Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

PhD Thesis

Designing for the Future: A Complex Systems Approach to Communication Networks

Author:
Merim Džaferagić

Supervisor:
Prof. Nicola Marchetti
Dr. Irene Macaluso

12th February 2020

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

I consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Signed:

[illegible]

Merim Džaferagić, 12th February 2020

Summary

The network size and the deployment density of wireless networks continue to increase from year to year. Additionally, networks are experiencing an operational shift that affects both: (1) the network architecture, and (2) the service implementation. The network architecture is changing from a traditionally rigid hierarchical - hardware first - to a more flat and flexible - software first - implementation. This shift enables innovation related to, among other things, network infrastructure ownership, dynamic resource sharing, on-demand resource allocation. The evolution of the network architecture underpins the shift related to the service implementation. New services have strict requirements (e.g. latency, throughput, reliability), dynamically demanding resources on a more granular level. The growing size and operational changes demand scalability and adaptability to be part of the network design. Not only are individual networks becoming more sophisticated, but it is increasingly infeasible to consider any one kind of network in isolation; networks such as cellular, Wi-Fi, vehicular and Internet of Things (IoT), increasingly have interdependencies. In addition, many subsets of networks are no longer centrally planned and rolled-out by a single owner, but evolve over time based on user deployed infrastructure. The network can be viewed as a living organism, that evolves over time and adapts to the changes in its environment. Focusing on making nodes more capable and intelligent as done to date by the cognitive radio community often results in higher cost, limiting the scalability of the adopted techniques.

In this thesis, we propose a complex systems science approach to communication networks. We focus on three complex systems principles, i.e. complexity, degeneracy and emergence. Additionally, we divide the complex systems tools and techniques into three layers, i.e. analysis, modeling and design. The problem that we address in this thesis can be stated in the form of the following research question: "What does the analysis of the micro-scale structures tell us about the macro-scale performance, and how do the local interaction rules lead to global organization/synchronization in communication networks?" Our first step was to identify what aspects of the network (e.g. infrastructure, network functions, signal processing) could be analyzed in the context of complex systems. The focus is on network functions (e.g. clustering in Wireless Sensor Networks (WSNs), frequency allocation in cellular networks), allowing us to have a better understanding of the impact that different protocol procedures have on the network operation. The research question can be broken down into three parts, which drove the development of the thesis:

1. How to quantify the impact that micro-scale structures have on the macro-scale

performance of communication networks?

2. How to identify macro-scale/system-level topologies that are constructed out of diverse micro-scale structures (i.e. degenerate structures)?
3. How to achieve global organization through limited knowledge and local interactions?

We start with a discussion of different tools and techniques that have been developed and applied to understand unexplored system properties (e.g. the capability of a system to store, communicate and process information) in sciences like physics, neurobiology, urbanism, social networks, etc. We address the above-mentioned questions by resorting to a wide range of tools for analysis (e.g. network science, information theory, statistical mechanics), modeling (e.g. Equation-Based Modeling, Agent-Based Modeling) and design (e.g. Reference Design, Trial and Error, Analytical Approach). We also discuss the evolution of communication networks, showing that the need for flexibility, scalability and adaptability demands new approaches to analysis, modeling and design.

We then propose a framework to model the underlying structure of network functions with graphs, allowing us to study the organizational characteristics of network functions as complex systems. In order to quantify the impact that micro-scale structures have on the macro-scale performance, we propose a complexity metric called *functional complexity*. This allows us to correlate the organizational structure to the performance of different network functions (e.g. frequency allocation in cellular networks, clustering in WSNs).

Then we shift our focus from complexity to degeneracy and reapply the same modeling approach, i.e. modeling network functions with graphs. Here we focus on the identification of macro-scale/system-level topologies that are constructed out of diverse micro-scale structures in IoT networks by studying degeneracy, i.e. the multiplicity of computational graphs that allow us to perform the same computation by using different subgraph structures.

Finally, we move on to the third layer, i.e. design. We design local rules of interaction between base stations in a cellular network that emerge in a desirable global property, i.e. formation of handover regions that minimize the signaling and latency related to mobility management. Here, we apply the tools from all three layers, i.e. analysis, modeling and design, showing the full potential and benefits of the application of complex systems tools to design scalable, adaptive and robust network functions for future communication networks.

Acknowledgements

First, I would like to express my gratitude to my supervisors Nicola Marchetti and Irene Macaluso for all the support, motivation and belief in my abilities. It was a pleasure to work with you and learn from you throughout this process. This research would not have happened without you.

I would like to thank my co-workers and friends for creating an amazing work environment and making every day in the lab so interesting. Deepest thanks to Conor for being a true friend and for reading and correcting every word that I have ever written in English (without the intention of reading a single page of my research). To Nick for all the discussions about science, sports and languages - you were an invaluable part of my early research steps. To Elma and Arman, I will never forget my first weeks in Dublin, and all the wine and late night conversations that made me feel at home (and my back will never forget the couch). To Francisco for all the "short" pub-discussions about politics, science, sports and foosball. To Marcelo for being the friendliest person I have ever met. To Joao and Erika for being great friends and fellow AVENGERS. To Maicon for sharing my enthusiasm for gadgets that none of us actually needs. I am very grateful to share this office with such great people, Jernej, Nima, Alan, Jonathan, Georgios, Parna, Danny, Tom, Fadhil, Boris, Harun, Sandip, Yi, Stefan, Andrea, Jacek, Harleen, Andrei, and so many others. Thanks for all of the fun times and the great memories. Thanks also to Darijo for always being a true friend and for your great support throughout this process. Thanks to Demir, Haris, Amer, Tarik and all of my friends who kept me going when I needed distractions outside of research, and who reminded me that home is just one phone call away.

A huge thank you to my mom Ajka, my dad Abid, my sister Merima and my uncle Sulejman for all their love and encouragement, and for raising me the way I am. You are the source of my confidence and your support means everything to me. This would not have been possible without you. And last but not least, a huge thank you to my brother-in-law Almir and to my cousins Emir, Hana, Emina, Admir, Semir and Nermina for keeping me up-to-date with all the small and big things happening at home.

Contents

List of Figures

List of Tables

1 Introduction

Introduction

The architecture and the service implementation of telecommunication networks are rapidly changing due to the constant increase in size and deployment density. Due to the heterogeneous nature of nodes, ranging from simple sensors to sophisticated cognitive entities, and a variety of networks, such as cellular, Wi-Fi, vehicular and Internet of Things (IoT), that coexist in the wireless ecosystem, we can not consider any one kind of network in isolation. Indeed, the increasing demand for resources precipitates the need for cooperation and collaboration in the new generation of communication networks.

The network architecture is changing from a rigid hierarchical - hardware first - to a flat and flexible - software first - implementation. Additionally, some parts of the network architecture that were traditionally static (e.g. Base Stations (BSs), network functions - mobility management, authentication, load balancing) are evolving and moving towards a more dynamic implementation. For example: (1) Unmanned Aerial Vehicles (UAVs) are being used as BSs allowing the physical topology to change over time; (2) Network Function Virtualization (NFV) allows us to dynamically place and move instances of network functions throughout the network.

The innovation happening in the network architecture enables the fast development and evolution of services. Unlike traditional communication networks, current and especially future networks are not simple pipelines for information transfer. Instead, they generate information (e.g. sensor networks gather information about the environment, vehicular networks provide information about the traffic), and have the capability to store, communicate and process them.

The changes related to the network architecture and the development of different network services results in the need to develop new approaches to network control. The mainstream ideas related to network control suggest the decoupling of the data and control plane and propose a centralized all-knowing controller that would optimize the network configuration. However, this control architecture may not be suitable to all scenarios due to scalability, reliability (e.g. single point-of-failure) and reliance on high-speed and low latency communications between the controller and the network elements. The approaches taken to tackle all these issues rely on distributed mechanisms (e.g. multiple cooperating controllers, closing the control loop on remote devices that are latency sensitive).

All the above-mentioned changes force us to change the way we study and design commu-

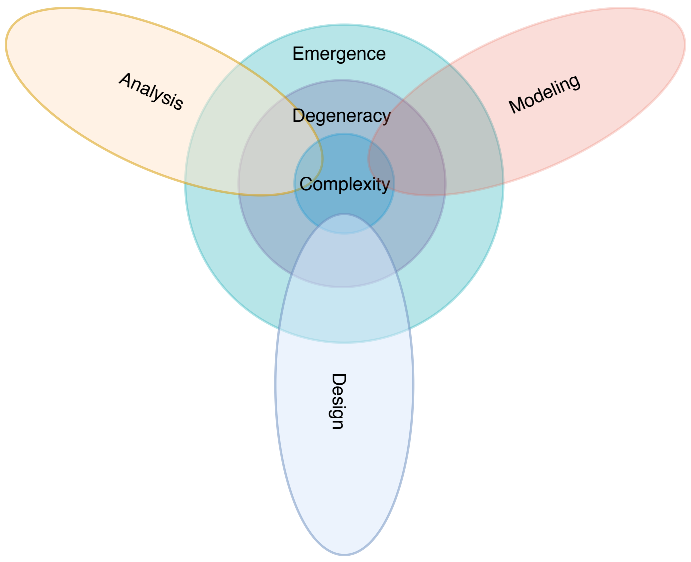


Figure 1.1: Thesis roadmap showing the three complex systems properties (i.e. *Complexity*, *Degeneracy* and *Emergence*) and the three layers (i.e. *Analysis*, *Modeling* and *Design*) of interest.

nication networks. There is no straightforward network theory that can describe the overall network or the interplay between different networks. The network is like a living organism that evolves over time and adapts to the changes in its environment. That is why we focus on studying communication networks from a complex systems perspective, allowing us to use tools and methodologies built with dynamic, cooperative and collaborative behavior of the system parts in mind.

1.1 Scope

In this thesis we focus on studying the impact that the organizational structure has on the operation of communication networks. As highlighted in the previous section, structure in communication networks can refer to different properties (e.g. physical topology, network function architecture and placement, interference graph). We focus on the interaction between network nodes during the execution of a network function taking into account the physical topology. The main research question that drove the development of this thesis can be stated as: "What does the analysis of the micro-scale structures tell us about the macro-scale performance, and how do the local interaction rules lead to global organization/synchronization in communication networks?"

As shown in Figure 1.1, we focus on the study of three properties that characterize a complex system: *Complexity*, *Degeneracy* and *Emergence*. Figure 1.1 also depicts the classification of the complex systems methodologies and tools into three layers - analysis, modeling and design. Figure 1.1 will be used as a roadmap throughout this thesis to highlight the layers and properties that are of interest in different chapters.

We have broken the main research question into three parts:

How to quantify the impact that micro-scale structures have on the macro-scale performance of communication networks?

We address this research question in Chapter 3 by studying different graph structures and multiple network functions, like frequency allocation in cellular networks and clustering in Wireless Sensor Networks (WSNs). To capture the non-linear joint impact that those structures have on the performance of the network functions we take a bottom-up approach and investigate all subgraph structures that are involved in the execution of a network function. To quantify this impact we propose a suitable metric - *functional complexity*.

How to identify macro-scale/system-level topologies that are constructed out of diverse micro-scale structures (i.e. degenerate structures)?

Unlike the previous question, here we start from the macro-scale topology and work our way down to study the potential of a physical topology (macro-scale) to provide multiple functional alternatives on the micro-scale (i.e. degenerate structures). In particular, we focus on in-network computing and study the multiplicity of available partially overlapping graphs that enable distributed in-network computing with a given macroscopic observable property (e.g. delay).

How to achieve global organization through limited knowledge and local interactions?

Here, we take a bottom-up approach and iterate through the *Analysis*, *Modeling* and *Design* layers. More precisely, we design rules of interaction between BSs in a cellular network - local rules and limited knowledge - and study the emergent properties on the global level.

1.2 Outline

The remainder of the thesis is divided into five Chapters. Chapter 2 provides an overview of different complex systems tools to study the capability of a communication network to store, communicate and process information. In Chapter 3, we study the organizational characteristics of network functions as complex systems. Then, in Chapter 4, we use the modeling framework proposed in Chapter 3 (i.e. functional topology framework) to study the degeneracy of distributed computing in an IoT network. In Chapter 5, we develop rules of interaction between BSs in a cellular network that guide the network towards a desirable state, i.e. formation of handover regions that minimize the signaling and latency related to mobility management. Finally, in Chapter 6, we summarize the main insights gained, and discuss the future directions and applications of complex systems tools to communication networks.

Chapter 2 - Background

Chapter 2 starts with the introduction to complex systems science. First, we introduce the term *complexity* by highlighting the differences between a complicated and a complex system. This allows us to explain the different approaches to understand those systems: reductionist approach for a complicated and a complex systems approach for a complex system. Then, we provide an overview of different tools to study the complexity of a wide range of systems in sciences like physics, neurobiology, urbanism, social networks, etc. We categorize these tools into three layers: *Analysis*, *Modeling* and *Design*. We also provide an overview of properties that characterize a complex system (e.g. *Complexity*, *Degeneracy* and *Emergence*). Finally, the role of organization in complex systems allows us to explain the connection between the complex systems tools and the concepts of *Complexity*, *Degeneracy* and *Emergence*.

Chapter 3 - Functional Complexity Framework for the Analysis and Modeling of Networks

In Chapter 3 we build on the tools introduced in the previous chapter. In particular, we focus on the *Analysis* and *Modeling* layers to study the functional aspects of telecommunication networks. We introduce a framework to map the network functions into graphs called functional topologies. This allows us to study the structural organization related to the information exchange during the execution of a network function. We also introduce a new complexity metric, i.e *functional complexity*, that quantifies how much more information exists in the interactions between nodes than it would be expected from a naive linear representation of these interactions. In the end, we show that the *functional complexity* captures the correlation between the organizational structure and the performance of different network functions (e.g. frequency allocation in cellular networks, clustering in WSNs).

Chapter 4 - Degeneracy Framework for the Analysis of In-Network Computing in IoT

In Chapter 4 we reapply the same modeling approach, i.e. modeling network functions with graphs, but instead of *Complexity*, we focus on *Degeneracy*. We start by modeling the execution of a distributed computation in an IoT network. The modeling approach allows us to provide a formal definition of degeneracy, i.e. the multiplicity of possible options - set of functional topologies - available within the network to perform the same function with a given macroscopic property (e.g. delay). Then, we propose an algorithm to enumerate all the feasible functional topologies for any physical topology. Finally, we study the computational complexity of the algorithm and the benefits of exploring degenerate computational alternatives.

Chapter 5 - Self-Organization in Cellular Networks Based on Local Interactions

We start Chapter 5 with the introduction of a simulator we developed to study self-organization and dynamics in large cellular networks. Then, we design a distributed self-organization algorithm that relies only on locally available information to minimize signaling and latency related to mobility management in cellular networks. We use the developed simulator to simulate real user behavior in a cellular network (we use BS locations and user mobility datasets for San Francisco, USA). The simulator allows us to experiment with the designed algorithm and to iterate through the three layers (i.e. *Analysis*, *Modeling* and *Design*), introduced in Chapter 2, to refine various algorithm parameters.

Chapter 6 - Conclusions and Future Work

Chapter 6 starts with the discussion of the main applications of the proposed complexity tools to study network functions. Then, we summarize the main strengths and limitations of the approaches presented in each chapter, and discuss the potential ways to improve or extend these methodologies.

1.3 Contributions

In this section we outline the main research contributions made in this thesis. The contributions range from conceptual insights into complex systems science, and the analysis and modeling of complex phenomena in communication networks, to the design of local rules of interaction that lead to self-organization. However, the general contribution lies in recognizing the problems facing future communication networks and proposing a methodology to study them. The contributions are grouped based on the chapter in which they appear.

Chapter 2 - Background

In this chapter we discuss the methodologies and tools used to understand and design communication networks as complex systems. We categorize the related work into three layers, namely *Analysis*, *Modeling* and *Design*. This systematic approach to the study of complex phenomena in communication networks allows us to examine and design micro-scale structures that lead to high performance macro-scale/network-level properties.

Chapter 3 - Functional Complexity Framework for the Analysis and Modeling of Networks

In this chapter we focus on the *Analysis* and *Modeling* layers to study the *Complexity* of structures that enable the execution of network functions. The main contributions can be summarized as follows:

- We propose a framework to model network functions as graphs called functional topologies;
- We introduce a new complexity metric that in conjunction with the functional topology allows us to quantify the organizational structure of a telecommunication network function;
- We show that the functional complexity highlights the structural patterns that lead to higher scalability and energy efficiency of a clustering algorithm implementation in WSNs, and that it captures the trade-off between these two properties;
- We correlate the characteristics of the outcome with the complex relationships that underpin the functional structure by making a connection between the implementation and the outcome of the frequency allocation function in cellular networks.

Chapter 4 - Degeneracy Framework for the Analysis of In-Network Computing in IoT

The main contributions of this chapter are:

- We propose an algorithm that allows us to enumerate all possible alternatives to compute a function in a distributed way (i.e. all functional topologies) satisfying a given delay requirement, for any physical topology;
- We provide a computational complexity analysis of the proposed algorithm, and a study of its applicability to realistic network topologies;
- We propose a formal definition of degeneracy and redundancy for distributed computing over IoT networks;
- We compare the degenerate in-network computing with a traditional multi-hop scheme that selects a single graph for the computation, showing that it is possible to significantly increase the probability of successful computation by exploiting degeneracy.

Chapter 5 - Self-Organization in Cellular Networks Based on Local Interactions

In this chapter we iterate through all three layers, i.e. *Analysis*, *Modeling* and *Design*, to study self-organization through local interactions (*Emergence*). We make the following contributions:

- We developed a simulator that allows us to study self-organization and dynamics in large cellular networks;
- We design local rules of interaction between BSs in a cellular network that lead to patterns on the network level resulting in low signaling and latency related to user mobility;
- We show that self-organization through local interactions results in high scalability and adaptability, while the time complexity of the algorithm stays very low ($O(1)$);
- We also demonstrate a centralized 4G/5G compliant approach to minimize signaling and latency related to user mobility in cellular networks.

1.4 Dissemination

In this section we list the publications disseminated during this thesis. The publications that are related to the content presented in this thesis are marked with an asterisk. The remainder of the dissemination comprises papers published within the scope of side projects.

1.4.1 Journal Articles

1. *M. Dzaferagic, N. Marchetti, I. Macaluso, "Optimizing the Mobility Management Function based on Users' Mobility Patterns", *IEEE Systems Journal*, 2019 (under review, second stage)
2. *M. Dzaferagic, N. McBride, R. Thomas, I. Macaluso, N. Marchetti, "Improving In-Network Computing in IoT Through Degeneracy", *IEEE Systems Journal*, 2019 (under review, second stage)
3. M. M. Butt, I. Dey, Member, M. Dzaferagic, M. Murphy, N. Kaminski, and N. Marchetti, "Agent-Based Modeling for Distributed Decision Support in an IoT Network", *IEEE Internet of Things Journal*, (under review, second stage)
4. *M. Dzaferagic, N. Kaminski, N. McBride, I. Macaluso, N. Marchetti, "A Functional Complexity Framework for the Analysis of Telecommunication Networks", *Oxford Academic Journal of Complex Networks*, vol. 6, no. 6, pp. 971-988, December 2018

1.4.2 Conference Papers

1. A. Bonfante, L. Galati Giordano, D. Lopez-Perez, A. Garcia-Rodriguez, G. Geraci, P. Baracca, M. Butt, M. Dzaferagic, N. Marchetti, "Performance of Massive MIMO Self-Backhauling for Ultra-Dense Small Cell Deployments", *IEEE Global Communications Conference (GLOBECOM)*, December 2018
2. *K. Pattanayak, A. Chatterjee, M. Dzaferagic, S.S. Das, N. Marchetti, "A Functional Complexity Framework for Dynamic Resource Allocation in VANETs", *International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2018
3. *M. Dzaferagic, N. Kaminski, I. Macaluso, N. Marchetti, "Relation between Functional Complexity, Scalability and Energy Efficiency in WSNs", *International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2017
4. *M. Dzaferagic, N. Kaminski, I. Macaluso, N. Marchetti, "How Functional Complexity affects the Scalability-Energy Efficiency Trade-Off of HCC WSN Clustering", *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2017

1.4.3 Patents

1. *M. Dzaferagic, I. Macaluso, N. Marchetti, "Method and System to Minimize the Signalling and Delay Caused by Mobility Management Function in Cellular Networks",

UK Patent office number GB1818733.6, filed on 16 Nov 2018

1.4.4 Awards

1. *M. Dzaferagic, C. Sexton, I. Macaluso, “High Potential Startup Idea, First Place”, *NDRC*, May 2019
2. *M. Dzaferagic, C. Sexton, I. Macaluso, “Business Plan Competition, Second Place”, *The Ireland Funds*, June 2019

2 Background

Background

Simple is better than complex.
Complex is better than complicated.

- Zen of Python, *Tim Peters*

The term *complexity* is often misused and usually implies confusion and uncertainty. That is why the expression “it is complex” is often confused with “it is complicated”. To understand the difference, we have to start from the stem of these words. The word “Complicated” is composed of the *Latin* words: **com**-(meaning: together) and **plic**-(meaning: folded,layered). Similarly, the word “Complex” is composed of two *Latin* words: **com**-(meaning: together) and **plex**-(meaning: woven). The stem of both words refers to a system that consists of multiple parts, but the second part of the word suggests that:

- Complicated system - is a folded, multilayered system that consists of multiple parts. To understand the system we can apply a reductionist approach, meaning that understanding the system is the same as understanding its individual parts.
- Complex system - is also a system that consists of multiple parts. As opposed to a complicated system, understanding the individual parts is not enough to understand the system as a whole. That is why we need a complex systems approach that, similarly to the reductionist approach, starts from the notion of multiple system parts, but instead of understanding them separately, puts the emphasis on understanding the relationships between those parts.

According to [1], in the early stage classical science rejected complexity because of three fundamental explanatory principles:

1. Universal determinism: illustrated by Laplace’s Demon, which is an articulation of causal determinism in conceptual form stating that if the precise location and momentum of every atom in the universe is known, their past and future values can be calculated from the laws of classical mechanics;
2. Reductionism: knowing any composite from only the knowledge of its basic elements;
3. Disjunction: isolating and separating cognitive difficulties from one another, which leads to separation between disciplines.

Since then, complexity has come a long way. Complexity made its grand appearance with the second law of thermodynamics stating that the total *entropy* of an isolated system can only increase or remain constant (thermodynamic equilibrium) over time, but can never decrease. This law accounts for the irreversibility of the system and the asymmetry between the future and the past, as opposite to other laws discovered till then that were in principle reversible. Hence, as the author of [1] puts it: “The arrival of disorder, dispersion, disintegration, constituted a fatal attack to the perfect, ordered, and determinist vision.” The author also highlights the importance of understanding order and disorder, and to associate them with the principles of organization. Considering the old principle that the whole is more than the sum of its parts, complex systems should not be defined only by their parts but also by the relationships between them. Hence, the notion of organization becomes crucial, because the organization of system parts results in emergent qualities appearing and inhibited qualities disappearing.

The author of [2] provides the most widely accepted definition of a complex system: “A system in which large networks of components with no central control and simple rules of operation give rise to complex collective behavior, sophisticated information processing, and adaptation via learning or evolution.” Networks are growing in size and deployment density, which makes it difficult to control the network centrally. Additionally, networks are changing from simple pipelines for information transfer to systems that generate, store, communicate and process information. Hence, the definition of a complex system in [2] is perfectly aligned with the vision of future mobile networks in this thesis, which is based on self-organization, dynamic resource allocation and adaptation. It also aligns with the research question addressed in this thesis, by putting the emphasis on the organization of system elements that underpins the collective behavior and adaptation.

What does the analysis of the micro-scale structures tell us about the macro-scale performance, and how do the local interaction rules lead to global organization/synchronization in communication networks?

Now, we will look into the tools and techniques used to analyze, model and design complex systems. We will also provide an overview of complex systems properties, e.g. complexity, degeneracy and emergence, and explain the known relationships between them.

2.1 Interdisciplinarity of Complex Systems Tools

Different sciences faced the problem of increasing complexity differently. The authors of [3] highlight that due to the evolution of all existing systems (e.g. ecological, datacenters, streaming services, financial markets) the complexity of our globalized, data-saturated present is far beyond the capabilities of conventional social science theory and methods.

This interdisciplinary field draws attention from researchers in physics, biology, mathematics, engineering and many others. Papers like [4–10] testify about the interdisciplinary nature of the complex systems analysis. In [11], the authors identify similar problems, such as issues related to system scale, discovering emerging patterns in the systems and understanding the interactions between system entities, faced by different research areas, like geography, economics, ecology, sociology, politics, etc. An interdisciplinary approach, which implies borrowing solutions from other scientific fields, would save time and effort that researchers put into solving problems. The authors of [3] explain that the complexity sciences are more a revolution in methods than theory. They emphasize that we have always known that the world was complex, but the challenge was figuring out how to model it more effectively.

Even though tools like: information theory, game theory, network science, statistical mechanics, computational modeling, distributed optimization and many others, are very important to study different system properties, none of them individually allows us to fully comprehend a complex system. For example:

1. Complexity metrics are mainly based on information theory (e.g. mutual information, entropy). However, instead of modeling the amount of randomness in the system or the amount of information exchanged between two system entities, complexity metrics measure how much the behavior of system parts deviates from the behavior that is expected from a completely random or a linear system.
2. Complex systems rely on distributed decision making, which allows us to use game theory approaches to a certain extent. However, due to the heterogeneous set of agents and rules of interaction between them it is usually impossible to describe the system with a set of selfish agents or closed form equations. Unlike game theory approaches that rely on strategic interactions between rational decision-makers, individual agents within a complex system often exhibit a behavior that would not be categorized as rational, and therefore the decision making process is not always based on the maximization of the mathematical expectation of the cost function.
3. Statistical mechanics tools allow us to study systems that are composed of a big number of entities, which is important to have a better understanding of the aggregate/average behavior of the system, but in terms of modeling and analysis of complex systems statistical treatment is sometimes not enough. Indeed the aggregate system behavior often depends on the chronology of interactions rather then the current states of the individual elements.
4. Network science allows us to model highly complex systems by focusing on the modeling of interactions rather than the individual parts themselves. Moreover, network science provides a set of tools that are helpful to understand the correlation between the system entities, but similarly to the statistical mechanics tools, it does not allow to model the interactions between the system entities on the micro scale (e.g. information exchange and node state transition) that are needed to understand the causal relationships between the system states on the macro scale (e.g. interference, clustering).

By carefully reviewing the literature on applications of complex systems science to a

Table 2.1: The classification of complex systems methodologies and tools into: *Analysis*, *Modeling* and *Design*.

Analysis	**Modeling**	**Design**
[12–28]	[29–44]	[39, 45–52]

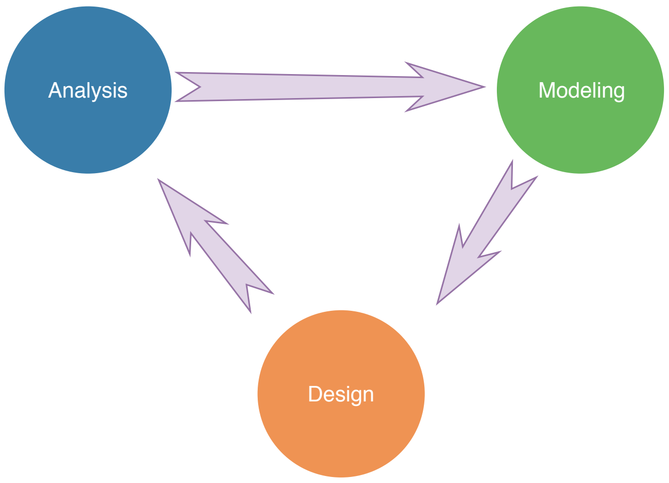


Figure 2.1: Framework to understand and design complex systems. The cyclic iteration through these three stages allows us to gradually increase our understanding of the system and reduce the impact of the various limitations of each layer, leading to better system models and design.

variety of research areas (e.g. social networks, the behavior of mobile users, network science, wireless communications), we can organize the methodologies and tools into three different layers (see Table 2.1):

1. Analysis - includes the methodologies and metrics needed to quantify the complexity of a system and comprehend the implications of those metrics on the performance of the system.
2. Modeling - includes the different approaches to system modeling (e.g. Agent-Based Modeling (ABM), Equation-Based Modeling (EBM), dynamical systems theory) and the understanding of the advantages and disadvantages of each modeling technique.
3. Design - relies on the above-mentioned layers and includes a range of system design approaches (e.g. evolutionary approaches, distributed optimization techniques, genetic algorithms).

Sections 2.1.1, 2.1.2 and 2.1.3 will provide more details related to the literature mentioned in Table 2.1.

Figure 2.1 depicts the framework to understand and design complex systems. The framework is based on the cyclic iteration through the mentioned layers. The analysis layer provides details about the relationships between system entities allowing us to create better system abstractions by using the techniques studied on the modeling layer. Those models are then used to design and improve system properties that lead to robustness, scalability and high performance. In the next subsections we will present the tools and techniques used for different stages, and discuss some of the drawbacks of those approaches. For example, different complexity metrics focus on a specific aspect of the system and ignore others, modeling techniques capture only a subset of the real world system properties and design methodologies rely on heuristics and trial and errors. Therefore, the iteration through these layers allows us to improve the understanding of the system on the analysis layer, which results in better models and consequently in a better system design.

2.1.1 Analysis

The analysis of Complex Systems includes a variety of tools and techniques that rely on information theory, network science, social science, neuroscience, etc. All of them provide a specialized set of tools that allows us to have a better understanding of how the system works and evolves over time. The combination of those techniques allows us to define complexity metrics that provide a holistic understanding of how the underlying micro level structures emerge over time and form the complex structures that are visible on the macro level.

The authors of [12, 13] emphasize the importance of the evolution of the tools we use to analyze and understand complex systems. The authors of [12] focus on social networks and user behavior. In [12], the authors propose a data-to-model process which allows them to analyze complex social interactions. This approach enables the prediction of the developments of eventual disasters in the system, which makes it possible to prepare for disaster recovery scenarios. In [13], the authors analyze the human travel patterns based on the trajectory of 100,000 mobile phone users. The understanding of the mobility patterns allows them to predict the movement and therefore the influence on spreading viruses, urban planning, mobile network planning, etc.

A graph representation of a system helps us to study the relationships between system entities, which has its application in different research areas. For example: (1) parallel computing: understanding the relationships between tasks allows us to divide those tasks into groups and process them in parallel, while minimizing the need for inter-processor communication; (2) multi-cast in Content Delivery Network (CDN): understanding the relationship between user demands and the network structure allows us to group streaming requests and minimize the load in the network. These problems are usually solved with traditional graph partitioning techniques, in which the number of partitions is known before. Community detection, by contrast, assumes that the network divides naturally into subgroups; the number and size of the groups are determined by the network itself. For example, community detection is used for studying the structure of: social networks, internet data or biochemical networks. Authors like [14–18] use community detection to analyze the structural organization of a complex system. In [14] the authors investigate the underlying interactions between mobile phone users which determine the affiliation to a community. The authors of [17, 18] study modularity, which measures the strength of division of a network into modules - non-overlapping communities. The authors of [15, 16] focus on links rather than nodes, which enables the detection of overlapping communities. Additionally, their approach allows them to analyze the hierarchical structure of a complex system.

Another approach to analyze and understand complex networks is network science. In [19–21], the authors emphasize the importance of graph theoretical tools to get a better understanding of the increasing complexity of systems. In [19], the author highlights the main difference between complex and non-complex graphs and introduces graph theoretical metrics to categorize, analyze and understand certain properties of these graphs. The author of [20] discusses how eigenvalues of the adjacency matrix and structures of graphs are interrelated. The authors of [21], discuss different tools to model and extract knowledge from complex graph structures. They mainly focus on graph mining, with the emphasis on the tools and metrics (e.g. graph generation, graph patterns, community structure, clustering, path-length) used to model complex relationships between system entities.

2.1.2 Modeling

Modeling is an inseparable part of many sciences. Each research area has its own preferred approaches to modeling. The final model is a simplification of the system that includes all the characteristics that are needed to fully describe the system within the specific context of the analysis. This means that, depending on the approach, different models can be used to describe the same system. A model is only as good as the expertise which lies behind its formulation. Each modeling approach involves its own theories and mathematical techniques. For example, we can distinguish between deterministic and stochastic models, simulation and analytical models, linear and non-linear models, single-agent and multi-agent models, etc. Rather than looking into all possible modeling approaches, we will focus on modeling approaches that can be applied to complex systems (i.e. complex adaptive approaches). The main difference between the complex adaptive and reductionist Newtonian modeling approaches is that complex adaptive modeling approaches assume that there is no predictability of the full trajectory of the system [29]. Since the discovery of the *Second Law of Thermodynamics*, the importance of the concept of irreversibility became evident. Hence, history matters and in contrast to the reductionist approach time can not easily be reversed to calculate previous system states.

An important topic in understanding different modeling techniques is the difference between inductive and deductive reasoning [30]. Inductive reasoning refers to a process in which the starting point is the observation of a phenomenon, which leads to the discovery of patterns and in the end to generalized theories. Such an approach is called “bottom-up”. On the other hand, deductive reasoning starts with a theory and follows the opposite direction as compared to inductive reasoning. Such an approach is called “top-down”.

Considering the interdisciplinarity of the tools used to analyze complex systems, naturally a spectrum of modeling approaches evolved over time ranging from system dynamics, mean field theory, Pattern-Oriented Modeling, equation-based modeling, cellular automata to ABM.

System dynamics modeling is a bottom-up computational modeling method used to understand the dynamic behavior of a complex system. As highlighted in [31], system dynamics models are not derived statistically from time-series data. Instead, they are statements about system structure and the policies that guide decisions. Therefore, the structure of a system, including its multiple-feedback-loop and the nonlinear nature, is as important in determining its behavior as the individual components themselves. This approach allows us to study the emergence of properties on the system level, that can not be found on the system elements.

