Académie de Lyon

Université Claude Bernard Lyon I

Habilitation à Diriger des Recherches

préparée au Laboratoire de l'Informatique du Parallélisme
ENS de Lyon, UCBL, CNRS et INRIA

Spécialité : **Informatique**

Structures of Complex Networks and of their Dynamics

par

Christophe Crespelle

Préalablement rapportée par :

M. Ulrik Brandes, Professeur, Université de Konstanz .... rapporteur
M. Pierre Fraigniaud, Directeur de recherche, CNRS .... rapporteur
M. Renaud Lambiotte, Professeur, Université d'Oxford .... rapporteur

Soutenue le 26 septembre 2017 devant le jury composé de :

M. Pierre Fraigniaud, Directeur de recherche, CNRS .... rapporteur
Mme. Isabelle Guérin-Lassous, Professeur, Univ. Claude Bernard Lyon 1 ... examinatrice
M. Bertrand Jouve, Directeur de recherche, CNRS .... examinateur
M. Renaud Lambiotte, Professeur, Université d'Oxford .... rapporteur
M. Matthieu Latapy, Directeur de recherche, CNRS .... examinateur
M. Rolf Niedermeier, Professeur, Université technique de Berlin .... examinateur

À Dương

Résumé des travaux

Ce manuscrit présente une synthèse de mes travaux de recherche dans le domaine des réseaux complexes. Ces activités ont commencé en 2008, un an après l'obtention de mon doctorat en informatique. Ces travaux touchent à des problématiques diverses du domaine : la métrologie de l'Internet, l'analyse des réseaux dynamiques, la modélisation des réseaux statiques et les codages efficaces pour les graphes. Chacune de ces thématiques donne lieu à un chapitre. Une caractéristique essentielle de ce mémoire est qu'il mêle l'utilisation de méthodes empiriques (mesure et simulation) et de méthodes formelles (preuve). C'est une particularité de mon activité de recherche et ce manuscrit est écrit avec l'intention d'illustrer comment ces deux ensembles de méthodes, souvent séparés, peuvent s'enrichir mutuellement.

Le premier chapitre traite de métrologie, qui est la science de la mesure. Du fait qu'ils proviennent de contextes concrets, les réseaux complexes ne peuvent être connus que par une opération de mesure. Dès lors, se pose la question de savoir si le résultat de la mesure est fidèle au réseau réel ou s'il est induit par le procédé de mesure lui-même, auquel cas on parle de *biais.* Une grande controverse a éclaté à ce propos concernant la distribution des degrés de l'Internet. Toutes les mesures effectuées jusqu'à présent ont confirmé que cette distribution est en loi de puissance. Cependant, il a été montré, à la fois empiriquement et formellement, que le résultat de ces mesures ainsi que

le procédé utilisé pour les obtenir sont biaisés. Il en ressort que nous n'avons actuellement à disposition aucune estimation fiable de cette distribution, qui est pourtant d'importance primordiale pour la gestion du réseau. Ce premier chapitre présente une méthode de mesure pour estimer rigoureusement la distribution des degrés du coeur de l'Internet. Deux implémentations de la méthode sont exposées : une dédiée à la topologie logique, au niveau IP, et une dédiée à la topologie physique.

Le deuxième chapitre concerne l'analyse des réseaux dynamiques. Il contient une étude de cas et deux travaux méthodologiques. L'étude de cas porte sur le réseau dynamique des contacts entre individus dans un hôpital, enregistrés avec une précision de 30s pendant une durée de six mois sur toute la population de l'hôpital, patients et personnels. Cette mesure a été effectuée dans le but de mieux comprendre la diffusion des souches de staphylocoque en milieu hospitalier et l'apparition de souches résistantes aux antibiotiques. Nous présentons une analyse de la structure des contacts, dans sa dimension topologique et sa dimension temporelle, selon un découpage prédéfini de l'hôpital en services et en catégories socio-professionnelles. Le premier des deux travaux méthodologiques concerne la structure des changements de la topologie d'un réseau dynamique au cours du temps. Le réseau est décrit comme une série de graphes et on s'intéresse aux différences entre deux graphes consécutifs de la série. La question est de savoir si les changements de topologie d'un instant à l'autre sont répartis dans l'ensemble du réseau ou s'ils sont au contraire concentrés autour d'une partie restreinte des noeuds. Le deuxième travail méthodologique concerne la description des réseaux dynamiques par une série de graphes. Nombre de réseaux dynamiques sont naturellement des *flots de liens*, c'est-à-dire un ensemble de triplets (u, v, t) signifiant qu'il y a un lien entre les noeuds u et v au temps t. La plupart des travaux sur ces réseaux commencent par les transformer en séries de graphes sur lesquelles sont menées toutes les analyses. Le procédé utilisé pour ce faire est l'*agrégation*. Il consiste à former le graphe des liens ayant existé dans une fenêtre de temps choisie. Malheureusement, cette transformation induit une perte d'information qui altère le flot de liens original. Cette altération est d'autant plus grande que la largeur de la fenêtre utilisée est grande. Nous proposons une méthode pour déterminer quelle elle est la largeur maximale de la fenêtre d'agrégation qui garantisse que la série de graphes formée conserve, pour l'essentiel, les propriétés du flot de liens original.

Le troisième chapitre est dédié à la modélisation des réseaux statiques, c'est-à-dire la génération de topologies synthétiques réalistes. Le but est de générer aléatoirement des graphes qui ont les propriétés connues des réseaux issus du monde réel, en particulier : une faible densité globale, des distances courtes, une distribution des degrés hétérogène et une densité locale élevée (appréciée par le coefficient de *clustering*). La méthode connue sous le nom de *modèle de configuration* permet de générer des graphes présentant les trois premières propriétés. Depuis, le domaine bute sur la difficulté à générer aléatoirement des graphes ayant une forte densité locale. Nous explorons deux nouvelles voies pour lever cette difficulté. La première consiste à générer des graphes non par leurs arêtes mais par leurs cliques, en respectant la structure de chevauchement de ces dernières. Cela soulève le problème de la terminaison d'un procédé itératif de factorisation de bicliques dans un graphe multiparti, pour lequel nous élaborons deux solutions. La deuxième voie de modélisation que nous proposons consiste à approximer les réseaux réels par des graphes fortement structurés, c'est-à-dire définis par une propriété mathématique. Il s'agit de représenter un réseau complexe par une paire formée d'un graphe fortement structuré et de l'ensemble des différences entre ce graphe et le réseau original, ces deux parties de la topologie étant ensuite générées indépendamment. Dans le but d'obtenir de telles représentations pour des réseaux issus du monde réel, nous développons ou améliorons plusieurs algorithmes d'édition et de complétion minimale de graphes, notamment pour les classes des graphes d'intervalles, des graphes de permutation et des cographes. L'approche de modélisation proposée est testée en utilisant les résultats fournis par l'algorithme d'édition minimale développé pour les cographes.

Le but du quatrième et dernier chapitre est de développer des codages efficaces pour les graphes, qui soient à la fois compacts en espace et qui ne pénalisent pas le temps d'exécution des requêtes faites par les algorithmes. Cela est primordial en pratique pour stocker en mémoire limitée les immenses jeux de données constitués par les réseaux complexes sans allonger les temps de traitement de ces données. Nous nous intéressons à garantir un temps d'exécution optimal pour la requête de voisinage (lister les voisins d'un sommet donné), qui est probablement la requête la plus utilisée par les algorithmes et qui est également utilisée pour l'exploration et la visualisation. Deux paramètres de codage, la *contiguïté* et la *linéarité*, sont étudiés. Ils sont basés sur un ou plusieurs ordres linéaires des sommets du graphe considéré, dont le but est de grouper autant que possible les voisinages des sommets. La contiguïté utilise un seul ordre dans lequel les voisinages des sommets peuvent être segmentés en plusieurs intervalles. La linéarité, que nous introduisons, utilise plusieurs ordres dans lequel chaque sommet retient un unique intervalle formé par certains de ses voisins. Il découle de leurs définitions que la linéarité est toujours au plus égale à la contiguïté. Nous montrons qu'il existe des familles de graphes pour lesquelles la linéarité est asymptotiquement négligeable devant la contiguïté, ce qui implique que le codage par linéarité est strictement plus puissant que celui par contiguïté. Au passage, nous fournissons des bornes supérieures et inférieures atteintes sur la contiguïté et la linéarité dans le pire des cas des cographes à n sommets.

Le manuscrit se termine en décrivant deux directions de recherche ouvertes que je crois particulièrement importantes pour le domaine des réseaux complexes dans les prochaines années. La première est le développement d'une théorie des *flots de liens*, comme un nouvel objet mathématique, pour étudier les réseaux dynamiques sans passer par les graphes, ce qui est actuellement à l'origine de nombreux blocages. La deuxième direction concerne l'approximation des réseaux complexes par des graphes fortement structurés et l'avènement d'une théorie algorithmique des *graphes presque structurés.*

Contents

Introduction

This thesis presents my research work in the field of complex networks, started in 2008, one year after the end of my PhD in computer science. It is intended to give a synthetic description of my contributions and does not provide the details of their technical content, which can be found in the corresponding articles. Instead, in this manuscript, I concentrate on, and discuss in depth, the meaning of the results I obtained, the main ideas that support them and the perspectives they raise. As a result, this document can be used by non-specialists as an introduction to the field, through a very partial sample of some of its problematics, or by specialists to have a synthetic view of my contributions and of some open research directions they raise.

This manuscript also positions my results in the general picture of the field, but avoids to make an exhaustive description of it. The domain is so vast that making its exhaustive description is practically unfeasible and would give an outcome quite unexciting to read. Instead, I prefer to mention the articles that had a strong influence on the field or on my own research, and the articles that open new directions in which I particularly believe. Extensive and specific states of the art of the topics I worked on can be found in the articles I published.

Almost all works presented here have been accomplished in collaboration with other researchers and many of them in collaboration with students that I advised or co-advised for their research internships or for their PhD thesis. In particular, the work on Internet metrology presented in the first chapter has been accomplished within the PhD thesis of Elie Rotenberg, which I co-advised with Matthieu Latapy. In Chapter 2, the analyses of the contact network between individuals in a hospital have been realised in the PhD thesis of Lucie Martinet and the work on aggregation of link streams into series of graphs is part of the PhD thesis of Yannick Léo, whom I both co-advised with Eric Fleury.

The synthesis of my research activity presented here leans on articles that I cite throughout the document. All of them are either published in journals or acts of conferences or under review at the time I write this manuscript, all texts being available on my personal webpage. Only the last part of Chapter 3 diverges from this rule and describes an original work that has not been published nor presented anywhere else previously. This is the reason why this part is technically more detailed than the rest of the document.

Complex networks

A *network* (see example in Figure 1) is a set of entities, called *nodes*, which are involved in some pairwise relationships between them, called *links*. *Complex networks* are those networks encountered in practice in various contexts, such as computer science, social sciences, biology, linguistics, medicine, transportation, communications, industry, economy and others. In computer science, well-known examples include the Internet, where nodes are computers and links are communication cables between them, and the web, where nodes are web pages and links are hypertext links between these pages. In social sciences, nodes are often people and links are the acquaintances between them, in biology, nodes may be proteins and links the possible interactions between them (see Figure 1), etc. From one context to another, the meaning of nodes and links is usually very

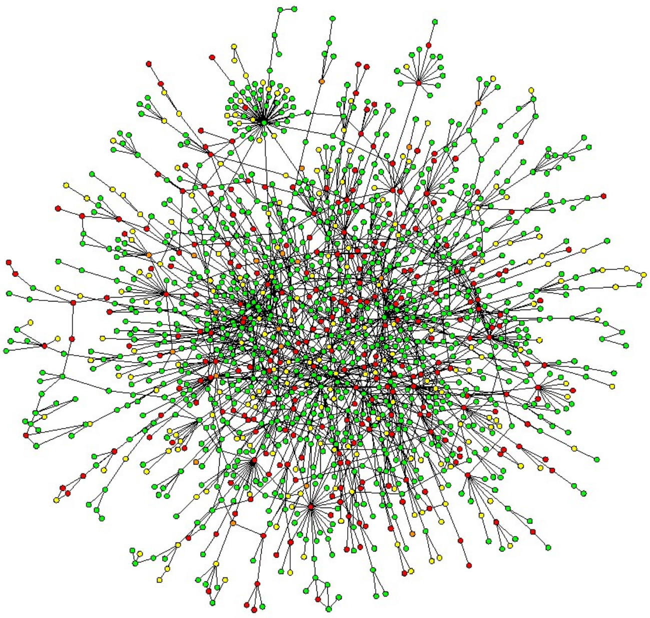


Figure 1: Example of a complex network: a protein-protein interaction network. Nodes are proteins and links are possible biochemical interactions between them [41].

different, but if we forget these meanings, the underlying object has the same mathematical nature: it is a network.

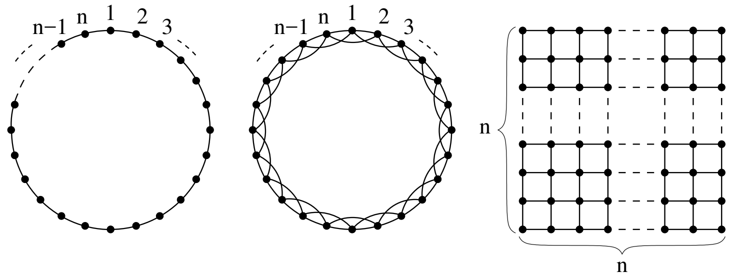
The reason why these networks are called *complex* is that it is usually very difficult to catch a comprehensive view of their organisation at the first glance. And often, even after extensive analysis, including the use of statistics, visualisation tools and other investigation means, their organisation remains only partially revealed and explained. There are mainly two reasons for this. The first one is that the size of the complex networks studied in the contexts mentioned above is usually very large, typically from thousands to billions of nodes and links, which is far beyond the cognitive capacities of humans. The second reason to call these networks complex is that the structure of their links does not appear to be regular. In other words, it seems that this structure cannot be caught by a simple rule that would allow to think of the network as a very large but clearly organised object, like a *cycle* or a *grid* for example (see Figure 2). In the absence of any such rule given *a priori*, they seem not to follow any. Their organisation appears rather anarchic and messy (see example in Figure 1). Combined with their large size, this makes these networks difficult to understand and to mentally visualise: they are said to be complex.

At the time when I am writing this manuscript, the domain of complex networks is well established in the landscape of international research. There are a large number of venues and journals specifically dedicated to complex networks or where complex networks are one of the major topics. Despite this, it is not that clear why it is a scientific domain in itself. One reason is that the domain is young, less than 20 years, and its development strongly leans on the recent digitisation of all sorts of human activities that nowadays provides a constant flow of data of huge size (known as *big data*) to analyse and exploit, a good proportion of it being organised in the form of a network. Then, one may legitimately wonder if complex networks is not simply a scientific trend rather than a long-lasting research domain.

Another reason is that the domain of complex networks is intrinsically interdisciplinary, which makes it difficult to identify. The questions that are studied about complex networks are totally dependent on the contexts where these networks come from and they are expressed using the terms and the concepts of these specific contexts. For example, in the Internet context, one asks whether the network is resilient to failure and attacks; in social sciences, one wants to understand how the connections between people affect the adoptions of new ideas and beliefs; in molecular biology, one wants to understand the role of the proteins involved in processes occurring into a cell. All these questions appear to be quite different and specific to the disciplines where they arise. Therefore, it is unclear why they should be grouped together and form a common scientific domain.

There are two main reasons that make it relevant to do so. These studies are unified by a common general problematics and common methods of investigation. Their common problematics is that they address questions which go far beyond the sole structure of the network (because they involve for example psychological issues, or chemical properties, etc.) and for which, at the same time, this organisation into a network is a crucial ingredient to answer the questions addressed, therefore requiring to interweave these different aspects in order to obtain accurate answers. Furthermore, even though the considered questions strongly depend on the context, the set of methods used to take into account the structure of the network shares a wide common basis composed of statistics, experiments, simulations, visualisation and algorithmic treatments. As a conclusion, complex networks is a scientific domain in the sense that it is united by a common type of objects, a common general problematics and common methods of investigation.

Very often, researchers of the domain justify its unity and its existence by the two following reasons. Firstly, despite the fact that complex networks come from very different contexts, their structures often share common properties (such as low density, short distances, heterogeneous degree distribution and high clustering coefficient, see technical preliminaries, page 13, for definitions) which makes it relevant to study them as a same class of objects. Secondly, even though the questions studied are very specific to the context, on the structural side, they often involves the same questions, such as community detection and the study of diffusion properties for example. These two facts are largely responsible for the identification and the emergence of the domain of complex networks. The reason why I



(a) cycle on n vertices

(b) 2nd power of the cycle on n vertices

(c) n×n grid

Figure 2: Three examples of strongly structured networks. The cycle on n vertices (a), the k^{th} power (depicted with $k = 2$) of the cycle on n vertices (b), used by Watts and Strogatz [76], and the $n \times n$ grid (c), used by Kleinberg [43].

took some time here to give my own, and different, definition of the domain is that I believe that the most exciting perspectives for its future development are precisely to go beyond the two points mentioned above. Efforts in the domain should be put on i) distinguishing different types of topology and exploiting their properties in order to provide better answers than in the generic case and ii) using and defining structural properties that are more specific, and then more relevant, to the questions considered.

A subtopic of graph theory?

Since the domain of complex networks studies networks[1], whatever may be its motivations, one could think of it as a subtopic of graph theory. There are strong reasons why, in my opinion, it

[1]Which is a synonym for graphs. I will clarify this point further in the technical preliminaries, page 13.

should not be considered as such, despite the very strong ties between the two domains (which remain insufficiently developed).

Firstly, while the purpose of graph theory is to understand networks, the questions pursued by the domain of complex networks are beyond the scope of networks themselves: e.g. how to retrieve relevant information in the web? how does a living cell works? or how does a language evolve? These are questions from other disciplines whose answers do not involve only network concepts. Moreover, the part concerning networks cannot always be isolated and networks often have to be augmented with an amount of external information to encompass the other aspects of the question. This either gives objects derived from graphs that are considered denatured by graph theorists, or this gives questions that are so constrained by these other aspects that they loose their interest for graph theorists.

Secondly, rather surprisingly, complex networks do not correspond to any kind of graphs studied in graph theory. Graph theory mainly studies two cases. The first one is the case of classes of graphs satisfying a given mathematical property, for example graphs excluding a certain configuration, or graphs admitting a special representation or again graphs having a certain parameter bounded by a constant. The second case of study is random graphs. In this case, one studies the expected properties of a graph chosen uniformly among a usually very broad[2] class of graphs, such as graphs on n vertices or graphs on n vertices and m edges for example, for fixed n and m.

It turns out that complex networks precisely lay in the gap between these two cases of study, and it could stand for a definition of them. They do not satisfy any strict mathematical property and they are very different from random graphs (on the same number of vertices and edges). This is the core idea of the paper of Watts and Strogatz [76] which is often consider as the

[2]When the class considered is more restricted, *i.e.* satisfies some mathematical property, one comes back to the previous case.

founding act[3] of the domain. [76] shows that complex networks cannot be considered as random graphs and suggests that they could instead be seen as perturbations of strongly structured graphs. In other words, they are nearly structured graphs and this "nearly" is the reason why they have been mainly out of scope of graph theory so far. This also explains why until now these objects are usually treated using empirical approaches or techniques from other disciplines, in particular from physics.

Therefore, despite the fact that the study of complex networks inevitably relies on graphs, it considers problems that are external to graph theory, on a kind of objects that is not considered classically in graph theory and with methods that are very different from those used in graph theory.

Relationship with computer science and other disciplines

By nature, the domain of complex networks is interdisciplinary. It involves all disciplines where these objects appear together with questions they raise and it also involves disciplines that provide methods to address these questions. There are three disciplines that, from my point of view, have deeper and special involvement in the domain of complex networks.

Among the disciplines interested in complex networks because of their need to treat such objects in their field of study, social sciences is probably the one who contributes more to the domain. This comes from the fact that this is one of the context where the concept of the link (the *tie* as it is called in this domain) has the deeper meaning and importance. This led social sciences to study many social systems organised in the form of a network (which also constitute important case studies for other disciplines interested in methodological aspects) and to develop

[3]This is the beginning of this trend of study and according to Google, the paper has been cited by nearly 30 000 other documents since it was released in 1998.

approaches that deal with the structure of these networks. It is striking to see that pure concepts of graphs, such as the small world phenomenon [73] and the distinction between strong and weak ties [35], that one would expect to have been introduced in the field of complex networks by graph theorists during the recent rise of the domain were actually introduced in social sciences almost fifty years ago.

There are some other disciplines that contribute on the methods but not necessarily generate objects. This is the case of physics for example, which offers only few objects of study but which have a strong interest in the domain for the methodological aspects. In particular, one central question in complex networks is the relationships between the local structure of interactions in the network, at the microscopic scale, and its global organisation and behaviour, at the macroscopic scale. This question has been pursued by statistical physics from centuries and largely explains the interest and the achievements of physicists in the domain. Interestingly, this question is also a point of convergence with computer science where the local vs global approaches are transverse in many topics such as graph theory, distributed algorithms and multiagent systems.

Finally, computer science is also one of the main actor in the field of complex networks. However, it could be expected at an even higher level of contribution taking into account that the domain of computer science has all the reasons to be involved in the study of complex networks. Firstly, the domain generates an important number of objects of huge size organised in the form of a network, having a technological nature or related to information and communication. Let us cite for example computer networks such as the Internet, mobile and ad-hoc networks, smartphone networks, networks of sensing and intelligent devices (at home, in buildings, in cities), the web, blogs, emails, online social networks, peer to peer networks, and others. All these contexts raises questions about these networks. Secondly, graphs are an important object of study in theoretical computer science, in particular for the algorithmic aspects. Consequently, computer science actively participates in the development of graph theory and has a natural interest for these objects. Finally, complex networks constitute huge amounts of data that need to be represented, stored and treated, algorithmically and computationally. These aspects lay in the scope of computer science and are another reason of the natural involvement of the discipline in the domain of complex networks.

My personal approach

One important characteristic of this thesis is that it uses both empirical methods (mainly in Chapters 1 and 2), *i.e.* experimentation and simulation, and formal methods (mainly in Chapters 3 and 4), *i.e.* proofs. The reason is that the aim of my research work is to provide practical solutions to concrete problems, as measuring the topology of the Internet or limiting the spread of nosocomial infections in hospitals. In this case, all means are valuable as soon as they provide a better understanding of the problem. To this regard, formal and empirical approaches are complementary.

From a certain point of view, empirical methods are more powerful, in the sense that they often provide a simpler way to investigate dependencies between properties, which may be out of reach by formal approaches in some cases. This greater simplicity has a price: empirical methods do not provide indubitable conclusions. Concretely, it means that there is a possibility that the conclusions derived are induced by the way they are derived. This cannot happen with proofs, in theory, because their validity, based on logic principles, is universal and can be checked. Nevertheless, if the experimental protocol or the simulation protocol is designed carefully enough, the results obtained from it are usually quite informative, and gives a very valuable insight into the problem. Because of their nature, complex networks, which do not satisfy any strict mathematical property, are prone to be treated by empirical approaches, which easily accommodate with errors, uncertainty and randomness that are contained inside these objects.

Nevertheless, formal methods are also very useful to the field. They can be used for example to ensure that the conclusions derived are not particular to the networks on which they are obtained or induced by an undesired particularity of the experimental protocol (which, for practical reasons, cannot always be as general as one would wish). In other cases, certainty is required, for example to design and to prove the correctness of the computational methods used in empirical approaches, and formal methods are then needed as well.

But beyond their specific contributions and interests, both approaches, empirical and formal, rely on a common basis: on concepts, properties and relations between them. This is why, they are not to be opposed but are instead strongly complementary. Empirical approaches must nourish themselves from the rich set of concepts developed in formal contexts and formal approaches must benefit from the relations pointed out by empirical means. This is how I found very useful and fascinating to work with both at the same time. This thesis is written with the intention to provide examples of how this mutual enrichment can operate.

Outline of the thesis

This thesis contains four chapters. Each of them includes a partial conclusion that outlines the direct perspectives opened by the works presented in the chapter. In the second and third chapter, these perspectives are not postponed to the end of the chapter but instead detailed at the end of each section. In addition, the manuscript also contains a general conclusion which discusses two directions of research in which I particularly believe for the domain of complex networks, and that I will pursue in the next years.

The first chapter deals with the measurement of the degree distribution of the Internet topology. Our approach is based on a new principle called *property oriented measurement*, which provides more faithful information by focusing only on one target property, without collecting a map of the network. Two methods are presented, one for measuring the degree distribution of the logical topology and one for measuring the degree distribution of the physical topology.

Chapter 2 presents three works on the dynamics of complex networks. The first one is a case study on the dynamic network of contacts within a hospital and the other two are methodological developments dedicated to dynamic networks in general. One is about characterising the structure of changes of the topology of a dynamic network, the other one addresses the problem of finding appropriate time scales in order to aggregate a dynamic network into a series of graphs.

Chapter 3 deals with complex network modelling. The aim is to design random generation processes that output synthetic networks having properties as close as possible from those observed for real-world networks. We describe two different approaches toward this goal. One is based on the entanglement structure of maximal cliques, while the other one is based on the approximation of complex networks by strongly structured graphs.

Finally, the last chapter considers the problem of designing very compact encodings of graphs that do not penalise the queries made during algorithmic treatments, such as listing the neighbours of one given vertex. We investigate the efficiency of two related encodings, named *contiguity* and *linearity*, which use linear orderings of the vertices of the graph.

Technical preliminaries

We denote $|X|$ the cardinal of a finite set X, *i.e.* its number of elements. When a set X, not necessarily finite, is totally ordered by $\leq$, we denote $[a, b]$ the interval of X comprised between a and b, where $a, b \in X$ and $a \leq b$, that is $[a, b] = \{x \in X \mid a \leq x \leq b\}$. If X is a discrete set, typically integers, we denote $[\![a, b]\!]$ instead of $[a, b]$.

In the domain of complex networks, *network* and *graph* are synonyms. In this manuscript we use both terms. We use the term network in the context of works using mainly empirical methods (Chapters 1 and 2) and for real-world objects, while we use the term graph in the context of works using mainly formal methods (Chapters 3 and 4) and for the definitions given in these technical preliminaries.

A *graph* G (or *network*) is a couple denoted $G = (V, E)$, where V is the (finite) set of vertices (called *nodes* with the vocabulary of networks) and E is a subset of pairs of vertices which are called *edges* (or *links* with the vocabulary of networks). The set of vertices of a graph G is also denoted $V(G)$ and its set of edges $E(G)$. In the following, an edge $\{x, y\} \in E$ is simply denoted xy or yx, indifferently. In all the manuscript, the number of vertices is denoted $n = |V|$ and the number of edges is denoted $m = |E|$. The *density* of a graph (also called *global density*), denoted ρ, is the number of edges divided by the number of possible edges (*i.e.* the number of couples of vertices), that is $\rho = \frac{m}{\frac{n(n-1)}{2}} = \frac{2m}{n(n-1)}$. For $X \subseteq V$ a subset of vertices, the *graph induced* by X, denoted $G[X]$, is defined as $G[X] = (X, \{xy \in E | \ x \in X \text{ and } y \in X\})$.

