is no trivial grouping for the right-hand cluster. So the property of *website main language* is mapped to node colors, see Figure 3.18 (d). We observe that the websites of both left-hand and right-hand clusters are mostly written in French (in blue), but the clusters also contain some websites written in English (in red). A sub-cluster (in red) in the right cluster is also confirmed; it corresponds to the red cluster detected by the Louvain algorithm. Finally, we observe that one of the websites connecting the two clusters is written in English, and it is connected to the other websites in English. This node is clearly an outlier, and more importantly we can explain how and why this node has a particular role in the network. This observation indeed supports the hypotheses that the existence of hyperlinks between websites is correlated to websites language, and that the outlying website seems to play a key role for websites written in English. Data mining techniques are therefore very helpful for visualization but we have seen that they are no substitute to them in exploratory approaches, as illustrated here in the interpretation of clusters.

The methods for static networks may be used when dealing with temporal networks, if static networks are extracted from time periods. Nonetheless, temporal networks calls for more specific approaches, which we cover in the following Section.

3.1.8 Visual Event Detection in Temporal Networks

Temporal networks (i.e. networks where nodes and links appear and disappear over time) have been the subject of increasing interest, given their potential as a theoretical model and their promising applications. Most of the temporal networks we have encountered so far are encoded using three different techniques. The first one consists in a series of networks (usually called snapshots) representing the state of the network at different moments of time. The second technique consists in a series of changes, like the addition and removal of nodes and links. The third technique consists of a network where node and link existence is bound to time intervals (i.e. selection of time points). Surprisingly, a fourth technique is sometimes found: static networks contain nodes which represent periods of time, where entities existing in these periods

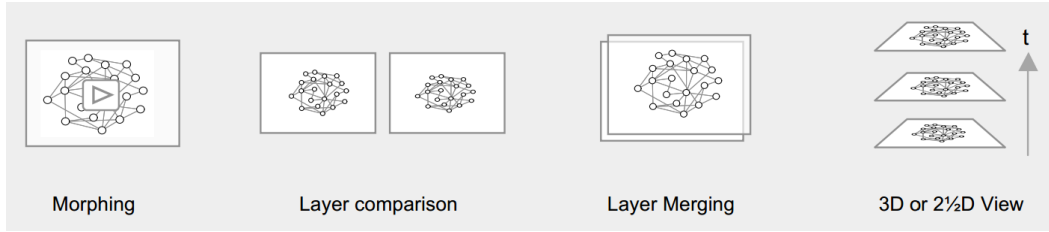


Figure 3.19: Different methods of visualizing network dynamics [WZF11].

are connected to their respective nodes [LJV⁺]. Challenges rely on integrating time into visualization for each encoding technique, but most researches focus on the first case in practice.

3.1.8.1 Temporal Network Visualization

Known as the dynamic network visualization problem, divergent solutions appear when considering two kinds of analysis tasks [SWW11]: the first one consists in identifying general features of the temporal evolution of the network; the second one focuses on a specific node to study its properties and neighborhood over time. However combined solutions appear to support both temporal analysis, and the analysis of relational features and intermediate structures like clusters.

Approaches for the study of important time features are divided into four categories (see Figure 3.19). The first one, called *morphing*, relies on the animation of the network with either fixed or dynamic layouts (like a movie, see Figure 3.20) using a timeline or slider component [BdM06]. The advantage of this visualization technique lies in its real dynamicity, which allows for a “real-time” reproduction of structural evolution that can be visibly accelerated or decelerated by selecting different frame rates. A disadvantage of morphing is the fact that a comparison of two different network shapes can only be achieved in a viewer’s memory and, therefore, precise matching of different time layers is not possible. Software tools that support the morphing approach are among others SoNIA [BdM06], Visone [BBB⁺02], Pajek [dNMB05], and ORAnet [CCD⁺08].

The second one, called *layer comparison*, represents the network evolution during a time period in a single view by either splitting it in small multiples (i.e. series of small graphics) representing the network state at (at least two) different instants [APP11a], or by tracing trajectories. Small multiples may be embedded into a timeline to support navigation over time-based animated networks [BPF⁺12]. The simultaneous, side-by-side-representation of these multiple images allows the reader to carry out an immediate, parallel comparison of inter-frame differences. This permits a detailed change analysis, as well as projected target-performance comparisons.

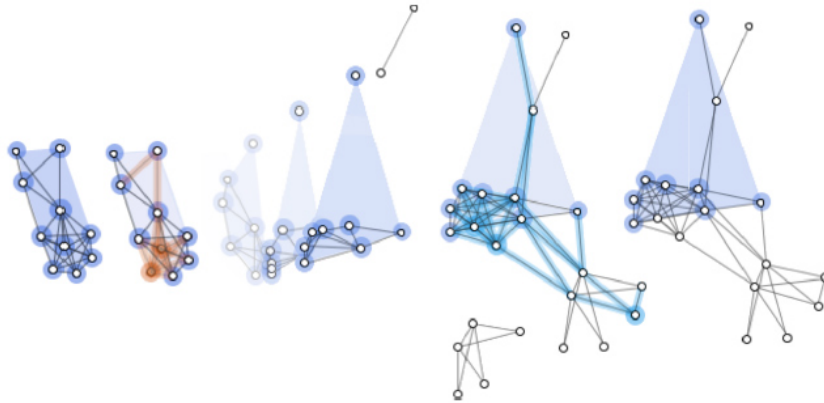


Figure 3.20: Dynamic network animation (from left to right): initial state, element removal (red), layout change, element addition (blue) and final state [BPF⁺12].

The third one, called *layer merging*, involves the visual fusion of two different static snapshots of one changing network. The result is a “two-in-one” representation in which the different time layers are visually distinguished (primarily using different colors or styles). One advantage of this method is the potential visual amplification of structural differences by subtracting identical relations and highlighting the variations. A possible handicap – especially for large-scale high density networks – could be increased intersections and opacities [Kre05]. This option is used, for example, in the Visone [BBB⁺02] and Condor [Glo07] tools.

The last one, called *2.5D view*, is a 2.5-dimensional modelling method that uses the third dimension to display network changes. Building on a two-dimensional visualization that establishes the basis of a cube (x- and y-axes), height represents time (z-axis) and can be used to stack different time layers [Dwy05]. This approach originated in time-geographical modelling and can be found in the GEOMI [ADF⁺06] or GeoTime [KW05] tools as an adapted visualization environment for network evolution. A detailed review of these techniques is available in [WZF11].

In comparison with static representation, animation seems to increase positive reactions to track changes, because the corresponding static graph is often perceived as overloaded [CDBF10, NTY01]. Furthermore, it is noted that animation better reflected how the data changed over a period of time [TK07]. [GMH⁺06] found out that the participants of their study identified patterns more correctly and faster with the usage of animation than they did with small multiples. However, the level of success for the identification of patterns or groups depended on how strong the moving clusters and patterns were. The results of the comparison study concerning the different animation

types by [MCF07] pointed out that the participants have clear preferences in combination with specific time spans. For example, circular animation is preferred by most participants if the time is perceived as a repeating pattern (e.g., 24-hour period) and a linear animation is more attractive for the participants for more common linear models which have non-cyclic character. It is also mentioned that animation causes participants to get lost, especially if too many data points were moving across the screen which possibly distracted the viewers' attention [FQ11,NTY01,RFF⁺08]. In that case, [FQ11] suggests to use animation sparingly. Furthermore, it also plays a significant role if the participants have the possibility to control the animation or not. For example, [RFF⁺08] points out that participants have to find very rapidly the data they want to observe when they have no control over the animation. In comparison if the animation or small multiples are more effective for analysis tasks, [RFF⁺08] found out that their animation approach to show trends in data is valuable especially for presenting data. For tasks of data analysis the usage of small multiples appeared more effective. But [AP12] has found the opposite: small multiples are more efficient for presentation (probably because they offer a quick access to different points in time), but animation performed better on orientation tasks such as locating nodes and following long paths.

Evaluation studies [GMH⁺06,TKSP08] have shown that the speed should also be considered to assess how well participants identify changes in the animation. For example, the participants do not have sufficient time to realize changes if the speed of the animation is too fast. However if the speed is too slow, the visual system has problems to maintain the Gestalt grouping.

In most of the evaluation studies reported in [KPS12], the participants have the possibility to interact with functions like play, pause, control the speed and jump back or forth between keyframes. In addition to buttons for play, pause or stop, the play controller often includes a slider to play the animation back or forward in order to regulate the desired speed. Although [APP11b] found out that many of their participants did not use the slider (because they felt pressed for time), it was noted that its use was useful to answer the questions of the tasks faster. The results of the evaluation studies have shown that the participants first scan the whole animation to get a comprehensive impression and for detailed analysis they use the play controller to go to a position of the animation or to influence the animation speed. It was observed that especially for the analysis of changes in data the participants go back and forth in the time lines to compare the frames and play the animation often extremely slowly (see e.g., [APP11b,BDB06,FQ11,NTY01,RFF⁺08]).

Various visualization paradigms have also been proposed beyond node-link diagrams and matrices. Space-filling techniques such as pixel-oriented visualization of changes [SWS10], complex representations of event attributes (*when*, *where*, *what*) [LAM⁺05], circular layouts for ego-networks visualization [ALGL11,SWW11], Hilbert curves and barcodes [ALGL11] (see Fig-



Figure 3.21: Barcode metaphor [ALGL11]: a white dot at coordinates (i, j) means that node j exists at time step i . The full dynamics is captured in a single view. We circle an outlier block of nodes (left) and an outlier time step (right).

ure 3.21), to name but a few.

While a single visualization is rarely considered as being suitable to cover the entire visual analysis task, researchers also explore solutions to animate transitions between combined views [HSS11]. Generally, the switch between temporal and relational perspectives can be attained by using interactions like zoom, pan, and rotation in 3D space, or by switching to a different view through a smooth transition. [FAM⁺11] proposes a solution derived from filmmaking, the *vertigo zoom*, which is a synchronized combination of a dolly movement and a zoom. This interaction technique enables smooth transitions between the relational perspective (node-link diagrams and scatter plots) and the time perspective (trajectories and line charts), supporting a seamless visual analysis and preserving the user’s mental map.

Dynamic network visualization is also considered as a sub-problem of graph drawing. Solutions developed so far usually rely on force-directed layouts and circular layouts using optimization methods to balance between the layout readability at the current time period [FT08, BIM12, Pet11, SWW11] and the preservation of the analyst’s mental map (i.e. the structural cognitive information a user creates internally by observing the layout of the graph), which should remain consistent through animations to preserve user’s understanding [PHG07, AP13]. Effectiveness on the perception of animated node-link diagrams remains also largely unclear [FQ11, GEY12], whereas graph layouts have a significant effect on the interpretation and understanding of graph structures [BMK96, Pur97].

Clustering is also a particular and important issue for dynamic network visualization. [FBS06] shows only clusters of nodes and their interrelations. [SMM13] tracks the evolution of pre-computed clusters: an optimal clustering is computed for each time-step for defining two visual representations. The first one is an overview showing how clusters evolve over time and providing an interface to find and select interesting time-steps. The second view consists of a node-link diagram of a selected time-step which uses the clustering to

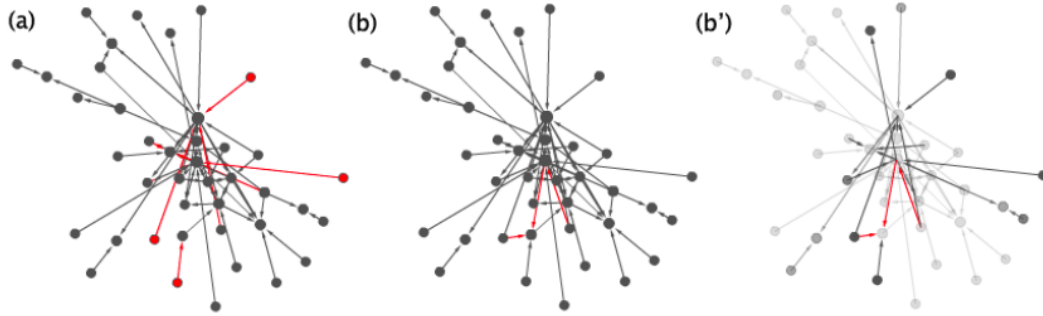


Figure 3.22: Comparing graph snapshots [ATMS⁺11]: new nodes and links are colored in red, older activities may be in lower intensities (b') to show their age.

efficiently define the layout. The authors argue that time-dependent clustering ensures visualization stability and preserves user mental map by minimizing node motion, while simultaneously producing an ideal layout for each time step. The clustering is computed in advance to update the second view in linear time, allowing for interactivity for large graphs manipulation.

3.1.8.2 Visual Event Detection in Temporal Networks

Graph layout can be used to compare evolving structures and to spot unexpected ones. Focusing on a local view around a node, [SWW11] proposes a new layout encoding time in space, thus providing a static view for each node; navigation is achieved by switching the focus node according to user interactions. Authors argue that the complexity of the dynamic network is greatly reduced without sacrificing the network and time affinity central to the focus node. This approach is applied to detect spammers in a communication network.

Color saturation is used in TempoVis [ATMS⁺11] on a node-link diagram (see Figure 3.22) and a timeline to encode time information, allowing to spot active nodes and links from a series of graph snapshots. In the node-link diagram, the nodes and links that are active in the current period are colored in red and the ones that were active before the current period are grey. The intensity of the grey decreases with age. Analysts can use the time slider (below the timeline) to navigate within time to see snapshots of each period. The timeline show how the frequency of the node-link activities change over time.

In [SFPY07], communities are tracked in a real-time fashion using a data mining algorithm, and an adjacency matrix is plotted to identify unusual temporal behavior. Events are detected by plotting the time series of a “compression cost” metric. When applied to the Enron dataset, the detected change

points appear to coincide with key events.

Combining data mining algorithms and interactive visualization, [YAPM08] provides a visual-analytic toolkit for dynamic interaction graphs. It defines a temporal snapshot $S_i = (V_i, E_i)$ of a graph $G = (V, E)$ as a graph representing only entities and interactions active within a particular time interval $[T_{s_i}; T_{e_i}]$, called the snapshot interval. The authors extract a set of S non-overlapping temporal snapshots, and eventually study the evolution of clusters.

The tool incorporates common visualization paradigms such as zooming, coarsening and filtering while naturally integrating information extracted by an event-driven framework to characterize the evolution of networks. The visual front-end provides features that are specifically useful for the analysis of interaction networks. The tool provides the user with the option of selecting multiple views, designed to capture different aspects of the evolving graph from the perspective of a node, a community or a subset of nodes of interest. Critical changes that have occurred during the evolution of the network are highlighted in the event view.

The event view is designed to provide information regarding transformations that occur in the graph over time. This view displays a set of all critical events that occur between the current and previous intervals. The user can choose different time intervals and observe the events that emerge among them. The user can vary the parameters of the event detection algorithm and examine the critical events obtained. The GUI provides a list of all critical events observed. The user can select one of these events. The detailed representation of the event is visualized on the screen giving the user a representation of the nodes involved and the change that has occurred.

The approach and the tool are very interesting but they have been designed to analyze sequences of graph snapshots, therefore missing a large amount of information that may be extracted from link streams [CE07].

Comparing the similarity of various large network diagrams through visualization is difficult, therefore a data mining process is used in ENAVis [LBVS10] to assist visualization for intrusion detection in enterprise networks. The system administrator needs to find out which hosts the compromised user account has used. Have those hosts been compromised as well? What applications did that user invoke? What data files did this user account access during the past two weeks since the user revealed his password? In this approach, the administrator generates a network graph around the compromised user node, to see which hosts and files the user has touched during the time frame and what applications the user used. A visual graph has been considered very helpful to see which hosts and users have been contacted by the compromised user account and which applications it has attempted to launch.

However, an investigator does not always know exactly what he/she is looking for. Thus an entry point is computed, and data mining algorithms are

used to augment the domain knowledge of human investigator and guide the visual exploration process only towards important things (e.g. by highlighting the part of the graph that is most suspicious and needs further investigation).

In order to help detect abnormal activity, ENAVis attempts to determine the set of variant and invariant sub-graphs. Using a graph edit distance, the difference of daily network graphs is visualized to label as events the days when the score is abnormally far from the score of expected graphs. In parallel, the cluster visualization of the whole graph helps understanding the communities within the enterprise network data (e.g., Firefox users and web related traffic, or users using similar set of applications). It eventually helps identify potential anomalous user behaviors (i.e. outliers).

This approach is thus only suited for graph snapshots analysis, and would be applicable to link streams only at the cost of a large loss of information [CE07].

Interactive coordinated multiple views are also proposed in [HHY⁺10, NJKJ05, MFKN09, LSH⁺11] to support the identification and analysis of network anomalies. In particular, [HHY⁺10] provides a node-link diagram, a scatterplot, and a time histogram that allow interactive analysis from different perspectives, as some network anomalies can only be identified through joint features. Spectral analysis methods are integrated to provide visual cues to identify malicious nodes. An adjacency-based method is developed to generate the time histogram, which allows users to select time ranges in which suspicious activity occurs.

Finally, event detection has received a large attention in the specific case of intrusion detection in computer networks. Intrusion detection systems must not only flag malicious events but also provide information needed for the assessment of security incidents. This need for explainable decisions seems to motivate the combination of data mining techniques and visualization [RL09]. These systems focus on finding entry points in data [GBT⁺09] and use complex network metrics like degree and graph distance to detect suspicious periods automatically [LJRH11], and to correlate events [LAM⁺05, QHP12]. In particular in [LJRH11], the anomaly detection module receives flows from monitoring systems and then obtains traffic flow graphs from these flows. Afterwards, these graphs are analyzed over time to detect anomalies. The attack identification module is used to visually explore causes of anomalies after detecting abnormal traffic. An alarm is eventually emitted if an attack is identified in the abnormal traffic.

These approaches are essentially intended to deal with series of graph snapshots. They are not suitable for the analysis of link streams, as [CE07] shows that much information is lost in graph snapshots and biases appear at every time scale. In the following Section we present our contributions through the experimentation of two different types of visualization for event

detection in link streams.

3.2 Our Experiments

3.2.1 Visual Events with the Global Approach

We have incorporated features in Gephi [BHJ09] in collaboration with Daniel Bernardes, Cezary Bartosiak and Mathieu Bastian⁹ to study time-varying networks, where nodes and links appear and disappear over time. Time-varying networks have been a high concern of Gephi creators since the beginning of the project in 2008. However supporting them raises several issues in terms of data structure efficiency, responsiveness of the user interface, network size and time granularity. The results we present here are our latest iteration to provide analysis and visual exploration features to the community of Gephi users.

Gephi internally encodes time-varying networks using time intervals, for which the time unit is either a number or a date. Additionally, node and link property values are also bound to time intervals. Users can thus study the evolution of network structures over time, but also the evolution of node and link properties.

Our prototype provides a new *timeline* component to select a time interval for which a sub-graph (of nodes and links which appear at least once during this interval) is computed and displayed. Moreover, basic statistical properties can be computed over time given a time interval (e.g. value for each day): the number of nodes, links, graph density, average degree and individual node degree. The results are integrated using a *sparkline chart* [Tuf06] on the *timeline* background: this feature helps users focus on specific periods of the time-varying network, like bursts of connections or changes in graph density. Finally, the selected time interval is animated: users can make it slide as the corresponding network is being displayed, either manually or automatically by calibrating the animation speed and frame-rate. Users may apply a layout during the animation to update nodes positions. Whereas the benefit of animation is still discussed by researchers [APP11b], this feature has been asked for by the user community through the bulletin board forum of the community and through emails, for the creation of videos of the network evolution over time.

Figure 3.23 shows the Gephi *timeline* in action on the COFA Online network introduced in the Introduction section. This study was part of an analysis of the COFA Online audience on social media. Simon McIntyre, director of COFA Online, has explored the dynamics of communication and used our prototype to illustrate his findings. Our tool has helped him identify a few

⁹<https://gephi.org/2011/gsoc-mid-term-a-new-timeline/>

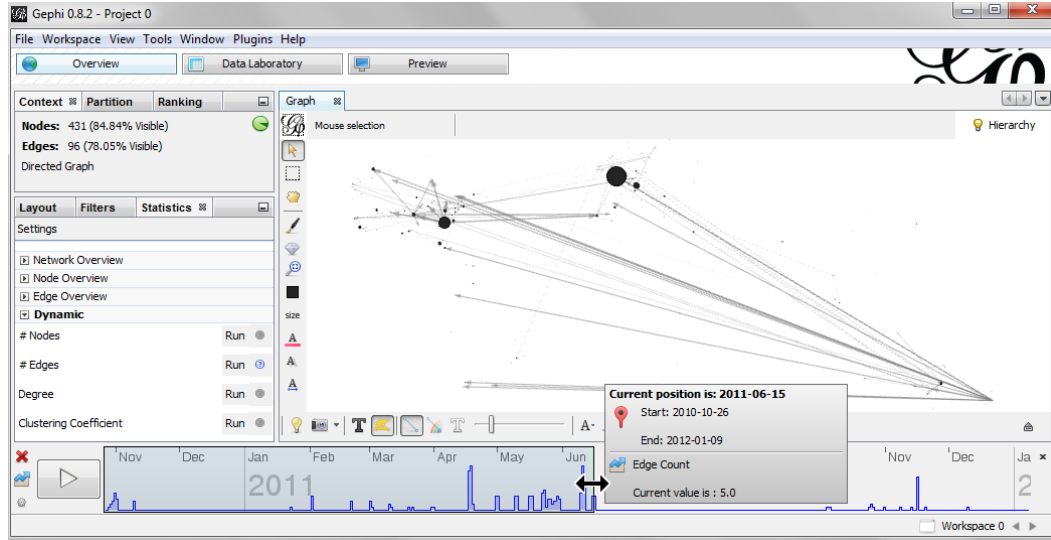


Figure 3.23: Screenshot of Gephi showing the dynamic network of Twitter users (connected when a user mentions another user in a tweet) talking about COFA Online from October 26, 2011 to June 15, 2012, filtered using the *timeline*. The *sparkline* chart on the *timeline* background at the bottom of the Figure corresponds to the number of observed links (here tweets) each day. Nodes are geolocated and positioned according to the Mercator projection.

Twitter users who were particularly active, but we were not able to detect clear outliers. We have investigated the events that appear on the timeline: they are correlated to multiples mentions of a user after a message is posted. These events are not surprising and our approach brings little but the direct selection of the sub-graph corresponding to these periods.

Therefore we do not consider this first approach as satisfying. Browsing the graph is harder as the graph size increases, and is also computationnaly costly as all data is stored in memory, thus our prototype cannot handle graphs with millions of links. Despite that, Gephi users use it mostly to analyze Twitter networks of hundreds of nodes and make videos of their evolving graphs. They eventually publish them on online sharing platforms like YouTube¹⁰ and Vimeo¹¹.

3.2.2 Interest Points on Static Networks with the Local Approach

We have experimented a global approach to outlier detection in the previous Section, but we have not found it satisfying. In this Section we present our

¹⁰https://www.youtube.com/results?search_query=gephi+dynamic

¹¹<https://vimeo.com/search?q=gephi+dynamic>

experiments on a collaborative project called Knot [UCC⁺13] in which the central idea is to let the expert interactively shape local views of the graph based on his knowledge and questions. Whereas the dataset is static, this prototype and case study has led to interesting preliminary results. The approach is particularly relevant when data is highly incomplete, biased towards a few nodes, or with vague information. In such cases, the global approach may lead to a misinterpretation of the visualizations.

We have contributed to Knot software¹², a digital tool for exploring historical social networks, developed within a multidisciplinary research context involving designers, humanities scholars and computer scientists. The goal of Knot software is to provide scholars and researchers with an environment for exploring multi-dimensional and heterogeneous data, allowing them to discover and create explicit and implicit relationships between people, places and events. The graphical user interface runs on Web browsers, and queries a remote database to support various devices. The main challenge relies on the high level of uncertainty and incompleteness of data coming from MRofL, due to a number of reasons such as the nature itself of the data (e.g. letters from the 17th century), the process of acquisition and digitization (e.g. letters are handwritten making it difficult, if not impossible, to recognize and process the content) and the heterogeneity of different sources (e.g. each data collection provides different content and meta-data).

What distinguishes this approach to traditional network exploration and analysis is a shift in attention on the construction of the network graph through the visual interface, rather than on its static contemplation. While visualization is often conceived as the last step in the exploration of data, our idea is, instead, to put it in the middle of a broad process of understanding and exploration [MVC⁺10]. In this way, the whole tool (not just the visualization itself) has to be considered as an environment where to engage with the data and to perform interpretative activities.

Users can decide where to start their exploration using a search engine (see Figure 3.24). It gives the possibility to search the archive for a particular person or a group of people that share some attributes (e.g. born in the same country) and to add them into the representation. An autocomplete feature helps the user during the search query, suggesting the available data that match the request and giving some basic biographical information (birth and death date), in order to disambiguate homonyms. This action can be performed anytime: to enrich the visualization with other nodes, but also to look for a specific node (or group of nodes) in the visualization with the search

¹²This software has been created within the context of the Mapping the Republic of Letters initiative (MRofL), in partnership with the Stanford Humanities Center (<http://shc.stanford.edu/>), the DensityDesign Lab (<http://www.densitydesign.org/>) at the Politecnico di Milano University, and the Gephi Consortium (<https://consortium.gephi.org/>). More information in Section 2.2.2

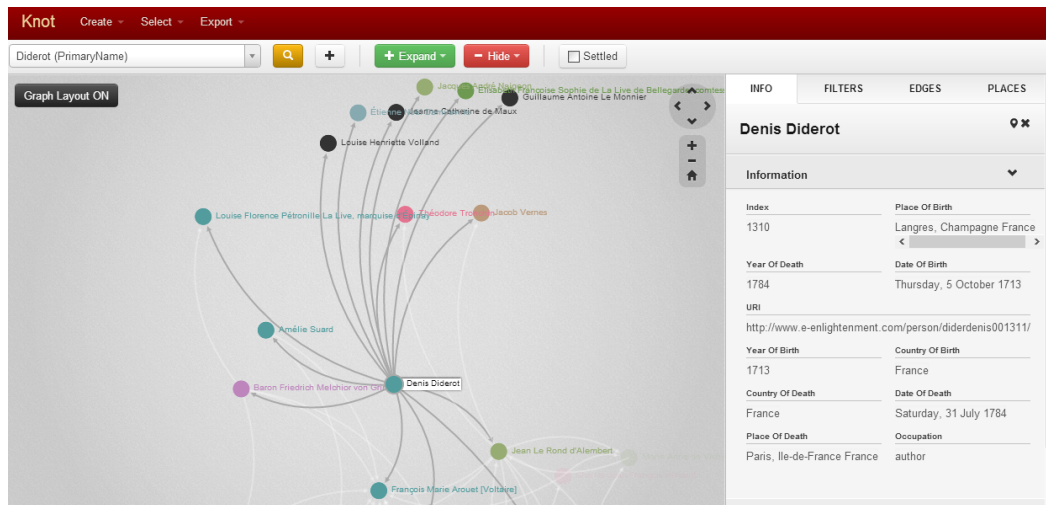


Figure 3.24: Screenshot of Knot retrieved on May 2013, showing the local view around Denis Diderot. Nodes represent people, and links represent letters exchanged. The sidebar displays the properties of the selected node.

bar. Users can select nodes singularly or through multiple selection features (e.g. inverse selection, selection of a certain degree, of common nodes between two or more nodes). A panel on the right shows the main information for the selected nodes and allows the user to add explicit relationships or remove nodes from the canvas. By selecting “Create” on the top menu, the user can add new nodes and new relationships between individual nodes, to enrich data or to investigate some hypotheses. Users can also apply force-directed layouts, display specific relationships only, and filter the network to refine the representation (see Figure 3.24).

The project is still under active development and also aims at exploring new opportunities for interface design and information visualization within the definition of novel research practices in the Humanities, bringing together scholars, HCI, design, and computer science communities.

During this process we have learnt how a local approach could be applied to the study of static networks. Preliminary results are encouraging as they help infer knowledge when most information is not available in data but belongs to the expert. We will see in Chapter 6 how we use this approach to investigate network events detected statistically.

3.3 Conclusion

Network visualization is a powerful medium between data and analysts. A wide range of methods and tools are available to study static datasets, however little has been done on temporal networks, especially in terms of event

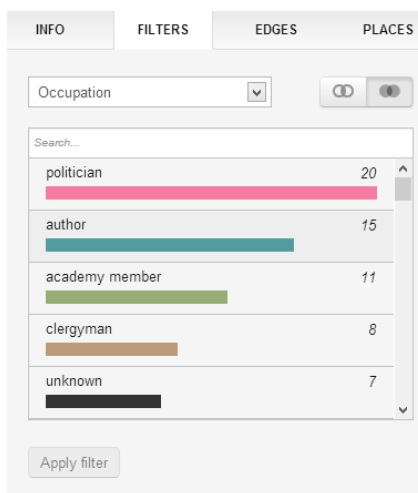


Figure 3.25: Screenshot of Knot showing the Filters panel. Dots of the node-link diagram are colored according to the people's occupation.

detection. We have reviewed two types of approaches in this Chapter:

- The global approach is well studied but is difficult to apply to large networks (because there is too much information to see anything), or to incomplete datasets where visual results may be strongly biased.
- The local approach opens promising perspectives for the interactive exploration of large and dynamic networks.

We have then experimented both approaches. We have first proposed a global visualization system to study a relatively small dynamic interaction network. It has been successfully used but presents technical limitations and is not suitable to event detection. We have therefore proposed another system to help experts build local views of a network interactively. This promising approach is explored further for event detection in Chapter 6.