As highlighted in [32], the mean-field theory idea was first introduced in physics and has been used in the context of Markov Process models of systems like plasma and dense gases where interaction between particles is weak, meaning that the strength of the interaction is inversely proportional to the size of the system. The mean-field theory approach approximates the effect of all the individuals on any given individual by a single averaged effect, which reduces a many-body problem to a one-body problem. That means that on the one hand the behavior of the system can be obtained at a relatively low cost, and on the other hand, due to the one-body approximation a lot of system characteristics are being neglected.

Pattern-Oriented Modeling is a bottom-up modeling approach that was developed for complex ecological systems. As highlighted by [33], patterns are defining characteristics of a system and often they are indicators of essential underlying processes and structures. Patterns contain information on the internal organization of a system. Pattern-Oriented Modeling is based on the assumption that observing a pattern at a specific scale is not sufficient to reduce the uncertainty in modeling the whole system. Therefore, multiple patterns, observed in real systems, are used to guide the design of model structures.

Equation-based modeling is one of the most widely used approaches to modeling in science. However, it is not the best approach to model complex systems. Examples of equation-based modeling applications to self-organization and pattern formation can be found in [34–38]. The authors of [34, 35], observe patterns by modeling the growth and differentiation of biological systems with differential equations. The authors of [36–38] model the emergence of synchronization in systems of coupled oscillators. As highlighted by [39], despite its broad application to different scientific areas, equation-based modeling is not flexible enough to model diverse components and realistic types of connectivity in real-world systems, which limits it to describing pattern formation, oscillations and other simple collective phenomena.

As defined in [40], cellular automata are mathematical models of physical systems that consist of a regular uniform lattice (or "array"), usually infinite in extent, with a discrete variable at each site ("cell"). Any physical system satisfying differential equations may be approximated as a cellular automaton by introducing finite differences and discrete variables. The dynamical Ising Model, which is usually modeled by applying a mean-field theory approach to modeling, can be modeled as a cellular automaton made non-deterministic by "noise" in the local rules at finite temperature. These models are discrete in time and space and physical quantities are completely specified by the values of the variables at each cell. The evolution of the cellular automaton is defined by the change of values of the variables at each cell. The values of the variables at each cell are affected by the values of the variables at cells in its "neighborhood" (i.e. surrounding cells) at the previous time step.

To understand a complex system we have to comprehend the underlying organizational characteristics that lead to stability, resilience, adaptability, scalability and high performance. This is not an easy task. From the modeling perspective, the first problem that we face is how to create a model that covers all the essential properties that affect the operation of the system. This problem can be addressed by using one or combining multiple of the abovementioned modeling techniques. Additionally, we need a computational form of modeling that is flexible enough to support the whole spectrum of different models (e.g. homogeneous and heterogeneous agents, static and dynamic agents, evolutionary agents). That is where ABM can provide additional value. As defined by [41], an agent is an autonomous computational individual or object with particular properties and actions. ABM is a form of computational modeling whereby a phenomenon is modeled in terms of agents and their interactions, allowing us to avoid simplification of problems that are essentially complex.

According to [41], one of the core ideas of ABM is that since complex systems consist of a large number of elements, they can effectively be modeled with agents, an environment, and a description of agent-agent and agent-environment interactions, and there are eight main uses of ABM:

1. Description - a model is descriptive of a real-world system;
2. Explanation - models are explanatory in that they point out the essential mechanisms underlying a phenomenon;
3. Experimentation - models can be run repeatedly to note variations in their dynamics and in their outputs;
4. Providing sources of analogy - since models are simplifications of reality, they enable us to find similarities with other such simplifications even if the modeled phenomena are apparently very different;
5. Communication/education - models provide us with an educational tool that allows us to communicate the ideas of how a complex system works;
6. Providing focal objects or centerpieces for scientific dialogue - the simplification of a system through modeling allows us to focus on the essential mechanisms that define the system and to test alternative hypotheses;
7. As thought experiments - models can present new phenomena that are not necessarily about real-world phenomenon, allowing us to think about abstract ideas that might find its application to science and engineering (e.g. cellular automata, fractals,

particle swarms);

8. Prediction - models are often used to experiment and study the behavior of real world systems, allowing us to make predictions about the systems behavior in the future and its response to different inputs.

Choosing the proper modeling technique is of crucial importance when analyzing a complex system. Depending on the system properties each modeling technique has its advantages and disadvantages. For example, equation-based modeling suits problems for which the aggregate behavior is well known and therefore allows us to test our hypothesis against the aggregate output of the system. On the other hand, equation-based modeling has a continuous nature, which creates problems due to the discrete nature of many real-world problems. Additionally, equation-based models typically make assumptions of homogeneity in order to model the aggregate behavior of a system. In contrast, ABM models individuals and therefore it does not require knowledge of the aggregate phenomena and can model a heterogeneous population, allowing the interactions between the agents to be complex [41]. Due to the infinite possibility of different agent types, simple rules to describe their interactions allow us to create a very rich tapestry of interactions. ABM also generates more detailed results (e.g. aggregate and individual results) compared to equation-based modeling that only provides results related to the aggregate behavior. Due to the possibility to keep a history of interactions, ABM allows individual agents to change their behavior and strategies over time based on past events. Another important property of ABM is the rich conception of time. While modeling complex systems, we have to keep in mind that some events occur before others and that the chronology of events can affect the final outcome of the system. Hence, as highlighted by [41], ABM allows us to move beyond a static snapshot of the system and toward a dynamic understanding of the system's behaviors. In contrast, problems with large number of homogeneous agents are often better modeled using an aggregate solution like cellular automaton, mean field theory or system dynamics modeling.

Considering the discussion above, the best approach to modeling is a combination of all available tools. As highlighted by [33], finding the optimal level of resolution in a bottom-up model's structure is a fundamental problem. Having a model that is too simple means neglecting essential mechanisms of the real system. However, a model that is too detailed results in an analysis that will be unmanageable and bogged down in detail. Therefore, combining for example Pattern-Oriented Modeling and ABM allows us to create a model with a resolution that fits the complexity of the system. Pattern-Oriented Modeling ties the model's structure to the internal organization of the real system by observing multiple patterns at different system scales. Those observations can be used to define the behavior of agents and their interaction in ABM.

ABM was used by the authors of [42] to study social networks and user behavior. The authors propose an agent-based framework for modeling competitive and cooperative behavior under conflict (i.e. Common-Pool Resource (CPR) Dilemma). The framework allows them to study how and why we reach certain outcomes, and to determine the conditions needed to achieve desirable outcomes in a complex system. ABM is being adopted by the telecommunication community as well. Papers like [43, 44, 53] emphasize the growing complexity of the next generation of communication networks, which demands the development of new frameworks and tools to model and analyze these networks. The authors of [43] propose an ABM framework to model and analyze the performance of sensor networks. While the authors of [43] focus more on the theory behind complex systems and ABM as a tool to model and analyze such systems, the authors of [44] apply ABM to a real world Internet of Things (IoT) problem. In [44], the authors analyze the impact of Medium Access Control (MAC) protocol selection on communication performance in terms of spectrum utilization and accuracy of information. In [53], the authors present an ABM approach to study the secondary use spectrum market, which allows incumbent spectrum users to lease unused portions of their assigned spectrum to third parties.

2.1.3 Design

As highlighted at the end of Section 2.1, the design layer relies on the Analysis and Modeling layers. These two layers are used to extract knowledge about existing complex systems - Analysis layer - that enables the modeling of the system structure and the relationships between system entities - Modeling layer. Due to the emergent characteristics of complex systems (i.e. the system exhibits properties that do not exist on the micro-level), the “reverse-engineering” of the rules that lead to the collective system-level behavior is very difficult - often impossible. Hence, the difficulty of designing a complex system lies in the fact that as opposed to the common engineering approach of top-down system design, a complex system has to be built bottom-up. Papers like [39, 45–48] highlight the difficulties related to complex systems design. The interactions between entities of a complex system are often represented as swarm behavior, hence, the “reverse-engineering” of the rules that lead to the collective system-level behavior becomes the “reverse-engineering” of micro-level interactions between agents that generate a macro-level pattern.

The authors of [45–47] propose an approach that relies on finding new swarms and rules that generate a particular swarm by evolving the rules for known swarms. However, they agree that designing the individual level rules of behavior that will result in a desired collective pattern in a group of agents is difficult because the group’s aggregate-level behavior may not be easy to predict or infer from the individuals’ rules. Computational modeling techniques enable the reconstruction of the forward mapping, i.e. from micro-rules to macro-behavior, but the inverse problem, i.e. finding micro-rules that lead to an interesting macro-behavior is challenging, especially because the meaning of “interesting” macro-behavior may not be known ahead of time. The authors of [46] propose an approach inspired by evolutionary computational algorithms. This exploratory design method starts with a set of known original configurations, that can be either slightly mutated or combined with other configurations (crossover) to produce a new generation of the system configuration. The new generation of configurations is being evaluated and the best ones are kept for the next evolutionary cycle, i.e. survival of the fittest. This approach allows us to start from one set of known rules and discover a whole new set of “interesting” macro-behaviors.

In [39], the authors highlight the importance of self-organization to pattern formation that occurs by the cooperative behavior of individual entities. In other words, the pattern formation occurs without any external influence. The authors single out 5 different approaches to complex systems design:

1. Analytical approach: relies on an analytical model of the system, i.e. differential equations representation. The authors highlight that this approach to model a moderately complex system is not feasible (e.g. the Lotka-Volterra model [49]). The best application of the analytical approach is to help discover certain aspects of the system by introducing assumptions that allow us to analyze a simplified version of the model.
2. Reference Design: relies on the existing examples of self-organization in domains like: biology, physics, mathematics, economics. Specifically, the bottom-up approach relies on the abstract study of existing systems, meaning that the first attempts to reconstruct an existing system are done without an application in mind. Afterward, results of these studies are used in technical applications (e.g. artificial neural networks, routing algorithms based on ant foraging behavior [50]).
3. Trial and Error: Due to the high-dimensional search space of complex systems models, a simple Monte-Carlo method, with random configurations being tested until a desired global behavior is achieved is not feasible. The work in [51, 52], proves that finding meaningful properties of interaction between system entities that will result in a predictable emergent behavior is difficult.
4. Evolutionary Algorithms: As previously mentioned evolutionary algorithms (e.g. genetic algorithms, simulated annealing, swarm-based optimization) have been used by multiple authors ([45–47]) to guide the search for a configuration on the micro-level that results in a desired system behavior.
5. Markov Models and Finite State Machines: relies on deriving local interaction rules by learning from a reference model. Similarly to the reference design, depending on the complexity of the system, it might be impossible to create a model that is detailed enough to be analyzed using Markovian analysis and then rebuilt in a Finite State Machine.

The authors of [48] focus on the application of complex systems design to wireless networks. They highlight that designing mutual synchronization in wireless networks requires consideration of the dynamic behavior of a possibly large set of coupled oscillators, which is often hard or even impossible to do and calls for the stability analysis of a system of coupled linear or nonlinear equations. The authors also emphasize that the design of such systems has to consider energy efficiency, scalability and application specificity.

2.2 Complex Systems Science Principles

The study of complex systems is built around the understanding of the relationships between system parts that lead to a system behavior that exhibits stability, resilience, adaptability, scalability and exceptional performance. These system properties can be studied through a combination of the different tools mentioned in Section 2.1, which allows us to define:

1. Complexity: quantifies the organization of non-repeating structures in a system. In other words, complexity allows us to quantify where a system is positioned between a random and a completely regular system. The authors of [54] use language as an example of a random, structured and complex system: "Neither a random string nor a periodically repeating string of letters is complex, while a string of English text certainly is."
2. Degeneracy: is the ability of structurally different elements to perform the same function. As highlighted in [55], due to the possibility that structurally different elements produce different outputs in different contexts, degeneracy should be distinguished from redundancy, which occurs when the same function is performed by identical elements.
3. Emergence: refers to the rise of novel and coherent structures, patterns, and properties during the process of self-organization in complex networks [56]. These emergent phenomena are conceptualized on the macro-level, without traces of them appearing on the micro-level components and processes out of which they arise.

These three properties are not orthogonal and causal relationships can be defined to explain how one affects the others and vice versa (shown in Figure 2.2 - adopted from [57]). The authors of [55, 57–59] highlight the central role of degeneracy in complex systems, showing the positive correlation between degeneracy and complexity, evolvability and robustness.

The concept of organization underpins all three complex systems principles and explains the relationships between them. The system structure enables the propagation of information throughout the system, which allows the local patterns to propagate and build higher levels of organization leading to complexity, degeneracy and emergence in the system.

2.2.1 Complexity

As highlighted by the authors of [60], discovering organization in nature is one of science's central goals. The authors emphasize that these efforts had their origin in randomness. However, it was realized that measures of randomness do not capture the property of organization. This led to the development of measures that capture a system's complexity - organization, structure, memory, symmetry, and pattern. Complexity allows us to quantify the hidden micro-level relationships between system parts that result in the system properties obtainable on the macro-level. As highlighted in [54, 61], we believe that we can recognize complexity when we see it, but complexity is an attribute that is often without

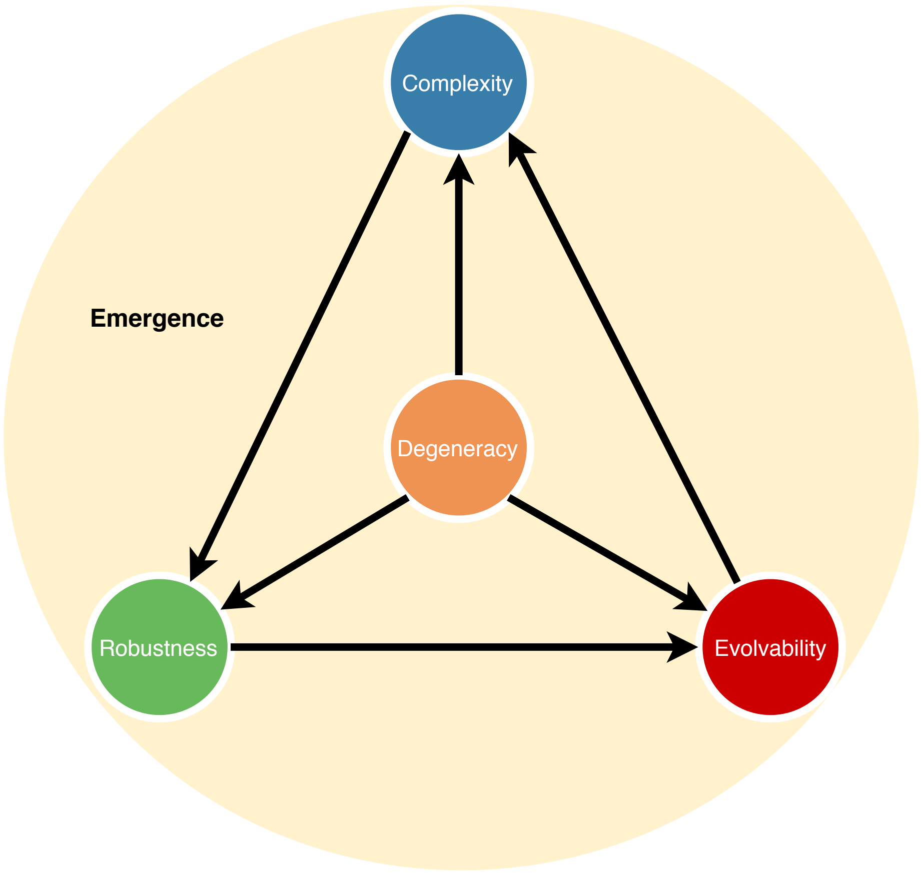


Figure 2.2: Relationship between complexity, degeneracy and emergence. Degeneracy and complexity are the source of robustness; degeneracy and evolvability lead to complexity; emergence refers to the rise of coherent structures, patterns and properties that are a prerequisite of complexity, degeneracy, evolvability and robustness.

any conceptual clarity or quantification. The authors emphasize that a complex system lives in-between a random and a completely regular system, leading to the conclusion that a lot of the so called complexity metrics (e.g. Kolmogorov complexity in algorithmic information theory, dimensional complexity in neurobiology) do not measure "true complexity" because they do not attain small values for both random and regular systems. Random systems are systems with no structure at any level, which results in high entropy and low complexity. On the other hand, regular systems exhibit low entropy and low complexity due to the repetition of structures on multiple levels. Therefore, it is obvious that complexity and entropy are two distinct quantities. As highlighted in [62], entropy captures the disorder and inhomogeneity rather than the correlation and structure of a system.

Similarly to the work in [54] in which the authors talk about completely regular, random and complex systems, the author of [63] speaks about simple, disorganized complex and organized complex systems. He highlights that the emphasis of science (e.g. calculus, statistics) was finding ways to model the first two, while the problem of organized complexity has not been tackled properly before. He also emphasizes that solving problems related to organized complexity is challenging because:

- Similarly to disorganized systems, organized complex systems are comprised of a large number of dynamic variables which make microscopic predictions impossible;
- The larger, self-organizing, emergent whole created by the system parts is dependent and comprised of multiple causal models.

Hence, as highlighted in Section 2.1, organized complex systems can not be simply modeled by mathematical formulas, qualitative description or statistics. Instead, new approaches to science, along with new modeling methods are needed. The connection that the author of [63] makes between system organization on the micro and macro level highlights the relationship between complexity metrics and the emergent behavior of those systems.

The author of [64] explains that when we are trying to quantify the complexity of a system, three main questions have to be answered:

1. How hard is it to describe?
2. How hard is it to create?
3. What is its degree of organization?

These questions can then be used to classify different measures of complexity. The degree of difficulty involved in completely describing the system (typically quantified in bits) includes measures like: information, entropy, code length, etc. The difficulty of creation (typically quantified by time and energy) includes measures like: computational complexity, cost, crypticity, etc. The degree of organization includes measures like: excess entropy, hierarchical complexity, tree subgraph diversity, correlation, mutual information. Not all of the mentioned metrics are a measure of complexity per se, but all of them can be used to define different complexity metrics. Some examples include:

- Excess entropy is an information-theoretic measure of complexity. The excess entropy measures the amount of apparent randomness at the micro-level that is “explained away” by considering correlations over larger and larger blocks [62]. It is low for completely random and fully structured system configurations and high for structures that exhibit a certain level of organization without pattern repetition on different system scales.
- Neural complexity measures the amount and heterogeneity of statistical correlations within a neural system in terms of the mutual information between subsets of its units [7]. It captures the interplay between global integration and functional segregation. It is high for a system in which functional segregation coexists with integration and low when the components of a system are either completely independent (segregated) or completely dependent (integrated).
- Matching complexity reflects the change in *neural complexity* that occurs after a neural system receives signals from the environment [65]. It measures how well the ensemble of intrinsic correlations within a neural system fits the statistical structure of the sensory input. *Matching complexity* is low when the intrinsic connectivity of a simulated cortical area is randomly organized and high when the intrinsic connectivity

is modified so as to differently amplify those intrinsic correlations that happen to be enhanced by sensory input.

- Statistical measure of complexity is defined as the product of two probabilistic measures: disequilibrium and entropy [8]. Disequilibrium gives an idea of the probabilistic hierarchy of a system, and it is high for highly regular systems and low for disordered/random systems. On the other hand, entropy is low for ordered systems and high for disordered systems. The product of these two values provides a complexity measure that is high for systems that are in-between ordered and disordered systems.
- Self-dissimilarity measures the amount of extra information using a maximum entropy inference of the pattern at one scale, based on the provided pattern on another scale [66]. It characterizes a system's complexity in terms of how the inferences about the whole system differ from one another as one varies the information-gathering space.

The account of complexity metrics does not stop here. Authors like [67–72] have taken different approaches by relating complexity to mathematical formulations of thermodynamics or statistical entropy and information. The approach in this thesis builds on the results of those authors and takes a step further into understanding and quantifying the complexity of structures underlying the functions in telecommunication networks.

2.2.2 Degeneracy

Degeneracy - a ubiquitous characteristic of biological systems existing at all levels of biological organization - refers to the ability of elements that are structurally different to perform the same function or yield the same output [73]. Hence, degenerate systems enjoy two primary attributes:

1. System robustness without compromising efficiency;
2. Increased adaptability based on providing multiple options to deal with changes.

Due to its property of providing alternative ways of performing the same functionality, degeneracy is often confused with redundancy. The authors of [55] highlight that in technical usage, redundancy refers to duplication or repetition of elements within electronic or mechanical components to provide alternative functional channels in case of failure and in information theory, it refers to the repetition of parts or all of a message to circumvent transmission error. Unlike redundant elements, degenerate elements are structurally different and they may produce different outputs in different contexts. In [55], the authors express the degeneracy of a system as a function of its redundancy, showing that degeneracy is high when the redundancy of the system with respect to the output is high and at the same time the average redundancy for small subsets of the system is lower than it would be expected from a linear increase over increasing subset sizes. This suggest that the relationship between the properties on the micro- and macro-scale are non-linear, which lead the authors to the analysis of the correlation between complexity and degeneracy. They show that systems that exhibit high degeneracy with respect to a given output also show an increase in matching complexity, suggesting that high complexity values may reflect the statistical structure of a complex environment as well as responses to various selective pressures with multiple alternatives yielding adaptive outputs.

In [57], the author shows that degeneracy plays a central role in the evolution and robustness of complex systems. The emphasis is on the role of degeneracy as an effective mechanism for creating distributed robust systems, enabling long-term evolvability of the system. In other words, degeneracy enables robustness and evolution through diversity.

The authors of [59] refer to degeneracy as distributed redundancy and highlight that it contributes significantly to the system resilience. Similarly to the work in [55], the authors emphasize the difference between regular redundancy and degeneracy, emphasizing that the distinction is clearly seen in the comparison between design and selection. Redundancy is built into the design to provide fail-safe operation, ruling out the possibility of adaptation as a response to failure. On the other hand, degeneracy originates from evolution and selection, making it an emergent property of complex systems. Additionally, according to their definition, degeneracy is high when the mutual information between the system and the output is high and when smaller subsets of components are contributing more to the output than the larger one. This makes it fundamentally different compared to redundancy (where the components are structurally identical) which is high if the sum of the mutual information between each component and the output is much larger than the mutual information between the system and the output.

The conditional interchangeability of elements within the system and the ability to reconfigure the system to adjust in the event of change have long been part of cellular and wireless communication systems, though typically not recognised as degeneracy. For example, the overlapping communication system resulting from license-exempt networks and cellular networks is an example of degeneracy 'between different networks' [74]. The authors of [75] employ a statistical mechanics definition of degeneracy to analyze the number of frequency allocation configurations which possess a specified interference level in telecommunication networks. Looking to the future of networks and communication systems, the concept of degeneracy will become increasingly important both within specific networks and between different networks. For example:

- Increasingly heterogeneous networks open the possibility of functional degeneracy between the different tiers of networks (e.g. macro tier, low power tier etc.).
- Increasing virtualisation of networks creates more possibility to combine chains of virtual functions in different ways, allowing structurally different software entities to perform the same function.
- Advances in software defined networking, software defined radio, and cognitive radio support much greater reconfigurability of networks, enabling the dynamic selection of resources to perform the same function.

This leads us to believe that there is great potential to rethink design and operation of networks. Indeed as degenerate systems "reside on richly connected neutral plateaux" of the design space, as opposed to traditionally designed systems which operate on isolated peaks [76], degeneracy allows us to rethink redundancy on the basis of increased stability inherently imbued at design time, leading to vast implications for network design and operation.

2.2.3 Emergence

Emergence can be defined as the rise of novel and coherent structures, patterns, and properties through the interactions of multiple distributed elements [41]. The author of [77] emphasizes that those system level properties arise spontaneously from the interactions of system entities and that these interactions are completely distributed, meaning that no central authority exists that would guide the system behavior - the system “self-organizes”. Hence, the emergent properties of the system are dynamic and perturbing them often results in them dynamically reforming. Since probabilistic and random processes are essential to the creation of order in complex systems, in [41], the authors propose to think about emergent structures not as entities, but rather, as processes holding the structure in place.

Experience shows that when an explanation of a complex system is achieved in terms of its simpler, lower level components, what is gained is not simply knowledge of the low-level theories but rather a new interdisciplinary theory [78]. As previously emphasized in Section 2.1.3, the “reverse-engineering” of the rules that lead to the collective system-level behavior is very difficult. This nontrivial relation between microscopic (local) actions and macroscopic (global) system outcomes is one of the central questions related to emergence and is named Micro-Macro Link [77]. Therefore, the emergent order cannot be linked in any obvious way to system’s lower level parts [79–81]. Solving the Micro-Macro Link problem and understanding emergence in complex systems requires the understanding of the topologies and major chains of causal connections to identify the possible types and forms of emergent phenomena which can appear in the system. In particular, one refers to downward causation as the case where emergent features gain a degree of autonomy from lower levels, and, because of that autonomy, explanations of what happens at lower levels may sometimes have to refer to events at the emergent level [82–84].

As highlighted by the authors of [85], the term self-organization occurs in many branches of science (e.g. biology, chemistry, communication networks), but still without a commonly accepted definition in the literature. The definition provided by the authors of [85] is broad enough to hold in different sciences, and aligns with the work in this thesis: “A system is self-organized if it is organized without any external or central dedicated control entity, meaning that the individual entities interact directly with each other in a distributed peer-to-peer fashion.”

In general, the relationship between the topology structure and the propagation of information leading to emergence in complex networks is not well studied. The author of [86] suggests that so far the research has been focusing on regular network topologies (e.g. chains, grids, lattices and fully-connected graphs) to allow the study of complexity caused by the non-linear dynamics of the nodes, without bothering about the additional complexity in the network structure itself. Papers like [87–91] emphasize the importance that structure has in achieving emergence. The correlation between micro-level entities is represented by a graph, and the analysis is focused on the impact that different graph properties have on the propagation of information throughout the network. [87] introduces coupling strategies for coupled oscillators showing that sync/emergence depends on network properties, i.e. node number, average node degree. In particular [88] found that small world properties, such as short average path length and high clustering, imply a high degree of emergence ("sync").

Due to the difficulties related to the Micro-Macro Link problem and the prediction of emergent properties based on local interactions, according to Section 2.1.3, the design of such systems requires a bottom-up approach. There are severe engineering challenges associated with bottom-up approaches, e.g. how to choose local rules and which neighborhood they shall act upon, how to estimate and feedback the system state, how to model and account for non-linearity in system parts' interactions and for system-wide correlations. When referring to non-linearity in this thesis, we mainly focus on the spread of information throughout the system. The system's non-linearity and consequent emergence, might make system operation and its evolution in time difficult to predict and control. In communication networks, it can cause problems in case of spreading the effect of faulty node behavior or the spread of a malicious piece of code [12, 13]. Similar mechanisms are common in nature (examples include bird flocks, ant colonies, etc.). However, engineers do not have the time and number of samples available to natural evolution and suitable fall-back mechanisms have to be designed. Hence, engineers rely on the analysis, modeling and design approaches discussed in Section 2.1. On the other hand, the concept of non-linearity in terms of information spreading does not necessarily imply problems in the system. It also benefits the information accessibility and can lead to faster convergence of distributed self-organizing algorithms [28].

2.2.4 The Network as A Complex System

As previously highlighted, unlike the existing literature that studies the complexity of the user behavior and the impact that external factors have on the network, in this thesis we study the communication network itself as a complex system. We focus on studying the organizational structure of communication networks that affects the execution of network functions by studying their complexity, degeneracy and the principles of emergence.

Authors of [22–28] apply complex systems principles to understand different phenomena in communication networks. The authors of [22, 23] analyze complex phenomena in telecommunication networks (spreading patterns of mobile viruses and connection strengths between nodes in a social network) which are the result of complex user behavior. The authors of [24] study the correlation between the structure of a mobile phone network and the persistence of its links, highlighting that persistent links tend to be reciprocal and are more common for nodes with low degree and high clustering. The authors of [25] study the human dynamics from mobile phone records. This studies reveal the mean collective behavior at large scale and focus on the study of anomalies. These studies of human dynamics have direct implications on spreading phenomena in networks. However, unlike the work in these papers that focuses on the complexity of the user behavior, the work in this thesis examines the complexity, degeneracy and emergence of the operation of communication networks themselves. The authors of [27, 28] analyze the complexity of outcomes of a self-organizing frequency allocation algorithm. In [28], they analyzed the relationship between robustness and complexity of the outcome. More precisely, the work in [27, 28] does not focus on the interactions between network nodes. They rather focus on the patterns that emerge from those interactions, i.e. they study the patterns after the self-organizing allocation algorithm converges. On the other hand, we study the interaction between the network nodes during the execution of network functions and its effects on the overall network operation.

Studies focusing on the concept of degeneracy have mainly been related to biological systems. However, authors like [74, 75] apply different definitions of degeneracy to artificial systems. In [74], the author discusses the importance of interchangeably using different networks to provide reliable and adaptable communication services. In [75] the authors study the multiplicity of frequency allocations that possess the same level of interference in wireless networks. Unlike the approach taken in these papers, we focus on the structure that underpins both communication and computation, by studying the multiplicity of computational graphs that can be created on top of the same physical topology. In other words, we study the potential of the physical topology to accommodate a wide variety of alternatives to perform the same computation.

As highlighted in Section 2.1.3 and Section 2.2.3, there has been a lot of studies focusing on distributed self-organizing techniques. However, due to the densification of communication networks (i.e. more nodes being deployed, and more services running on top of those deployments) and the evolution of business models that are being introduced (e.g. shared infrastructure, dynamic up- and down-scaling of resources), the need for scalability, collaboration and cooperation when designing new protocols is more important than ever. Therefore, we focus on the development of algorithms that rely on very simple actions and reactions to the changes in the environment. These algorithms are based on very simple rules of interaction (i.e. cooperation and collaboration) that lead to desirable network states and therefore exhibit high scalability.

2.3 The Role of Organization

One of the key questions in complex systems science relates to the degree of organization of a system, in terms of both: (1) the difficulty in describing its organizational structure; (2) the amount of information shared between the parts of the system as a result of the organizational structure. Sections 2.1 and 2.2 have one goal in mind: "Understanding the tools and techniques to analyze, model and design structures that enable the emergence of complex and degenerate network properties." Why is network anatomy so important to characterize? Because structure always affects function [86].

Many attempts to capture the interactions between system entities, such as network science, graph signal processing or information theory, highlight the significant impact that structure has on the system operation. Examples include road networks, railway networks, social network, power grids. The structure of road networks affects the traffic flow and solving congestion problems is often related to rethinking the traffic flow and structure rather then simply adding lanes to the roads. The railway network is another great example of how structure affects performance. Increasing the speed of the trains speeds up the transportation between two points that are directly connected, but the structure of the whole network affects the transportation bandwidth and traffic flow between any two points in the network. The structure of social networks affects the spread of information and viruses. The structure of power grids affects stability and robustness of the system.

Traditionally, communication networks have been relying on rigid hierarchical architectures. Considering the limited service and resource demands in the past, this approach worked very well to date. Now, with the increasing demand for resources (e.g. computing, storage, spectrum) and the variety of services ranging from simple voice calls to self-driving cars, augmented reality and a connected world with the introduction of IoT, the structures underlying the communication networks have to be designed to support the dynamic resource allocation and service behavior that is being executed on top of them. As highlighted in Sections 2.1 and 2.2, complex systems science provides the necessary tools to understand and design such structures.

Why is Structure Important to be Characterized in Communication Networks?

The transition of humanity into the Information Age has precipitated the need of new paradigms to comprehend and overcome a new set of challenges. Specifically, the telecommunications networks that underpin modern societies represent some of the largest scale construction and deployment efforts ever attempted by humanity, with renovations occurring nearly continuously over the course of decades. The result is networks that consist of numerous subsections, each of which following its own trajectory of development, commingled into a complex cacophony. Considering the high degree of heterogeneity and dense interplay of network elements in proposed 5G and beyond and IoT systems, achieving holistic understanding of network operation is poised to become an even more challenging prospect in the near future.

A few emerging trends confirm the picture just drawn. Mobile and wireless networks are getting denser and more heterogeneous in nature. Nodes in the network vary hugely in form and functionality - ranging from tiny simple sensors to sophisticated cognitive entities.

There is a wider range of node and network-wide parameters to set, many of which are interdependent and impact heavily on network performance. Networks are becoming more and more adaptive and dynamic, and many parameters are set during run-time in response to changing contexts. As networks evolve, all of the above issues become more exaggerated - e.g. 5G and beyond networks are seeing more antennas, more Base Stations (BSs) and devices, more modes of operation, more variability, and more dynamism. In a world like that, there is no way to systematically capture network behavior. There is no straightforward network theory or information theoretic approach that can be used to describe the overall network or the interplay between the different networks.

Complex systems science has found its way into different branches of information and communications technology, e.g. social networks, artificial virus spreading, analysis of mobile phone user behaviors. The literature mainly draws on experiences from network science, focusing on the analysis of complex behaviors of users (e.g. user mobility in networks, virus spreading, friendship relationships in social networks) [17, 18, 22, 25]. The interactions are usually represented by a graph, and the analysis focuses on the structural characteristics of those graphs, e.g. graph theory metrics (e.g. degree distribution, centrality), spectral characteristics (e.g. community detection). Additionally, authors like [27, 28] explore frequency planning in cellular networks from a complex systems perspective. They employ excess entropy to quantify the complexity of the outcome of different implementations of frequency allocation algorithms. The authors capture the structural patterns of frequency assignments in a two dimensional space represented by a cellular automaton. This has important implications for design and roll-out phase of networks, deployment of small cells, and network operation.

The application of complex systems approaches to capture structure in telecommunication networks does not stop at the analysis layer. In [44, 53, 92, 93], conceptual structure of networks informs an ABM paradigm to examine the interactions between the different entities that shape the network. In particular, the authors of [92] apply ABM to self-organizing mobile nodes and peer to peer networks. In [53, 93], ABM is used to study the conditions under which secondary spectrum markets - incumbent spectrum users lease their unused spectrum to third parties - are likely to emerge. ABM is used to investigate the impact of several MAC component technologies on the Key Performance Indicators (KPIs) of both telecom networks and applications, for example in the case of a wireless sensor network aiding an IoT system [44].