Two vertices x and y of a graph G are *adjacent* if there is an edge between them. In this case, we also say that x is a *neighbour* of y, and reciprocally that y is a neighbour of x. The set of neighbours of x is called the *neighbourhood* of x and is denoted $N(x)$. The degree of x, denoted $d(x)$ (or $d^\circ(x)$) , is its number of neighbours, *i.e.* $d(x) = |N(x)|$. The *mean degree* $\bar{d}$ of a graph is the arithmetic mean of the degrees of all its vertices, that is $\bar{d} = \frac{\sum\limits_{x \in V} d(x)}{n}$. Note that it is related to the density of the graph by $\rho = \frac{\bar{d}}{n-1}$, as for any graph $\sum\limits_{x \in V} d(x) = 2m$. The *closed neighbourhood* of x is denoted $N[x]$ and defined as $N[x] = N(x) \cup \{x\}$. For sake of clarity, when we use the notion of closed neighbourhood, we usually use the term *open neighbourhood* instead of just neighbourhood.

The *degree distribution* of a graph G (and more generally the distribution of a function f) is the function $k \mapsto P(k)$, where $P(k)$ is the proportion of vertices having degree exactly k (resp. the proportion of elements x such that $f(x) = k$). One classically distinguishes two families of distributions. The first one is *homogeneous distributions*, which are centred around their mean and where there is consequently a notion of normal value. One typical example of homogeneous distributions is the *Poisson distribution*, $P(k) = \frac{\lambda^k}{k!} e^{-\lambda}$, where λ is a parameter, namely the mean of the distribution. Opposed to homogeneous distributions are *heterogeneous distributions*, where there is no notion of normal value as the mean is not representative. Instead, heterogeneous distributions span a wide range of values spread on several order of magnitudes. They are said to be *heavy tailed*, meaning that the value of $P(k)$ slowly decreases when k goes through very large values. In most cases, slowly means polynomially, as opposed to homogeneous distributions that exhibit an exponential decay. The typical example for heterogeneous distributions is the *power-law* distribution, where $P(k)$ is proportional to $k^{-\alpha}$, with $\alpha > 0$ being one parameter, called the exponent of the power law.

A *path* in a graph is a sequence of vertices $P = a_0, a_1, \ldots, a_k$, where $k \geq 0$, such that for all $i \in [\![0, k-1]\!]$, a_i and a_{i+1} are adjacent. When $a_0 = a_k$, P is called a *cycle.* The length of a path $P = a_0, a_1, \ldots, a_k$ is k. The distance $\delta(x, y)$ between two vertices x, y is the minimum length of a path between these two vertices, *i.e.* $a_0 = x$ and $a_k = y$, if such a path exists and the distance is infinite otherwise, by convention.

A subset $X \subseteq V$ of vertices is *connected* if and only if for all vertices $a, b \in X$, there exists a path from a to b. If V itself is connected, we say that the graph G is connected. The *connected components* of a graph are its connected subsets of vertices that are maximal for inclusion. The *diameter* of a graph is the maximum of the distance between all couples of vertices, it is infinite if G is not connected. The *mean distance* $\overline{\delta}$ of a connected graph is the arithmetic mean of the distances between all couples of vertices, that is $\overline{\delta} = \frac{\sum\limits_{x,y \in V} \delta(x,y)}{\frac{n(n-1)}{2}}$.

Distinct from the notion of global density, there is a notion of *local density* of a graph which aims at quantifying how dense is the graph in the neighbourhoods of its vertices. Usually, the local density is appreciated by the *clustering coefficient* of the graph. There are two definitions of clustering coefficient, which we denote $CC_1(G)$ and $CC_2(G)$. The most immediate one, $CC_1(G)$, is based on the local clustering coefficient of a vertex x, which is denoted $cc_1(x)$. $cc_1(x)$ is the density of the graph induced by the neighbourhood of x, that is $cc_1(x) = \rho(G[N(x)]) = \frac{|E(G[N(x)])|}{|V(G[N(x)])|(|V(G[N(x)])|-1)/2}$. $CC_1(G)$

is then defined[4] as the mean of $cc_1(x)$ for all vertices $x \in V$, that is $CC_1(G) = \frac{\sum\limits_{x \in V} cc_1(x)}{n}$. The second definition of the clustering coefficient, $CC_2(G)$, does not go through local quantities, it is directly expressed as $CC_2(G) = \frac{|Triangles|}{|Pretriangles|}$, where $Pretriangles = \{(a, b, c) \in V^3 \mid ab \in E \text{ and } ac \in E\}$ and $Triangles = \{(a, b, c) \in V^3 \mid ab \in E \text{ and } ac \in E \text{ and } bc \in E\}$. The two clustering coefficients $CC_1(G)$ and $CC_2(G)$ are incomparable in the sense that there does not exist any constant $K > 0$ such that for all graphs G, $CC_1(G) \leq K \,.\, CC_2(G)$ or for all graphs G, $CC_2(G) \leq K \,.\, CC_1(G)$.

Trees are a particular kind of graphs that play a very special role in graph theory. In this manuscript, we often use *rooted trees*. A rooted tree is a graph $G = (V, E)$ which can be defined in the following way. For each vertex $x \in V$, except one vertex $r \in V$ called the *root*, choose a vertex distinct from x to be the *parent* of x, denoted $parent(x)$, in such a way that the edge set $E = \{\{x, parent(x)\} \mid x \in V\}$ makes V connected.

A *partial order* on a set X is a binary relation, denoted $\leq$, that is reflexive ($\forall x \in X, x \leq x$), antisymetric ($\forall x, y \in X$, if $x \leq y$ and $y \leq x$ then $x = y$) and transitive ($\forall x, y, z \in X$, if $x \leq y$ and $y \leq z$ then $x \leq z$). A *total order*, or *linear ordering*, on a set X is a partial order $\leq$ such that all the elements of X are comparable. Formally, this means that $\forall x, y \in X, x \leq y$ or $y \leq x$. In a linear ordering on a finite set X, there exists one unique element m such that $\forall x \in X, m \leq x$ and there exists one unique element M such that $\forall x \in X, x \leq M$. The element m is called the *minimum* of $\leq$ and M its *maximum*.

[4]Note that this definition is actually subject to slight variations as well, depending on how the density is defined in the special case where the neighbourhood of x contains 0 or 1 vertex, and depending on whether such vertices x are taken into account in the mean or not.

Some classes of graphs

Throughout the manuscript, we sometimes present or use algorithmic results and representation results for some graph classes, such as chordal graphs, interval graphs, permutation graphs and cographs for example. We therefore give a very brief introduction to these classes. For more, the reader may refer to [11].

Chordal graphs are the graphs that do not contain any induced cycle of length at least 4. One of their most useful characterisations is that their maximal cliques can be arranged into a tree T such that for each vertex x, the set of maximal cliques containing x forms a (connected) subtree of T. Chordal graphs are known to admit very efficient solutions to algorithmic problems that are hard for graphs in general.

Interval graphs is one of their subclasses. Their most immediate definition is that they are the intersection graphs of intervals of the real line. In other words, a graph G is an interval graph if and only if it admits an *interval realiser*, *i.e.* its vertices can be mapped to a set of intervals of the real line such that two vertices x and y of G are adjacent if and only if their corresponding intervals intersect. Even stronger than for chordal graphs, their maximal cliques can be arranged into a path having the property that the maximal cliques containing any vertex x are consecutive on the path. Such paths are called *consecutive arrangements* of the maximal cliques and they are the canonical interval realisers of the graph. It is remarkable that, while there may be many of them, up to $\Omega(n!)$, the set of all canonical realisers can be represented very efficiently, thanks to a tree called the *PQ-tree*, in space $O(n)$, which is also the space needed to represent one single of these realisers.

Permutation graphs are the intersection graphs of segments whose endpoints lay on two parallel lines. They are represented by two linear orderings of their vertices: the orders of occurrence of the endpoints of the segments on the two parallel lines. As for interval graphs, such a representation is called a realiser and all realisers of a permutation graph can also be stored in space $O(n)$ thanks to a tree called the *modular decomposition tree*.

In the following, we are particularly interested in a subclass of permutation graphs called *cographs*. Cographs are the graphs that have no induced path on 4 vertices. They are also the graphs that can be obtained from single vertices using the disjoint union and the complete union of graphs (disjoint union plus all possible edges between the graphs involved). The minimal tree encoding the operations used to build a cograph is an $O(n)$-space representation of it and it is also its modular decomposition tree.

Chapter 1

Property oriented measurements: the case of the degree distribution of the Internet

Context

In 1999, Faloutsos et al. [32] showed, using a data set collected in 1995 by Pansiot and Grad [62], that the degree distribution of routers in the Internet approximately follows a power law. This means that the proportion $P(k)$ of routers of degree k is proportional to $k^{-\alpha}$ for some $\alpha > 0$, which we write $P(k) \sim k^{-\alpha}$. Since then, huge campaigns of measurement of the Internet topology have been led and this observation was confirmed many times independently. This fact is crucial as it is known that the degree distribution of routers of the network has a key impact on many aspects of the management of the Internet, in particular on the resilience of the network to failures and attacks, on the speed of spreading phenomenons (e.g. viruses) and the ability to contain them and on the quality of services on the network (e.g. for routing or broadcast).

Beyond the sole Internet topology, this result participated to a wide stream of works exhibiting the scale-free property of many complex networks from various contexts, meaning that the degree distribution of these networks is *heavy tailed*, like a power law. In this trend of work, the power law is practically erected as a universal law of nature, as the Gaussian law of errors was before, that appears almost everywhere and not only in the degree distribution of complex networks. In this latter context, the appearance of the power law was partially explained by an argument concerning the formation and the evolution of networks, known as preferential attachment [5].

The impact these observations had on the scientific community also brought a counter-stream of works that objectively criticised the way some of these power laws had been observed, arguing that they were sometimes introduced by the way the distribution had been evaluated [13]. In the case of the degree distribution of the Internet, this stream turned into a big scientific controversy. The data coming from measurements, on which was appreciated the degree distribution, was shown to be biased [45]. This bias was even explained both empirically [45] and formally [1], showing that, with the methods of measurement employed, it is possible to observe power-law look-alike distributions even when the real distribution is homogeneous, which is the diametrically opposed case. In addition to these theoretical arguments, there were voices from practitioners and engineers that design the network that rised to argue that anyway, the real degree distribution of the network is certainly not heavy tailed since, for practical and technological constraints, a router can have only a very limited number of neighbours. Nevertheless, it remains that all the measurements of the Internet degree distribution made until now find it to be heavy tailed. In the light of the controversy above, it points out that there is no trustful estimation of this distribution available so far.

This question on the accuracy of the observation of properties of complex networks is not particular to the Internet but is general to all networks. Indeed, as complex networks come from a real-world context, they are always known through an operation of measure, specific to this context, e.g. by downloading webpages to measure the web graph, or by questioning people to measure their social ties or by performing chemical reactions to measure interactions between proteins. Since the knowledge we have of these networks comes from a measurement operation, the question arises to know whether the result of the observation is consistent to the real network or whether it is shaped by the measurement method itself, in which case we say the observation is *biased*. This is the question addressed by *metrology*. This general question calls several sub-questions: can we detect the bias in the result of a measurement? can we quantify it? can we correct it? and finally, can we design measurement methods that are free of bias? One of the beauty of metrology is that it is sometimes possible, as in the case of the Internet, to detect the bias introduced by the measurement even without knowing how the original network looks like. On the other hand, correcting it is in general difficult as the bias introduced is usually not reversible: many different original networks may be observed in the same wrong way.

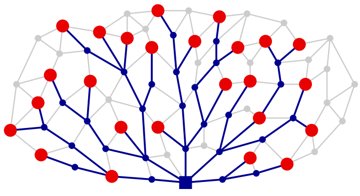
Approach

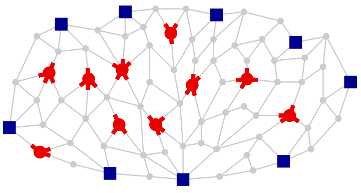
The question we address here is to design a measurement method that is free of bias. Of course, the possibility of successfully implementing such a method entirely relies on the measurement primitives available in the context of the measure. In the case of the Internet, we show how these primitives can be used in order to obtain non biased information on the network. Nevertheless, it is worth noting that the idea underlying our approach is not particular to the case of the Internet topology and may be potentially applied in any context. The idea is to avoid to entirely map the network and instead design a measurement method which is only dedicated to one property of interest, here the degree distribution. By operating such a restriction, it is possible to obtain a more faithful information than the one obtained when trying to get the complete knowledge of the topology of the network. This idea is similar to a well known principle in physics[1], called the Heisenberg's uncertainty principle, which states that some pairs of physical properties cannot be known simultaneously with an infinite precision. This sets an absolute limit on the possibility of knowing accurately all the properties of one system simultaneously. But this limit does not stand if one restricts oneself to one specific property. Our work shows that this idea is relevant to network measurements and can improve the quality of the information obtained.

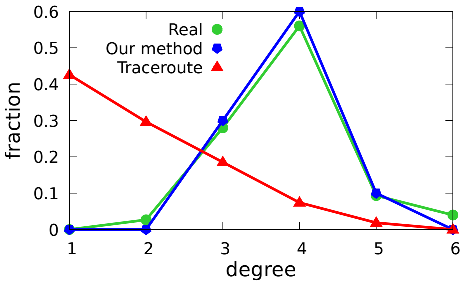
The classical method used for estimating the degree distribution of the Internet is based on `traceroute`. This software provides a path in the network between one *source*, where is run the software, and one *destination*, which can be any machine of the Internet. The tendency until now was to perform extensive `traceroute` measurement campaigns. One launches `traceroute` from a usually restricted set of sources toward as many destinations as possible[2] and one collects the obtained paths. Then, these paths are merged together into one single network, which is the result of the measurement and on which all properties are estimated, including the degree distribution (see

[1]It also carries an idea similar to the one of the popular wisdom stating that "you can't have your cake and eat it too".

[2]There is no real limitation on the number of destinations one can use. One can probe any IP address, *i.e.* any 32-bit integer. On the opposite, having many sources is a real challenge as one must possess a user account on them in order to launch the `traceroute` program.


(a) classical method


(b) our method



(c) observed distributions

Figure 1.1: Comparison of our method to the classical `traceroute` method. (a): a `traceroute` measurement from 1 monitor (square node) toward 25 targets (bullet nodes). This measurement needs 97 probes. (b): measurement with our method from 9 monitors (square nodes) toward 10 targets (bullet nodes). See Figure 1.2 for the detail of how the links of each target are discovered. This measurement needs 90 probes. (c): the true degree distribution of the network together with the estimates obtained by both methods.

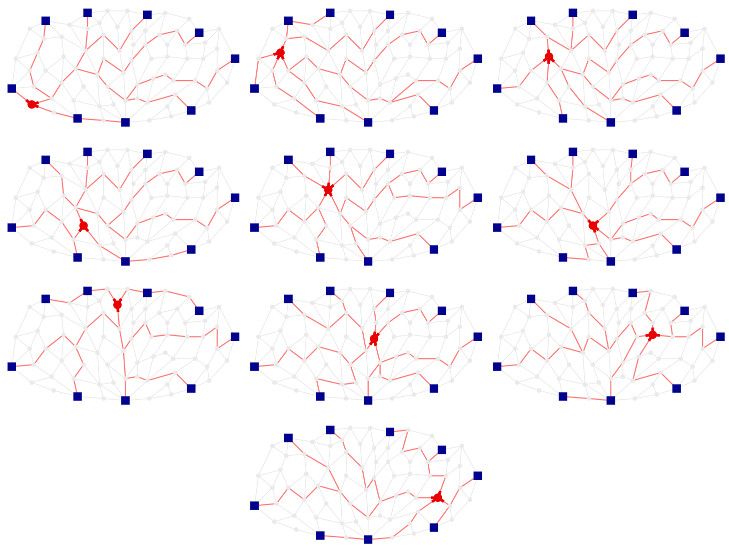


Figure 1.2: Measurement of the degree of 10 targets using our method. We display 10 copies of the network, one for each target measurement. On each copy we show the routes followed by the probes sent from the 9 monitors (square nodes) toward the corresponding target (bullet node).

example in Figures 1.1.a and 1.1.c). The idea underlying this approach is that if the measurement is extensive enough then the network obtained from it faithfully represents the original network. This is precisely the idea that is contradicted in [45, 1] where it is shown that this method is intrinsically biased and that conducting wider measurements does not necessarily lead to more accurate observations.

The measurement method we design is drastically different. We do not resort to any map of the network. Instead, we randomly select one target router and directly measure its degree by sending probes to this router from monitors scattered all over the Internet (see example in Figures 1.1.b and 1.1.c), each probe discovering one neighbour of the target (see Figure 1.2).

Another originality of our approach is to take into account some particularity of the topology of the Internet in the design of our measurement method, as this particularity has a strong influence on the information obtained from the measurement primitive. Our method is specifically intended to measure the degree distribution of the *core* of the network, which is the key part of its topology. Formally, the *core* (2-core with the vocabulary of graph theory) of the network is the part that remains after one iteratively removes degree-one nodes until there are none of them left. More intuitively, the set of nodes removed during this iterative process, which is called the *border*, form trees that are attached to the network and the core is the rest of the nodes (see Figure 1.3.a).

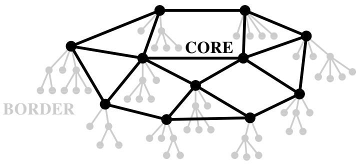
The behaviour of our distributed probing method is very different depending on whether the target is in the border or in the core of the network. If the target is in the border (see Figure 1.3.c), then our method is likely to miss most of its neighbours, which are deeper in the border, and to discover only the neighbour of the target which is its parent in its border tree. When the target is in the core, the situation is quite different: if our set of monitors is distributed well enough, it is likely to discover all or most of the neighbours of the target that are in the core, and then to correctly estimate the core degree of the target, which is our goal.

Therefore, our method relies on two key points:

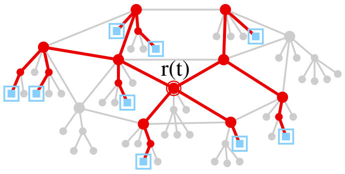
1. uniformly randomly select one router in the core, and

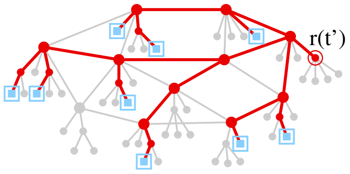
2. rigorously measure its degree in the core.

Doing so for many randomly selected targets in the core of the Internet gives an estimate of its degree distribution. Unlike in



(a) core and border





(b) a target in the core **(c)** a target in the border

Figure 1.3: (a): the core and the border of the network; the border is the set of all trees connected to the network, the core is the part remaining when one removes these trees. (b): a set of monitors (the squared nodes) send probes toward a target core router $r(t)$ and obtain its four neighbours in the core (or, equivalently, its four core interfaces). (c): the same monitors send probes toward another target router $r(t')$ belonging to the border and miss most of its neighbours (or, equivalently, most of its interfaces).

the classical `traceroute` method, if we are able to achieve the two tasks above, the estimated degree distribution is guaranteed to converge to the actual one when the random sample of core routers grows. It must be clear that both of these two tasks constitute challenges in themselves. In the next section, we show by simulation that our distributed probing approach satisfyingly fulfils the second point. The question of uniformly randomly selecting routers in the core will be addressed later.

Proof of concept [27]

In [27], we conduct an extensive set of simulations. Its goal is to determine whether our primitive for the measurement of the degree of one router in the core, which consists in sending probes from a set of monitors scattered over the Internet toward this router, is able to correctly estimate its degree. The main questions we want to answer are: given a number of monitors, what is the risk that we do not correctly estimate the degree of one node of degree k? how many monitors are necessary in order to correctly estimate the degree of most of the nodes of the network?

To answer these questions we randomly generate two different kinds of synthetic topologies: networks with a Poisson degree distribution, parameterised by their number of nodes n and their mean degree $\bar{d}$, representative of the homogeneous distributions, and networks with a power-law degree distribution, parameterised by their number of nodes n and the exponent α of their power-law, representative of the heterogeneous distributions. For both kind of topology, we make their parameters vary in order to appreciate the generality of the conclusions derived with one setting.

For the set of targets, we use all the nodes in the core of the network. For monitors, as we want to appreciate the influence of their number, we make it vary from one dozen to some hundreds, which corresponds to the range of values we can expect in the practical settings of our measurements. As one can guess, the quality of observation does not only depend on the number of monitors but also on their location. Since for simulation purposes there is no clear choice of where should be located the monitors, we choose to locate them uniformly at random among the nodes of the network. As this choice is certainly favourable to the performance our method, in [48], we design metrics to appreciate how well distributed is the set of monitors we use in practice in our measurements. These metrics indicate that it is indeed rather fairly distributed over the network. Consequently, the settings of our simulations, where monitors are chosen uniformly randomly among the nodes of the network, are relevant.

Finally, we also need to model the path of the network taken by the probe between the monitor and the target. As in most works, we choose to model them by a shorter path from the monitor to the target, randomly breaking the ties[3] between paths of equal length.

Our simulations show that our method is able to correctly estimate the degrees of the nodes in the core of the network. For example, in the case of Poisson topology of mean degree 25, using 100 monitors, this estimation is almost perfect for almost all core nodes. For power-law topology of exponent 2.1, using 200 monitors, the quality of the observation is excellent for core nodes of reasonably low degree, let say at most 10. This shows that in this theoretical setting, the only limitation of our method seems to be the observation of high-degree nodes in power-law networks. One important second conclusion is that even for those nodes, the observed degree is in the same order of magnitude than the actual degree. For example, nodes of degree 60 in the power-law topology are observed with an average degree of 37. This implies that despite the fact that our method may not be able to correctly estimate the degree of high-degree nodes, it will succeed in detecting their presence, because a high-degree node in the core is observed with a similarly high degree. Finally, one important conclusion of [27], is that the quality of observation does not depend on the size of the network, but only on the kind of topology (homogeneous or heterogeneous) and on the ratio between the number of monitors and the degree of the observed node. Therefore, the results of these simulations, made on networks of some millions of nodes, should still hold for networks of the size of the Internet.

[3]In [27], we investigate three different ways of randomly choosing one shortest path. Our results shows that the choice of one of these three ways has only a very little impact, if any, on the degree estimated for each node.

Measurement of the IP level topology [21]

In the controversy about the degree distribution of the Internet topology, a pretty large amount of confusion has been added by the fact that there are two different topologies, physical and logical. One usually has in mind the physical one, made of the cable connections between routers of the Internet. However, the topology measured by the classical `traceroute` approach is the logical one. In this topology, two routers are linked if they are separated by only one IP hop, which means that they can send IP packets to each other directly without packets being handled by any other router. For many links, there is no difference between the two topologies. If two routers are linked by a cable, they are also separated by one IP hop. But the converse is not true, two routers may be at one IP hop from each other even if they are not connected by a cable, as there exist lower level intermediate devices that connect routers of the Internet but that are invisible at the IP level, such as *switches* for example. These intermediate (non-router) devices may be connected to many routers, making all of them potentially appear at one IP hop from each other, giving rise to a clique in the IP level topology. For this reason, nodes of high degrees are much more plausible in the IP level logical topology than in the physical one.

Despite these differences, the two topologies coincides on many parts and measuring the IP level topology has often been considered as a proxy for measuring the physical topology. Another reason why measurements are usually conducted at IP level is that most measurement primitives, including `traceroute`, give information at IP level. Here, for sake of clarification, we make a clear distinction between these two topologies and we design one measurement method for each of them.

Another great benefit of doing so is that the technical challenges raised in both cases are different and require different solutions. Until now, we considered only two kinds of limitations applying to Internet measurements: i) the network is too large and ii) we cannot use every machine as a monitor. As a result, we can obtain only a partial view of the topology, which motivated the paradigmatic shift we operate in the way we measure the network. Actually, there are some other very practical limitations that affect the result of the measurement. The two more important are: 1) some routers do not answer to probes and their information is therefore not available and 2) paths output by `traceroute` contain false links, *i.e.* links that do not exist. These undesirable effects have been quantified by the networking community. They are rather frequent and have non-negligible consequences on the accuracy of measurements. It is remarkable that, beside the fact it provides theoretically sound foundations for the measurement method, the property oriented approach also allows to reduce and almost eliminate the consequences of the practical limitations of the measurement primitives.

The first limitation mentioned above is the result of the security policy applied by some network administrators and one must accept that this information is not available. Consequently, for our measurements, these nodes simply do not exist. Nevertheless, the impact of the absence of this information is entirely dependent on the measurement method adopted. In the conclusion of this chapter, we explain why this impact is limited in the case of our property oriented measurements, both for the logical topology and the physical one.

In the measurement method we design for the IP level topology [21], we address the problem of false links. This constitutes another valuable improvement on the classical `traceroute` method, where the effects of this erroneous information are often ignored. Many false links in `traceroute` outputs come from the conjunction of two facts: i) each router on the output path is discovered by a different and independent probe which follows one path from the source to the destination and ii) for sake of the repartition of the charge over the network (called *load balancing*), there exist in general several different paths from the source to the destination. Because of the property oriented approach we use, we are interested only in the router immediately preceding the destination on the path output by `traceroute`, which is the only one contributing to the degree of the destination. It turns out that, when all the paths from the source to the destination have the same length, even if there are many of them, the router preceding the destination is always a real neighbour of it. Consequently, to avoid the presence of false neighbours of the routers targeted by our measurement, we make 10 repetitions of `traceroute` for each couple m, t of monitor and target. If t is not always discovered at the same distance from m in all the 10 repetitions, then it indicates that there exist routes of different lengths between m and t. Consequently, the output of `traceroute` is likely to contains false neighbours for the target, so we simply discard the information obtained from monitor m to observe target t (however, m may still be used to observe other targets). On the opposite, if all the 10 repetitions of `traceroute` from m to t have discovered t at the same distance, then we assume that there is no route that would discover t at another distance (the probability that it is not true is sufficiently low) and that consequently, all the neighbours of t observed by m are indeed real neighbours of t in the topology. We then keep the information provided by m in the observation of t.