Visualization is not enough to deal with large dynamic networks. Link streams in particular raise difficult challenges as they are not graph snapshots ready for visualization. In the following Chapter, we introduce a data mining technique applied prior to visualization, in order to reduce the amount of information on screen. More precisely, this technique automatically detects statistical events from the link stream, which analysts may explore visually to validate and ultimately interpret these events.

CHAPTER 4

Automatic Event Detection

Contents

4.1	Related Work	71
4.1.1	Outlier Detection in Static Networks	71
4.1.2	Event Detection in Dynamic Data	73
4.1.3	Event Detection in Temporal Networks	76
4.1.4	Conclusion	83
4.2	Outskewer	84
4.2.1	Skewness Signature	84
4.2.2	Our Method	85
4.2.3	Experimental Validation	87
4.2.4	Real-World Applications	94
4.3	Conclusion	99

A frequent need in terms of complex network analysis is to monitor network evolution and automatically raise alerts on abnormal behaviors, i.e. that are statistically different from most others. As explained before, this challenging task is generally called *outlier detection*, and *event detection* for temporal data. In spite of many works on this question during the last decades in various fields, the diversity of cases leading to different outlier definitions makes it hard to create a single universal method. While visualization provides flexible solutions, we have seen its limits in the previous Chapter. We now explore an automatic solution for outlier detection.

Let us consider for instance a property measured on an evolving network (Figure 4.1). How can we automatically and reliably identify outliers in it? This is challenging because these data exhibit at the same time regime changes (i.e. sudden changes of the average values in the time series) due to the evolution of the normal behavior, and outlying values that deviate globally or locally from the main trend. Moreover, we have no prior knowledge on the data, and events may occur at different time scales. Finally we want an on-line method for real-time analysis. These settings are known to pose a difficult problem.

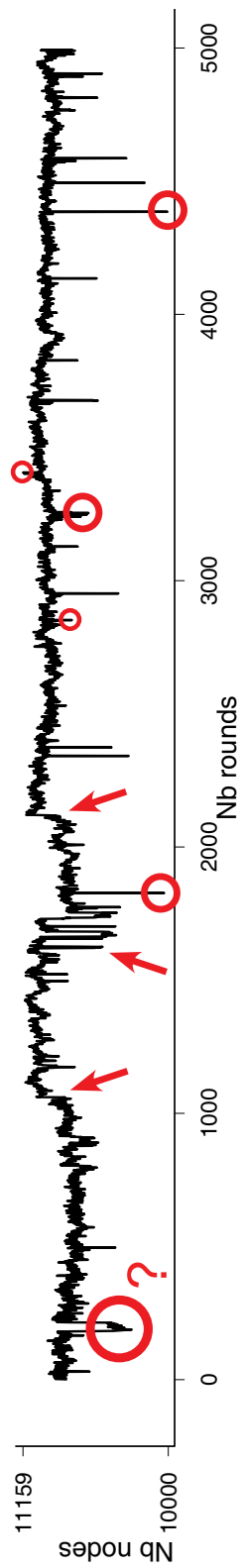


Figure 4.1: Evolution of a property measured on a network over time. Some outliers are circled. Regime changes are pointed by arrows.

In Section 4.1 we review the related work in “automatic” outlier and event detection in static and dynamic environments respectively, i.e. not relying on visualization techniques appropriate to human perception. Based on the identified limitations of existing approaches, we introduce a new method and tool called *Outskewer* in Section 4.2 to automatically detect outliers in sets of numbers and events in time series. The source code is available online [HLM12]. We illustrate its relevance on the detection of abnormal events in three use cases: the evolution of the French population during the 20th century, the dynamics of Internet topology, and the logs of a peer-to-peer search engine.

4.1 Related Work

4.1.1 Outlier Detection in Static Networks

Graph topology-based approaches for outlier detection are suitable for handling data that cannot be easily analyzed with traditional non-graph-based data mining approaches [NC03]; they are applied in several domains, notably intrusion detection in computer networks and fraud detection.

Unusual links are discovered in [C⁺03], using various metrics to define the commonality of paths between nodes. With this approach the user can determine whether a path between two nodes is interesting or not, without having any preconceived notions of meaningful patterns. This approach assumed that the user queries the system to find interesting relationships regarding certain nodes. The AutoPart system [Cha04] presents a non-parametric approach to finding unusual links in graph-based data. Part of this approach is to look for outliers by analyzing how edges that are removed from the overall structure affect the Minimum Descriptive Length (MDL)¹ of the graph [Ris89]. Representing the graph as an adjacency matrix, and using a compression technique to encode node groupings of the graph, it looks for the groups that reduces the compression cost as much as possible.

Looking for **anomalous subgraphs**, the authors of [NC03,EH07] identify subgraphs that appear infrequently, using a variant of the Minimum Description Length (MDL) principle. This approach has been recently applied to intrusion detection [EGH10]. The notion of entropy is also used by Shetty and Adibi [SA05] in their analysis of a real-world dataset: the famous Enron scandal. They use what they call “event based graph entropy” to find the most interesting people in an Enron e-mail dataset. Using a measure similar to what [NC03] have proposes, they hypothesize that the important nodes are

¹In minimum descriptive length, clustering is performed to minimize the number of bits required to represent the graph, and change points occur when there is significant change in the representation.

the ones that have the greatest effect on the entropy of the graph when they are removed.

In the more specific context of the detection of **nodes with anomalous neighborhood**, the authors of [SQCF05] identify outlier nodes in bipartite graphs based on properties of their neighborhood. Using an adjacency matrix, they assign a “relevance score” such that every node has a relevance score with every node, whereby the higher the score the more related the two nodes. Similarly, [AMF10b] proposes *OddBall*, which characterizes node’s neighborhoods in unipartite graphs by analyzing some of their features to define a “normal” neighborhood, and to detect abnormal ones on weighted graphs.

Networks of dynamic systems can be highly clustered [WS98]. A community, defined as a collection of individual objects that interact unusually frequently, is a very common substructure in many networks [GN02], including social networks, metabolic and protein interaction networks, financial market networks, and even climate networks. In social networks, a community is a real social grouping sharing the same interests or background [GN02]. In biological networks, a community might represent a set of proteins that perform a specific function together. Communities in financial market networks might denote groups of investors that own the same stocks, and communities in climate networks might correspond to regions with similar climate or climate indices.

Many algorithms have been developed for detecting community structures in static graphs. [GN02] have proposed a community discovery algorithm based on the iterative removal of links with high betweenness scores. To reduce the computational cost of the betweenness-based algorithm, [CNM04] have developed a modularity-based algorithm. [BGLL08] have designed an efficient algorithm to find communities by optimizing modularity. In contrast, [PDFV05] does not focus on detecting separate communities, but on finding overlapping communities. Defining communities as maximal cliques (i.e. subgraph in which each node is connected to each other), [SSTP09] propose a parallel, scalable, and memory-efficient algorithm for their enumeration.

Community outliers correspond to nodes with unusual position regarding communities. For instance, node attributes may be used to detect nodes in communities which have an unusual value for a given attribute, compared to the other nodes [GLF⁺10]. They are *local* outliers. In overlapping communities, outliers may be nodes that do not belong to any of the communities [NPNB08].

Comparing multiple graphs, **outlying graphs** may be seen as graphs which do not overlap well with the frequent subgraph patterns [Agg13]. The use of a k -nearest neighbors outlier detection algorithm is also a viable alternative in this case, because of a wide variety of available algorithms for similarity search in graphs. Numerous similarity functions are commonly used such as

the graph edit distance, largest common subgraph, largest matching node set, in order to perform the similarity search. While these methods correspond to generic extensions of multi-dimensional outlier detection algorithms, [Agg13] notices that not much work has been specifically targeted towards outlier detection in this domain.

Finally, a **cautionary word** should be written about so-called outlier detection methods which are reported in [Agg13], and the difference between important data points and outliers. As we have seen in Chapter 1, outliers are values which deviate significantly from the rest, i.e. there is a “normal behavior” in contrast to abnormal ones. Therefore the condition to the existence of outliers in univariate data is that its distribution of values should be homogeneous. Roughly speaking, the mean should be a relevant indicator, otherwise there is no sense of normality in the data, hence there cannot be outliers in it. However in [Agg13], which is the latest survey on outliers in networks, many cited methods based on network features that exhibit power-law (or heterogeneous) distributions, and flag as outliers the values which deviate from these power laws. However the method seems irrelevant because such distributions should not be described by the mean value due to their shape: there is no notion of “normal values” in power-laws, so there is no sense of finding abnormal values either. Although the detected values may be important, they are not outliers according to our definition.

These methods are suitable for static networks but further studies are required to understand their limits on dynamic networks. Previous works [AG10, AG11] have shown for instance that communities are highly unstable in dynamic graphs, even those with very small changes in the topology, and raise new questions. The extension to dynamic networks should therefore be considered very carefully.

4.1.2 Event Detection in Dynamic Data

4.1.2.1 Temporal Data

The detection of outliers in temporal data relies mainly on two approaches using time series. In the first one, points which deviate from a temporal model are marked as outliers. Parametric models for time series outliers proposed by Fox in [Fox72] represents the first work on outlier detection for time series data. Several models have been proposed in statistics literature after Fox’s work, including autoregressive moving average (ARMA) [BJ76], autoregressive integrated moving average (ARIMA), vector autoregression (VARMA), CUMulative SUM Statistics (CUSUM), exponentially weighted moving average, finite-state automaton models [Kle02], etc. A detailed presentation is beyond the focus of this Chapter, and more information can be found in [BL94, Haw80, RL87]. In the second approach, points very different from

other points within a sliding window are marked as outliers. Regime changes (i.e. sudden changes of the mean between two (near-)steady states in time series) may be considered as anomalies (drifts) as well [Car88, TY06]. The impact of resolution (i.e. time scale) on detected events is however rarely studied [ZZJ⁺08].

A more comprehensive survey on general methods to detect outliers in temporal data can be found in [GGAH13]. Whereas numerous methods exist, we focus now on distance-based techniques in data streams, because they are more relevant to our study.

4.1.2.2 Distance-based Outliers for Sliding Windows

Given a data stream, at any time point, one can discover outliers from the set of points lying within the current sliding window. Distance-based outliers can be discovered both in a global as well as in a local sense on points within the current sliding window. Variants of such distance-based outliers for sliding windows have also been discussed in the literature.

Distance-based Global Outliers: Given a data stream, the problem is to find distance-based outliers in any time window [AF07, YRW09]. As the window moves, old objects expire and new objects come in, see Figure 4.2. Since objects expire during stream evolution, the number of preceding neighbors of any object decreases. Therefore, if the number of succeeding neighbors of an object is lower than k , the object could become an outlier only because of the stream evolution. Conversely, since any object expires before its succeeding neighbors, values having at least k succeeding neighbors will be considered as normal for any stream evolution. Such values are called “safe inliers”.

[AF07] proposes an exact algorithm to efficiently compute distance-based outliers using a new data structure called Indexed Stream Buffer (ISB) which supports a range query. Further they also propose an approximate algorithm which uses two heuristics: 1) it is sufficient to retain in ISB only a fraction of safe inliers; 2) rather than storing the list of k most recent preceding neighbors, it is enough to store only the fraction of preceding neighbors which are safe inliers with regards to the total number of safe inliers.

[YRW09] states that maintaining all neighbor relationships over time may be very expensive, so abstracted neighbor relationships should be maintained. But maintaining such cluster abstractions is expensive too. Hence, the authors exploit an important characteristic of sliding windows, namely the “predictability” of the expiration of existing objects. In particular, given the objects in the current window, the pattern structures that will persist in subsequent windows can be predicted by considering the objects (in the current window) that will participate in each of these windows only, and these predicted pattern structures can be abstracted into “predicted views” of each

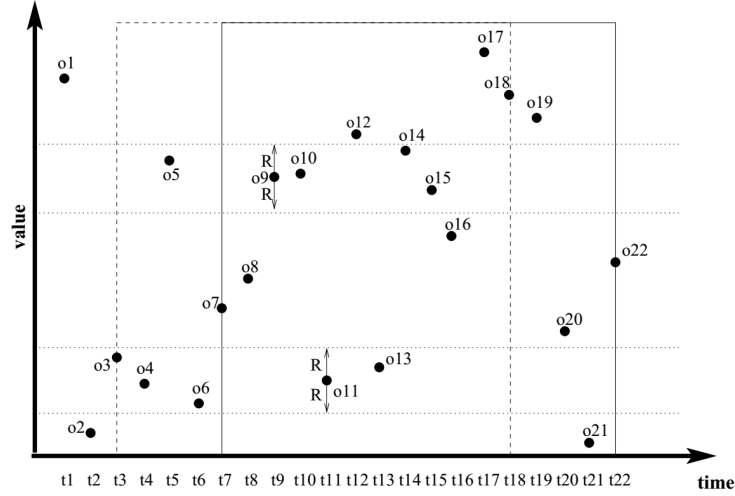


Figure 4.2: Consider the current window (the dashed one) at time t_{18} : both o_9 and o_{11} are inliers, since o_9 has four neighbors (o_5 , o_{10} , o_{14} , o_{15}), and also o_{11} has four neighbors (o_3 , o_4 , o_6 , o_{13}). Moreover, since o_9 has three succeeding neighbors, it is a safe inlier, while o_{11} is not a safe inlier. Indeed, consider instant t_{22} . The object o_9 is still an inlier: object o_5 expired, but o_9 has still three (succeeding) neighbors. Conversely, o_{11} is now an outlier: objects o_3 , o_4 and o_6 expired, and now it has only one neighbor [AF07].

future window. An efficient algorithm is proposed which makes use of the predicted views to compute distance-based outliers.

This method is suited for time series and streams of multivariate data points. It has three parameters: the maximum distance between two neighbors, the number of neighbors under which a data point is flagged as outlier, and the size of the time window within which neighbors are computed. One should thus test various configurations to find a good tradeoff between accuracy and precision of the computed outliers. On the contrary, our approach is limited to univariate data but it only needs one parameter (the size of the time window). Moreover, events are not interpreted in the multivariate dataset; it may therefore be hard to explore a dataset using this method. As it focuses on one variable at a time our method is more suitable for event interpretation.

Distance-based Local Outliers: Static Local Outlier Factor (LOF) [BKNS00] can be applied to the incremental LOF problem (i.e., the stream setting) in three ways: 1. periodic LOF, i.e., apply LOF on entire dataset periodically, or as 2. supervised LOF, i.e., compute the k-distances, local reachability density (LRD) and LOF values using training data and use them to find outliers in test data or as 3. iterated LOF where the static LOF algorithm can be re-applied every time a new data record is inserted into the dataset. A better approach is proposed in [PLL07] where the incremental LOF algorithm com-

computes LOF value for each data record inserted into the dataset and instantly determines whether the inserted data record is an outlier. In addition, LOF values for existing data records are updated if needed. Thus, in the insertion part, the algorithm performs two steps: a) insertion of new record, when it computes the reachability distance, LRD and LOF values of a new point; b) maintenance, when it updates k -distances, reachability distance, LRD and LOF values for affected existing points.

[VGN08] proposes a problem similar to the local outlier factor for data streams. Let D_p be the average distance of a point p from its k nearest neighbors. Let μ_p and σ_p be the mean and the standard deviation of D for all neighbors of p . Then the relative outlier score for p is given by $1 - (D_p/\mu_p)$. The point p is called an outlier if the absolute value of its relative outlier score $> (3\sigma_p/\mu_p)$. Given this definition, the authors propose a Relative Neighborhood Dissimilarity algorithm to detect outliers, that maintains a set of n clusters that represents the k nearest neighbors knowledge captured from the historical data.

These methods are also suited for multivariate data points. They maintain a data model to cluster the incoming data points and to spot as outliers the points far from these clusters. However the interpretation of these events is not performed in the papers so we cannot validate the relevance of the methods. In the following, we review specific methods to detect events in temporal networks.

4.1.3 Event Detection in Temporal Networks

The detection of events in evolving networks addresses two problems:

- **Change points detection:** given a sequence of graphs or a stream of links, find time points at which graph changes significantly.
- **Attribution:** find (top k) nodes / links / regions that change the most.

In this Section we cover recent methods in series of graphs and graph streams, for detecting change points in time and region (or community) outliers.

4.1.3.1 Graph Streams Outliers

The studies reviewed here assume an incoming stream of graph objects (or *snapshots*), denoted by $G_1 \dots G_t \dots$. Thus, the i^{th} graph in the stream is denoted by G_i . Each graph G_i has a set of nodes, which are drawn from the node set N . Each node is associated with a unique identifier, which may be a string, such as the IP address in a computer network, or the user-identifier in a social network. The set N may be very large in web-scale applications. For example, in a network application, the node set may consist of millions of IP addresses, whereas in a social network, it may contain millions of users.

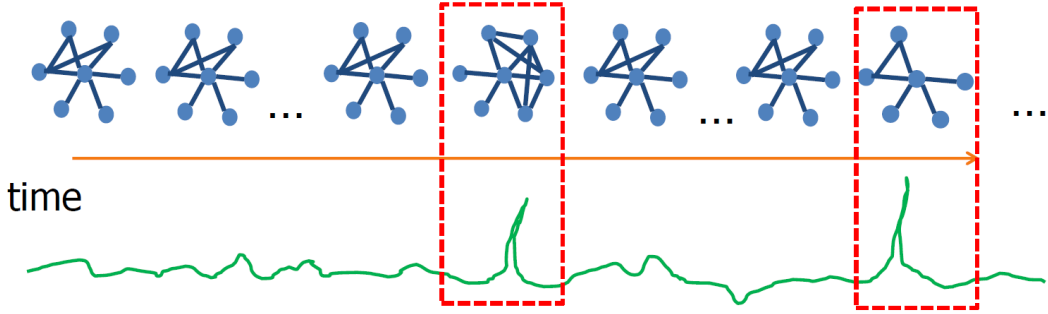


Figure 4.3: Series of graphs, or snapshots, with the time series corresponding to a measured feature of these graphs [AF10].

The main framework for detecting anomalous graph objects involves calculating a similarity/distance scores between two consecutive graphs (see Figure 4.3), then spotting as outliers the graphs for which the scores are above a certain threshold, or are too far from the values predicted by a model of the time series. Ideally the methods should scale to millions of nodes and links for real-world applications.

Using various graph topology distance measures, [Pin05] models each corresponding time series as an ARMA process. By assuming the stationarity condition, outliers are time points where fitting residuals exceed a threshold. The distances between two graphs are metrics, i.e. $d(G, H) \in \mathbb{R}^+$; the zero distance is equivalent to graph isomorphism, all distance measures are symmetric, i.e. $d(G, H) = d(H, G)$, and they satisfy the triangle inequality, i.e. $d(G, F) \leq d(G, H) + d(H, F)$.

For instance, the graph edit distance (see Figure 4.4) between graphs G and H is calculated by evaluating the sequence of edit operations required to make graph G isomorphic to graph H using the formula $d(G, H) = |V_G| + |V_H| - 2|V_G \cap V_H| + |E_G| + |E_H| - 2|E_G \cap E_H|$, where E_G and V_G are the links and nodes of graph G , and E_H and V_H are the links and nodes of graph H . [Pin07] provides an extensive study of ten distance metrics.

The main limit of the approach is the ARMA modeling that involves the detection of outliers as residuals being beyond a threshold parameter.

Instead of using a global graph distance, an alternative is to use a scan statistics framework [PCMP05]. For each small sub-part of the data, called “scan region”, a local statistic is computed. The maximum of such local statistic is called the scan statistic of the data. The approach is to check whether this scan statistic is greater than a threshold value. In that case, there is an anomaly according to this method. In networks, the sub-part to be scanned is the k^{th} order neighborhood of a node. Local statistics may be the number of nodes or links, the density, etc. The approach is however not efficient because each neighborhood must be computed for each node.

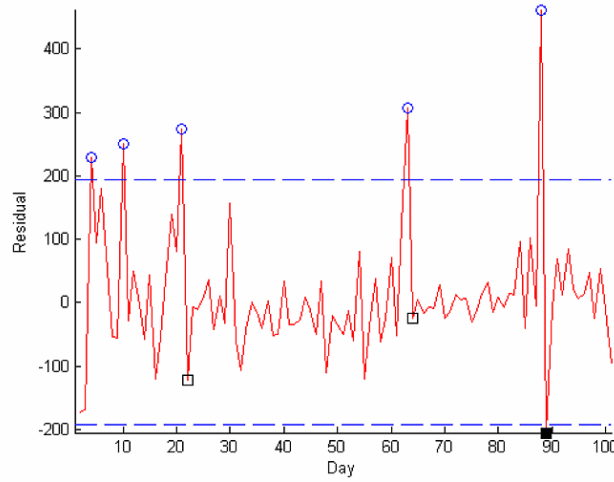


Figure 4.4: Residuals for median edit distance. [Pin05].

[PDGM10] gives a noticeable method using graph distance that compares the performance of five graph similarity measures for anomalous weighted Web graph detection: Node Ranking (“two graphs are similar if the rankings of their nodes are similar”), Sequence Similarity (“two graphs are similar if they share many short paths”), Vertex/Edge Overlap (“two graphs are similar if they share many nodes and/or links”), Vector Similarity (“two graphs are similar if their node/link weight vectors are close”), and Signature Similarity (“two objects are similar if their signatures are similar”). Performance is evaluated against the following anomalies (see Figure 4.5):

- **Missing Connected Sub-graphs:** all algorithms except Sequence Similarity successfully detect this type of anomaly.
- **Missing Vertices (i.e. nodes):** signature Similarity has the desired sensitivity at anomalies of different significance, and the behavior of Vector Similarity and Vertex/Edge Overlap is close to the desired one. In contrast, Vertex Ranking and Sequence Similarity fail to discern significant from insignificant anomalies.
- **Connectivity Change:** sequence Similarity outperforms the other methods in detecting row skipping anomalies; however, Signature Similarity and Vector Similarity are also successful.

[ZLBM06] proposes an approach to detect events from the web by analyzing the click-through data of web search engines. There are different types of events such as periodic events (national holidays), burst events (unpredictable accidents), etc. A key feature is the incorporation of the dynamic nature of the click-through data in the event detection process: the authors model the semantic and evolution pattern-based similarities between query-page pairs in

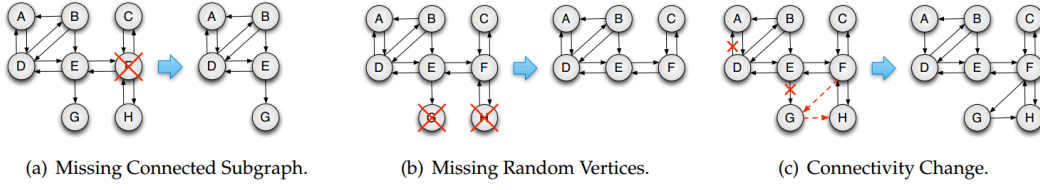


Figure 4.5: Examples of web graphs that are affected by anomalies. [PDGM10].

the click-through data as a vector-based graph. Then, the real world event is defined as a set of query-page pairs that are similar with respect to the semantic and evolution pattern-based similarities. The problem of event detection is then addressed by a two-phase graph cut algorithm on the dual graph of the vector-based graph. Experimental results are run on 15 billion records of query-page pairs over 32 days. The entire click-through data is partitioned into 768 collections, where each collection consists of the query-page pairs occurred during an interval of one hour. This method is able to detect relevant events but is out of the scope of our research problem. It is indeed not flexible enough to be applied on any link stream because of the semantic analysis and the necessary bipartite nature of the underlying network.

Another set of methods rely on graph connectivity measures, where the goal is to find anomalous time points at which many nodes “change” their behavior in a way that deviates from the norm, and to **spot the nodes that contribute to the change** the most. In the technique used in [AF10] (similar to [IK04], see Figure 4.6), each node is summarized by a set of features (e.g. in-degree, out-weight, local clustering coefficient, etc.) extracted from its egonet (i.e. the node itself, its neighbors, and all the interactions between these nodes), thus each node feature is described by a time series. Next, a sliding time window of 7 days is defined over these time series, and a correlation matrix is computed between all pairs of time series vectors. A new correlation matrix is computed at each slide of the time window (each day in the study), and an eigenvector is computed for each matrix. The value for each node in the eigenvector is thought as the “activity” of that node; that is, the more correlated a node is to the majority of the nodes, the higher its “activity” value is. Then, a “typical” eigenvector is computed from eigenvectors back in time, and the current eigenvector is compared against it, which provides an angle between the two vectors. Perpendicular vectors indicate a change point. The angles are then normalized. Outliers are the top k angles, or angles above a given threshold. Finally, to attribute change to nodes for a given time point outlier, values of the corresponding eigenvector and projected onto a scatter plot against the values of the typical eigenvector. Far-distant values from the norm are manually flagged as outliers. Experiments on a large mobile phone

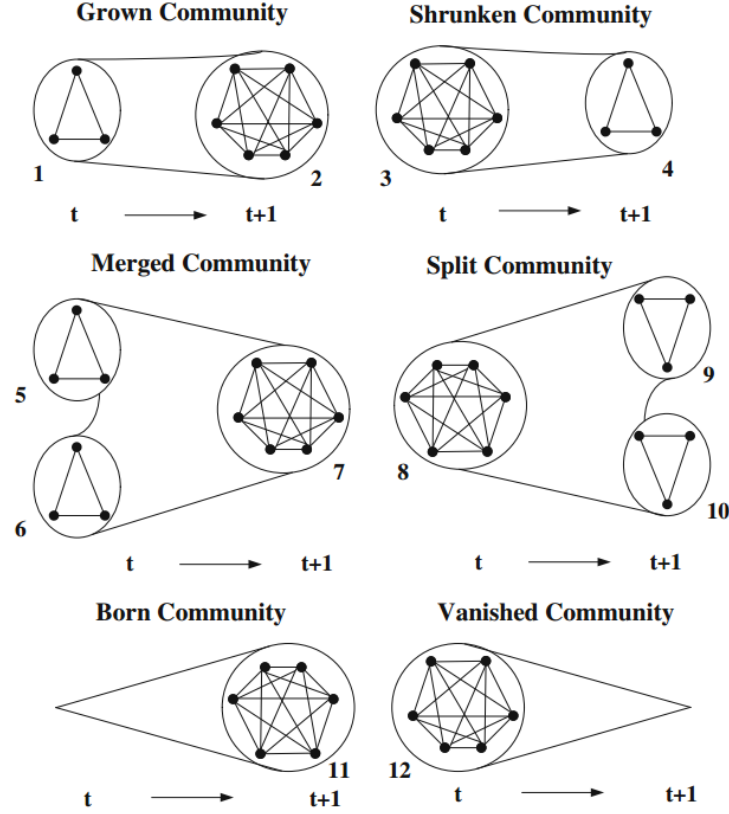


Figure 4.7: Types of community-based anomalies in dynamic networks [CHS12].

ple, is there a community at snapshot t that splits into smaller communities or merges with others at snapshot $t + \Delta t$? Does any community at snapshot t disappear at snapshot $t + \Delta t$, or does any new community appear at snapshot $t + \Delta t$? Do the sizes of the communities change over time?

[CHS12] demonstrates that there are only six possible types of community-based anomalies in dynamic undirected graphs, using maximal cliques as a definition of communities (see Figure 4.7). Given a time-varying sequence of undirected graphs $G = \{G_1, G_2, G_3, \dots\}$, where the nodes in each graph can belong to different communities, they detect the clique-based anomalies between consecutive graphs, including *grown*, *shrunk*, *merged*, *split*, *born*, and *vanished* cliques.

A key problem in discovering trends in community transition is computing matching communities across two distinct snapshots. [AG10] tracks communities over time with the static Louvain community detection using a partition edit distance, and extend it in [AG11] for multi-step community detection and hierarchical time segmentation: instead of detecting communities in each time step, they detected a unique decomposition into communities that is relevant

for (almost) every time step during a given period called the time window. Since the time window length is a crucial parameter of the technique, they also propose a unsupervised hierarchical clustering algorithm to build automatically a hierarchical time segmentation into time windows. This clustering relies on a novel similarity measure based on community structure. Seifi et al. [SG12] detected community cores in evolving networks to deal with the issue of unstable results provided by classical community detection algorithms, but complexity issues make scaling impossible.

Finally, [AZY11] proposes to find surprising links between clusters. Unusual relationships in the graphs may be represented as links between regions of the graph that rarely occur together. The goal of such a stream-based outlier detection algorithm is to identify graph snapshots which contain such unusual bridging links. In order to handle the sparsity problem of massive networks, the network is dynamically partitioned in order to construct statistically robust models of the connectivity behavior. For robustness, multiple such partitionings are maintained. The authors proposes a probabilistic algorithm for maintaining summary structural models about the graph stream. These models are maintained with the use of an innovative reservoir sampling approach for efficient structural compression of the underlying graph stream. Using these models, link generation probability is defined and then graph object likelihood fit is defined as the geometric mean of the likelihood fits of its constituent links. These objects for which this fit is t standard deviations below the average of the likelihood probabilities of all snapshots received so far are reported as outliers. The main limit of this method is the destruction of information due to link sampling, which may prevent fine event detection. The scope of this method is thus limited.