Finally, in addition to the research that has been focusing on the analysis and modeling of complex properties in communication networks, work has been done on the design layer as well. In particular, the complex systems research was focused on the design of self-organizing algorithms and emergent behavior of communication networks. Self-organization in networks has been identified by the 3rd Generation Partnership Project (3GPP) as one of the key concepts to reduce the operating cost associated with the management of a large number of nodes. The approach to self-organization taken by the 3GPP in [94] is conservative and only scrapes the surface of the huge potential of this concept. As highlighted in [94], 3GPP Self-Organizing Network (SON) solutions tackle self-configuration, self-optimization and self-healing procedures in networks. However, due to the pre-planning of conflict resolution procedures and the isolation between SON functionalities, the benefits of self-organization are limited and mainly serve as support features for network engineers. Unlike the 3GPP approach to self-organization, we take a broader approach to SON that is based on the concept of emergence in complex systems science. This allows us to focus on the group behavior that arises from the local interactions and gives rise to a desired network state without the consideration of every individual scenario. In [87], the authors introduce a family of coupling strategies for pulse-coupled oscillators and prove that synchronization emerges for systems with arbitrary connected and dynamic topologies. However, the authors explain that structure significantly affects the synchronization, i.e. the increasing average node degree and the increasing network size reduce the convergence time. Authors like [95–98], focus on the design of self-organizing algorithms for time synchronization in wireless systems, resource scheduling and reducing energy consumption in sensor networks. The authors rely on reference designs inspired by synchronization in nature (e.g. the fireflies in South-East of Asia) to develop protocols to achieve emergent behaviors in communication networks.

3 Functional Complexity Framework for the Analysis and Modeling of Networks

Functional Complexity Framework for the Analysis and Modeling of Networks

3.1 Introduction

The increasing size and density of communication networks precipitates the need of new paradigms to comprehend and overcome a new set of challenges. The increasing demand for resources affects both the network architecture and the implementation of network functions. The traditionally rigid hierarchical - hardware first - architecture is evolving into a more flat and flexible - software first - implementation. The implementation of network functions has to follow the trend of the underlying architecture to support the needed flexibility.

Current approaches focus on making nodes more capable and intelligent, which makes nodes adapt to changes in the network independently. This may result in non-desirable behaviors, and implicitly limits the scalability and stability of the adopted techniques [99, 100]. Even though successful applications of self-organizing and distributed optimization techniques exist, the ongoing densification of the node deployment, the wide range of services running on top of these infrastructures and the diversification of the business models that are being introduced in networks leads to the need to put more emphasize on scalability, collaboration and cooperation when designing future networks. Cost of devices and energy consumption will play a huge role in such ultra dense networks, therefore cheaper nodes executing less sophisticated computations will be a crucial component of the next generation of communication networks. Additionally, limiting the research on the advances related to the node functionality fails to take advantage of the potential that lies in the collaboration between the nodes. In other words, the collaborative approach focuses on network intelligence rather than the intelligence of single nodes, enabling scalability, adaptability and resilience by design.

Every telecommunication network is designed to provide different services. Network functions are the building blocks of these services. Understanding the mechanisms that provide network functions implies understanding the function and thus the network itself. The implementation of network functions (e.g. frequency allocation and mobility management in cellular networks, clustering in Wireless Sensor Networks (WSNs)) usually involves the interaction between multiple nodes. Authors like [14–21] highlight different approaches

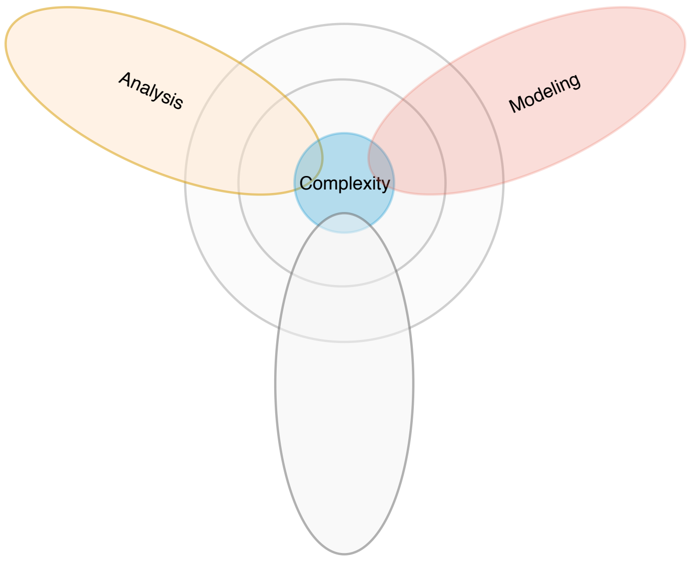


Figure 3.1: Thesis roadmap: In this Chapter we focus on the *Analysis* and *Modeling* layers to study *Complexity*.

to model and analyze interactions between system entities with graphs. The organizational structure is then analyzed by studying different graph metrics, e.g. clustering, degree distribution, average path length, or discovering patterns and communities within those graphs. We take a similar approach to modeling the interaction between functional entities in a communication network. As opposed to the network science approach in [14–21], our approach to analyze the organizational structure is based on information theory and complexity analysis.

Other authors, like [22, 23, 27, 28], have been looking into the analysis and modeling of communication networks as complex systems. The authors of [22, 23] focus on the complex behavior of users, and its impact on spreading patterns of mobile viruses. In [27, 28], the authors study the relationship between excess entropy of the outcomes of a self-organizing frequency allocation algorithm and robustness. Similarly to the work in [27, 28], we also study network functions and their complexity. However, instead of looking at the outcome of the function, we rather focus on the implementation, i.e. the structural organization related to the information exchange during the execution of the network function.

Scope and Contribution

The work in this Chapter is guided by the following question: “How to quantify the impact that micro-scale structures have on the macro-scale performance of communication networks?” In other words, we study how the probability of interaction between functional parts in communication networks (micro-scale) is affected by the underlying structure, and then we correlated these measures with well known network Key Performance Indicators (KPIs) (macro-scale performance). Building on the tools mentioned in Section 2.1 and the complex systems principles highlighted in Section 2.2, the work in this Chapter focuses on the *Analysis* and *Modeling* layers to study the *Complexity* of structures that enable the execution of network functions (Figure 3.1).

In order to analyze functional aspects of a telecommunication network we introduce a framework to map the network function into a functional topology. The functional topology enables a complex systems approach to model and analyze functions of telecommunications networks. This framework isolates and analyzes a network function as a complex system. Starting from a complexity metric proposed in neurobiology [7] - *neural complexity* - we also define a complexity metric C_F (*functional complexity*) which allows us to quantify the variety of structural patterns and the roles of nodes in the functional topology. With the term "complexity" we refer to a specific set of complex systems science quantities, related to the interactions between functional entities (rather than to the entities themselves). Specifically, the proposed metric analyzes the underlying communication between network entities which provide the network function.

Our complexity metric provides a wholly new approach to study the operation of telecommunication networks. We study the relationship between C_F and graph structures by analyzing graph theory metrics in order to recognize complex organizations. C_F is equal to zero for both a full mesh topology and a disconnected topology, aligning with the main principle of complexity [54] stating that complex systems live in-between a completely regular and a random system. We show that complexity is high for a structure with shorter average path length and higher average clustering coefficient, which aligns with the work in [89]. We associate the functional complexity, robustness and response to changes that may appear in the system configuration. We also make a connection between the implementation and the outcome of a network function which correlates the characteristics of the outcome with the complex relationships that underpin the functional structure.

We demonstrate the application of our framework to model and analyze clustering in WSNs. We employ the functional complexity metric (C_F) to quantify the potential of the functional topology to transport information. Our analysis highlights the trade-off between scalability and energy efficiency, showing that higher values of C_F indicate higher scalability and lower energy efficiency.

The main contributions of this Chapter are:

- We introduce a framework which enables the functional analysis of a telecommunication network;
- We provide a new complexity metric that quantifies the organizational structure of a telecommunication network function;
- We provide several examples that show how to map a network function into a functional topology;
- We correlate the characteristics of the outcome with the complex relationships that underpin the functional structure by making a connection between the implementation

and the outcome of a network function;
- Our study shows how the density of local connections in the functional topology affects the scalability of a clustering implementation;
- Our results highlight the structural patterns in the functional topology that lead to higher energy efficiency of the WSN;
- Our functional complexity allows us to analyze the trade-off between energy efficiency and scalability of a clustering algorithm implementation in WSNs.

3.2 Functional Topology Modeling Framework

The approach to planning, configuration, management and optimization of network functions is changing and moving towards self-organization. The traditional approach to most network functions involves the use of central control or optimization. However, the increasing heterogeneity of wireless technologies contributes to the rapid evolution, change, and growth of networks that makes the centralised approach unsustainable. If controlled in a self-organizing way, network functions such as handover, transmit power control, user allocation, data rate control and frequency allocation provide more flexibility and robustness in response to changes that may appear in the network [101].

In order to specify and analyze the complexity of a network function, we introduce the *functional framework*. Our framework represents an abstraction of a telecommunication network, modeling its operation by capturing all elements, i.e. nodes and connections, necessary to perform a given function. Our framework includes *functional topologies* which are graphs created based on the functional connectivity between system entities. In contrast to logical topologies which refer to data flows between network nodes, functional topologies, depict the connectivity pattern of a network function, which implies a broader meaning to nodes and links.

A node in our topology represents a functional entity or any information source that is part of the given network function. The links indicate dependencies between nodes. The topology as a whole depicts the specific implementation of the network function. The topology allows us to apply our model to analyze the complexity of the implementation. The functional complexity quantifies the organizational structure by analyzing the variety of relationships between system entities and roles that these entities have in the topology.

As an instructive example, we focus on self-organization from a frequency allocation perspective. More precisely, we use the frequency allocation algorithm from [28] to describe the process of mapping a network function into a topology and in the end to calculate the functional complexity.

The self-organizing frequency allocation algorithm considers a cellular network as a two-dimensional cellular automaton. Each cell in the model represents a self-organizing wireless system. The algorithm works based on the local information that nodes gather from their

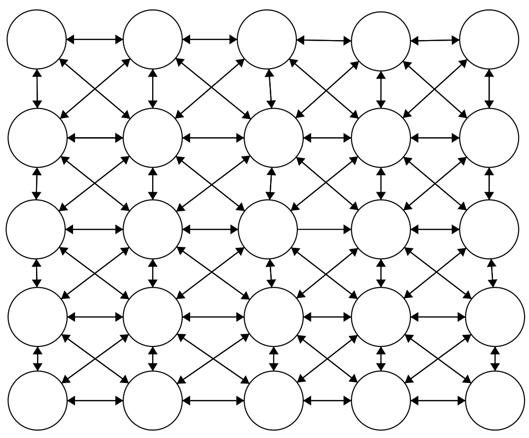


Figure 3.2: The physical topology according to the Moore neighborhood from cellular automata.

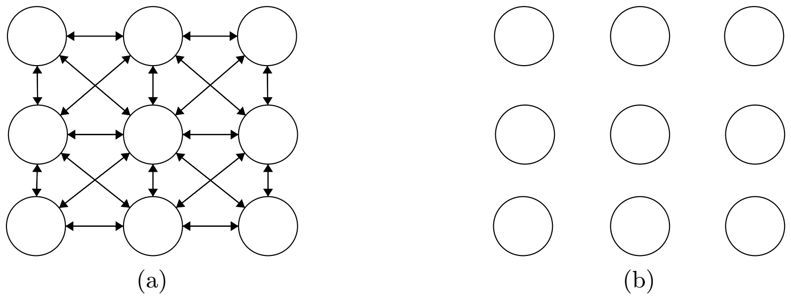


Figure 3.3: (a) The functional topology that represents the distributed self-organizing frequency allocation algorithm; the Moore neighborhood motif. (b) The functional topology that represents the random frequency allocation approach. Independent network functions have no interconnections.

neighbors. Briefly, every cell senses the given frequency channels, and allocates a channel with no interference. For more details about the algorithm the reader is referred to [28].

We focus on two frequency allocation algorithms (self-organizing algorithm and random frequency assignment) from [28]. Here we analyze the implementation rather than the outcome of the function. The work in [28] assumes a physical topology based on the Moore neighborhood, as shown in Figure 3.2. Even though the physical topology is the same for both implementations, the functional topologies are not.

We start with the random frequency allocation algorithm. It assumes that every cell assigns the frequency completely randomly from the set of available frequency channels. In order to create the functional topology let us imagine a virtual decision maker entity that is moving from one cell to another. At every cell the decision maker entity has no information about the allocated frequencies of other cells, which means that there are no functional connections between any two nodes. The result is a functional topology represented with a non-connected graph (Figure 3.3b).

To examine the functional topology of the self-organizing frequency allocation algorithm we use the same approach presented in the previous example. We imagine a virtual decision maker entity which is responsible for the frequency allocation of every node. In order to determine the functional connections of a node we analyze the interactions with other nodes in the process of decision making at our target node. As the self-organizing algorithm assumes only the knowledge about allocated frequencies at the neighboring cells, the functional connections exist only between physical neighbors. This results in a functional topology that is equivalent to the Moore neighborhood motif of the physical topology (Figure 3.3a).

In general network functions are not as simple as the above example. Therefore, to analyze the organizational structure of a network function we apply a multi-scale approach. The functional topology itself is built upon the local relationships between network cells according to the specified frequency allocation algorithm. The local interactions represent the lowest scale size. Analysis of higher scales is enabled by a multi-hop interaction examination. The reachability among nodes represents the interactions between them. By restricting the reachability to a certain number of hops along the topology we may examine the scale size of interest.

In [5] the authors discussed the relation between the scale size and the complexity, and they emphasize the importance of the scale size noting that the same system can show high complexity on one scale and low complexity on the other. The functional connectivity of each node represents the lowest scale in the functional framework and describes the interactions in the process of making a decision on this particular node. From the functional point of view the lowest scale size models a decision that is made by one node (in our case the frequency allocation of one cell). In order to apply a multi-scale approach we include higher scales in our analysis as follows. As every node represents a functional part, a subtask or an informational source, every group of nodes represents a group of subtasks that are part of the same function. The 2-hop scale size considers the node, its neighboring cells and neighbors of its neighbors. On this scale size the topology includes the interactions of nodes that are inside the radius of two hops. In our examples this scale size represents the frequency allocation of a two hop neighborhood and the interactions between nodes in the process of making a decision among a group of nodes. Note that the reachability of nodes increases with the scale size. In this case every node can reach any other node that is two or less than two hops away. The highest scale size implies a full mesh topology, where every node can reach any other node in the topology. This approach enables a multi-scale functional analysis which is presented in the results section.

3.3 Functional Complexity Metric

The traditional reductionist approach attempts to explain an entire system in terms of its individual components. In contrast, complex systems analysis is based on the relations between system parts which result in a greater outcome than would be expected from a simple sum of the outcomes of the individual parts. Therefore, complex systems move the focus from the node to the network.

As we represent network functions with topologies, the complex systems approach allows us to analyze the relations between functional parts. Such analysis captures the joint effort of functionally interconnected system parts and provides a measurement of the deviation of the complex behavior compared to a linear (non-complex) system. In order to capture this deviation, we analyze the joint effort of system sub-parts which are represented with subgraphs of the functional topology. Different subgraphs with the same size capture the variety of organizational structures in the topology, whereas different subgraph sizes allow us to understand the information gain from increasing number of nodes and interactions.

In order to capture the non-linear joint effort of functional parts we analyze the interactions of nodes involved in the decision making process on different scales. The scale size r determines the maximum number of allowed hops between two nodes in order to reach each other. In modeling a network function, our focus from the complexity point of view is the degree of interconnectivity which is represented by the reachability of functional parts. R is the maximum scale size of the system, which is defined as the longest shortest path in the entire topology (i.e. the graph diameter of the functional topology). If $r = R$ the reachability of every node in the topology is equal to 1.

In order to capture the relationships that underpin the operation of a given network function, we consider the interactions enabled by the structure of the functional topology on various scales of operation. Specifically, we consider that a node interacts if another node can reach this node on a particular scale. As such, we employ the Bernoulli random variable x_n to describe the potential for a node to interact on a particular scale, under the assumption that interactions are uniformly initiated by all nodes within the topology. Therefore, the probability distribution of x_n is determined by the reachability of node n for a given scale size r. equation (3.1) provides the relative reachability of node n, where i_r^n is the number of nodes that can reach node n for a given scale r and j represents the number of nodes for the given subgraph. Table 3.1 summarizes the notation used in our equations.

$$p_r(x_n = 1) = \frac{i_r^n}{j} \tag{3.1}$$

Shannon entropy for each node of the given subgraph with size j, and scale size r is calculated with equation (3.2).

$$H_r(x_n) = \sum_{x_n \in X} p_r(x_n) \cdot \log_2 \frac{1}{p_r(x_n)} \tag{3.2}$$

Since the probabilities in equation (3.1) and (3.2) indicate the relative reachability of a node, the entropy in our functional model represents the uncertainty of interaction of node n during the operation of a network function for a given scale size r. The uncertainty of interaction of a node depends on the role of this node in the graph that represents the function (e.g. hub, stub, disconnected node). A node with zero entropy functionally represents

Table 3.1: The notation used in the equations.

Symbol	**Meaning**
n	node
N	total number of nodes in the functional topology
j	subgraph size - number of nodes in the subgraph
r	scale size
R	maximum scale size, which is defined as the longest shortest path in the whole functional topology
Λ_k^j	one of the k subgraphs with j nodes that is induced from the functional topology graph
i_r^n	number of nodes that can reach node n for a given subgraph
x_n	Bernoulli random variable. $x_n = 1$ indicates that an interaction in the course of function operation involves node n, whereas $x_n = 0$ indicates that the interaction does not involve node n, for a given scale
X	since x_n is a Bernoulli random variable $X = \{0, 1\}$
$p_r(x_n = 1), p_r(x_n = 0)$	probabilities that any given interaction in the course of function operation involves or does not involve node n for a given scale r
$H_r(x_n)$	entropy of node n which indicates the uncertainty of involvement of node n in the operation of a network function for a given scale size r

a hub or a disconnected node. A hub in the functional topology is a node connected to all other nodes, which means that the reachability of this node is $p_r(x_n = 1) = 1$. Conversely, a disconnected node has the non-reachability $p_r(x_n = 0) = 1$. A hub is a functionality or informational source that interacts with each subtask in the functional topology. In contrast, a disconnected node in the functional framework represents a functionality not related to the modeled function, which means that this node does not provide information of interest for any part of our model.

The total amount of information of the k^{th} subgraph with j nodes for scale r is calculated with equation (3.3). Λ_k^j is one of the k subgraphs with j nodes. The total amount of information represents the total uncertainty which is related to the actual roles of nodes that appear within a subgraph and different subgraph patterns.

$$I_r(\Lambda_k^j) = \sum_{n \in \Lambda_k^j} H_r(x_n) \tag{3.3}$$

The average amount of information for a given subgraph size j is calculated with equation (3.4). β_j in the equation represents the number of connected subgraphs with size j.

$$\langle I_r(\Lambda^j) \rangle = \frac{1}{\beta_j} \sum_{k=1}^{\beta_j} I_r(\Lambda_k^j) \tag{3.4}$$

The functional complexity is calculated with equation (3.5).

$$C_F = \frac{1}{R-1} \sum_{r=1}^{R-1} \sum_{j=1+r}^{N} |\langle I_r(\Lambda^j) \rangle - \frac{r+1-j}{r+1-N} I_r(\Lambda^N)| \tag{3.5}$$

Note that Λ^N is the set of all nodes in the functional topology. For $j < 1+r$ the amount of information is equal to zero, because every subgraph with size $j < 1+r$ for the scale size r represents a full mesh topology in terms of reachability, therefore $H_r(x_n) = 0$ and $I_r(\Lambda_k^j) = 0$.

As shown in equation (3.5), in order to determine the complexity of an implementation of a network function, we compare this particular implementation ($\langle I_r(\Lambda^j) \rangle$) to a non-complex model of itself $\left(\frac{r+1-j}{r+1-N} I_r(\Lambda^N)\right)$. A non-complex model can be described by the sum of the functionalities of its parts. Therefore, the non-complex model assumes that every functional part always contributes the same amount of information, meaning that increasing the number of analyzed nodes leads to a linear increase in the total amount of information and in equation (3.5) it is represented by $\frac{r+1-j}{r+1-N} I_r(\Lambda^N)$. Conversely, a complex implementation implies that the network function relies on communication between its parts which results in a higher utility outcome, i.e. greater than the sum of its parts. Our functional complexity metric captures this difference between the amount of information for the complex and non-complex implementation, which is modeled with the inner sum in equation (3.5). Since we express the potential of a node to interact with other nodes as its reachability in the functional topology, Shannon entropy calculated based on this reachability represents the uncertainty of interactions for a subgraph of the functional topology. In equation (3.5), we capture the average uncertainty of interactions for all subgraph sizes ($\langle I_r(\Lambda^j) \rangle$), and compare it to the uncertainty of interactions which is expected from the non-complex model, which is the functional complexity and in equation (3.5) it is represented by the inner sum. This means that the functional complexity captures how much more information exists in the interactions between nodes than it would be expected from a naive linear representation of these interactions. In engineering, non-linear problems are often approximated as linear problems for small changes in the input of the system. This approximation often results in models that have limited applicability, and the functional complexity quantifies how much information is lost by this linear approximation. When referring to non-linearity in this thesis, we mainly focus on the spread of information throughout the system. In other words, we study how much the probability of a node's accessibility to information (i.e. its reachability) differs from what is expected by studying the whole system. The underlying communication structure affects the degrees of freedom of each individual node and restricts or amplifies its access to information from other nodes in the network. This can cause problems in case of spreading the effect of faulty node behavior or the spread of a malicious piece of code. However, it can also result in a positive effect on the network operation. For example, as shown in [28], the amplified access to information from other nodes in the network can lead to faster convergence and higher robustness of distributed self-organizing algorithms.

Shannon entropy reaches its maximum if the reachability of the nodes is $p_r(x_n = 1) = 1/2$, which means that a sparse graph with almost uniform distribution of links among nodes results in high values of $H_r(x_n)$. This again leads to high values of $I_r(\Lambda_k^j)$, which result in high functional complexity. The functional complexity is zero for a fully connected and for a disconnected graph. The reachability of each node in a fully connected and disconnected graph are equal to 1 and 0 respectively, which results in zero Shannon entropy and zero complexity.

The variety of roles emphasises the importance of interconnectivity of functional parts. Since nodes represent functional parts, different roles refer to different interactions that underpin the operation of a network function. If one node is highly connected to a group of nodes, this means that it influences the entire group. At the same time the group has a high influence on the operation of this node. In our frequency allocation example, the disconnected topology resulting from the random frequency allocation function shows that every node is independent, which results in one simple role (disconnected node) for every subgraph. Describing the relationship between a single element and the rest of the functional topology provides all necessary information needed to understand the relationships of the entire topology. Therefore this structure is not considered as complex. Conversely, the self-organizing algorithm provides different roles for nodes. This results in a complex structure, where the outcome depends on the communication between the structural parts, which means that the communication between system entities becomes more important to understand than the entities themselves.

3.4 Analysis of the Functional Complexity Properties

In this section we apply our complexity model to several functional topologies to examine the main properties of the functional complexity metric. In addition to the two implementations

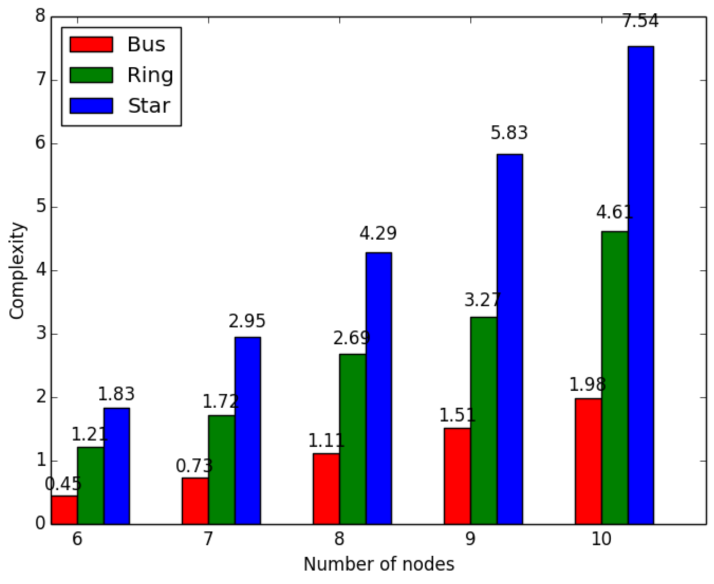


Figure 3.4: Complexity of a bus, ring and star topology.

of the frequency allocation function, we also investigate common graph structures (e.g. ring, star, bus, full mesh) that describe complex and non-complex functional relations. In the course of this analysis, we present the relationship between the complexity of a function and graph theory metrics that describe the functional topology representation of the network function.

A complex system, and hence a complex function, is determined by many parameters. We compare our metric to the following graph theoretic notions to highlight the utility of our methodology:

- number of nodes,
- average path length,
- clustering coefficient.

Intuitively, if the number of nodes increases the system becomes more complex. The average path length indicates the distance between elements, which implicitly refers to the relationship strengths in the network. The clustering coefficient provides information about the overall strengths of functional dependencies between nodes grouped around a central entity. All these metrics provide useful information about certain organizational characteristics of the system. We start the analysis with common graph structures like a bus, ring, star and full mesh topology to investigate the impact of different graph structures on the amount of complexity. Figure 3.4 shows the complexity values of theses topologies in the range of six to ten nodes. Additionally, in Section 3.4.2, we present the correlation between the complexity metric and different graph theory metrics.

3.4.1 Bus, Ring, Star and Full Mesh Topologies

A full mesh topology results in zero complexity for any number of nodes. Therefore, even though a full mesh topology provides a densely connected structure, the overall relationships between functional parts represented by a full mesh topology are non-complex. The structures of any subgraph in a full mesh topology are also full mesh connectivity patterns. This means that a full mesh topology represents a function in which every functional part interacts with all other entities in order to make a decision. The functional structures and roles of nodes in a graph that represents such functions are always the same (each individual node is a hub). This means that in order to describe the system (the function), it is enough to describe an individual element and such functions are considered as non-complex.

All the nodes in a bus topology are fairly spread out (i.e. the average path length between nodes is high). In other words, the bus topology represents a connectivity pattern with weak overall connections among parts of the topology. The bus topology does not provide a great variety of organizational structures, which results in low complexity. According to Figure 3.4, the functional complexity of the bus topology increases with the increase in the number of nodes.

Figure 3.4 shows that the functional complexity of a ring topology is higher than the complexity of a bus topology for the same number of nodes. The ring topology conforms to the same trend in which complexity increases with the number of nodes. The ring topology compared to the bus is less spread out (i.e. the average path length is shorter) and therefore provides tighter connections between the nodes. As the nodes are closer to each other, due to the additional path, this functional topology represents tighter functional relationships compared to the bus topology. Tighter functional relationships result in higher complexity.

The star topology is a functional relationship in which one node interacts with all other nodes in the decision making process. It also conforms to the same trend in which complexity increases with the number of nodes in the topology. The star structure has tight connections between its parts. This means that the star topology depicts stronger functional relationships compared to the bus and ring topology. Therefore, the star topology has the highest complexity value compared to the other structures presented in Figure 3.4.

The analysis of simple graph structures (bus, ring, star and full mesh) gives a general overview of the impact of different graph organizations on the amount of complexity. The functional complexity metric captures the diversity of structures and roles of nodes, which provides a different perspective to analyze the complex relationships between system entities. According to Figure 3.4, the star topology is the most complex structure compared to the bus, ring and full mesh topologies. Considering the sparse connectivity and short average path length between entities represented with a star topology, we believe that this type of relationship contributes to the increase in complexity. Based on the analysis of different sizes of ring and bus topologies, we believe that spreading (distributing) the functionality all over the topology also affects the increasing complexity values. We expect that a combina-

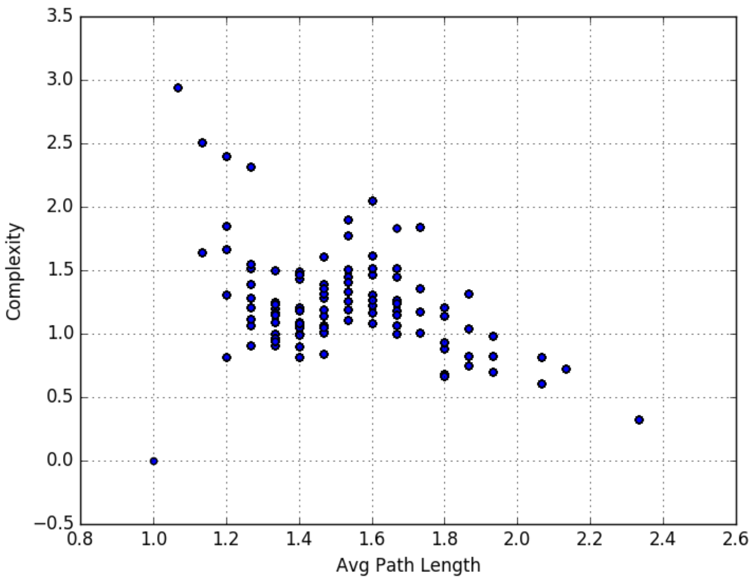


Figure 3.5: The relationship between the average path length and functional complexity for every possible functional topology with six labeled nodes (we consider connected graphs only). We consider all unique combinations of edges starting with the minimum of five up to the maximum of fifteen. The Pearson product-moment correlation coefficient for these two variables (average path length and functional complexity) is -0.43.

tion of these two types of relationships results in highly complex organizational structures. Such a structure consists of local centres (star subgraphs/clusters) that are highly interconnected. As functional complexity captures the variety of structural patterns, we expect that complexity is high if the number of local centres is high.

3.4.2 Comparison with Graph Theory Metrics

In this subsection we analyze the relationship between complexity and different graph theory metrics. We also want to emphasize that the graph theory metrics do not include the quantity that is measured by equation (3.5). In order to do so, we investigate the correlation between these metrics, and show that the graph theory metrics focus on individual aspects of the topology (e.g. clustering, connectivity, degree distribution), whereas functional complexity quantifies the overall uncertainty of interactions between functional entities which emerge from the local interactions between them. We also study the correlation between functional complexity and different centrality metrics, for a set of *Small-World* and *Erdos-Reny* graphs.

To describe the relationship between the average path length and the functional complexity of an implementation of a network function we analyze all possible graph organizations (distributions of links) between six nodes - see Figure 3.5. The Pearson product-moment correlation coefficient for these two metrics is -0.43 (see Table 3.2). Notice in Figure 3.5 that the highest functional complexity for six nodes is 2.9. The maximum complexity achievable

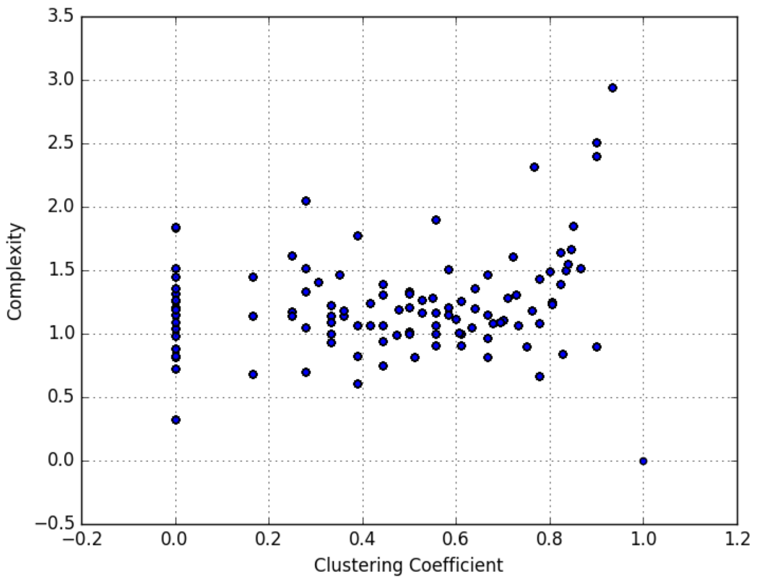


Figure 3.6: The relationship between the clustering coefficient and functional complexity for every possible functional topology with six labeled nodes (we consider connected graphs only). We consider all unique combinations of edges starting with the minimum of five up to the maximum of fifteen. The Pearson product-moment correlation coefficient for these two variables (clustering coefficient and functional complexity) is 0.15.

with 6 nodes is at least 1.5 times higher than the complexity of the bus, ring and star topology with the same number of nodes An important fact is that for the same average path length we have a range of different complexity values. Therefore, despite the high correlation between the average path length and complexity, we cannot predict the complexity of a structure based on the average path length. Additionally, the maximum value of complexity occurs for an extremely low average path length, which is still higher than one. Figure 3.5 also shows that the envelope of the complexity values is a non-monotonic function. Notice also that complexity is equal to zero for the average path length equal to one. This value of the average path length represents a full mesh topology.

As shown in Table 3.2, the correlation between the clustering coefficient and complexity is low (0.15) for six nodes. Figure 3.6 depicts the relationship between the average clustering coefficient and complexity for six nodes. The average clustering coefficient is a graph theory metric that measures the average number of neighboring nodes that are neighbors to each other [102]. Therefore, the average clustering coefficient quantifies the strength of functional connections between nodes grouped around local centres. Again, the envelope of the complexity values in this relationship is also non-monotonic. Notice in Figure 3.6 that complexity has its maximum for an average clustering coefficient close to one. This is expected because this value of the clustering coefficient represents a network with tight functional connections between nodes grouped around local centres. Notice also that complexity is zero for the average clustering coefficient equal to one. According to the definition

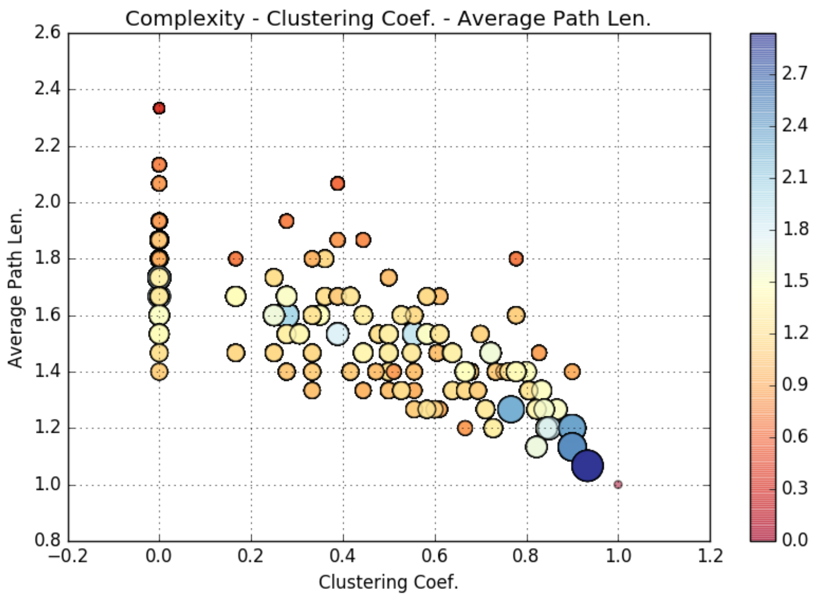


Figure 3.7: The relationship between the clustering coefficient, average path length and the functional complexity for every possible functional topology with six labeled nodes (we consider connected graphs only). We consider all unique combinations of edges starting with the minimum of five up to the maximum of fifteen. The complexity value is represented by the color and diameter of circles.

of the clustering coefficient, the value one indicates that all neighbors of every node in the functional topology are neighbors to each other, which represents a full mesh topology.