We implemented our method on a distributed set of monitors and conducted a measurement. We built a list of targets by probing 250 000 IP addresses chosen uniformly at random and keeping the first 10 000 that answered to our `ICMP ECHO REQUEST` message (the same kind of message used by `traceroute`). For the set of distributed monitors, we used the nodes of PlanetLab. This is a consortium that provided at that time (July 2009) a set of 952 machines made available to researchers by 483 institutions (mainly research labs) widely distributed in the world. The measurement protocol is as follows. On each monitor, we upload the list of 10 000 target IP addresses and the monitor shuffles the list (for balancing in time the load induced on each target). Then, every second, it sends one traceroute to the next target in the list, and repeat the list 10 times. The measurement took 30 hours. We filtered the obtained data by keeping for each target the monitors that observed it only with routes of the same length, and we discarded targets that were not observed by sufficiently many monitors. We thereby obtain a set of 6101 targets that have been observed by at least 350 monitors with routes of the same length.

This measurement is only intended to demonstrate the practical efficiency of our approach. It does not directly provide an estimation of the degree distribution. To this purpose, one should still address two important questions. Firstly, we did not select routers uniformly at random but we instead selected addresses uniformly at random, which is quite different as routers do not all have the same number of addresses. Secondly, the list of neighbours we obtain for one target is a list of IP addresses. Some of these IP addresses actually belong to the same router. In order to get the degree of the target in the IP topology, we should identify such subsets of addresses and count them as one neighbour. Doing so, is a classical problem in networking, known as *anti-aliasing*, for which relatively satisfying solutions exist, but we did not implement one of them. We address these two questions rigorously for the physical topology, in the next section.

Nevertheless, this measurement still provides one valuable conclusion. Among the 6 101 target addresses that we measured, the maximum number of neighbouring IP addresses we observed is 57. For the reasons explained above, the actual number of routers designated by these 57 addresses is even expected to be less than 57. Since the targets have been selected at random, it indicates that the probability for a router in the Internet to have more than 57 neighbours in the IP level topology is quite low, in the order of magnitude of 10^{-3}. Then, if such routers exist, there are only very few of them.

Measurement of the physical topology [47, 48, 67]

The method presented in this section is dedicated to the measurement of the degree distribution of the physical topology [47, 48]. The general principle of the approach is the same as for the IP level topology, but the technical implementation is different. Here, we want to measure the number of interfaces of one router that are connected, by cables, to other devices. To this purpose, we do not use `traceroute` but a measurement tool we designed, which we call `udp-ping`. Unlike `traceroute`, the monitor m running `udp-ping` sends only one probe to the target t, designated by one IP address. This probe is a UDP packet sent to an unallocated port. On receiving such a packet, the target answers with an ICMP error message `Destination unreachable`, with an error code meaning `Port unreachable`. The key point that we exploit is that if the target correctly implements the recommendations of the RFC 1466 of IETF[4], the source address of the ICMP error message sent by the target t is the address of the interface it actually uses to send the message to m. Then, by sending `udp-ping` probes from a set M of distributed monitors toward a target t, we discover the set of interfaces that t uses to send messages to the monitors in M. Therefore, if t is a core router and if the monitors of M are well distributed enough over the Internet, we expect to discover all the core interfaces of t.

Note that in this case, unlike in the case of the IP level topol-

[4]The Requests For Comments (RFC) are a series of official documents, published by an open international organisation named the Internet Engineering Task Force (IETF), which contain standards and recommendations on the technical aspects of the design, the use and the management of the Internet.

ogy, there is no problem of anti-aliasing: all the interfaces discovered by `udp-ping` belong to the target t. The only question that remains to be addressed is to uniformly randomly select one router (not one IP address) in the core of the Internet. We now give a rigorous way to do so. It relies on the possibility to distinguish between core and border routers and between interfaces that are directed toward the core of the network and those that are directed toward its border, which we simply call *core interfaces* and *border interfaces* respectively. To decide whether an interface observed in our measurement is directed toward the core or toward the border of the Internet, we use an auxiliary measurement. In this measurement, which is very light, from each monitor m in our set, we probe randomly chosen addresses over the Internet and we determine (in a way which is not detailed here) what is the set of border interfaces that are visible from m. In this way, for each interface discovered in our main measurement, we can decide whether it is a border interface by checking whether it appears in the list of border interfaces visible from some monitor m in our set. The other interfaces are necessarily core interfaces. Once we are able to distinguish between core interfaces and border interfaces, we can do the same for routers themselves: border routers are those that have exactly one core interface, the interface linked to their parent in their border tree. All core routers have by definition at least two core interfaces and, provided that our monitor set is well distributed enough, we actually discover at least two of them.

In order to uniformly randomly sample Internet core routers in a rigorous way, we proceed as follows. We randomly choose a 32-bit IP address t and keep it only if it answers to one test `udp-ping` probe, otherwise it is discarded. Then, we launch our distributed measurement on this target address and we determine whether the router $r(t)$ to which it belongs is a core router and whether t is a core interface of $r(t)$, as explained above. If the two tests are positive, we keep $r(t)$ in our sampling, otherwise it is discarded. At the same time, we obtain a measure of the degree k of $r(t)$ in the core of the network. Doing so for a set of randomly generated 32-bit integer, we obtain a sample of Internet core routers. This sample is biased, in the sense that routers are not selected uniformly at random. But because we discard all routers that were selected by one of their non-core interfaces, we know precisely what is this bias and we can correct it *a posteriori* when inferring the degree distribution. Indeed, the probability of one router r to be in our random sample is proportional to its number of core interfaces, which we determine by our measurement. Then, the observed fraction p'_k of routers of core degree k sampled with this bias is proportional to k times the fraction p_k of routers of core degree k sampled uniformly at random: $p'_k \sim k \cdot p_k$. As a consequence, we obtain:

$$p_k = \frac{p'_k}{k} \cdot \frac{1}{\sum_{i>1} \frac{p'_i}{i}}$$

where the second term is nothing but a normalisation constant to ensure that $\sum_k p_k = 1$. We then use this formula to infer the true degree distribution p_k from the observed one p'_k.

As for the IP-level degree distribution, we implemented our distributed measurement method for the degree distribution of the physical topology using the monitors of PlanetLab. For the targets, we built a list of 3 millions random IP addresses that answer `udp-ping` probes. This took approximately 10 hours using one single machine. Then, we uploaded this target list on each monitor (shuffled for sake of load repartition on the targets) and asked it to probe the list three times in a row. One probing round on the whole list, simultaneously from all monitors, lasted 4 hours. Therefore, in total, the measurement, building the list and probing it three times from each monitor, lasted less than 24 hours. We then gathered the data locally, applied our sampling process and corrected the resulting bias in order to infer the degree distribution, as described above. Figures 1.4 and 1.5 shows the corrected distributions determined by the three consecutive iterations of our measurement procedure. They constitute the

	measurement		
deg	1-st	2-nd	3-rd
2	0.74770	0.74371	0.75214
3	0.19434	0.19838	0.19258
4	0.02727	0.02727	0.02585
5	0.01551	0.01588	0.01486
6	0.00708	0.00640	0.00644
7	0.00206	0.00224	0.00230
8	0.00175	0.00196	0.00147
9	0.00127	0.00131	0.00145
10	0.00057	0.00044	0.00052
11	0.00056	0.00052	0.00047
12	0.00040	0.00044	0.00047
13	0.00020	0.00023	0.00017
14	0.00025	0.00031	0.00031
15	0.00032	0.00009	0.00017
16	0.00014	0.00025	0.00024
17	0.00023	0.00018	0.00015
18	0.00007	0.00007	0.00007
19	0.00007	0.00009	0.00009
20	0.00002	0.00000	0.00002
21	0.00008	0.00015	0.00008
22	0.00006	0.00000	0.00004
23	0.00000	0.00000	0.00002
24	0.00002	0.00000	0.00002
25	0.00000	0.00005	0.00002
26	0.00000	0.00002	0.00002
27	0.00002	0.00000	0.00002
28	0.00000	0.00002	0.00000
29	0.00002	0.00000	0.00001

Figure 1.4: The degree distributions obtained from our three measurements (after bias correction). For each degree k, we give the estimated fraction p_k of core routers with degree k.

most reliable estimations of the degree distribution of the Internet available so far.

The three measurements obtain very close values of p_k for each degree (see Figure 1.4), which shows that the procedure is robust. The first striking observation is that 75% of the nodes in the core have core degree 2 and 19% have core degree 3. Note that, since our method is able to correctly estimate the degrees of low-degree nodes independently of the kind of topology, there is a high confidence on the first values of the distribution, say until degree 10. At the first glance, it is difficult to assert whether the distribution observed by our measurement is heterogeneous or not. However, there are some clear and objective conclusions that can be derived from the result of our measurement. First, given the accuracy of observation of low-degree nodes, if the distribution is a power-law then the exponent of this power-law

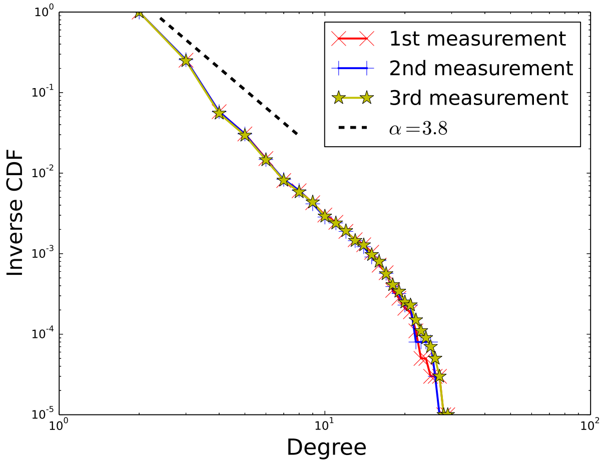


Figure 1.5: Inverse cumulative degree distributions obtained from our three measurements, after bias correction. For each value x on the horizontal axis, we plot the fraction of core routers having degree higher than or equal to x (log-log scale). We also plot the power law of exponent $\alpha = 3.8$ to show that obtained distributions are incompatible with a power law of exponent lower than this.

is necessarily greater than 3.8 (see Figure 1.5). Practically, for such exponents, it is difficult to distinguish between an exponential and a polynomial decay of the tail of the distribution. Therefore, in both cases, the degree distribution of routers in the core of the Internet appears not to be heavy tailed. The maximum degree we observed in our measurement is 29. Of course, we may miss some of the core interfaces of this router but from our simulations, there is only little chance than its true degree is much higher. This indicates that if there exist routers of very high degree in the core Internet, they are very rare. Therefore, the few high values observed in the distribution seem to be outliers rather than consequences of its supposed heterogeneity.

Finally, let me mention that in [67] we showed that the result of our measurements can be used to obtain information not only about the topology of the network but also about routing tables of the routers in the core Internet.

Conclusion

Beside the knowledge it brings on the actual degree distribution of the Internet, the property oriented approach and our measurement method possesses remarkable features, which open exciting perspectives.

On the purely methodological side, our work shows that restricting one's attention to one single property and restricting the information one uses allows to design methods that are theoretically sound and free of bias. In the case of our measurement of the physical topology, discarding the information of routers that are not selected by one of their core interfaces introduces an artificial bias in the sampling, which we can correct *a posteriori* while we could not do so for the original uncontrolled bias. Limiting the information one uses also allows to limit the impact of the errors in the measurement primitives available, as in our measurement of the logical topology where we discard couples of monitor and target for which there exist routes of different lengths.

Finally, the use of sampling instead of exhaustive collection of information has a very positive impact on the sensitivity of the method to missing information. For example, we mentioned that some routers implement a security policy that does not allow them to answer `traceroute`-like probes, which include classical `traceroute` probes and our `udp-ping` probes. This makes these routers invisible for such measurements. To understand the difference of sensitivity to this missing information, imagine two perfect measurements having an exact information on the degree of all nodes: one made by the classical method and one by the `udp-ping` method. Randomly choose half of the nodes and forbid them to answer to the probes of both measurements. With the classical method, the observed degrees of the remaining nodes are drastically affected and so is the output degree distribution. On the opposite, in the `udp-ping` method, the observed degree distribution is about the same, it is simply estimated on half of the nodes, but the information on the degree of the remaining nodes is not affected by the disappearance of the information of the other nodes.

Applying these general principles to the measurement of networks from other contexts or to other properties of the Internet could lead to significant improvements of the accuracy of the knowledge we have of these networks. In the case of the Internet, the method for measuring the degree distribution of the IP level topology could be completed by uniformly sampling core routers (the method we design for the physical topology applies in this case as well) and solving anti-aliasing. This would allow to compare the two kinds of topology, which is an exciting perspective in itself. Moreover, this would open the way to measure another property of interest, namely the clustering coefficient, that cannot be measured with the `udp-ping` approach but which could be measured at the IP level.

More generally, our work also calls for measurement methods focusing on one very precise part of the topology or one very specific kind of configuration in the network. For example, one could determine what is the size of the core and the size of the border, design a specific method to measure the degree distribution of border routers, design a method to precisely evaluate the proportion of high-degree nodes. This specific information would be highly valuable for network design and management.

One key benefit of our measurement method is that it induces a very low load on the network compared to the classical `traceroute` method. There are two reasons for this: i) each monitor sends only one probe to each target, while one execution of `traceroute` sends 10 to 15 probes in average, and ii) with our method, which uses random sampling, probing only a relatively limited number of targets is statistically sufficient to infer the distribution. This low load on the network has some advantages. Our measurement is very quick, 4 hours only, while the data is usually gathered over months or years in the classical approach. This prevents the measurement from being too sensitive to the dynamic of changes in the structure of the network (*i.e.* changes in routing). In first approximation, these changes can be neglected on a period of some hours, while they cannot on a period of some months. In other words, the measurement with our method can be considered as almost instantaneous compared with the time scale of the dynamics of the network topology. Moreover, since it induces a low load on the network, it can be periodically repeated to study the dynamics of the degree distribution of the Internet[5]. This is a key perspective as we have practically no information on the way the Internet evolves. Understanding this evolution and managing it is very important from a practical point of view to guarantee the stability of the services leaning on the Internet. From the theoretical side, it would open the way to answer some appealing questions. Among them is the question of knowing whether the growth of the Internet approximately follows the law of evolution of the preferential attachment [5], or whether it follows a different law. Then, the next step would be to assert whether this law will asymptotically respect the shape of the distribution we measure nowadays or if the characteristics of the topology are in deeper transformation.

[5]Of course, studying the dynamics of other structural properties of the Internet, like those mentioned above, would be of great interest. But dedicated measurement methods should first be developed.

Chapter 2

Dynamic networks

The study of complex networks started with *static networks*, where the existence of nodes and links is independent from any notion of time. Static networks have concentrated the earliest developments of the field and also the most advanced ones. Nevertheless, in parallel, the field has quickly come to the study of another kind of objects, referred to as *dynamic networks*, where the notion of time applies and where the existence of links and nodes is time sensitive: they do not necessarily exist at all times.

It is worth emphasising that dynamics is not just a characteristics of a network among other characteristics, nor a variant or an augmented version of static networks. They are a fundamental concept. Many networks encountered in real-world contexts are intrinsically dynamic and time variation is a key dimension of their organisation and life. For example, contacts between individuals change along one day, communications by messages are intrinsically dynamic, etc. Going further, in several cases, the static networks one studies are an abstraction or a simplification of a real object which is actually dynamic. For example, social networks are usually thought of as being networks where people are linked by acquaintances essentially stable over time. But these relationships actually lay on interactions between people that happen at a very short scale in time, like some minutes or some hours, and which are repeated several times over the period of study. This is the repetition of these interactions that gives rise to the abstraction of one acquaintance between the involved individuals. Therefore, in these situations, dynamic networks fit better to the context of study and constitute a more fundamental form of organisation for the network considered.

In this manuscript, and in particular in this chapter, the term dynamic applies to the changes of the topology of the network. The same term is often used to refer to the dynamics of phenomena taking place over the network, like spreading phenomena for example. Such phenomena are dynamic in the sense that the state of nodes in the network is changing, but not necessarily the topology of the network. The spreading of epidemics for example has received a lot of attention in the context of static networks, where the topology is fixed but the state of nodes, e.g. *infected* or *recovered*, can change over time. Recently, the tendency has been to encompass the two kinds of time evolution under the same term of dynamics. This is in particular motivated by the fact that some important case study of dynamic phenomena take place on highly dynamic topologies, like contacts between individuals, and that the interplay between the dynamics of the topology and the one of the phenomenon itself plays a key role in this context. Nevertheless, the two kinds of time evolution should be distinguished, as one may happen without the other one. Here, except explicitly mentioned otherwise, the terms dynamics and dynamic networks refer to the time evolution of the topology of the network: appearance and disappearance of nodes and links.

The domain of complex networks is young. Even for static networks, which have received earlier attention and a greater amount of developments, and which directly benefit from the notions of graph theory, there are still many important challenges that need to be addressed in order to set solid bases for the domain. In the case of dynamic networks, this is even more true. Taking into account the evolution of their topology appears to be quite challenging. Most of the notions and concept to deal with these objects are still to be developed and to be exploited for their analysis and their modelling. One revealing fact about the state of advancement of the domain of dynamic networks is that there is no clear consensus even on the way to represent these networks. One reason for this is that adapting the notions from graph theory to the dynamic case by incorporating the time inside these notions turns out to be difficult and gives rise to concepts that are often unsatisfying or that only partially capture the nature of dynamic networks.

Dynamic networks have often been thought of as graphs that are changing in time. This is appropriate in some contexts. For example the Internet topology, both at IP level or physical level, is a graph which is changing under the effect of rerouting and rewiring. In this case, the network is mainly stable and changes are events. But in some other contexts, links exist only during a very restricted time, which has a short duration or which is an instant, and the network is constituted by the sum of these versatile links. For example, the network of emails sent between employees of a company is such a dynamic network: links only exist at the very precise moment when the email is received. In this case, there is no stable network and the events are the existence of links.

The dynamic networks we consider in this chapter are of this latter kind. The chapter presents three works, a case study and two methodological developments. The case study deals with the network of contacts between people in a hospital, the second work deals with the structure of changes in dynamic contact networks and the last one considers the choice of an appropriate time scale to form the graph series describing the dynamics. As these three works are mainly independent, the perspectives of each of them are discussed at the end of the corresponding section. Nevertheless, broader perspectives raised by this series of works for the study of dynamic networks are detailed in the general conclusion of the manuscript.

The dynamic network of a hospital [56, 57]

In the past decades, the worldwide rise of AMRB (AntiMicrobial Resistant Bacteria) in hospital environment has became a major issue for public health. The number of persons contaminated by AMRB in hospitals has strongly increased and at the same time AMRB became more and more resistant to currently available antibiotic treatments. This has dramatic consequences such as delays or failures of therapies, prolonged hospitalisation stay and increased mortality. When an individual is colonised by an AMR bacteria, he becomes an occult carrier and can disseminate the bacteria to other individuals, for example when transferred to other facilities. In this context, rehabilitation centres are considered to be a large reservoir of AMRB, offering a great potential for development and dissemination into the community. There are many factors that impact the spread of AMRB, but it is widely acknowledged that the support of transmission is *close proximity interactions*[1] between individuals, which we simply call *contacts* here, and that the next step to reduce transmissions is to control the flux of contacts within the hospital.

To this purpose, within the framework of the MOSAR project, the contacts between individuals in the hospital of Berck-sur-Mer, France, were recorded using sensor networks. The work presented here is about the analysis of the collected data. Its goal is to get an insight into the structure of con-

[1]A *close proximity interaction* is an event consisting of two individuals being at a short distance from each other (typically less than 1 to 1.5 meter) during a while.

tacts within the hospital, both in the topological dimension and in the temporal dimension, the goal *in fine* being to exploit this understanding for lowering the risk of spreading within the hospital. Compared to other data sets collected in similar environments, the data set we analyse here possesses several key characteristics. First, it includes all the individuals of the hospital, both patients and staffs, and not just one part of them, which is crucial in order to accurately evaluate the possibilities of transmission. Second, the measurement has been made on a long period of time, more than 6 months, and with a fine-grain resolution, 30 seconds, which provides a precise view of the contacts and allows to assess the generality of observations made on shorter periods of time, such as one day or one week.

In our data set, one contact between two individuals has a duration which is a multiple of 30s, the resolution of timestamps in the measurement. We analyse both the average daily pattern of contacts in the hospital and the evolution hour by hour of these contacts over one week. The regularity of contacts observed from one day to another justifies that the average daily network is meaningful and representative of one day of activity of the hospital. Our analyses consider mainly three quantities associated to contacts: the *number of contacts* during one period of observation (like one day or one hour), the *cumulated length of contacts* during this period and the *number of adjacency pairs*, *i.e.* the number of couples of individuals that had at least one contact during the period. These analyses are led following a predefined segmentation of the hospital into groups, which are relevant to the management of the hospital: 1) segmentation of the hospital into 9 services dedicated to different cares and tasks, 2) separation between patients and staffs and 3) segmentation into 12 socio-professional categories of staffs, which have distinct roles. Our results [56, 57] show that these *a priori* defined axes are very relevant for the analysis of contacts in the hospital. They reveal clearly marked differences that are likely to have a strong impact on the way bacteria may spread into the dynamic network of the hospital.

For example, the activity of patients and staffs is not similar at all: patients have a much longer cumulated duration of contacts while staffs interact with a higher number of different people during one day. We also observe clear differences in the level of activity of patients or staffs depending on the service they belong to.

In order to study the impact of the segmentation into services on the contacts occurring within the hospital, we computed, for each service, the ratio between the amount of contacts inside the service and the amount of contacts with individuals from other services. We then compared this ratio in the real network of the hospital and in some reference random network where contacts have been randomly redistributed, while preserving the level of activity of each service. This shows that services are strongly introverted, meaning that they all strongly favour contacts inside the service itself rather than contacts with other services. By fixing different parameters of the reference random network used for comparison, we can even show that this propensity to internal contacts cumulates for our three quantities of interest. In other words, even knowing the tendency of forming internal rather than external adjacency pairs, there is in addition a tendency to prefer repetition of contacts for adjacency pairs within one service. And knowing these two tendencies, it is still visible that a longer cumulated duration of contacts is favoured inside services rather than outside. The fact that services are widely introverted suggests that spreading will propagate fast inside services. The possibilities of propagation in the whole hospital then relies on the structure of interconnections between services.

By using again comparisons to well-chosen random networks, we could draw a map of the affinities between services without suffering the effect of the differences in their sizes and in their levels of activity, see Figure 2.1. It shows, in the current configuration of the hospital, which are the couples of services that tend to favour contacts between them. A very interesting fact is

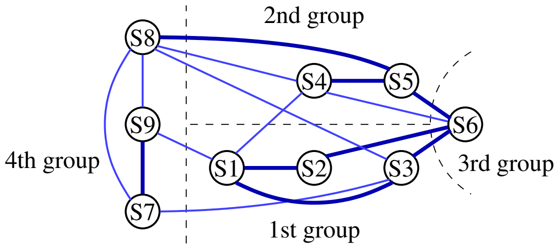


Figure 2.1: Structure of the favoured relationships between services of the hospital. Only clearly favoured relationships are depicted. Bold lines are for relationships that are particularly strongly favoured.

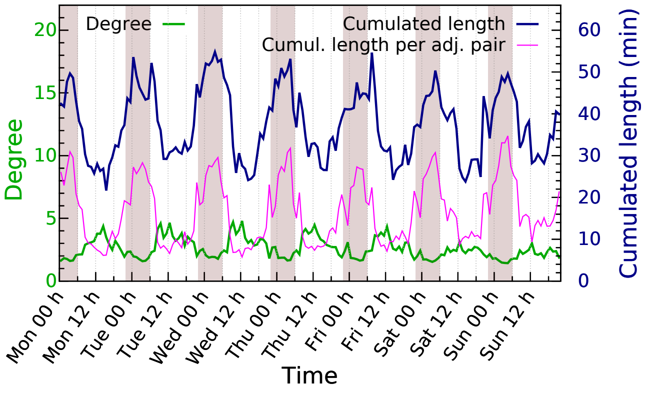
that most of the relationships that are not clearly favoured are actually clearly unfavoured. This means that the structure of contacts between the services of the hospital is clearly marked and probably impacts the spreadings that may occur in the dynamic network. These observations are very useful in practice. They raise questions for practitioners on the reasons why these relationships are favoured and not others, giving the opportunity to discover new elements of understanding and opening the way to try to control the flux of contacts in the hospital, in order to limit the configurations that may favour spreadings.

The analysis of the temporal evolution of contacts within the hospital also reveals some important points. As one can expect, the level of activity of the hospital during one day is very heterogeneous in time (see Figure 2.2). In particular, there is a circadian effect that makes the activity during day time and night time very different. There are also some moments in the day when the quantity of contacts is much higher, for example around 12 AM, at lunch time. Of course, these moments are particularly sensitive for the spread of infections and this characteristics of the dynamics of contacts certainly plays an important role in the way infections can propagate and in their speed of propagation.

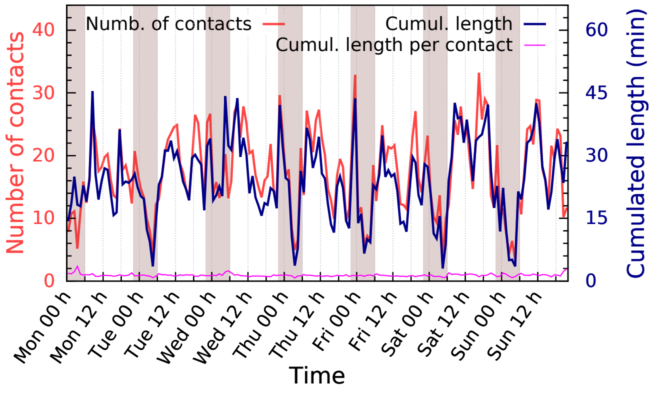
Another striking fact revealed by separately observing the time evolution of the cumulated length of contacts of patients (Figure 2.2.a) and of staffs (Figure 2.2.b) is that the activity patterns of these two kinds of individuals are quite contrasting. For patients, the evolution of their cumulated length of contacts matches perfectly the evolution of the mean cumulated length per adjacency pair. This means that their cumulated length of contacts is higher when they stay longer with the individuals they meet. For staffs, the situation is very different, the evolution of their cumulated length of contacts follows the evolution of their number of contacts, but not the evolution of the mean length per adjacency pair (not depicted). This means that their cumulated length is made by accumulating numerous contacts with various individuals. Again, these differences between patients and staffs are crucial to understand the role that each of these two groups plays in the propagation of infections.

Conclusion. Our work constitutes a first step toward understanding the influence of the structure of the dynamic network of contacts on the spreadings taking place in the hospital. The next step would be to assess the actual impact of the characteristics of the contacts we highlighted above, such as the introversion of services and their specific affinities, on the possibilities of diffusion, for example by using synthetic diffusions.

Another exciting perspective is to use the micro-biological data collected during the MOSAR experiment, which traces week by week the bacteriological strains carried by participants. The data set contains several cases of individuals becoming new carriers. One should design metrics of exposure, based on the contacts of a participant, that are able to explain the contaminations observed in the data set. This would be of great interest as it could also allow to identify in advance individuals that are more likely to be affected by future spreadings.