4.1.3.3 Outliers in Link Streams

Link streams are data formatted as series of time-stamped links (see the definition in 2.1). We have found few studies about them, maybe because such datasets have become available only very recently, and they are always dedicated to a specific application. Let us cite [Tho12], who proposes a novel approach to event detection in streaming communication data, based on temporal communication patterns across each link. Comparing all links or groups of appeared links is too computationally expensive, so they build a stochastic model to keep the most recent links in memory by defining a score of link recency. In particular, they estimate the distribution of inter-arrival times across each network link as a power-law distribution, as suggested in [Bar05]. Then they define a measure of anomaly for an arbitrary region based on the likelihood of its recent activity given past behavior. Finally, they identify the region with the most anomalous activity. This approach is focused on the detection of changing regions and is thus too specific for our concern: it is

presupposed that such regions can be determined and are comparable.

Another kind of link stream is clickstream data, i.e. the recording of the parts of the screen a computer user clicks on while browsing the web or using another software application. Every time users click anywhere in the web page or application, the action is logged on a client or inside the web server, as well as possibly the web browser, router, or proxy server. Clickstream analysis is useful for web activity analysis, software testing, and market research. In Web traffic monitoring, the MAWILab [FBAF10] project aims for instance at providing a dataset to help researchers evaluate their traffic anomaly detection methods. These methods are however restricted to IP traffic analysis and cannot be generalized to all link streams. [CHN08] proposes an event detection method for search engine queries: based on query topics clustering in a 2-d space, it is able to find real-world events which happened in 2006 like the Los Angeles Marathon. This method based on textual analysis thus cannot be generalized to all link streams and is therefore not relevant to our problem definition.

4.1.4 Conclusion

Few methods deal exclusively with link streams. The state of the art relies on outlier detection in static graphs, and in series of graph snapshots observed at a given frequency. Our study on link streams is thus original. Moreover, existing methods address various problem categories:

- the detection of anomalous graphs in a series of graph snapshots.
- the detection of anomalous evolution of sub-graphs or communities.
- the identification of nodes and links responsible for abnormal evolution of the complete or partial graph.

As we stand in an exploratory approach with no knowledge on the observed system, we do not want to make any hypothesis on its behavior, and attempt to define the most generic framework possible. This way we may eventually adapt it to more specific questions like the dynamics of communities. As such studies presuppose the existence of observable communities (hence a system property), we put these questions aside to focus on a more general setup: a generated time series describes the evolution of a graph property over time, and our goal is to detect significant changes in it. Current methods of time series analysis are not acceptable though, because they usually rely on a fixed threshold parameter (found by a prior knowledge) to spot as outlier the values beyond the threshold, or they rely on a time series model (e.g. ARMA) to spot as outliers the values which diverge too much from the model (given a threshold). In the following Section, we propose a generic method to analyze

univariate samples and time series with no prior hypothesis on data, which can be applied to time series of link streams statistics.

4.2 Outskewer

We propose a new unsupervised univariate method that reliably detects multiple outliers on either static or temporal datasets given the following setting, which is known to be challenging: values may not be independent and identically distributed; we have no prior knowledge of the underlying process which generated the data, or of the probability distribution; in time series, regime changes may exist due to the evolution of the normal behavior (non-stationarity), and also outlying values which deviate globally or locally from the main trend. We finally want an on-line method for real-time monitoring.

Our method relies on the notion of *skewness* of distributions (see Section 4.2.1) and its evolution when extremal values are removed, which we call *skewness signature*; we use this to detect outliers in multisets of numbers and in time series. Our method has the following advantages: (a) it uses a novel approach based on the study of the skewness of distributions, and is easy to interpret; (b) it looks for outliers only when the notion of outlier is relevant in the considered dataset; (c) it is easy to use, as the only parameter is the size of the time window for multi-scale analysis of time series, and (d) it may be used on-line.

4.2.1 Skewness Signature

We consider a multiset (i.e. a set in which members may appear more than once) X of n values. The distribution of these values is the fraction P_x , for each x , of values in X which are equal to x . Such distribution samples are basically described by their mean $\bar{x} = \sum_{x \in X} (x/n)$ and standard deviation $\sigma = \sqrt{1/(n-1) \cdot \sum_{x \in X} (x - \bar{x})^2}$. Going further, the sample skewness is a measure of distribution asymmetry, and can be estimated by:

$$\gamma(X) = \frac{n}{(n-1)(n-2)} \sum_{x \in X} \left(\frac{x - \bar{x}}{\sigma} \right)^3.$$

Intuitively a negative skewness indicates a tail on the left of the distribution more pronounced than the one on the right, while a positive skewness means the converse, see Figure 4.8. If no tail exists, i.e. all values are equal, $\gamma(X)$ is undefined because $\sigma = 0$. If both tails exist on each side and are equal, $\gamma(X) = 0$. For normal distributions ($P_x = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$), $\gamma(X) = 0$, while for Pareto distributions ($P_x = \frac{ab^a}{x^{a+1}}$ where $0 < b \leq a$), $\gamma(X) > 0$. Examples of unimodal skewed distributions are shown on Figure 4.8.

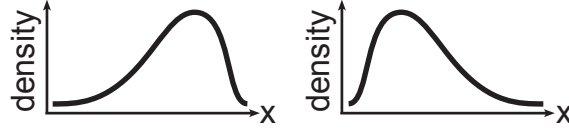


Figure 4.8: Example of negative (left) and positive (right) skewed distributions.

The skewness has the interesting feature to be **influenced by values which are far from other values**, because it is based on the cubed distance from values to the mean. Hence its value changes a lot if they are removed. We show now how to use this feature for outlier detection.

We consider the evolution of the skewness of a distribution of values in a multiset X when extremal values are removed one by one from X , which we call the *skewness signature* of X . The extremal value of X , denoted by $e(X)$, is:

$$e(X) = \begin{cases} \max(X) & \text{if } \gamma(X) > 0, \text{ and} \\ \min(X) & \text{otherwise.} \end{cases}$$

In practice, the skewness is almost never equal to zero, hence always choosing $\min(X)$ in the case where $\gamma(X) = 0$ induces a negligible bias.

We define a series of multisets as follows: $X_0 = X$, $X_i = X_{i-1} \setminus \{e(X_{i-1})\}$, for all $i > 0$. In other words, X_i is the multiset obtained by removing one occurrence of the largest (resp. smallest) value of X_{i-1} if the distribution of values in X_{i-1} has a positive (resp. negative or zero) skewness. Finally, we define the skewness signature as the function $s(p, X) = \gamma(X_{\lfloor p \cdot n \rfloor})$, where n is the size of X and $X_{\lfloor p \cdot n \rfloor}$ is the multiset obtained from X by removing $\lfloor p \cdot n \rfloor$ extremal values, i.e. a fraction p of extremal values.

For example, if $X = \{-3, -2, -1, -1, 0, 1, 2, 3, 7\}$, values 7, 3, 2, -3, 1, -2, 0 are removed in this order², and the values of the skewness signature are 1.09, 0.22, 0.17, 0, 0.40, 0, 1.73.

The skewness signature may be used to find outliers in unimodal distributions because outliers lie at their extremities, and because skewness is sensitive to the removal of outliers.

4.2.2 Our Method

4.2.2.1 Outlier Detection in Samples

Our method relies on the following hypotheses: outliers are extremal values which cause the skewness to be far from zero; the skewness signature converges to zero (i.e. the distribution becomes more symmetric) when outliers are

²Values are removed until $\gamma(X)$ is not computable: our skewness estimator is only relevant for datasets with at least 3 values.

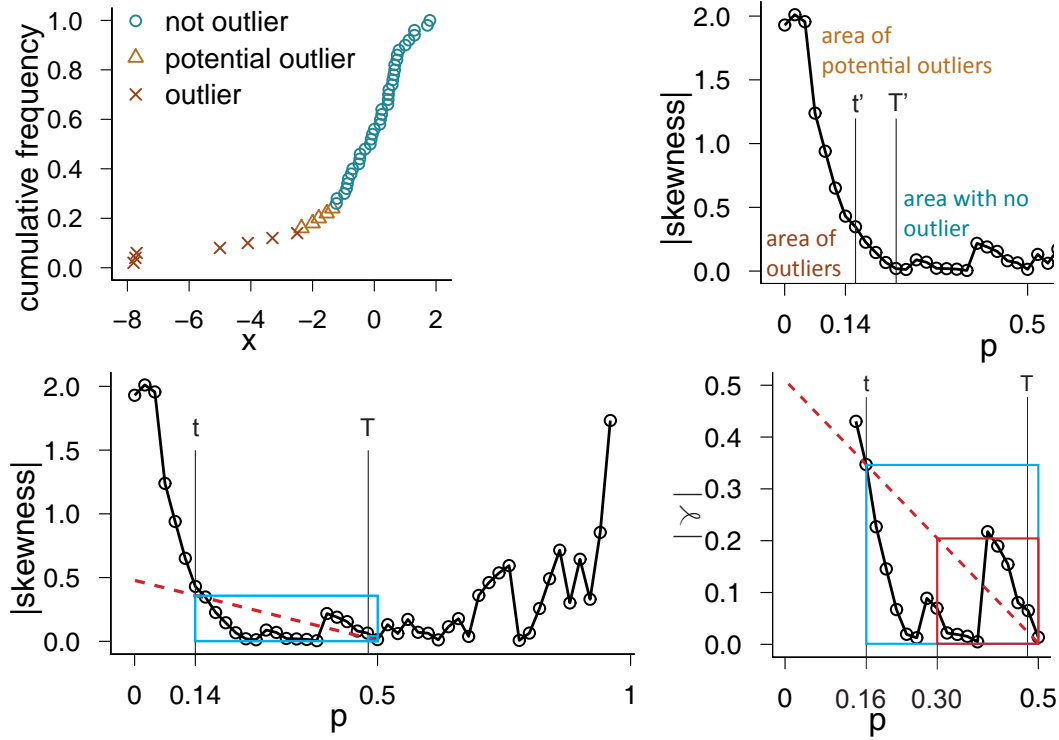


Figure 4.9: Example of 50 values with 7 outliers and 5 potential outliers (counter-clockwise from top-left): cumulative distribution; absolute values of the skewness signature; zoom on it; absolute values of skewness for which outliers and potential outliers are detected. We obtain $t = 0.14$, $T = 0.48$, $t' = 0.16$, $T' = 0.24$.

removed one by one. Therefore, the distance of the skewness to zero can be used to identify outliers. Extremal values which cause this distance to be too large should be classified as outliers. But how is it possible to determine that the distance is too large without making any hypothesis on the dataset?

We propose to consider the distance relatively to the proportion of extremal values removed: the more extremal values removed, the closer to zero the skewness is expected to be. For any $p \in [0; 0.5]$ we say that s is **p -stable** if and only if $|s(p', X)| \leq 0.5 - p$, for all $p' \in [p, 0.5]$. We do not consider values of p larger than 0.5 because this corresponds to a removal of more than half of all values; in such situations, the skewness has little to do with the original data, and it may vary much if too many values are removed.

Let t be the smallest value such that s is t -stable, and T be the largest value such that s is T -stable. When s is never p -stable for any p , t and T do not exist. This case indicates that it is irrelevant to look for outliers in the given dataset, according to our notion of outlier; in this case our method classifies all values in the dataset as **unknown**. Otherwise we find outliers as

follows.

We denote the smallest and largest numbers in X_i by $\min_i = \min(X_i)$ and $\max_i = \max(X_i)$. Then, $\min_{[p..n]}$ (resp. $\max_{[p..n]}$) is the smallest (resp. largest) remaining value when a fraction p of all values has been removed. Let t' (resp. T') be the smallest value of p such that $|\gamma(X_{[t'..n]})| \leq 0.5 - t$ (resp. $|\gamma(X_{[T'..n]})| \leq 0.5 - T$). Our method concludes as follows: below $\min_{[t'..n]}$ and above $\max_{[t'..n]}$, values are **outliers**; between $\min_{[t'..n]}$ and $\min_{[T'..n]}$ included (resp. $\max_{[t'..n]}$ and $\max_{[T'..n]}$), values are **potential outliers**; values are **not outliers** otherwise. Notice that when $t' = T'$, $\min_{[t'..n]} = \min_{[T'..n]}$ (resp. $\max_{[t'..n]} = \max_{[T'..n]}$). In this case, values equal to $\min_{[t'..n]}$ (resp. $\max_{[t'..n]}$) are potential outliers. Figure 4.9 illustrates our method on an example.

4.2.2.2 Dynamic Extension for Time Series

Our method may be used on time series representing the evolution of a system's property. Let $\{x_0, x_1, \dots, x_n\}$ be a time series. We consider the multisets which contain w values: $X^i = \{x_{i-w+1}, \dots, x_i\}$. Any value x_i of the series belongs to $X^i, X^{i+1}, \dots, X^{i+w-1}$. We use our method on all these w multisets, and consider the final class of x_i to be the one which occurs most often among these w classifications. In case of equality, we give priority of *outlier* upon *potential outlier* upon *not outlier*, because we prefer to detect too much outliers than too few. This choice is straightforward but can be modified to lower the risk of false positives detection. Outliers in that case are called *events*.

4.2.3 Experimental Validation

The validation of outlier detection methods is difficult because of the various outlier definitions, hypotheses and use cases [Bg05]. Labelled datasets also raise the issue of prior criteria to label the data. We consider that our method should detect outliers, if any, if the notion of outlier is relevant for the given dataset. In particular, we consider for our experimental validation the following cases: (a) distributions like Power laws (e.g. Pareto and Zipf's law) commonly contain extremal values far from the mean (i.e. *heterogeneous*), so it is erroneous to consider them as outliers, moreover heterogeneous distributions are asymmetrical so our method should conclude that looking for outliers in them is irrelevant; (b) normal distributions are symmetrical and extremal values far from the mean are uncommon (i.e. *homogeneous*), so no outlier should be detected but these extremal values when they occur; (c) half-normal distributions ($P_x = \frac{2a}{\pi} e^{-x^2 a / \pi}$, where $a > 0$), which are basically the absolute of normal distributions with mean equal to 0, are asymmetrical but homogeneous, so this case is ambiguous and should be unclear for our method as well; (d) symmetric Pareto distributions ($P_x = \frac{ab^a}{2} |x|^{-1-a} 1_{|x|>b}$, where $0 < a < 2$ and $b > 0$), which are basically the mirror symmetric of

Pareto distributions about the vertical axis, are symmetrical but heterogeneous, so we study the behavior of our method in this case.

We first study the relevance of our method on these four distributions, and we study the effect of the sample size. Then we study the performance of our method to detect outliers, and evaluate the rate of true outliers and false outliers detected. We finally study the behavior of our method when regime changes occur in temporal data.

4.2.3.1 Relevance

Our method is applicable if and only if the given dataset is p -stable for at least one value of p between 0 and 0.5. A necessary condition for this is that $|s(0.5, X)| < 0.5$. We show in this section that this is true for normal distributions (even with a few outliers) and false for Pareto distributions, which is the expected behavior: normal distributions are symmetrical and homogeneous and Pareto distributions are asymmetrical and heterogeneous.

We study the behavior of s on normal $\mathcal{N}(0, 1)$ and Pareto (shape=6, location=2) probability distributions³. For each one, we randomly generate 1,000 samples of 100 numbers to obtain skewness signatures; we compute and plot the skewness signature of each sample in Figure 4.10. We observe that the values of normal signatures oscillate around zero, whereas the values of Pareto signatures globally decrease and are above zero until $p \approx 0.5$. The cumulative frequency distributions of $s(p, X)$ on Figure 4.10 confirm these observations. We also computed the skewness signatures of normal and Pareto distributions with various parameters, and also various symmetrical distributions⁴ which we do not present here due to space constraints. All of them exhibit patterns similar to normal signatures.

It is clear that the probability for Pareto skewness to be within $[-0.5; 0.5]$ increases with p . We estimate $\mathbb{P}(|s(0.5, X)| < 0.5)$ on 1,000 Pareto and 1,000 normal samples. We obtain that this probability is equal to zero for Pareto samples, and is greater than 0.95 for normal samples. We conclude that our method is able to characterize symmetrical and homogeneous versus asymmetrical and heterogeneous distributions at a confidence level of 0.95. Moreover, the addition of some outliers in these distributions produces almost the same signatures than without outliers, because extremal values are firstly removed. Therefore existing outliers do not notably change the characterization.

Let us study the evolution of $s(0.5, X)$ when the sample size n varies. We generate 1,000 normal and Pareto samples for each value of n between 3 and 1,000, then compute $s(0.5, X)$ for each sample, and we finally obtain the quartiles, min and max of the values of $s(0.5, X)$ at each n . We observe in Figure 4.11 that the results converge to zero for the normal distribution,

³Other parameters lead to similar results.

⁴Cauchy, Laplace, some Gamma and Weibull distributions.

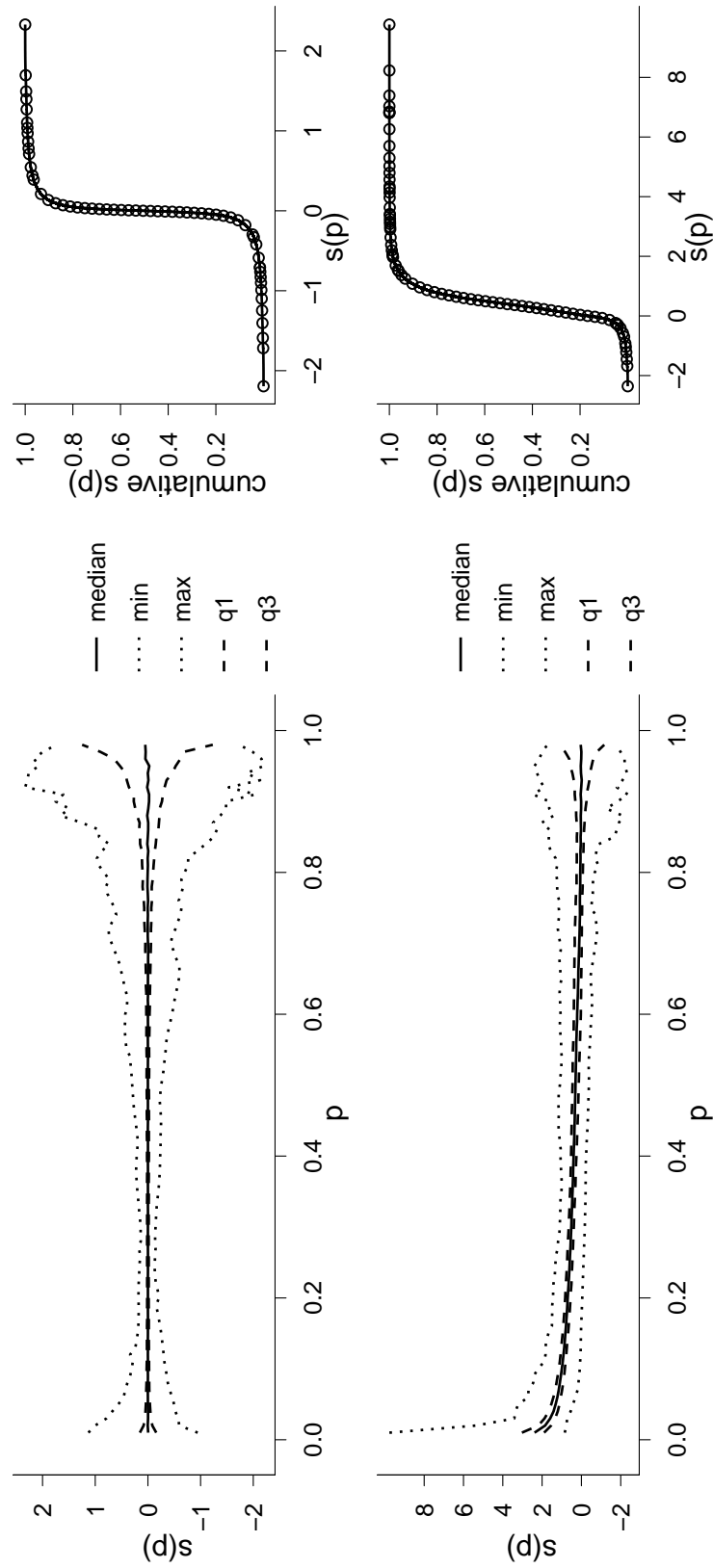


Figure 4.10: Quartiles, min and max of $s(p, X)$ on 1,000 normal (top) and Pareto (bottom) samples with the cumulative frequency distributions of $s(p, X)$.

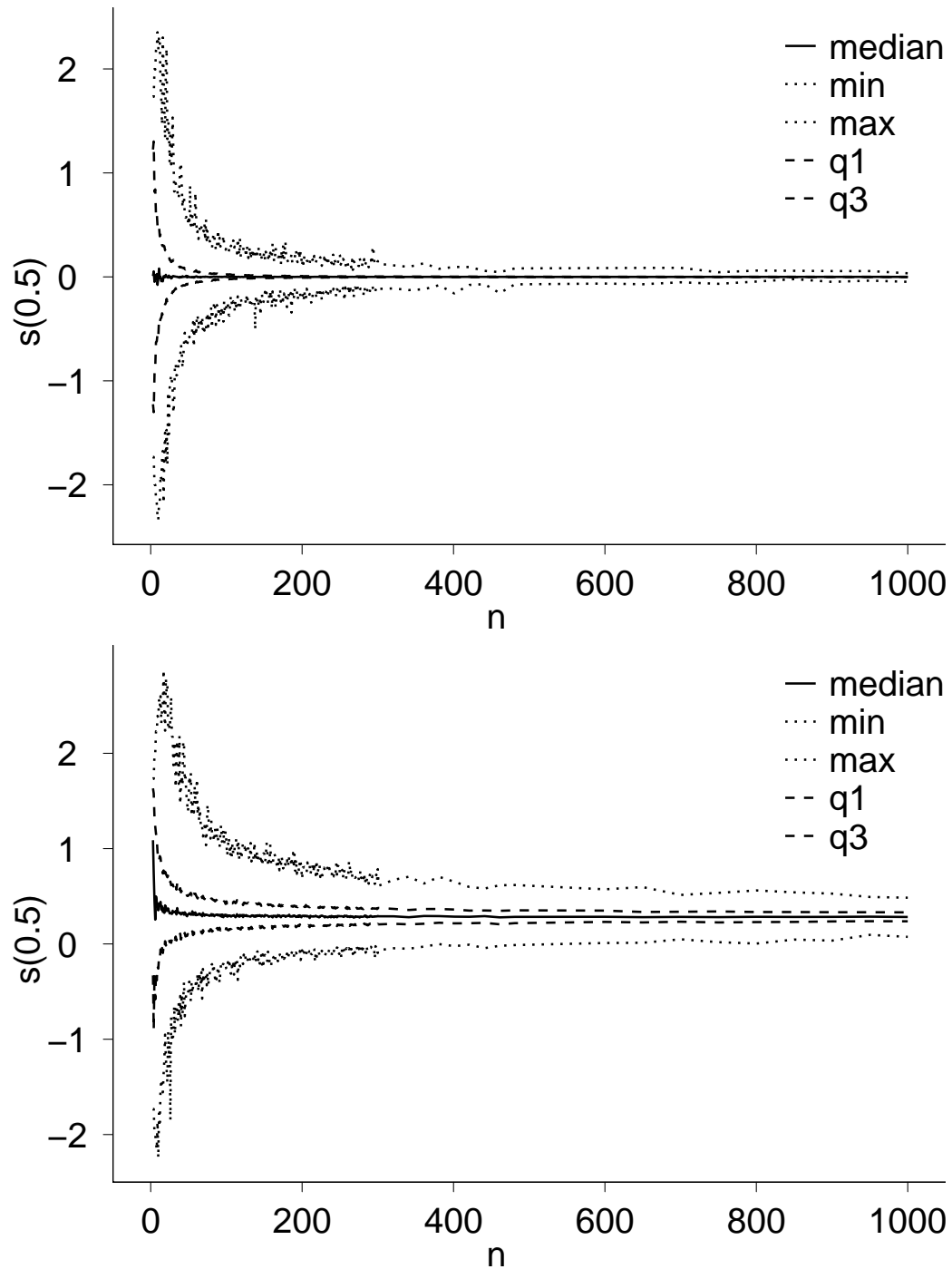


Figure 4.11: Quartiles, min and max of $s(0.5, X)$ as a function of n for normal (top) and Pareto (bottom) distributions.

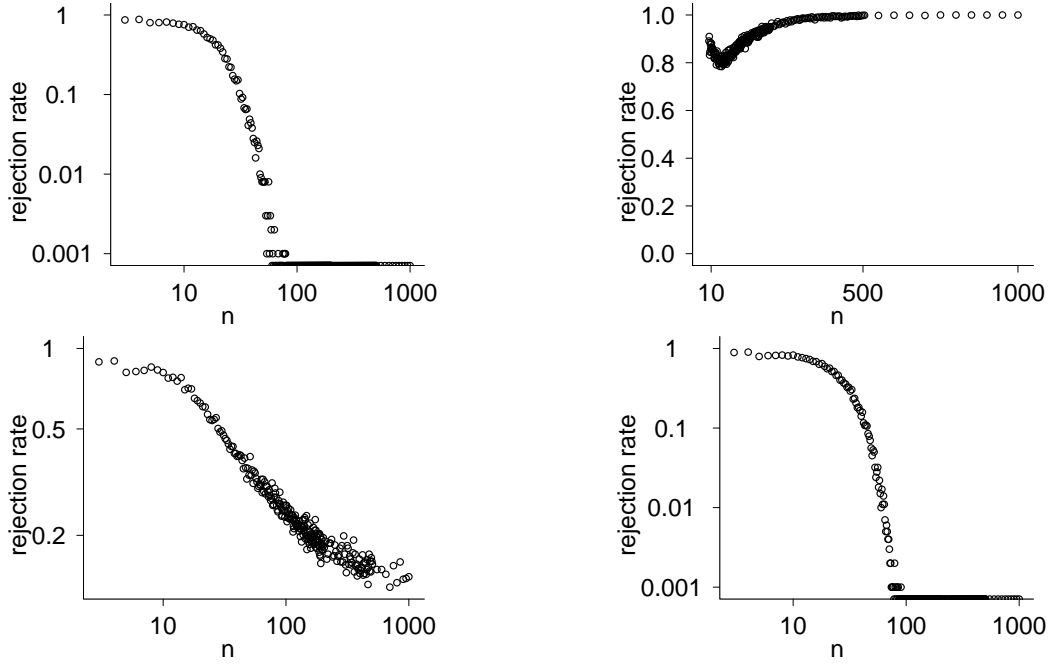


Figure 4.12: Fraction of samples for which s is never p -stable as a function of the sample size n , for normal (top left), Pareto (top right), half-normal (bottom left), and symmetric Pareto (bottom right) distributions.

and to ≈ 0.3 for the Pareto distribution. Thus, increasing n should lead to a better characterization.

We verify this hypothesis by evaluating the rate of samples where s is never p -stable, for 1,000 normal and Pareto samples for each size n . We observe in Figure 4.12 that it seems to follow a fast decrease for normal samples. For $n \geq 37$, less than 5% of normal samples are incorrectly characterized, and less than 5% for $n \geq 55$. We also observe that it increases with n for $n > 50$ on Pareto samples. The minimum is 79% at $n = 52$, is around 85% at $n = 100$, around 95% at $n = 240$, and above 99.5% for $n > 500$.

We also evaluate this rate for half-normal and symmetric Pareto samples. We observe in Figure 4.12 that it seems to follow a fast decrease for symmetric Pareto samples, but a slow decrease for half-normal samples. This result is not surprising because the theoretical skewness of half-normal distributions⁵ is ≈ 1 , and the skewness decreases slowly when extremal values are removed one by one. As expected, our method has unclear results in this case.

We conclude that our methods characterizes samples with size 100 very well, and is excellent on samples of size 1,000. Our method also considers that the symmetric Pareto distribution should contain no outlier.

In addition, we study the skewness range within which our method consid-

⁵ $\gamma = (\sqrt{2} \cdot (4 - \pi)) / (\pi - 2)^{3/2}$

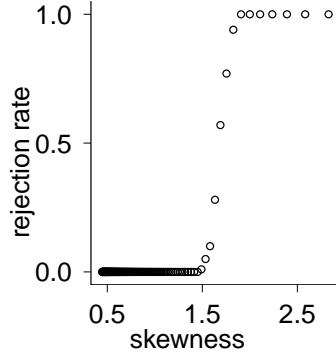


Figure 4.13: Fraction of samples for which s is never p -stable as a function of the skewness for Gamma samples (shape varying from 0.3 to 20).

ers s to be p -stable at least once. We vary the shape parameter of a Gamma distribution (thus its skewness) to incrementally generate 1,000 samples of 100 numbers for each skewness value, from Pareto-like samples to normal-like samples, and compute the rate of s that are p -stable at least once for each skewness. We remind that s is p -stable if and only if $|s(p', X)| \leq 0.5 - p$, for all $p' \in [p; 0.5]$. The result in Figure 4.13 shows that s is always p -stable at least once for samples of skewness below 1.5, and never p -stable for samples of skewness above.

4.2.3.2 Performance

We study the effect of the sample size on outlier detection in normal, Pareto, half-normal and symmetric Pareto distributions. We generate 1,000 samples for each distribution and size n , then we detect outliers on each sample. Normal and Pareto samples contain no outlier by definition, so no outlier should be detected; they are called *false outliers*.

We observe in Figure 4.14 that the rate of false outliers is low, with at most 3% for the normal distribution and at most 5% for Pareto. This rate decreases when n increases to be less than 1‰ above $n \approx 100$ for the normal distribution, and above $n \approx 500$ for the Pareto distribution. We also evaluate the rate of outliers detected for the symmetric Pareto distribution: reaching 5% at most, it seems to follow a fast decrease when n increases, to reach 1‰ at $n \approx 1000$. For the half-normal distribution, this rate is between 8% and 12% for $n > 100$, and is consistent with the fraction of samples for which s is never p -stable. We conclude that our method detects few false outliers on samples of size 100, and almost none on samples of size 1,000, which is an excellent performance; it rarely detects outliers on symmetric Pareto samples, which is the expected behavior regarding the characterization.