In order to analyze the joint relationship between complexity and both of the graph theory metrics presented above (the average path length and the average clustering coefficient) we plot all these metrics together on one graph (see Figure 3.7). The x and y axes represent the average clustering coefficient and the average path length respectively and the complexity is represented by the radius of the circles. Additionally, the circles are colored differently to emphasize the different complexity values. The multivariate correlation between these three variables is 0.47 (see Table 3.2). Overlapping circles with different radii indicate that for the same average path length and average clustering coefficient there exist multiple topologies with different functional complexity values. The functional complexity is an information theory metric, while the average path length and the average clustering coefficient are graph theory metrics, meaning that they quantify different properties, and therefore this was expected. Graph theory metrics allow us to analyze individual aspects of the functional topology, whereas complexity adds additional information about the diversity of relationships between functional parts and provides a different approach to analyze complex network functions.

Figure 3.8 shows that the correlation between complexity and different graph theory metrics decreases as the number of nodes in the graph increases. This is expected, because the number of graph combinations increases exponentially with the number of nodes. The

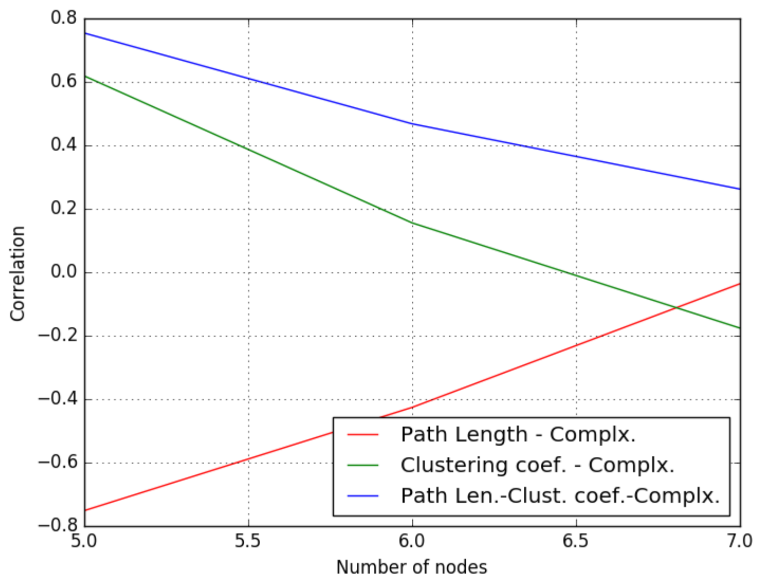


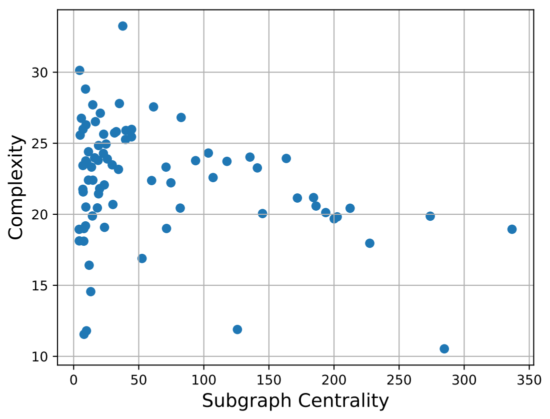
Figure 3.8: The relationship between the correlation coefficients of the graph theory metrics (average path length, clustering coefficient) and the complexity metric for different sizes of functional topologies. These values were calculated for all unique combinations of n nodes and from $n-1$ to $(n*(n-1))/2$ edges.

Table 3.2: Correlation between graph theory metrics and the functional complexity metric for all possible distributions of links between six nodes.

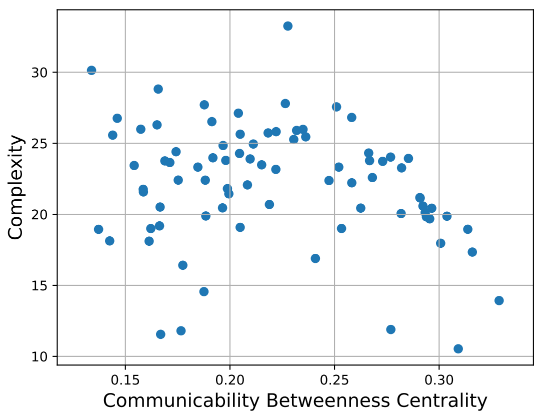
Correlation variables	Correlation
Average Path length - Complexity	-0.43
Clustering coefficient - Complexity	0.15
Avg. Path length - Clustering coef. - Complexity	0.47

rapid increase of graph combinations allows us to create a variety of complex structures for the same value of certain graph theory metrics. Hence, the complexity range increases for an individual value of a graph theory metric, meaning that if we would represent these structures with the same plot as shown in Figure 3.7, we would get even more overlapping circles with different complexity values for the same average path length and average clustering coefficient. The decrease in the correlation between complexity and different graph theory metrics suggests that graph theory metrics do not target the same properties of the system. In other words, the complexity metric adds new information about the relationships between system entities.

We also study the relationship between functional complexity and various centrality metrics (e.g. degree centrality, eigenvector centrality, closeness centrality, betweenness centrality) for Erdos-Reny and Small-World graphs. The results presented in the reminder of the section refer to 100 graphs of 20 nodes. We start with the Erdos-Reny graph. As



(a)



(b)

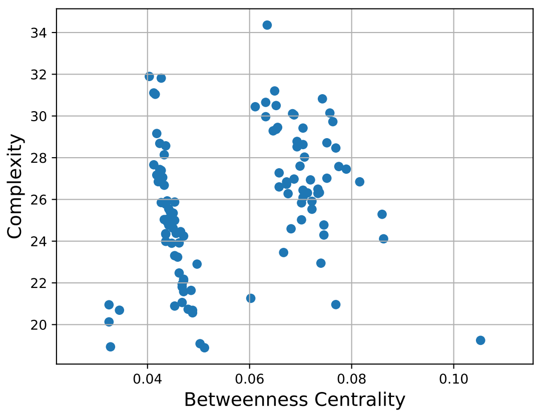
Figure 3.9: Relationship between two centrality metrics, i.e. Subgraph Centrality and Communicability Betweenness Centrality, and Functional Complexity in the case of Erdos-Reny graphs.

Table 3.3: Correlation between functional complexity and various centrality metrics for Erdos-Reny and Small-World graphs.

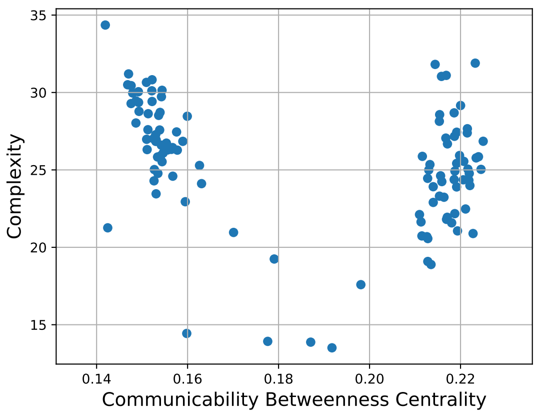
Correlation variables	Erdos-Reny	Small-World
Degree Centrality - Complexity	-0.15	-0.04
Eigenvector Centrality - Complexity	0.12	-0.18
Closeness Centrality - Complexity	-0.03	0.18
Current Flow Closeness Centrality - Complexity	-0.08	0.02
Betweenness Centrality - Complexity	-0.03	-0.37
Current Flow Betweenness Centrality - Complexity	0.09	-0.17
Communicability Betweenness Centrality - Complexity	-0.22	-0.38
Load Centrality - Complexity	-0.03	-0.37
Subgraph Centrality - Complexity	-0.38	-0.27

shown in Table 3.3, in the case of Erdos-Reny graphs, the correlation between functional complexity and any of the studied centrality metrics is very low. From all the presented correlations in Table 3.3, Subgraph Centrality and Communicability Betweenness Centrality, even though low, exhibit the highest correlation with functional complexity. Figure 3.9a shows the relationship between the Subgraph Centrality and functional complexity and Figure 3.9b the relationship between Communicability Betweenness Centrality and functional complexity. Similarly to Figure 3.6, Figure 3.9 shows that for the same value of the Subgraph or Communicability Betweenness Centrality there is a range of different functional complexity values.

Table 3.3 also shows low correlation between functional complexity and any of the studied centrality metrics in the case of Small-World graphs. In this case, Betweenness Centrality

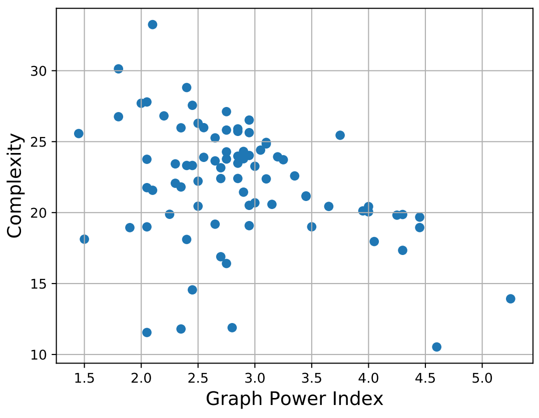


(a)

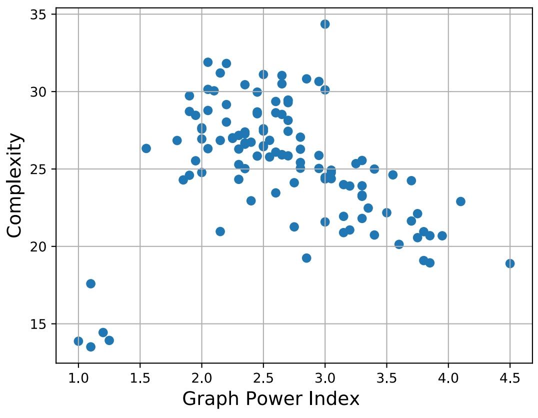


(b)

Figure 3.10: Relationship between two centrality metrics, i.e. Betweenness Centrality and Communicability Betweenness Centrality, and Functional Complexity in the case of Small-World graphs.



(a)



(b)

Figure 3.11: Relationship between the Graph Power Index and Functional Complexity for both, Erdos-Reny (on the left) and Small-World (on the right) graphs.

and Communicability Betweenness Centrality exhibit the highest correlation with functional complexity. The relationship between these two centrality metrics and functional complexity is shown in Figure 3.10. Similarly to Figure 3.9, Figure 3.10 shows that for the same value of the centrality metrics there is a range of different functional complexity values.

Besides the centrality metrics, we have also studied the correlation between the functional complexity and the Fiedler value [103] and the Graph Power Index [104] for Erdos-Reny and Small-World graphs. Our analysis shows that the correlation between the functional complexity and the Fiedler value is very low for both types of graphs, i.e. 0.05 for Erdos-Reny and −0.04 for Small-World graphs. The correlation between the functional complexity and the Graph Power Index is −0.36 and −0.15 for Erdos-Reny and Small-

World graphs respectively. As shown in Figure 3.11 there is a range of different functional complexity values for the same value of the Graph Power Index.

3.4.3 Complexity of the Frequency Allocation Algorithms

Now we compute the functional complexity of the two frequency allocation algorithms presented in the framework section. Our complexity metric, when applied over the functional topology, gives insight into the functional relationships between system entities, which allows us to focus on the joint effect of functional parts in order to execute the modeled network function.

The random frequency allocation algorithm is represented by a disconnected graph. Each element represents a closed system which results in the absence of connections between them. From the system perspective the functional parts of this implementation represent a set of independent elements. As the elements do not interact with each other, in order to describe the set we can simply describe its elements. More precisely, to describe the set we have to describe only one element, because each element of the set is the same. Each element of the set simply assigns one of the available frequencies. Such a set that is represented with independent elements has zero functional complexity.

The self-organizing distributed frequency allocation algorithm is represented by the functional topology shown in Figure 3.3a. In order to analyze the complexity of this implementation of the frequency allocation function we apply our complexity metric over the functional topology representation. Figure 3.12a depicts the deviation in the amount of information with increasing subgraph sizes from a linear increase (i.e. the functional complexity). In equation (3.5) the linear increase is represented by $\left(\frac{r+1-j}{r+1-N}I_r(\Lambda^N)\right)$. As shown in equation (3.5), it is calculated as the uncertainty which is expected from the calculation performed on the whole system $(I_r(\Lambda^N))$. The linear increase represents an equivalent simple system implementation, meaning that all the subgraphs of a specific size feature the same uncertainty of interaction. Every functional part contributes with the same amount of information, which results in a linear increase over increasing subgraph sizes.

The non-linear increase represents the heterogeneity of structures within the functional representation of a network function, which indicates that the relationship between functional parts is complex. As shown in equation (3.5), the non-linear increase is calculated as the average amount of information for a specific subgraph size $(\langle I_r(\Lambda^j)\rangle)$, and it represents the uncertainty of interaction among nodes for this specific subgraph size. According to equation (3.5), the complexity is calculated as the sum of differences between the average amount of information for a specific subgraph size and the amount of information which is expected from the calculation performed on the whole system (the non-linear and linear functions in Figure 3.12a).

Considering that for the functional topology of the self-organizing algorithm we have that $R = 2$, the multi-scale approach applied on this functional topology implies only a

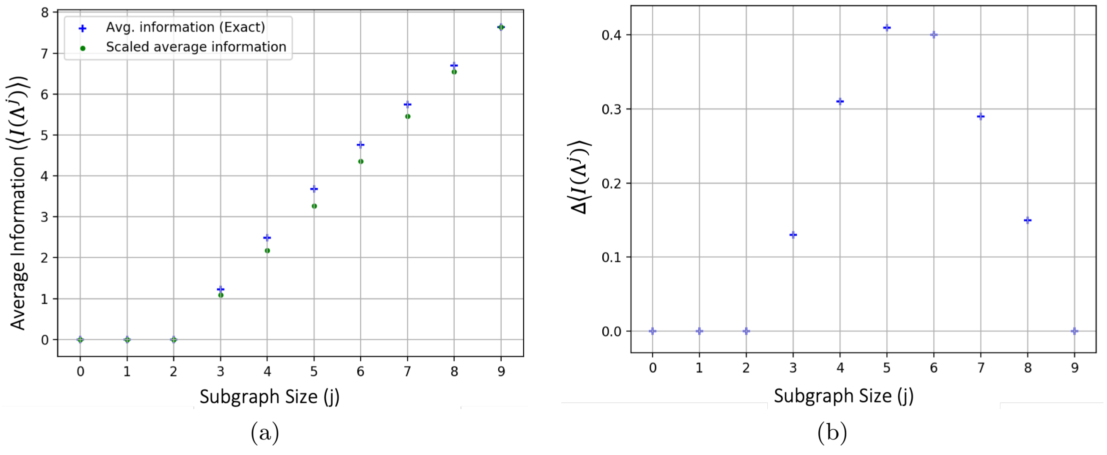


Figure 3.12: The functional topology has nine nodes; the maximum scale size R is 2; the functional complexity is represented by the difference between the linear and non-linear function. (a) The relationship between the uncertainty of interactions for all subset sizes ($\langle I_r(\Lambda^j)\rangle$) and the uncertainty which is expected from the calculation performed on the whole system $\left(\frac{r+1-j}{r+1-N} I_r(\Lambda^N)\right)$. (b) The difference between the uncertainty of interactions for all subset sizes and the uncertainty which is expected from the calculation performed on the whole system $\left(\Delta I(\Lambda^j) = |\langle I_r(\Lambda^j)\rangle - \frac{r+1-j}{r+1-N} I_r(\Lambda^N)|\right)$, i.e. the functional complexity of the self-organizing frequency assignment approach.

single scale analysis, that is $r = 1$. The subgraph sizes of interest are in the range of two to nine. Considering that $r = 1$, subgraphs with the size two or smaller represent full mesh topologies. The maximum value of the subgraph size is nine because the functional topology consists of nine nodes. The complexity of the self-organizing frequency allocation algorithm is 1.69 and it is represented by the difference between the linear and non-linear functions in Figure 3.12a. Figure 3.12b shows the distance between the uncertainty of interactions for all subset sizes and the uncertainty which is expected from the calculation performed on the whole system, i.e. the functional complexity of the self-organizing frequency allocation approach. More precisely, Figure 3.12b depicts the single scale complexity function which is represented by the inner sum of equation (3.5). According to equation (3.5), the functional complexity is calculated as the sum of the values of the function depicted in Figure 3.12b.

Similarly, the authors of [28] analyze the two frequency allocation algorithms. The complexity metric used in [28] is excess entropy. Even though we are not aware of any direct relation between our functional complexity and excess entropy, the work proposed by the authors of [28] is the closest to ours in terms of applying complex systems tools to analyze network functions. In contrast to our work, they analyze the complexity of the outcome (i.e. the frequency allocation) of these implementations. The results that are presented in their work show that the random approach produces an outcome with zero complexity, whereas the self-organizing distributed frequency allocation algorithm produces a complex outcome. Here, our functional complexity metric analyzes rather the complexity of the implementation itself than the output of the network function. As discussed above, the functional complexity of the random frequency allocation algorithm is zero, and the functional complexity of the self-organizing distributed frequency allocation algorithm is 1.69. This suggests that a complex implementation of a function results in a complex outcome. In [28] the authors show that the response to a change of a system with highly complex output happens in a more manageable fashion causing less disruption. The authors of [28] also emphasize the higher robustness of a complex outcome compared to a non-complex outcome. Making a connection between the implementation and the outcome of a function correlates the characteristics of the outcome with the complex relationships that underpin the functional structure.

3.5 Relationship between Functional Complexity, Scalability and Energy Efficiency in WSNs

In this section we analyze network functions by applying our functional complexity framework to clustering algorithms in WSNs. The goal is to demonstrate the broad applicability of the framework to understand the organizational structure of functions within a range of different networks (e.g. cellular networks, WSNs, Vehicular ad-hoc networks (VANETs)). We employ our functional framework to model different implementations for clustering in WSNs. Our goal is to highlight the structural patterns present in the functional topology that result in certain properties (scalability and energy efficiency) of the implementation. This allows us to understand the underlying mechanisms that are a product of complex interactions between functional entities.

The topology of WSNs is subject to constant changes due to the disposable nature of wireless sensors and the constant need to expand and densify the sensing area. Therefore, besides the energy efficiency, clustering algorithms for WSNs have to exhibit scalability. Scalability is the capability of the network to adapt to new nodes joining the network, existing nodes leaving the network, and other nodes migrating from one cluster to another [105].

Clustering is a technique that allows us to divide a network of nodes into groups called clusters. It is a network reorganization technique that is commonly employed to adjust to the constant changes in the physical topology. In addition to the energy efficiency, which explicitly affects the network lifetime, clustering algorithms have a great influence on scalability, load balancing, fault-tolerance, delay reduction, etc. [106]. Since different implementations of clustering in WSNs differ in organizational and communication patterns, the understanding of these patterns allows us to comprehend the aspects of the implementations that lead to certain characteristics. We focus on the implications for scalability and energy efficiency. The manner in which parts of the network share information and the extent to which information spreads throughout the network are important aspects of a clustering algorithm, as they directly impact the system longevity and the scalability of the network. The complex systems science approach we adopt allows us to investigate the amount of information about the system subparts by examining the system as a whole and comparing this to the actual amount of information that exists within the system subparts. In other words, this allows us to quantify the amount of uncertainty of interaction that exists within smaller subparts of the system compared to the uncertainty of the whole system. We identify the structural patterns in the functional topology that lead to higher energy efficiency of the WSN. We also show how the density of the connections in the functional topology is related to the scalability of the network.

There is no shortage of clustering algorithms in literature [105, 107–118]. The authors of [106] and [119] do not focus on proposing new algorithms, instead they analyze the properties of clustering algorithms proposed by other authors, like scalability, fault-tolerance, connectivity, redundancy elimination, network lifetime, clustering criteria. We focus on the Low Energy Adaptive Clustering Hierarchy (LEACH) algorithm proposed in [117] and the Hierarchical Control Clustering (HCC) algorithm proposed in [105], due to the importance of these algorithms for WSNs (the LEACH algorithm is one of the most cited clustering algorithms, and the HCC algorithm is the most cited multi-tier hierarchical clustering algorithm [106]).

3.5.1 Clustering Algorithms in WSNs

Considering the limited energy resources and the size of WSNs, the best approach is to partition the network into interconnected subgroups (clusters), whose local behavior gives rise to the global objective of the network (e.g. minimizing the energy consumption). Generally, clustering includes two phases:

- Set-Up Phase - during this phase the cluster-heads are selected. Cluster-heads are the nodes responsible for the coordination of the clustering process.
- Maintenance Phase - this phase covers the reorganization of the clusters in order to add new nodes to the cluster, deal with nodes that leave the cluster, or simply change the roles of nodes (cluster-head or ordinary node) in the cluster in order to achieve higher energy efficiency [120].

As the maintenance phase implies the reorganization of clusters in order to achieve higher energy efficiency and to adjust to changes in the topology, we focus on this phase. As mentioned previously, we consider two clustering algorithms (the LEACH and the HCC algorithms) proposed in [117] and [105].

The LEACH algorithm is based on rounds. Each round includes both phases (set-up and maintenance phase). Once the network is deployed, during the first set-up phase, the nodes decide randomly which role to play (cluster-head or ordinary node). When a node decides to be a cluster-head it broadcasts this information, whereas a node that decides to be an ordinary node listens to the broadcast messages and joins the closest cluster-head. After all ordinary nodes join one of the cluster-heads, the initial set-up phase finishes and the

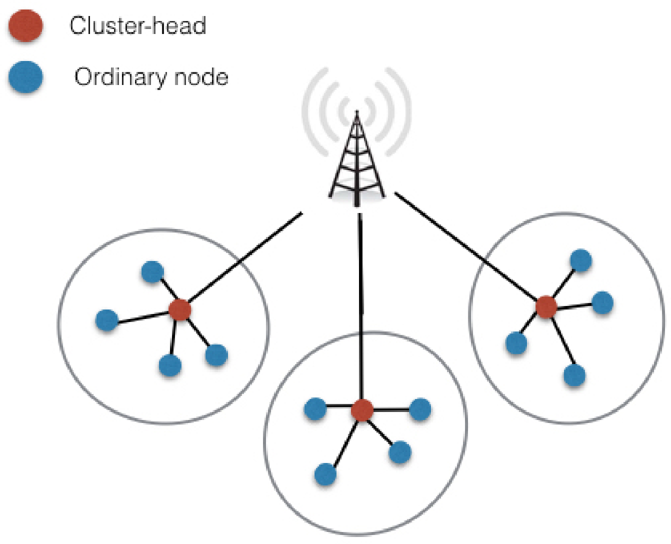


Figure 3.13: The topology created according to the LEACH algorithm. Ordinary nodes transfer sensing information to the cluster-heads, which forward this information to the BS.

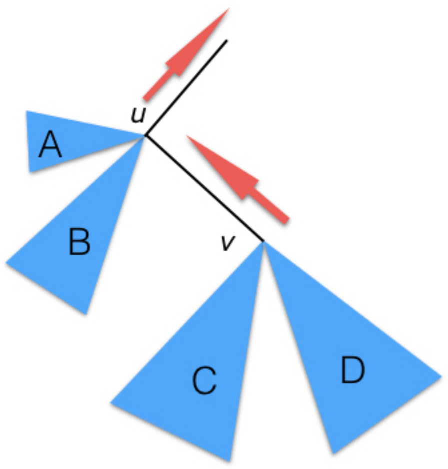


Figure 3.14: Cluster formation according to the HCC algorithm. Each node (v, u) discovers its subtree size and forwards this information upstream to its parent. If the subtree size is big enough, the subtree becomes a cluster (A, B, C, D).

maintenance phase starts. In the maintenance phase nodes that belong to the same cluster transfer sensing information to the cluster-head (Figure 3.13). Moreover, as the LEACH algorithm involves rotating the role of cluster-heads between nodes in a cluster, the nodes that belong to the same cluster communicate to each other in order to choose the next cluster-head candidate. The next round starts with the set-up phase, but now, instead of randomly choosing the cluster-heads, they are chosen based on the information exchanged during the maintenance phase (e.g. node battery level, nodes proximity to the gateway).

The HCC algorithm is a multi-tier hierarchical clustering algorithm which has proven to be highly scalable. The set-up phase involves the Breadth-First Search (BFS) tree discovery and cluster formation. The tree discovery is a distributed formation of a BFS tree rooted at the initiator node. The cluster formation is shown in Figure 3.14. The clustering formation process is distributed and it is executed on each node. Based on the tree discovery each node knows its parent node and its child nodes. Each node i discovers the size of its downstream subtree $|V_i|$ and reports this to its parent node. The information about the subtree sizes allows us to create the clusters based on the defined cluster size k (number of nodes per cluster - this number is predefined). Each node compares its subtree size $|V_i|$ to the defined cluster size k and if $k \leq |V_i| < 2k$ the node initiates the cluster formation on its subtree. As the BFS tree denotes the routes to the Base Station (BS), the sensing information of all sensors is transmitted using these routes. The maintenance phase involves keeping up-to-date information on each node about its parent and child nodes.

3.5.2 Application of the Functional Framework

In order to analyze the underlying connectivity patterns of a clustering algorithm, we employ the functional framework introduced in Sections 3.2 and 3.3. This allows us to model the implementation of a clustering algorithm with functional topologies and use the functional complexity (C_F) to capture the non-linear effort of functional parts by analyzing the interactions between nodes involved in the clustering process. In Section 3.2, we introduce an approach which allows us to analyze the functional complexity on different scales r. The scale size r is defined as the number of allowed hops between two nodes. As the analyzed clustering algorithms rely on the communication established between one hop neighbors, in our analysis we focus on the scale size $r = 1$.

The functional complexity defined in equation (3.5) compares the uncertainty of interactions for a smaller subset ($\langle I_r(\Lambda^j)\rangle$) to the amount of information which is expected from the calculation performed on the whole system $\big(I_r(\Lambda^N)\big)$. Due to the focus on the one hop communication ($r = 1$) for the given algorithms, in our case equation (3.5) can be simplified and the single-scale complexity is calculated with equation (3.6).

$$C_F = \sum_{j=2}^{N} |\langle I_1(\Lambda^j)\rangle - \frac{2-j}{2-N} I_1(\Lambda^N)| \tag{3.6}$$

$I_1(\Lambda^N)$ is the total uncertainty calculated by considering the whole functional topology for the single scale $r = 1$. $I_1(\Lambda^j_k)$ is the total amount of information of the k$^{\text{th}}$ subgraph with j nodes for $r = 1$, and it is calculated with equation (3.7).

$$I_1(\Lambda^j_k) = \sum_{n \in \Lambda^j_k} H(x_n) \tag{3.7}$$

To maximize the total amount of information of the k$^{\text{th}}$ subgraph $\big(I_1(\Lambda^j_k)\big)$, the number

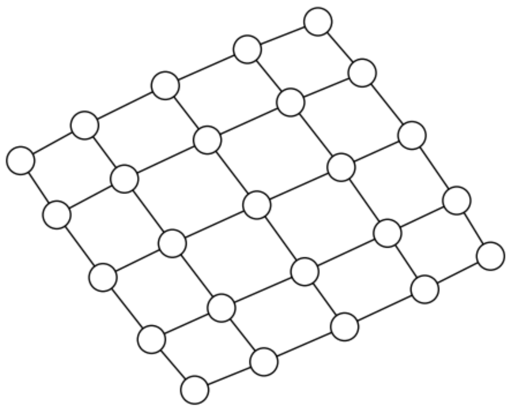


Figure 3.15: An example of a physical topology of a WSN according to the Von Neumann neighborhood.

of links in this subgraph should be small and the links should be uniformly distributed, because Shannon entropy, as defined in equation (3.2), reaches its maximum in this case. Such graphs (sparse graphs with uniformly distributed links for subgraphs with the size $j < N$) result in high functional complexity, whereas a fully connected and a disconnected graph have functional complexity equal to zero.

The graph shown in Figure 3.15 is an undirected graph created according to the Von Neumann neighborhood, and it depicts an example of a physical topology of a WSN deployed in a regular lattice. A wireless node A can transmit information to another wireless node B only if node B is within the transmission radius R_A of node A. In the case of WSNs, we consider the communication between two nodes established only if both nodes can transmit information to each other. In other words, we consider that two nodes can exchange information only if the distance between them is $d(A, B) \leq min\{R_A, R_B\}$.

In Section 3.2, we presented an approach to map different frequency allocation algorithms into functional topologies. We apply the same approach to examine the functional topologies of the LEACH and HCC algorithms.

We start with the LEACH algorithm. As discussed before, we focus on the maintenance phase of the algorithm. According to the approach in Section 3.2, we imagine a virtual decision maker entity that is moving from one node to another. At each ordinary node the decision maker entity communicates with the cluster-head and all other ordinary nodes that belongs to the same cluster. At each cluster-head the decision maker entity communicates with each ordinary node that belongs to the same cluster and with the BS. This results in a functional topology with dense local connections (Figure 3.16).

We now examine the functional topology of the HCC algorithm during the maintenance phase of the algorithm in which nodes maintain their positions in the BFS tree. The virtual decision maker entity would collect information from the nodes that belong to the subtree

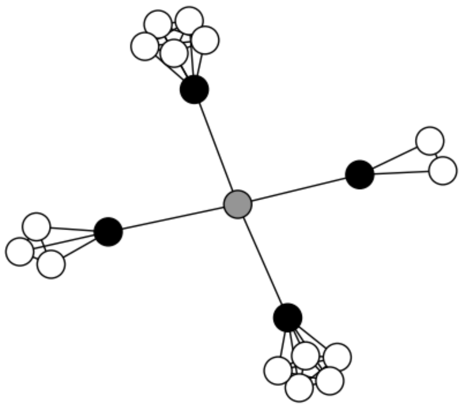


Figure 3.16: The functional topology of the LEACH algorithm for a network of twenty nodes which are divided into four clusters. The white nodes represent ordinary nodes, the black nodes represent cluster-heads, and the gray node represents the BS.

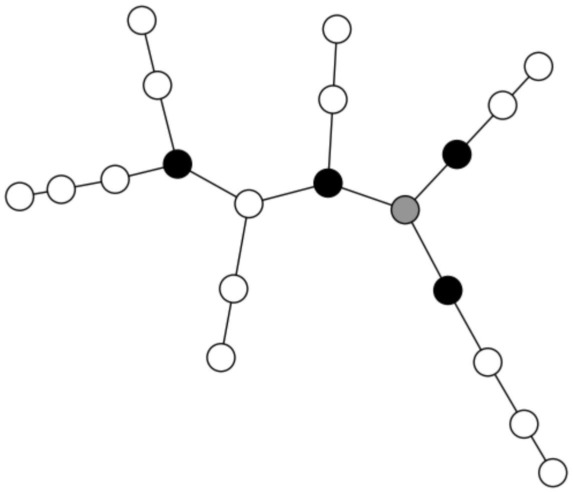


Figure 3.17: The functional topology of the HCC algorithm for a network of twenty nodes, which represents the BFS tree created based on the physical topology shown in Figure 3.15. The white nodes represent ordinary nodes, the black nodes represent cluster-heads, and the gray node represents the BS.

of the given node and forward this information to its parent in the BFS tree. Therefore, the functional topology of the HCC algorithm is equivalent to the BFS tree of the physical topology (Figure 3.17).

3.5.3 Results and Implications

In order to transfer the information collected at a sensor node to a BS, WSNs establish paths in an ad-hoc manner. Two nodes can exchange information if and only if they are within the transmission radius of each other. However, after the set-up phase of the clustering algorithm finishes, nodes do not communicate to all other nodes that are within their coverage area. At this stage, the information exchange depends on the rules of the clustering algorithm.

The maintenance phase of the LEACH algorithm invokes the set-up phase, where nodes that belong to the same cluster talk to each other to decide which node is the next candidate for the cluster-head role, and each node transfers sensing information to the current cluster-head. Adding a new node to any cluster means establishing a connection to all nodes within the cluster because each node has to be aware of all other nodes that belong to the same cluster, in order to maintain the process of cluster-head elections.

The maintenance phase of the HCC algorithm is simpler than that of the LEACH algorithm. Each node forwards the information of its children and transfers its own sensing information to its parent node. Therefore, adding a new node to the cluster simply means that the new node gets connected to one of the existing nodes which is going to be its parent node. The new node does not need to inform all nodes in the cluster about its arrival, which simplifies the process of adding nodes to the network.

The authors of [107–110] and [121] discuss different objectives of clustering algorithms. Among other things, the authors focus on scalability issues and network longevity (energy efficiency) of these algorithms. Hierarchical approaches proved themselves to be more scalable than their non-hierarchical counterpart. The authors of [109] showed that a trade-off exists between network scalability and energy consumption in clustering schemes. Our goal is to investigate the relationship between these objectives and the functional complexity, which would allow us to analyze them together. In [107–110] and [121], the authors emphasize that the LEACH algorithm is designed to extend the network longevity, whereas the HCC algorithm targets scalability as the main objective. In other words, the LEACH algorithm is less scalable and the HCC algorithm is less energy efficient.