(a) patients



(b) staffs

Figure 2.2: (a): for patients, mean number of adjacency pairs, denoted degree, per individual and per hour (light green), cumulated length of contacts per individual and per hour (dark blue) and mean cumulated length per adjacency pair and per hour (thin purple line). (b): for staffs, mean number of contacts per individual and per hour (light red), cumulated length of contacts per individual and per hour (dark blue) and mean cumulated length per contact (thin purple line).

Finally, the analyses we presented here are led at the level of large groups of individuals. It would be highly desirable to be able to detect particular roles played by some individuals as well. For example, one individual or a restricted subgroup of individuals may play a key role in connecting two or more services together, or in connecting individuals inside a given service. Identifying such individual configurations and their roles in spreadings would be a valuable contribution toward the goal of limiting the diffusion of AMRB in hospitals.

Structure of changes in dynamic networks [60]

In the case study presented above, we separately studied where the contacts occur in the hospital during one day and the evolution of the quantity of contacts in the whole hospital hour by hour. Doing so, we separated the topological dimension and the temporal dimension of the dynamic network and we studied them independently. This approach gives some valuable information about the network under study. Nevertheless, separating the topological and temporal dimensions of dynamic networks does not allow to fully take into account the deep nature of these objects, where the two dimensions are intimately interwoven. Currently, the domain of dynamic networks does not propose fully satisfying methods to deal with this dual nature.

In the lack of such methods, there is at least one object available to encompass simultaneously both dimensions of dynamic networks: *graph series*. In the series describing a dynamic network, each graph, called a *snapshot*, is formed by the links that have existed between the nodes of the network during a certain time window. In the rest of the chapter, like in many studies, the time windows are disjoint, they all have the same length and they span the entire period of study of the dynamic networks[2].

[2]But the tools presented here are useful in the other cases as well.

Moreover, the set of nodes of the network is fixed, it is the same in all the graphs of the series. In many contexts, this is natural, and in others, this is not a limitation as one can always consider the set of nodes that have existed along the whole period of study. In this way, we concentrate our attention on the dynamics of the links.

The main reason why researchers use the formalism of graph series to describe dynamic networks is because it reduces the study of dynamic networks to the study of graphs. This is very practical as graphs are a widely studied object. Consequently, one immediately benefits of the notions and concepts developed in graph theory. The convenience of using graphs actually hides a risk in which many studies on dynamic networks fall: to focus on the properties of the snapshots instead of the properties of the series. Indeed, since most of these studies consider snapshots independently, they are unable to capture a key information about the dynamics: the relationships between consecutive graphs of the series. This information is essential to understand the evolution of the network, which is more than a simple juxtaposition of states unrelated to each other.

This failure to take into account the relationships between the snapshots has a particularly strong impact in modelling. Producing a synthetic series of graphs by generating independently each graph of the series does not provide a satisfying model of the dynamics. Even if the properties of each graph of the synthetic series are identical to the properties of the corresponding graph in the real series, the produced series is not similar to the original one, because correlations between consecutive graphs are lost. For this reason, several works introduced correlations between the consecutive states of the network, either at the level of links (see e.g. [14]) or at the level of nodes (see e.g. [63]). The limitation of such models is that topological correlations are not taken into account and, as a consequence, the structure of the snapshots produced is not realistic. This comes back to the situation of considering separately the two dimensions of dynamic networks. Therefore, many approaches have spent some effort to encompass simultaneously temporal and topological correlations in the evolution of dynamic networks. The most flourishing trends in the recent years are probably approaches using external characteristics of nodes, like in latent space models (see e.g. [38]) and extensions of stochastic block models (see e.g. [78]), or using agent based modelling (see e.g. [70]). But despite some promising works [68, 49], the domain is still lacking a consensual and satisfying solution to capture both temporal and topological correlations in the evolution of dynamic networks.

The approach we follow in [60] to reconcile these two dimensions is based on *difference graphs*. The difference graph ΔG of two consecutive graphs G_1 and G_2 in the series has the same set of vertices as G_1 and G_2, and for two vertices u and v, uv is an edge in ΔG if and only if either uv is an edge in G_1 and not in G_2, or uv is an edge in G_2 but not in G_1[3]. In other words, the difference graph is the graph whose edge set is formed by all the pairs of vertices whose adjacency relationship changes between G_1 and G_2. This is the rationale behind difference graphs: they contain the changes between two consecutive graphs of the series, and therefore capture the structure of the time evolution of the topology.

The question we investigate in [60] is to characterise the structure of the difference graphs of the series describing a dynamic network. In particular, we want to know whether the changes from one step to another in the series are well spread all over the network or rather concentrated in some restricted part of it. To this purpose, we use two distinct metrics on difference graphs, aiming at evaluating the concentration of changes in the series. The first one is very simple, this is the number of vertices affected by changes, that is the vertices for which at least one of their adjacency relationships with the rest of the

[3]More formally, denoting Δ the symmetric difference on sets, we have $V(\Delta G) = V(G_1) = V(G_2)$ and $E(\Delta G) = E(G_1) \ \Delta \ E(G_2)$.

vertices of the graph has changed. This is exactly the number of non-isolated vertices in the difference graphs, *i.e.* vertices having at least one neighbour. The second metrics we use is a well known graph parameter called the *Minimum Vertex Cover* (MVC) of the difference graph. Formally, a vertex cover in a graph is a subset of vertices such that all edges of the graph are incident to at least one vertex in this subset. The MVC is the minimum number of vertices in a vertex cover. It is worth to note that if S is a vertex cover in ΔG, then the rest of the vertices $V \setminus S$ have no edges between them, meaning that the adjacency relationships linking them are unchanged between G_1 and G_2. All the changes involve at least one vertex of the vertex cover S. Then, intuitively, the MVC of the difference graph gives the minimum number of vertices that can be held responsible for all the changes occuring in the network.

We apply these metrics to a specific kind of dynamic networks, which we call *dynamic contact networks*. They are networks of close proximity interactions between individuals (cf. footnote p. 44), measured by sensors carried by participants to an experiment. We study data sets gathered in various contexts, ranging from a scientific conference to a rollerblade tour in Paris, and including the experiment conducted in a hospital that we presented in the previous section. For each of these data sets, we formed a graph series describing the dynamic network and we computed the series of difference graphs between two consecutive graphs of the series. Forming the initial graph series implies the choice of a length, called *aggregation period*, for the windows on which the contact data of the original dynamic networks is aggregated. We choose these values based on our intuition of what is an appropriate time scale to study the dynamics of the network, depending on the context and the settings in which the data was collected. Beside this, we also study the impact of the chosen aggregation period on the results we obtain. The question of systematically determining an appropriate aggregation period for representing a dynamic network by a graph series is the object of the next section.

Once the series of difference graphs is computed, we compute for each of these difference graphs its number of non-isolated vertices and the value of its MVC. Computing the MVC of an arbitrary graph is NP-hard. In order to obtain the optimal value, we use a straightforward exponential algorithm together with a simplification rule known as the *leaf removal* technique (see e.g. [77]). This technique is known to perform very well on sparse random graphs and it turns out that it performs even better on the real-world data we use. Thanks to this, we could compute the exact value of the MVC for series of more than 500 difference graphs over more than 300 vertices.

Our results show that in almost all the dynamic contact networks we studied, the changes from one graph of the series to the next one are very concentrated. For these networks, the number of non-isolated vertices of the difference graph, *i.e.* the number of vertices affected by changes between two consecutive graphs, is quite restricted. It is much less than the number of non-isolated vertices in random graphs on the same number of vertices and edges. This shows that this observation is not only due to the sparsity of these graphs but instead denotes a very specific property of concentration on a limited proportion of vertices of the graph. This property of concentration is further emphasised by the MVC. Indeed, the number of vertices involved in a MVC is even much more restricted than the number of non-isolated vertices. And again, it is much less than what is expected for graphs of this number of vertices and edges. Pushing further, we generate random graphs having as many edges as the original difference graph and as many vertices as its number of non-isolated vertices. We observe that the expected value of the MVC in such synthetic graphs is clearly higher than the one of the original difference graph. This shows that the low value of the MVC of the difference graphs of the original series is not only a consequence of their number of non-isolated vertices but instead denotes an even stronger property of concentration of changes.

In order to assess the generality of these observations, we also vary the value of the aggregation period we use to form the original series. The conclusions we obtain, though they are quantitatively slightly different, remain qualitatively the same on a wide range of aggregation periods around a reasonable value (ratio of 10 between the smaller and the larger period used). This shows that the property of concentration of changes is a fundamental characteristics of these dynamic networks, which appears independently of the aggregation period, at least when it is chosen in an intuitively reasonable range of values[4].

Conclusion. Our results show a very special structure of the changes between two consecutive graphs in the series describing a dynamic contact network. This constitutes a promising characteristics to exploit in order to encompass both temporal and topological dimensions in the analyses and modelling of dynamic networks, which is a key challenge for the field.

One of the first perspectives of our work is to extend these observations to other structural properties of the difference graphs and to other types of dynamic networks. Concerning the use of difference graphs for modelling purposes, much remains to be done in order to get a satisfying model. One natural way to generate synthetic dynamic networks is to start from a seed graph and to decide at each time step what are the adjacency relationships to be modified. Instead of doing so independently for each edge, like in the Markovian Edge model [14, 6], which raises the limitations previously mentioned, one could generate simultaneously all the changes, *i.e.* the difference graph, by respecting their special structure, for example with regard to their property of concentration around a few vertices. Unfortunately, reproducing the structure of the difference graphs is not enough. One must also decide which vertices should play

[4]For example, 5 to 45 min for the experiment led in the conference environment.

the role of the non-isolated vertices and which ones should play the role of the vertices in the MVC. Consequently, in addition to the structure of the difference graph, one should also study how this structure connects to the structure of the graph of the series itself. To this purpose, distinguishing between edges of the difference graph that correspond to the appearance of one edge and those that correspond to the disappearance of one edge would probably be of great help.

Aggregation of dynamic networks into series of graphs [51, 50]

Many real-world dynamic networks are naturally given in the form of a finite collection $\mathcal{L}$ of triplets (u, v, t), which we call a *link stream*, where $u, v \in V$ are two nodes of the network and t is a timestamp, with the meaning that nodes u and v have a link between them at time t. Depending on the context, these links can represent IP packets sent between two machines of the Internet, emails sent between people, commercial transactions between companies, etc. As we mentioned before, a very common approach to study those dynamic networks is to transform them into series of graphs. The process used to do so is called *aggregation*. It consists in choosing a time window $[a, b] \subseteq [0, T]$ in the initial series, where T is the length of the period of study, and forming the graph $G_{[a,b]}$ with all edges u, v such that there exists a triplet $(u, v, t) \in \mathcal{L}$ with $t \in [a, b]$. Doing so for a collection of windows that covers the entire period of study, one obtains a representation of the dynamic network as a graph series. Very often, as here, the windows are disjoint and all have the same length, but in some studies they may overlap or have different lengths. In all cases, once the series is formed, all subsequent analyses are led on it.

One major motivation for aggregating link streams is to represent them by graphs, and then benefit from the rich set of notions of graph theory to analyse these dynamic networks. Another reason is that, in many cases, it does not make sense to study the network at the scale of the time resolution of the timestamps of the given link stream. For example, in an email dataset, the timestamps of the events (sending of emails) are often given with a 1-second resolution. However, studying the dynamic network at this time scale does not give a general and comprehensive view of its organisation. Hence, aggregation allows to study the network at a scale which is relevant compared to its activity.

If the benefits of aggregation are clear, on the other hand, they also come with some important concerns. Indeed, the length chosen for the aggregation window usually has a strong impact on the properties of the aggregated graph series, see e.g. [65]. This raises the question of which time scale should be chosen to study a given dynamic network and how much the properties studied, based on which conclusions are derived, are sensitive to the length of the aggregation period used. As a consequence, this period should not be chosen without well established evidence, as it is currently done in most of the studies.

Pushing further, it is not even clear whether an aggregated series faithfully describes the original link stream. Indeed, the aggregation process goes along with a loss of information: in each aggregation window, the information on the exact times at which links occur in this window is lost. In particular, in a given time window, it is impossible to know whether a given link (a, b) has occurred before or after another one (b, c). This question, which determines whether it is possible to go from node a to node c, via b, which we call a *transition*, within this time window (only if ab has occurred before bc) is crucial for many phenomena taking place on the dynamic network, such as epidemic spreads, possibilities of communication and cascades of influence for example. The wider the aggregation period, the greater the amount of information lost. At the limit, aggregating a link stream over the whole period of study yields one single static network which misses all the information on the order of occurrences of links and which therefore very poorly captures the structure of the original dynamic network, see e.g. [59]. Then, more generally, for a given aggregation period, one can ask whether the obtained graph series is a faithful representation of the original link stream.

There have been several works aiming at determining appropriate time scales for aggregation of link streams into graph series. The work of [72] is probably the closer to our motivations as they also care about the loss of information due to aggregation. The specificity of our approach is that we want to avoid to decide what is the amount of loss which is acceptable. Instead, we aim at observing a natural change in the way the link stream responds to aggregation at a given time scale, which we call the *saturation scale* and which we denote γ. The difficulty of doing so, as the need for doing so, comes from the fact that when the aggregation period grows, the classical properties of the aggregated graph series drift from one extremal value to another. Moreover, they do so monotonically and smoothly and do not undergo any noticeable change at any time scale.

A key contribution of our work [51, 50] is to exhibit a property that is able to reveal a qualitative change in the way the link stream responds to aggregation at a certain time scale. Our method is based on the *occupancy rate* of *minimal trips* in the aggregated graph series. These notions, which we introduce, are based on the classical notion of *temporal path* in a graph series. A temporal path is a series of links occurring chronologically at pairwise distinct times in the graph series.

Definition 1 (Temporal path) *In a series of graphs $\mathcal{G} = (G_k)_{1 \leq k \leq K}$, a temporal path P is a sequence (u_i, v_i, t_i) of triplets, with $1 \leq i \leq l$ and $l > 0$, such that $\forall i, u_i v_i \in E(G_{t_i})$ and $\forall i > 1, u_i = v_{i-1}$ and $\forall i, j$, if $i < j$ then $t_i < t_j$.*

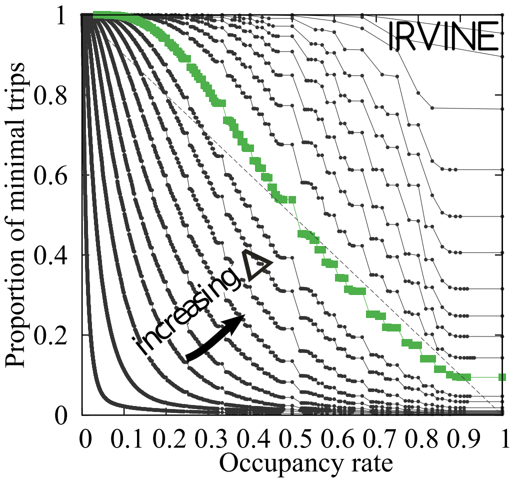
There are two notions of length associated to a temporal path P: the *topological length*, also called *number of hops* and denoted *hops*(P), which is the number l of links in the path and the *temporal length*, more simply called *duration* and denoted *time*(P), which is the number of graphs $t_l - t_1 + 1$ of the series between the starting time t_1 and the arriving time t_l of P.

Definition 2 (Trip and minimal trip) *A* trip *is a quadruplet* (u, v, t_{dep}, t_{arr}) *such that there exists a temporal path from u to v whose starting time from u and arriving time at v are both in the interval* $[t_{dep}, t_{arr}]$*. A trip* (u, v, t_{dep}, t_{arr}) *is* minimal *if there exists no trip from u to v in an interval* $[t'_{dep}, t'_{arr}]$ *strictly included in* $[t_{dep}, t_{arr}]$ *(*i.e. $[t'_{dep}, t'_{arr}] \subsetneq [t_{dep}, t_{arr}]$*).*

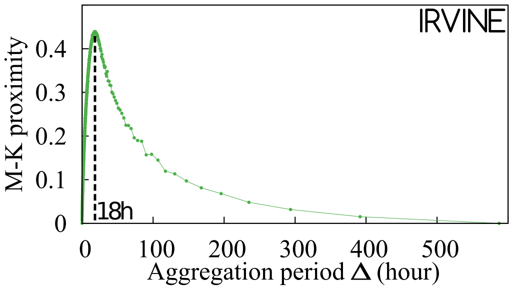
Definition 3 (Occupancy rate) *For a graph series* $\mathcal{G}$ *and a temporal path P in* $\mathcal{G}$*, the* occupancy rate *of path P, denoted occ*(P)*, is defined as occ*$(P) =$ *hops*$(P)/$*time*(P)*. The occupancy rate of a minimal trip* $(u, v,\ t_{dep}, t_{arr})$ *is the occupancy rate of a temporal path starting from u at* t_{dep} *and arriving at v at* t_{arr} *and having the minimum number of hops among such paths.*

Our method to determine the saturation scale γ is as follows. We make the aggregation period Δ vary from its minimal value, the resolution of the timestamps, until the whole length T of the period of study of the network. For each value of Δ we form the aggregated graph series $\mathcal{G}_\Delta$ for which we compute the set of minimal trips and their occupancy rates. Then, for each Δ, we plot the distribution of occupancy rates of all the minimal trips in $\mathcal{G}_\Delta$ (considering all pairs of nodes and all time intervals), see example in Figure 2.3.a for the Irvine network, a network of online messages between students. The key observation here is that when Δ increases, the distribution of occupancy rates goes from values concentrated around 0 to values concentrated around 1 (which is expected) in a very particular manner[5]: it first stretches until it occupies almost uniformly the segment

[5]We applied our method to several link streams and observed the same behaviour.



(a) distributions



(b) M-K proximity

Figure 2.3: (a): Inverse Cumulative Distributions (ICD) of the occupancy rates (x-axis) of the minimal trips of the aggregated series $\mathcal{G}_\Delta$ for several values of the aggregation period Δ in the range $[1, T]$, for the Irvine network. (b): M-K proximity (y-axis) of these distributions with the uniform density distribution according to Δ (x-axis).

$[0, 1]$ and then concentrates again, around the value 1. The value of Δ for which the distribution is maximally stretched on $[0, 1]$ is the saturation scale returned by our method. It is computed as the value of Δ maximising the proximity of the distribution to the uniform density distribution on $[0, 1]$ for the $M - K$ distance, see Figure 2.3.b.

This very particular behaviour of the distribution of occupancy rates is intimately related to the loss of information in the aggregation process. A very low occupancy rate for most minimal trips of the series denotes that the data in each aggregation window is sparse, which implies that the information contained in the link stream is mainly preserved in the graph series. On the opposite, a very high occupancy rate for most of the minimal trips reveals a loss of information. Indeed, this indicates that, at each time in the graph series, there is a high probability to find a next hop to perform on any given shortest path, meaning that, in each snapshot, a high proportion of nodes are involved in a high number of links. Then, at the same time, the information on the temporal order between these links, *i.e.* the existence or the non existence of a transition using two of these links, is lost, which constitutes the essential loss resulting from the aggregation process.

Around the saturation scale γ, there is a change in the way the aggregation affects occupancy rates. In the first phase of variation of the aggregation period, below γ, only the low values of the distribution increase, while the proportion of high occupancy rates stays nearly constant. This means that during this phase, the effect of increasing the aggregation period is mainly to fill the lack of data in the aggregation windows without inducing a significant loss of information. On the opposite, in the second phase, beyond γ, there is a strong increase of the proportion of minimal trips having a very high occupancy rate, 1 or close to 1, indicating that the loss of information due to aggregation becomes non-negligible. Therefore, the saturation scale γ appears as a separation between the range of values, below γ, where the aggregated graph series still faithfully describes the original link stream and the range of values, beyond γ, where aggregation alters the properties of propagation of the original link stream.

Conclusion. Our method applies to both discrete and continuous time, and to both undirected and directed links. Unfortunately, it is able to deal only with links that are punctual events. However, in some contexts, the links of the dynamic network last during an interval of time (e.g. physical contacts between individuals). One of the main perspective of our work is to adapt our method to this type of networks. A promising way would be to develop a notion of minimal trip that is specifically adapted to links that have a duration.

Another desirable enhancement of the method would be to detect different aggregation scales for different parts of the dynamics. Many link streams have an activity level which is heterogeneous in time. As a result, the time scale at which the loss of information occurs during the aggregation process depends on the part of the dynamic considered. Therefore, the saturation scale returned by our method should be seen as an average scale taking into accounts all parts of the dynamics. It would be very interesting to enhance the method so that it can automatically identify different aggregation scales relevant to different parts of the dynamics. This would allow either to aggregate the link stream with a variable length window or to use the shorter scale detected, which is also the more preserving one, to aggregate the whole link stream.

Chapter 3

Modelling complex networks

The word model has many different meanings depending on the community in which it is used. Talking about complex network topology, which is a quite precise topic in modelling, three are at least three kinds of models: 1) models that explain the topology of complex networks, 2) models used as definition of complex networks and 3) models that generate complex networks. Some models eventually fulfils several of these goals.

The two most famous models of the first type are the model of Watts and Strogatz [76], which shows how some characteristics of complex networks can be obtained from strongly structured networks by perturbing them by a relatively small amount of randomness, and the Barabási-Albert model [5], which shows that the power-law degree distribution can result from a growth process of the network based on the *preferential attachment* rule, stating that new incoming nodes tend to favour links with high degree nodes. Such models are not intended to be used in order to generate synthetic networks, though they are sometimes used, inadequately, to do so. For this reason they have received a lot of undeserved criticism, as their true goal is not to generate realistic topologies but simply to give more understanding on some characteristics of the topology of complex networks.

The second kind of models, which I call definition models, aims at capturing some key properties of the topology of complex networks in a mathematical definition. One classical way to do so is to equip the class of networks satisfying a given set of properties with a uniform probability, like in [31, 58] for example. Complex networks are then thought of as a uniformly randomly chosen network among the class. This fulfils the lack of mathematical definition of complex networks by artificially giving them one, which is very useful to reason and to make proof about these networks.

The goal of generation models, the third type above, is to give a construction process to generate synthetic network topologies. They do not necessarily provide a nice or useful mathematical definition of the class of networks they generate. Nevertheless, they share a common goal with definition models: delimiting, by a generation process instead of a definition, a class of networks which has all the common properties observed for real-world networks and which remains general enough, meaning that the class does not satisfy other undesired properties that are in general not satisfied by real-world networks. Note that one definition model $\mathcal{M}$ would always give rise to a generation model if it was possible to design a process to uniformly randomly pick one network in the class mathematically defined by $\mathcal{M}$. Unfortunately, designing such a process in a way which is computationally efficient is sometimes out of reach.

In this chapter, we adopt the point of view of generation. Our goal is to design generation processes that faithfully reproduce the structure of complex networks. Ideally, we would like to obtain a model that reproduces the four main properties observed for these networks, while remaining general enough: low global density, short distances, heterogeneous degree distribution and high local density (compared to the global one). Some of these properties are more easily reproducible by models, because they are properties of random graphs. Indeed, random graphs parameterised by their number of vertices and their density have short distances even with a very low density. This is the reason why, until the recent rise of complex network science, the Erdös-Rényi model [31] was used to model networks encountered in practice. It is only after the discovery of the fact that most of these networks actually have a heterogeneous degree distribution (power-law like) that a new model has been used, referred to as the *configuration model* [7, 58]. The configuration model generates random graphs parameterised by their number of vertices and by their degree distribution (which, in particular, fix the global density). It therefore produces networks that have the three first desired properties: low global density, short distances and heterogeneous degree distribution (fixed as a parameter of the model). This is probably the most widely used model for generating complex network topologies.

However, this model also shows a sharp limitation: the graphs it produces have a low local density (close to their global density). Going beyond this limitation is a major challenge for the domain of modelling of complex networks. Despite several interesting approaches and many noticeable results [44, 30, 40, 52, 55, 61, 36, 33, 75, 69, 9], the domain does not offer until now a fully satisfying solution for generating graphs with high local density compared to their global density. The difficulty to do so comes from the fact that a high local density is not a property of random graphs. On the opposite, it is instead the property which most clearly reveals that complex networks differ from random graphs. Following the idea of the Watts and Strogatz model, the structure of complex networks can be understood as a balance between a strongly structured organisation of their topology, responsible for the high local density of these graphs, and a random organisation of some links in the network, responsible for short distances.

Therefore, a very important challenge for the domain is to randomly generate graphs with high local density. In this chapter, we present two new ways toward this objective. For both of them, the general idea sustaining our approach is to obtain all properties of complex networks cited above as a consequence of a higher order property. The first way proposes to focus on the maximal cliques of complex networks and on their overlapping structure, and to generate graphs having a similar structure of their maximal cliques. The second way we present consists in generating separately the strongly structured part of complex networks and the part of randomness they contain. The key point to obtain a satisfying generation model with this approach is to generate strongly structured topologies that are not caricatural and that are indeed close to the actual topologies of complex networks. To this purpose, we use graph editing problems to determine which combinatorial structures are closer to those of real-world networks.

Overlaps of maximal cliques [46, 20, 26]

In this section, our goal is to generate synthetic graphs having the classical properties of real-world networks not by trying to encode these properties directly in the generation process but instead by obtaining them as consequences of another, more fundamental, property. We are in particular interested in local density which is the property that appears until now to be the most difficult to reproduce by random generation processes. Usually, local density is measured thanks to the clustering coefficient. This coefficient admits two distinct definitions (cf. technical preliminaries) which are based on similar ideas to appreciate the density within the neighbourhood of a vertex, but which are not mathematically equivalent up to a constant factor. There may be several reasons why it is so difficult to generate graphs with prescribed clustering coefficients. This may come from the fact that the notions of clustering coefficient are not necessarily the best ones to appreciate local density, or from the fact that, as mentionned above, local density is not a property of random graphs. In order to overcome these difficulties, we generate graphs by their maximal cliques instead of generating them by their edges. Doing so, we obtain graphs with a high clustering coefficient, induced by the properties of their maximal cliques.

Clearly, generating cliques directly forces a high local density for the graph. Moreover, generating cliques of prescribed sizes is easy. It can be done thanks to the configuration model [7, 58] applied to the vertex-clique-incidence bipartite graph $B(G)$ of G. Bottom vertices of the bipartite graph $B(G)$ are the vertices of the graph G to be generated and top vertices are the maximal cliques of G. Each vertex of $B(G)$ is assigned a number of half edges corresponding to its degree and the half edges of top vertices are randomly matched with the half edges of bottom vertices. The parameters of the model are the number of bottom vertices of $B(G)$ and their degree distribution, as well as the number of top vertices of $B(G)$ and their degree distribution.