Now we estimate the ability to detect true outliers by generating a sample of size 1,000 composed of a normal sample of variance equal to 1 and a uniform

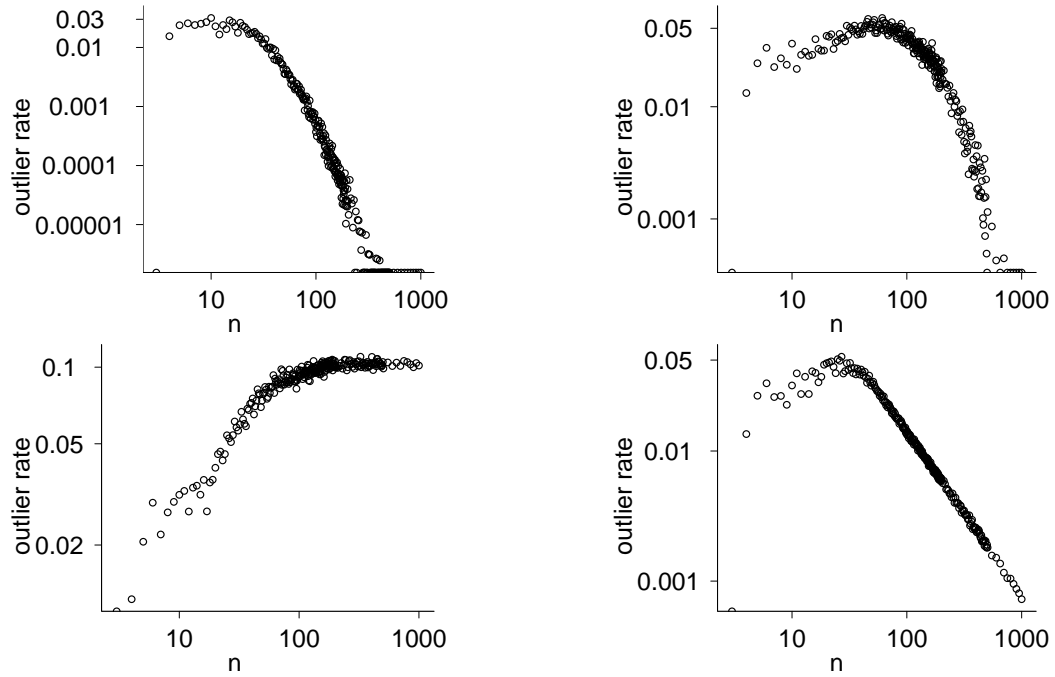


Figure 4.14: Fraction of sample points classified as outlier as a function of n for normal (top left), Pareto (top right), half-normal (bottom left), and symmetric Pareto (bottom right) distributions.

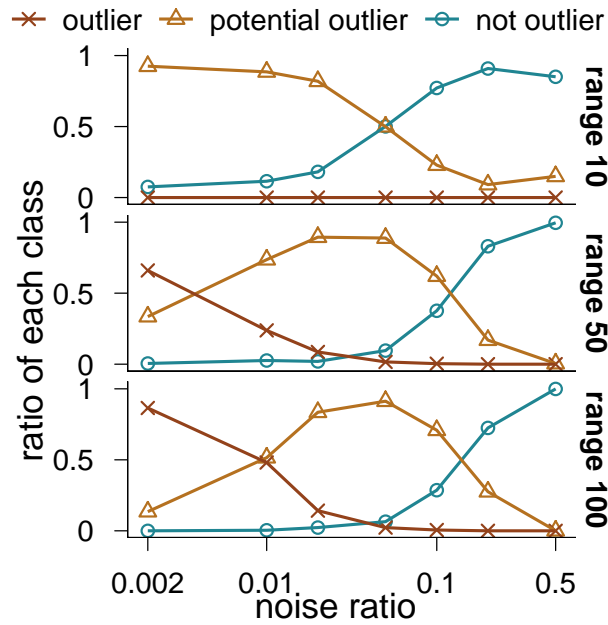


Figure 4.15: Ratio of noise points detected as outliers, potential outliers and not outliers as a function of the proportion of noise, for different noise ranges.

sample (called the *noise*) of size varying from 0.2% to 50% of the total number of values. We then count the number of noise points which are classified as outliers and potential outliers. It is the worst case because the initial skewness is close to zero and outliers are uniformly distributed around the mean with no gap between them and the rest of the distribution. This is also a way to evaluate the robustness of our method against a problem known as the *masking* effect [BK85], occurring when some outliers are not detected because of the presence of other outliers close to them.

We generate uniform samples of various ranges (i.e. largest minus smallest value). The range of normal samples of size 1,000 is roughly 6 and the range of samples of size 10^6 is roughly 10, so we select noise ranges larger than this: 10, 50 and 100. We observe in Figure 4.15 that noise points very close to the signal points (range 10) are classified as potential outliers. Larger ranges increase the number of detected outliers. We also see that the lower the noise, the higher the power to detect true outliers. However almost no outlier can be detected with more than 10% of uniform noise.

4.2.3.3 Regime Changes

Regime changes are change points in time series that are observed by sudden changes of the mean. When they occur we are faced with non-trivial distributions. We study now how our method deals with them. We simulate a stream of values by generating two normal samples of size 110 with mean equal to 0 and 3 respectively. t indicates the order of appearance of the values. Figure 4.16 shows our method applied dynamically with a sliding window of size $w = 100$. The outlier status of values is unknown at the beginning. At the end, none of them are outliers except one potential outlier. Our method is hence robust against regime changes. Notice that 72 values are classified as potential outliers when our method is applied to the whole dataset at once.

4.2.4 Real-World Applications

In this Section we show the application of our method to three various real-world datasets. See Section 2.2 for the full description of the datasets.

4.2.4.1 French Population in the 20th Century

This first dataset is not related to graphs. It is the time series of the evolution of the French population during the 20th century. In order to detect anomalous increase rates of the population, we compute the difference over the years, and apply our method on this time series. We see in Figure 4.17 that the drops happening during the two world wars (1914-1918 and 1939-1945) are clearly detected, as well as the “baby boom” effect in the sixties.

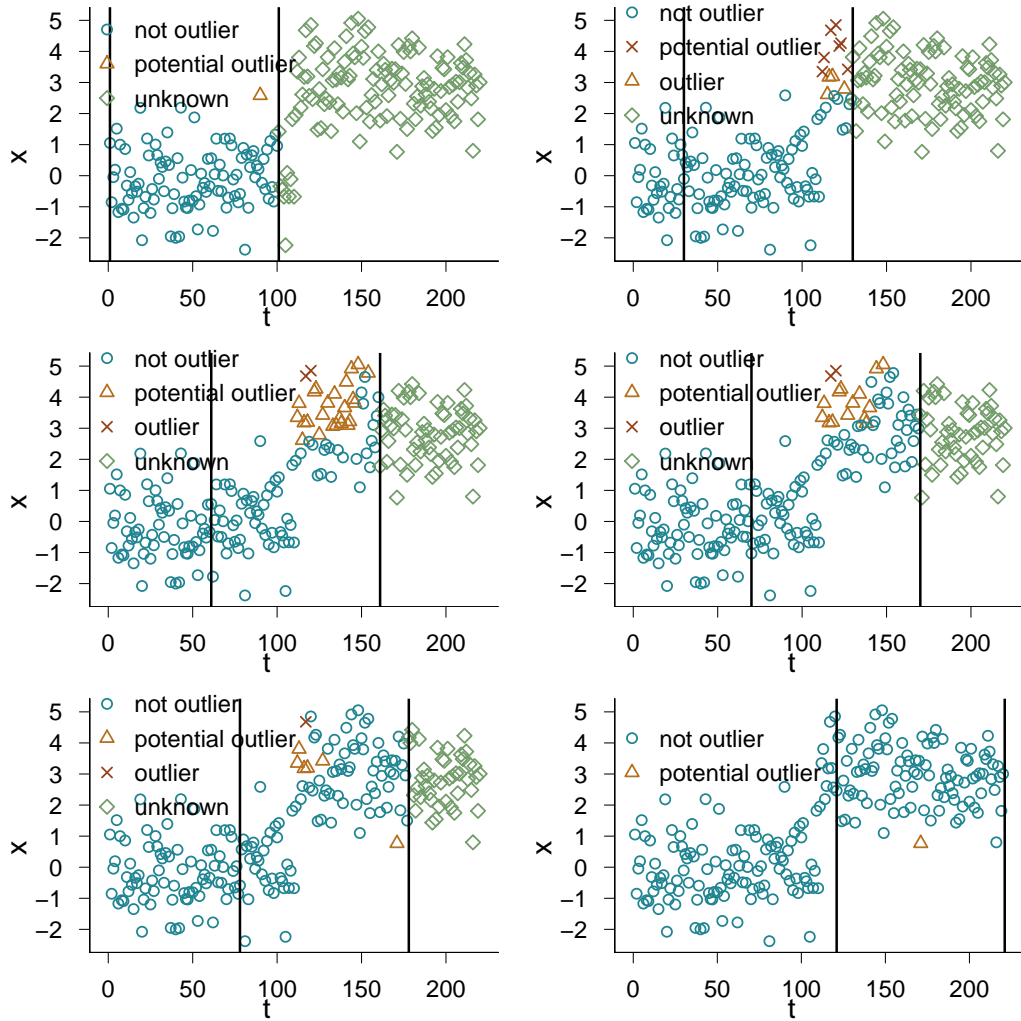


Figure 4.16: From top-left to bottom-right, evolution of the outlier status of values in a time series of size $n = 220$, with one regime change (mean value changing from 0 to 3). Vertical lines indicate the time window boundaries between which outliers are detected.

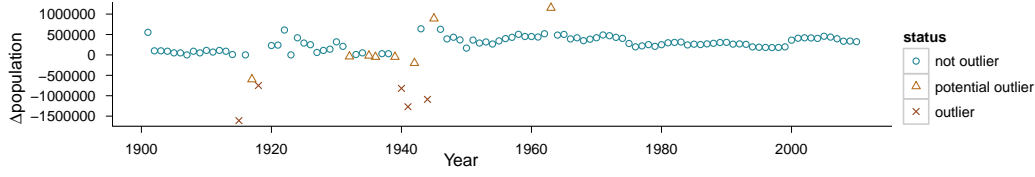


Figure 4.17: Difference of the number of inhabitants compared to the previous year in France during the 20th century.

4.2.4.2 Dynamics of Internet Topology

The second dataset is a series of measurements of IP addresses observed in the network neighborhood of a computer. We applied our method to this INTERNET dataset, which is a series of graphs.

The most natural idea to detect events in the dynamics captured by a radar measurement from a given monitor certainly is to study the number N_i of nodes observed at each round i . We plot it for a typical case in Figure 4.18. Clear outliers appear under the form of sharp decreases of N_i for some values of i , but this brings little information because they may be due to losses of connectivity by the monitor. Except from these statistical outliers, which are detected by our method, the number N_i of nodes observed at each round i in Figure 4.18 is very stable.

We thus compute the number of distinct nodes seen in five consecutive rounds to avoid the outliers which only reveal losses of connectivity in one round of measurement. We observe events in the dynamics shown in Figure 4.19, where many decreases existing in Figure 4.18 have disappeared. Figure 4.19 is well centered around a typical value, but still exhibits sharp increases and decreases. This means that these outliers, which were also detected by our method, may reveal real events in the dynamics of this network. Outliers above the typical values indicate a sudden appearance of many new nodes in the network, while outliers below the typical values may indicate longer losses of connectivity or a sudden disappearance of many nodes.

Our approach is hence relevant for studying the evolution of ego-centered views of the internet topology, and for raising automatic alerts in real-time when significant changes of connectivity occur.

4.2.4.3 Search Engine Queries

We finally applied our method on the P2P dataset, which is a link stream.

In order to study the number of queries related to the film *Harry Potter and the half blood prince*, we filtered the queries to retrieve only those which contain the word sequence “half blood prince”. Then for every 10 minutes we counted the number of queries made during the last hour of measurement. Outliers were finally detected using a sliding window of size $w = 1,008$ (7

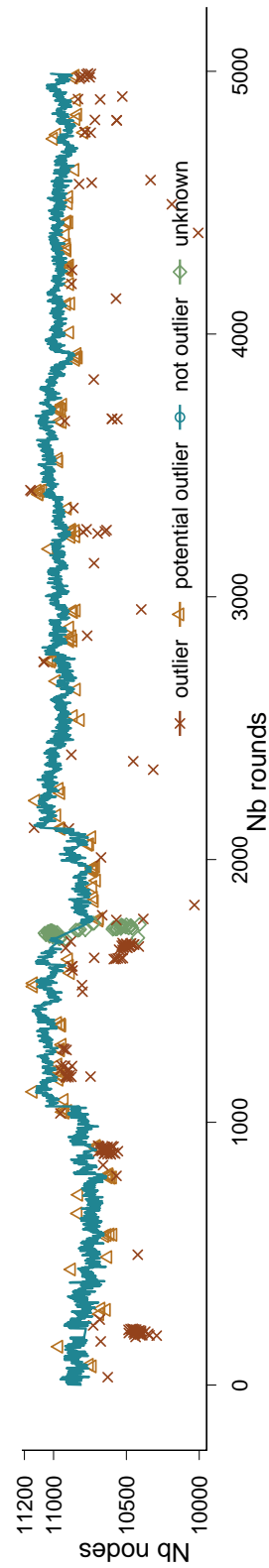


Figure 4.18: Number of nodes at each round of radar measurement; outliers (i.e. rounds of measurement) are detected using a sliding window of 100 rounds (25 hours).

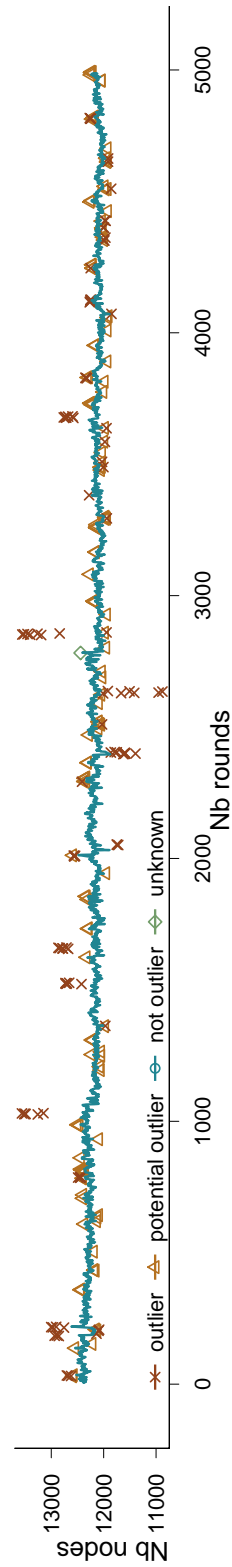


Figure 4.19: Number of nodes in the union of 5 consecutive rounds of radar measurement; outliers (i.e. union of 5 rounds of measurement) are detected using a sliding window of 100 rounds.

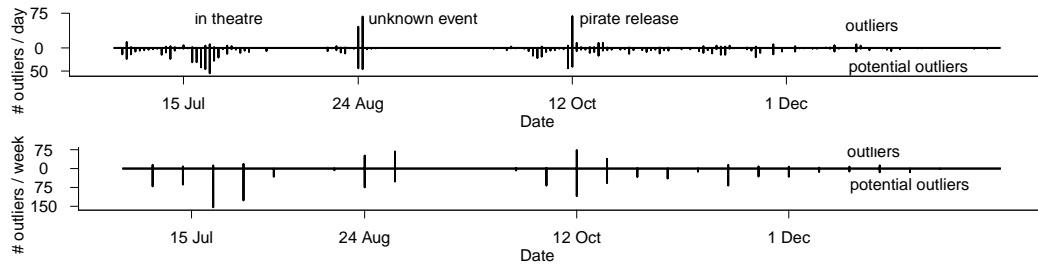


Figure 4.20: Number of outliers and potential outliers each day (top) and each week (bottom) in the number of search queries containing the word sequence “half blood prince”.

days) to capture meaningful events at the scale of one week. We plot in Figure 4.20 the number of outliers and potential outliers observed each day and each week. The scale of a day seems adequate enough for observing fast increases of user queries.

We identify three main events: we observe many values marked as potential outliers during the week after July 15, 2009, when the film was out in theatres. Then an unknown event appears from August 23 to 25, when almost all values are outliers. The last automatically detected event, from October 10 to 12, coincides with the release of a pirated version of the film on October 10 on BitTorrent, another P2P network, as discovered by searching on <https://thepiratebay.se>. We suppose that this release was made from a promotional DVD, because the commercial DVD was released on December 7 only; we observe no noticeable event on this day.

Our approach is hence relevant for studying logs of search queries, and for detecting bursts of queries related to a same topic.

4.3 Conclusion

We have proposed the *Outskewer* method to detect statistically significant outliers in samples and events in time series. *Outskewer* uses a novel approach based on the study of the distribution skewness. Our method is easy to interpret because values are classified as *outliers*, *potential outliers* or *not outliers*. The class of all values is unknown when the notion of outlier is not relevant in the considered dataset. Our method is also easy to use because it requires no prior knowledge on the data, and the only parameter is the size of the time window for multi-scale analysis of time series. Moreover, it may be used on-line to detect events on the fly.

We have applied it to three datasets representative of diverse use cases: evolution of the French population during the 20th century, evolution of ego-centered views of the internet topology, and logs of queries entered into a

search engine. We clearly identify events in the evolution of ego-centered views of the internet topology as shown in Figure 4.18 and Figure 4.19. We also automatically detect the release of a pirated version of a film in a P2P system, through the queries entered by users in the search engine, as show in Figure 4.20.

The strengths of our method are its statistical reliability with very few data points (100 data points are enough), and its ability to deal with regime changes in time series as shifts or normality. However we still have to study its algorithmic complexity.

Our method opens the way to further investigation of the use of the skewness to detect multiple outliers in samples, and to detect events at different time scales in time series. But more importantly, when applied to a time series of a graph statistic, the detected outliers provide time points of interest to be investigated later. *Outskewer* may therefore be used to process large link streams before a more local visual analysis, focused on these moments. It is thus mandatory to characterize the behavior of our method on such data by studying the impact of time unit and time scale on the detected events. This is the matter of the following Chapter.

CHAPTER 5

Intrinsic Dynamics

Contents

5.1	Related Work	102
5.1.1	Notions of Topology Dynamics	102
5.1.2	Time Scale	103
5.1.3	Time Unit	107
5.2	Generalization of <i>Outskewer</i>	108
5.2.1	Use of a Sliding Time Window	108
5.2.2	Choice of the Time Unit	110
5.2.3	Sliding Window Width (Time Scale)	113
5.3	Application: Study of Github Internal Links	116
5.4	Conclusion	120

In the previous Chapter we have proposed an automatic method called *Outskewer* to detect statistically significant events in time series. We have rigorously validated it, and applied it to three different datasets.

In this Chapter we generalize the application of *Outskewer* to link streams. This generalization is not trivial and raises multiple questions on the way times series that show the evolution of graph statistics are generated, and on their characterization for the purpose of event detection. In particular, we show that traditional time units like the second and the hour may hide events because of the impact of cyclical human activity, and we introduce the notion of *intrinsic time*, based on link appearances, to avoid this issue. We also study the impact of time scale on event detection, which is related to the characterization of link streams. We finally apply our (generalized) method to the GITHUB dataset¹.

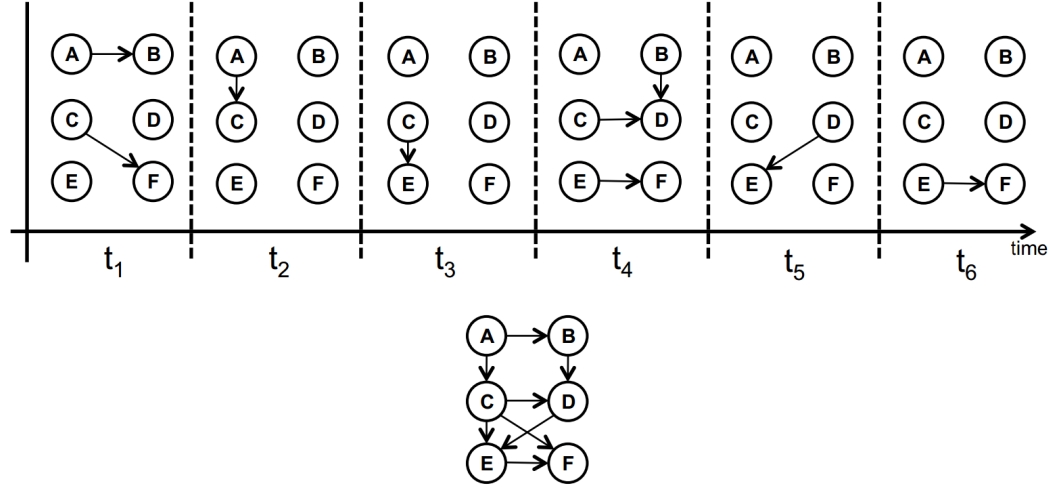


Figure 5.1: Example of graph snapshots and their aggregated graph [TMML11].

5.1 Related Work

5.1.1 Notions of Topology Dynamics

Notions related to the topology of static networks (like node degree, centrality measures, clustering and paths) have been developed before the notions related to dynamic networks. When studying sequences of graph snapshots (see Figure 5.1), researchers have naturally started to observe the evolution over time of these static graphs statistics. For instance [LKF05] has studied a range of different networks using snapshots of static graphs, from several domains, and the authors have focused specifically on the way fundamental network properties vary over time. They show that (i) networks are becoming more dense over time, with increasing average degree (and hence with the number of edges growing super-linearly with the number of nodes); (ii) the densification seems to follow a power-law pattern; (iii) in many cases the effective diameter decreases as the network grows.

Another example is the analysis of a dynamic social network comprising 43,553 students, faculty, and staff in a large university [KW06]. Interactions between individuals are inferred from time-stamped e-mail headers recorded over one academic year and are matched with affiliations and attributes. The authors compute daily probability of links and triangles on a sliding time window of 60 days. They correlate the evolution of these indicators to different populations, and find that network evolution is dominated by a combination of effects arising from network topology itself and the organizational structure in which the network is embedded. However such effects are not studied.

¹See Section 2.2 for the description of the GITHUB dataset.

The common notions of static networks are not sufficient to capture the dynamics of networks over time [TLS⁺13]. Temporal graph metrics can lead to new observations of behaviors at various scales, and provide a better understanding of biases induced by measurement parameters. Multiple appropriate notions have been introduced, for instance the journey (that extends the notion of *path* in static networks) [SQF⁺11, XFJ03, PS11]. Other notions include temporal distance metrics [TMML09], contact duration [ISB⁺11], inter-contact durations [EMS04, CFL09, BC13], temporal reachability graphs [TMML10, WDdACG12a], temporal motifs [KKK⁺11], temporal communities [SG12], temporal node centrality [KA12], link persistence [NTM⁺13], among others. Extensive surveys on temporal metrics are available [CFQS11, HS12].

5.1.2 Time Scale

Time scale (or time resolution) is a critical parameter in many dynamic network analysis methods. Unfortunately, fine-grained temporal connectivity data is often difficult to obtain for real social systems. The traditional approach to study network dynamics in sociology (see, for instance, [Was94]) tends to rely heavily on interaction data self-reported by study participants, which exhibit significant bias and noise [Mar90]. Several recent studies in physics and computer science have utilized web-based or other indirect sources of dynamic social network data [OdC04, VOB05, LKF05, HMPKE07, BWS06]. In general, empirical studies convert the available temporal data into a short sequence of non-overlapping network snapshots). However only a few studies analyze the impact of time resolution on the estimation of diffusion processes and topological properties.

Many studies indeed assume a specific time scale for the measurement of statistical properties [AZY11, AMF10a, LMO08, ZLBM06]. As [Tho12] noticed:

“Too short a window length may result in high variance and sample data that is not representative of the underlying graph structure; too long a window length may have a smoothing effect that hides shorter-term deviations in behavior. Furthermore, different time granularities may be appropriate for different nodes or edges within the same network.”

[CE07] is a study of a highly resolved data set of a dynamic proximity network of 66 individuals [EP06]. The authors show that the topology of this network evolves over a very broad distribution of time scales, that its behavior is characterized by strong periodicities driven by external calendar cycles (see Figure 5.2), and that the conversion of inherently continuous-time data into a sequence of snapshots can produce highly biased estimates of network structure. They suggest a method to find an optimal time scale

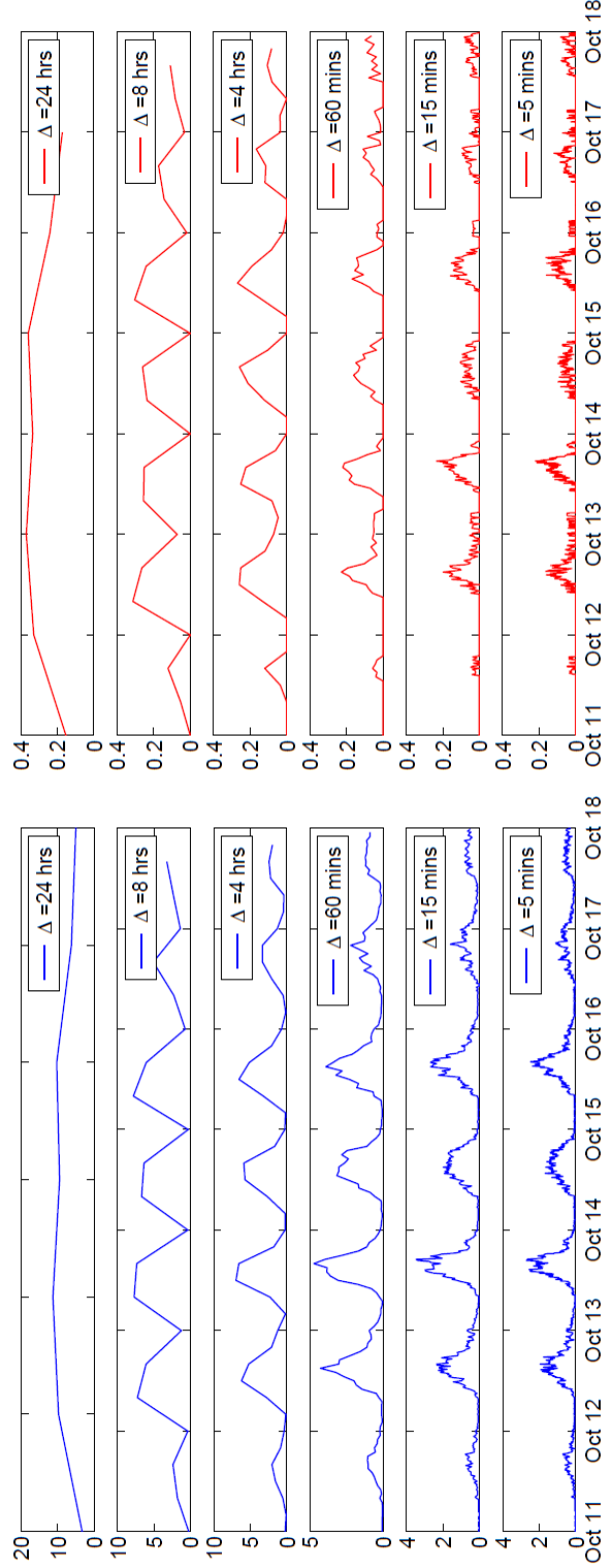


Figure 5.2: The (a) mean degree and (b) mean clustering coefficient of the Reality Mining phone call network [EP06], as a function of time for snapshot rates $\Delta = \{1440, 480, 240, 60, 15, 5\}$ (minutes) during the week of 11 October through 17 October for the core 66 subjects. As Δ grows, under-sampling clearly averages out higher frequency fluctuations [CE07].