The authors of [106–110] show that the cluster-head - BS communication highly affects the network longevity. Hence, we calculate the energy efficiency of a clustering implementation as the ratio between the average number of intra-cluster connections and the number of links between the BS and each cluster-head in the functional topology. If the ratio increases the energy efficiency increases, due to the larger number of intra-cluster connections as compared to the number of connections between the BS and the cluster-heads. As the scalability of the networks represents the adaptability to changes, we calculate it as the average number of messages sent when a new node joins the network. If the average number of messages increases, the scalability of the network decreases.

The relationship between the uncertainty of interactions calculated for all subset sizes and the amount of information which is expected from the uncertainty of the whole system for the LEACH algorithm is shown in Figure 3.18. According to equation (3.6), the difference between these two measures of uncertainty is the functional complexity of the LEACH algorithm. Figure 3.16 depicts the functional topology of the LEACH algorithm and it shows that the topology has dense local connections (intra-cluster connections), whereas the inter-cluster connections are sparse. The dense intra-cluster connections result in low scalability of the algorithm, due to the need of interacting with all nodes that belong to the cluster in order to add a new node. In other words, in order to add a node to a cluster,

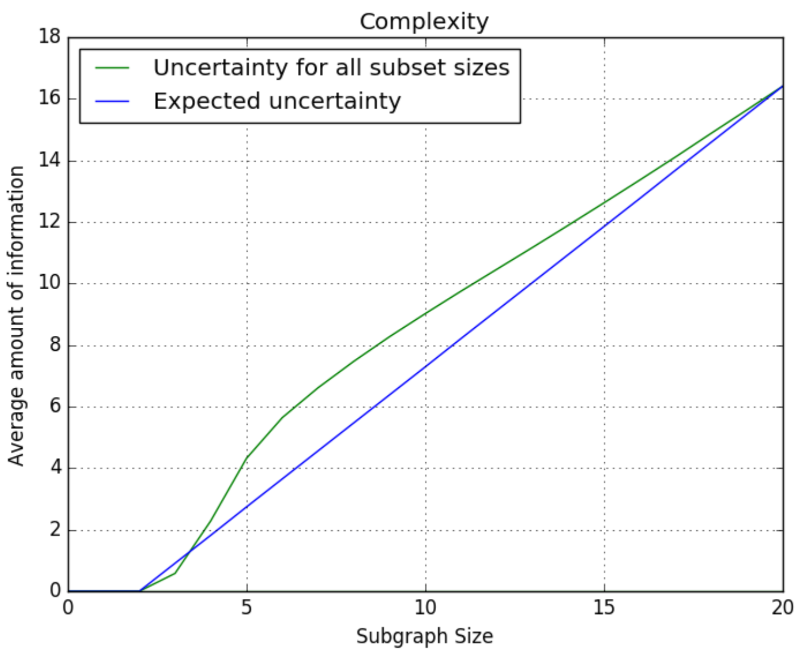


Figure 3.18: Functional complexity of the LEACH algorithm; the functional topology has twenty nodes and the nodes are divided into 4 clusters; the maximum scale size R is 2; the functional complexity is the sum of the difference between the green and the blue curves.

all nodes that belong to the cluster have to update their information. Our analysis shows that for a network of twenty nodes which are divided into four clusters the average number of messages sent when a node joins the network is 3.75. The dense local connections result in smaller values of the average uncertainty of interactions for subsets with size three than we expected from the calculation performed on the whole topology. Figure 3.18 shows that the uncertainty for subgraphs with the size greater than or equal to four is much higher than expected from the uncertainty of the whole system. This is because of the sparse inter-cluster connections, due to which we can find a lot of subgraphs with these sizes which have uniform distributions of links among nodes. For example, in Figure 3.16, if we choose any two ordinary nodes that belong to different clusters, we can create a sparse subgraph with size five. The functional complexity of the LEACH algorithm is 19.24, which is, as we will see below, relatively low compared to an algorithm that is highly scalable.

Figure 3.19 depicts the functional complexity of the HCC algorithm. As shown in Figure 3.17 the functional topology of the HCC algorithm has sparse intra and inter cluster connections. As the links in the functional topology represent functional dependencies between nodes, a sparse connectivity pattern indicates weak dependencies which result in high scalability. In other words, adding a new node to the network means simply sending a single message to one of the nodes in the topology, declaring it as the parent of the given node. Therefore, the information about a new node joining the network does not have to be transmitted throughout the cluster. The functional complexity of the HCC algorithm is 38.31, which is high compared to the functional complexity of the LEACH algorithm.

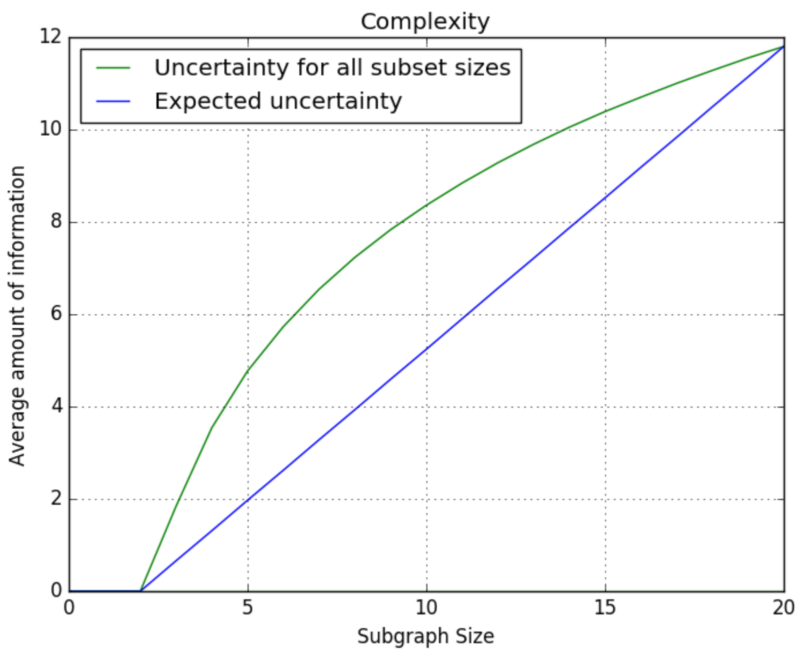


Figure 3.19: Functional complexity of the HCC algorithm; the functional topology has twenty nodes; the maximum scale size R is 2; the functional complexity is the area between the green and the blue curves.

The energy efficiency of the HCC algorithm is 0.61, which is low compared to the LEACH algorithm (energy efficiency is 1.08) for the same number of nodes which are divided into four clusters.

Table 3.4 shows how the functional complexity, the energy efficiency and the scalability of the LEACH algorithm change for different number of clusters. As shown the functional complexity increases until the number of clusters reaches sixteen, and then it starts decreasing. This is due to the fact that if the number of nodes is fixed, increasing the number of clusters decreases the number of nodes per cluster, which makes the graph more and more sparse until it reaches a point where the graph is not clustered any more and each node talks directly to the BS, i.e. a star graph. If the number of nodes per cluster decreases, the number of intra-cluster connections decreases, and hence, the implementation becomes more scalable. On the other hand, the implementation becomes less energy efficient, due to the increasing number of nodes that communicate directly to the BS.

Changing the number of clusters for the HCC algorithm does not affect the functional complexity of the implementation, because the functional topology does not change. As each node transfers information to its parent node, increasing the number of clusters decreases the number of nodes per cluster, but the information paths do not change (and therefore the functional topology does not change). As the information paths do not change and the information travels according to the established BFS routes, the main purpose of clustering is to logically group nodes in order to provide scalability (due to the hierarchical control

Table 3.4: Functional complexity, scalability and energy efficiency of the LEACH algorithm for different number of clusters; the functional topology has twenty nodes.

#of clusters	**3**	**4**	**5**	**6**	**16**	**19**
C_F	14.35	19.24	22.69	25.55	32.4	31.85
Energy efficiency	1.91	1.08	0.72	0.51	0.07	0.05
Avg. #msg. if node joins	5.33	3.75	2.8	2.16	1	1

architecture) and enable aggregation/fusion of the sensing information at the chosen cluster-heads.

Our analysis confirms the observation made by the authors of [109], which highlights the trade-off between scalability and energy efficiency. With our complex systems science approach we showed that these aspects of different clustering algorithms can be analyzed together by analyzing the functional complexity of the specific implementation. According to Table 3.4, increasing values of C_F lead to the increase of scalability and the decrease of energy efficiency.

3.6 Conclusion

The growing size and heterogeneity of telecommunication networks leads to a need to change the way we analyze and model them. Also the rapid evolution of network services demands a new approach to analyze them. The aim of this Chapter was to contribute to this new way of studying networks. We focus on network functions as building blocks of services. We consider network functions as complex systems. In order to provide a new approach to analyze and understand the impact of complex functional relationships between system entities, we developed a new framework that allows us to visualize an implementation of a network function with graphs which we call functional topologies. We provide several examples that show how to map a network function into a functional topology. The graph representation of the implementation allows us to focus on the relationships between entities rather than the entities themselves.

The next step after mapping an implementation of a network function into a functional topology was to provide a metric that quantifies the organizational structure of the topology. Our functional complexity (C_F) captures the variety of roles that each node in the topology has and the variety of structural patterns present in the topology. C_F quantifies the deviation of the implementation of a network function from the non-complex model of itself. In other words, C_F quantifies how much information we can not capture simply by studying the whole system, due to the complex relationships that exist between smaller subsets of the system. The quantification of this deviation as presented in Section 3.3 provides a new approach to understand increasingly complex telecommunication networks.

In order to study the impact of different structural patterns on the functional complexity in Section 3.4, we start by analyzing the complexity of simple graph structures (bus, ring, star and mesh). Additionally, we provide a detailed study that investigates the impact of several graph theory metrics on the functional complexity. We investigated the correlation between the combination of the graph theory metrics and complexity, showing the absence of high correlation between any single graph theory metrics and the functional complexity. This analysis allowed us to make conclusions about the organizational structures that are needed in order to achieve high complexity.

In this Chapter we also analyzed the functional complexity of two different implementations of the frequency allocation function (random and self-organizing). First, in Section 3.2 we explain how to map these implementations into functional topologies, and in Section 3.4, we apply our complexity metric to quantify the functional complexity of these implementations. We showed that the random frequency allocation has zero complexity (and represents therefore a non-complex implementation), whereas the self-organizing implementation shows higher complexity. This allows us to make a connection between our results and the results from [28], where the authors calculated the complexity of the outcome of these implementations and obtained the same kind of results. This suggests that a complex implementation results in a complex outcome of the function.

The growing interest in WSNs due to the various applications in Internet of Things (IoT) results in a great variety of clustering algorithms that support these applications. Very often, one compares these algorithms based on certain properties (e.g. fault-tolerance, delivery delay, energy efficiency, scalability). Comparing these algorithms based on a broad range of properties is very difficult. Additionally, a lack of understanding exists when we try to comprehend the mechanisms that lead to these properties. That is why, in Section 3.5, we apply a complex systems science approach to analyze these mechanisms. We focus on two clustering algorithms, i.e. the LEACH and the HCC algorithms. Our goal is to investigate the mechanisms that provide high scalability in the case of the HCC algorithm and high energy efficiency for the LEACH algorithm. We also study the trade-off between these two network properties. After briefly introducing the algorithms, we map their implementations into functional topologies, and calculate the corresponding functional complexities. We show that high functional complexity indicates greater scalability of the implementation. We also show that increasing values of the functional complexity with the number of clusters, indicates lower energy efficiency. We then highlight that a sparse functional topology results in higher complexity. Therefore, our functional topology explicitly explains both the higher scalability (sparse graph) and lower energy efficiency of the HCC algorithm as compared to the LEACH algorithm. In [122], we also apply our functional framework to model the self-organized time division multiple access (SOTDMA) algorithm in the VHF maritime mobile band for universal shipborne automatic identification systems (AIS).

Our end goal, after investigating the relationships between functional complexity and different network KPIs, is to use functional complexity to tune the tradeoff between these network KPIs. The main advantage of using a metric like functional complexity instead of the KPIs to tune and optimize the network is that functional complexity can be computed offline without relying on data collected in the network, which speeds up the network optimization process. On the other hand, as shown in [123], the first step in the process of calculating a KPI is to collect measurements. In order to collect the measurements the network has to be operational, meaning that the process of optimizing certain aspects of the network is possible only by monitoring the real network dynamics. As presented in this Chapter, the functional topologies capture the relations between functional parts of the network based on the rules that are defined by the communication protocols themselves. In other words, the functional topology captures certain network dynamics without relying on measurements. Therefore, the functional complexity can be calculated even if the network is offline. By showing high correlation between the complexity metric and network KPIs, we show that a network operator would be able to optimize the network aspects related to those KPIs before the network is operational. Overall, our complexity metric quantifies telecommunication networks in terms of their functional relationships and provides a wholly new approach to understanding the operation of networks. More precisely, the complexity metric quantifies the relationships employed during the operation of network functions. This provides a new approach to study the next generation networks.

4 Degeneracy Framework for the Analysis of In-Network Computing in IoT

Degeneracy Framework for the Analysis of In-Network Computing in IoT

4.1 Introduction

The purpose of traditional data networks such as the Internet is to enable end-to-end information transfer. Information streams in such networks are carried across point-to-point links in which intermediate nodes forward data packets without meaningfully modifying their payloads. Internet of Things (IoT) networks, however, perform actions or make decisions powered by access to raw or processed data gathered by spatially distributed "things". Distributed in-network computation approaches, i.e. the processing of raw data within the IoT network, are becoming increasingly popular in that they can potentially achieve higher energy efficiency, lower computational delay, and higher robustness compared to the traditional approach, which involves transmitting raw data to the sink, then performing the computation and, for certain applications, activate the corresponding actuator(s).

As IoT networks become increasingly larger, the communication within the network becomes complex enough as to resemble a thermodynamic system of many interacting objects. This leads to a physics-inspired study of their behavior using statistical mechanics and a concept known as *degeneracy*. Degeneracy arises from structurally different configurations of a system (microstates) having functionally similar/identical macroscopic properties (macrostates). A concise definition of degeneracy comes from a modern review of the use of the term: "degeneracy describes the ability of different structures to be conditionally interchangeable in their contribution to system functions" [124]. In the context of in-network computing, we view these different structures as functional topologies: subsets of the network which enable the computation of a distributed function, as introduced in Chapter 3.

Scope and Contribution

Figure 4.1 highlights the focus of this Chapter. Similarly to the work in Chapter 3, this chapter examines the *Analysis* and *Modeling* layers, but we shift our focus from *Complexity* to *Degeneracy* by studying concepts related to the following question: "How to identify

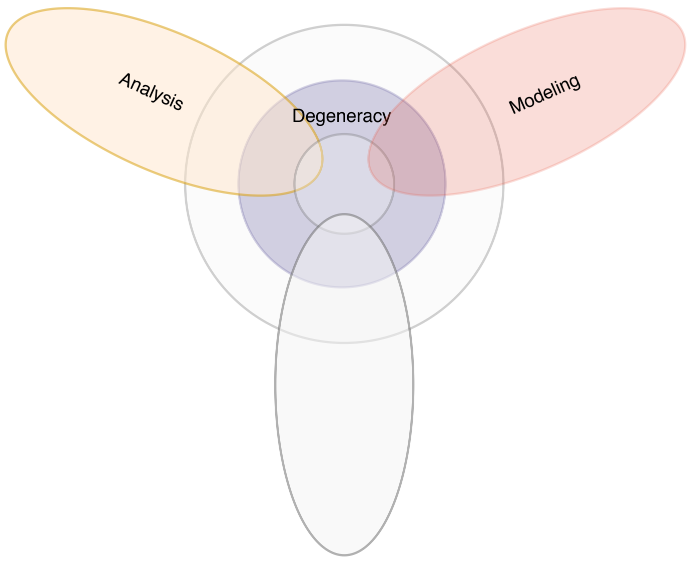


Figure 4.1: Thesis roadmap: In this Chapter we focus on the *Analysis* and *Modeling* layers to study *Degeneracy*.

macro-scale/system-level topologies that are constructed out of diverse micro-scale structures (i.e. degenerate structures)?" Our studies result in an algorithm that allows us to quantify the multiplicity of functional topologies (micro-scale structures) that can be created on top of a physical topology (macro-scale topology), while considering the same set of constraints.

We start by modeling the execution of a distributed computation in an IoT network with functional topologies. This modeling technique allows us to define *degeneracy* in such an environment by using concepts from statistical mechanics. We define degeneracy for in-network computing as the multiplicity of possible options available within the network to perform the same function with a given macroscopic property (e.g. delay). Then, we present an efficient algorithm to determine all these alternatives. Our results show that by exploiting the set of possible degenerate alternatives, we can significantly improve the successful computation rate of a symmetric function, while still being able to satisfy requirements such as delay or energy consumption.

The intended application of an IoT network usually defines the type of functions being considered. This in turn naturally defines the observables (macro-states) of interest to analyze these functions. For example, in alarm networks, the quantity of interest is the threshold value (i.e. max or min) of the sensor readings, whereas in environmental monitoring, a relevant statistic would be the mean or the trend measurements (i.e. increase, decrease or oscillation over time). The distributed computing literature [125–129] usually assumes an error free scenario, i.e. no node failures, and mainly focuses on the optimization of the computational rate and the communication cost (e.g. energy, delay). On the other hand, we are interested in a more realistic network setup, in which node failures do exist, and we study the impact of degenerate computation and communication paths on the success rate of the computation.

Authors like [125–131] address communication aspects of distributed in-network computation. They focus on the optimization of the communication/computation parameters, like computational complexity, energy efficiency, computational throughput and computational delay. In contrast, we focus on the degeneracy of the network by finding the multiple feasible computational graphs that satisfy a given requirement (e.g. delay), rather than a single optimal computational graph. Our results show that such an approach makes it possible to fully harness the computational capability of the network and significantly increase the robustness of the computation, while still being able to satisfy requirements such as delay or energy consumption.

To date, the in-network computing literature features analysis of functions such as distributed Fast Fourier Transform, Singular-Value Decomposition [132], MapReduce and Dryad [127], collaborative compressed sensing [133], distributed neural networks [130], and Kalman filtering [134]. The majority of IoT applications, however, rely on relatively simple aggregate functions like max, min, count and average [125, 131, 135–139]. The distributed computation of these functions is usually modeled with tree structures, e.g. Steiner trees [125, 126]. Closely related to the approach proposed in our work is the standard Steiner Tree Packing problem[1] [140] and its well know relaxation, i.e. the Fractional Steiner Tree Problem. Instead of using the standard Steiner Tree Packing, we rather extend the search from minimum weight Steiner trees to all existing Steiner trees in which the sum of the weights on the links is lower than a predefined value. Due to the need to discover all trees that satisfy a set of requirements rather than searching for the minimum weight trees, the search space is expanded when compared to the standard Steiner Tree Packing problem.

Closely related to the problem discussed in this Chapter is the Virtual Network Embedding problem. Network virtualization, as one of the most promising technologies for future networks, enables the dynamic instantiation of virtual/functional nodes throughout the network [141]. This leads to innovation in terms of technology development and business models. Future architectures will be based on the Infrastructure as a Service [142], which introduces two new players on the market: (1) Infrastructure Providers: who deploy and maintain the physical network; and (2) Service Providers: who are in charge of deploying end-to-end services through the instantiation of virtual nodes on top of the physical infrastructure. As emphasized by [143], the Virtual Network Embedding problem, the embedding of virtual networks in a physical topology, is the main resource allocation challenge in network virtualization. This involves using embedding algorithms that optimize the resource discovery and allocation to satisfy different embedding objectives (e.g. provide QoS-compliant embeddings, maximize the profit of the infrastructure providers, provide resilient virtual network embeddings). The Virtual Network Embedding problem is, among

[1]The Steiner Tree Packing problem is to find the maximum number of edge-disjoint subgraphs of a given graph that connect a given set of required points.

others, being applied to Service Chain Provisioning for Network Function Virtualization in communication networks; and optimization of the virtual node placement for Cloud Service Providers, Application Service Providers and Internet Service Providers. Unlike the usual approach taken in the literature [144–147], which focuses on the matching of the physical resources with the demands of the virtual networks, our approach builds on top of those studies and focuses on the analysis of the structure that connects the physical resources. By studying the physical connectivity between the network resources, while considering all communication constraints that affect the interoperability between them, we are able to quantify the size of the feasible space for the embedding problem.

As presented above, the literature on in-network computing and Virtual Network Embedding focus on the embedding of functions on top of the physical topology. Service Providers are mainly concerned with these this process. However, we focus on the study of degeneracy, which defines the cardinality of the feasibility space for the embedding problem. Therefore, degeneracy can be used to evaluate the potential of a physical network to accommodate embeddings. In addition to evaluating a deployment, degeneracy could also be used to assess potential changes to the network topology. For example, in case of physical node rewiring, only the rewiring that will result in higher degeneracy should be allowed. These processes are of interest to Infrastructure providers.

The main contributions of this Chapter are:

- We propose an efficient algorithm that generates all functional topologies satisfying a given delay requirement, for any physical topology;
- We analyze the time complexity of the proposed algorithm, and its applicability to realistic network topologies;
- We provide a formal definition and calculation of degeneracy and the related concept of redundancy in terms of distributed computing over IoT networks;
- We provide a comparison of degenerate in-network computing and a traditional multi-hop scheme that selects a single graph for the computation, showing that it is possible to significantly increase the probability of successful computation by exploiting degeneracy.

4.2 Degeneracy and Redundancy Framework

We define a physical network $G = (V, E)$, as a simple graph that contains no self-edges or repeated edges, where the set of nodes V represents the physical nodes in the network (e.g. sensors in an IoT network), and the set of edges E represents the physical links between the nodes that can interact directly with each other. Mathematical functions, f, can be represented as subgraphs H of the overall network which we refer to as functional topologies. Even though the definition of functional topologies provided in Chapter 3 is much broader, in terms of in-network computing the functional topologies are directed, rooted trees, i.e. Steiner trees, with all edges pointing towards the root, Y. This root/sink node is where the result of a particular function f must reach. The leaves of this in-tree are the inputs of the function, X. These inputs may take the form of entries in a distributed database or measurements of the environment such as temperature. The set of inputs, $x_i \in X$, are generated in nodes v_i.

The individual operations of a given function, $f(X)$, are mapped to specific nodes in the network. Each functional topology, H, is some subset of $U \subset V$ and $D \subset E$ in which each node and edge is involved in computing and routing $f(X)$. Multiple, unique functional topologies which model the same function, $f(X)$, can be chosen from the same physical topology, G. Distinct functional topologies that perform the same calculation and result in the same observable macrostate of the network function are considered to be degenerate. The overall delay associated with a given functional topology is the principal macrostate of interest in this Chapter. We define the delay as the maximum graph distance between the sink node, Y, and any other node, $u \in H$. Alternative macrostates of potential interest are energy efficiency, which could be approximated by the number of edges in functional topology H, and computational throughput (number of function calculations per unit delay).

In [136], the authors made the distinction between two classes of network functions, namely divisible and indivisible. Examples of divisible functions include max, min, mean and computing the frequency histogram of sensor measurements. On the other hand, examples of indivisible functions include Fast Fourier Transform, Singular-Value Decomposition, distributed neural networks. Denote a set $W = \{1, \ldots, |W|\}$ and a subset $S = \{i_1, \ldots, i_k\} \subset W$, where $i_1, < i_2, < \ldots, < i_k$. Let x_S be a set $\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$. A function, $f : x_S \to y$, is divisible if given any partition of S, $\Pi(S) = \{s_1, \ldots, s_j\}$ of $S \subset W$, there exists a function $g^{\Pi(S)}$ such that for any x_S,

$$f(x_S) = g^{\Pi(S)}(f(x_{s_1}), \ldots, f(x_{s_k})). \tag{4.1}$$

Otherwise, f is indivisible. A special case of divisible functions are so called *Symmetric Functions*, which are divisible functions that are invariant with respect to permutations (σ) of their arguments:

$$f(x_S) = f(\sigma(x_S)), \forall \sigma \tag{4.2}$$

4.2.1 Degeneracy of Functional Topologies

Degeneracy can be considered in terms of these two classes of network functions. We define strong degeneracy to be the multiplicity of functional topologies of an indivisible network function. We define weak degeneracy to be the multiplicity of functional topologies of divisible functions. As previously discussed in Section 4.1, aggregate functions are of great importance for distributed computing, and since they are symmetric, this Chapter focuses on weak degeneracy.

For a physical topology, G, a network function, f, and an input and output set of nodes, X and Y, we can define *weak degeneracy* of the physical topology, G, with respect to an observable (e.g. delay, energy efficiency) as the multiplicity of functional topologies,

$$g_W(G, X, Y, f, d) = |\{H(G, X, Y, f, d)\}|, \tag{4.3}$$

with a given delay, d. For completeness, *strong degeneracy* is defined as the multiplicity of functional topologies on top of a physical topology G with respect to an observable, with input nodes X, performing an indivisible function in which the sub-operations must be performed in an exact ordering, $g_S(G, X, Y, f, d)$.

Since $f(X)$ is symmetric, we can perform the calculation using any partition s of X and combine the result of each subset $f(x_s)$ in any order. Weak degeneracy of a full mesh physical topology is therefore dependent on the number of ways of partitioning a set, known as the Bell number,

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_n, \tag{4.4}$$

for a set of $n+1$ elements and $B_0 = B_1 = 1$. It should be noted that the Bell number is an upper bound for the weak degeneracy. The weak degeneracy of a generic physical topology has to further account for the number of mappings of these partitions to a given functional topology. This is the number of ways we can route each partition such that the suboperation $f(x_{s_i})$ on partition s_i occurs when the elements of partition s_i intersect for the first time.

Our degeneracy analysis allows us to both identify the set of subgraphs which perform a function and to sort these with regard to observables like delay, energy efficiency or computational throughput. This insight may be used in future to improve computational throughput, computational resilience, deploy superior network configurations, or to analyze the degeneracy potential of a routing protocol, i.e. how many of the feasible functional topologies can a routing protocol discover.

4.2.2 Redundancy of Functional Topologies

Degeneracy arises from structurally different functional topologies exhibiting functionally identical properties. We can also define redundancy in functional topologies, which is related to degeneracy, with two redundant functional topologies not sharing any common nodes or edges apart from their common X and Y. The redundancy between two functional topologies, $r(H_a, H_b)$, is defined as

$$r(H_a, H_b) = \begin{cases} 1, \text{ if } U_a \cap U_b = X \cup Y, \\ 0, \text{ otherwise.} \end{cases} \tag{4.5}$$

We define the average redundancy,

$$R(G, Y, f) = \frac{1}{|\{X\}|} \sum_{\{X\}} \frac{1}{N_X} \sum_{\substack{\{H_i, H_j\} \\ i \neq j}} r(H_i, H_j), \tag{4.6}$$

as the mean over the number of functional topologies which are redundant to any other given functional topology. The sum is over all N_X possible functional topologies and the set of all possible input sets, $\{X\}$. The total redundancy gives a measure of the number of ways to choose alternate functional topologies that bypass nodes in case of node/link failures.

It is to be noted that both degeneracy and redundancy directly affect the computational robustness of in-network computing. However, while degeneracy defines the cardinality of the feasibility space for the functional embeddings, redundancy informs us about the potential of the physical topology to perform parallel computations under the same delay constraints.

4.2.3 Find All Weakly Degenerate Functional Topologies

In the context of in-network computing, a functional topology describes the interactions between network nodes in the course of computing a distributed function. For the in-network computing of divisible functions, functional topologies are Steiner trees, i.e. they include only the nodes and edges from the physical topology that are involved in the computation and/or routing of the function.

To generate a functional topology, consider a partition S of the input set X. For each subset $s \in S$, the function f acts on the elements in s at their first meeting point node in H. For each input set X we construct a set of paths, $P_X = \{p_1, p_2, \ldots, p_{|X|}\}$, where $p_i = (x_i, v_1, \ldots, Y)$. Let us denote by P_s the set of paths between each node $i \in s$ and Y, i.e. $P_s = \{p_i | i \in s\}$. The sub-operation $f(s)$ occurs where the paths P_s meet for the first time. For example, let us look at an input set $X = \{x_1, x_2, x_3, x_4\}$. We construct paths $P_X = \{p_1, p_2, p_3, p_4\}$, where $p_1 = (x_1, v_1, \ldots, v_l, \ldots, Y)$, $p_2 = (x_2, u_1, \ldots u_j, \ldots, Y)$, $p_3 = (x_3, w_1, \ldots, w_m, \ldots, Y)$, and $p_4 = (x_4, t_1, \ldots t_q, \ldots, Y)$. Let us consider a possible partitioning of the input set X into two partitions - $s_1 = \{x_1, x_2, x_3\}$ and $s_2 = \{x_4\}$. The sub-operation $f(s_1)$ occurs at the meeting point $v_l = u_j = w_m$. The sub-operation

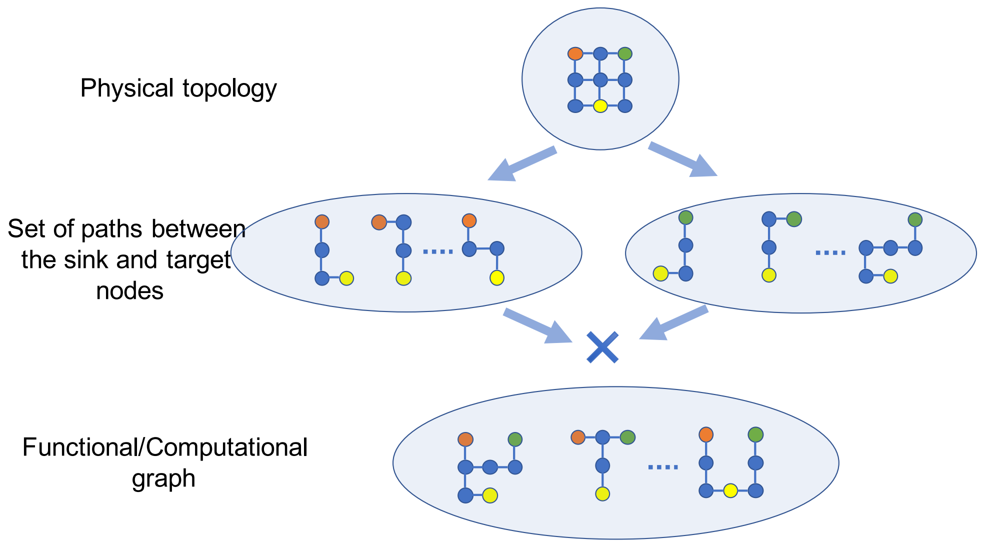

Physical topology
Set of paths between the sink and target nodes
Functional/Computational graph

Figure 4.2: Process to identify all functional topologies from a generic physical topology. The first step is to find all simple paths from the input nodes (green and orange) to the sink node (yellow). Then we take the union of those paths to generate a set of subgraphs, that is being analyzed to identify unique functional topologies.

Algorithm 1 Our algorithm to generate all functional topologies of a graph G, for a symmetric function f, with input nodes X, sink node, Y, and a maximum delay cutoff $d_{\max}$.

```
1: procedure FINDFTS($G$, $X$, $Y$, $f$, $d_{\max}$)
2:     Find all directed, simple paths, $\{p_{x_i}\}$, from all source nodes, $X$, to the target node,
       $Y$ of length $d_{\max}$ or less.                               ▷ e.g. Breadth-First Search
3:     Take the union of edges and nodes of all possible combinations, $C$, of paths from
       different input nodes $p_{x_i} \forall x_i$.
4:     if A given combination, $c \in C$, is not a tree (contains one or more cycles). then
5:         Find all unique spanning trees of $c$.
6:         for All unique spanning trees do
7:             if Spanning trees has leaf nodes, $L \not\subseteq X$. then
8:                 Remove leaf nodes, $L$, since they are not an input.
9:     Remove any duplicate functional topologies.
```

$f(s_1)$ is then rooted to Y following any of the original paths, i.e. $(v_l, \ldots, Y)$, $(u_j, \ldots, Y)$, or $(w_m, \ldots, Y)$. The next sub-operation $f(s_1, s_2)$ is then computed at the meeting point of any chosen original path from s_1 and the path p_4. Paths p_1, p_2, p_3, p_4 must contain at least one crossing point since they have a common root at Y. Since f is divisible, the order in which we combine paths between any input node $x \in X$ and Y does not matter.

If we consider a specific path from each input node X to Y and take the union of the nodes and edges in each path, we generate a subgraph of G rooted at Y and with leaves X. If the union of the paths join and later diverge, one or more undirected cycles have been formed. Each independent path around the cycle results in a different functional topology. These functional topologies can be identified by finding the unique spanning trees of the cycle-containing subgraphs and removing the resulting leaves which are not inputs, X, since they do not perform a computational or routing role.

We assume that G is connected and there exist paths from each input node to the sink. We restrict the maximum path length (delay), $d_{\max}$, to the maximum graph distance between the sink and any other node, known as the eccentricity. Figure 4.2 depicts this process of extracting the functional topologies from a physical topology and Algorithm 1 outlines the proposed algorithm.

Our algorithm extends the standard Steiner Tree Packing problem and its relaxation, i.e. the Fractional Steiner Tree Problem (which are known to be NP-hard [140]), by extending the search from minimum weight Steiner trees to all existing Steiner trees in which the sum of the weights on the links is lower than a predefined value. To the best of our knowledge, this problem has not been addressed in the literature so far.