This approach, named the *bipartite model*, has been initiated in [61, 36, 37] and showed very promising results: in this way, one can obtain graphs having a high local density (thanks to the clique structure) and a heterogeneous degree distribution that is controlled by the degrees of the vertices in the vertex-clique-incidence bipartite graph. However, the bipartite model suffers from a severe limitation: when generating the edges of the bipartite graph at random, the obtained neighbourhoods of the upper vertices intersect only on one (or zero) vertex with a very high probability (see [36, 37]). This is not the case in real-world networks, where most of the maximal cliques have *non-simple*[1] overlaps with some others. Thus, even though it gives the desired properties concerning degree distribution and local density, the bipartite model results in graphs having a caricatural structure and a too high global density (or equivalently too many

[1]In all this section, *non-simple* means of cardinality at least two.

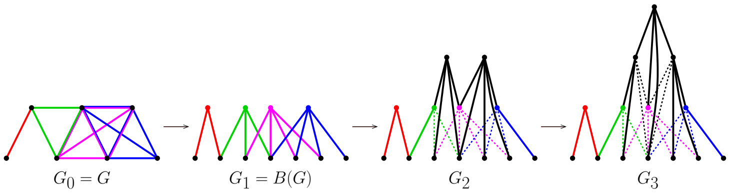


Figure 3.1: Example of the weak-factor series of some graph G. From left to right: the original graph $G = G_0$, its vertex-clique-incidence bipartite graph $B(G) = G_1$, the tripartite graph G_2 of the series, and the quadripartite graph G_3 on which the series terminates. The dashed edges are those belonging to some non-simple maximal biclique used in the factorisation steps. Note that the multipartite graph obtained at termination of the series constitutes an unambiguous encoding of the original graph, as the factorising operation is reversible.

edges). This is an intrinsic limitation of the prescribed-degree generation technique in the case where the bipartite graph is sparse.

In [46], we introduce a new object intended to correct this drawback. The idea is to encode the non-simple intersections of maximal cliques of a graph G by the neighbourhoods of some vertices in some other suitably defined graph, so that such objects can be randomly generated using the prescribed-degree generation technique of [7, 58]. In order to define such an encoding of a graph G, we proceed as follows. We start from the vertex-clique-incidence bipartite graph $B(G) = (V_0, V_1, E)$ of G and we create a new level V_2 where each vertex x corresponds to a non-simple *maximal biclique*[2] B of $B(G)$, x being made adjacent exactly to the vertices of B. Then, we can delete the

[2]A *biclique* in a bipartite graph is a subset of vertices inducing a complete bipartite graph, *i.e.* with all possible edges between top and bottom vertices. Here, maximal means maximal for inclusion.

edges between the vertices of B as they are now encoded by the presence of x. Doing so simultaneously for all non-simple maximal bicliques of $B(G)$ gives a tripartite graph in which the neighbourhoods on V_0 of vertices at level V_1 have no non-simple intersections anymore. Then, we can iteratively repeat the operation by considering, at each step of the process, the maximal bicliques between the vertices on the uppermost level and the rest of the vertices of the multipartite graph. This operation is called the *weak-factor* and the series $(G_k)_{k\geq 0}$, with $G_0 = G$ and $G_1 = B(G)$, obtained by iteratively applying the weak-factor operator is called the weak-factor series of G.

When the weak-factor series terminates, see example in Figure 3.1, we obtain a multipartite graph[3] without any non-simple intersection of neighbourhoods. We can therefore generate similar structures at random using the prescribed-degree generation method without bumping into the problem raised by [36, 37]. Of course, in order to obtain a multipartite graph that has no non-simple neighbourhood intersections, it is mandatory that the iterative factorising process terminates. It turns out that this termination property is not granted with the most immediate definition of the factorising operation, namely the weak-factor operator defined above. There exist graphs for which the weak-factor series does not terminate, see example in Figure 3.2.

This is the reason why the rest of this section is devoted to slightly modifying the definition of the factorising operation in order to obtain termination of the series for all graphs, and therefore always obtain an object suitable for our modelling purposes. Let me mention that this approach is new and somehow orthogonal to the questions traditionally investigated in clique graph theory. Most works in the field try to characterise the graphs for which the series defined by an operator terminates and those for which it does not, without questioning the definition of the operator itself. Here, the specificity of our approach

[3]Note that this multipartite graph is an encoding of the original graph, as the factorising operation is reversible.

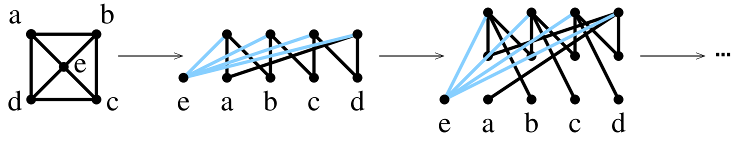


Figure 3.2: An example graph for which the weak-factor series is infinite. From left to right: the input graph G, its vertex-clique-incidence bipartite graph $B(G)$, and the tripartite graph denoted $B(G)^+$ of its weak-factor series. The shaded edges are the ones involving vertex e, which plays a special role: all the vertices of the upper level of the multipartite graph are adjacent to e. The structure of edges between vertices of V_2 and vertices of $V_1 \cup \{e\}$ in $B(G)^+$ is identical to the one between levels V_1 and V_0 in $B(G)$, revealing that the series will not terminate.

comes from the fact that we use the series to define a (multipartite) graph, which requires termination.

In the example above (see Figure 3.2), the non-termination of the weak-factor series is due to the fact that vertex e is the base for an infinite number of factorisation steps. A natural idea to avoid this is to restrict the set of bicliques used for creating the vertices of level V_k. Instead of using all maximal bicliques between the uppermost level V_{k-1} and the rest of the vertices, we use only the bicliques that have at least two vertices at level V_{k-2}, and that are maximal for inclusion. In this way, we forbid a vertex to be responsible for the creation of a new vertex an infinite number of times, as the creation of a new vertex depends only on levels V_{k-1} and V_{k-2}. We call this restricted factorising operation the *factor graph*. Unfortunately, this sole restriction still does not guarantee the termination of the series. In [26], we provide an example of a graph that gives rise to an infinite factor series.

Consequently, in order to guarantee convergence, we constrain further the definition of the factorising operation we use. We obtain two distinct definitions that guarantee convergence, which we call respectively the *clean factor* [20] and the *strong factor* [26]. They are not directly comparable in terms of strength of the restriction, though we give some arguments explaining that the second one, the *strong factor*, can be considered as a lighter restriction than the one operated by the *clean factor*. As a counterpart, the *clean factor* sheds some light on the structure of the multipartite graph obtained at termination which is remarkable in itself.

In the clean factor we do not only require that the neighbourhoods of vertices at level V_{k-1} involved in the creation of a new vertex at level V_k share at least two vertices at level V_{k-2} but we also require that those vertices have the same neighbourhood at level V_{k-3}. Intuitively, this avoids redundant creations of vertices that would threaten termination. But this supplementary condition is not only a technical condition used to guarantee termination. Actually, with this definition, the graph on which the clean-factor series terminates is a fundamental combinatorial object. More explicitly, its vertices are in bijection with the chains of the inclusion order $\mathcal{L}$ of the non-simple intersections of maximal cliques of the input graph G. In addition to the surprising link it shows between these two combinatorial objects, it also opens the perspective to use directly $\mathcal{L}$ for modelling purposes. Moreover, it makes the speed of termination of the series explicit: the number of levels in the multipartite graph on which the series terminates is exactly $h + 3$, where h is the height of order $\mathcal{L}$.

One consequence of restricting the factorising operation is that the multipartite graph obtained at termination, which we use for modelling, still contains couples of vertices for which the intersection of their neighbourhood is non-simple (even though the bigger part of those intersections has been removed by the factorising process). The less restricted the factorising operation used, the less important is this phenomenon. Therefore, in order to obtain the best possible multipartite graph for modelling purposes, it is highly desirable to constrain the factorising operation as little as possible. This is the reason why in [26] we prove termination of the series under a different restriction of the factorising operation called the *strong factor*. In the strong factor, as in the clean factor and the factor, we require that the neighbourhoods of vertices at level V_{k-1} involved in the creation of a new vertex at level V_k share at least two vertices at level V_{k-2}. On the other hand, we do not require equality of the neighbourhoods at level V_{k-3} of the involved vertices at level V_{k-1}. Instead, we ask that the intersections of the neighbourhoods of the vertices at level V_{k-1} is at least 2 on all the other levels, that is from level V_{k-2} to level V_0. It is not immediate to see why this definition of the factorising operation is less restrictive than the one of the clean factor. In [20], we prove that the equality condition on neighbourhoods at level V_{k-3} actually implies equality of neighbourhoods at all levels and also implies that all these neighbourhoods have cardinality at least 2. Therefore, though it does not appear directly in their definitions, the factorising operation of the strong factor is less constrained than the one of the clean factor. Moreover, it is quite remarkable that, despite the fact that it is less restricted, the strong factor series never terminates later than the clean factor series. It means that the multipartite graph obtained at termination of the strong factor series never has more levels than the one resulting from the clean factor series, though its number of vertices is usually larger.

For the practical feasibility of our modelling approach, it is essential to be able to compute the clean factor series and the strong factor series for real-world networks. One difficulty for doing so is that the problem of enumerating the maximal cliques of a graph and the maximal bicliques of a bipartite graph is computationally hard. On the positive side, there exist algorithms to do so in (low) polynomial time per output maximal clique and biclique. Then, the practical feasibility of computing the series we defined mainly depends on the number of vertices in the multipartite graph obtained at termination. For arbitrary graphs, this number (which is at least the number of maximal cliques of the graph) may be exponential in the number of vertices of G. Nevertheless, for the clean factor graph, we could show that under reasonable assumptions of the structure of the initial graph G, namely if the size of its maximal cliques and the number of maximal cliques in which is involved one vertex are both bounded by constants, the number of vertices in the multipartite graph on which the series terminates only grows linearly in the number of vertices of G. In practice, we could compute the clean-factor series for graphs of hundred of thousands of vertices and millions of edges.

Using the results of these computations, we could test our modelling approach. To this purpose, we used only the first three levels of the multipartite graph. We found out that the graphs generated in this way offer a clear improvement on the bipartite model: as expected, their local density is high due to the fact that they are generated by their cliques and at the same time, their number of edges remain much closer from the number of edges in the original graph, which was the effect wanted when introducing the multipartite model instead of the bipartite one. These preliminary results are quite promising and could be improved by using the upper levels of the multipartite graph as well. On the other hand, in the generation of the tripartite graph, the prescribed-degree generation technique cannot be applied directly. In order to get good results as the one we obtained on the graphs we tested, we needed to couple this technique with a rewiring step whose purpose is to avoid that some vertices of the tripartite graph receive a too high number of neighbours compared to what they should. This supplementary step makes the generation process more intricate and less attractive.

Conclusion. The main perspective of this series of works is to use the multipartite graphs that we define to generate synthetic graphs. There are a number of way to do so that should be explored. The first one would be to control in a simpler way the degree of vertices in the generation process of the tripartite model and also to extend this generation process to a higher number of levels in order to get the full benefit of the multipartite model.

Another appealing way is to avoid the direct generation of the multipartite graph and instead use mixing techniques. These techniques start from a given graph (here a multipartite graph) and apply a series of random elementary modifications to this graph, which preserves a certain property of the graph. In the case of our multipartite model, such a property and an elementary modification preserving it are still to be defined.

Finally, the correspondence we highlighted between the clean factor series and the inclusion order $\mathcal{L}$ of the non-simple intersections of maximal cliques of the input graph G suggests that $\mathcal{L}$ could also be used for modelling. The question is whether generating $\mathcal{L}$ is easier than generating the multipartite graph.

Approximation by strongly structured graphs [28, 29, 25, 24]

After the work of Watts and Strogatz [76], it became clear that complex networks are not completely random graphs, as testified by their local density. Though they have some properties of random graphs, they also have a special structure. Then, a natural idea is to describe their topology as the superposition of two parts, a structured part and a random part, making them inherit of some characteristics of both parts. Their structured part is induced by the context where they come from, which imposes constraints on the way links are established within the network, e.g. geographical constraints, technical constraints, economical constraints or sociological constraints. Their random part, on the other hand, comes from the fact that these constraints are not strict: most of the links follow them but a few links do not.

For example, most of one's acquaintances are geographically centred around the place one lives but one may still know one or a few people living in a completely different part of the world. The idea of separating a structured part and a random part in the topology of the network is the one underlying the models of [76] and [43], which are used to explain one phenomenon but are not intended to generate realistic topologies.

On the contrary, here we use this idea for generation purposes. We want to describe complex networks, and then generate them, as strongly structured graphs, *i.e.* graphs mathematically defined as those exactly satisfying some property, altered by a set of random modifications. There are many types of modifications of interest one may consider, such as rewiring of edges (as in [76]), addition of edges (as in [43]), deletion of edges, swap of two edges [64], contraction or subdivision of edges for example. Here, we focus on addition and deletion of edges. These two simple elementary operations can simulate many of the other kinds of modifications cited above. Then, the base idea of our modelling approach is to start from a strongly structured graph and to apply to it a set of randomly chosen edge additions and edge deletions in order to obtain a new graph which is the result of our generation process. Randomly generating such modifications is an easy task. The main challenge in order to obtain realistic synthetic graphs is to start the generation process with strongly structured graphs that do not have a caricatural structure but a structure close to the actual structure of real-world networks. In order to determine whether there exist such graphs and to find them, we need to consider the following problem: given a real-world network, can we decompose it into a strongly structured graph plus a small set of modifications?

This is precisely the question we address here, thanks to *graph editing* problems which are formulated to this purpose. Editing a graph consists in changing some of the adjacency relationships between its vertices: add some edges and delete some others. The goal is to perform a number of modifications which is as small as possible so that the modified graph satisfies a target property given in advance (e.g. being chordal). The number of adjacencies to be changed measures how far the initial graph is to satisfy the target property. The interest is to retrieve the underlying combinatorial structures of real-world networks by removing the noise hiding them (corresponding to the changes performed). In this way, we can describe real-world complex networks as strongly structured graphs that are slightly altered. In the rest of this chapter, we design algorithms to edit an arbitrary graph into some target classes of graphs and we use the results of edition obtained for real-world networks to test the performances of our modelling approach based on the approximation of complex networks by strongly structured graphs.

Algorithms

This approach relies on the possibility, given an arbitrary graph and a target property, to find a number of modifications to perform on the graph (addition or deletion of edges) which is as small as possible to obtain the considered property. Unfortunately, these problems are in general computationally difficult (NP-hard) if one wants the minimum number of modifications. This means that computing an exact solution to the problem is practically unfeasible when the input graph is large, as it is typically the case of complex networks encountered in practice. Dealing with this computational difficulty is a key issue to be addressed for the feasibility of our modelling approach. Fortunately, it is often possible to design polynomial time algorithms that provide non-optimal but satisfying solutions to the problem. To obtain such heuristics, one classical approach is to restrict the modifications allowed to addition of edges only (*completion problems*) and to ask for a set of edges to be added which is only minimal for inclusion but not necessarily of minimum cardinality. This works for many target properties and in particular for completion into chordal graphs, interval graphs and cographs, for which there already exist polynomial-time algorithms. In this section, I present improved algorithms for interval graphs and cographs as well as the first algorithm for permutation graphs. These classes (see [11] for definitions, or the technical preliminaries of this manuscript) are particularly interesting in the present context as they have some characteristics similar to those of real-world networks. Cographs, and permutation graphs which are a generalisation of them, have a natural multiscale community structure induced by their modular decomposition tree. The edition problems into the class of cographs and into one of their subclass, quasi-threshold graphs, have even been already used to analyse real-world complex networks [10, 42]. Interval graphs, and chordal graphs which generalise them, have simultaneously a high local density and a low global density due to the arrangement of their maximal cliques along a linear structure (or a tree-like structure in the case of chordal graphs).

Most of the minimal completion algorithms in the literature, as well as those presented here, are incremental. Starting from an empty set of vertices, the vertices of the graph are added one by one and at each step the algorithm computes a minimal completion restricted to the subset of vertices treated so far. The key property that allows to design efficient algorithms in this way is that, provided that the target class of graphs is closed by induced subgraph and by addition of a universal vertex (which is the case for all the three classes we consider here), at each step, one does not need to reconsider the minimal completion computed previously. Indeed, it is sufficient to compute only a completion of the neighbourhood of the new incoming vertex x_{i+1}. Adding such a set of edges to the minimal completion computed so far for $G[\{x_1, x_2, \ldots, x_i\}]$ always results in a minimal completion of $G[\{x_1, x_2, \ldots, x_i, x_{i+1}\}]$. This is why, in the sequel, we only describe one incremental step of the algorithms. The order in which the vertices of the graph are added does not matter. Of course, one may still wish to use a special order with the aim of either improving the running time of the algorithm or lowering the cardinality of the set of modifications returned, but we do not seek to exploit this possibility here. In the algorithms we describe, the order in which the vertices are treated is arbitrary.

The algorithmic contributions presented here are as follows. We improve the time complexity of minimal completion into an interval graph from $O(nm)$ to $O(n^2)$, we design the first algorithm for minimal completion into a permutation graph, which works in the same complexity, and we design an important improvement of the algorithm for cographs: a real edition algorithm that uses both deletions and additions of edges and which gives a set of modifications which is minimal for inclusion, running in $O(n^2)$ time as well.

Minimal interval completion. There were already several existing algorithms for minimal interval completion of an arbitrary graph, the best complexity being $O(nm)$ [71]. The problem of interval completion is closely related to the problem of chordal completion (interval graphs are a subclass of chordal graphs), which has been extensively studied because of its links with treewidth and fast multiplication of sparse matrices. The two best complexity for chordal graph completion are $O(nm)$ [66] and $O(n^\alpha \log n)$ [39], with $\alpha < 2.376$, where $O(n^\alpha)$ is the time required for the multiplication of two $n \times n$ matrices. For the problem of interval completion we devised an $O(n^2)$-time algorithm [28, 29]. In practice, this improvement of the complexity is meaningful and can allow to treat graphs significantly larger than those that can be treated with an $O(nm)$-time algorithm. On the theoretical side, it is interesting to note that it brings the complexity for interval completion below the best complexity known for chordal completion.

Technically speaking, this result leans on two main ingredients. Firstly, given an interval realiser $\mathcal{R}$ of G and a vertex x to be added to G along with its neighbourhood in G, we are able to find efficiently a completion H of the neighbourhood of x such that *i)* $G+x$ is an interval graph and *ii)* this completion respects[4] $\mathcal{R}$ and *iii)* it is minimal for inclusion among the completions satisfying the two preceding conditions. Unfortunately, depending on the realiser $\mathcal{R}$ which is used for G, the minimal completion respecting $\mathcal{R}$ may not be minimal among all completions of the neighbourhood of x. This difficulty can be overcome by using a special representation of interval graphs, called PQ trees, which represents all interval realisers of an interval graph in space $O(n)$. Then, the second key ingredient we use to solve the problem is to find, thanks to the PQ tree, an interval realiser for which it is ensured that all minimal completions respecting it are minimal among all completions of $G+x$. One incremental step of the algorithm requires at most $O(n)$ time, resulting in an $O(n^2)$ overall time complexity.

Minimal permutation completion. In [25], we designed an $O(n^2)$-time algorithm for computing a minimal completion of an arbitrary graph into a permutation graph. To the best of our knowledge, this is the first minimal permutation completion algorithm. The approach we followed is similar to the one we use for interval graphs, only the details dealing with the structure of the target class are different. The first difference is that in the case of permutation graphs, computing a minimal completion of $G+x$ respecting a given permutation realiser is technically much more challenging than in the case of interval graphs. Nevertheless, our algorithm remains conceptually simple as it can be decomposed into two independent steps. The first step finds a couple of intervals (I_1, I_2) in the permutation realiser (π_1, π_2), with I_1 in π_1 and I_2 in π_2, such that it is guaranteed that the segment of the new vertex x can be inserted in a way that results in a minimal completion respecting the realiser, without determining the way it has to be inserted. Then, in the sec-

[4]A completion H of $G+x$ respects a realiser $\mathcal{R}$ of G iff there exists some realiser $\mathcal{R}'$ of H such that removing x from $\mathcal{R}'$ results in $\mathcal{R}$.

ond step, the algorithm finds such an insertion position for the segment of x by scanning simultaneously the two intervals I_1 and I_2. Like in the case of interval graphs, the fact that the minimal completion respecting a realiser is minimal among all completions depends on the chosen realiser. Fortunately, as for interval graphs there exists a tree representation of permutation graphs, called the modular decomposition tree, that represents all realisers of a given permutation graph in $O(n)$ space. Using this representation, our algorithm is able to compute a minimal completion of $G + x$ in $O(n)$ time.

Minimal cograph edition. We also considered the problem of minimal completion of a graph into a cograph, whose class is a proper subclass of permutation graphs. Before our work, there was already an algorithm for minimal cograph completion [53], running in linear time with regard to the size of the output graph, *i.e.* $O(n + m')$, where m' is the number of edges of the completed cograph. We designed an algorithm that, in the same complexity, at each incremental step, computes a completion of the neighbourhood of the incoming vertex which is not only minimal but also of minimum cardinality [24]. It must be clear that this does not guarantee that the final completion output at the end of the algorithm is of minimum cardinality, this problem being NP-hard. But on the other hand, this feature is highly desirable in practice to improve the quality of the completion returned by the heuristic. I also extended this work to the general edition problem, where both addition and deletion of edges are allowed. I designed an incremental algorithm that computes at each incremental step a minimum number of modifications of the adjacencies between x and G such that $G + x$ is a cograph. The interest of using the general edition problem instead of the completion problem is that the number of modifications to be performed is by definition at most the number of additions, and can be much less in some cases. In particular, in [24], we show that very sparse random graphs are very likely to require $\Omega(n^2)$

in any completion, while there exists some edition with only $O(n)$ modifications. Obtaining a cograph with a lesser number of modifications is very valuable for the modelling purposes we pursue, as we get a closer approximation of the input complex network.

Finally, for the completion only algorithm, in the case where any minimal completion is suitable at each incremental step, we show that the complexity can be reduced to $O(n \log^2 n)$ [24], by using advanced dynamic data structures. With the remark above on the number of edges required in the completion of many very sparse graphs, this shows that for those graphs the resulting improvement of the running time of the algorithm is substantial, namely in the order of magnitude of $n/\log^2 n$.

Real-world networks as almost-cographs

This section presents results of minimal cograph edition for several real-world networks and the properties of the synthetic graphs obtained by using these edition results as input for the generation of *almost-cographs*, *i.e.* cographs altered by a number of modifications. This section is singular in the manuscript in the sense that it presents original results that have not been published anywhere else previously.

The algorithm I implemented is the minimal cograph edition algorithm described above, handling both deletions and additions of edges, giving a minimum cardinality edition of the neighbourhood of the new incoming vertex at each step of the incremental algorithm, with a randomly chosen incoming order on the vertices. The theoretical complexity of the algorithm is quadratic but it seems faster than expected in practice, both on real-world data and random graphs, which may suggest that its average complexity is lower than the one in the worst case. I ran the algorithm on 35 real-world networks coming from various contexts (see Table 3.1) and 52 Erdös-Rényi random graphs having between 1 000 and 1 000 000 vertices and mean degree between 2 and 80, used for comparison purposes. For each graph, I made 10 runs of the algorithm, each of them with a different random incoming order for the vertices. I used 11 machines with 2.4 GHz processors, divided in 16 to 64 cores, with 16 GB to 256 GB memory. Only two graphs, namely cit-Patents and LiveJournal, required more than 12h of computation per run, respectively 32h and 22h.

Results are shown[5] in Table 3.1 and Figure 3.3. There was a very limited variability of results over the 10 runs, even for real-world data. Table 3.1 shows the number of modifications performed, averaged over the 10 runs. For real-world networks, for 30 graphs out of 35, the difference between the minimum number of modifications and the mean over the 10 runs is less than 2.5%, for 4 of them it is less than 5.5% and only the 183 vertices graph, named *foodweb*, has a difference of about 12%. For all the 35 graphs, most of the modifications performed are deletions, more than 90% for 25 of them, between 80% and 90% for 9 of them, and about 70% for *foodweb*. This emphasises the interest of considering not only addition but also deletion of edges.

The number of modifications performed to obtain a cograph, expressed as a percentage of the number of edges in the input graph, are widely spread in a range between 12% and 93%. For 8 graphs, it is less than 35%, for 8 of them it is more than 75% and for the 19 remaining graphs it is comprised between 43% and 71%. Then, it appears that the proximity with the class of cographs highly depends on the graph considered. Very interestingly, the context where the graph comes from seems to play a preponderant role. The graphs that were found closest from being cographs come from engineered objects, such as web

[5]For random graphs, only the results for the graphs on 1 000 000 vertices and mean degree at most 20 are reported in Table 3.1. The results for smaller graphs are very similar to those obtained with 1 000 000 vertices and graphs with mean degree greater than 20 all received at least 95% of modifications (see Figure 3.3).