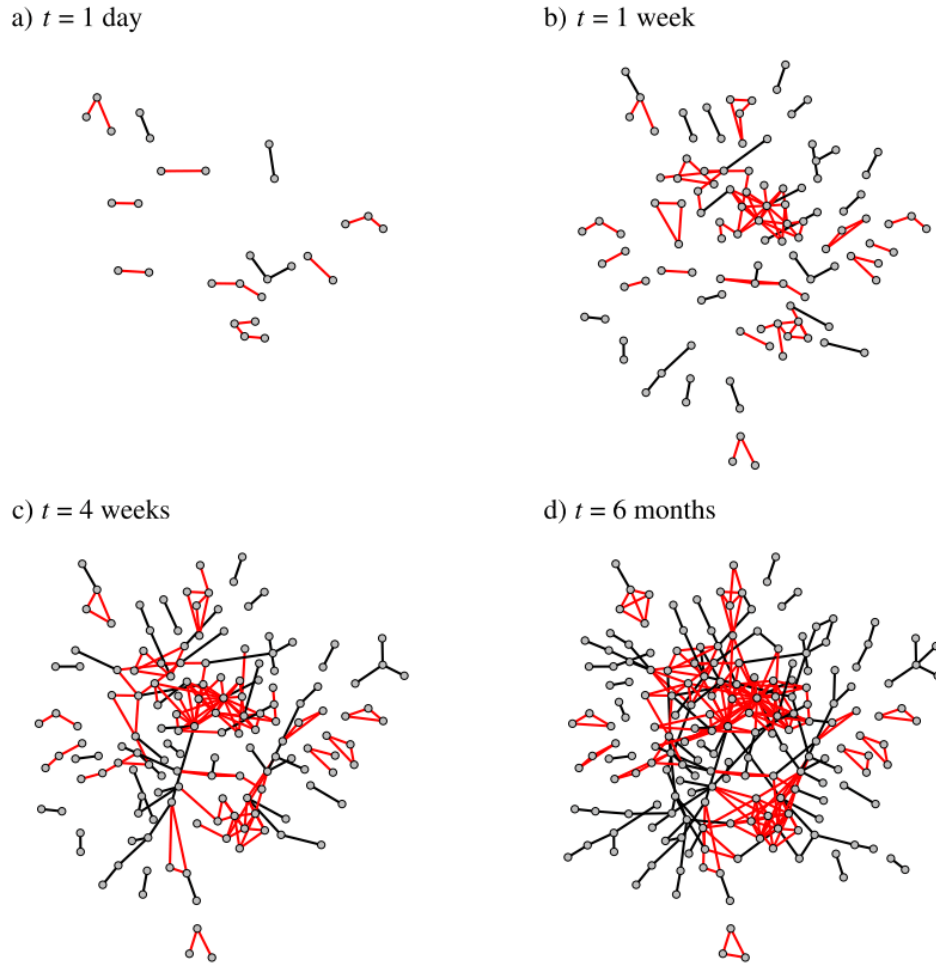


Figure 5.3: Aggregated network at different time scales. Links that participate in triangles in the final 6-month aggregated network are colored red, while the rest are black [KKB⁺12].

that smoothes out high frequency variation while preserving important low frequency structural patterns.

[Pap06] also computes an optimal scale for pattern detection in time series. Assuming specific properties, [ZZJ⁺08] explores a multi-resolution approach to anomaly detection for the internet.

The problem of time resolution is closely related to the sampling rate during a measurement. Using a simulation of a social dilemma game over a dynamic network of various resolutions, [CPN⁺13] shows that the resilience of cooperation among people is hindered when high-resolution time-varying graphs are considered. In particular, high temporal resolution strongly affects the persistence of cooperation in the paradigmatic social dilemmas. As a result, cooperation can emerge and persist also for moderately high time

resolutions. Thus, not only temporal resolution but also temporal correlations across consecutive snapshots are fundamental for cooperation to emerge. These findings suggest that the frequency at which the connectivity of a given system is sampled has to be carefully chosen, according to the typical time-scale of the social interaction dynamics. For instance, as stock brokers might decide to change strategy after just a couple of interactions, other processes like trust formation in business or collaboration networks are likely to be better described as the result of multiple subsequent interactions. Both the over-sampling and the under-sampling of a time-evolving social graph and the use of the finest/coarsest temporal resolution could substantially bias the results of a game theoretic model played on the corresponding network. [RLNZ09] raises as well the issue of downsampling time series for storage, while preserving the capacity to detect anomalies.

Different conclusions may therefore be reached depending on the particular time window over which interactions are aggregated and the size of that window. Thus, determining the appropriate window used to define a network is critical. Choosing the correct window size can be done by measuring lag times between two connections of the same nodes (lagged association rates) as we have seen before [CE07]. Alternatively, window size can be chosen by determining when time series of network statistics constructed from different temporal subsets of the data become stationary [CBWG11]. Also, [BM10] proposes a methodology to estimate the size of the observable window for a rigorous characterization of any network property. Another approach is to use prior knowledge about natural time scales in the system [LMS⁺08, CVdBB⁺10].

Unfortunately, there is currently no consensus on the best method for choosing a window size, because the notion of optimal time scale is much more complex. [KKB⁺12] has investigated the structural features of mobile telephone call networks aggregated over aggregation intervals of increasing lengths. The results show that short aggregation intervals yield networks where strong links associated with dense clusters dominate; the seeds of such clusters or communities become already visible for intervals of around one week. Degree and weight distributions are seen to become stationary around a few days and a few weeks, respectively. An aggregation interval of around 30 days results in the most stable similar networks when consecutive windows are compared. For longer intervals, the effects of weak or random links become increasingly stronger, and the average degree of the network keeps growing even for intervals up to 180 days. The results show that the placement of the time window affects the outcome: for short windows, different behavioral patterns play a role during weekends and weekdays, and for longer windows it is seen that networks aggregated during holiday periods are significantly different. An exploratory analysis is therefore necessary to determine the time scale and time placement of the time window.

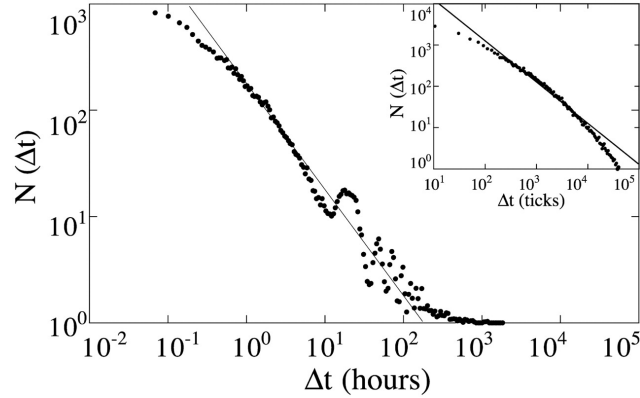


Figure 5.4: Distribution of the response time until an email message is answered. (Inset) The same distribution is measured in ticks, i.e., units of messages sent in the system. Binning is logarithmic. The solid lines follow Δ_t^{-1} and are meant as guides for the eye [EMS04].

5.1.3 Time Unit

Alternative time units are proposed to overcome the issue of the “wall-clock” time units. In [EMS04] the dynamics of 200,000 e-mails is studied, involving 10,000 users during 83 days at a university. The authors have defined Δ_t as the time delay between a message going from A to B and a response going from B to A. Although no clear power law is evident in Figure 5.4, they claim that the behavior can be approximated by $P(\Delta_t) \approx -1$. The appearance of a peak ranging from $\Delta_t = 16$ hours to $\Delta_t = 24$ hours can be explained by sociological behavior involving the time (usually 16 hours) between the moment when people leave work and when they return to their offices. This (already weak) peak disappears when considering a “tick” of the system (i.e., a message sent) as the basic time unit. Choosing the basic tick of the clock (the sending of a message in the network) as a variable time unit smoothens many features (as in Figure 5.4 Inset). In particular, the slowing down of the network over nights and weekends is eliminated. This result is particularly important to our study, because as we explain later, we also face cyclical behavior of temporal properties in our datasets.

More recently, [GPCB13] has focused on the probability distribution of arrival times for the diffusion process (SI model) unfolding over a temporal network. In terms of “wall-clock” time, the arrival time at a given node is defined as the time elapsed between the start (seeding) of the diffusion process and the time when the process reaches the chosen node. [PBC⁺11] has shown that the distribution of these arrival times is extremely sensitive to several heterogeneities of the empirical data with regards to the seeding time. In general, it displays strong heterogeneities due to the non-stationary and bursty behavior of empirical temporal networks that cannot be captured by simple

statistical models. Thus, the authors have shifted to a node-specific definition of “time”: each node is assigned its own “*activity clock*” that measures the time that this node has spent in interaction or, similarly, the number of contact interactions that it has been involved in. The “time” measured by this clock does not increase when the node is isolated from the rest of the network. The “*arrival time*” of the epidemic process at a given node is defined as the increase of its activity clock reading from the moment the diffusion process is seeded to the time when it reaches the node. Arrival times discard by definition many temporal heterogeneities of the empirical data and usually exhibit a well-defined distribution [PBC⁺11] that is robust with respect to changes in the starting time of the process and across temporal networks of human contact measured in different contexts.

To conclude, we have seen that the characterization of network dynamics is an important task in the perspective of event detection. Many studies have been done on the characterization of graph snapshots dynamics using metrics for static graphs. These notions are however limited, and new ones have been proposed to take time directly into account. The choice of time scale is an issue as it impacts measurements, but no consensus has been found on an optimal time scale and no study has been done on its impact on event detection. Adjacent time windows raise the additional issue of time position, e.g. windows starting at midnight would lead to different findings than those starting from mid-day. Finally, new time units have been proposed based on minimal changes in the network and on local time units, which provide ground for our work.

5.2 Generalization of *Outskewer*

In this Section we study the characterization of link streams in the perspective of generalizing *Outskewer* to event detection in network statistics. We have seen previously that time unit and time scale raise unresolved issues for network measurement using time windows, and the impact of these parameters remains unknown for event detection. We thus propose to study the impact of time unit and time scale in a time series of a network statistic computed over a sliding time window. Our findings will help calibrate network measurements, within which *Outskewer* may detect regular and abnormal behaviors.

5.2.1 Use of a Sliding Time Window

Three classical approaches exist for the study of a network property in a dynamic context. The first one consists in studying the growth of the graph over time, displayed in a cumulative way. For instance the cumulative number of nodes is shown in Figure 5.5, where the number of nodes is plotted as a

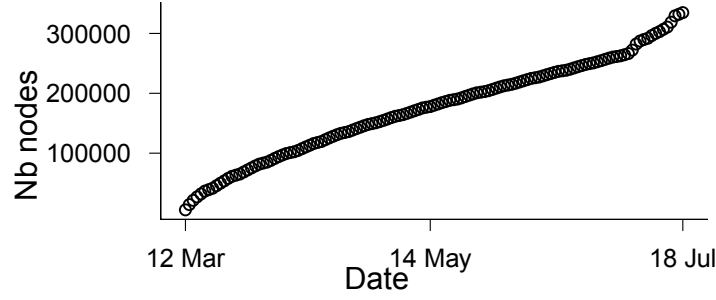


Figure 5.5: Number of distinct nodes as a function of total number of links.

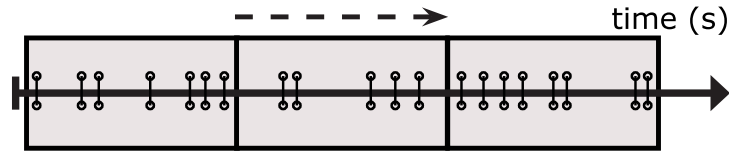


Figure 5.6: Stream of appearing links split into contiguous time windows.

function of the total number of links observed since the beginning of the capture. This plot displays a regular growth with a regime change at the end, but we obtain little information on the underlying dynamics. The second approach consists in splitting the stream using contiguous time windows to build a series of sub-graphs, as illustrated in Figure 5.6. We then compute the selected statistical property on each sub-graph. For instance, the number of nodes of each sub-graph captured over time is shown in Figure 5.7. This plot displays a regular trend and a few spikes, however we may miss more subtle events and the precise moment of their appearance. So we use a third approach, which is the generalized version of this approach. It consists in extracting consecutive sub-graphs from a sliding time window, as illustrated in Figure 5.8.

Our approach is as follows: given a link stream F (as defined in Section 2.1), we compute the time series S_w of the graph statistic $S_w^{i,2}$. If w is a function of time, e.g. a value in seconds, the sliding window is the multiset which contain all links observed during w seconds.

As far as we know, all studies on evolving networks which make use of a sliding window define its width in seconds. The apparent simplicity of this approach brings little attention because it is easy to set up and involves a common time unit. However it raises non-trivial questions (detailed later) which are not addressed in most studies.

The use of a sliding window for the analysis of graph dynamics raises several questions: which unit (traditional time-based or link-based) should we

²A relevant property for bipartite graphs is studied in Section 5.3

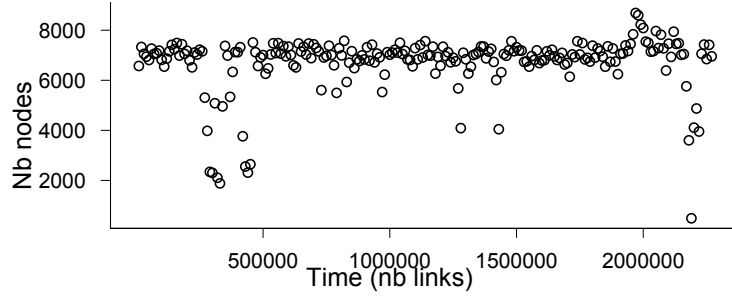


Figure 5.7: Number of distinct nodes in the union of 10,000 consecutive links, computed every 10,000 links, as a function of the number of links.

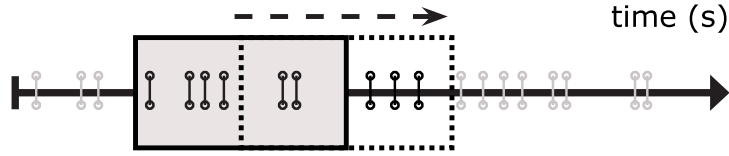


Figure 5.8: Sliding time window over a stream of appearing links.

use to detect events? Moreover, what is the impact of the window length on the evolution of the studied property? In the following sections we empirically study the impact of these different concepts of time as well as various time scales on a trivial statistical property: the number of nodes observed in the network over time. We aim at determining the consequences of such choices on our ability to characterize dynamics, and to detect statistically significant events. We have found that these parameters have an important impact on the observed results.

5.2.2 Choice of the Time Unit

5.2.2.1 Intrinsic vs Extrinsic Time (Time Unit)

Time is a controversial concept that one can see as a dimension in which changes occur in sequence. In this perspective, time is considered as absolute, i.e. changes happen independently from the flow of time [New87, Kan81]. But if we consider time as a relative concept, time then depends on space. This debate remains open, however in practice time is experienced as relative because we can only measure it through the relative movements of bodies (in space). Many techniques exist to measure it. The unit adopted by the International System of Units is the second, which is defined as the transition between two states of the caesium 133 atom [dlCdM06]. This unit is therefore related to movements measured in the physical space.

However information networks make the physical space transparent by con-

necting elements whatever their geographical distances. In graph theory, the distance between two nodes (also called *geodesic distance*) is indeed defined as the number of links in a shortest path connecting them. Under the hypothesis that network distances are independent from geographical distances, we consider the physical space as absolute from a network point of view. Conversely if we reject this hypothesis and correlate network distances to physical distances, the observation of such effects may hide the effects which are not related to the physical space. In the first case, measuring distances with physical units is not relevant. In the latter case, it brings little information on the network itself. This question is difficult because effects have been found even for social networks and the Web, which are designed to abolish the physical distances between people. For instance, there is a higher probability on Facebook to be friend with someone from the same country [UKBM11]. On Github, open source developers based in North America receive a disproportionate amount of attention [TH10]. These studies shed light on the way the geographical location of users influences the network, but they do not address the reciprocal question of how the network allows users to be connected to one another despite geographic boundaries. Hence existing works do not study the endogenous effects at stake in the network (i.e. which come from inside).

Notwithstanding the high potential impact of a time unit derived from the physical space, most studies use the absolute time in evolving networks: statistical properties are measured as a function of the second and its derivative units (e.g. days and years). As a consequence, they detect exogenous activities on these networks (i.e. which come from outside) [AG11, PBC⁺11, WD-dACG12b]. For instance, click-stream data of Web traffic naturally reveal a day-night pattern in the network because of usual human activity [MMF⁺08]. While this finding may be of interest, it provides more information on the users activity than on the network itself. Such trends may hide the patterns which are only related to the network, preventing us to characterize the endogenous dynamics of the network.

We thus introduce a concept of relative time in a network point of view, called *intrinsic time* of the network, as opposed to the *extrinsic time*, which is a concept of absolute time. Let the *extrinsic time* of the network be the time measured using the second. We call it *extrinsic* because its flow is independent from the changes that occur in the network. Let the *intrinsic time* of the network be the time measured by the transition between two states of the network. The unit is thus the (spatial) change of the network, i.e. the addition or removal of one node or one link. This unit is minimal because nothing can happen in the network between two consecutive changes. We call it *intrinsic* because time depends on the changes that occur in the network, and changes depend on such time to happen. The relation between time and space in networks is however out of scope of this thesis.

Whereas the *extrinsic time* is broadly used without notice, we find out in

the following section that using it has a high impact on the measurement of statistical properties of evolving networks. We will see that using the *intrinsic time* avoids biases and allows us to reveal network dynamics. In the following, the unit of *intrinsic time* is the appearance of a link, because we focus on link stream data in this thesis.

5.2.2.2 Empirical Impact

We have conducted our experiment on the GITHUB dataset described in Section 5.2.1 for various network metrics. We report the results related to the evolution of the number of nodes, which is representative of the impact of both time concepts. We indeed obtain similar results for the following metrics, which are classical properties of networks: the evolution of the number of distinct links³, the number of connected components⁴, the average degree⁵ and the maximum degree⁶.

The observed number of nodes in the GITHUB dataset reflects both the number of users and the number of software repositories. The temporal evolution of this statistical property when considering *extrinsic time* reveals a daily and weekly pattern. On the contrary, the overall number of nodes is very stable when *intrinsic time* is used, which confirms that different types of dynamics are observed according to the time unit. The events which correspond to spikes may clearly be extracted from the overall trend: this shows that the graph has normal dynamics in the statistical sense (i.e. the mean value is a relevant indicator for the description of the distribution of values) and that statistical anomalies (i.e. values which deviate significantly from the mean) may be identified. Although some events also seem to appear in the curve obtained with *extrinsic time*, their characterizations are in practice much more difficult⁷. Intrinsic time therefore seems to be more relevant to perform dynamic measures.

Figure 5.9 represents the evolution of the number of nodes over time, where the size of the sliding window is one hour and ten minutes. The plot displays a daily fluctuation of the number of nodes. We thus observe more nodes during the day than during the night. The plot also displays a weekly fluctuation. We thus observe a greater number of nodes during the week than during the weekend, revealing the dynamics of users activities on the network. Hence

³The number of links is the window width, thus it is constant.

⁴Let $\mathcal{C}(G)$ be a connected component of $G(V, E)$ (where V is the set of nodes and E is the set of links): it is a connected sub-graph of G , i.e. for each pair of nodes $(u, v) \in \mathcal{C}(G)$, a path exists between u and v . The number of connected components is therefore $|\{C \in \mathcal{C}(G)\}|$.

⁵Let $d(u)$ be the degree of the node u , i.e. its number of neighbors. The average degree of the graph $G(V, E)$ is $2 \times |E|/|V|$.

⁶Let $d(u)$ be the degree of the node u . The maximum degree of the graph $G(V, E)$ is the maximum number of neighbors of nodes in the graph, i.e. $\max(D)$, $D = \{d(u), \forall u \in V\}$.

⁷Chapter 6 is dedicated to the interpretation of detected events.

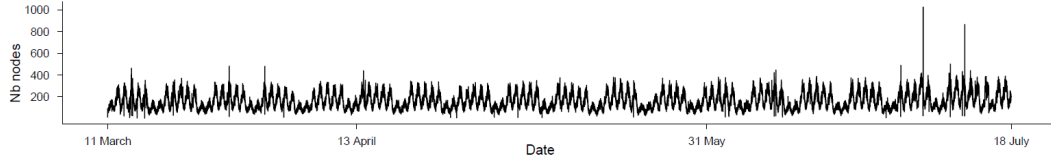


Figure 5.9: Number of nodes in a sliding window of 10 minutes, plotted at each minute.

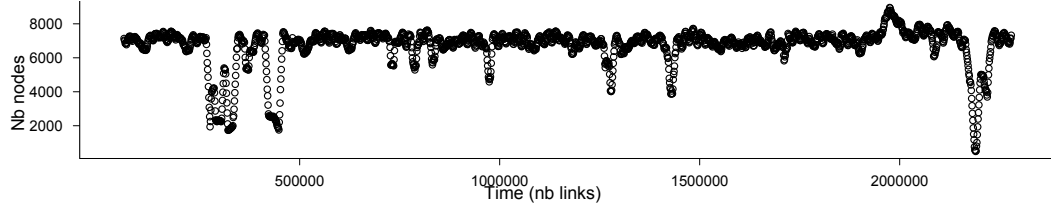


Figure 5.10: Number of nodes in a sliding window of 10,000 links, plotted for each set of 1000 links.

one can see the network as an artifact which is able to capture and reveal phenomena that happen outside of it.

Figure 5.10 represents the same property, but the size of the sliding window is 10,000 appearing links. This plot does not display such fluctuations. On the contrary, we observe that the number of nodes is globally stable with only a few variations and spikes. It is also the case for other properties that we have studied, like the number of distinct links, the number of connected components, the average degree and the maximum degree.

While we study the same property (number of nodes), the time unit has a high impact on the resulting curves. We thus show that **observed results are bound to an underlying concept of time**. Using the *intrinsic time* of the network instead of the traditional *extrinsic time*, we reveal totally different dynamics for the total number of nodes, which is a trivial property. We have also observed different dynamics for the other properties that we have studied further during this thesis. This study is hence of primary importance in metrology. Our results support the hypothesis that the intrinsic dynamics of the network is not captured by measures bound to an *extrinsic time* unit. An *extrinsic time* unit seems indeed more likely to capture the dynamics of exogenous activities on the network (i.e. which come from the outside), like the day-night and weekly patterns. Further studies with other datasets are however necessary to draw a firm conclusion.

5.2.3 Sliding Window Width (Time Scale)

In this Section we study the evolution of different metrics **at various time scales** for both *extrinsic* and *intrinsic* times, i.e. for different sizes w of the

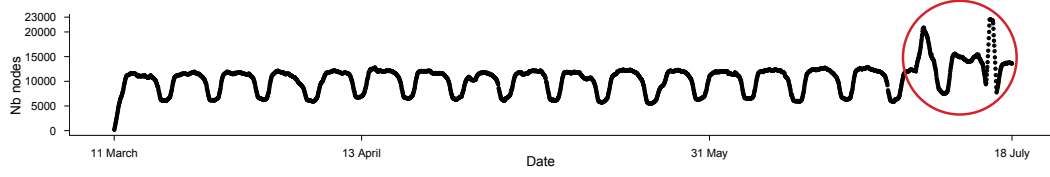


Figure 5.11: Number of nodes in a sliding window of 24 hours, plotted every hour. Significant spikes are circled.

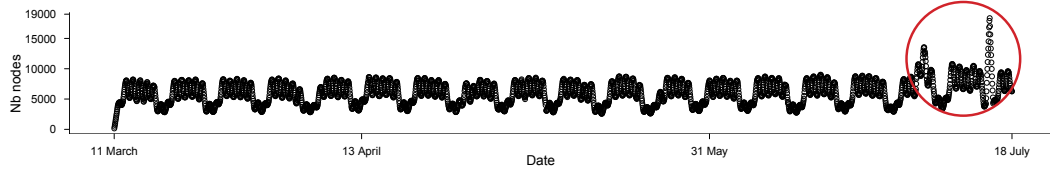


Figure 5.12: Number of nodes in a sliding window of 12 hours, plotted every hour. Significant spikes are circled.

sliding window. As in the previous subsection, we report the results for the evolution of the number of nodes, because this property is representative of the time scale's impact.

5.2.3.1 Extrinsic Time

We have computed the number of nodes as a function of time for a sliding window of size $w = 10$ minutes (Figure 5.9), 1 hour (Figure 5.13), 12 hours (Figure 5.12) and 24 hours (Figure 5.11) on the Github dataset. Each plot for w from 10 minutes to 12 hours clearly exhibits a daily trend. A weekly trend is also observable for all studied w . These patterns are exogenous activities as explained in Section 5.2.2. Spikes appear clearly for w equal to 10 minutes and 1 hour; they are less extreme for $w = 12$ hours, and most of them have disappeared for $w = 24$ hours. Surprisingly, the two remaining spikes for $w = 24$ hours are more extreme than for smaller w .

We expect that a large window width smoothes the resulting curve. The plot corresponding to a 24 hours window (Figure 5.11) confirms this intuition: a regular variation can be observed, which looks more disrupted with the 12 hours window (Figure 5.12) and 1 hour window (Figure 5.13). This "noise" is due to users daily activities on the system; 5 oscillations may be identified during the week and 2 oscillations during the week-end. The plot using 24 hours window therefore only reflects the Monday to Friday period, displaying lower user activity during the week-end. However we may observe the appearance of events or the growth of their amplitude (positive peaks) when the size of the window increases (circled spikes in these figures). The consequences of these results are the following:

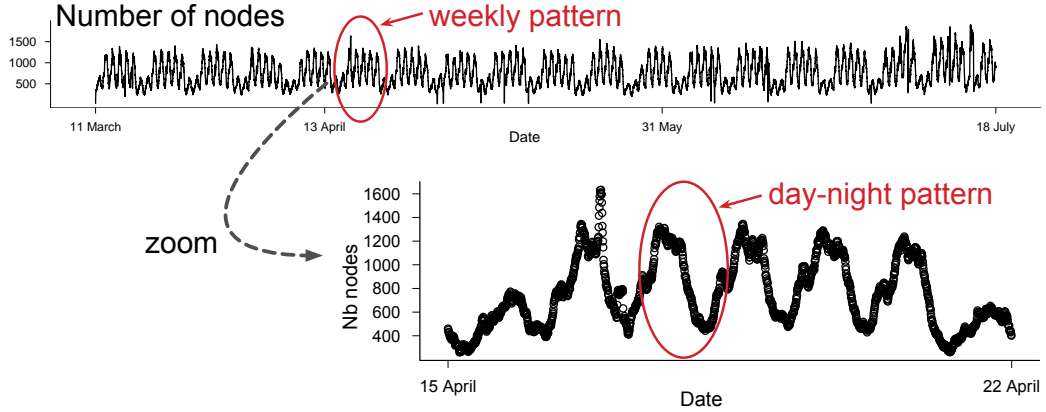


Figure 5.13: Number of nodes in a sliding window of 1 hour, plotted every 5 minutes. Spikes which are significant at larger scales are circled. We zoom on the dotted area.

- Smoothing the curve using a large window size is not a good option, as it modifies the shape of the plot and changes the perception of events.
- Trying to detect all events using a small window size is not a good option either, as events taking place at higher time scales are missed. This approach is also extremely costly in terms of computing time.

We now focus on the number of nodes using an *intrinsic time* unit to determine whether or not we observe the same effects.

5.2.3.2 Intrinsic Time

We have computed the number of nodes as a function of time for a sliding window of size $w = 50,000$ links (Figure 5.14) and 1000 links (Figure 5.15). We observe in these figures that the global trend and the regime changes (i.e. sudden changes of the mean of the time series) are similar for all studied w . Spikes observed on small w values disappear progressively when w increases.

Our intuition has led us to set a large w for removing non-significant events in order to smooth the global trend while keeping significant events. But we have discovered that this strategy alters the trend because some spikes that are not present for smaller w can appear. One should thus consider the duration of expected events to set the size of the sliding window accordingly, and consider the results to be valid for this specific time scale only. In our data, $w = 10,000$ is a good tradeoff to observe all events, see Figure 5.10.

In this Section we have discussed which concept of time is relevant to characterize the intrinsic dynamics of the system. We have also seen that **there is no optimal time scale for the characterization of events** for

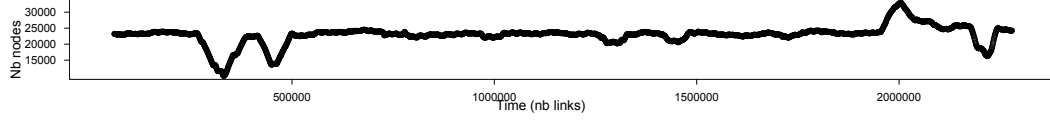


Figure 5.14: Number of nodes in a sliding window of $w = 50,000$ links, for each 1000 links.

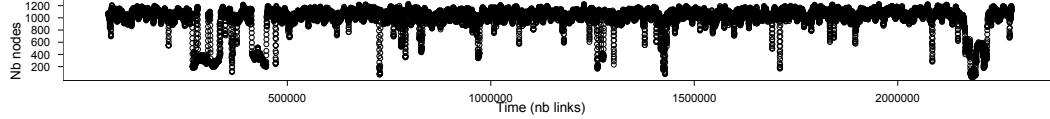


Figure 5.15: Number of nodes in a sliding window of $w = 1000$ links, for each 100 links.

both extrinsic and intrinsic time. In order to capture the intrinsic dynamics of user-system interactions, we propose in the following Section a specific property for studying the core interactions in the GITHUB dataset, modeled as a bipartite graph, using the concept of *intrinsic time*.

5.3 Application: Study of Github Internal Links

We propose to focus on the most stable interactions in the GITHUB system to capture its essential topology dynamics (and neglect marginal -noisy- variations). We recall that GITHUB is modeled as a bipartite graph of link stream (see definition in Section 2.1), where *top* nodes represent users and *bottom* nodes represent software repositories.

The property we suggest to consider for monitoring the dynamics of most stable user-system interactions is the number of *internal links*⁸. Whereas it has been shown that internal links are able to capture interesting statistical properties of bipartite networks, this notion has never been used for the study of evolving networks. We measure the number of T-internal links using a sliding window of 10,000 links⁹. We observe on Figure 5.16 that the number of T-internal links is globally stable around 2400 links, thus 24% of links inside the sliding windows are internal links. This result is interesting because it provides us a characterization of a **normal behavior of the system**. We also observe significant events with fast increases and decreases of the values which are statistical anomalies of the system's behavior. These events are however not new to us: we find also them in the evolution of the number of nodes, or in the evolution of the number of distinct links (which plot is not shown because similar to the number of nodes). So the number of internal

⁸As defined in Section 2.1

⁹This is the size selected in Section 5.2.3.