Complexity Analysis

In Algorithm 1, the most computationally expensive task is the search algorithm that finds all directed, simple paths p_{x_i}. For example, in general the time complexity of the Breadth-First Search algorithm can be expressed as $O(|V| + |E|)$. However, Algorithm 1 defines the maximum path length d_{max}, which reduces the search space, resulting in a time complexity that can be expressed as $O\left(b^{d_{max}}\right)$, where b is the branching factor (i.e. the average out-degree) [148]. The steps after the search algorithm include:

- combining the discovered paths - the time complexity of the union operation is $O(n)$, where n is the number of elements/nodes in the set. In our case, due to the upper bound in terms of path length, the maximum number of elements in the edge set is $|X|\, d_{max}$. Hence the time complexity of the union operation is $O\left(|X|\, d_{max}\right)$.
- finding the spanning trees - the time complexity of the Kruskal algorithm [149] to find spanning trees is $O\left(m \log\left(n\right)\right)$, where m is the number of edges and n is the number of nodes in the graph. Due to our limit in terms of d_{max} the time complexity can be expressed as $O(|X|\, d_{max} \log\left(|X|\, d_{max}\right))$

The time complexity of Algorithm 1 can be calculated as:

$$T(n) = b^{d_{max}} + |X|\, d_{max} + |X|\, d_{max} \log\left(|X|\, d_{max}\right) \tag{4.7}$$

The time complexity of the search algorithm is the dominant one, meaning that the complexity of the whole algorithm can be expressed as:

$$T(n) = O\left(b^{d_{max}}\right) \tag{4.8}$$

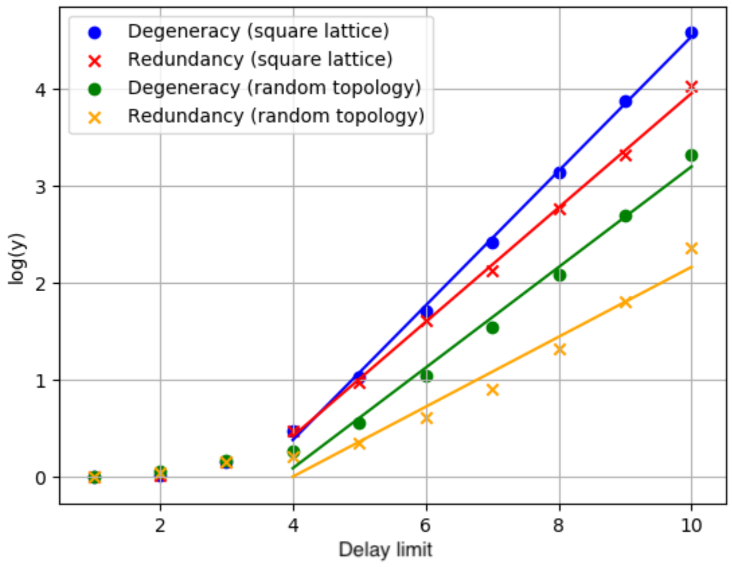


Figure 4.3: Log plot (base 10) of cumulative degeneracy and cumulative redundancy. For the square lattice, the slopes of the linear fits for the cumulative degeneracy and cumulative redundancy are: 0.69 and 0.59, respectively. For the random topology, the slopes of the linear fits for the cumulative degeneracy and cumulative redundancy are 0.51 and 0.36, respectively.

The maximum path length between nodes in a sensor network (d_{max}) is a design parameter that is usually set to a small value to ensure low latency and high energy efficiency. The network topology is usually planned in a way that keeps this value small even for a growing network size. An example of a network that is constantly growing without affecting the average path length between the nodes is the Internet (its topology is believed to exhibit a power low distribution [150]). Therefore, Algorithm 1 is efficient enough to be used in a realistic network scenario.

4.3 Results and Implications

We now present the numerical analysis of the degeneracy and redundancy of three different physical topologies: (1) an 11×11 square lattice; (2) a randomly placed sensor network with the same number of nodes; and (3) a real-world sensor network deployed at the Intel Research Lab in Berkeley [151]. In the lattice case, we place the sink node, Y, in the centre of the lattice. In the case of a random topology, we consider an area of 1×1 km, place the sink node at the centre of the area, and randomly distribute the remaining 120 nodes. The edges are chosen to connect each node to at most four of its closest neighbors within the range of 20 m - short range communication. It is worth noting that this results in a non-uniform distribution of the edges. In the case of the real-world sensor network deployed at the Intel Research Lab, the network consists of 54 Mica2Dot sensors, which collected data for about 5 weeks. We used the averaged connectivity data, consisting of sender id,

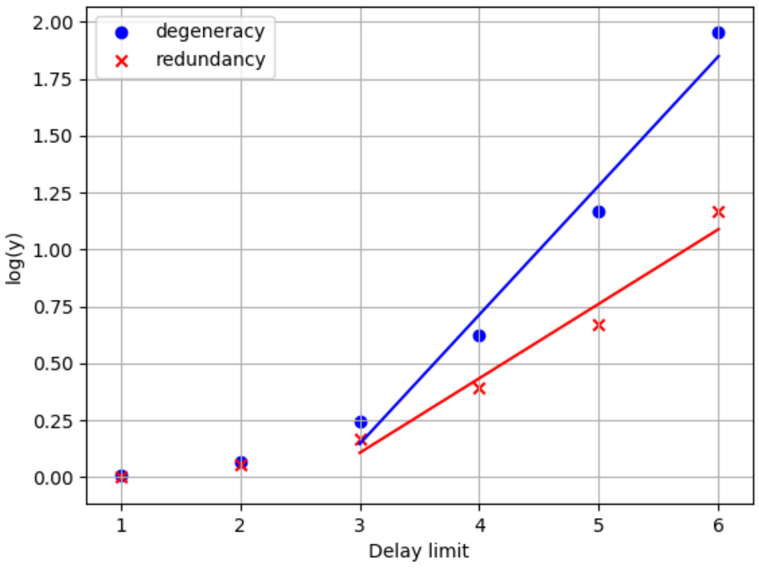


Figure 4.4: Log plot (base 10) of cumulative degeneracy and cumulative redundancy for the Intel dataset. The linear fits for the cumulative degeneracy and cumulative redundancy are: 0.66 and 0.39 respectively.

receiver id, and probability of a message from a sender successfully reaching a receiver, to compute the physical topology. The simulation involves calculating all functional topologies for randomly sampled input node pairs, X, which are chosen to be within three hops from each other, since in-network computation is typically used for data collected by nearby sensors. To calculate the average number of degenerate functional topologies with a given delay, this process is repeated 500 times. The standard errors on the means are estimated using bootstrap resampling of all simulations, and are in-fact smaller than the plot markers.

The cumulative degeneracy of functional topologies and cumulative average redundancy for the lattice and random topologies are shown in Figure 4.3, while the same results for the network deployed at the Intel Research Lab are shown in Figure 4.4. For all three topologies the maximum delay considered is equal to the eccentricity of the sink node, which is 10 in the case of the lattice and random topology, and 6 for the Intel dataset. For the lattice case, the cumulative number of degenerate functional topologies (blue markers) is seen to increase exponentially with the increasing delay limit. This same exponential behavior has also been seen in independent experiments on different lattice sizes. It is a result of the branching process associated with functional topologies of increasing size gaining access to more and more nodes and their respective edges. The cumulative redundancy of functional topologies (red markers) also increases exponentially with delay limit. Although the cumulative degeneracy and redundancy are lower in the case of a random topology than those observed in a lattice, they still increase exponentially as the delay limit increases. As can be observed by comparing Figure 4.3 and Figure 4.4, the degeneracy values of the Intel deployment are similar to the square lattice deployment, whereas the redundancy values are

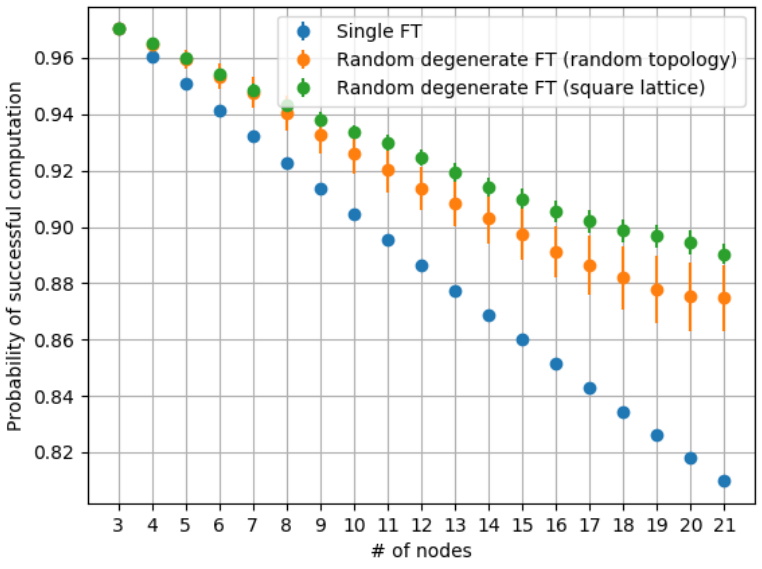


Figure 4.5: Probability of successful computation vs number of nodes involved in the computation. The blue dots refer to a single functional topology; the green and orange dots refer to a randomly selected functional topology that can perform the same function for the square lattice and random topology respectively. The delay limit for all calculations is set to be equal to 10.

lower compared to the square lattice and higher as compared to the random network deployment. The increase in cumulative redundancy demonstrates that the ability for physical topologies to perform parallel computing using independent functional topologies is higher for less restrictive delay limits.

Using the numerical estimates of degeneracy, we now examine the computational robustness of in-network computing. Robustness defines the stability of the system against external perturbations and internal variability. Our results show the benefit of exploiting degeneracy on the probability of a successful computation, which is used here as a proxy for computational robustness. Let us first define the probability of computational success, α_i, at node i as the number of successful computations as a fraction of the total number of attempts. The probability of a node failing during the computation of a function is then found to be,

$$P_{f_i} = P_{c_i} \cdot (1 - \alpha_i), \tag{4.9}$$

where, P_{c_i}, is the probability of a node occurring in any feasible functional topology. A computation may fail at a node for a number of reasons such as excessive load or during a sleep cycle. The probability of a successful computation is given as

$$P_{S_c} = \prod_{i \in U} (1 - P_{f_i}), \tag{4.10}$$

where U are the nodes in the functional topology that performs the computation. Figure 4.5

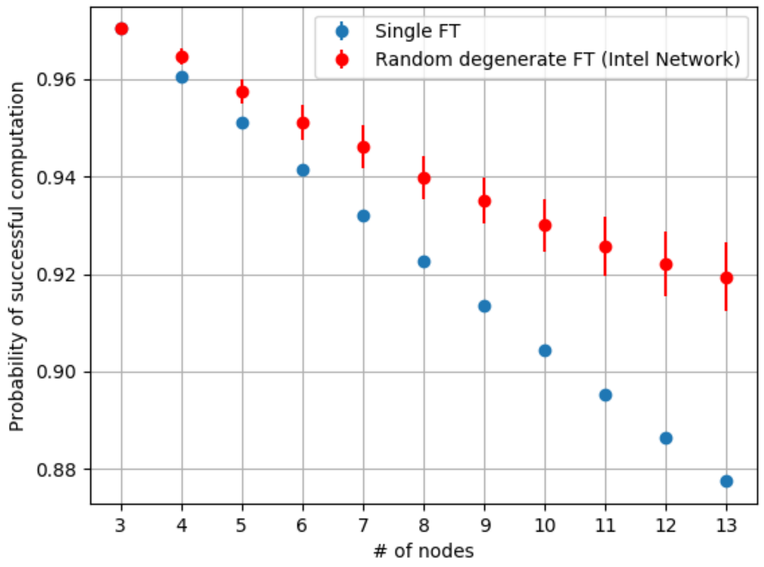


Figure 4.6: Probability of successful computation vs number of nodes involved in the computation for the Intel Network. The blue dots refer to a single functional topology; the red dots refer to a randomly selected functional topology that can perform the same function for the Intel Network. The delay limit for all calculations is set to be equal to 6.

compares the probability of successful computation using a single functional topology, e.g. selected as the optimal computational graph in terms of delay or energy consumption, and a random selection of one of the multiple functional topologies that can perform the same computation while satisfying a maximum delay requirement (10 for the results in Figure 4.5) for the lattice and random topology. Figure 4.6 shows the same results for the Intel network deployment. In this case the delay limit is 6. In the case of a single optimal functional topology, $P_{c_i} = 1$ for all nodes, since all nodes in the optimal graph must be used for in-network computing. Considering the degeneracy of functional topologies from Figure 4.3, each computation can potentially be performed by using any of the degenerate functional topologies. Therefore, we estimate P_{c_i} for the randomly selected functional topology, that is going to perform the computation, as the frequency of occurrences of the node i in all the degenerate functional topologies. This P_{c_i} is then used to compute the corresponding P_{S_c} according to equations (4.9) and (4.10).

Results are then averaged over all the 500 input pairs. The results in Figure 4.5 show that it is possible to significantly increase the probability of successful computation by exploiting the multiplicity of functional topologies, i.e. the degeneracy.

The previous results show that the awareness of computational alternatives allows us to increase the probability of successful computations. We now take a closer look at the square lattice topology to understand how the structure of the physical topology affects the degeneracy and redundancy in terms of in-network computing. Figure 4.7 shows the heat-map of the square lattice topology representing the probability of each node being included in

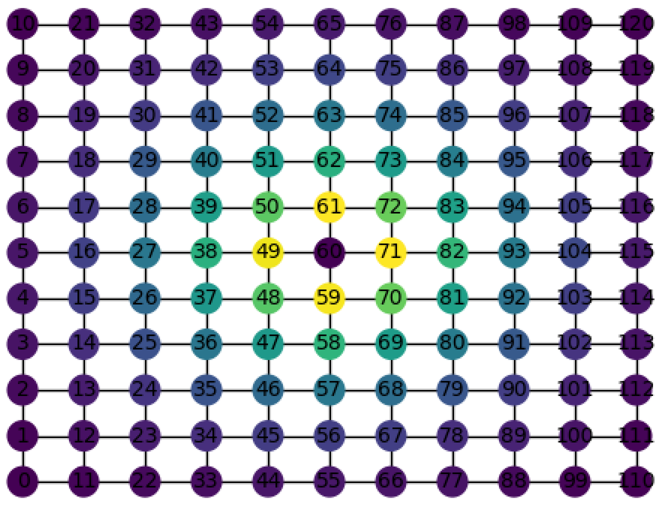


Figure 4.7: Heat-map of the square lattice topology showing the probability of a node being included in a functional topology. Brighter colors represent higher probabilities. The sink node is in the center of the square lattice and the input nodes are randomly sampled and within three hops from each other.

a functional topology. Brighter colors correspond to higher probabilities. As expected, nodes closer to the sink are being included in more computations, i.e. more functional topologies. Therefore, the number of nodes connected to the sink node directly affects the number of ways to reach the sink node within a limited number of hops, which is defined by d_{max}. These nodes are also the communication bottlenecks of the physical topology. Additionally, Figure 4.7 shows that for example node 72 has a higher probability to be included in a functional topology than node 82. This is interesting because both nodes (72 and 82) are two hops away from the sink node. However, the difference arises from their positions and the connectivity pattern in the physical topology: due to the Von Neumann neighborhood connectivity, node 72 is located so that it provides the shortest path connectivity for more nodes (i.e. the nodes in the upper and right part of the graph) compared to node 82 (i.e. the nodes in the right part of the graph).

The regularity of the square lattice results in a fairly uniform probability of nodes being included in the functional topologies (see Figure 4.7). However, things are quite different for topologies that are not so regular, e.g. the random topology or the Intel deployment. This non-regularity explains the high variance in Figure 4.5 and Figure 4.6, and the larger gap between degeneracy and redundancy of the random topology compared to the square lattice shown in Figure 4.3, and between degeneracy and redundancy of the Intel topology in Figure 4.4.

4.4 Conclusion

In this Chapter we presented a novel way of considering in-network computing using ideas from statistical mechanics. In particular, we provide a formal definition and calculation of degeneracy and the related concept of redundancy for distributed computations in a network. We also introduce an algorithm to efficiently compute all degenerate and redundant functional topologies.

The time complexity of the proposed algorithm is exponential with the increasing path length in the network. However, the maximum path length is a design parameter which is usually kept small for a growing network size, allowing us to use the proposed approach to analyze degeneracy and redundancy of realistic network topologies.

Our results show that the cumulative degeneracy and redundancy of functional topologies increase exponentially as the accepted delay limit for the computation is increased. The analysis of different physical topologies shows that the underlying network structure affects the operation of network functions and results in different values of degeneracy and redundancy. The successful computation rate of a symmetric function is shown to be significantly higher using the set of possible degenerate functional topologies as opposed to exclusively using the optimal functional topology. This means that we can considerably improve the robustness of the computation, while still being able to satisfy requirements such as delay or energy consumption. Future work will focus on exploiting degenerate and redundant functional topologies to perform parallel computing. The case of redundant, i.e. independent functional topologies, is of particular interest in that bottlenecks can be avoided with no coordination between the nodes of different functional topologies when performing the same network function. We will investigate mechanisms to balance nodes usage for redundant topologies implementing multiple coexisting functions across the same physical network. The same mechanisms will be exploited for the case of degenerate topologies.

5 Self-Organization in Cellular Networks Based on Local Interactions

Self-Organization in Cellular Networks Based on Local Interactions

5.1 Introduction

Due to the increasing densification of cellular networks and mobile devices (e.g. IoT devices for logistics and supply chain management), the number of handovers will significantly grow. Recent 3rd Generation Partnership Project (3GPP) documents [152, 153] have identified mobility management optimization based on User Equipment (UE) mobility pattern recognition as a key issue. These changes in cellular networks impose strict requirements in terms of latency, throughput and mobility. Some of these requirements are conflicting, due to the network architecture and protocol procedures. For example, handover procedures are inevitable to ensure mobility, but at the same time they greatly affect latency. Therefore, in order to support the dynamic network demands the network control mechanisms should be automated, flexible and highly responsive. 3GPP proposes Self-Organizing Networks (SONs) solutions to tackle three types of processes in networks [94]:

1. Self-Configuration - Involves the dynamic plug-and-play configuration of newly deployed network nodes. Use cases involve the self-configuration of newly deployed Base Stations (BSs), which includes procedures like: connectivity configuration (between the BS and the core network nodes), the Automatic Neighbor Relation discovery, Physical Cell Identity and Cell Global Identity configuration.
2. Self-Optimization - Includes the optimization of coverage, capacity, handover and interference. Examples include: Mobility Load Balancing, Mobility Robustness Optimization and Random-Access Channel (RACH) optimization.
3. Self-Healing - Includes the automatic detection of network failures and correction of the corresponding network parameters. Use cases include: automatic correction of capacity problems, coverage adjustment, minimization of drive tests through UE parameter report analysis.

The SON architectures can be centralized, distributed or a hybrid solution. The SON algorithms themselves are not standardized by the 3GPP, which means that it is up to the equipment vendors and network operators to design and implement them [94].

As highlighted in [94], the main goal of the SON concept is to reduce the operating expenditure associated with the management of a large number of nodes from more than one vendor. Hence, the algorithms that have been proposed so far (e.g. Automatic Neighbor Relation discovery, Physical Cell Identity configuration, Mobility Load Balancing) are focusing on the automation of the tasks that would traditionally be performed manually. This means that the so called "cooperation" between the nodes in SON scenarios is limited and based on conservative principles that isolate different functions and resolve problems in the network based on pre-planning of conflict resolution procedures.

We take a broader approach to SONs that is based on the concept of emergence in Complex Systems Science. Emergence can be defined as the rise of novel and coherent structures, patterns, and properties through the interactions of multiple distributed elements [41]. This approach allows us to design the rules of interaction between the network nodes in a way that ensures high network performance and adaptability, without the need for isolation between different network functions and pre-planning. Unlike the 3GPP approach, which looks at automation as a way to reduce the involvement of engineers in the process of setting up the network, optimizing it and resolving known alarm scenarios, we are aiming at designing the rules of interaction between the network nodes in a way that guides the network operation towards a desired state and the changes in the network (e.g. new nodes being deployed, node failures) are taken into account in this process through local interactions. This makes the process robust and eliminates the need for separate self-organization procedures (i.e. self-configuration, self-optimization, self-healing). In other words, instead of encoding the self-organizing procedures for known network fault scenarios within the nodes (i.e. node intelligence), our approach aims at the group behavior that arises from the local interactions and gives rise to a desired network state without the consideration of every individual fault scenario. For example, faulty nodes, adding or removing nodes will not trigger different procedures. The node status changes, which affects the decision making of the surrounding nodes, but the interactions between the nodes are the same and the process guides the collective behavior towards a desired state.

Scope and Contribution

As shown in Figure 5.1, this Chapter focuses on the concept of *Emergence* and includes work on all three layers: *Analysis, Modeling* and *Design.* The following research question guides our work in this Chapter: "How to achieve global organization through limited knowledge and local interactions?" In particular, we propose a distributed self-organizing algorithm that takes advantage of locally available information to minimize the signaling overhead and delay related to mobility management functions in cellular networks.

We start by introducing a simulator that was developed for the purpose of studying self-organization in cellular networks. The simulator is completely modular, allowing the user to choose the level of abstraction (e.g. which signaling procedures should be included, whether or not to compute the Signal-to-Noise and Interference Ratio (SINR) for the UEs).

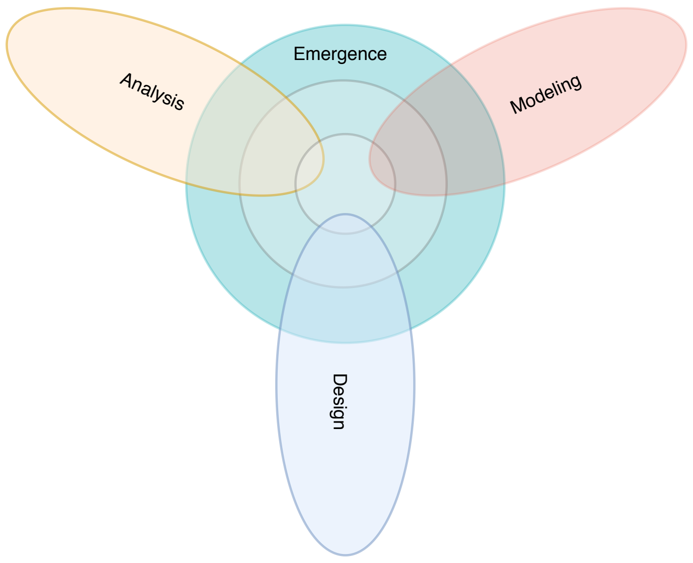


Figure 5.1: Thesis roadmap: In this Chapter we focus on the *Analysis*, *Modeling* and *Design* layers to study *Emergence*.

This allows us to study large networks - networks with a large number of BSs and a large number of UEs. Then, we propose a 4G/5G compliant Network Level Mobility Management Optimization approach based on UE Mobility to minimize signaling (i.e. handover signaling, paging and tracking area updates) and handover latency by dynamically re-configuring the Radio Access Network (RAN)-to-core association (i.e. creating and dynamically re-configuring handover regions). In this Chapter we also rely on the functional topology framework proposed in Chapter 3. We use the functional topology framework to model the relationships between BSs, which allows us to model the problem of the handover region creation as a graph partitioning problem. We focus on both 4G and 5G for two reasons: the RAN-to-core association is similar in both architectures, and the transition from 4G to 5G will be evolutionary rather than revolutionary meaning that both technologies will coexist for a long time [154, 155].

The main contributions of this Chapter are:

- We developed a completely modular simulator that allows us to study dynamics in large cellular networks;
- We demonstrate a distributed and centralized 4G/5G compliant approach to minimize signaling and latency related to user mobility in cellular networks;
- We show that by taking advantage of the principles of emergence, we can design distributed algorithms with very low computational complexity ($O(1)$), which are highly scalable and adaptive.

5.2 Agent-Based Network Mobility Simulator

As highlighted in [156], simulators are generally categorized as:

- Link Level Simulator: The main focus is on the Physical Layer functionalities for a single link between two wireless transceivers (e.g. MATLAB or OCTAVE Link Layer simulations to obtaining BLock Error Rate (BLER) statistics).
- System Level Simulator: The main focus is on the air interface for studying the operation of multiple network nodes (e.g. VIENNA simulator for assessment of coverage, spectral efficiency, throughput by taking into account the interference generated by neighboring nodes).
- Network Level Simulator: The focus is on investigating specific protocols and their interactions with upper and lower layers. e.g. the *ns-3* simulator that treats BSs as network entities capable of exchanging messages between each other.

Simulating self-organization in a cellular network is a task that involves the investigation of individual nodes, the communication between those nodes and the relationship between the nodes and the network environment. *ns-3* would be the obvious choice for simulating these network scenarios. However, the huge amount of memory and *CPU* power needed to simulate a relatively small number of network nodes is a limiting factor [156]. The right level of network layer/functionality abstraction can sometimes help to overcome the limitations related to memory and processing power needs. However, choosing the right level of abstraction is not a straightforward task. On one hand, a low level of abstraction requires a lot of resources. On the other hand, a high level of abstraction results in a non-realistic network setup.

Although system level simulations are crucial for the performance evaluation of new mobile network technologies [157], the lack of upper layer protocol implementations is a limiting factor. The VIENNA simulator is a good example of a powerful system level simulator. It is designed to assess both Homogeneous and Heterogeneous Long Term Evolution (LTE) network deployments. The VIENNA simulator also implements the physical layer with a low level of abstraction, which provides a powerful means of testing techniques that require detailed physical layer implementation, e.g. Multiple-Input Multiple-Output (MIMO). However, the lack of core network functionality support, upper layer protocol implementations and the challenges associated to simulating realistic user mobility datasets with handover functionalities, make it challenging to simulate self-organizing capabilities of network functions (e.g. mobility management, load balancing, throughput optimization).

We developed a network simulator for the purpose of studying self-organization in cellular networks. The main goal was to build a simulator that is modular and therefore extendable, which allows us to abstract the functionalities on each layer of the protocol stack. This is achieved by combining the *singleton design pattern*[1], which ensures a single instance

[1]The singleton design pattern is a software design pattern that restricts the instantiation of a class to

of the simulator, the *factory design pattern*[2], which allows us to use different levels of abstraction and the *event-driven architecture*[3], which introduces the notion of time to the simulation entities. These three software design patterns combined with the Agent-Based Modeling (ABM) principles, i.e. rich concept of time and dynamic understanding of the network behavior that goes beyond a static snapshot of the network, allow us to perform realistic network simulations.

Due to its capability to simulate protocols and their interactions on upper and lower layers, the Agent-based NeTwork mobility Simulator (ANTS) allows us to study the network behavior on a more granular level. It allows us to simulate the user mobility in a way that considers every user to be completely independent from each other, while still following the rules of behavior that are enforced by the environment (e.g. strictly moving along the streets). It also allows us to simulate self-organization through the implementation of the distributed algorithm on every network node. Unlike the above-mentioned simulators, the ANTS simulator allows us to take advantage of the abstraction of the functionalities that are not being investigated, which reduces the amount of computational resources needed to simulate a large network. For example, the *ns-3* simulator, due to its memory and *CPU* requirements was not able to simulate a network of 8424 BSs.

5.2.1 Simulator Architecture

Figure 5.2 depicts the building blocks of our ANTS. As per the definition of ABM, phenomena in our simulator are modeled in terms of agents and their interactions. Each block is implemented as an agent - an autonomous computational object with particular properties and actions. The agents interact with each other and they also interact with the environment. The interactions between agents are modeled through the exchange of protocol messages between the nodes, or the nonverbal communication between them - a node can infer the status of its neighbors without explicitly exchanging protocol messages with them. The interactions between the agents and the environment is modeled through a set of parameters that depends on the specific type of agent (e.g. UEs are aware of their location and they won't move outside of the simulated area, BSs model the environment by implementing different path loss and interference models). The three main building blocks are:

- *Core Node Agent*: This agent, as the name suggests, allows us to simulate the functionalities of the nodes in the core network. Due to the high level of flexibility, the simulator allows us to choose to implement/use only the nodes in the core network that are of interest to our simulations. This reduces the amount of memory and com-

one single instance.

[2]The factory design pattern is a creational software design pattern that uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created.

[3]The event-driven architecture is a software architecture pattern promoting the production, detection, consumption of, and reaction to events.

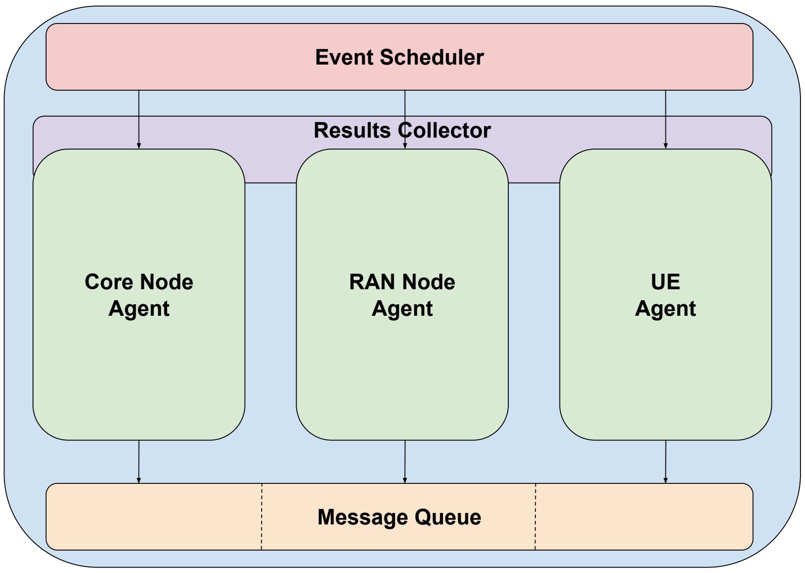


Figure 5.2: The block diagram of the ANTS Simulator; the three main building blocks (*Core Node Agent*, *RAN Node Agent*, *UE Agent*) are autonomous computational individuals (i.e. Agents) that interact with each other and the network environment.

putational resources needed to run simulations that do not require the implementation of the whole protocol stack and all network functionalities.

- *RAN Node Agent*: This agent simulates the functionalities of the nodes in the RAN. Similarly to the *Core Node Agent*, we can choose the level of abstraction needed for our simulations through the implementation of protocols and signaling procedures on the Air interface and the interface between the BS and the core network.
- *UE Agent*: This agent simulates the functionalities and mobility of the UEs. Again, the simulator allows us to implement only the needed functionalities and abstract the signaling and protocol messages that are not related to our simulations (e.g. running simulations to study coverage does not require the implementation of the authentication protocol procedures in the network).

In addition to these three building blocks of our simulator, we introduce three helper modules:

- *Event Scheduler*: This module allows us to introduce the notion of time to our simulator. ANTS is a discrete time network level simulator, and the *Event Scheduler* sends periodic signals to all the agents indicating the end of the current simulation time step and the beginning of the next one. This allows us to synchronize all the agents in the simulator.
- *Message Queue*: This module handles all the protocol messages in the simulator. It serves as a shared queue that is accessible by all the agents in the simulator. Multiple queues can be instantiated to serve as dedicated *Message Queues* between specified

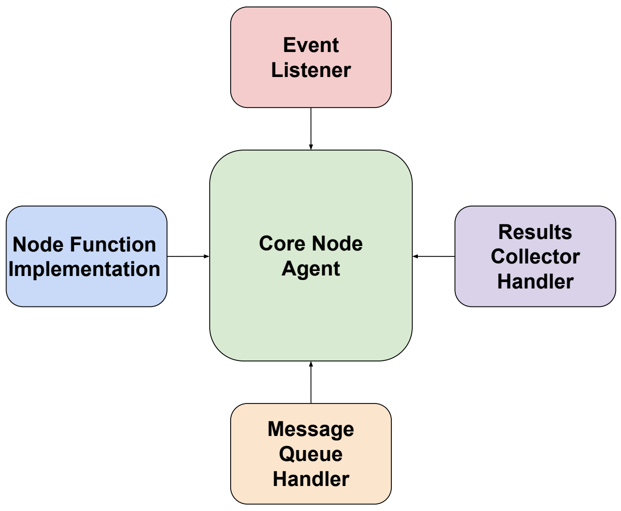


Figure 5.3: The block diagram of the *Core Node Agent*; the modular approach (block structure) allows us to easily extend the node functionalities and rules of interactions with the network environment and other agents.

types of agents. This allows us to separate for example the protocol messages between the RAN and the core from the messages on the Air interface (e.g. a separate queue for the exchange of messages between the BS and the *Core Node Agents*, which is not accessible by the *UE Agents*). Whenever a message is sent from one agent to another, the source agent has to place this message on the *Message Queue*. The target agent collects the message from the *Message Queue* and continues with the processing of the message according to the rules defined in the simulation.

- *Results Collector*: This module allows us to collect the relevant information from all agents in the network. e.g. collecting the state of each agent at any time during the simulation, collecting the number of messages in the *Message Queues*, collecting the position of each moving agent.

5.2.2 Core Node Agent

Figure 5.3 depicts the building blocks of the *Core Node Agent*. All these blocks have to be implemented in order to have a functioning *Core Node Agent* that can operate within the simulation environment and communicate with the other agents. As shown in Figure 5.3, these blocks are:

- *Event Listener*: This module is directly connected to the *Event Scheduler* and it is responsible for the event detection and apprehension. Once an event is detected the *Event Listener* will call the appropriate *Node Function* to start the execution of the suitable node procedures.

- *Message Queue Handler*: This module serves as the interface between the *Core Node Agent* and the *Message Queue.* It allows us to implement different scheduling algorithms to deal with the incoming messages and to simulate the communication bandwidth between different nodes.
- *Results Collector Handler*: This module is the interface between the *Core Node Agent* and the *Result Collector.* It exposes the agent state and different internal agent parameters to the *Result Collector*, allowing us to collect them and store them for further analysis.
- *Node Function Implementation*: The functions used to implement this module are triggered by the *Event Listener.* Those functions implement the procedures that are executed during the operation of a specific node (e.g. mobility management, routing, security).

Unlike the implementation of the *Event Listener*, which assumes one instance per agent, the *Message Queue Handler*, *Results Collector Handler* and *Node Function Implementation* can have multiple instances and implementations. For example, a *Core Node Agent* could be accessing and placing messages on multiple *Message Queues* (e.g. one for the communication with other *Core Node Agents* and one for the communication with *RAN Node Agents*, which requires the implementation of two instances of Message Queue Handlers - one for each *Message Queue*).