Context	Network	n	m	d°	%mod	#mod	#add	#del
WWW	in-2004	1 148 875	12 281 937	21.4	0.12	1 482 193	259 959	1 222 234
WWW	cnr-2000	227 058	2 187 201	19.3	0.19	408 278	56 988	351 290
PROTEIN	reactome	5 973	145 778	48.8	0.22	31 961	4 723	27 238
SOFTWARE	jdk	6 434	53 658	16.7	0.29	15 665	2 671	12 994
SOFTWARE	jung-j	6 120	50 290	16.4	0.29	14 576	2 709	11 867
WWW	eu-2005	835 044	15 718 784	37.7	0.29	4 631 218	469 270	4 161 948
CO-AUTHOR	ca-GrQc	4 158	13 422	6.5	0.34	4 500	560	3 940
CO-AUTHOR	ca-HepPh	11 204	117 619	21.0	0.34	40 572	2 369	38 203
SPECIES	foodweb	183	2 434	26.6	0.43	1 045	310	735
CO-AUTHOR	dblp	317 080	1 049 866	6.6	0.45	472 354	45 215	427 139
WORD-REL.	wordnet	145 145	656 230	9.0	0.48	316 288	24 742	291 546
COMMUNIC.	wiki-Talk	2 388 953	4 656 682	3.9	0.49	2 302 468	17 226	2 285 242
CO-SOLD	amazon	334 863	925 872	5.5	0.49	453 288	74 719	378 569
CO-AUTHOR	ca-CondMat	21 363	91 286	8.6	0.52	47 895	3 977	43 917
RANDOM	ER-Gnm_1M-2	796 208	958 827	2.4	0.52	502 798	87 364	415 434
CO-AUTHOR	ca-HepTh	8 638	24 806	5.7	0.54	13 349	1 296	12 053
INTERNET	as2000	6 474	12 572	3.9	0.54	6 744	439	6 304
ROAD	roadNet-TX	1 351 137	1 879 201	2.8	0.54	1 012 350	193 074	819 276
INTERNET	as-caida2007	26 475	53 381	4.0	0.55	29 123	1 457	27 666
CO-AUTHOR	ca-AstroPh	17 903	196 972	22.0	0.59	115 684	5 840	109 844
INTERNET	topology	34 761	107 720	6.2	0.61	65 642	4 437	61 205
RANDOM	ER-Gnm_1M-3	940 987	1 494 643	3.2	0.63	945 250	125 046	820 204
INTERNET	as-skitter	1 694 616	11 094 209	13.1	0.64	7 112 672	690 571	6 422 101
CO-OCCUR	bible-names	1 707	9 059	10.6	0.67	6 100	372	5 728
PROTEIN	figeys	2 217	6 418	5.8	0.67	4 271	182	4 089
CITATION-SCI.	cora	23 166	89 157	7.7	0.68	60 992	5 194	55 797
SOCIAL	youtube	1 134 890	2 987 624	5.3	0.69	2 058 540	91 168	1 967 372
CO-ACTOR	actor-col.	374 511	15 014 839	80.2	0.71	10 674 141	154 844	10 519 297
P2P-CONNECT.	p2p-Gnutella	62 561	147 878	4.7	0.71	104 334	5 916	98 418
RANDOM	ER-Gnm_1M-4	980 191	1 999 203	4.1	0.71	1 414 699	150 710	1 263 989
CITATION-SCI.	citeseer	365 154	1 721 981	9.4	0.75	1 296 670	75 661	1 221 009
CITATION-PAT.	cit-Patents	3 764 117	16 511 740	8.8	0.76	12 551 073	739 054	11 812 018
SOFTWARE	linux	30 817	213 208	13.8	0.77	163 177	10 784	152 393
SOCIAL	LiveJournal	3 997 962	34 681 189	17.4	0.78	27 122 844	1 336 779	25 786 065
CITATION-SCI.	cit-HepTh	27 400	352 021	25.7	0.79	278 632	19 123	259 509
RANDOM	ER-Gnm_1M-6	997 479	2 999 988	6.0	0.79	2 381 169	182 870	2 198 299
CITATION-SCI.	cit-HepPh	34 401	420 784	24.5	0.81	341 515	20 521	320 994
RANDOM	ER-Gnm_1M-8	999 684	3 999 999	8.0	0.84	3 362 818	201 406	3 161 412
RANDOM	ER-Gnm_1M-10	999 952	5 000 000	10.0	0.87	4 346 649	210 128	4 136 521
RANDOM	ER-Gnm_1M-15	1 000 000	7 500 000	15.0	0.91	6 822 082	222 329	6 599 753
SOCIAL	orkut	3 072 441	117 185 083	76.3	0.91	107 125 003	2 992 486	104 132 518
RANDOM	ER-Gnm_1M-20	1 000 000	10 000 000	20.0	0.93	9 302 868	224 073	9 078 795
WORD-REL.	Thesaurus	23 132	297 094	25.7	0.93	274 847	3 697	271 150

Table 3.1: Results of minimal cograph edition for some real-world networks (white rows) and some Erdös-Rényi random graphs (grey rows). The columns give the number of vertices (n), the number of edges (m) and the mean degree (d°) of the input graph, as well as the quantity of modifications performed by the algorithm, expressed both in proportion of the number of edges in the input graph (%mod) and in absolute number (#mod), the number of these modifications that are additions of edges (#add) and the number of deletions of edges (#del).

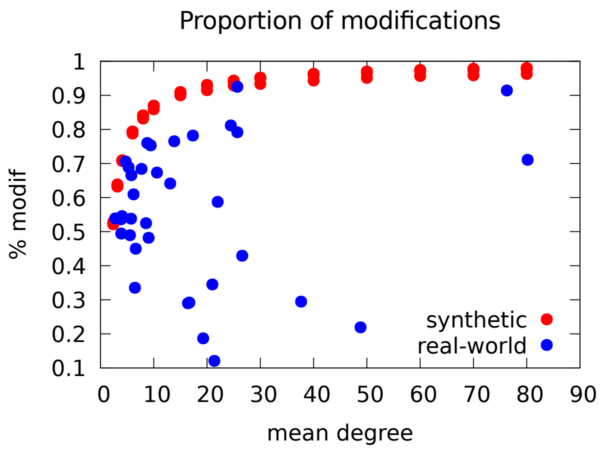


Figure 3.3: Proportion of modifications made by the minimal cograph edition algorithm (y-axis) as a function of the mean degree of the input graph (x-axis), for the real-world networks (real-world) and the Erdös-Rényi random graphs (synthetic) of Table 3.1.

graphs and graphs of dependency between classes in object oriented software, and also include scientific co-authorship graphs and the interactions between proteins of the *reactome* graph. On the opposite, scientific citations and online social networks were found quite far from being cographs, while spatially constrained graphs such as the Internet topology and the road network of Texas lay in the middle, between 54% and 64%.

For random graphs, the proportion of modifications to perform to obtain a cograph is entirely driven by their mean degree, independently of their size, as shown in Figure 3.3. This percentage rises very quickly with the mean degree and reach 79% for mean degree 6. Almost all real-world networks give results that are much lower to what is expected for their mean degree, denoting the well-known specificity of their structure. More interestingly, the graphs that were found relatively close from being cographs (those with less than 35% of modifications) have mean degrees that cover a wide range, between 6.5 and 48. This shows that their proximity with cographs is clearly a specificity of them and of the context they come from.

Using the results of edition for real-world networks, I tested the idea of modelling complex networks as almost-cographs, *i.e.* cographs altered by a number of modifications (both addition and deletion of edges). To this purpose, for each graph, I used the result of the first of the ten runs. I started from the cograph returned by the minimal edition algorithm and I randomly deleted from it nb_{add} edges and randomly added nb_{del} edges, where nb_{add} is the number of edges that were added by the minimal edition algorithm and nb_{del} the number of edges that were deleted. Using only one particular run of the algorithm avoid to bump on the question of defining what is the mean cograph returned over the 10 runs and is justified by the low variability of the results over the 10 runs. For each real-world graph, I generated 5 random almost-cographs in this way, as well as 5 configuration models using the degree distribution of the original graph as input and 5 Erdös-Rényi graphs using the $G_{n,m}$ model with the number of edges in the original graph as input. In all these three models, the number of edges, and so the density and the mean degree, is guaranteed to be exactly equal to the one of the original graph. The synthetic graphs generated by the three models were then compared to the original graph in terms of mean distance, global clustering coefficient (Table 3.3 also gives the number of triangles) and degree distribution. Using 5 random generation rounds allowed to check that the variability on the properties of the obtained graphs is very limited: Tables 3.2 and 3.3 give the mean value of each parameter for the 5 generated graphs, while Figures 3.5 and 3.6 show the degree distribution of the first generated graph.

input graph (REAL) for the models						mean distance			
Context	Network	n	m	d°	%edit	REAL	ACG	CM	ER
WWW	in-2004	1 148 875	12 281 937	21.38	0.12	8.82	5.31	3.51	4.85
WWW	cnr-2000	227 058	2 187 201	19.27	0.19	9.27	4.70	3.28	4.50
PROTEIN	reactome	5 973	145 778	48.81	0.22	4.21	3.01	2.80	2.66
SOFTWARE	jdk	6 434	53 658	16.68	0.29	2.12	2.12	2.66	3.43
SOFTWARE	jung-j	6 120	50 290	16.43	0.29	2.11	2.08	2.66	3.43
WWW	eu-2005	835 044	15 718 784	37.65	0.29	4.64	3.68	3.13	4.03
CO-AUTHOR	ca-GrQc	4 158	13 422	6.46	0.34	6.05	5.67	4.08	4.68
CO-AUTHOR	ca-HepPh	11 204	117 619	21	0.34	4.67	3.91	3.19	3.39
SPECIES	foodweb	183	2 434	26.60	0.43	2.15	1.95	2.08	1.87
CO-AUTHOR	dblp	317 080	1 049 866	6.62	0.45	6.79	7.42	5.39	6.92
WORD-REL.	wordnet	145 145	656 230	9.04	0.48	5.53	5.62	4.31	5.65
COMMUNIC.	wiki-Talk	2 388 953	4 656 682	3.90	0.49	3.90	5.58	4.00	10.90
CO-SOLD	amazon	334 863	925 872	5.53	0.49	11.95	8.14	6.36	7.63
CO-AUTHOR	ca-CondMat	21 363	91 286	8.55	0.52	5.35	5.14	4.27	4.89
CO-AUTHOR	ca-HepTh	8 638	24 806	5.74	0.54	5.95	5.69	4.69	5.38
INTERNET	as2000	6 474	12 572	3.88	0.54	3.71	4.71	3.93	6.58
ROAD	roadNet-TX	1 351 137	1 879 201	2.78	0.54	415.71	16.29	17.41	13.68
INTERNET	as-caida2007	26 475	53 381	4.03	0.55	3.88	5.29	4.02	7.43
CO-AUTHOR	ca-AstroPh	17 903	196 972	22	0.59	4.19	3.64	3.36	3.52
INTERNET	topology	34 761	107 720	6.20	0.61	3.77	4.82	3.72	5.94
INTERNET	as-skitter	1 694 616	11 094 209	13.09	0.64	5.07	5.00	3.73	5.84
CO-OCCUR	bible-names	1 707	9 059	10.61	0.67	3.38	3.43	3.13	3.42
PROTEIN	figeys	2 217	6 418	5.79	0.67	3.84	4.07	3.60	4.58
CITATION-SCI.	cora	23 166	89 157	7.70	0.68	5.86	5.12	4.38	5.16
SOCIAL	youtube	1 134 890	2 987 624	5.27	0.69	5.28	6.62	4.46	8.58
CO-ACTOR	actor-col.	374 511	15 014 839	80.18	0.71	3.71	3.31	3.06	3.24
P2P-CONNECT.	p2p-Gnutella	62 561	147 878	4.73	0.71	5.94	7.21	5.82	7.27
CITATION-SCI.	citeseer	365 154	1 721 981	9.43	0.75	6.47	5.73	4.65	5.95
CITATION-PAT.	cit-Patents	3 764 117	16 511 740	8.77	0.76	8.15	7.24	6.07	7.22
SOFTWARE	linux	30 817	213 208	13.84	0.77	3.25	3.35	3.09	4.22
SOCIAL	LiveJournal	3 997 962	34 681 189	17.35	0.78	5.57	5.68	4.56	5.65
CITATION-SCI.	cit-HepTh	27 400	352 021	25.69	0.79	4.28	3.46	3.24	3.51
CITATION-SCI.	cit-HepPh	34 401	420 784	24.46	0.81	4.33	3.63	3.41	3.63
SOCIAL	orkut	3 072 441	117 185 083	76.28	0.91	4.22	3.86	3.46	3.86
WORD-REL.	Thesaurus	23 132	297 094	25.69	0.93	3.49	3.44	3.39	3.45

Table 3.2: Mean distances: comparison between the almost-cograph model (ACG), the configuration model (CM) and Erdös-Rényi random graphs (ER), using the real-world networks (REAL) of Table 3.1 as input for the models.

input graph (REAL) for the models						global CC				#triangles			
Context	**Network**	**n**	**m**	**d°**	**%edit**	**REAL**	**ACG**	**CM**	**ER**	**REAL**	**ACG**	**CM**	**ER**
WWW	in-2004	1 148 875	12 281 937	21.38	0.12	0.096	0.094	0.016	0.000	452 887 093	420 782 426	60 610 327	1 607
WWW	cnr-2000	227 058	2 187 201	19.27	0.19	0.009	0.009	0.006	0.000	17 701 650	16 200 763	5 874 109	1 206
PROTEIN	reactome	5 973	145 778	48.81	0.22	0.606	0.592	0.061	0.008	4 187 395	3 621 475	383 441	19 399
SOFTWARE	jdk	6 434	53 658	16.68	0.29	0.011	0.012	0.022	0.003	194 842	190 054	83 661	784
SOFTWARE	jung-j	6 120	50 290	16.43	0.29	0.011	0.013	0.023	0.003	182 139	191 201	78 372	729
WWW	eu-2005	835 044	15 718 784	37.65	0.29	0.015	0.015	0.007	0.000	193 930 463	168 938 889	65 833 626	8 912
CO-AUTHOR	ca-GrQc	4 158	13 422	6.46	0.34	0.629	0.619	0.010	0.002	47 779	38 833	737	46
CO-AUTHOR	ca-HepPh	11 204	117 619	21	0.34	0.659	0.775	0.056	0.002	3 357 890	2 615 794	251 807	1 558
SPECIES	foodweb	183	2 434	26.60	0.43	0.332	0.345	0.246	0.145	11 292	9 924	5 344	3 113
CO-AUTHOR	dblp	317 080	1 049 866	6.62	0.45	0.306	0.342	0.000	0.000	2 224 385	1 324 081	1 470	54
WORD-REL.	wordnet	145 145	656 230	9.04	0.48	0.096	0.104	0.002	0.000	1 144 648	555 996	23 980	122
COMMUNIC.	wiki-Talk	2 388 953	4 656 682	3.90	0.49	0.002	0.000	0.024	0.000	9 203 514	42 055	47 172 837	10
CO-SOLD	amazon	334 863	925 872	5.53	0.49	0.205	0.164	0.000	0.000	667 129	390 607	195	31
CO-AUTHOR	ca-CondMat	21 363	91 286	8.55	0.52	0.262	0.212	0.002	0.000	171 051	70 337	1 606	105
CO-AUTHOR	ca-HepTh	8 638	24 806	5.74	0.54	0.281	0.232	0.003	0.001	27 869	14 123	274	30
INTERNET	as2000	6 474	12 572	3.88	0.54	0.010	0.004	0.018	0.001	6 584	1 066	5 808	11
ROAD	roadNet-TX	1 351 137	1 879 201	2.78	0.54	0.060	0.080	0.000	0.000	81 195	131 132	1	3
INTERNET	as-caida2007	26 475	53 381	4.03	0.55	0.007	0.003	0.016	0.000	36 365	3 241	42 111	11
CO-AUTHOR	ca-AstroPh	17 903	196 972	22	0.59	0.318	0.285	0.010	0.001	1 350 014	577 645	42 342	1 793
INTERNET	topology	34 761	107 720	6.20	0.61	0.049	0.057	0.041	0.000	554 749	183 288	253 557	40
INTERNET	as-skitter	1 694 616	11 094 209	13.09	0.64	0.005	0.006	0.005	0.000	28 769 842	14 923 142	19 522 290	356
CO-OCCUR	bible-names	1 707	9 059	10.61	0.67	0.162	0.093	0.048	0.006	19 936	3 649	4 613	200
PROTEIN	figeys	2 217	6 418	5.79	0.67	0.008	0.024	0.060	0.003	897	752	4 584	35
CITATION-SCI.	cora	23 166	89 157	7.70	0.68	0.117	0.074	0.003	0.000	78 791	22 751	1 829	69
SOCIAL	youtube	1 134 890	2 987 624	5.27	0.69	0.006	0.003	0.005	0.000	3 056 386	169 712	1 943 389	23
CO-ACTOR	actor-col.	374 511	15 014 839	80.18	0.71	0.166	0.193	0.006	0.000	346 728 049	93 575 410	11 423 272	85 861
P2P-CONNECT.	p2p-Gnutella	62 561	147 878	4.73	0.71	0.004	0.017	0.000	0.000	2 024	4 249	198	18
CITATION-SCI.	citeseer	365 154	1 721 981	9.43	0.75	0.050	0.035	0.001	0.000	1 350 310	291 958	17 047	142
CITATION-PAT.	cit-Patents	3 764 117	16 511 740	8.77	0.76	0.067	0.050	0.000	0.000	7 514 922	3 036 162	1 403	119
SOFTWARE	linux	30 817	213 208	13.84	0.77	0.003	0.004	0.013	0.000	170 862	84 726	340 674	437
SOCIAL	LiveJournal	3 997 962	34 681 189	17.35	0.78	0.125	0.233	0.000	0.000	177 820 130	76 031 974	297 064	890
CITATION-SCI.	cit-HepTh	27 400	352 021	25.69	0.79	0.120	0.055	0.011	0.001	1 478 698	259 977	123 116	2 830
CITATION-SCI.	cit-HepPh	34 401	420 784	24.46	0.81	0.146	0.055	0.004	0.001	1 276 859	225 011	38 946	2 425
SOCIAL	orkut	3 072 441	117 185 083	76.28	0.91	0.041	0.022	0.001	0.000	627 584 181	70 745 008	8 226 644	74 184
WORD-REL.	Thesaurus	23 132	297 094	25.69	0.93	0.040	0.006	0.016	0.001	409 174	15 561	152 803	2 814

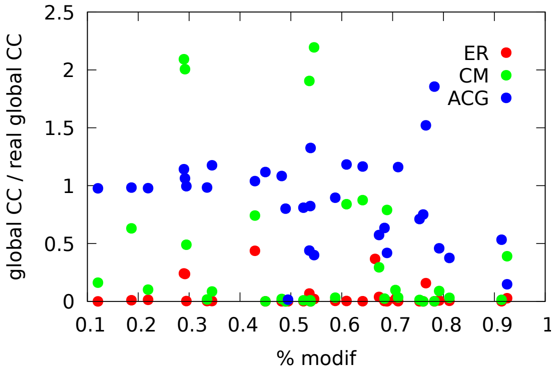
Table 3.3: Clustering coefficient and number of triangles: comparison between the almost-cograph model (ACG), the configuration model (CM) and Erdös-Rényi random graphs (ER), using the real-world networks (REAL) of Table 3.1 as input for the models.

As expected, all three models correctly reproduce the property of short distances in the original graphs (see Table 3.2) since they all contain a part of randomness which induces this property. Concerning the clustering coefficient and the number of triangles, as shown in Table 3.3 and in Figure 3.4, for almost all graphs, the values of the almost-cograph model are in the same order of magnitude as those of the original graph, and are even very close to the actual value for graphs that are close from being cographs (less than 35% of modifications). The configuration model has a larger variability of these two parameters and is often some orders of magnitude below the actual value, while, as expected, Erdös-Rényi graphs have values even lower by some more orders of magnitude.

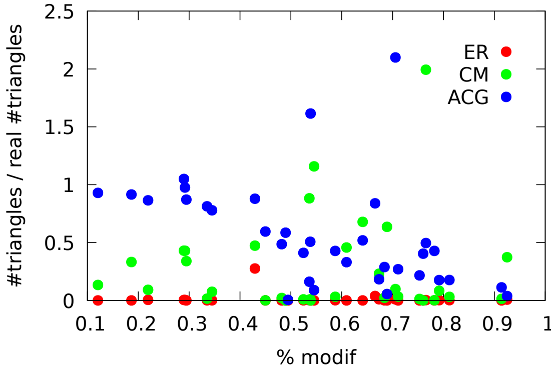
Figures 3.5 and 3.6 show the degree distributions obtained with the three models for a selection of real-world networks spanning the range of the percentage of modifications performed in the edition problem. The configuration model and Erdös-Rényi random graphs are two opposite limit cases: the degree distribution of the configuration model is by definition exactly the same as the one of the original graph, since this is the parameter given as input to the model, and the degree distribution of Erdös-Rényi graphs, which is by definition homogeneous, always completely misses the heavy tail of the real distribution. For the almost-cograph model, the situation is different. The very satisfying point is that it always succeeds in capturing the heavy tail of the degree distributions, which is the main targeted criteria. On the other hand, the shape of the beginning of the distribution is sometimes very well reproduced, even for graphs having about 50% of modifications, like e.g. *amazon*, and sometimes altered, even for graphs close from being cographs, like e.g. *reactome* and *ca-HepPh*. As one could expect, for graphs which are very far from being cographs, such as *LiveJournal* and *orkut*, which obtained a score of more than 80% of modifications in the edition algorithm, the degree distribution is strongly altered. But even in this case, the heavy tail of the distribution is mainly conserved. Even more surprisingly, the almost-cograph model sometimes quite satisfying reproduces the whole shape of the distribution even for graphs not close from being cographs, see e.g. the Internet topology of the *skitter* graph (64% of modifications) and the citation networks *cora* (68% of modifications) and *Patents* (76% of modifications).

Conclusion. The approach based on representing complex networks as approximation of strongly structured graphs is very appealing. The results of minimal cograph edition of real-world networks presented here show that some of these networks are indeed close from having a cograph structure. Exploiting this proximity to cographs gives very interesting results for modelling purposes, even in some cases for networks that are not so close from being cographs. Moreover, it appears that the property of being close to a cograph highly depends on the network considered and on the context it comes from, therefore asking for the development of methods of comparison to other strongly structured classes of graphs that may suit better to some specific contexts.

Concerning random generation, the very satisfying point is that the almost-cograph model produces graphs that have both a high local density and a heterogeneous degree distribution, which is currently the main challenge of the domain of complex network modelling. These results are then very promising. Nevertheless, there is still one important goal to achieve in order to make the almost-cograph model a complete model: it needs to be enhanced with genericity. To this purpose, the cotrees given as input to the generation process should be synthetic structures generated by the model itself, exactly as generic power law can be given as input to the configuration model. One of the key interest of the almost-cograph model is that this goal is technically reachable: there exist methods to randomly generate trees satisfying certain properties [2]. Therefore, the next step is to systematically study the shape of cotrees resulting from edition



(a) global clustering coefficient



(b) number of triangles

Figure 3.4: Global clustering coefficient (plot (a), y-axis) and number of triangles (plot (b), y-axis) as a function of the number of modifications made in the minimal edition algorithm (x-axis), for the almost-cograph model (ACG), the configuration model (CM) and Erdös-Rényi random graphs (ER). Values on the y-axis are normalised by the value in the graph given as input to the model.

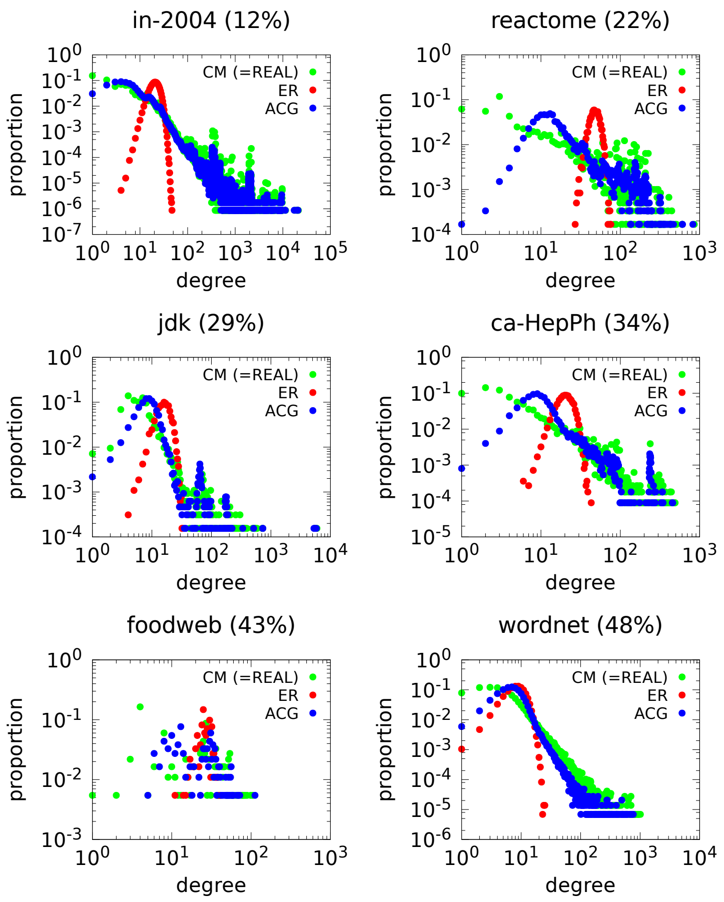


Figure 3.5: Comparison between the degree distribution of the almost-cograph model (ACG), the configuration model (CM) and Erdös-Rényi random graphs (ER), for some selected real-world networks of Table 3.1 given as input to the models. More networks on Figure 3.6.

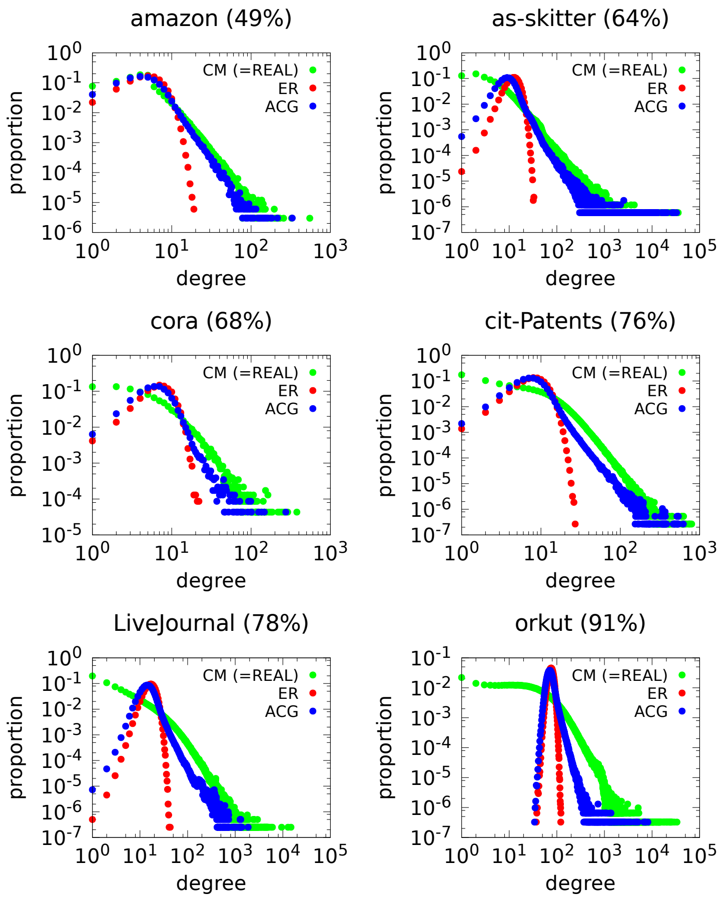


Figure 3.6: Comparison between the degree distribution of the almost-cograph model (ACG), the configuration model (CM) and Erdös-Rényi random graphs (ER), for some selected real-world networks of Table 3.1 given as input to the models. More networks on Figure 3.5.

of real-world networks, to determine their main properties and to design processes to uniformly randomly generate trees with these properties.

The work presented here also points out that there may be room to improve further the use of minimal cograph edition algorithms as heuristics for minimum edition. For example, the results of cograph edition obtained here for some real-world networks are not as good as the one presented for quasi-threshold edition, for the same networks, in [10]. Nevertheless, the minimum number of modifications for cograph edition must be no more than the minimum number of modifications for quasi-threshold edition, since the former class properly contains the latter. One reason for this may be that the algorithm presented here only considers each vertex once, while [10] continues the algorithm by removing and inserting again each vertex until no improvement is possible for any vertex of the graph. This possibility, and others, should be explored in order to improve the quality of results returned by heuristics. Another appealing question is about the complexity of the minimal cograph edition algorithm. Is it possible to prove that the average complexity is better than the quadratic worst-case complexity, as suggested by experiments? Or is it even possible to design an algorithm having a sub-quadratic complexity in the worst-case? All these questions are open for cographs ans should also be considered for other classes of graphs.