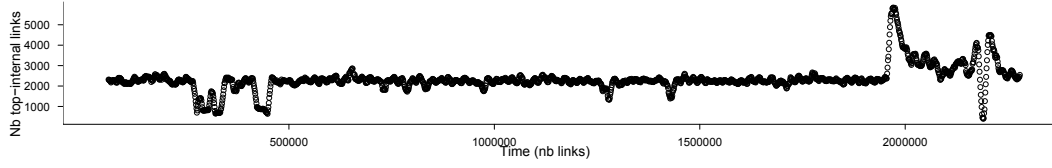


Figure 5.16: Number of T-internal links in a time window of width $w = 10,000$ links, for each 1000 links.

links seems strongly correlated to the number of distinct links.

This observation has led us to study the **normalized number of internal links**, which is the number of internal links divided by the number of distinct links in the sliding window. This property gives the ratio of internal links observed in the sliding window. This ratio is different from the ratio we can compute on the previous property because only distinct links are counted. It is relevant because the measure of internal links is independent from the fact that more than one link exist between two nodes.

We automatically detect statistically significant events using *Outskewer* on the evolution of the normalized number of internal links. We annotate the resulting plot on Figure 5.17 with a rectangle around each set of values which are abnormal in the plot, or which are abnormal in other plots, and we label the events with a capital letter. Points of the curve are colored as a function of their outlying class: red for *outliers*, orange for *potential outliers*, blue for *not outliers*, green for *unknown*. We observe that *Outskewer* is able to detect automatically all events we identified manually. They are either identified by a set of *outliers*, or by a set of *unknown* values. The latter case happens when there is no normal behavior in the related sliding windows, but we consider that something unexpected happens at this moment.

We identify one new event (compared to the number of internal links), the small event *A*, while events *E* and *G* have disappeared. The event *I* is interesting: it is the only one with an increasing spike on the plot of the number of internal links, and it is also revealed by the plot of the maximum degree (not shown here). Moreover, we discover that all decreasing spikes in the plot of the number of links are increasing instead in this plot. The ratio of internal links increases when the number of distinct links decreases. This effect is due to an intrinsic bias of this property, which counts as internal links the links attached to nodes of degree equal to 1 (a node of degree 1 is a node with 1 single neighbor in the graph). We may remove this bias by ignoring the nodes of degree 1. We thus obtain the proportion of internal links among the links that connect nodes which have at least two neighbors. We see in Figure 5.18 that this filter has removed most events, thus keeping the most significant connections.

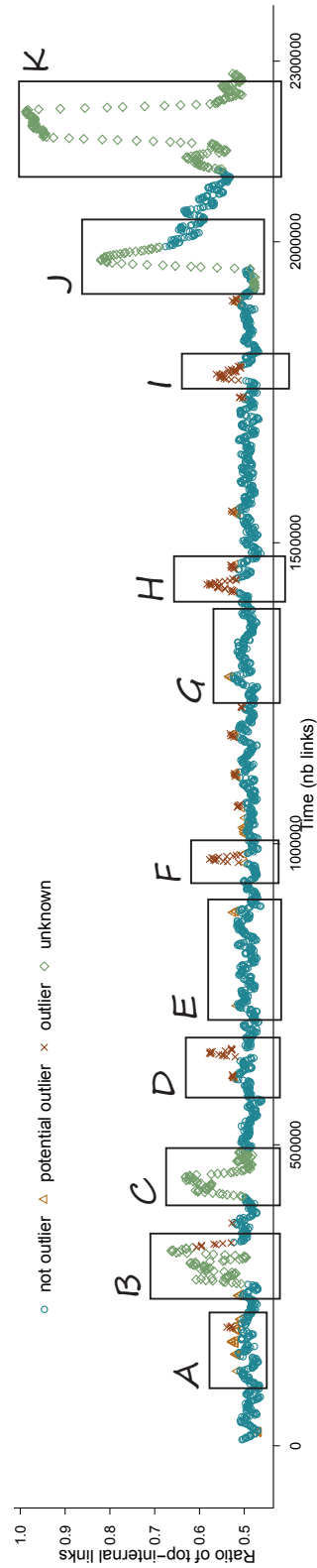


Figure 5.17: Number of \top -internal links divided by the total number of distinct links in the time window of width $w = 10,000$ links, for each 1000 links. Events are circled and labeled by a letter. Colors represent the outlying class of values: *outliers* in red, *potential outliers* in orange, *normal values* in blue, *unknown* values in green.

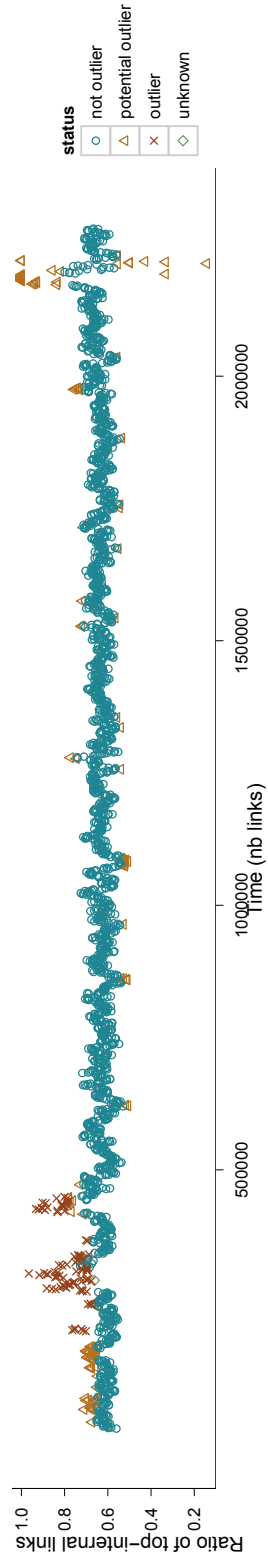


Figure 5.18: Ratio of T-internal links among the links connecting nodes of degree > 1 in a time window of width $w = 10,000$ links, for each 1000 links. Colors represent the outlying class of values: *outliers* in red, *potential outliers* in orange, *normal values* in blue, *unknown* values in green.

5.4 Conclusion

We have proposed in this Chapter a generalization of *Outskewer* for event detection in link streams. This approach is easy to set up but raises challenging questions: which time unit should be used, and at which time scale? We have studied the evolution of the GITHUB network to characterize normal behavior and to detect abnormal dynamics, through the measurement of specific statistical properties over a sliding window. We observe that “wall-clock” time units like the second and the hour may hide events because of the impact of cyclical human activity. We have thus introduced the notion of *intrinsic time* (as opposed to traditional *extrinsic time*), based on link appearances, to avoid this issue. Intrinsic time reveals totally different dynamics from those observed with extrinsic time: the time series of the network statistic exhibit a stable behavior with some peaks and regime changes, allowing us to detect clear events that we could not see otherwise.

We have also studied the impact of time scale on event detection with both intrinsic and extrinsic time. As new events appear at various time scales, we have seen that there was no optimal time scale (i.e. optimal width for the sliding window), which is counter-intuitive. The smallest resolution is thus not able to capture all events. Many existing studies on the impact of time scale in dynamic network measurements propose different methods to compute an optimal time window, however they never verify the impact on event detection.

We have finally applied our (generalized) method to the GITHUB dataset¹⁰ using an intrinsic time unit. We have successfully captured the normal behavior of this system using a property called the *internal links* which reveals the core interactions and significant events. The observation of internal links enables us to find novel events, which one could not see otherwise, while it confirms events detected using basic metrics. Our approach may be applied to any interaction system to model its behavior, define its regular evolution and detect potential anomalies (“events”). It may ultimately be used in a real-time fashion to raise alerts automatically when abnormal behaviors occur.

However an exploratory analysis is required before any automation. First, a multi-scale analysis of detected events would shed light on the nature and scale of the events. Second, an investigation of the network is needed to validate and eventually interpret the events. Visualization techniques may help in this task, and events detected by *Outskewer* can be used as entry points in the network to explore its state at the time of the event. The complete chain may form an exploratory framework for event detection in link streams. This is what we study in the following Chapter.

¹⁰See Section 2.2 for the description of the GITHUB dataset.

CHAPTER 6

Event Detection & Visual Investigation

Contents

6.1	Exploratory Method for Event Detection in Link Streams	122
6.2	Description of the Prototype	124
6.3	Application: Study of Github Events	125
6.3.1	Automatic Event Detection	125
6.3.2	Visual Event Validation	127
6.4	Conclusion	130

In this Chapter we propose our **hybrid exploratory method that combines automatic detection of statistically significant events (see Chapter 5) with visual graph mining techniques (see Chapter 3) to validate and interpret them if possible**. We first use *Outskewer* on a specific graph statistic. Afterwards, we pick up one of the detected events and perform a preliminary visualization of the corresponding sub-graph as a node-link diagram; we then identify an abnormal pattern, if any. A second visualization focused on the nodes of this pattern allows studying its temporal evolution and thus verifying whether this pattern is new (based on its existence or absence in the past). We consider that a new pattern (or a new combination of patterns) at the time of the event is correlated to it, and therefore may explain the event.

We apply our complete investigative method to the GITHUB on-line social network¹, which allows us to validate relevant events proposed by *Outskewer* and reject those for which no abnormal pattern is found. We thus show the complementary of the automatic method (based on a statistical analysis) and the manual method (based on interactive visualization) for event detection in a large link stream.

This Chapter is organized as follows. Section 6.1 describes the method. Section 6.2 describes a case study on the GITHUB dataset. We conclude in Section 6.4.

¹Presented in Section 2.2

6.1 Exploratory Method for Event Detection in Link Streams

Our method is divided into two steps. The first one consists of the automatic identification of statistically significant changes in the network topology over a sliding time window, based on a graph statistic as described in Chapter 4 and 5. These windows are classified as *outlier*, *potential outlier*, or *normal*. This method processes the whole graph quickly, so that the analyst can focus on abnormal windows. This second step aims at validating the relevance of the previously detected events. To do so, we propose to visually identify abnormal patterns in the sub-graph corresponding to the abnormal time window (as suggested in Chapter 3), then to check that these patterns are actually abnormal over time.

Step 1: an automatic event detection is performed on a time series S_w (of a graph statistic S_w^i) as defined in Section 2.1 using the *Outskewer* method.

Step 2: the visual analysis of events aims at validating the previously detected events i , and at interpreting them if possible by finding correlations with abnormal patterns in the corresponding sub-graphs G_w^i . For instance, the sudden and frequent recurrence of a link appearance contributes to dramatically reduce the number of unique nodes observed over a time window at this moment. This event may be automatically detected but may have multiple explanations as we will see in the following of the Chapter.

We propose an interactive visualization that allows analysts to investigate events detected by *Outskewer*. We assume that they know how to read and interpret node-link diagrams, which are the most common representation of networks in scientific visualization systems (e.g. SoNIA [MMBd05], Visone [BW04], ViENA [WZF11], Gephi [BHJ09], NodeXL [SSMF⁺09], TempoVis [ATMS⁺11], and GraphDiaries [BPF⁺12]). Our prototype has to respect the following constraints:

- Represent the network structure as a node-link diagram.
- Do not display the entire network, which may be made of millions of nodes and links, making the diagram unreadable².
- Consider that the events detected by *Outskewer* may be very sparse over time due to long periods of abnormal activity.

The event validation and interpretation steps rely on the visualization of each sub-graph G_w^i whose statistic S_w^i has a value classified as *abnormal* by *Outskewer* (i.e. the event). Although abnormal patterns may appear in G_w^i ,

²as explained in Section 3.1.5.4

6.1. Exploratory Method for Event Detection in Link Streams 123

this view is not enough to correlate a pattern and an event. We consider that the pattern should exceptionally appear at this time of F . The visual event investigation follows three steps:

1. Extract the sub-graph G_w^i .
2. Identify one or more suspicious patterns according to the analyst's criteria.
3. Disambiguate event interpretation: for each abnormal pattern, determine whether it occurs only during the event. If this is the case, we consider that the pattern is correlated to the event.

These three steps are detailed below:

1. Selection of events to be investigated: we recall that an event is an abnormal value of S_w^i or a set of consecutive abnormal values $\{S_w^i, S_w^{i+1}, \dots, S_w^j\}, i \leq j$. We study the sub-graph G_w^i that corresponds to the event, or to the first value observed at the beginning of the event. Other possibilities exist such as studying the sub-graph of highest anomaly score in $[i, j]$.

2. Identification of an abnormal pattern: the analyst selects an event i in the combo list of events. The sub-graph G_w^i is displayed; the analyst may reduce the time window size w' for display (thus reducing the sub-graph size) or keep the size w used during automatic detection. The analyst observes the node-link diagram to find suspicious patterns (regarding the graph statistic used by *Outskewer*), such as stars, connected components, or a weighted link. If a suspicious pattern is found, the analyst explores the graph evolution around the time of the event. If this pattern remains observable before or after this time, then it is obviously not correlated to the event. The analyst may look for another pattern, or consider that the event is invalid.

3. Event interpretation: once a suspicious pattern is found, the analyst investigates if it is unusual or repetitive. This step tests the correlation of the pattern and the event. As illustrated in the following Section, the analyst selects the nodes of the pattern and hides the others. He/she colors the interaction diagram and sees directly whether the links between the nodes of the pattern appear at other periods (in green). If this is not the case, then the pattern is unique and considered as correlated to the event. Otherwise the analyst moves the cursor over each period of appearance, to explore the shape of the topology among these nodes. The node position remains stable so that he/she can compare multiple versions of the pattern over time by exporting graph pictures (through the tool bar). If the pattern is redundant, then one cannot assert that it is correlated to the event, even if it may be abnormal in G_w^i . Conversely, if the pattern appears only at the time of the event, then one can assert that it is correlated to the event and interpret the event accordingly.

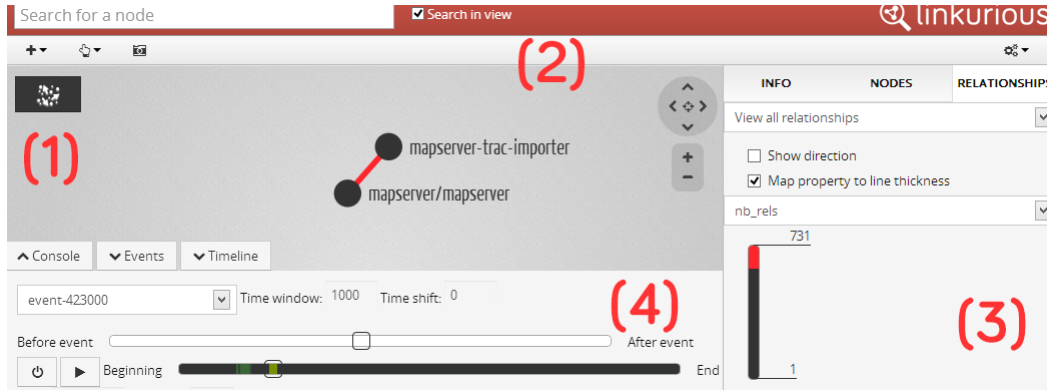


Figure 6.1: GUI of the prototype, derived from the Linkurious software.

6.2 Description of the Prototype

Our prototype extends the features of the commercial software Linkurious³ that we have co-developed. The user interface consists of 1) a graph visualization canvas represented as a node-link diagram, 2) a tool bar, 3) lateral information and statistical panels, to which we add on the bottom of the screen 4) an event selection panel and a temporal navigation panel. The prototype is developed in HTML5 and Javascript to run in a Web browser. It connects to a Neo4j graph database⁴. These elements of the interface are detailed below:

(1) *ForceAtlas 2* [JHVB11], a force-directed algorithm, places the nodes as a node-link diagram in a 2-d space. Links are weighted according to their number of appearances in the displayed graph. This way, the displayed graph is not a multi-graph (i.e. graph with multiple links between two nodes), which would have been difficult to represent and to read. Two visual variables⁵ associated to links may represent their weight: line thickness and color. Thickness is related proportionally to weight, and color is a linear gradient from grey (small value) to red (high value). Discs represent nodes and their surface is proportional to the node weighted degree. The user may also zoom and move the camera.

(2) The search bar helps find a node according to an attribute associated to it, such as the name of the entity represented by the node. The view is then centered on the node corresponding to the selected result. The tool bar allows to select, expand and hide nodes and links.

(3) The lateral panels display the attributes of the selected nodes and links. The user can color nodes according to an attribute, and modify line thickness and color. A diagram represents the distribution of link weights using colors, thus showing the rank of a link in the distribution.

³<http://linkurio.us>

⁴<http://www.neo4j.org>

⁵These variables reinforce their effects and speed up visual search of anomalous link.

(4) The bottom panels are the novelty we bring to Linkurious. They allow to select an event i in a combo list, to set the size of a time window w' (which may differ from the size w of the time window of *Outskewer*), and to display the corresponding sub-graph $G_{w'}^i$. A slider allows to shift the position of the time window slightly before or after the time of the event (up to 10,000 links upstream or downstream) to observe the evolution of the graph around the event. An *interaction diagram* represents the whole link stream F and a cursor on it allows to “teleport” the window anytime. Each percentage of the diagram width is colored according to the number of link appearances between the nodes displayed on the canvas at these periods: from grey (no appearance in the period) to light green (maximum number of appearances in the period). If one node is displayed, then its colors corresponds to the number of links connecting this node to other nodes. This diagram aims at showing the frequency of a link between nodes of a graph pattern over the whole graph evolution. This diagram is compact (10 pixels per 500 pixels in our implementation) but its accuracy is inversely proportional to the number of links m in F . For $m = 1000000$ and a width of 500 pixels, a width of 1 pixel encodes indeed 2000 links. Interactive features like a zoom could improve its accuracy.

We show in the following Section the result of a case study obtained with this prototype.

6.3 Application: Study of Github Events

6.3.1 Automatic Event Detection

We illustrate the interest of our approach on a basic statistic: the number of unique nodes $S_w^i = \text{card}(V^i)$. Following our study [HLG13b] of the impact of time window size on event detection (see Chapter 5) in this dataset, we compute S_w^i on F with $w = 10000$ links and we apply *Outskewer* on S with $w_o = 200$ links. These values are indeed a good tradeoff to detect events.

We observe on Figure 6.2 that the values are globally stable over time (in blue), but some decreasing peaks appear sometimes. The values of these peaks are classified as *outliers* (in red), and some values are *potential outliers* (in orange). We extract a list of events from the results⁶. Note that *Outskewer* may not be able to classify values during periods that fluctuate too much because no normal behavior can be found. In this case, values are classified as *unknown* (in green).

⁶see the definition of event in Section 2.1

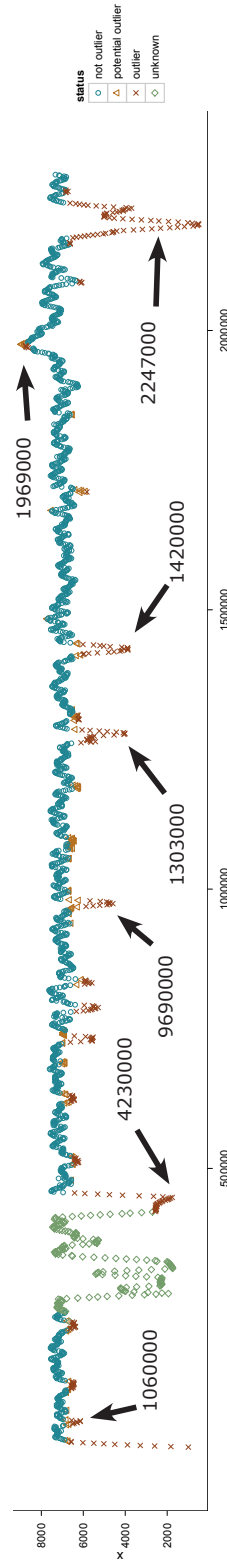


Figure 6.2: Time series of the number of unique nodes in the GITHUB link stream, with $w = 10000$ and $w_o = 200$ links. Abnormal values are in red, and the arrows point to the events analyzed in this Chapter.

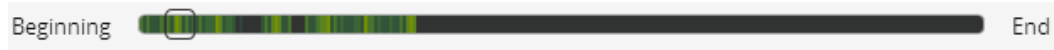


Figure 6.3: Diagram of interactions between the “goneri” user and the “fusinv/glpi” project. We clearly see that the user stops working on the project at a specific time. The event period is circled.



Figure 6.4: Diagram of interactions of the “mxcl/homebrew” project. This project receives contributions of users during the whole duration of the dataset. The event period is circled.

6.3.2 Visual Event Validation

We apply our visual method to check the relevance (i.e. validate) of most of these events and to interpret them, if possible.

Event 106000: $G_{w=10000}^{i=106000}$ has 6509 unique nodes, 4313 unique links, and 2233 connected components. The visualization brings out the relationship between the “goneri” user and the “fusinv/glpi” project, with 207 link appearances during this period, as well as the star relationships of the “mxcl/homebrew” project. These suspicious patterns are however not correlated to the event. We observe in the interaction diagrams of Figure 6.3 and 6.4 that these interactions are usual during the whole time of F for “mxcl/homebrew”, and happen until a specific time between “goneri” and “fusinv/glpi”. Thus we do not validate the event.

Event 423000: $G_{w=10000}^{i=423000}$ has 2531 unique nodes, 1525 unique links, and 1007 connected components. The visualization brings out the relationship between the “mapserver-trac-importer” user and the “mapserver/mapserver” project⁷, with 6993 link appearances during this period. We observe in Figure 6.1 that the interactions are very intense, with 731 link appearances in a window

⁷<https://github.com/mapserver/mapserver>

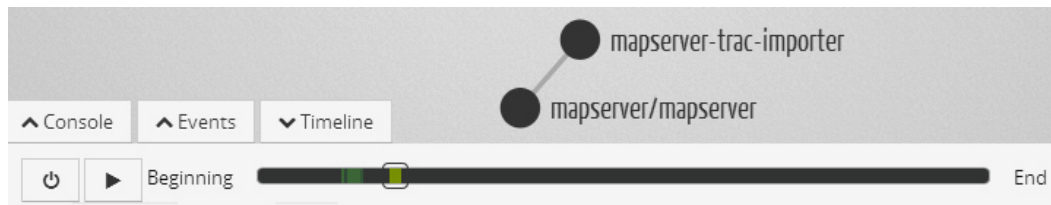


Figure 6.5: Interactions are usual between this pair of nodes during the event 423000, confirmed by the area in light green of the interaction diagram. We note another area in dark green earlier in the stream.

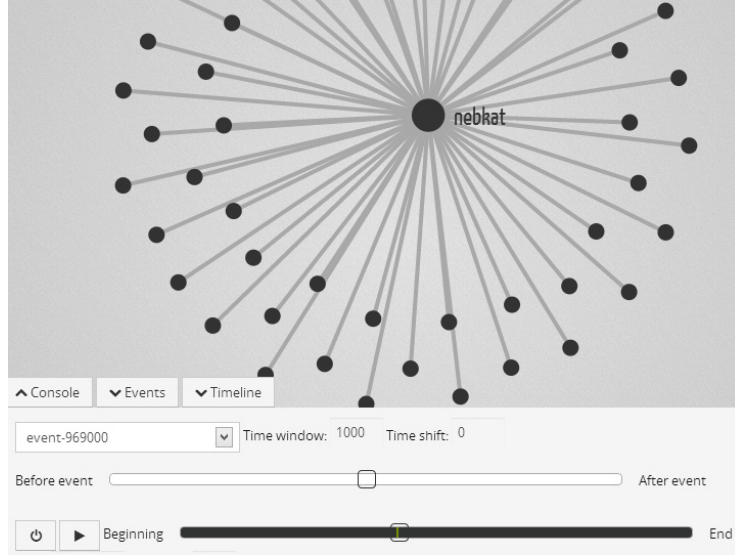


Figure 6.6: Interactions of the “nebkat” user with a large amount of projects during the event 969000. The interaction diagram is green only at the time of the event.

of size $w' = 1000$. We observe in the interaction diagram Figure 6.5 two short interaction periods, and the one of greatest intensity happens during the event. **We can therefore validate and interpret the event.** The node name “mapserver-trac-importer” indicates that the related user account is not human. We check this assumption on its activity page on Github.com⁸, and we note that the account has been renamed since then as “mapserver-bot”. This bot aimed at migrating the source code and bug list of the “mapserver” project from Trac to Github. We find traces of the discussion which started on March 19, 2012 inside the developer community on a public mailing-list⁹. The event hence corresponds to the time of the migration of the source code or the bug list: the bug #1¹⁰ dates indeed from April 2, 2012 on Github, which corresponds to the starting date of the event, on April 3, 2012 at 17:37:55 ($t_{i=413000}$).

Event 969000: $G_{w=10000}^{i=969000}$ has 5681 unique nodes, 3752 unique links, and 1958 connected components. The visualization brings out a star around the “nebkat” user, who has suddenly contributed to more than 100 projects owned by the “android-mirror” user, with 2138 link appearances. We also observe a connected component mostly made of projects coded in Rails and Javascript with more than 240 nodes, but which disappear at a smaller time scale (at $w' = 1000$). We decide to focus on the “nebkat” star. We observe on the

⁸<https://github.com/mapserver-bot?tab=activity>

⁹<https://lists.osgeo.org/pipermail/mapserver-dev/2012-March/012100.html>

¹⁰<https://github.com/mapserver/mapserver/issues/1>

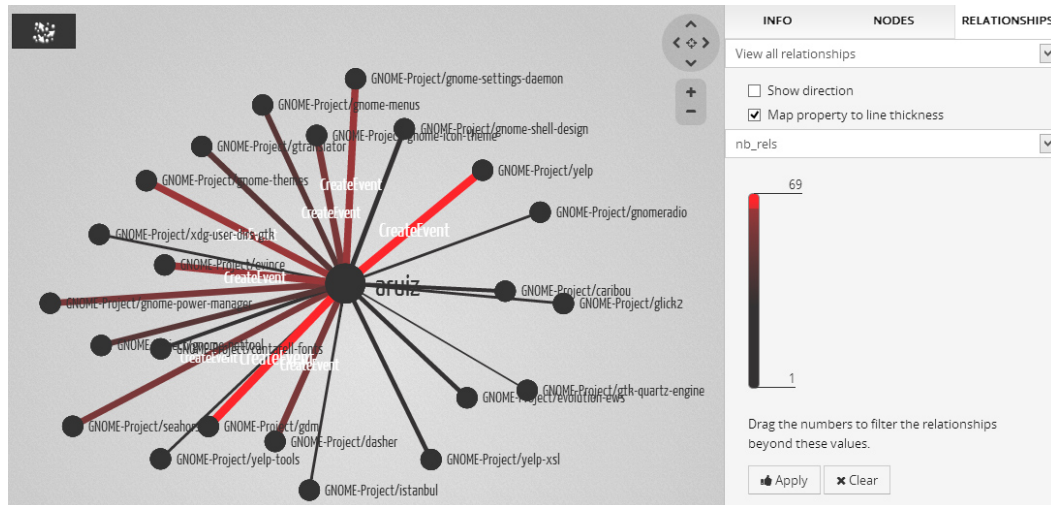


Figure 6.7: Visualization of interactions with the “aruiz” user during the event 1420000 with $w' = 1000$.



Figure 6.8: Interaction diagram of the “aruiz” user.

interaction diagram of Figure 6.6 that “nebkat” interacts only at the time of the event. The number of link appearances at this very moment encourages us to **validate the event** as correlated to this pattern. However the “android-mirror” account has been deleted from Github.com before our study, so we cannot interpret the event.

Event 1303000: $G_{w=10000}^{i=1303000}$ has 6642 unique nodes, 4395 unique links, and 2318 connected components. No pattern appears clearly. 17 nodes have between 50 and 110 interactions each, and a connected component has more than 380 nodes. We may reasonably make the assumption that the abnormal value of the statistic is due to a combination of factors. We thus do not validate the event.

Event 1420000: $G_{w=10000}^{i=1420000}$ has 4049 unique nodes, 2606 unique links, and 1458 connected components. The visualization shows a star around the “aruiz” user, who has contributed to group projects of “GNOME-Project”, with 5049 link appearances (see Figure 6.7). We also observe a star around the project “twitter/bootstrap”, with 208 link appearances. We decide to focus on the “aruiz” star because it has twenty times more link appearances. We observe in its interaction diagram on Figure 6.8 that it interacts at the time of the event only. We thus **validate the event**, however the available information on Github.com is not sufficient to interpret it.

Event 2247000: $G_{w=10000}^{i=2247000}$ has 7315 unique nodes, 4967 unique links, and 2451 connected components. The visualization displays a star around the

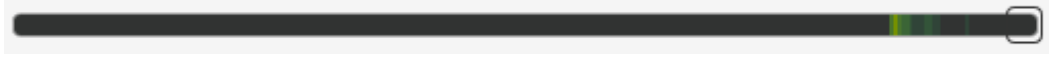


Figure 6.9: Interactions of “Try-Git” over time. We observe that the interactions with this node appear much before the event 2247000 and start suddenly at the event 1969000.

“Try-Git” user with 300 link appearances. We also observe two connected components of 270 and 300 nodes respectively. The event is ambiguous. We observe in the interaction diagram of “Try-Git” Figure 6.9 that most of its interactions happen during the previous event 1969000. We do not validate the event, and we analyze the event 1969000.

Event 1969000: $G_{w=10000}^{i=1969000}$ has 8651 unique nodes, 6817 unique links, and 1855 connected components. The visualization brings out a star around the “Try-Git” user with 3658 link appearances, which confirms the event suggested by its interactions diagram Figure 6.9. We find out on this project’s Web page that it is a tutorial for Git, one of Github’s underlying tools; the first action required from the user in this tutorial is to create a clone with a new project (by sending this user a *CreateEvent* message). The instant of the event corresponds to the moment when *Try-Git* was made public, on July 4th, 2012 at 5 pm (this information was confirmed by a post on the Github.com blog¹¹). We thus **validate the event**.