5.2.3 RAN Node Agent

As shown in Figure 5.4, in addition to the *Event Listener*, *Message Queue Handler*, *Results Collector Handler* and *Node Function Implementation* blocks, which serve the same purpose as previously explained in Section 5.2.2, the main building blocks of the *RAN Node Agent* are:

- *Mobility Handler*: This module allows us to define the location and mobility properties of the agent. For static agents this module does not have to be implemented. By including this module in the *RAN Node Agent*, we enable the simulation of moving networks, i.e. networks in which the infrastructure is mobile (e.g. Unmanned Aerial Vehicle (UAV) or Unmanned Ground Vehicle (UGV) BSs).
- *Physical Layer Module*: As shown in Figure 5.4, this module is part of the *Node Function Implementation.* It allows us to configure different channel conditions and physical layer implementations (e.g. different path loss models, MIMO).
- *Medium Access Control (MAC) Layer Module*: Similarly to the previous module, this module is part of the *Node Function Implementation.* It allows us to configure different MAC layer properties like: different access technologies, different retransmission procedures, priority handling, etc.

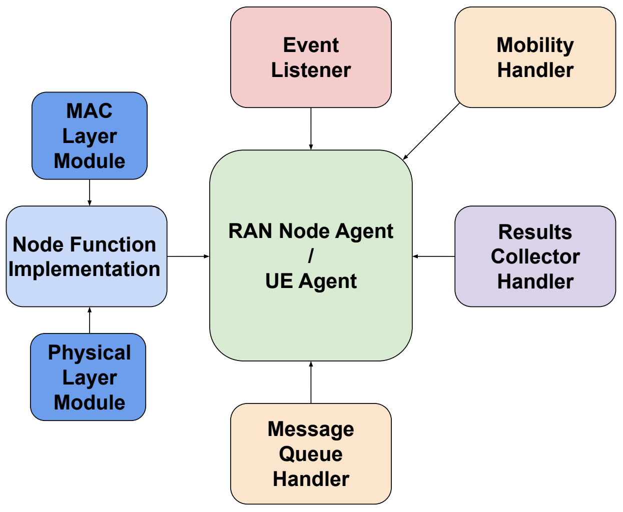


Figure 5.4: The block diagram of the *RAN Node Agent*/*UE Agent*; the modular approach (block structure) allows us to easily extend the node functionalities and rules of interactions with the network environment and other agents.

5.2.4 UE Agent

As shown in Figure 5.4, the *UE Agent* consists of the same building blocks as the *RAN Node Agent*. Even though the building blocks are the same, the implementation is obviously different. For example, besides the obvious difference in the *Node Function Implementation* - different functionalities of the UE and any RAN node, the implementation of the *Physical Layer Module* and the *MAC Layer Module* is also different (e.g. up-link and down-link communication). However, these two modules have to be matched with the implementation on the RAN side to allow the agents to communicate with each other.

5.2.5 Simulation Configuration

Figure 5.5 depicts the flow chart of a simulation instance within the ANTS simulator. As shown in Figure 5.5, the execution process of a simulation consists of two phases: (1) the initialization phase; and (2) the execution phase.

The initialization phase starts with the initialization of the *Event Scheduler*, the *Message Queues* and the *Results Collector*. Then, we initialize all the agents, which includes: the initialization of the agent specific parameters (e.g. location, mobility behavior, functionality implementation), the initialization of the *Event Listener* - connects to the *Event Scheduler*, the initialization of the *Message Queue Handlers* - connects to the appropriate *Message Queues* and the initialization of the *Results Collector Handler* - connects to the *Results Collector*.

As shown in Figure 5.5, the execution phase includes the following procedures:

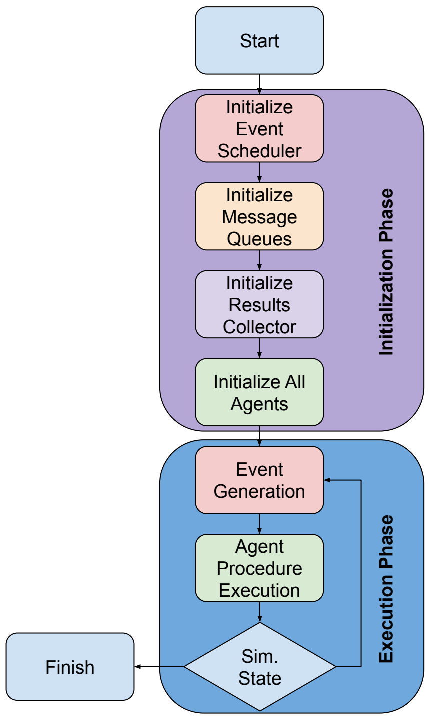

Start
Initialize Event Scheduler
Initialize Message Queues
Initialize Results Collector
Initialize All Agents
Initialization Phase
Event Generation
Agent Procedure Execution
Sim. State
Execution Phase
Finish

Figure 5.5: The flow chart of a simulation instance within the ANTS Simulator; the execution process consists of two phases: (1) the initialization phase; and (2) the execution phase.

- *Event Generation*: The *Event Scheduler* sends a signal to all the agents indicating the end of one and the beginning of the next simulation time step.
- *Agent Procedure Execution*: All the agents execute their *Node Function Implementation* procedures which includes: accessing/placing information from/to the *Message Queue* through the *Message Queue Handler*, processing information, changing the agent state parameters, sharing the agent state and internal agent parameters with the *Results Collector* through the *Results Collector Handler*.
- *Checking the Simulation State*: The simulator checks if the simulation time has finished or if any of the agents has aborted the simulation. If none of the conditions is satisfied, a new signal is being generated by the *Event Scheduler* and the simulation continues. Otherwise, the simulation is finished and the results are being saved.

5.3 Optimizing the Mobility Management Function based on Users' Mobility Patterns

Services like Augmented Reality, Virtual Reality, autonomous driving and smart buildings/cities are steering the development of the next generation of communication networks. These services are an important part of the Fifth Generation (5G) concept, imposing strict requirements in terms of latency, throughput, and mobility. Mobility plays a central role in this context, because it affects both latency and throughput. For example, handover procedures are inevitable to ensure mobility, but at the same time they greatly increase latency. To manage mobility, RAN nodes in both 4G and 5G are grouped into a hierarchy of geographical areas. Not all handovers come with the same cost in terms of latency and signaling to the core network. In fact, the association between nodes in the RAN and nodes (e.g. Mobility Management Entity (MME) in the case of 4G), functions (e.g. Access and Mobility Function (AMF) in the case of 5G), and Location Regions (e.g. Tracking Area (TA), Registration Area, TA List) in the core network plays a major role in determining the amount of signaling between RAN and core and it is the focus of this Chapter. For example, a handover with MME reallocation requires on average 50% more signaling messages compared to other types of handover [158].

There has been changes in the approach related to network control. A big driver in this field is the Software Defined Networking (SDN) community. The proposed approach involves decoupling the data and control plane and mainly relies on real-time and agile centralized control. However, the centralized control plane has been identified as one of the main disadvantages of the proposed technology, due to the single point of failure, single point of attack, and scalability issues [159, 160]. This led to solutions that rely on distributed SDN controllers [160]. The distribution of these functionalities requires the development of communication procedures for information sharing and load balancing between the controllers. Additionally, the closed loop solutions based on the centralized control mechanisms rely on low-latency and reliable backhaul connectivity. However, certain use cases involving sensor networks, remote areas, ultra low latency control applications and applications that involve moving architecture (e.g. UAV, UGV) for applications like: search and rescue, monitoring and diagnostics of industrial machines, power plant surveillance can not guarantee the necessary reliability and connectivity. Therefore, the development of distributed network functions is necessary for both, the development of scalable and more robust SDN solutions and use cases that can not rely on the centralized control plane.

We demonstrate a 4G/5G compliant Network Level Mobility Management Optimization solution based on UE Mobility to minimize signaling (i.e. handover signaling, paging and tracking area updates) and handover latency by dynamically re-configuring the RAN-to-core association. Our approach can be implemented in a distributed or centralized manner. The distributed node re-configuration relies on existing protocol messages and network management systems [161–163]. The centralized mechanism also relies on the same information, but due to the lack of requirements for these messages to be forwarded to the Operations Support System (OSS), it has to be implemented as a passive probe in the core network. The computational complexity of the distributed solution is $O(1)$ as compared to centralized one that is an NP-hard problem.

The proposed approach also balances the load between the MME/AMF instances and is independent of the specific handover mechanisms implemented in the network (e.g. whether the handover is network-controlled or mobile-valuated, or one uses cell range expansion techniques, etc.). Moreover, it successfully adjusts to both, the up- and down-scaling of resources in the core network (e.g. new AMF instance or shutting down an AMF instance) and the changes in the RAN (e.g. changes in UE mobility patterns) by dynamically reconfiguring the RAN-to-core nodes association.

In the literature, a number of approaches have been proposed for autonomic network optimization. In [164] a mechanism to minimize the number of Serving Gateway (SGW) relocations is proposed. While MME relocation is also mentioned, the authors focus on the SGW. They propose the introduction of a Service Area for idle users, which is a subset of the Service Area of each SGW. Although the results show a decrease of the number of SGW relocations for users in active mode, the mechanism to determine the idle-mode service areas is not provided and the implications in terms of existing standards are not discussed. In [165] a centralized heuristic mechanism to deploy network functions so as to minimize the number of SGW relocations is proposed. The minimization of MME relocations is only mentioned as a possible application. However, the proposed solution relies on a greedy algorithm with high computational complexity (i.e. $O(N^3)$) that does not assure a significant improvement in terms of SGW relocations. Moreover, the authors do not discuss the improvements compared to the existing industry standard techniques for network planning and they do not provide details about the integration of their solution with existing or future networks. In contrast, our solution relies on graph partitioning and it exploits standardized protocols and mechanisms, which will facilitate its adoption.

Another approach in literature for inter-region handover optimization relies on a new architecture, proposed in [166]. The authors propose a recursive hierarchical algorithm to minimize the number of inter-region handovers in [167]. In contrast, the solution we propose is designed to work within existing (4G) and emerging (5G) mobile networks so as to maximize technology adoption. Specific solutions for handover latency reduction have been proposed in the case of femto cells [168, 169]. These two patents detail an optimized intra-HeNB GW (Home eNodeB Gateway) handover mechanism that reduces signaling to and from an MME. While our approach also aims at reducing the signaling to and from the MME, our network-wide approach is not restricted to the specific femto cell case and we take into account users' mobility when optimizing the nodes configuration. In [170, 171], the inventors detail a mechanism to re-assign BSs to MMEs so as to balance the load of the MMEs. Our solution, while also considering the load, re-configures the nodes based on the user movements.

Our solution is a general approach to the node to x-area association, where x can be a handover region, a TA or a set of TAs. Therefore, closely related to the question addressed by our solution is the autonomous configuration of TAs and TA Lists. Several solutions aiming at minimizing the signaling overhead caused by the periodic TA updates and paging have been developed in the research literature [172] or patented [173, 174]. The approaches in [172–174] are centralized and require information that does not exist in the network (e.g. the UE average speed, the average number of UE per cell). Additionally, the authors do not provide implementation details in terms of data collection, and reconfiguration of

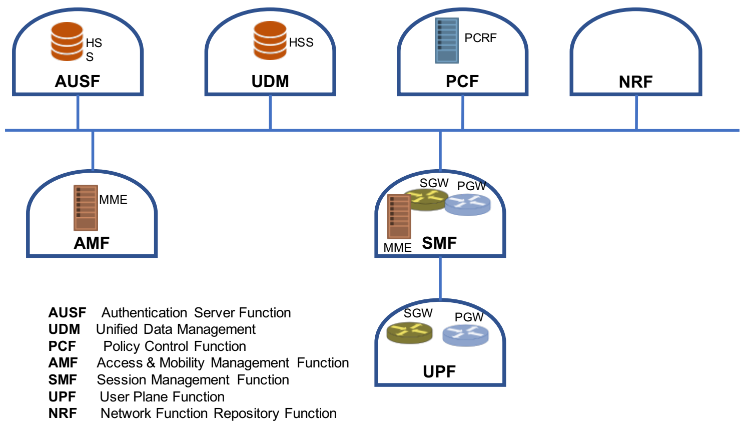

HSS
AUSF
HSS
UDM
PCRF
PCF
NRF
MME
AMF
SGW
PGW
MME
SMF
SGW
PGW
UPF
AUSF Authentication Server Function
UDM Unified Data Management
PCF Policy Control Function
AMF Access & Mobility Management Function
SMF Session Management Function
UPF User Plane Function
NRF Network Function Repository Function

Figure 5.6: Mapping the 4G Evolved Packet Core to the 5G Service Based Architecture.

the nodes. In contrast, our approach relies only on information available in the network allowing an easy integration with the existing 4G and future 5G systems.

5.3.1 Problem Description

The MME and the SGW are the two main signaling entities in the Evolved Packet Core (EPC) architecture [175]. The MME serves as the main control entity that handles all signaling events related to the core network (e.g. mobility management, paging, bearer setup, subscriber information). The SGW is the local mobility anchor point, and it is in charge for the forwarding and routing of user-data packets between the BSs and the Packet Gateway (PGW). Network entities like the MME and SGW, that are implemented as hardware nodes in the EPC, are softwarized and decomposed into functional entities, which are accessible as services in the new 5G Service Based Architecture (SBA), which is shown in Figure 5.6. The MME is mapped into the AMF and Session Management Function (SMF), and the SGW into the SMF and User Plane Function (UPF). Since we focus on the network mobility management, our node of interest in the EPC is the MME and in the SBA the function of interest is the AMF.

From the point of view of RAN-to-core nodes association three types of handovers exist in LTE:

- Intra-MME/SGW: this type occurs when a UE moves between BSs that belong to the same MME/SGW pool, and we refer to it as intra-region handover. An MME Pool Area and an SGW service area are defined as areas within which a UE may be served without the need to change the serving MME and SGW respectively [158].
- Inter-MME/SGW: this type occurs when a UE moves between BSs that belong to different MME pools or SGW service areas, and we refer to it as inter-region handover.
- Inter-RAT: this type occurs when a UE moves between two radio technologies (e.g. from LTE to WCDMA).

Based on [152] the 5G network architecture includes similar concepts. The main difference is that the functionalities are virtualized, and the equivalent to the MME Pool Area is the AMF Region. An AMF region consists of one or multiple AMF Sets. An AMF Set consists of AMFs that serve a given geographical area [152]. Hence, the inter-region handovers are the handovers with AMF re-selection and intra-region handovers are the handovers without AMF re-selection.

The hierarchy of node groupings in the network is a crucial element of the handover region optimization. A TA is a logical collection of BSs used to trace the geographical location of a UE in the idle state [158]. According to [152], the 5G architecture will rely on the same hierarchy. An MME/AMF always manages whole TAs and TAs can also be managed by multiple MMEs/AMFs. Traditionally, a BS gets assigned to a TA based on its geographical location. Due to the static deployment of the BSs, the TA assignment is static as well. The idea behind our algorithm is to *dynamically rearrange* the handover regions, based on the *UE moving patterns*, in order to *minimize the number of inter-region handovers*. As a side effect the TAs are being rearranged as well.

The TA optimization problem targets the minimization of the signaling related to location updates and paging procedures. Optimizing the TAs only with respect to location updates would result in creating one big TA. This means that the UE, once connected to the network, does not have to inform the network of its movements, because it is never moving between multiple TAs. However, this situation results in an exponential increase of signaling messages related to paging. Whenever the network has to reach out to the UE it has to ask all the BSs in the network about the current location of the UE. On the other hand, focusing only on paging and optimizing the TAs in order to minimize the amount of signaling related to paging would result in one TA per BS. This results in a situation in which the network is always aware of the exact location of the UE, and therefore it can always reach out to it. However, this implies an exponential increase of signaling messages related to location updates, because every change of connection includes a location update. Therefore, the optimization of TAs involves taking care of the trade-off related to the amount of location updates and paging messages.

The reduction of the overall number of inter-region handovers directly affects the overall amount of signaling between the RAN and the core network, which consequently affects the signaling delay related to handover operations. Inter-region handovers require on average 50% more signaling messages compared to the intra-region ones [158]. That results in higher latency of the inter-region handover procedures compared to the intra-region handovers [176]. We take this into consideration while minimizing the latency and the amount of signaling in the core network by introducing a smart design of the handover regions, based on the UE mobility pattern information.

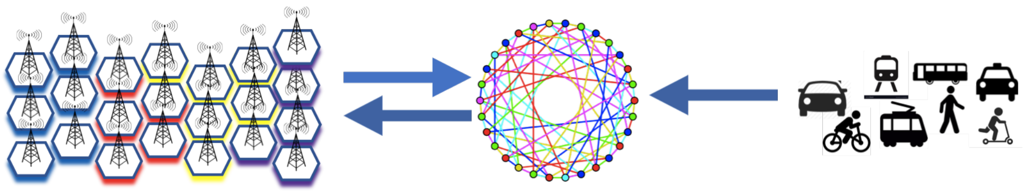


Figure 5.7: The association of BSs to different MME/AMF (represented with different colors) is dynamic and depends on UE mobility patterns.

5.3.2 Solution

It is possible to minimize the number of inter-region handovers by dynamically re-configuring the association between BSs and nodes in the core, i.e. by grouping the BSs that exchange more users into the same handover region (see Figure 5.7). In this Subsection we present two alternatives to the handover region optimization problem. We first start with a centralized optimal formulation. Due to the computational complexity and signaling overhead of the Centralized Graph Partitioning (CGP) approach, we also propose a distributed approach for the optimization. Finally, we show that the proposed distributed solution can be implemented by leveraging information already collected and exchanged within the network. While the centralized mechanism also relies on the same information, current standards do not require that information to be forwarded to the OSS. This means that the centralized mechanism could realistically be implemented as a passive probe in the core network.

Centralized Graph Partitioning Approach

We adopt a complex systems approach to model the problem. The handover occurrences are mapped into a functional topology, a graph in which the nodes represent the BSs and the links between the nodes indicate that a handover occurred between the connected nodes, as described in Chapter 3. This allows us to formulate the RAN-to-core nodes association re-configuration as a k-way Graph Partitioning Problem (GPP) on an undirected weighted graph $G = (V, E, w)$ that represents BSs as the set of vertices V and the occurrence of handovers between BSs as the set of edges $E \subseteq V \times V$. The weight $w_{i,j}$ is the normalized number of handovers that occurred between nodes i and j:

$$w_{i,j} = \frac{h_{i,j} - h_{min}}{h_{max} - h_{min}}, \tag{5.1}$$

where $h_{i,j}$ is the number of handovers that occurred between nodes i and j, h_{max} is the maximum number of handovers that occurred between any two nodes in the graph, and h_{min} is the minimum number of handovers that occurred between any two nodes in the graph.

For each edge $\{i, j\} \in E$ let us introduce a binary variable $z_{i,j}$, which is equal to 1 iff i and j belong to two handover regions, i.e. to two MMEs/AMFs. We also define a binary variable $x_{i,k}$ for each vertex $i \in V$ and each region $k \in K$, which is the set of all available

MMEs/AMFs. $x_{i,k}$ is equal to 1 iff node i belongs to handover region k. We denote the traffic load of each vertex $i \in V$ with l_i and the load threshold of region $k \in K$ with $L_{k_{th}}$. The RAN-to-core nodes association can be formulated as an integer linear programming problem:

$$\begin{aligned}
\text{minimize} \sum_{\{i,j\} \in E} & w_{i,j} z_{i,j} & & \\
\text{subject to} \sum_{k \in K} & x_{i,k} = 1, & \forall i \in V & \\
\sum_{i \in V} & l_i x_{i,k} \leq L_{k_{th}}, & \forall k \in K & \qquad (5.2) \\
& x_{i,k} - x_{j,k} \leq z_{i,j}, & \forall \{i,j\} \in E, \forall k \in K & \\
& x_{j,k} - x_{i,k} \leq z_{i,j}, & \forall \{i,j\} \in E, \forall k \in K & \\
& z_{i,j} \in \{0,1\} & \forall \{i,j\} \in E & \\
& x_{i,k} \in \{0,1\} & \forall i \in V, \forall k \in K &
\end{aligned}$$

The objective function in equation (5.2) is minimized when the number of inter-region handovers is minimized. The first constraint ensures that each node belongs to only one handover region. The second constraint guarantees that the sum of the loads of all BSs in one handover region does not exceed the load threshold of that region. The third and fourth constraint ensure that if two nodes i and j belong to two handover regions the link connecting them is considered to be an inter region link, i.e. $z_{i,j} = 1$. The same constraints guarantee that if nodes i and j belong to the same handover region, then $z_{i,j} = 0$.

Due to the NP-hard nature of the k-way GPP, we rely on the METIS library for graph partitioning. METIS is based on the work in [177], and implements various multilevel algorithms.

Even though the heuristics implemented in METIS speed up the partitioning of the graph, computational complexity is not the only drawback of the centralized handover region optimization problem. In fact, the centralized approach to the handover region optimization problem assumes that the information of all the handovers is collected at one node in the core network, thus involving a huge amount of signaling. Indeed the signaling overhead of the whole network would increase, and at the same time the re-configuration of the network would be delayed due to the multiple stages involved in the optimization (handover data collection at the BSs, transferring the data to a centralized node in the core network, running the optimization, and transferring signaling messages in order to perform the re-configuration of the network).

Distributed Self-Organization Approach

We now present a Distributed Self-Organized (DSO) solution to the problem of handover region optimization. The proposed approach consists of two main components, one running

Algorithm 2 RAN node optimization procedure

```
counters ← #handovers, #handovers from each region
while Cell is operational do
    Wait for event {handover, reassign request}
    if event == handover then
        Update counters
        Calculate energy of attraction
        MakeAssignmentDecision()
    if event == reassign request then
        MakeReassignment(MME/AMF_id)

function MakeAssignmentDecision()
    M ← List of available MMEs/AMFs
    k ← number of MMEs/AMFs that are managing the cell
    A ← List of energies of attraction towards MMEs/AMFs
    if max(A) - current(A)> ping pong threshold then
        Send request to max(A) MME/AMF
        Reset counters

function MakeReassignment(MME/AMF_id)
    A ← List of energies of attraction towards MMEs/AMFs
    Exclude the MME/AMF with MME/AMF_id from A
    Sort A in descending order
    while MME/AMF not assigned do
        Try to get assigned to an MME/AMF from A
        Reset counters
```

on the RAN nodes and the other on virtual instances of core nodes (e.g. MMEs/AMFs).

The component that runs on the RAN nodes can be formalized in Algorithm 2. It relies on handover counters already available at the BSs (e.g. number of handovers, source handover MME). The initial assignment of a BS can be random or chosen based on the majority of its neighbors. The optimization process is triggered whenever a handover occurs or in case an MME/AMF sends a reassignment request. In case of a handover, the BS updates its counters and the algorithm calculates the energy of attraction towards all available MMEs/AMFs. The **energy of attraction** of node n towards the m-th MME/AMF is calculated as:

$$A_m(n) = \frac{H_n(m)}{\sum\limits_{i \in M} H_n(i)} \tag{5.3}$$

where $H_n(m)$ is the number of handover requests that arrived at node n from nodes that are assigned to the m-th MME/AMF. Therefore, the energy of attraction towards an MME/AMF is the ratio between the number of handover requests originating from this MME/AMF and the total number of handover requests that arrived on the observed node.

Once the counters are updated and the energy of attraction is calculated, the BS decides whether to change its MMEs/AMFs assignment based on the energy of attraction. It should be noted that the DSO enables overlapping handover regions, i.e. a BS can be connected to multiple MMEs/AMFs. The *ping pong threshold* restricts the BS from constantly changing its assigned MMEs/AMFs. Once a BS changes its assignment, the counters are reset to default values and the adaptation process restarts. The reset to the default values can be a hard reset, which resets the values to zero, or a soft reset, that sets the values of the counters by using a moving average. In case an MME/AMF requests a reassignment of the BS to another MME/AMF, the BS attempts to get assigned to the next best (based on the energy of attraction) available MME/AMF.

Algorithm 3 MME/AMF optimization procedure

```
$N \leftarrow$ List of cells assigned to the MME/AMF
$L \leftarrow$ Current MME/AMF load
$L_{max} \leftarrow$ Load limit
$A \leftarrow$ List of energies of attraction of the cells
Wait for assignment request from $n$
if $L + L(n) < L_{max}$ then
    Assign cell to the current MME/AMF
else
    if $A(n) > min(A) + \delta$ then
        if $L - L_{min(A)} + L(n) < L_{max}$ then
            Assign cell to the current MME/AMF
            Inform the cell with min(A) to get reassigned
            Remove the cell with min(A) from N and A
    else
        Reject the request
```

The second component runs on a virtual instance of the MME/AMF and is formalized in Algorithm 3. The MME/AMF waits for a request from a node that wants to get assigned to it. If the combined load of the MME/AMF and the load coming from the node requesting the assignment is lower than a threshold (L_{max}), the request is accepted. In case the total load is greater than the threshold, the assignment can be accepted or rejected depending on the energy of attraction of the cell that is requesting the assignment. In particular, the assignment request is accepted only if among the cells currently attached to the MME/AMF there exists one with lower energy of attraction which, if removed, would free up enough resources to manage the requesting cell. If multiple such cells exist, the one with the lowest energy of attraction is selected, informed to change its assignment, and removed from the list of cells assigned to the current MME/AMF.

Integration with Network Protocols

From an implementation point of view it is important to highlight that the approaches described in the previous section have been designed to rely on information that is available

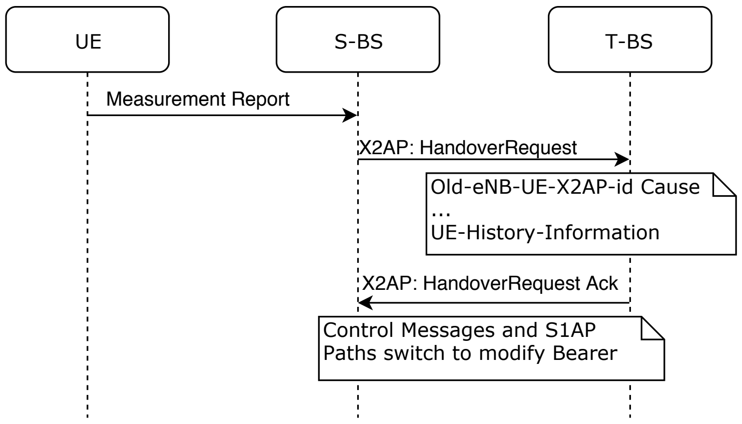


Figure 5.8: Simplified sequence diagram of the intra-region handover procedure, with the emphasis on the *HandoverRequest* message which carries the *UE History Information*. S-BS and T-BS stand for Source and Target BS.

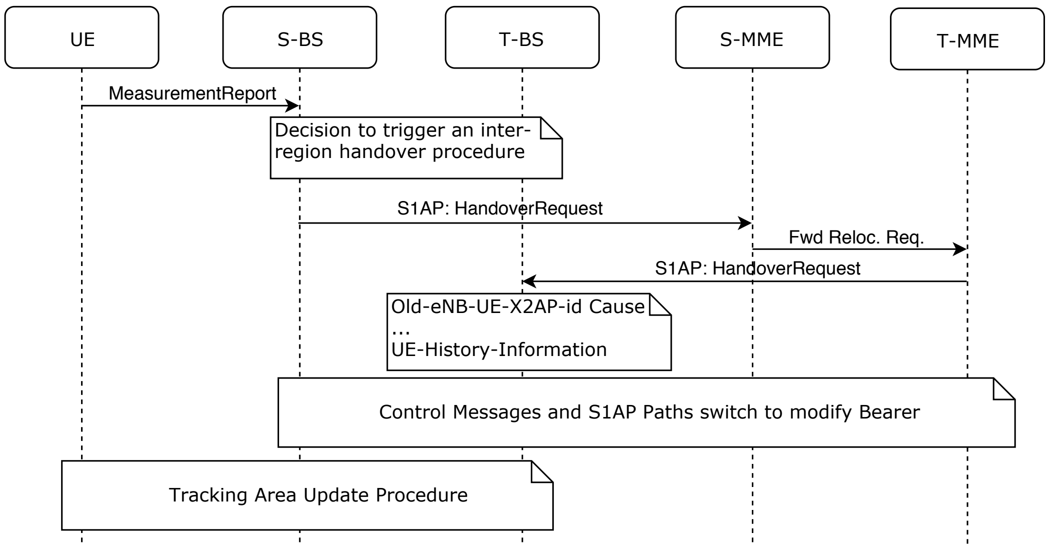


Figure 5.9: Simplified sequence diagram of the inter-region handover procedure. The *HandoverRequest* message carries the *UE History Information*, and the TAU procedure is triggered after the handover.

through already existing signaling messages, removing the need for additional signaling and at the same time simplifying the integration with the existing architecture. The information of interest is the type of handover, the number of handovers and the source MME from which the handover originated. To explain this let us consider the procedures used to perform the two different types of handover (inter and intra-region).

As shown in Figure 5.8, the X2 handover procedure assumes direct communication between the involved nodes. The *HandoverRequest* message contains a field reserved for the *UE History Information*, which contains information about the last visited cells by the UE [161]. This information is used to determine where the handover is coming from. Since the request is sent through the X2 link between two nodes, the handover type is obviously intra-region, i.e. the source MME is equal to the destination MME. This means that all

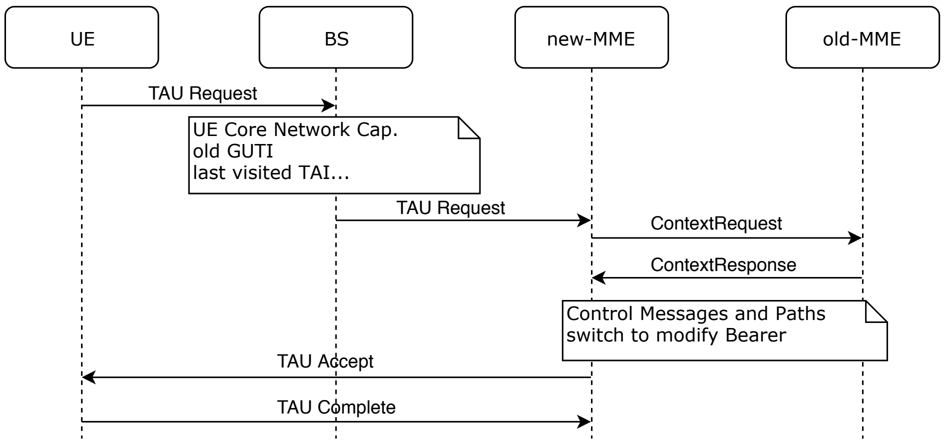


Figure 5.10: Simplified sequence diagram of the TAU procedure. The *TAU Request* message carries the *old GUTI* used to determine the source MME.

counters can be updated appropriately (both the total number of handovers and the number of intra-region handovers are increased by 1).

Figure 5.9 shows the procedure that is used to perform a handover between nodes associated with different MMEs. The initial *HandoverRequest* message contains a field that transfers the information from the source to the target node (*TargeteNB-ToSourceeNB-TransparentContainer*), which contains the *UE History Information* similarly to the previous case [162]. In case of an inter-region handover, we have to determine the source MME as well in order to update all counters needed to calculate the energy of attraction in equation (5.3). Considering that an MME covers whole TAs, the inter-region handover includes a Tracking Area Update (TAU) as well (the last part of the handover procedure shown in Figure 5.9), and in case of an intra-region S1 handover (an S1 handover without MME reallocation) the TAU is not required, which indicates to our algorithm that the source and target handover region are the same. As shown in Figure 5.10, the first message that is sent from the UE to the BS is the *TAU Request*, which contains information like *UE Core Network Capability, old Globally Unique Temporary UE Identity (GUTI), last visited TA*, etc. [158]. The old *GUTI* is the identifier of interest to our algorithm, because it consists of two main components, namely the *Globally Unique MME Identifier (GUMMEI)* and the *MME Temporary Mobile Subscriber Identity (M-TMSI)*. Since the *GUMMEI* uniquely identifies the MME which has allocated the *GUTI*, we have all the information needed to update all counters to calculate the energy of attraction with equation (5.3).

5.3.3 Simulation Model

ABM is a ground-up computational modeling approach whereby a phenomenon is modeled in terms of agents and their interactions. An agent is an autonomous computational individual or object with particular properties and actions [41]. It is suitable for modeling heterogeneous and self-organizing systems. The agents in our model are the BSs and the instances of MMEs/AMFs whose behavior is formalized in Algorithms 2 and 3. By shifting the focus from the holistic model of the whole system to the modeling of individual agents and their interactions, ABM enables to accurately simulate an environment with distributed decision making.

In order to simulate a large network of BSs and UEs in our ANTS simulator, we implemented only the essential functionalities related to mobility management. All agents implement the *Event Listener, Message Queue Handler* and the *Result Collector Handler.* The *Core Node Agent* implements the MME/AMF functionalities related to mobility management. Similarly, the *RAN Node Agent* implements only the BS functionalities related to mobility management - handovers, meaning that other functionalities like power control or load balancing are ignored. The *UE Agent* is used to simulate the user mobility - *Mobility Handler* - and in addition to that it implements only the handover capabilities, ignoring other functionalities like power control or any other application layer functionalities.

For the purpose of our simulations we used the simulator that was introduced in Section 5.2 and we draw on the datasets collected by [178, 179]. The dataset [178] provides the information about BS locations worldwide and [179] provides information about the movements of taxis in the San Francisco Bay Area, USA. Since our algorithm relies on the information related to user movements, we combine these two datasets to simulate the handover occurrences between BSs in San Francisco, Oakland and Berkeley. The dataset includes $8,424$ BSs. In our simulations we followed the 3GPP recommendations [180] for the BS coverage - 500m coverage radius. We analyze the taxi movements for two days in order to capture the impact of the changes in user mobility patters. We assume that the load of each BS is the same - $l_i = 1, \forall i \in V$, and the load threshold of each region is $L_k = \frac{|V|}{K} + 1, \forall k \in K$. Hence, the partitioning defined with equation (5.2) results in an equal number of nodes in each partition.

5.3.4 Performance Evaluation

In this Section, we compare the performance of the DSO and the CGP approach detailed in Section 5.3.2, and the traditional static approach which groups the BSs into regions based on their location.