Chapter 4

Efficient encodings of graphs

Because of their size, any treatment that is to be performed on complex networks, either statistic, algorithmic or for exploration, requires to use a computer to automatically perform all or part of the necessary actions. When one wants to do so, there are two kinds of limitation that immediately come up. The first one is about the space necessary to store the graph into the machine, as its memory is finite, and the other one is about the time necessary to perform the desired treatment, as beyond a certain waiting time the results of most treatments become unuseful. Because the objects to be manipulated are very large, sometimes even huge, these two limitations are critical. Usually, the first one that comes in mind to theorists is the one concerning time and the efficiency of algorithmic treatments. But practitioners know that, even though execution time is an important issue in most cases, for very large instances, space can turns out to be the first limiting factor.

One reason for this is that, unlike the one concerning time, the constraint on space is strict: it may be acceptable to wait for a result slightly longer, but it is simply impossible to extend the memory of the machine. Therefore, if the graph does not fit entirely into the memory, it is impossible to process it by standard methods. Let me emphasise that we talk here about random-access memory (RAM), which can be accessed very efficiently during the execution of a program, but which is sharply limited, say to some dozen (sometimes hundreds) of gigabytes. This is not the case of the capacity of storage on hard disk drives (HDD) which, nowadays, can store any huge graph data available very easily. The problem is that using this capacity of storage as a replacement of the memory is totally unpractical: accessing the hard drive repeatedly during the execution of a program is prohibitive for its execution time. To deal with this fact, one approach consists in developing *ad-hoc* algorithmic techniques that allow to load partially the graph into memory and to process it piece by piece. This is the purpose of online and streaming algorithms. Of course, the possibility to proceed in this way highly depends on the considered algorithmic problem and often allows to obtain only approximated solutions.

Another approach to deal with the restricted space available in memory, since it is not extensible, is to compress the graph, *i.e.* encode it using less space, so that it fits in memory. This motivates a rich trend of works on compression techniques specific to graphs (see [74] for example). As one can guess from the discussion above, in order to be useful from a practical point of view, these compression techniques must satisfy one additional requirement: accessing the compressed data should not penalise the running time of the algorithm. In particular, it must be possible to access the data without decompressing it, or doing so only partially and without significant extra-cost.

The way the data needs to be accessed depends on the task to be performed. Nevertheless, there are two atomic operations that constitute the base of most algorithmic treatments: listing the set of neighbours of one given vertex x, which is called the *neighbourhood query*, and deciding whether two given vertices x

and y are adjacent, which is called the *adjacency query*. In this chapter, we concentrate on the neighbourhood query, which is probably the most widely used query in graph algorithms and which is also useful for exploring and visualising data. Our general goal is to design representations of graphs that are as compact as possible and that allow to answer neighbourhood queries on vertex x in optimal time, that is $O(d)$ time, where d is the degree of x. In this way, the graph can be stored using a lesser amount of space in memory, in a way which is completely transparent for all algorithms based on neighbourhood queries: the list of neighbours of a vertex can be accessed in the same complexity as if the graph was stored by adjacency lists.

Classes of graphs admitting special encodings [16, 17]

We start by studying some graph classes known to admit very compact encodings, namely *permutation graphs* and *interval graphs*, to check whether these encodings allow to efficiently answer neighbourhood queries. Interval graphs are the intersection graphs of intervals of the real line and permutation graphs are the intersection graphs of segments having their two extremities on two distinct parallel lines (see [11] for more formal definitions). Interestingly, the intersection models of these two classes provide an $O(n)$-space encoding of the graph that allows to answer adjacency queries in $O(1)$ time. Unfortunately, these natural representations do not directly provide a way to answer neighbourhood queries efficiently. The neighbourhood of one vertex x may be scattered in the intersection model in such a way that one needs to scan almost all the models in order to find the neighbours of x, therefore using a time proportional to the size of the model, that is $O(n)$ time. This is rather surprising as interval graphs and permutation graphs are known to have usually all nice algorithmic properties that one can expect.

This motivated our deeper attention to these classes and in [16], we show that for both of them, it is possible to extend their natural representation by an additional $O(n)$-space structure in order to answer neighbourhood queries in $O(d)$ worst-case time, where d is the degree of the considered vertex. This extension relies on an advanced data structure to answer *maximum range queries* in constant time, *i.e.* to provide the maximum value of a function $f(x)$ for x in an interval of a fixed linear ordering. Using this to query the models of interval graphs or permutation graphs, which are made of linear orderings of their vertices, allows to find each neighbour of a given vertex x in constant worst-case time without scanning the entire model.

In [17], we also show that the existence of a good encoding for a class of graphs not only allows to store the graphs in an efficient way but can also improve the complexity of some algorithmic problems, even for problems of very low complexity. This happens in a quite surprising way for the *Breadth First Search* (BFS for short, see [15] for a definition) of *trapezoid graphs*, for which we designed an $O(n)$-time algorithm. Trapezoid graphs are the intersection graphs of trapezoid whose bases lay on two parallel lines (see [11]) and they properly contain both interval graphs and permutation graphs.

The above statement may first sound strange or even incorrect as BFS is usually thought of as a traversal of all the edges of the graph, which necessarily results in an $\Omega(m)$ complexity. The ambiguity comes from the fact that in the case of BFS, we usually do not distinguish between the algorithm and the problem it solves. This is misleading as algorithmic problems should not be defined by what must be done by the algorithm but only by the result expected from its execution. We therefore adopt the following definition of the BFS problem, based on the classical features expected from a BFS. Given a graph G and a vertex $x \in V(G)$, BFS returns:

1. a BFS tree T rooted at x

2. the distance from each vertex y to the source x
3. for each vertex y, the list of neighbours of y that are one level upper in T (*i.e.* closer to the root x)

Note that Feature 3 provides a way to retrieve efficiently all the shortest paths from any vertex y to x, as the possible next steps on such a path are precisely the neighbours of y that are one level upper in T. For a graph G, there are many different BFS trees, depending on the way ties are broken between the unvisited neighbours of the current vertex along the execution of the algorithm. An important characteristic of the $O(n)$-time BFS algorithm we designed is that it takes as input any trapezoid model of the graph and can produce as output any BFS tree, not only a particular tree depending on the model given as input. To this purpose, the input must include, in addition to G and x, an arbitrary linear ordering σ of the vertices of G according to which ties are broken along the algorithm. Different BFS trees can then be obtained using different linear orderings.

On the technical side, it is worth mentioning that in order to achieve the $O(n)$ time complexity, the lists of neighbours associated to each vertex y are not stored independently. They are actually interval sub-lists of a global list L containing all the vertices at one given level of the tree. Therefore, they are encoded by two pointers toward respectively their first element and their last element in L, which requires only $O(1)$ space per vertex y.

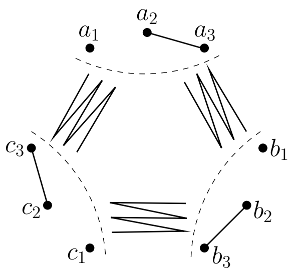
This shows that even for problems that are solvable in very low complexity, efficient encodings can be used to lower further their complexity. This perspective is particularly interesting for complex networks, for which even low-complexity algorithms can be difficult to apply because of the size of the considered graphs. This is a strong motivation for finding efficient encodings of complex networks that have nice combinatorial properties and that are able to boost classical algorithms.

Two general encodings: contiguity and linearity

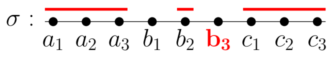
In the rest of this chapter, we study two graph encodings that are defined thanks to one or more linear orderings of the vertices of the graph, with the purpose of making the neighbourhood of each of the vertices of the graph appear as a union of a few intervals. Similar ideas are known to be efficient in practice and some compression techniques dedicated to the web graph or social networks [3, 12, 54] are based on linear orderings of the vertices having nice properties with regard to the neighbourhoods of the vertices. Some of these techniques indeed exploit the intervals made by the consecutive neighbours of one vertex [4, 8] for compression.

Assume that a graph G admits an order σ on its vertices such that the neighbourhood of each vertex x of G is an interval in σ. In this way, one can store the order σ on the vertices of G and assign two pointers to each vertex: the first one toward its first neighbour in σ and the second one toward its last neighbour in σ. This provides an $O(n)$-space encoding of the graph that allows to answer adjacency queries in $O(d)$ time, by simply listing the vertices appearing in σ between its first and second pointer.

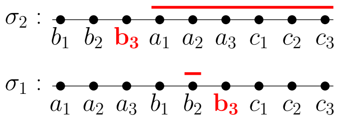
Of course, such an order on the vertices of G does not exist for all graphs G. Then, a natural way to relax the constraints of the problem so that it admits a solution for a larger class of graphs is to allow the neighbourhood of each vertex to be split in at most k intervals in order σ. The minimum value of k which makes it possible to encode the graph G in this way is a parameter called *contiguity* [34] and denoted by $cont(G)$. Another natural way of generalisation is, instead of using k intervals taken in the same order, to use k orders $\sigma_1, \ldots, \sigma_k$ on the vertices of G and to choose, for each vertex x, one interval in each of the k orders, in such a way that the neighbourhood of x is the union of these k intervals. This defines a parameter called the *linearity* of G, which we introduced in [16] and which



(a) graph G


σ :
a_1 a_2 a_3 b_1 b_2 $\mathbf{b_3}$ c_1 c_2 c_3

(b) encoding by contiguity 3


σ_2 :
b_1 b_2 $\mathbf{b_3}$ a_1 a_2 a_3 c_1 c_2 c_3
σ_1 :
a_1 a_2 a_3 b_1 b_2 $\mathbf{b_3}$ c_1 c_2 c_3

(c) encoding by linearity 2

Figure 4.1: Example of a graph G (a), an encoding of G by contiguity 3 (b) and an encoding of G by linearity 2 (c). The graph G is such that $\forall x, y \in \{a, b, c\}, \forall i, j \in \{1, 2, 3\}$, x_i is adjacent to y_j. In the encoding by contiguity, the neighbourhood of vertex b_3 is split into 3 intervals in order σ, and no other vertex has its neighbourhood split in more than 3 intervals: this is an encoding by contiguity 3. In the encoding by linearity, the neighbourhood of vertex b_3 is obtained as the union of one interval in order σ_1 and one interval in order σ_2, and the same holds for all other vertices of G: this is an encoding by linearity 2.

is denoted $lin(G)$. An example of encodings of a graph using contiguity and linearity is given in Figure 4.1. The additional flexibility offered by linearity (using k orders instead of just 1) results in a greater power of encoding, in the sense that if a graph G admits an encoding by contiguity k, using one linear order σ and at most k intervals for each vertex, it is straightforward to obtain an encoding of G by linearity k: take k copies of σ and assign to each vertex one of its k intervals in each of the k copies of σ.

For each of contiguity and linearity, there are actually two variants, respectively named *open* and *closed*, depending on whether one considers the *open neighbourhood* or the *closed neighbourhood* of vertices (including the vertex itself). But these two variants are equivalent up to a multiplicative constant. Therefore, unless specified otherwise, we do not distinguish these two variants as all the results mentioned below hold for both of them.

Only little is known about contiguity and linearity, except that the problem of deciding whether a graph G has closed contiguity at most k is NP-complete for any fixed $k \geq 2$. Note that despite the fact that it was not formally proven, there is a strong presumption that the corresponding decision problem for linearity is NP-complete as well. On the positive side, the class of graphs having closed contiguity exactly 1 (or equivalently closed linearity exactly 1) and the class of graphs having open contiguity exactly 1 (or equivalently open linearity exactly 1) have been characterised as being respectively the class of *proper interval graphs*, a subclass of interval graphs, and *biconvex graphs*, a subclass of bipartite permutation graphs. Knowing this, and since their intersection models can be defined thanks to linear orderings of their vertices, one could expect that arbitrary interval graphs and arbitrary permutation graphs also have bounded contiguity and linearity. Rather surprisingly, in [16], we show that on the contrary, for both classes, the linearity (and so the contiguity) can be up to $\Omega(\log n / \log\log n)$.

This apparently negative result is also encouraging. It does not mean that contiguity and linearity are inefficient. It simply shows that the hierarchy of difficulty of encoding captured by these parameters is different from the classical hierarchy offered by interval graphs, permutation graphs and their generalisations such as k-trapezoid graphs and comparability graphs of dimension k (see [11] for definitions). This is interesting in itself as they may provide a different point of view on the structure of graphs with regard to difficulty of encoding, which may be more appropriate for some cases. In addition, note that the lower bound of $\Omega(\log n / \log\log n)$ on the worst case value of the two parameters is not very high. If it was the actual worst possible value, then the encoding provided by contiguity and linearity would still be fairly efficient, suggesting that they may be powerful parameters.

Worst-case contiguity of cographs [18, 19]

In our proof that the linearity (and so the contiguity) of interval graphs and permutation graphs can be up to $\Omega(\log n / \log\log n)$, one interesting and surprising fact is that the subfamily of graphs we exhibit is the same for both classes. It belongs to the class of *quasi-threshold graphs* (see [11] for a definition), which is properly included in interval graphs and permutation graphs. If simple classes like quasi-threshold graphs can have an unbounded contiguity and linearity, they can help to understand what makes the value of these parameters high. To that purpose, we studied the contiguity of a class more general than quasi-threshold graphs and included in the class of permutation graphs (but not in the class of interval graphs): the class of *cographs*. Cographs admit a nice representation by a uniquely-defined rooted tree, named the *cotree*, whose leaves are the vertices of the graph and whose internal nodes carry the information that encodes adjacency relationships between the vertices (see [11] for details).

The question we addressed is to know what is the worst-case contiguity of a cograph on n vertices. In [19], we show that it can be greater than $\Omega(\log n/\log\log n)$ by exhibiting a family of cographs that have contiguity $\Omega(\log n)$. We also show that this lower bound is tight as we prove an upper bound of $O(\log n)$ on the contiguity of any cograph. This bound is proven by using a structural parameter on the cotree of G. This parameter, called the *rank*, is the maximum height of a complete binary tree included, as a minor[1], in the cotree of G. Since a complete binary tree of height h has 2^h leaves, it follows that the rank of T cannot exceed $\log n$. In [19], we prove that for any cograph G, $cont(G) \leq 2\,rank(T) + 1$, which yields the $O(\log n)$ upper bound on the contiguity of G.

Going further, we prove that the rank of the cotree T of G does not provide only an upper bound on the contiguity of G but is also equivalent to it up to a multiplicative constant. This is very interesting as it provides a structural characterisation of what makes the contiguity of a cograph high. Using this equivalence between the rank of T and the contiguity of G, in [18], we design a linear time ($O(n)$ time) algorithm that, given the cotree of a cograph, computes an approximation of its contiguity within a constant factor of the optimal.

Since the linearity of a graph cannot exceed its contiguity, the $O(\log n)$ upper bound we obtain for the contiguity of a cograph on n vertices also holds for its linearity. However, the $\Omega(\log n)$ lower bound does not hold for the linearity. As a consequence, for the worst-case linearity of cographs, there is a gap to be filled between the $\Omega(\log n/\log\log n)$ lower bound and the $O(\log n)$ upper bound.

[1] A tree T' is a minor of a tree T if and only if T' can be obtained from T by a series of *edge contractions*, each of which consisting in deleting one node u (distinct from the root) and giving its children to the parent of u.

Relative power of encoding of contiguity and linearity [22, 23]

As mentioned before, linearity offers a greater power of encoding than contiguity, in the sense that the subset of graphs having linearity at most k includes the subset of graphs having contiguity at most k. Consequently, the size of an encoding by linearity k, which uses k orders, is greater than the size of an encoding by contiguity k, which uses only 1 order. Nevertheless, very interestingly, the sizes of these two encodings are equivalent up to a multiplicative constant. Indeed, storing an encoding by contiguity k requires to store a linear ordering of the n vertices of G, *i.e.* a list of n integers, and the bounds of each of the k intervals for each vertex, *i.e.* $2kn$ integers, the total size of the encoding being $(2k+1)n$ integers. On the other hand, the linearity encoding also requires to store $2kn$ integers for the bounds of the k intervals of each vertex, but it needs k linear orderings of the vertices instead of just one, that is kn integers. Thus, the total size of an encoding by linearity k is $3kn$ integers instead of $(2k+1)n$ for contiguity k and therefore the two encodings have equivalent sizes.

Then the question naturally arises to know whether there are some graphs for which the linearity is significantly less than the contiguity, which would result in a significantly more compact encoding of the graph. More formally, does there exist some graph family for which the linearity is asymptotically negligible in front of the contiguity? Or are these two parameters equivalent up to a multiplicative constant? Depending on the answer, one parameter may be preferred for encoding, or not.

We answer this question in [22, 23] by exhibiting a family of graphs for which the linearity is asymptotically negligible in front of the contiguity. This is the family of cographs $(G_k)_{k\geq 1}$ whose cotree is the complete binary tree of height k. From what precedes, we know that $cont(G_k) = \Omega(k) = \Omega(\log n)$, with n the number of vertices of G_k. To obtain the result we aim at, we show that every cograph on n vertices has linearity at most $O(\log n/\log\log n)$. For G_k, this gives $lin(G_k)/cont(G_k) = 1/\log\log n$ and the result follows.

As a by-product, this solves the question of the worst-case linearity of cographs. Indeed, the $O(\log n/\log\log n)$ upper bound matches the $\Omega(\log n/\log\log n)$ lower bound we show in [16]. Moreover, as for contiguity, the proof of the upper bound is based on a structural parameter dedicated to appreciate the complexity of a cotree T. This parameter is called the *factorial rank*, denoted $factrank(T)$, and is defined as the maximum height of a *double factorial tree* (see [23] for a definition) included as a minor in the cotree of G. Since the double factorial tree of height h has more than $h!$ leaves, then the factorial rank of T is $O(\log n/\log\log n)$. In [23], we prove that for any cograph G, $lin(G) \leq 2\, factrank(T) + 3$, which gives the $O(\log n/\log\log n)$ upper bound on the linearity of any cograph. This provides some explanation on what makes the difficulty of encoding a cograph by linearity, but the question to know whether the factorial rank of the cotree of G is equivalent to the linearity of G, up to a multiplicative constant, remains open.

Conclusion

We showed that linearity provides a strictly more powerful encoding for graphs than contiguity does, meaning that the ratio between the contiguity and the linearity of a graph is not bounded by a constant. From a practical point of view, the meaning of our result is that using several orders, instead of just one, for grouping neighbourhoods of vertices can greatly enhance compression rates in some cases. The first perspective following these results is to effectively compute encodings by linearity and contiguity that are as close as possible from the optimal. This requires to develop techniques to deal with the difficulty of computation[2], such as approximation algorithms, parameterized complexity or heuristics for example. The complexity of these algorithms should be very low in order to apply them to large real-world graphs and see whether these encodings can enhance the size of the graphs that can practically be loaded into memory.

Beside this, there are several natural questions that remain to be answered about the power of encoding of linearity and contiguity. The first one is to determine the worst-case contiguity and the worst-case linearity of arbitrary graphs. It is straightforward to see that both of these values are upper-bounded by $n/2$. Conversely, since there are $2^{n(n-1)/2}$ graphs on n labelled vertices and since contiguity and linearity do not depend on the labels of the vertices, then both encodings must use at least $n(n-1)/2 = \Omega(n^2)$ bits in the worst case for graphs on n vertices. Moreover, when the value of contiguity or linearity is k, the size of the corresponding encoding is $O(k\,n)$ integers, that is $O(k\,n \log n)$ bits. Consequently, both parameters must be at least $\Omega(n/\log n)$ in the worst case. We actually proved that the worst-case contiguity of arbitrary graphs can be up to $\Omega(n)$. Is the worst-case linearity the same as the worst-case contiguity or is it lower? This question is particularly interesting. Indeed, if the linearity of a graph turned out to be always $O(n/\log n)$ this would imply that linearity is an optimal encoding with regard to the worst-case size of encoding of graphs on n vertices, that is $\Omega(n^2)$ bits as explained above. Linearity would then equal the compactness of the adjacency matrix in the worst case, which is indeed n^2 bits, and would outperform the compactness of adjacency lists in the worst case, which is $\Omega(n^2 \log n)$ bits. What would be remarkable then is that at the same time, linearity defines a hierarchy of difficulty of encoding that is able to encode some families of graphs much more compactly than in the

[2]Computing the exact value of the contiguity of an arbitrary graph is NP-hard. The complexity of doing so for linearity is still to be determined, but it is likely to be NP-hard as well.

worst (and average) case, using only $O(n \log n)$ bits for example for families having bounded linearity. This feature is very interesting in practice and is not provided by encodings like the adjacency matrix that achieve the optimal worst-case size but always produce encodings of this size.

Another appealing question, which is closely related, is to know what is the maximum gap between contiguity and linearity for arbitrary graphs. In other words, let $(G_n)_{n\geq 1}$ be a family of graphs on n vertices and let $f(n) = cont(G_n)/lin(G_n)$. Can $f(n)$ tends to infinity faster than $\Omega(\log \log n)$? faster than $\Omega(\log n)$? What is the maximum asymptotic growth possible for $f(n)$?

Finally, it would be very interesting to bound the value of both parameters, not only depending on the number of vertices n but also on the number of edges m. How do these parameters behave depending on the number of edges of the graph? Do they work better when there are many edges or few edges? What is the guarantee offered by these parameters for sparse graphs? Are the encodings they provide as efficient as adjacency lists? Can contiguity and linearity be very different for sparse graphs? Answering these questions would be of great interest both from a theoretical point of view and from a practical point of view for the field of graph encoding.

Perspectives

Along the manuscript, I already gave some conclusions derived from the different pieces of work presented and I discussed some of the immediate questions and perspectives they raise. As a general conclusion, I would like to describe two broad open research axes in which I particularly believe for the field of complex networks and to which I will devote an important part of my activity within the next years.

Link stream theory

Many dynamic networks are actually naturally link streams, *i.e.* collections of triplets (u, v, t) with the meaning that there is a link between nodes u and v at time t (cf. Figure 4.2). Examples of link streams include networks of emails, SMS, phone calls, proximity contacts between individuals, financial or commercial transactions, and more generally any network of interactions localised in time. In some link streams, these interactions may be instantaneous (e.g. emails, SMS), in some others, they may have a duration (e.g. phone calls, contacts), which is then short compared to the whole period of study of the network. Most of the works on such link streams, including those presented in Chapter 2 of this manuscript, transform them into series of graphs prior to any further analysis. The rationale behind this is to lean on familiar objects, graphs, and to benefit from the rich

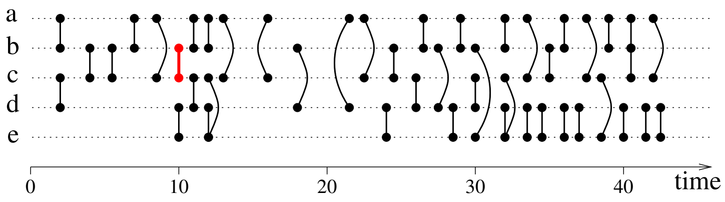


Figure 4.2: Graphical representation of a link stream involving 5 nodes a, b, c, d, e between time 0 and time 45. For example, the red bold line means that there exists a link between nodes b and c at time 10.

concepts of graph theory for their study. The process commonly used to transform link streams into graph series is aggregation, which is specifically studied in the last section of Chapter 2. It consists in forming the graph of all links occurring during a given time window, for a family of time windows that covers the whole period of study. Aggregation comes with two major limitations. The first one is that it denatures the original object and induces a loss of information (cf. Chapter 2). The second one is that, anyway, graph theory does not provide adequate tools for studying graph series and adapting the concepts available for graphs to series of graphs turns out to be difficult and non-intuitive. These difficulties do not come from link streams themselves or from the questions that are asked on these objects, but they rather come from the approach chosen to answer them. In order to go beyond these obstacles, it is necessary to abandon the misleading way of describing dynamic networks by graphs and to develop a theory for a mathematical object of a different nature, called link streams, which is a collection of triplets (u, v, t), where u and v are nodes and t is a time.

In order to build such a theory, the first step is to develop a set of basic notions and a vocabulary to describe link streams, exactly as graph theory does for graphs. This will start by translating and adapting basic graph concepts (such as degree, density, path, distance, connectivity, clique and clustering coefficient for example) to the framework of link streams and this will continue by developing notions that are specific to link streams and that make sense only in this context. The main challenge for adapting graph concepts is to incorporate time into their definitions. To this purpose, one can exploit a fundamental characteristics of link streams, which they share with graphs and which is very useful to build definitions: they naturally admit notions of sub-streams, induced by a subset of nodes, a subset of links or by a time period. A good example of the way graph concepts can be adapted to link streams and of the interest of doing so is provided by the notion of minimal trip developed in Chapter 2, which is a dynamic equivalent to shortest path. Depending on the notion considered, incorporating time can be straightforward or very difficult. In the latter case, such difficulties should not always be addressed technically. It may also simply indicate that the considered notion is not meaningful for link streams and that a different one should be defined and used instead. Finally, in the efforts for building the basis of a link stream theory, one should pay attention to extend, as often as possible, the notions developed in a particular context to both directed and undirected links and to both instantaneous links and links with a duration, so that these notions are able to deal with all the practical contexts where link streams are encountered.

Developing a theory of link streams will open new exciting perspectives for the field of complex networks. It will allow to tackle some new questions that are currently out of reach and to revisit some others that are blocked by the lack of an adequate formalism. One of the questions for which the approach based on link streams is particularly interesting is community detection. In static networks, community detection generally consists in partitioning the set of nodes into dense subgroups, much denser than the relationships between these groups. In the dynamic case, the current approach, driven by the vision based on snapshots, is to find partitions of nodes at different moments of the life of the network, *i.e.* different snapshots, and to follow the evolution of these communities along time: birth, death, split and merge of some of them. Though it has brought some interesting insight, nevertheless, this approach bumps on an intrinsic technical difficulty due to the instability of the algorithms for partitioning one single snapshot into communities. The deep reason for this is that, in this approach, time is not part of the definition of dense groups, which are defined on a snapshot of the network. With a link stream vision, the situation is quite different. It becomes natural to incorporate time in the definition of a dense sub-stream: the interactions in such a sub-stream should be dense for the topology induced between the nodes involved and should happen at a high frequency during the period of the sub-stream. This also points out that the natural notion of community in a link stream may differ from the one in a static network. With the definition proposed above, a community in a link stream corresponds more to a *conversation* rather than a group defined by common characteristics of its members. Efficient algorithms to detect such dense sub-streams, with a definition of density that directly incorporates time, are still to be designed. I am convinced that this approach will avoid the obstructions currently encountered in dynamic community detection and will give a much more meaningful view of the structure of a link stream, by decomposing it into conversations.