6.4 Conclusion

We have presented a new hybrid method of investigation in a large link stream to detect significant events in interactions between users of the online social network Github, by combining the *Outskewer* statistical method with a visualization system. *Outskewer* detects statistically significant events according to a graph statistic, and points to some moments of the evolution of the graph to be investigated. All these events are not necessarily relevant, and we need further validation, facilitated by the visualization. The latter shows a subgraph corresponding to the time of the events and facilitates longitudinal monitoring of abnormal patterns present in this sub-graph, which allows to precisely locate abnormal patterns at these times, and see if they appear to be related to the event.

We have illustrated our complete method on the GITHUB dataset with no a priori on the investigated events, and show through several examples that we are able to detect relevant events, and to reject events proposed by *Outskewer* but for which we do not find any anomaly in the network. We have thus shown the complementarity of statistics and visualization to detect events

¹¹<https://github.com/blog/1183-try-git-in-your-browser>

in a large link stream. Nonetheless, once a typology of events is identified for an investigation task, it is possible to fully automate the detection and validation of events by replacing the visualization by statistical analysis of patterns. Methods like Oddball [AMF10b] could detect abnormal patterns in the sub-graphs corresponding to abnormal windows.

We discuss our future work in the following Chapter.

Conclusion & Future Work

Contents

7.1	Conclusion	133
7.2	Future Work	135
7.3	Outro	136

7.1 Conclusion

In this thesis we have focused on the detection of events in link streams, which are ideal to model temporal interactions in many complex systems. Our main contribution is the creation of **a general methodology of exploratory analysis of link streams, in order to detect events in the evolution of the underlying network topology**. This exploratory framework combines automatic event detection with visual validation. The proposed approach is generic, and it may be adapted to specific use cases¹ by selecting an appropriate statistic for event detection and by specifying the valid kinds of events. It is ultimately possible to make the detection and validation of events entirely automatic once the events are characterized.

In spite of the diversity of complex systems which may be modeled as link streams (such as online social networks, telecommunication networks, search engine queries, and credit card payment systems) and their potential to describe the dynamics of large networks at a very fine granularity, link streams and their dynamics remain insufficiently explored. Our first contribution in Chapter 1 is thus a **rigorous definition** of a link stream; we have also defined the notions related to the study of their dynamics, in particular the concept of statistically significant event. The study of events is tightly related to the characterization of link streams: nodes and links usually appear and disappear over time in such streams, so a core research question is to determine to what extent such dynamics is regular, and in which cases irregularities (i.e. events) occur. Our methodology provides an experimental framework for the exploration of real-world data, which is an important step towards the proposal of theoretical frameworks.

¹We have worked on several datasets.

We have started with the **investigation of two visualization-based approaches** in Chapter 3. The first one based on a *timeline* helps users select a sub-graph corresponding to an event observed in the time series of the evolution of a graph statistic, but it remains limited by the graph size as we initially display the complete graph. The second technique is based on the incremental exploration of a graph, that allows experts to reveal points of interest in static graphs. This experiment has encouraged further studies of **local approaches to visual exploration**, which nonetheless have to be preceded by an automatic detection of events in the graph evolution.

In Chapter 4 we have proposed an **automatic approach to characterize time series**. Most current methods for event detection need a priori knowledge on the observed system, but we have chosen to design an exploratory approach with no prior information. We have thereby proposed a novel method, called *Outskewer*, which allows to detect statistically significant anomalies both in samples and time series, with no parameter but the time scale (i.e. the size of the sliding window on time series) for multi-scale analysis. We have performed experimental validation on synthetic data to control its accuracy and performance. We have successfully applied it to three various datasets to detect relevant events. *Outskewer* may help characterize link streams dynamics by distinguishing regular dynamics (i.e. when nodes and links appear and disappear regularly over time) from abnormal dynamics.

However it raises multiple questions on the way times series that show the evolution of a graph statistic are generated, and on their characterization in the perspective of event detection. We have indeed observed in GITHUB online social network that the time series of graph statistics presented day-night and weekly patterns due to user activity (Chapter 5). The observation of cyclical human activity, which appears when traditional time units like minute are used, prevents us to observe the network’s intrinsic dynamics and to detect related events. We have therefore introduced the **concept of *intrinsic time***, which provides new time units based on the appearance of links in the stream. This operation enables us to **generalize the use of *Outskewer*** to the precise observation of link stream dynamics. It especially allows detecting events unseen before, hidden by periodical patterns. However the choice of the method’s parameter (i.e. sliding time window) has non-trivial impacts on event detection: we have shown on the GITHUB dataset that there is no optimal time scale because events can appear at various scales, which is counter-intuitive. The smallest resolution is thus not able to capture all events. We have proposed a graph statistic based on the ***internal links of a bipartite graph***, which is able to capture the core interactions and reveal significant events after identifying a good size of the time window.

Finally, we have proposed in Chapter 6 an exploratory method to visually validate and interpret the statistically detected events. For each sub-graph corresponding to the identified events, we have looked for abnormal patterns in

the node-link diagram; if they only appear at this particular moment, then we can reasonably interpret the events as correlated to these patterns, and use the associated meta-data to suggest an explanation. **We have illustrated our complete methodology on the GITHUB dataset:** we have shown through multiple examples that it allows detecting relevant events and rejecting the others. This new methodology is perfectly suited for an exploratory approach, because no prior knowledge on data is required.

7.2 Future Work

Although we have used various datasets along this thesis to validate each component separately, the complete methodology has only been applied to the GITHUB dataset. In order to generalize our framework we will experiment it with other datasets and assess its effectiveness by an expert. These studies will contribute to a better understanding of the limits of our approach, and will provide ground for alternative solutions. Moreover, the statistics and events detected so far are very generic, and may be used for any link stream. Further studies may explore statistics and events that are more specific to one use case, e.g. in fraud detection where specific patterns are significant. These studies should therefore focus on integrating existing knowledge into the exploratory process. To conclude the general remarks, let us note that a study of the algorithmic complexity of our method is missing. We have been able to process a 2 million link stream (representing 4 months of Github’s life) in a minute with *Outskewer*, however a rigorous study is necessary to understand its limits and to optimize it. Other important factors are the graph statistics used to compute the time series analyzed by *Outskewer*, and the shape of the patterns correlated to the expected events.

Our methodology provides an exploratory framework and as such, it opens more specific research questions. First, we have defined the intrinsic time as based solely on link appearances. It would be interesting to generalize this concept to streams of appearances but also removals of nodes and links. Such streams represent series of graph changes and may encode “lifetimes” of nodes and links, like contact duration in face-to-face human interactions [BC13].

Second, the notion of *intrinsic time* may be applied to other contexts like dynamic community detection, time-based graph visualization, and diffusion. We have contributed to preliminary works [AGHLG13] showing that applying it to the study of diffusion processes reveals alternative phenomena from those observed using traditional time units.

Finally, little has been done on the local exploration of networks whereas it is an interesting approach to visualization, especially on very large networks. We have thus co-developed Linkurious², an enterprise-level software for the

²<http://linkurio.us/>

visualization of graph databases of dozens of millions of nodes and links. It combines a search engine and a local view of the search results (as nodes) with the ability to navigate from node to node by hiding and expanding connections. It is used for tasks that do not require an overview of the network, and is particularly appreciated in knowledge management projects.

7.3 Outro

We would like to close this manuscript on the position of this thesis in the field of informatics (i.e. automatic information processing) and outline industrial perspectives. What is the final product of our methodology? It is neither a metric, nor a visualization as these are both used as intermediate results. Instead, it is really the generation of statements about events, like “an event happened at a certain *<date>* involving a set of *<entities>*, interpreted as a *<case>*”. The goal of our research was not the improvement of a particular statistical technique nor of a visualization technique alone. We have therefore focused our work on the key problems for the generation of such statement. Statistics and visualizations are processes embodied in our approach to support ground truth for the final statements. The statements are self-contained information that may be shared directly. Statistics and visualizations may be archived as proofs once the analysis is over, as they carry valid references from the object of study to the final statements. We have proposed an information processing framework which may be fully automated. Our thesis therefore contributes to informatics.

Our framework is able to compute statements regarding network events through a “collaboration” between original data, a computer, and a human analyst. The role of the human is to prepare data, to scrutinize the unexpected, and to question data and machine’s results. He/she may no longer be useful once events are formally characterized and a corresponding validation algorithm is implemented. But he is always required to improve the methodology and to adapt it to new kinds of events, like a reflexive consciousness of this information generator. One may smile at the reading of such oneiric description, but this type of collaborations do already exist with automated content technologies: computers write stories based on data. For instance, automated football news are generated based on event data produced during games. The computer creates the story, describes the events, and the (human) journalist is left to write a deeper analysis. See the companies Automated Insight³ and Narrative Science⁴ to discover other applications in financial information, real estate and marketing. May we envision such future for our methodology?

³<http://automatedinsights.com>

⁴<http://narrativescience.com>

Bibliography

- [AAB⁺12] David Auber, Daniel Archambault, Romain Bourqui, Antoine Lambert, Morgan Mathiaut, Patrick Mary, Maylis Delest, Jonathan Dubois, Guy Mélançon, et al. The tulip 3 framework: A scalable software library for information visualization applications based on relational data. 2012. (Cited on page 40.)
- [AB02] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002. (Cited on page 3.)
- [Ada06] Eytan Adar. Guess: a language and interface for graph exploration. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 791–800. ACM, 2006. (Cited on pages 40 and 42.)
- [ADF⁺06] Adel Ahmed, Tim Dwyer, Michael Forster, Xiaoyan Fu, Joshua Ho, Seok-Hee Hong, Dirk Koschützki, Colin Murray, Nikola S Nikolov, Ronnie Taib, et al. Geomi: Geometry for maximum insight. In *Graph Drawing*, pages 468–479. Springer, 2006. (Cited on page 58.)
- [AF07] Fabrizio Angiulli and Fabio Fasseti. Detecting distance-based outliers in streams of data. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 811–820. ACM, 2007. (Cited on pages 17, 74 and 75.)
- [AF10] Leman Akoglu and Christos Faloutsos. Event detection in time series of mobile communication graphs. In *Army Science Conference*, 2010. (Cited on pages 77, 79 and 80.)
- [AG10] Thomas Aynaud and Jean-Loup Guillaume. Static community detection algorithms for evolving networks. In *WiOpt’10: Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pages 508–514, Avignon, France, 2010. (Cited on pages 73 and 81.)
- [AG11] Thomas Aynaud and Jean-Loup Guillaume. Multi-step community detection and hierarchical time segmentation in evolving networks. 2011. (Cited on pages 73, 81 and 111.)

- [Agg13] Charu C Aggarwal. Outlier detection in graphs and networks. In *Outlier Analysis*, pages 343–371. Springer, 2013. (Cited on pages 72 and 73.)
- [AGHLG13] Alice Albano, Jean-Loup Guillaume, Sébastien Heymann, and Bénédicte Le Grand. A matter of time-intrinsic or extrinsic-for diffusion in evolving complex networks. In *Advances in Social Network Analysis and Mining, ASONAM 2013*, 2013. (Cited on page 135.)
- [AL97] Atocha Aliseda-Llera. *Seeking Explanations: Abduction in Logic, Philosophy of Science and Artificial Intelligence*. PhD thesis, Stanford University, 1997. (Cited on page 10.)
- [ALGL11] Alice Albano, Bénédicte Le Grand, and Matthieu Latapy. Détection visuelle d'événements dans des grands réseaux d'interaction dynamiques. application à l'internet. In *Atelier Visualisation et Extraction de Connaissances de la conférence EGC 2011*, 2011. (Cited on pages 59 and 60.)
- [AMF10a] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD10)*, 2010. (Cited on pages 54 and 103.)
- [AMF10b] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Advances in Knowledge Discovery and Data Mining*, pages 410–421. Springer, 2010. (Cited on pages 72 and 131.)
- [AP12] Daniel Archambault and Helen C Purchase. The mental map and memorability in dynamic graphs. In *Pacific Visualization Symposium (PacificVis), 2012 IEEE*, pages 89–96. IEEE, 2012. (Cited on page 59.)
- [AP13] Daniel Archambault and Helen C Purchase. Mental map preservation helps user orientation in dynamic graphs. In *Graph Drawing*, pages 475–486. Springer, 2013. (Cited on page 60.)
- [APP11a] Daniel Archambault, Helen Purchase, and Bruno Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 17(4):539–552, 2011. (Cited on page 57.)

- [APP11b] Daniel Archambault, Helen Purchase, and Bruno Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 17(4):539–552, 2011. (Cited on pages 59 and 64.)
- [AS94] Christopher Ahlberg and Ben Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 313–317. ACM, 1994. (Cited on page 13.)
- [ATML12] Oussama Allali, Lionel Tabourier, Clémence Magnien, and Matthieu Latapy. Internal links and pairs as a new tool for the analysis of bipartite complex networks. *Social Network Analysis and Mining*, 2012. (Cited on page 20.)
- [ATMS⁺11] Jae-wook Ahn, Meirav Taieb-Maimon, Awalin Sopan, Catherine Plaisant, and Ben Shneiderman. Temporal visualization of social network dynamics: prototypes for nation of neighbors. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 309–316. Springer, 2011. (Cited on pages 61 and 122.)
- [Aub04] David Auber. Tulip - a huge graph visualization framework. *Graph Drawing Software*, pages 105–126, 2004. (Cited on page 42.)
- [AZY11] Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. Outlier detection in graph streams. In *Proc. IEEE 27th International Conference on Data Engineering (ICDE)*. IEEE Computer Society, 2011. (Cited on pages 82 and 103.)
- [Bar05] Albert-Laszlo Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, 2005. (Cited on page 82.)
- [BB67] Jacques Bertin and Marc Barbut. *Sémiologie graphique: les diagrammes, les réseaux, les cartes*. Mouton Paris, 1967. (Cited on pages 29 and 33.)
- [BBB⁺02] Michael Baur, Marc Benkert, Ulrik Brandes, Sabine Cornelsen, Marco Gaertler, Boris Köpf, Jürgen Lerner, and Dorothea Wagner. Visone software for visual social network analysis. In *Graph Drawing*, pages 463–464. Springer, 2002. (Cited on pages 57 and 58.)

- [BC13] Alain Barrat and Ciro Cattuto. Temporal networks of face-to-face human interactions. In *Temporal Networks*, pages 191–216. Springer, 2013. (Cited on pages 6, 103 and 135.)
- [BCC⁺13] Alain Barrat, C Cattuto, V Colizza, F Gesualdo, L Isella, E Pandolfi, J-F Pinton, L Ravà, C Rizzo, M Romano, et al. Empirical temporal networks of face-to-face human interactions. *The European Physical Journal Special Topics*, 222(6):1295–1309, 2013. (Cited on page 4.)
- [BD11] Benjamin Blonder and Anna Dornhaus. Time-ordered networks reveal limitations to information flow in ant colonies. *PloS one*, 6(5):e20298, 2011. (Cited on page 48.)
- [BDB06] Anastasia Bezerianos, Pierre Dragicevic, and Ravin Balakrishnan. Mnemonic rendering: an image-based approach for exposing hidden changes in dynamic displays. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 159–168. ACM, 2006. (Cited on page 59.)
- [BdM06] Skye Bender-deMoll and Daniel A McFarland. The art and science of dynamic network visualization. *Journal of Social Structure*, 7(2):1–38, 2006. (Cited on page 57.)
- [BDW05] Michael Burch, Stephan Diehl, and Peter Weißgerber. Visual data mining in software archives. In *Proceedings of the 2005 ACM symposium on Software visualization*, pages 37–46. ACM, 2005. (Cited on page 52.)
- [Ber83] Jacques Bertin. *Semiology of graphics: diagrams, networks, maps*. 1983. (Cited on page 44.)
- [Bg05] Irad Ben-gal. Outlier detection. In O Maimon and L Rockach, editors, *The Data Mining and Knowledge Discovery Handbook: A Complete Guide for Researchers and Practitioners*. Kluwer Academic Publishers, 2005. (Cited on pages 7 and 87.)
- [BGLL08] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008. (Cited on pages 13, 44 and 72.)

- [BHJ09] Mathieu Bastian, Sébastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. In *Proc. AAAI International Conference on Weblogs and Social Media (ICWSM'09)*, 2009. (Cited on pages 42, 64 and 122.)
- [BHKL06] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54. ACM, 2006. (Cited on page 4.)
- [BIM12] Ulrik Brandes, Natalie Indlekofer, and Martin Mader. Visualization methods for longitudinal social networks and stochastic actor-oriented modeling. *Social Networks*, 34(3):291–308, 2012. (Cited on pages 47 and 60.)
- [BJ76] George Box and Gwilym M. Jenkins. *Time series analysis: Forecasting and control*. Holden-Day, 1976. (Cited on page 73.)
- [BK85] S. M. Bendre and B. K. Kale. Masking effect on tests for outliers in exponential models. *Journal of the American Statistical Association*, 80(392), 1985. (Cited on page 94.)
- [BKB05] Kevin W Boyack, Richard Klavans, and Katy Börner. Mapping the backbone of science. *Scientometrics*, 64(3):351–374, 2005. (Cited on pages 37 and 44.)
- [BKNS00] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000. (Cited on page 75.)
- [BL94] Vic Barnett and Toby Lewis. *Outliers in Statistical Data*. John Wiley & Sons Ltd., third edition, 1994. (Cited on pages 18 and 73.)
- [BM98] Vladimir Batagelj and Andrej Mrvar. Pajek-program for large network analysis. *Connections*, 21(2):47–57, 1998. (Cited on pages 28 and 41.)
- [BM10] Lamia Benamara and Clémence Magnien. Estimating properties in dynamic systems: The case of churn in p2p networks. In *IEEE Conference on Computer Communications Workshops, INFOCOM Wksp*, 2010. (Cited on page 106.)

- [BMK96] Jim Blythe, Cathleen McGrath, and David Krackhardt. The effect of graph layout on inference from social network data. In *Graph Drawing*, pages 40–51. Springer, 1996. (Cited on page 60.)
- [BPF⁺12] Benjamin Bach, Emmanuel Pietriga, Jean-Daniel Fekete, et al. Temporal navigation in dynamic networks. In *IEEE Vis Week*, 2012. (Cited on pages 57, 58 and 122.)
- [Bra01] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001. (Cited on page 44.)
- [Bra08] Ulrik Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, 2008. (Cited on page 13.)
- [BW04] Ulrik Brandes and Dorothea Wagner. Analysis and visualization of social networks. In *Graph drawing software*, pages 321–340. Springer, 2004. (Cited on page 122.)
- [BWD⁺12] Benjamin Blonder, Tina W Wey, Anna Dornhaus, Richard James, and Andrew Sih. Temporal dynamics and network analysis. *Methods in Ecology and Evolution*, 3(6):958–972, 2012. (Cited on page 5.)
- [BWS06] Tanya Y Berger-Wolf and Jared Saia. A framework for analysis of dynamic social networks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 523–528. ACM, 2006. (Cited on page 103.)
- [C⁺03] Hans Chalupsky et al. Unsupervised link discovery in multi-relational data via rarity analysis. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 171–178. IEEE, 2003. (Cited on page 71.)
- [Car88] E Carlstein. Non-parametric change point estimation. *Annals of Statistics*, pages 188–197, 1988. (Cited on page 74.)
- [CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection : A survey. *ACM Computing Surveys (CSUR)*, 41(3), July 2009. (Cited on page 7.)
- [CBWG11] Rajmonda Sulo Caceres, Tanya Berger-Wolf, and Robert Grossman. Temporal scale of processes in dynamic networks. In *Data Mining Workshops (ICDMW), 2011 IEEE*

- 11th International Conference on*, pages 925–932. IEEE, 2011. (Cited on page 106.)
- [CC05] Brock Craft and Paul Cairns. Beyond guidelines: what can we learn from the visual information seeking mantra? In *Information Visualisation, 2005. Proceedings. Ninth International Conference on*, pages 110–118. IEEE, 2005. (Cited on page 41.)
- [CCD⁺08] Kathleen M Carley, Dave Columbus, Matt DeReno, Jeff Reminga, and Il-Chul Moon. Ora user’s guide 2008. Technical report, DTIC Document, 2008. (Cited on page 57.)
- [CDBF10] Fanny Chevalier, Pierre Dragicevic, Anastasia Bezerianos, and Jean-Daniel Fekete. Using text animated transitions to support navigation in document histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 683–692. ACM, 2010. (Cited on page 58.)
- [CE07] Aaron Clauset and Nathan Eagle. Persistence and periodicity in a dynamic proximity network. In *Proceedings of DIMACS Workshop on Computational Methods for Dynamic Interaction Networks*, 2007. (Cited on pages 6, 62, 63, 103, 104 and 106.)
- [CFL09] Claudio Castellano, Santo Fortunato, and Vittorio Loreto. Statistical physics of social dynamics. *Reviews of modern physics*, 81(2):591, 2009. (Cited on page 103.)
- [CFQS11] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. In *Ad-hoc, Mobile, and Wireless Networks*, pages 346–359. Springer, 2011. (Cited on page 103.)
- [Cha04] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In *Knowledge Discovery in Databases: PKDD 2004*, pages 112–124. Springer, 2004. (Cited on page 71.)
- [Cha12] Duen H Chau. *Data Mining Meets HCI: Making Sense of Large Graphs*. PhD thesis, 2012. (Cited on page 54.)
- [CHN08] Ling Chen, Yiqun Hu, and Wolfgang Nejdl. Deck: Detecting events from web click-through data. In *Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*, pages 123–132. IEEE, 2008. (Cited on page 83.)

- [CHS12] Zhengzhang Chen, William Hendrix, and Nagiza F Samatova. Community-based anomaly detection in evolutionary networks. *Journal of Intelligent Information Systems*, 39(1):59–85, 2012. (Cited on page 81.)
- [CM84] William S Cleveland and Robert McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984. (Cited on pages 29 and 44.)
- [CMS99] Stuart K Card, Jock D Mackinlay, and Ben Schneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999. (Cited on page 28.)
- [CNM04] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004. (Cited on page 72.)
- [CPN⁺13] Alessio Cardillo, Giovanni Petri, Vincenzo Nicosia, Roberta Sinatra, Jesús Gómez-Gardeñes, and Vito Latora. Evolutionary dynamics of time-resolved social interactions. *NetSci*, 2013. (Cited on pages 6 and 105.)
- [CVdBB⁺10] Ciro Cattuto, Wouter Van den Broeck, Alain Barrat, Vittoria Colizza, Jean-François Pinton, and Alessandro Vespignani. Dynamics of person-to-person interactions from distributed rfid sensor networks. *PloS one*, 5(7):e11596, 2010. (Cited on page 106.)
- [DC05] Jana Diesner and Kathleen M Carley. Exploration of communication networks from the enron email corpus. In *SIAM International Conference on Data Mining: Workshop on Link Analysis, Counterterrorism and Security, Newport Beach, CA*. Citeseer, 2005. (Cited on page 5.)
- [DHLZ13] Peter Diggle, Patrick Heagerty, Kung-Yee Liang, and Scott Zeger. *Analysis of longitudinal data*. Number 25. Oxford University Press, 2013. (Cited on page 10.)
- [Dim12] Dana Diminescu. Digital methods for the exploration, analysis and mapping of e-diasporas. *Social Science Information*, 51(4):451–458, 2012. (Cited on page 55.)
- [dlCdM06] Organisation Intergouvernementale de la Convention du Mètre. The international system of units (SI). Technical

- Report 8, Bureau International des Poids et Mesures, 2006. (Cited on page 110.)
- [dNMB05] Wouter de Nooy, Andrej Mrvar, and Vladimir Batagelj. *Exploratory social network analysis with Pajek*, volume 27. Cambridge University Press, 2005. (Cited on page 57.)
- [Dwy05] Tim Dwyer. *Two-and-a-half-dimensional Visualisation of Relational Networks*. PhD thesis, School of Information Technologies, Faculty of Science, University of Sydney, 2005. (Cited on page 58.)
- [Ead84] Peter A. Eades. A heuristic for graph drawing. In *Congressus Numerantium*, volume 42, pages 149–160, 1984. (Cited on page 35.)
- [ECH97] Peter Eades, Robert F Cohen, and Mao Lin Huang. Online animated graph drawing for web navigation. In *Graph Drawing*, pages 330–335. Springer, 1997. (Cited on page 49.)
- [EGH10] William Eberle, Jeffrey Graves, and Lawrence Holder. Insider threat detection using a graph-based approach. *Journal of Applied Security Research*, 6(1):32–81, 2010. (Cited on page 71.)
- [EH07] William Eberle and Lawrence B Holder. Mining for structural anomalies in graph-based data. In *DMIN*, pages 376–389, 2007. (Cited on page 71.)
- [EHSF12] Benjamin Edwards, Steven Hofmeyr, George Stelle, and Stephanie Forrest. Internet topology over time. *arXiv preprint arXiv:1202.3993*, 2012. (Cited on page 4.)
- [EMS04] Jean-Pierre Eckmann, Elisha Moses, and Danilo Sergi. Entropy of dialogues creates coherent structures in e-mail traffic. *Proceedings of the National Academy of Sciences of the United States of America*, 101(40):14333–14337, 2004. (Cited on pages 103 and 107.)
- [Eng01] Douglas C Engelbart. Augmenting human intellect: a conceptual framework (1962). *PACKER, Randall and JORDAN, Ken. Multimedia. From Wagner to Virtual Reality*. New York: WW Norton & Company, pages 64–90, 2001. (Cited on page 11.)

- [EP06] Nathan Eagle and Alex Pentland. Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268, 2006. (Cited on pages 4 and 104.)
- [FAM⁺11] Paolo Federico, Wolfgang Aigner, Silvia Miksch, Florian Windhager, and Lukas Zenk. A visual analytics approach to dynamic social networks. In *I-KNOW*, page 47, 2011. (Cited on page 60.)
- [FBAF10] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of the 6th International Conference*, page 8. ACM, 2010. (Cited on page 83.)
- [FBS06] Tanja Falkowski, Jorg Bartelheimer, and Myra Spiliopoulou. Mining and visualizing the evolution of subgroups in social networks. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 52–58. IEEE Computer Society, 2006. (Cited on page 60.)
- [Few06] Stephen Few. The surest path to visual discovery. *Perceptual Edge*, 2006. (Cited on page 28.)
- [Fox72] A.J. Fox. Outliers in time series. 34(3):350–363, 1972. (Cited on page 73.)
- [FQ11] Michael Farrugia and Aaron Quigley. Effective temporal graph layout: A comparative study of animation versus static display methods. *Information Visualization*, 10(1):47–64, 2011. (Cited on pages 45, 59 and 60.)
- [FR91] Thomas MJ Fruchterman and Edward M Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991. (Cited on page 35.)
- [Fry04] Benjamin Jotham Fry. *Computational information design*. PhD thesis, Massachusetts Institute of Technology, 2004. (Cited on pages 10 and 11.)
- [FT08] Yaniv Frishman and Ayellet Tal. Online dynamic graph drawing. *Visualization and Computer Graphics, IEEE Transactions on*, 14(4):727–740, 2008. (Cited on page 60.)
- [Fur86] George W Furnas. *Generalized fisheye views*, volume 17. ACM, 1986. (Cited on page 13.)

- [FVWSN08] Jean-Daniel Fekete, Jarke J Van Wijk, John T Stasko, and Chris North. The value of information visualization. In *Information Visualization*, pages 1–18. Springer, 2008. (Cited on page 28.)
- [GBBK11] Prasanta Gogoi, D K Bhattacharyya, B Borah, and Jugal K Kalita. A survey of outlier detection methods in network anomaly identification. *The Computer Journal*, 54(4), 2011. (Cited on page 7.)
- [GBT⁺09] Joel Glanfield, Stephen Brooks, Teryl Taylor, Diana Pater-son, Christopher Smith, Carrie Gates, and John McHugh. Over flow: An overview visualization for network analysis. In *Visualization for Cyber Security, 2009. VizSec 2009. 6th International Workshop on*, pages 11–19. IEEE, 2009. (Cited on page 63.)
- [GEY12] Sohaib Ghani, Niklas Elmqvist, and Ji Soo Yi. Perception of animated node-link diagrams for dynamic graphs. In *Computer Graphics Forum*, volume 31, pages 1205–1214. Wiley Online Library, 2012. (Cited on page 60.)
- [GFC04] Mohammad Ghoniem, J-D Fekete, and Philippe Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 17–24. IEEE, 2004. (Cited on page 34.)
- [GGAH13] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):1, 2013. (Cited on pages 9 and 74.)
- [GL04] Jean-Loup Guillaume and Matthieu Latapy. Bipartite struc-ture of all complex networks. *Information Processing Letters (IPL)*, 90(5), 2004. (Cited on page 19.)
- [GLF⁺10] Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, and Jiawei Han. On community outliers and their efficient de-tection in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge dis-covery and data mining*, pages 813–822. ACM, 2010. (Cited on page 72.)
- [Glo07] Peter Andreas Gloor. *Coolhunting: chasing down the next big thing*. Amacom, 2007. (Cited on page 58.)