Another topic that will be fruitfully refreshed by the link stream vision is the detection of singularities, such as events and anomalies, in the dynamics of the network. Until now, there are mainly two approaches to these problems. The first one is based on graphs and tries to define topological configurations that reveal the singularities of the dynamics. This is in general difficult as such configurations actually also strongly depend on the temporal dimension. On the opposite, the signal processing based approach defines fine metrics to capture the singularities in the temporal variations of the network. The main limitation of this approach is that, in order to get quantities that can be treated by signal processing tools, either the signal of each pair of nodes is considered independently or the whole topology of the network is summarised in one unique numerical quantity. In both cases, this does not allow to track singularities that involve several links but only a restricted part of the topology of the network. Using link streams instead of graph series will largely remove these difficulties, as the notions referring to link streams naturally encompass both topological and temporal dimensions. In particular, a great advantage is that since time is already contained in the metrics used to reveal singularities, even simple statistics on these metrics will be able to do so.

Approximation of complex networks by strongly structured graphs

One major difficulty in the study of complex networks is that they seem not to satisfy strictly any mathematical property one can have in mind. Instead, they rather appear anarchically organised. On the methodological side, graph theory proposes two families of methods to algorithmically treat graphs. The first one deals with arbitrary graphs and the second one with classes of graphs defined by a mathematical property. These two sets of methods are very wide and rich. Unfortunately, none of them is able to treat real-world complex networks. Indeed, most of the methods for arbitrary graphs, including those reputed efficient in theory, such as polynomial time algorithms, suffer from a too high complexity to be applied to real-world instances, because of their huge size. As a consequence, even for simple problems, methods dedicated to arbitrary graphs are often impossible to use in practice. On the other hand, methods for strongly structured classes of graphs usually achieve very low complexity, even for problems that are difficult in general settings. To this regard, these methods would be able to deal with the huge size of real-world instances. But they cannot accommodate with graphs that do not strictly satisfy the property defining the class for which they are designed: if the input graph deviates even very slightly from this property, the computational method is not valid. As a result, the set of computational methods available to analyse complex networks is still very restricted and the domain has developed its own methods in order to describe and manipulate complex networks. These methods essentially rely on the statistical study of some of their numerical properties, such as degree distribution and clustering coefficient for example. Though this approach has given interesting insight into the structure of complex networks, it also shows clear limitations. The reason for this is that these numerical properties cannot fully capture the combinatorial properties of these objects. As a consequence, our understanding of the functioning of these systems and our ability to exploit them remain sharply limited.

This state of fact is particularly regrettable as we know that real-world networks actually have a strong structure, which we are unfortunately unable to fully exploit. This structure, which is conferred by the constraints of the concrete contexts they come from, makes them very different from average random graphs and points out that they should not be treated as arbitrary graphs. The alternative way I propose in this manuscript is to treat them as approximations of strongly structured graphs, by using graph editing problems in order to unveil their hidden combinatorial structures. Editing a graph consists in changing some of its adjacencies, *i.e.* adding and removing edges, in order to make it satisfy a target property fixed in advance, such as being chordal or being a cograph for example. The aim is to perform the minimum possible number of modifications, and this number is then a measure of how far is the input graph to satisfy the target property. Going further, using the result of the edition algorithm, the input graph can be represented as an

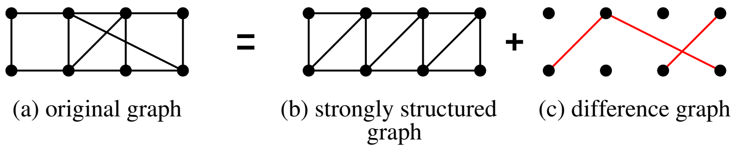


Figure 4.3: Representation of a graph as an almost structured graph. (a) original graph; (b) strongly structured graph obtained by edition; (c) graph of the differences between the two preceding graphs. The representation of graph (a) is formed by graphs (b) and (c). Indeed, graph (a) is obtained from graph (b) by removing edges of (c) that are also in (b) and by adding edges of (c) that are not in (b).

approximation of a strongly structured graph, called an *almost structured graph*. This representation is made of two graphs: the strongly structured graph obtained as output of the edition algorithm, and the graph of the differences between this output graph and the input original graph (see Figure 4.3). Remarkably, Chapter 3 shows, using cographs as target class, that some real-world networks are not far from having a perfect combinatorial structure, meaning that only a relatively small number of modifications, compared to their number of edges, is enough to confer them such a perfect structure. More surprisingly, even in cases where the real-world network is not so close from the target property, the representation scheme described above sometimes gives very useful results. This shows the great benefit of taking into account the underlying combinatorial structure of complex networks, even if they partially deviate from it.

The approximation of complex networks by strongly structured graphs opens exciting possibilities for almost all research topics of the domain. For modelling purposes, Chapter 3 demonstrates the interest of representing a real-world graph by a strongly structured graph plus the set of edges they differ from each other. In this way, the strongly structured part of the topology can control simultaneously various numerical properties, such as the degree distribution and the clustering coefficient, that we are otherwise unable to reproduce conjointly in random generation processes. Therefore the almost structured graph approach offers a very promising way to overcome the difficulties currently encountered in generating realistic synthetic topologies of complex networks.

There are a number of other topics in complex network theory that will benefit from this approach, such as analysis and encoding for example. As explained in Chapter 4, because of the huge size of many real-world networks, it is necessary to encode them and store them efficiently in the memory of the machine used for their manipulation. There is a double exigence on the encoding schemes used. First, they should provide structures that are very compact in space, so that the graph fits in the memory. Second, the encoding must allow to answer quickly the queries made on the graph by algorithms that process it. To this regard, the approximation of complex networks by almost structured graphs is a very appealing approach. Indeed, strongly structured classes of graphs, such as cographs, often admit encodings that satisfy both exigences mentioned above: very compact (e.g. $O(n)$ space for cographs) and allowing to answer queries in optimal time. Thus, when storing a real-world graph as an almost structured graph, like in Figure 4.3, the limiting factor is the encoding of the difference graph. But when the original graph and its strongly structured approximation are close enough to each other, the difference graph is much smaller than the original one, therefore improving the efficiency of the encoding both with regard to space and with regard to time of treatment of queries.

One fundamental goal of complex network analysis is to understand the global organisation of these systems and the particular roles played by their constituting elements. For this purpose as well, finding strongly structured graphs that are close to real-world graphs will provide a key information. Indeed, such graph classes are perfectly structured with regard to one mathematically defined kind of organisation. Therefore, finding such structures underlying those of complex networks will shed a new light on their own organisations. Using edition problems into different target classes of graphs, one can test and retrieve different elements of organisation of real-world networks. For example, classes related to planar graphs could be used to retrieve the spatially constrained part of the structure of some networks, cographs are prone to reveal nodes having similar roles, distance hereditary graphs could point out nodes acting as interfaces between disconnected parts of the network, chordal graphs could reveal a tree-like hierarchical organisation of the network, and so on. Approximation by these classes of graphs will then allow to retrieve such organisations even if they are not perfect and partially hidden by the part of randomness contained in the real-world data.

In order to use the proposed approximation approach for the different purposes listed above, one first needs to design algorithms and heuristics that can solve edition problems into various graph classes and that are efficient enough to handle real-world networks of very large size, like the cograph edition algorithm presented in Chapter 3. Among the classes of graphs for which it would be particularly interesting to design minimal edition algorithms, one can cite distance hereditary graphs, circle graphs, which are generalisations of cographs, chordal graphs and classes related to planar graphs. For some of these classes, there already exist minimal completion algorithms. Experimental results show that the number of modifications to be made in completion problems is usually much larger than the number of modifications in the corresponding edition problem. For sake of the performances of the approximation approach, it is therefore essential to develop edition algorithms that use both additions and deletions of edges.

Another set of questions that needs to be addressed is related to the use of minimal edition algorithms as heuristics for minimum edition. What can be expected from the quality of the solutions obtained in this way compared to the optimal? For incremental algorithms, can we quantify the improvement provided by the computation of a minimum cardinality edition of the neighbourhood of the incoming vertex at each incremental step, compared to the computation of any arbitrary minimal edition? What is the gain of not restraining the algorithm to one single round on the vertices but instead considering vertices several times until no further improvement is possible? Another exciting research direction is to compute exact solutions, for example by developing pre-processing techniques that are able to take benefit of the specific structure of complex networks, or to design heuristics providing nearly tight upper and lower bounds on the optimal solution. All these questions are interesting in themselves from a purely algorithmic point of view. On the practical side, they must be addressed in order to provide the proposed approximation approach with computational methods that i) can solve various edition problems, ii) are able to treat large instances of graphs and iii) provide solutions as close as possible from the optimal.

Finally, what is probably the most exciting and most promising way opened by the approximation of complex networks by strongly structured graphs is to use it to speed-up and enlarge the set of algorithmic treatments available for real-world networks. The idea to do so is to exploit the proximity of the real-world network considered with some strongly structured graph (e.g. a cograph), for which most of algorithmic problems (even difficult ones) can be solved very efficiently. As a first step, one can take the result obtained for the strongly structured graph as an approximation of the result for the original graph, which is expected to be better when the two graphs are closer in terms of the number of differences between them. Then, the quality of the approximate result should be studied, formally or empirically, depending on the proximity between the real-world network and its strongly structured approximation. Even with a poor guarantee, the possibility to obtain an approximate result for real-world instances of graphs, for which often no result is reachable with current approaches, is very valuable in practice.

Though, the most interesting perspective, both from a theoretic point of view and from a practical point of view, is to use the set of differences between the original graph and its approximation in order to obtain an exact result or a better guarantee on the approximate result. This is a major challenge for algorithmic graph theory in the next decades: develop an algorithmic theory of almost structured graphs. This can be seen as an extension of dynamic graph algorithms. In dynamic graph algorithms the difference between the two graphs is elementary (typically, the addition or the deletion of one single edge or vertex) and one considers a sequence of such modifications occurring independently and without knowledge of the modifications to come. Here, the difference between the original graph and its approximation is not elementary but can be arbitrary. This is actually a more favourable framework since it is equivalent to considering all dynamic modifications at once, which removes uncertainty on the future, helps to avoid redundant computations, and can therefore allow to achieve a better overall complexity. Another major difference with dynamic algorithms is that they traditionally consider only two cases: either both the starting graph and the modifications performed are arbitrary or the starting graph belongs to a restricted class of graphs and the algorithm considers only elementary modifications that result in a modified graph belonging to this class (otherwise the dynamic algorithm stops). In the framework proposed here, the idea is to consider arbitrary modifications while taking advantage of the fact that the starting graph is not arbitrary but belongs to a strongly structured class of graphs. The aim is to express the complexity of algorithms as a function of the number of differences with the original graph, plus eventually the size of the representation of the strongly structured graph (which usually admits a very compact representation). Developing such algorithms would open a new way in algorithmic graph theory and would allow to obtain results for real-world networks that are out of reach with current approaches.

Bibliography

[1] Dimitris Achlioptas, Aaron Clauset, David Kempe, and Cristopher Moore. On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. *J. ACM*, 56(4), 2009.

[2] Laurent Alonso and René Schott. *Random Generation of Trees*. Springer, 1995.

[3] Alberto Apostolico and Guido Drovandi. Graph compression by BFS. *Algorithms*, 2:1031–1044, 2009.

[4] Yasuhito Asano, Tsuyoshi Ito, Hiroshi Imai, Masashi Toyoda, and Masaru Kitsuregawa. Compact encoding of the web graph exploiting various power laws. In *4th International Conference on Advances in Web-Age Information Management (WAIM 2003)*, volume 2762 of *LNCS*, pages 37–46. Springer, 2003.

[5] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[6] Hervé Baumann, Pierluigi Crescenzi, and Pierre Fraigniaud. Parsimonious flooding in dynamic graphs. *Distributed Computing*, 24(1):31–44, 2011.

[7] Edward A Bender and E.Rodney Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3):296–307, 1978.

[8] Paolo Boldi and Sebastiano Vigna. The WebGraph framework I: compression techniques. In *WWW'04*, pages 595–602. ACM, 2004.

[9] Béla Bollobás, Svante Janson, and Oliver Riordan. Sparse random graphs with clustering. *Random Structures and Algorithms*, 38(3):269–323, 2011.

[10] Ulrik Brandes, Michael Hamann, Ben Strasser, and Dorothea Wagner. Fast quasi-threshold editing. In *Algorithms (ESA 2015)*, volume 9294 of *LNCS*, pages 251–262. Springer, 2015.

[11] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph Classes: a Survey.* SIAM Monographs on Discrete Mathematics and Applications. SIAM, 1999.

[12] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, Michael Mitzenmacher, Alessandro Panconesi, and Prabhakar Raghavan. On compressing social networks. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*, pages 219–228. ACM, 2009.

[13] Aaron Clauset, Cosma Rohilla Shalizi, and Mark Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.

[14] Andrea E.F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time in edge-markovian dynamic graphs. In *27th ACM symposium on Principles of distributed computing (PODC 2008)*, pages 213–222. ACM, 2008.

[15] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, third edition, 2009.

[16] Christophe Crespelle and Philippe Gambette. Efficient neighborhood encoding for interval graphs and permutation graphs and $o(n)$ breadth-first search. In *20th International Workshop on Combinatorial Algorithms (IWOCA 2009)*, volume 5874 of *LNCS*, pages 146–157. Springer, 2009.

[17] Christophe Crespelle and Philippe Gambette. Unrestricted and complete breadth-first search of trapezoid graphs in $O(n)$ time. *Information Processing Letters*, 110(12-13):497–502, 2010.

[18] Christophe Crespelle and Philippe Gambette. Linear-time constant-ratio approximation algorithm and tight bounds for the contiguity of cographs. In *7th International Workshop on Algorithms and Computation (WALCOM 2013)*, volume 7748 of *LNCS*, pages 126–136. Springer, 2013.

[19] Christophe Crespelle and Philippe Gambette. (Nearly-)tight bounds on the contiguity and linearity of cographs. *Theoretical Computer Science*, 522:1–12, 2014.

[20] Christophe Crespelle, Matthieu Latapy, and Thi Ha Duong Phan. On the termination of some biclique operators on multipartite graphs. *Discrete Applied Mathematics*, 195:59–73, 2015.

[21] Christophe Crespelle, Matthieu Latapy, and Elie Rotenberg. Rigorous measurement of IP-level neighborhood of Internet core routers. In *2nd IEEE International Workshop on Network Science and Communication Networks (NetSciCom 2010). In INFOCOM IEEE Conference on Computer Communications Workshops*, pages 1–6, 2010.

[22] Christophe Crespelle, Tien-Nam Le, Kevin Perrot, and Thi Ha Duong Phan. Linearity is strictly more powerful than contiguity for encoding graphs. In *14th International Symposium on Algorithms and Data Structures (WADS 2015)*, volume 9214 of *LNCS*, pages 212–223. Springer, 2015.

[23] Christophe Crespelle, Tien-Nam Le, Kevin Perrot, and Thi Ha Duong Phan. Linearity is strictly more powerful than contiguity for encoding graphs. *Discrete Mathematics*, 339(8):2168–2177, 2016. Submitted journal paper.

[24] Christophe Crespelle, Daniel Lokshtanov, Thi Ha Duong Phan, and Eric Thierry. Faster and enhanced inclusion-minimal cograph completion. In *11th International Conference on Combinatorial Optimization and Applications (COCOA 2017)*, number 10627 (Part I) in LNCS, pages 1–15, 2017.

[25] Christophe Crespelle, Anthony Perez, and Ioan Todinca. An $o(n^2)$-time algorithm for the minimal permutation completion problem. In *41st International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2015)*, volume 9224 of *LNCS*, pages 103–115. Springer, 2016.

[26] Christophe Crespelle, Thi Ha Duong Phan, and The Hung Tran. Termination of the iterated strong-factor operator on multipartite graphs. *Theoretical Computer Science*, 571:67–77, 2015.

[27] Christophe Crespelle and Fabien Tarissan. Evaluation of a new method for measuring the Internet degree distribution: Simulation results. *Computer Communications*, 34(5):635–648, 2011.

[28] Christophe Crespelle and Ioan Todinca. An $O(n^2)$-time algorithm for the minimal interval completion problem. In

7th Annual Conference on Theory and Applications of Models of Computation (TAMC 2010), number 6108 in LNCS, pages 175–186, 2010.

[29] Christophe Crespelle and Ioan Todinca. An $O(n^2)$-time algorithm for the minimal interval completion problem. *Theor. Comput. Sci.*, 494:75–85, 2013.

[30] Sergey N. Dorogovtsev, José Fernando Mendes, and Alexander N. Samukhin. Size-dependent degree distribution of a scale-free growing network. *Phys. Rev. E*, 63(6):062101, 2001.

[31] Paul Erdös and Alfréd Rényi. On the evolution of random graphs. *Math. Inst. Hung. Acad. Sci.*, 5:17–61, 1960.

[32] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the Internet topology. In *ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM 1999)*, pages 251–262. ACM, 1999.

[33] James P. Gleeson. Bond percolation on a class of clustered random networks. *Phys. Rev. E*, 80(3):036107, 2009.

[34] Paul W. Goldberg, Martin C. Golumbic, Haim Kaplan, and Ron Shamir. Four strikes against physical mapping of DNA. *Journal of Computational Biology*, 2(1):139–152, 1995.

[35] Mark S. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78(6):1360–1380, 1973.

[36] Jean-Loup Guillaume and Matthieu Latapy. Bipartite structure of *all* complex networks. *Information Processing Letters*, 90(5):215–221, 2004.

[37] Jean-Loup Guillaume and Matthieu Latapy. Bipartite graphs as models of complex networks. *Physica A*, 371:795–813, 2006.

[38] Creighton Heaukulani and Zoubin Ghahramani. Dynamic probabilistic models for latent feature propagation in social networks. In *30th International Conference on Machine Learning (ICML 13)*, volume 28, pages 275–283. JMLR Workshop and Conference Proceedings, 2013.

[39] Pinar Heggernes, Jan Arne Telle, and Yngve Villanger. Computing minimal triangulations in time $O(n^{\alpha} \log n) = o(n^{2.376})$. *SIAM J. Discrete Math.*, 19(4):900–913, 2005.

[40] Petter Holme and Beom Jun Kim. Growing scale-free networks with tunable clustering. *Phys. Rev. E*, 65(2):026107, 2002.

[41] Hawoong Jeong, S. P. Mason, Albert-László Barabási, and Zoltán N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411:41–42, 2001.

[42] Songwei Jia, Lin Gao, Yong Gao, James Nastos, Yijie Wang, Xindong Zhang, and Haiyang Wang. Defining and identifying cograph communities in complex networks. *New Journal of Physics*, 17:013044, 2015.

[43] Jon Kleinberg. The small-world phenomenon: An algorithmic perspective. In *32nd Annual ACM Symposium on Theory of Computing (STOC 2000)*, pages 163–170. ACM, 2000.

[44] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew Tomkins, and Eli Upfal. Stochastic models for the web graph. In *41st Annual Symposium on Foundations of Computer Science (FOCS 2000)*, page 57. IEEE, 2000.

[45] Anukool Lakhina, John W. Byers, Mark Crovella, and Peng Xie. Sampling biases in IP topology measurements. In *INFOCOM*, 2003.

[46] Matthieu Latapy, Thi Ha Duong Phan, Christophe Crespelle, and Thanh Qui Nguyen. Termination of multipartite graph series arising from complex network modelling. In *4th International Conference on Combinatorial Optimization and Applications (COCOA 2010)*, number 6508 (Part I) in LNCS, pages 1–10. Springer, 2010.

[47] Matthieu Latapy, Elie Rotenberg, Christophe Crespelle, and Fabien Tarissan. Measuring the degree distribution of routers in the core Internet. In *13th IFIP International Conference on Networking (Networking 2014)*, pages 1–9. IEEE, 2014.

[48] Matthieu Latapy, Elie Rotenberg, Christophe Crespelle, and Fabien Tarissan. Rigorous measurement of the Internet degree distribution. *Complex Systems*, 26(1), 2017.

[49] Guillaume Laurent, Jari Saramäki, and Márton Karsai. From calls to communities: a model for time-varying social networks. *The European Physical Journal B*, 88(11):301, 2015.

[50] Yannick Léo, Christophe Crespelle, and Eric Fleury. Non-altering time scales for aggregation of dynamic networks into series of graphs. Submitted journal paper.

[51] Yannick Léo, Christophe Crespelle, and Eric Fleury. Non-altering time scales for aggregation of dynamic networks into series of graphs. In *11th ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT 2015)*, pages 29:1–29:7. ACM, 2015.

[52] Xiafeng Li, Derek Leonard, and Dmitri Loguinov. On reshaping of clustering coefficients in degree-based topology generators. In *3rd International Workshop on Algorithms and Models for the Web-Graph (WAW 2004)*, pages 68–79. Springer, 2004.

[53] Daniel Lokshtanov, Federico Mancini, and Charis Papadopoulos. Characterizing and computing minimal cograph completions. *Discrete Appl. Math.*, 158(7):755–764, 2010.

[54] Jean loup Guillaume, Matthieu Latapy, and Laurent Viennot. Efficient and simple encodings for the web graph. In *3rd International Conference on Web-Age Information Management*, volume 2419 of *LNCS*, pages 328–337. Springer, 2002.

[55] Matteo Marsili, Fernando Vega-Redondo, and František Slanina. The rise and fall of a networked society: A formal model. *Phys. Rev. E*, 101(6):1439–1442, 2004.

[56] Lucie Martinet, Christophe Crespelle, and Eric fleury. Dynamic contact network analysis in hospital wards. In *5th Workshop on Complex Networks (CompleNet 2014)*, number 549 in Studies in Computational Intelligence, pages 241–249. Springer, 2014.

[57] Lucie Martinet, Christophe Crespelle, Eric fleury, Pierre-Yves Boëlle, and Didier Guillemot. The dynamic contact network of a whole hospital. Submitted journal paper.

[58] Michael Molloy and Bruce Reed. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, 6(2-3):161–180, 1995.

[59] James Moody. The importance of relationship timing for diffusion. *Social Forces*, 81(1):25–56, 2002.

[60] Vincent Neiger, Christophe Crespelle, and Eric Fleury. On the structure of changes in dynamic contact networks. In *Workshop on Complex Networks and their Applications (Complex Networks 2012). In 8th International Conference on Signal Image Technology and Internet Based Systems (SITIS 2012)*, pages 731–738, 2012.

[61] Mark Newman, Duncan J. Watts, and Steven H. Strogatz. Random graph models of social networks. *PNAS*, 99, suppl. 1:2566–2572, 2002.

[62] Jean-Jacques Pansiot and Dominique Grad. On routes and multicast trees in the Internet. *SIGCOMM Comput. Commun. Rev.*, 28(1), 1998.

[63] Nicola Perra, Bruno Gonçalves, Romualdo Pastor-Satorras, and Alessandro Vespignani. Activity driven modeling of time varying networks. *Scientific Reports*, 2:469, 2012.

[64] A. Ramachandra Rao, Rabindranath Jana, and Suraj Bandyopadhyay. A markov chain monte carlo method for generating random (0, 1)-matrices with given marginals. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 58(2):225–242, 1996.

[65] Bruno Ribeiro, Nicola Perra, and Andrea Baronchelli. Quantifying the effect of temporal resolution on time-varying networks. *Scientific Reports*, 3, 2013.

[66] Donald J. Rose, Robert E. Tarjan, and George S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.*, 5:146–160, 1976.

[67] Elie Rotenberg, Christophe Crespelle, and Matthieu Latapy. Measuring routing tables in the Internet. In *6th IEEE International Workshop on Network Science for Communication Networks (NetSciCom 2014). In INFOCOM IEEE Conference on Computer Communications Workshops*, pages 795–800. IEEE, 2014.

[68] Antoine Scherrer, Pierre Borgnat, Eric Fleury, Jean-Loup Guillaume, and Céline Robardet. Description and simulation of dynamic mobility networks. *Computer Networks*, 52(15):2842–2858, 2008.

[69] Ángeles Serrano and Marián Boguñá. Tuning clustering in random networks with arbitrary degree distributions. *Phys. Rev. E*, 72(3):036133, 2005.

[70] Tom A.B. Snijders, Gerhard G. van de Bunt, and Christian E.G. Steglich. Introduction to stochastic actor-based models for network dynamics. *Social Networks*, 32(1):44–60, 2010.

[71] Karol Suchan and Ioan Todinca. Minimal interval completion through graph exploration. *Theoretical Computer Science*, 410(1):35–43, 2009.

[72] Rajmonda Sulo, Tanya Berger-Wolf, and Robert Grossman. Meaningful selection of temporal resolution for dynamic networks. In *8th Workshop on Mining and Learning with Graphs (MLG 2010)*, pages 127–136. ACM, 2010.

[73] Jeffrey Travers and Stanley Milgram. An experimental study of the small world problem. *Sociometry*, 32(4):425–443, 1969.

[74] György Turan. On the succinct representation of graphs. *Discr. Appl. Math.*, 8:289–294, 1984.

[75] Erik Volz. Random networks with tunable degree distribution and clustering. *Phys. Rev. E*, 70(5):056115, 2004.

[76] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.

[77] Carsten Witt. Greedy local search and vertex cover in sparse random graphs. In *6th Annual Conference on Theory and Applications of Models of Computation (TAMC 2009)*, volume 5532 of *LNCS*, pages 410–419. Springer, 2009.

[78] Xiao Zhang, Cristopher Moore, and Mark Newman. Random graph models for dynamic networks. *European Physical Journal B*, 90(10):200, 2017.

Outline of the thesis

This manuscript is a synthesis of my research work in the field of complex networks, after my PhD in computer science. It contains four chapters. The first one deals with the measurement of the degree distribution of the Internet topology. Our approach is based on a new principle called *property oriented measurement*, which provides more faithful information by focusing only on one target property, without collecting a map of the network. Two methods are presented, one for measuring the degree distribution of the logical topology and one for measuring the degree distribution of the physical topology.

Chapter 2 presents three works on the dynamics of complex networks. The first one is a case study on the dynamic network of contacts within a hospital and the other two are methodological developments dedicated to dynamic networks in general. One is about characterising the structure of changes of the topology of a dynamic network, the other one addresses the problem of finding appropriate time scales in order to aggregate a dynamic network into a series of graphs.

Chapter 3 deals with complex network modelling. The aim is to design random generation processes that output synthetic networks having properties as close as possible from those observed for real-world networks. We describe two different approaches toward this goal. One is based on the entanglement structure of maximal cliques, while the other one is based on the approximation of complex networks by strongly structured graphs.

Finally, the last chapter considers the problem of designing very compact encodings of graphs that do not penalise the queries made during algorithmic treatments, such as listing the neighbours of one given vertex. We investigate the efficiency of two related encodings, named *contiguity* and *linearity*, which use linear orderings of the vertices of the graph.