- [GMH⁺06] Amy L Griffin, Alan M MacEachren, Frank Hardisty, Erik Steiner, and Bonan Li. A comparison of animated maps with static small-multiple maps for visually identifying space-time clusters. *Annals of the Association of American Geographers*, 96(4):740–753, 2006. (Cited on pages 58 and 59.)
- [GN02] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002. (Cited on page 72.)
- [GPCB13] Laetitia Gauvin, André Panisson, Ciro Cattuto, and Alain Barrat. Activity clocks: spreading dynamics on temporal networks of human contact. *arXiv preprint arXiv:1306.4626*, 2013. (Cited on page 107.)
- [Gru69] Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1), 1969. (Cited on page 18.)
- [GW12] Nils Gehlenborg and Bang Wong. Points of view: Networks. *Nature methods*, 9(2):115–115, 2012. (Cited on pages 33 and 34.)
- [HA04] Victoria J. Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2), 2004. (Cited on page 7.)
- [HACH12] Martin Harrigan, Daniel Archambault, Pádraig Cunningham, and Neil Hurley. Egonav: exploring networks through ego-centric spatializations. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 563–570. ACM, 2012. (Cited on page 54.)
- [Haw80] Douglas M Hawkins. *Identification of Outliers*. Chapman and Hall, 1980. (Cited on pages 18 and 73.)
- [HB05] Jeffrey Heer and Danah Boyd. Vizster: Visualizing online social networks. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 32–39. IEEE, 2005. (Cited on page 43.)
- [HBE95] Christopher G Healey, Kellogg S Booth, and James T Enns. Visualizing real-time multivariate data using preattentive processing. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 5(3):190–221, 1995. (Cited on page 29.)

- [HBE96] Christopher G Healey, Kellogg S Booth, and James T Enns. High-speed visual estimation using preattentive processing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 3(2):107–135, 1996. (Cited on page 44.)
- [HFM07] Nathalie Henry, J-D Fekete, and Michael J McGuffin. Node-trix: a hybrid visualization of social networks. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1302–1309, 2007. (Cited on pages 34, 36 and 42.)
- [HH11] Kasper Hornbæk and Morten Hertzum. The notion of overview in information visualization. *International Journal of Human-Computer Studies*, 69(7):509–525, 2011. (Cited on page 41.)
- [HHY⁺10] Lane Harrison, Xianlin Hu, Xiaowei Ying, Aidong Lu, Weichao Wang, and Xintao Wu. Interactive detection of network anomalies via coordinated multiple views. In *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, pages 91–101. ACM, 2010. (Cited on page 63.)
- [HKKS04] John Hopcroft, Omar Khan, Brian Kulis, and Bart Selman. Tracking evolving communities in large linked networks. *Proceedings of the national academy of sciences of the United States of America*, 101(Suppl 1):5249–5253, 2004. (Cited on page 80.)
- [HLG13a] Sébastien Heymann and Bénédicte Le Grand. Exploratory network analysis: Visualization and interaction. In *Complex Networks and their Applications*. 2013. (Cited on page 55.)
- [HLG13b] Sébastien Heymann and Bénédicte Le Grand. Monitoring user-system interactions through graph-based intrinsic dynamics analysis. In *Proc. IEEE International Conference on Research Challenges in Information Science (RCIS 2013)*, IEEE. IEEE, 2013. (Cited on page 125.)
- [HLM10] Assia Hamzaoui, Matthieu Latapy, and Clémence Magnien. Detecting events in the dynamics of ego-centered measurements of the internet topology. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 505–512. IEEE, 2010. (Cited on page 22.)

- [HLM12] Sébastien Heymann, Matthieu Latapy, and Clémence Magnien. Outskewer source code, 2012. (Cited on page 71.)
- [HMPKE07] Petter Holme, Sung Min Park, Beom Jun Kim, and Christofer R Edling. Korean university life in a network perspective: Dynamics of a large affiliation network. *Physica A: Statistical Mechanics and its Applications*, 373:821–830, 2007. (Cited on page 103.)
- [HS12] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012. (Cited on pages 6 and 103.)
- [HSS⁺97] Amy E Hurley, Terri A Scandura, Chester A Schriesheim, Michael T Brannick, Anson Seers, Robert J Vandenberg, and Larry J Williams. Exploratory and confirmatory factor analysis: Guidelines, issues, and alternatives. *Journal of Organizational Behavior*, 18(6):667–683, 1997. (Cited on page 10.)
- [HSS11] Steffen Hadlak, H-J Schulz, and Heidrun Schumann. In situ exploration of large dynamic networks. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2334–2343, 2011. (Cited on page 60.)
- [IK01] Laurent Itti and Christof Koch. Computational modelling of visual attention. *Nature reviews neuroscience*, 2(3):194–203, 2001. (Cited on page 29.)
- [IK04] Tsuyoshi Ide and Hisashi Kashima. Eigenspace-based anomaly detection in computer systems. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 440–449. ACM, 2004. (Cited on page 79.)
- [ISB⁺11] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. What’s in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1):166–180, 2011. (Cited on page 103.)
- [JHVB11] Mathieu Jacomy, Sébastien Heymann, Tommaso Venturini, and Mathieu Bastian. Forceatlas2, a graph layout algorithm for handy network visualization. *Paris <http://www.medialab.sciences-po.fr/fr/publications-fr>*, 2011. (Cited on pages 44 and 124.)

- [KA12] Hyounghick Kim and Ross Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107, 2012. (Cited on page 103.)
- [Kan81] Immanuel Kant. *Kritik der reinen Vernunft*. 1781. (Cited on page 110.)
- [KB09] Richard Klavans and Kevin W Boyack. Toward a consensus map of science. *Journal of the American Society for information science and technology*, 60(3):455–476, 2009. (Cited on page 37.)
- [KKB⁺12] Gautier Krings, Márton Karsai, Sebastian Bernharsson, Vincent D Blondel, and Jari Saramäki. Effects of time window size and placement on the structure of aggregated networks. *CoRR*, 2012. (Cited on pages 6, 105 and 106.)
- [KKK00] David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 504–513. ACM, 2000. (Cited on pages 4, 6 and 17.)
- [KKK⁺11] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005, 2011. (Cited on pages 6 and 103.)
- [KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003. (Cited on page 4.)
- [Kle02] Jon Kleinberg. Bursty and hierarchical structure in streams. In *Proc. 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 91–101. ACM, 2002. (Cited on page 73.)
- [KMSZ06] Daniel A Keim, Florian Mansmann, Jörn Schneidewind, and Hartmut Ziegler. Challenges in visual data analysis. In *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pages 9–16. IEEE, 2006. (Cited on page 53.)
- [Kob12] Stephen G. Kobourov. Force-directed drawing algorithms. In *Handbook of Graph Drawing and Visualization*. CRC Press, 2012. (Cited on pages 34 and 44.)

- [Kof35] Kurt Koffka. Principles of gestalt psychology. 1935. (Cited on page 31.)
- [KPS12] Simone Kriglstein, Margit Pohl, and Claus Stachl. Animation for time-oriented data: An overview of empirical research. In *Information Visualisation (IV), 2012 16th International Conference on*, pages 30–35. IEEE, 2012. (Cited on page 59.)
- [Kre05] Lothar Krempel. *Visualisierung komplexer Strukturen*. Campus, 2005. (Cited on page 58.)
- [KW05] Thomas Kapler and William Wright. Geotime information visualization. *Information Visualization*, 4(2):136–146, 2005. (Cited on page 58.)
- [KW06] Gueorgi Kossinets and Duncan J Watts. Empirical analysis of an evolving social network. *Science*, 311(5757):88–90, 2006. (Cited on page 102.)
- [LAM⁺05] Yarden Livnat, Jim Agutter, Shaun Moon, Robert F Erbacher, and Stefano Foresti. A visualization paradigm for network intrusion detection. In *Information Assurance Workshop, 2005. IAW’05. Proceedings from the Sixth Annual IEEE SMC*, pages 92–99. IEEE, 2005. (Cited on pages 59 and 63.)
- [Lat95] Bruno Latour. The ‘pedofil’ of boa vista: a photo-philosophical montage. *Common Knowledge*, 4(1):144–187, 1995. (Cited on pages 11 and 12.)
- [LBVS10] Qi Liao, Andrew Blaich, Dirk VanBruggen, and Aaron Striegel. Managing networks through context: Graph visualization and exploration. *Computer Networks*, 54(16):2809–2824, 2010. (Cited on page 62.)
- [LJRH11] Do Quoc Le, Taeyoel Jeong, H Eduardo Roman, and James Won-Ki Hong. Traffic dispersion graph based anomaly detection. In *Proceedings of the Second Symposium on Information and Communication Technology*, pages 36–41. ACM, 2011. (Cited on pages 53 and 63.)
- [LJV⁺] Bruno Latour, Pablo Jensen, Tommaso Venturini, Sébastien Grauwin, and Dominique Boullier. The whole is always smaller than its parts: A digital test of gabriel tarde’s monads. *to appear in British Journal of Sociology*. (Cited on page 57.)

- [LKF05] Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Knowledge Discovery and Data Mining*, pages 177–187, 2005. (Cited on pages 102 and 103.)
- [LMDV08] Matthieu Latapy, Clémence Magnien, and Nathalie Del Vecchio. Basic notions for the analysis of large two-mode networks. *Social Networks*, 30(1), 2008. (Cited on page 19.)
- [LMF] Matthieu Latapy, Clémence Magnien, and Raphaël Fournier. Quantifying paedophile activity in a large P2P system. *Information Processing and Management*. to appear. (Cited on pages 4 and 24.)
- [LMO08] Matthieu Latapy, Clémence Magnien, and Frédéric Ouédraogo. A radar for the internet. In *IEEE International Conference on Data Mining*, volume abs/0807.1, pages 901–908, 2008. (Cited on pages 4 and 103.)
- [LMO11] Matthieu Latapy, Clemence Magnien, and Frederic Ouédraogo. A radar for the internet. *Complex Systems*, 20(1), 2011. (Cited on page 22.)
- [LMS⁺08] Mayank Lahiri, Arun S Maiya, Rajmonda Sulo, Tanya Y Berger Wolf, et al. The impact of structural changes on predictions of diffusion in networks. In *Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference on*, pages 939–948. IEEE, 2008. (Cited on page 106.)
- [LPP⁺06] Bongshin Lee, Cynthia Sims Parr, Catherine Plaisant, Benjamin B Bederson, Vladislav Daniel Veksler, Wayne D Gray, and Christopher Kotfila. Treeplus: Interactive exploration of networks with enhanced tree layouts. *Visualization and Computer Graphics, IEEE Transactions on*, 12(6):1414–1426, 2006. (Cited on pages 43 and 49.)
- [LSH⁺11] Qi Liao, Lei Shi, Yuan He, Rui Li, Zhong Su, Aaron Striegel, and Yunhao Liu. Visualizing anomalies in sensor networks. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 460–461. ACM, 2011. (Cited on pages 53 and 63.)
- [Mac86] Jock Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics (TOG)*, 5(2):110–141, 1986. (Cited on page 29.)

- [Mar90] Peter V Marsden. Network data and measurement. *Annual review of sociology*, pages 435–463, 1990. (Cited on page 103.)
- [MCF07] Terje Midtbø, K Clarke, and S Fabrikant. Human interaction with animated maps: The portrayal of the passage of time. In *11th Scandinavian Research Conference on Geographical Information Science*, 2007. (Cited on page 59.)
- [MFKN09] Florian Mansmann, Fabian Fischer, Daniel A Keim, and Stephen C North. Visual support for analyzing network traffic and intrusion detection events using treemap and graph representations. In *Proceedings of the Symposium on Computer Human Interaction for the Management of Information Technology*, page 3. ACM, 2009. (Cited on pages 53 and 63.)
- [MHYLG13] A Mouakher, S Heymann, S Ben Yahia, and B Le Grand. Efficient visualization of folksonomies based on «intersectors». In *Flexible Query Answering Systems*, pages 637–648. Springer, 2013. (Cited on page 53.)
- [MMBd05] James Moody, Daniel McFarland, and Skye Bender-deMoll. Dynamic network visualization. *American Journal of Sociology*, 110(4):1206–1241, 2005. (Cited on page 122.)
- [MMF⁺08] Mark R. Meiss, Filippo Menczer, Santo Fortunato, Alessandro Flammin, and Alessandro Vespignani. Ranking web sites with real user traffic. In *Proc. ACM International Conference on Advances in Social Networks Analysis and Mining (WSDM’08)*. ACM, 2008. (Cited on page 111.)
- [Mor37] Jakob L Moreno. Sociometry in relation to other social sciences. *Sociometry*, 1(1/2):206–219, 1937. (Cited on page 28.)
- [MVC⁺10] Luca Masud, Francesca Valsecchi, Paolo Ciuccarelli, Donato Ricci, and Giorgio Caviglia. From data to knowledge-visualizations as transformation processes within the data-information-knowledge continuum. In *Information Visualization (IV), 2010 14th International Conference*, pages 445–449. IEEE, 2010. (Cited on page 66.)
- [NC03] Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003. (Cited on page 71.)

- [New87] Isaac Newton. *Philosophiæ Naturalis Principia Mathematica*. 1687. (Cited on page 110.)
- [New03] Mark EJ Newman. Ego-centered networks and the ripple effect. *Social Networks*, 25(1):83–95, 2003. (Cited on page 51.)
- [NJKJ05] Steven Noel, Michael Jacobs, Pramod Kalapa, and Sushil Jajodia. Multiple coordinated views for network attack graphs. In *Visualization for Computer Security, 2005.(VizSEC 05). IEEE Workshop on*, pages 99–106. IEEE, 2005. (Cited on page 63.)
- [Noa09] Andreas Noack. Modularity clustering is force-directed layout. *Physical Review E*, 79(2):026102, 2009. (Cited on page 55.)
- [Nor88] Donald A Norman. *The psychology of everyday things*. Basic books, 1988. (Cited on page 52.)
- [NPNB08] Tamás Nepusz, Andrea Petróczi, László Négyessy, and Fülöp Bacsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77(1):016107, 2008. (Cited on page 72.)
- [NSGS07] Galileo Mark Namata, Brian Staats, Lise Getoor, and Ben Shneiderman. A dual-view approach to interactive network visualization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 939–942. ACM, 2007. (Cited on page 53.)
- [NTM⁺13] Vincenzo Nicosia, John Tang, Cecilia Mascolo, Mirco Musolesi, Giovanni Russo, and Vito Latora. Graph metrics for temporal networks. In *Temporal Networks*, pages 15–40. Springer, 2013. (Cited on page 103.)
- [NTY01] Kumiyo Nakakoji, Akio Takashima, and Yasuhiro Yamamoto. Cognitive effects of animated visualization in exploratory visual data analysis. In *Information Visualisation, 2001. Proceedings. Fifth International Conference on*, pages 77–84. IEEE, 2001. (Cited on pages 58 and 59.)
- [OdC04] Roberto N Onody and Paulo A de Castro. Complex network study of brazilian soccer players. *Physical Review E*, 70(3):037103, 2004. (Cited on page 103.)

- [Pap06] Spiros Papadimitriou. Optimal multi-scale patterns in time series streams. In *Proc. ACM International Conference on Management of Data (SIGMOD'06)*, pages 647–658. ACM, 2006. (Cited on page 105.)
- [PBC⁺11] André Panisson, Alain Barrat, Ciro Cattuto, Wouter Van den Broeck, Giancarlo Ruffo, and Rossano Schifanella. On the dynamics of human proximity for data diffusion in ad-hoc networks. *Ad Hoc Networks*, 2011. (Cited on pages 107, 108 and 111.)
- [PBV07] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, 2007. (Cited on page 80.)
- [PCMP05] Carey E Priebe, John M Conroy, David J Marchette, and Youngser Park. Scan statistics on enron graphs. *Computational & Mathematical Organization Theory*, 11(3):229–247, 2005. (Cited on page 77.)
- [PDFV05] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005. (Cited on page 72.)
- [PDGM10] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications*, 1(1):19–30, 2010. (Cited on pages 78 and 79.)
- [Pei74] Charles Sanders Peirce. *Collected papers of charles sanders peirce*, volume 3. Harvard University Press, 1974. (Cited on page 10.)
- [Pet11] Elisha Peterson. Time spring layout for visualization of dynamic social networks. In *Network Science Workshop (NSW), 2011 IEEE*, pages 98–104. IEEE, 2011. (Cited on page 60.)
- [PHG07] Helen C Purchase, Eve Hoggan, and Carsten Görg. How important is the “mental map”?—an empirical investigation of a dynamic graph layout algorithm. In *Graph drawing*, pages 184–195. Springer, 2007. (Cited on page 60.)
- [Pin05] B Pincombe. Anomaly detection in time series of graphs using arma processes. *ASOR BULLETIN*, 24(4):2, 2005. (Cited on pages 77 and 78.)

- [Pin07] Brandon Pincombe. Detecting changes in time series of network graphs using minimum mean squared error and cumulative summation. *ANZIAM Journal*, 48:C450–C473, 2007. (Cited on page 77.)
- [PKGf03] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B. Gibbons, and Christos Faloutsos. LOCI: fast outlier detection using the local correlation integral. In *Proc. 19th International Conference on Data Engineering (ICDE)*. IEEE, 2003. (Cited on page 7.)
- [PLL07] Dragoljub Pokrajac, Aleksandar Lazarevic, and Longin Jan Latecki. Incremental local outlier detection for data streams. In *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*, pages 504–515. IEEE, 2007. (Cited on page 75.)
- [PS06] Adam Perer and Ben Shneiderman. Balancing systematic and flexible exploration of social networks. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):693–700, 2006. (Cited on page 42.)
- [PS08] Adam Perer and Ben Shneiderman. Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 265–274. ACM, 2008. (Cited on page 54.)
- [PS11] Raj Kumar Pan and Jari Saramäki. Path lengths, correlations, and centrality in temporal networks. *Physical Review E*, 84(1):016105, 2011. (Cited on page 103.)
- [Pur97] Helen Purchase. Which aesthetic has the greatest effect on human understanding? In *Graph Drawing*, pages 248–261. Springer, 1997. (Cited on page 60.)
- [PvH12] Adam Perer and Frank van Ham. Integrating querying and browsing in partial graph visualizations. Technical report, IBM Technical Report 12-01, 2012. (Cited on page 49.)
- [QHP12] Lin Quan, John Heidemann, and Yuri Pradkin. Visualizing sparse internet events: Network outages and route changes. *Computing*, pages 1–13, 2012. (Cited on page 63.)
- [RFF⁺08] George Robertson, Roland Fernandez, Danyel Fisher, Bongshin Lee, and John Stasko. Effectiveness of animation in trend

- visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1325–1332, 2008. (Cited on page 59.)
- [Ris89] Jorma Rissanen. *Stochastic complexity in statistical inquiry theory*. World Scientific Publishing Co., Inc., 1989. (Cited on page 71.)
- [RJTT⁺06] José F Rodrigues Jr, Hanghang Tong, Agma JM Traina, Christos Faloutsos, and Jure Leskovec. Gmine: a system for scalable, interactive graph visualization and mining. In *Proceedings of the 32nd international conference on Very large data bases*, pages 1195–1198. VLDB Endowment, 2006. (Cited on page 53.)
- [RL87] Peter J Rousseeuw and Annick M Leroy. Robust regression and outlier detection. *Wiley Series in Probability and Mathematical Statistics, New York: Wiley, 1987*, 1, 1987. (Cited on page 73.)
- [RL05] Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. Wiley. com, 2005. (Cited on page 10.)
- [RL09] Konrad Rieck and Pavel Laskov. Visualization and explanation of payload-based anomaly detection. In *Computer Network Defense (EC2ND), 2009 European Conference on*, pages 29–36. IEEE, 2009. (Cited on page 63.)
- [RLNZ09] Galen Reeves, Jie Liu, Suman Nath, and Feng Zhao. Managing massive time series streams with multiscale compressed trickles. *Proceedings of The Vldb Endowment*, 2:97–108, 2009. (Cited on page 106.)
- [SA05] Jitesh Shetty and Jafar Adibi. Discovering important nodes through graph entropy the case of enron email database. In *Proceedings of the 3rd international workshop on Link discovery*, pages 74–81. ACM, 2005. (Cited on page 71.)
- [SBF⁺08] Antoine Scherrer, Pierre Borgnat, Eric Fleury, J-L Guillaume, and Céline Robardet. Description and simulation of dynamic mobility networks. *Computer Networks*, 52(15):2842–2858, 2008. (Cited on page 4.)
- [SCG10] Karsten Steinhaeuser, Nitesh V Chawla, and Auroop R Ganguly. An exploration of climate data using complex networks.

- ACM SIGKDD Explorations Newsletter*, 12(1):25–32, 2010. (Cited on page 80.)
- [SFPY07] Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 687–696. ACM, 2007. (Cited on page 61.)
- [SG12] Massoud Seifi and Jean-Loup Guillaume. Community cores in evolving networks. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 1173–1180. ACM, 2012. (Cited on pages 82 and 103.)
- [Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996. (Cited on pages 40 and 44.)
- [SMM13] Arnaud Sallaberry, Chris Muelder, and Kwan-Liu Ma. Clustering, visualizing, and navigating for large dynamic graphs. In *Graph Drawing*, pages 487–498. Springer, 2013. (Cited on page 60.)
- [SMO⁺03] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504, 2003. (Cited on page 41.)
- [SNDC10] Sebastian Schmidt, Miguel A Nacenta, Raimund Dachsel, and Sheelagh Carpendale. A set of multi-touch graph interaction techniques. In *ACM International Conference on Interactive Tabletops and Surfaces*, pages 113–116. ACM, 2010. (Cited on page 40.)
- [Spe07] Robert Spence. Information visualization: Design for interaction, 2007. (Cited on page 41.)
- [SQCF05] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Data Mining, Fifth IEEE International Conference on*, pages 8–pp. IEEE, 2005. (Cited on page 72.)

- [SQF⁺11] Nicola Santoro, Walter Quattrociocchi, Paola Flocchini, Arnaud Casteigts, and Frederic Amblard. Time-varying graphs and social network analysis: Temporal indicators and metrics. *arXiv preprint arXiv:1102.0629*, 2011. (Cited on page 103.)
- [SSMF⁺09] Marc A Smith, Ben Shneiderman, Natasa Milic-Frayling, Eduarda Mendes Rodrigues, Vladimir Barash, Cody Dunne, Tony Capone, Adam Perer, and Eric Gleave. Analyzing (social media) networks with nodexl. In *Proceedings of the fourth international conference on Communities and technologies*, pages 255–264. ACM, 2009. (Cited on page 122.)
- [SSTP09] Matthew C Schmidt, Nagiza F Samatova, Kevin Thomas, and Byung-Hoon Park. A scalable, parallel algorithm for maximal clique enumeration. *Journal of Parallel and Distributed Computing*, 69(4):417–428, 2009. (Cited on page 72.)
- [STH02] Chris Stolte, Diane Tang, and Pat Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):52–65, 2002. (Cited on page 29.)
- [SWS10] Klaus Stein, René Wegener, and Christoph Schlieder. Pixel-oriented visualization of change in social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 233–240. IEEE, 2010. (Cited on page 59.)
- [SWW11] Lei Shi, Chen Wang, and Zhen Wen. Dynamic network visualization in 1.5 d. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE*, pages 179–186. IEEE, 2011. (Cited on pages 57, 59, 60 and 61.)
- [Tam07] Roberto Tamassia. *Handbook of graph drawing and visualization*. Chapman & Hall/CRC, 2007. (Cited on page 36.)
- [TBWK07] Chayant Tantipathananandh, Tanya Berger-Wolf, and David Kempe. A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 717–726. ACM, 2007. (Cited on page 80.)
- [TH10] Yuri Takhteyev and Andrew Hilts. Investigating the geography of open source software through github, 2010. (Cited on page 111.)

- [Tho12] Brian Thompson. The early bird gets the buzz: detecting anomalies and emerging trends in information networks. In *Proceedings of the fifth ACM international conference on Web search and data mining, WSDM '12*, pages 759–760, New York, NY, USA, 2012. ACM. (Cited on pages 82 and 103.)
- [TK07] Tatiana Tekusova and Jörn Kohlhammer. Applying animation to the visual analysis of financial time-dependent data. In *Information Visualization, 2007. IV'07. 11th International Conference*, pages 101–108. IEEE, 2007. (Cited on page 58.)
- [TKSP08] Tatiana Tekušová, Jörn Kohlhammer, Slawomir J Skwarek, and Galina V Paramei. Perception of direction changes in animated data visualization. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*, pages 205–205. ACM, 2008. (Cited on page 59.)
- [TLS⁺13] John Tang, Ilias Leontiadis, Salvatore Scellato, Vincenzo Nicosia, Cecilia Mascolo, Mirco Musolesi, and Vito Latora. Applications of temporal graph metrics to real-world networks. In *Temporal Networks*, pages 135–159. Springer, 2013. (Cited on page 103.)
- [TMML09] John Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 31–36. ACM, 2009. (Cited on page 103.)
- [TMML10] John Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Characterising temporal distance and reachability in mobile and online social networks. *ACM SIGCOMM Computer Communication Review*, 40(1):118–124, 2010. (Cited on page 103.)
- [TMML11] John Tang, Cecilia Mascolo, Mirco Musolesi, and Vito Latora. Exploiting temporal complex network metrics in mobile malware containment. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pages 1–9. IEEE, 2011. (Cited on page 102.)
- [Tre85] Anne Treisman. Preattentive processing in vision. *Computer vision, graphics, and image processing*, 31(2):156–177, 1985. (Cited on page 29.)
- [Tuf06] Edward R Tufte. *Beautiful evidence*, volume 23. Graphics Press Cheshire, CT, 2006. (Cited on page 64.)

- [Tuk62] John W Tukey. The future of data analysis. *The Annals of Mathematical Statistics*, 33(1):1–67, 1962. (Cited on page 10.)
- [Tuk77] John W Tukey. Exploratory data analysis. *Reading, MA*, 231, 1977. (Cited on pages 32 and 45.)
- [Tuk80] John W Tukey. We need both exploratory and confirmatory. *The American Statistician*, 34(1):23–25, 1980. (Cited on page 15.)
- [TY06] Jun-ichi Takeuchi and Kenji Yamanishi. A unifying framework for detecting outliers and change points from time series. *IEEE Trans. on Knowl. and Data Eng.*, 18(4):482–492, 2006. (Cited on page 74.)
- [UCC⁺13] Giorgio Uboldi, Giorgio Caviglia, Nicole Coleman, Sébastien Heymann, Glaucio Mantegari, and Paolo Ciuccarelli. Knot: an interface for the study of social networks in the humanities. In *Proceedings of the Biannual Conference of the Italian Chapter of SIGCHI*, page 15. ACM, 2013. (Cited on page 66.)
- [UKBM11] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. The anatomy of the facebook social graph, nov 2011. (Cited on page 111.)
- [VGN08] Nguyen Hoang Vu, Vivekanand Gopalkrishnan, and Praneeth Namburi. Online outlier detection based on relative neighbourhood dissimilarity. In *Web Information Systems Engineering-WISE 2008*, pages 50–61. Springer, 2008. (Cited on page 76.)
- [vHP09] Frank van Ham and Adam Perer. "search, show context, expand on demand": Supporting large graph exploration with degree-of-interest. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):953–960, 2009. (Cited on pages 13, 43, 49, 50, 51 and 52.)
- [VOB05] Alexei Vázquez, Joao G Oliveira, and Albert-László Barabási. Inhomogeneous evolution of subgraphs and cycles in complex networks. *Physical Review E*, 71(2):025103, 2005. (Cited on page 103.)
- [War00] Colin Ware. *Information visualization*, volume 2. Morgan Kaufmann San Francisco, 2000. (Cited on pages 29 and 31.)

- [Was94] Stanley Wasserman. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994. (Cited on pages 51 and 103.)
- [WDdACG12a] John Whitbeck, Marcelo Dias de Amorim, Vania Conan, and Jean-Loup Guillaume. Temporal reachability graphs. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 377–388. ACM, 2012. (Cited on page 103.)
- [WDdACG12b] John Whitbeck, Marcelo Dias de Amorim, Vania Conan, and Jean-Loup Guillaume. Temporal reachability graphs. pages 377–388, 2012. (Cited on page 111.)
- [Won10] Bang Wong. Points of view: Saliency. *Nature Methods*, 7(10):773–773, 2010. (Cited on page 30.)
- [WS98] Duncan J Watts and Steven H Strogatz. Collective dynamics of ‘small-world’ networks. *nature*, 393(6684):440–442, 1998. (Cited on pages 47 and 72.)
- [WZF11] Florian Windhager, Lukas Zenk, and Paolo Federico. Visual enterprise network analytics-visualizing organizational change. *Procedia-Social and Behavioral Sciences*, 22:59–68, 2011. (Cited on pages 57, 58 and 122.)
- [XFJ03] B Bui Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(02):267–285, 2003. (Cited on page 103.)
- [YAPM08] Xintian Yang, Sitaram Asur, Srinivasan Parthasarathy, and Sameep Mehta. A visual-analytic toolkit for dynamic interaction graphs. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1016–1024. ACM, 2008. (Cited on page 62.)
- [YRW09] Di Yang, Elke A Rundensteiner, and Matthew O Ward. Neighbor-based pattern detection for windows over streaming data. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 529–540. ACM, 2009. (Cited on page 74.)
- [ZLBM06] Qiankun Zhao, Tie-yan Liu, Sourav S. Bhowmick, and Weiyang Ma. Event detection from evolution of click-through

data. In *Knowledge Discovery and Data Mining*, pages 484–493, 2006. (Cited on pages 78 and 103.)

- [ZZJ⁺08] Lingsong Zhang, Zhengyuan Zhu, Kevin Jeffay, James Steve Marron, and Frank Donelson Smith. Multi-resolution anomaly detection for the internet. In *Proc. IEEE Conference on Computer Communications Workshops (INFOCOM)*. IEEE, 2008. (Cited on pages 74 and 105.)