Figure 5.11 zooms in on the grouping of BSs into handover regions in San Francisco. The BSs are grouped into four regions shown in green, blue, yellow and brown. Each region contains approximately the same number of BSs. The map on the top-left in Figure 5.11 shows the geographical approach to group the BSs into four regions with a balanced number of BSs. The map on the top-right shows the grouping of the BSs resulting from the CGP. Instead of relying on geographical proximity of BSs, the RAN-to-core node association in this case is performed based on the user movement information collected during Day 1. The map on the bottom in Figure 5.11 shows the RAN-to-core node associations at 9PM of the same day resulting from the DSO approach. The DSO approach adapts over time,

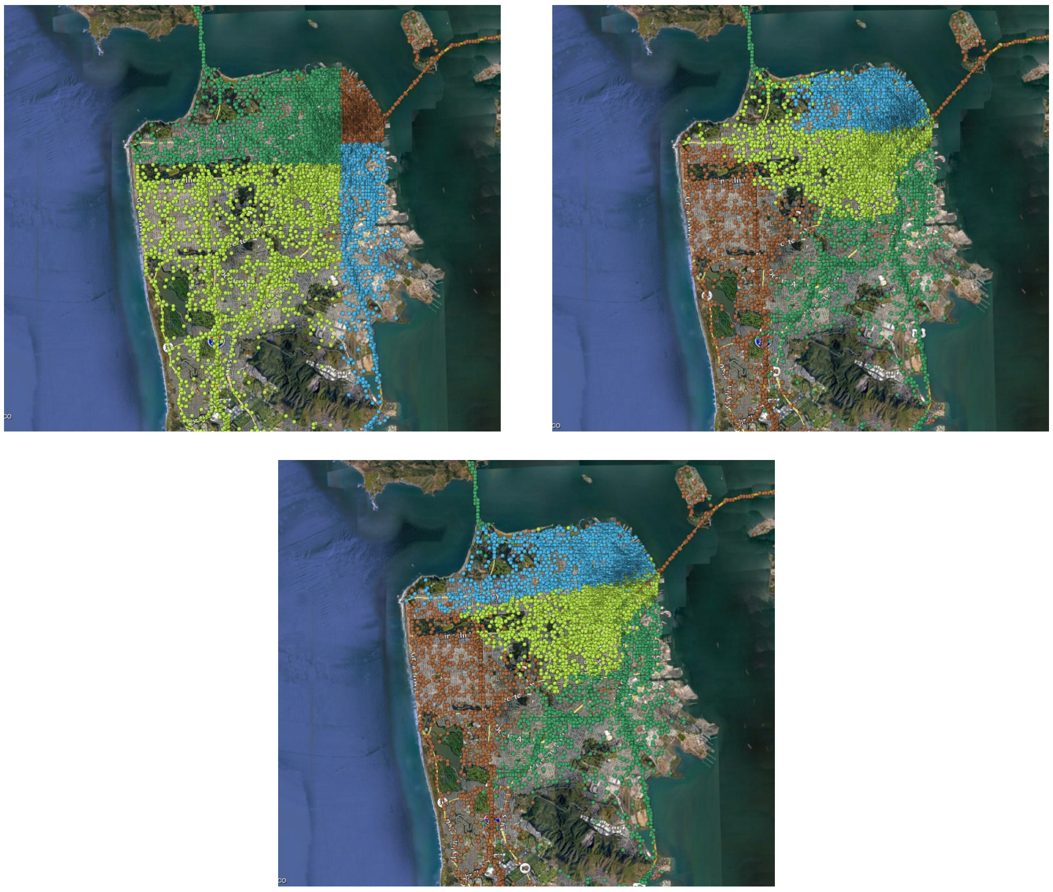


Figure 5.11: Map of San Francisco area showing the BS as colored dots; the grouping of BS into handover regions is represented by different colors. The maps show the grouping based on the geographical approach (top-left), the grouping resulting from the CGP based on data collected during Day 1 (top-right), and the grouping at 9PM of the same day resulting from the DSO (bottom). For clarity of presentation we show only the BS in the city.

Table 5.1: The absolute number of Inter- and Intra-region handovers for Day 1 and Day 2.

	Inter- (Day 1)	Intra- (Day 1)	Inter- (Day 2)	Intra- (Day 2)
Geographical	1720463	8111284	905673	4411991
CGP	1296928	8534819	716496	4601168
DSO	1281876	8549871	690645	4627019

meaning that the BSs constantly monitor their counters and make decisions about their association. The difference between the three maps is conspicuous, and it shows that the spatial proximity of BSs does not necessarily lead to a large exchange of users between them. For example, nearby BSs do not exchange a lot of users if their coverage areas overlap or if due to the terrain it is impossible to move between them. An interesting detail from the associations that resulted from the CGP and DSO optimization approach is that geographically disconnected regions exist. In fact, by looking at the maps on the top-right and on the bottom, we notice that the brown cluster consist of nodes that are geographically located in the West and East of San Francisco.

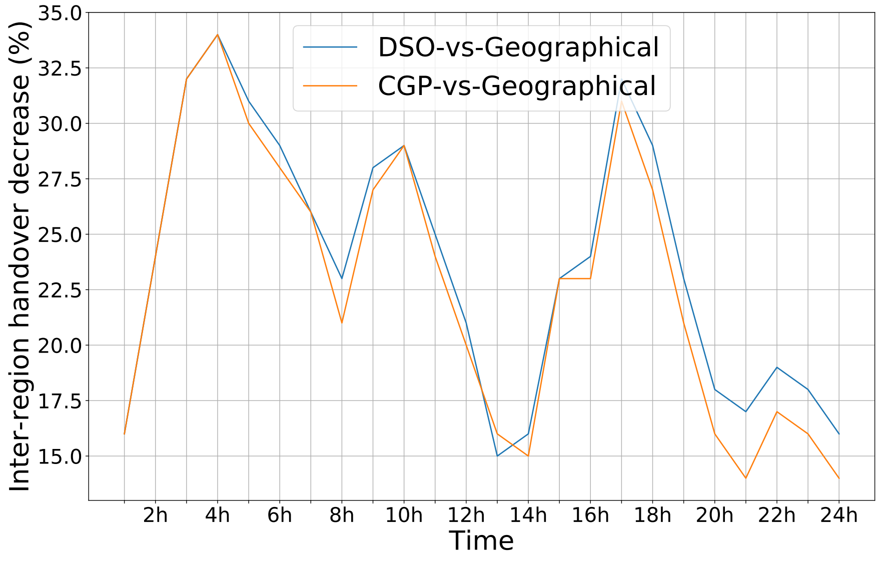


Figure 5.12: The hourly decrease of inter-region handovers during Day 1; the CGP approach relies on data for the whole 24 hours of Day 1.

The map shown on the top-right in Figure 5.11 is the ex post optimal static association (CGP) of nodes for the whole Day 1 under consideration, i.e. the information over the whole Day 1 is gathered and the best possible configuration for that period is computed. This might not be the best association at every point in time during the day since the number of handovers between BSs changes over time, and the weights $w_{i,j}$ in the objective in equation (5.2) are the normalized total number of handovers between BSs over a period of time (in this case Day 1). Since the CGP requires information to be centrally collected, although theoretically possible, due to the large number of handovers occuring in the network (see Table 5.1), it is unlikely that the CGP could be run over a shorter period.

Let us now take a closer look at how the share of inter-region handovers changes over time. Figure 5.12 shows the percentage of inter-region handovers during the Day 1 of the DSO and CGP (ex post) with respect to the geographical clustering. As shown in Figure 5.12 the decrease of inter-region handovers in case of the CGP approach is at least 15% and at some point goes up to 33%. Perhaps more importantly, the DSO approach either performs as well as CGP, or it outperforms it. This is due to the adaptive nature of the DSO approach, which acts based on locally available information making the readjustments agile. This enables a more granular optimization over time, allowing the network to adapt to the changes in the user movement patterns. It should also be noted that the results for the CGP approach in Figure 5.12 correspond to an ex post optimization, i.e. the information over the whole Day 1 is gathered, the best possible configuration for that period is computed and then the performance over the same day is evaluated.

Now, let us examine the overall impact on the number of inter-region handovers in the

Table 5.2: The overall inter-region handover reduction, and the number of Inter- and Intra-region handovers during Day 1.

	Geographical	CGP	DSO
Geographical	0	-24,6 %	-25.5 %
CGP	32.7 %	0	-1.2 %
DSO	34.2 %	1.2 %	0

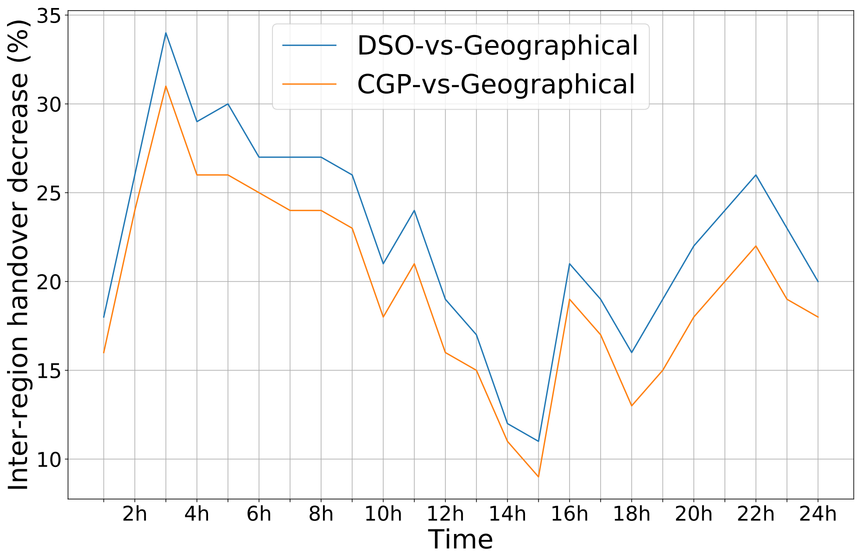


Figure 5.13: Hourly decrease of inter-region handovers during Day 2; the CGP configuration evaluated in this plot results from Day 1 mobility data to emulate the delay resulting from collecting and processing mobility data at the OSS.

network for Day 1. Table 5.2 shows that the overall number of inter-region handovers in the network is reduced by 32.7% for the CGP and 34.2% for the DSO approach compared to the geographical approach. Also, the DSO approach reduces the overall number of inter-region handovers by 1.2% compared to the CGP one.

Figure 5.13 shows the hourly decrease of inter-region handovers during Day 2. The performance of the CGP approach corresponds to the same RAN-to-core node association used in Figure 5.12. This is a more fair comparison between the CGP and the DSO approach, in that the RAN-to-core node association is computed based on historical data and then used in the network. In this case the gain achieved with the CGP approach is evidently lower compared to the DSO approach. Both approaches - even the CGP one relying on outdated information - significantly outperform the geographical association.

The gain achieved with the DSO approach is even more obvious from Table 5.3. The overall gain of the CGP and DSO approach compared to the geographical one are 26.4% and 31.1% respectively, while the gain of the DSO approach compared to the CGP one is 3.7%. This overall gain is more than two times higher compared to the gain during Day 1. This is due to changes in the user movement patterns: a static approach to optimizing the

Table 5.3: The overall inter-region handover reduction, and the number of Inter- and Intra-region handovers during Day 2.

	Geographical	CGP	DSO
Geographical	0	-20.9 %	-23.7 %
CGP	26.4 %	0	-3.6 %
DSO	31.1 %	3.7%	0

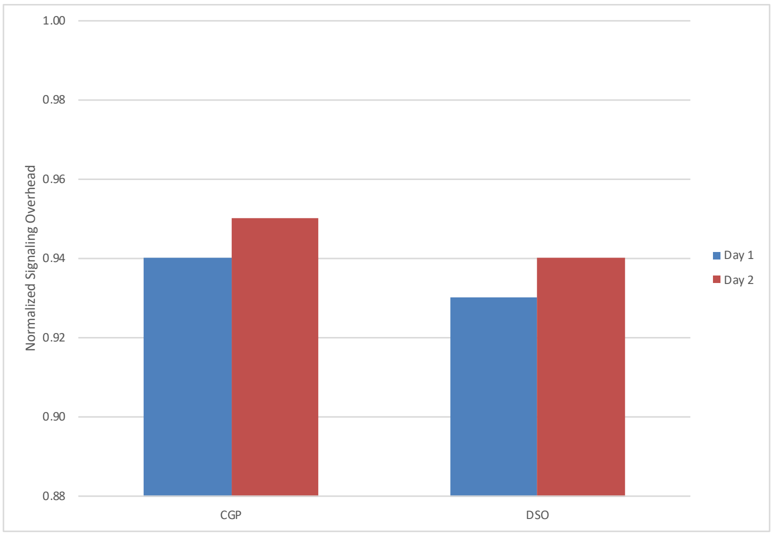


Figure 5.14: Normalized signaling overhead related to the execution of mobility management functions. The values are normalized by the signaling overhead that results from the geographical approach to RAN-to-core node association. For the handover events in our network deployment 5% of reduction is equal to 1 billion signaling messages.

RAN-to-core nodes association is suboptimal.

We also study the signaling overhead related to the handover procedures. As previously mentioned in Section 5.3, inter-region handovers require a significantly larger number of signaling messages compared to intra-region handovers. Therefore, by minimizing the number of inter-region handovers, we minimize the overall network signaling overhead. Figure 5.14 compares the normalized signaling overhead related to the execution of handovers for the CGP and DSO approaches. As shown in Figure 5.14, both optimization techniques, i.e. the CGP and the DSO, clearly outperform and minimize the signaling overhead compared to the traditional geographical clustering approach.

The reduction related to the signaling overhead affects the average handover procedure processing time. According to [181, 182] the average handover procedure processing time also depends on the type of handover. The processing of intra-region handovers is estimated to 50ms, whereas the processing time of inter-region handovers is estimated to be in the range from 100ms to 350ms. Figure 5.15, shows the comparison between the average handover procedure processing time for the CGP, DSO and geographical approach. As shown in Figure 5.15, the CGP and the DSO algorithms significantly reduce the average handover procedure processing time. The results presented in Figure 5.14 and Figure 5.15 clearly

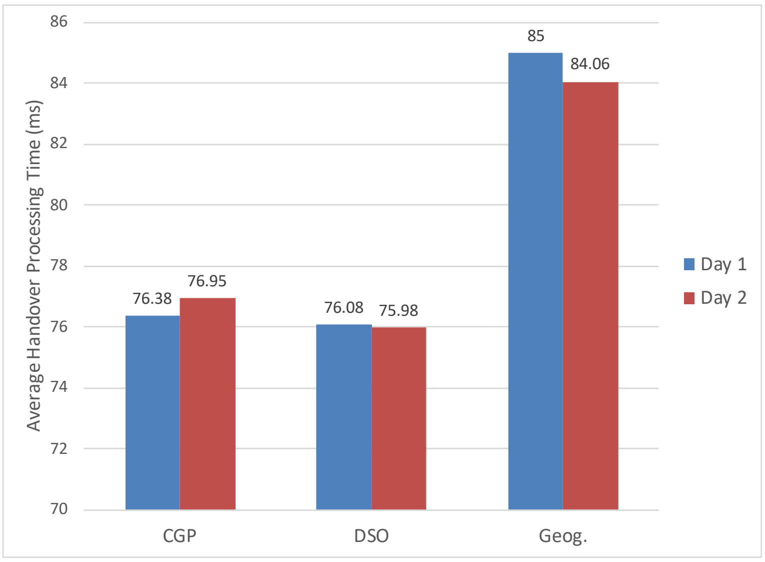


Figure 5.15: Average handover procedure processing time (ms).

highlight the benefits from the handover region optimization techniques presented in this Chapter. They also prove that the delay and signaling overhead related to handover procedures can be reduced without the need to change the procedures themselves, but rather by reconfiguring the association between the RAN nodes and the nodes in the core network.

5.4 Conclusion

Self-organizing and autonomic network management and optimization will play a key role in 5G networks in that they will be essential to fully exploit in a cost efficient way the increased network flexibility and the dynamic on-demand deployment of virtual network functions. However, due to the growing density of network deployments, the proposed approaches to network optimization have to be scalable, adaptable and they should not increase the signaling overhead. Therefore, distributed self-organizing approaches are the best choice.

In this Chapter we present a network simulator that allows us to study self-organization in cellular networks. The proposed simulator allows us to take advantage of ABM principles to model a communication network as a set of agents that communicate to each other. Due to the high configurability of the simulator, we were able to simulate a large network of nodes by focusing on the functions of interest (i.e. mobility management) and abstracting the other signaling procedures.

We also propose a general method to optimize the mobility management function based on user mobility patterns, relying on standardized protocols and mechanisms, which will facilitate adoption in both 4G and 5G networks. Our approach reorganizes nodes in the network so that the number of handovers requiring an MME/AMF reallocation is minimized. We presented a distributed (DSO) and a centralized approach (CGP); both outperform the traditional geographic clustering of BSs. By relying on distributed decision making, the

DSO approach can adapt to the changes in the user movement patterns as they happen, and outperforms the CGP removing also the need for collecting the information at the OSS.

Future work will focus on modeling overlapping handover regions in the centralized approach (which is currently only included in DSO) and examine the implications of dynamic up- and down- scaling of MMEs/AMFs.

6 Conclusions and Future Work

Conclusions and Future Work

The next generation of communication networks is not an evolution of the existing technologies, but rather a revolution in the way we integrate and use those technologies. Networks are changing from being simple pipelines for information transfer to dynamical systems that generate, store, process and communicate information. Networks are growing in size and density, different technologies are being integrated (e.g. Wi-Fi, Internet of Things (IoT), cellular, vehicular) and new services are being deployed and integrated with the existing ones. All these changes force us to change the way we study and design communication networks. We take a complex systems approach to study networks as living organisms that change over time and adapt to changes in the environment. By carefully reviewing the literature on applications of complex systems science to a variety of research areas, we organize the methodologies and tools into three layers: (1) *Analysis*, (2) *Modeling*, and (3) *Design*. We mainly focus on three properties of complex systems (i.e. *Complexity*, *Degeneracy*, and *Emergence*), that allow us to study how structure affects the performance of communication networks.

6.1 Functional Complexity Framework for the Analysis and Modeling of Networks

In Chapter 2, we provide an overview of different complexity metrics that are used to quantify where a system is positioned between a random and a completely regular system. Then, in Chapter 3, we build on this research by studying the *Complexity* of functions in communication networks. We start by introducing a framework to map network functions into graphs, that we call functional topologies. This framework allows us to model the communication between functional entities during the execution of a network function as a complex system.

Starting from a complexity metric proposed in neurobiology - *neural complexity* - we develop a complexity metric - *functional complexity* - which allows us to quantify the variety of structural patterns in the functional topology. By studying how frequency allocation in cellular networks is affected by the structure that underpins their operation, we correlate the characteristics of the outcome with the complex relationships that underpin the functional structure. In the case of Wireless Sensor Networks (WSNs) we show how the density of local connections in the functional topology affects scalability and energy efficiency.

From the engineering point of view, a metric like *functional complexity* has its advantages. Unlike the traditional way of computing network Key Performance Indicators (KPIs), which implies collecting measurements from an operational network, *functional complexity* relies on the organizational structure of the communication during the execution of a network function, which is predefined by the protocol procedures, allowing us to compute it offline. This means that by correlating complexity and different KPIs, we capture certain network dynamics without relying on measurements, allowing us to optimize protocol procedures even before the network deployment.

6.2 Degeneracy Framework for the Analysis of In-Network Computing in IoT

Chapter 4 relies on the functional framework introduced in Chapter 3, but instead of *Complexity*, we focus on the concept of *Degeneracy.* In this chapter, we also focus on all four aspects of the new generation of communication networks (i.e. generating, storing, processing, and communicating information) by studying distributed computation in an IoT network.

By relying on the functional framework, we are able to define *degeneracy* and *redundancy* by using concepts from statistical mechanics. Degeneracy arises from structurally different functional topologies having functionally identical properties. While degenerate topologies are partially overlapping, redundant functional topologies are completely independent, i.e. they do not have any common nodes or edges.

We developed an efficient algorithm that allows us to analyze any physical topology, and generate all functional topologies satisfying a given delay requirement. By studying the computational complexity of the proposed algorithm, we show its applicability to realistic network scenarios.

In the end, we provide a comparison between degenerate in-network computing and a traditional multi-hop scheme that selects a single optimal graph for the computation. We show that the probability of successful computation is significantly increased by exploiting the full degeneracy potential of the underlying physical topology.

6.3 Self-Organization in Cellular Networks Based on Local Interactions

Chapter 5 includes work on all three layers (i.e. *Analysis*, *Modeling* and *Design*) to study the concept of *Emergence* in cellular networks. Similarly to the previous chapters, we study the implementation of network functions as complex systems.

We emphasize the importance of self-organization in future communication networks. Since the concept of Self-Organizing Networks (SONs) has been used in communication networks before, we start by highlighting the difference between the 3rd Generation Partnership Project (3GPP) definition and the broader meaning of self-organization in complex systems. While 3GPP solutions only scrape the surface of the huge potential of self-organization in networks, the complex systems approach allows us to fully take advantage of the concept of emergence. More precisely, the complex systems approach to self-organization aims at designing the rules of interaction between the network nodes in a way that guides the network operation towards a desired state.

For the purpose of studying self-organization in cellular networks, we developed a simulator that relies on the principles of Agent-Based Modeling (ABM). The rich concept of time and dynamic understanding of the network behavior that goes beyond a static snapshot of the network, allow us to perform realistic network simulations. On the other hand, the modular design of the simulator allows us to choose the level of abstraction (e.g. which signaling procedures should be included, whether or not to compute the Signal-to-Noise and Interference Ratio (SINR) for the User Equipments (UEs)), allowing us to simulate networks with a large number of Base Stations (BSs) and a large number of UEs.

Then, we demonstrate a distributed and a centralized 4G/5G compliant approach to minimize signaling and latency related to user mobility in cellular networks. The simulator developed in this chapter is used to test the proposed approaches. Our simulations are based on datasets that contain the locations of BSs in San Francisco, Oakland and Berkeley; and real UE movements - taxi mobility traces around the above-mentioned locations.

We study the existing protocol messages related to mobility management in cellular networks, to show that the proposed approaches can be easily integrated within the existing network architecture. We also show that by taking advantage of the principles of emergence, we can design distributed algorithms with very low computational complexity ($O(1)$), which are highly scalable and adaptive.

6.4 Summary of the Findings and Future Work

Although complex systems science is still in its early stages, the applicability to different scientific fields accelerates its development. As shown in Chapter 2, the literature focuses on a wide range of system properties (e.g. scalability, robustness, evolvability) by studying concepts like complexity, degeneracy and emergence. The broad definition of these concepts results in a spectrum of different metrics to quantify them. For example, complexity metrics are trying to quantify either the degree of organization, or how hard it is to describe the system, or how hard it is to create it. Therefore, there is no one-size-fits-all solution to the problems that are being addressed by complex systems science in different scientific fields.

The application of complex systems science to communication networks is still in its infancy. In this thesis we study the functional aspects of communication networks as complex systems. We mainly focus on how the structural organization of these functions and the underlying physical deployment affect the network performance. There is still a lot of work to be done in terms of developing new complexity metrics and adapting existing ones to problems in communication networks, studying dynamic aspects of communication networks, and developing protocols that will enable the implementation of dynamic, robust, adaptable and scalable services.

In terms of complexity metrics, we started studying the relationship between complexity and different network KPIs. As future work, it would be interesting to further investigate the relationship between different complexity metrics and a wide range of KPIs, which would allow us to develop a better understanding of how structure affects different functions.

Since the degeneracy analysis identifies network structures that are capable of performing a network function with regard to observables like delay or energy efficiency, the study of degeneracy might be used in the future to improve computational throughput, computational resilience, deploy superior network configurations, or to analyze the degeneracy potential of a routing protocol. Future work should also focus on exploiting degenerate and redundant functional topologies to perform parallel computing. The case of redundant, i.e. independent, functional topologies, is of particular interest in that bottlenecks can be avoided with no coordination between the nodes of different functional topologies.

The design of emergent algorithms in cellular networks will play a key role in that it will be essential to fully exploit in a cost efficient way the increased network flexibility and the dynamic on-demand deployment of virtual network functions. Rethinking the concept of self-organization in cellular networks - introducing simple rules based on locally available information - allows us to optimize network functions without the need for new protocol messages. Hence, there is a lot of work to be done on the further development of algorithms that rely on local information to improve network-wide performance. Additionally, the concept of bad emergence, i.e. the rise of non-desirable system-level/network-wide properties through the interactions of multiple distributed elements, should be studied. For example, in the study we performed in Chapter 5, a non-desirable behavior would be associated with oscillating handover regions, which would increase the signaling overhead in the network. We prevent these oscillations by defining handover-event counters and ping-pong thresholds. However, the study of bad emergence should certainly be addressed in more detail. To support the development of emergent network algorithms, simulators should be developed with large and dense deployments, mobility, dynamic resource allocation and self-organization in mind.

Acronyms

ABM	Agent-Based Modeling
ANTS	Agent-based NeTwork mobility Simulator
AMF	Access and Mobility Function
BFS	Breadth-First Search
BLER	BLock Error Rate
BS	Base Station
CDN	Content Delivery Network
CGP	Centralized Graph Partitioning
CPR	Common-Pool Resource
DSO	Distributed Self-Organized
EBM	Equation-Based Modeling
EPC	Evolved Packet Core
GPP	Graph Partitioning Problem
GUMMEI	Globally Unique MME Identifier
GUTI	Globally Unique Temporary UE Identity
HCC	Hierarchical Control Clustering
IoT	Internet of Things
KPI	Key Performance Indicator
LEACH	Low Energy Adaptive Clustering Hierarchy
LTE	Long Term Evolution
MAC	Medium Access Control

MIMO	Multiple-Input Multiple-Output
MME	Mobility Management Entity
M-TMSI	MME Temporary Mobile Subscriber Identity
NFV	Network Function Virtualization
OSS	Operations Support System
PGW	Packet Gateway
RACH	Random-Access Channel
RAN	Radio Access Network
SBA	Service Based Architecture
SDN	Software Defined Networking
SGW	Serving Gateway
SINR	Signal-to-Noise and Interference Ratio
SMF	Session Management Function
SON	Self-Organizing Network
TA	Tracking Area
TAU	Tracking Area Update
UAV	Unmanned Aerial Vehicle
UE	User Equipment
UGV	Unmanned Ground Vehicle
UPF	User Plane Function
VANET	Vehicular ad-hoc network
WSN	Wireless Sensor Network
3GPP	3rd Generation Partnership Project
5G	Fifth Generation

Bibliography

[1] E. Morin, "Restricted Complexity, General Complexity," *Science and us: Philosophy and Complexity. Singapore: World Scientific*, pp. 1–25, 2007.

[2] M. Mitchell, *Complexity - A Guided Tour.* Oxford University Press, 2011.

[3] F. Capra and P. L. Luisi, *The Systems View of Life: A Unifying Vision.* Cambridge University Press, 2014.

[4] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Observability of complex systems." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 110, no. 7, pp. 2460–5, 2013.

[5] Y. Bar-Yam, "Multiscale Complexity/Entropy," *Advances in Complex Systems*, vol. 7, no. 1, pp. 47–63, 2004.

[6] L. Yaeger, "Evolution Selects For and Against Complexity," *Networks and Complex Systems Course, Indiana University*, 2007. [Online]. Available: http://vw.slis.indiana.edu/talks-fall07/Yeager.pdf

[7] G. Tononi, O. Sporns, and G. M. Edelman, "A Measure for Brain Complexity: Relating Functional Segregation and Integration in the Nervous System." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 91, no. 11, pp. 5033–5037, 1994.

[8] R. Lopez-Ruiz, H. L. Mancini, and X. Calbet, "A Statistical Measure of Complexity," *Physics Letters A*, vol. 209, no. 5-6, pp. 321–326, 1995.

[9] D. Balduzzi and G. Tononi, "Integrated information in discrete dynamical systems: Motivation and theoretical framework," *PLoS Computational Biology*, vol. 4, no. 6, 2008.

[10] N. J. Joshi, G. Tononi, and C. Koch, "The Minimal Complexity of Adapting Agents Increases with Fitness," *PLoS Comput Biol*, vol. 9, no. 7, 2013.

[11] T. P. Evans, E. Ostrom, and C. Gibson, "Scaling Issues With Social Data in Integrated Assessment Modeling," *Integrated Assessment*, vol. 3, pp. 135–150, 2002.

[12] M. J. Lanham, G. P. Morgan, and K. M. Carley, "Social network modeling and agent-based simulation in support of crisis de-escalation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 1, pp. 103–110, 2014.

[13] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding Individual Human Mobility Patterns," *Nature*, vol. 453, no. 7196, p. 779, 2008.

[14] Z. Wang, D. Zhang, X. Zhou, D. Yang, Z. Yu, and Z. Yu, "Discovering and profiling overlapping communities in location-based social networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, pp. 499–509, 2014.

[15] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, p. 761, 2010.

[16] T. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Physical Review E*, vol. 80, no. 1, 2009.

[17] M. E. Newman and M. Girvan, "Finding and Evaluating Community Structure in Networks," *Physical review E*, vol. 69, no. 2, 2004.

[18] M. E. Newman, "Modularity and Community Structure in Networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[19] ——, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[20] S. K. Butler, "Eigenvalues and structures of graphs," *UC San Diego, PhD dissertation*, 2008.

[21] C. Faloutsos and D. Chakkrabarti, "Graph Mining: Laws, Generators and Tools," *Lecture Notes in Computer Science*, 2008.

[22] P. Wang, M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding the Spreading Patterns of Mobile Phone Viruses," *Science*, vol. 324, pp. 1071–1076, 2009.

[23] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and a L Barabási, "Structure and Tie Strengths in Mobile Communication Networks." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 18, pp. 7332–7336, 2007.

[24] C. A. Hidalgo and C. Rodríguez-Sickert, "The Dynamics of a Mobile Phone Network," *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 12, pp. 3017–3024, 2008.

[25] J. Candia, M. C. González, P. Wang, T. Schoenharl, G. Madey, and A.-L. Barabási, "Uncovering Individual and Collective Human Dynamics from Mobile Phone Records," *Journal of physics A: mathematical and theoretical*, vol. 41, no. 22, pp. 1–11, 2008.

[26] H. Beigy and M. R. Meybodi, "Cellular Learning Automata With Multiple Learning Automata in Each Cell and Its Applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 1, pp. 54–65, 2010.

[27] I. Macaluso, H. Cornean, N. Marchetti, and L. Doyle, "Complex communication systems achieving interference-free frequency allocation," *IEEE International Conference on Communications (ICC)*, pp. 1447–1452, 2014.

[28] I. Macaluso, C. Galiotto, N. Marchetti, and L. Doyle, "A Complex Systems Science Perspective on Wireless Networks," *Journal of Systems Science and Complexity*, no. 10, pp. 1–27, 2016.

[29] G. Westhorp, "Using Complexity-Consistent Theory for Evaluating Complex Systems," *Evaluation*, vol. 18, no. 4, pp. 405–420, 2012.

[30] W. B. Arthur, "Inductive Reasoning and Bounded Rationality," *The American Economic Review*, vol. 84, no. 2, pp. 406–411, 1994.

[31] J. W. Forrester, "Counterintuitive Behavior of Social Systems," *Technological Forecasting and Social Change*, vol. 3, pp. 1–22, 1971.

[32] J.-Y. Le Boudec, D. McDonald, and J. Mundinger, "A Generic Mean Field Convergence Result for Systems of Interacting Objects," *Fourth International Conference on the Quantitative Evaluation of Systems*, pp. 3–18, 2007.

[33] V. Grimm, E. Revilla, U. Berger, F. Jeltsch, W. M. Mooij, S. F. Railsback, H.-H. Thulke, J. Weiner, T. Wiegand, and D. L. DeAngelis, "Pattern-Oriented Modeling of Agent-Based Complex Systems: Lessons from Ecology," *Science*, vol. 310, no. 5750, pp. 987–991, 2005.

[34] A. M. Turing, "The chemical basis of morphogenisis," *Philos. Trans. R. Soc. London*, no. 273, pp. 37–72, 1952.

[35] J. D. Murray, "Mathematical Biology: I. An Introduction," *Springer-Verlag*, 1989.

[36] M. C. Cross and P. C. Hohenberg, "Pattern Formation Outside of Equilibrium," *Reviews of Modern Physics*, vol. 65, no. 3, pp. 851–1112, 1993.

[37] R. E. Mirollo and S. H. Strogatz, "Synchronization of Pulse-Coupled Biological Oscillators," *SIAM Journal on Applied Mathematics*, vol. 50, no. 6, pp. 1645–1662, 1990.

[38] A. Pikovsky, M. Rosenblum, J. Kurths, and J. Kurths, *Synchronization: A Universal Concept in Nonlinear Sciences.* Cambridge university press, 2003.

[39] W. Elmenreich, R. DŚouza, C. Bettstetter, and H. de Meer, "A Survey of Models and Design Methods for Self-organizing Networked Systems," *Self-Organizing Systems*, pp. 37–49, 2009.

[40] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Reviews of Modern Physics*, vol. 55, no. 3, p. 601, 1983.

[41] U. Wilensky and W. Rand, *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo.* MIT Press, 2015.

[42] M. Bristow, L. Fang, and K. W. Hipel, "Agent-based modeling of competitive and cooperative behavior under conflict," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, pp. 834–850, 2014.

[43] K. Batool and M. A. Niazi, "Modeling the Internet of Things: A Hybrid Modeling Approach Using Complex Networks and Agent-Based Models," *Complex Adaptive Systems Modeling*, vol. 5, no. 1, p. 4, 2017.

[44] N. J. Kaminski, M. Murphy, and N. Marchetti, "Agent-Based Modeling of an IoT Network," *2016 IEEE international symposium on systems engineering (ISSE)*, pp. 1–7, 2016.

[45] E. Bonabeau, D. d. R. D. F. Marco, M. Dorigo, G. Theraulaz *et al.*, *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, 1999.

[46] E. Bonabeau, P. Funes, and B. Orme, "Exploratory Design of Swarms," *2nd International Workshop on the Mathematics and Algorithms of Social Insects*, 2003.

[47] I. Fehérvári and W. Elmenreich, “Evolutionary Methods in Self-organizing System Design.” *International Conference on Genetic and Evolutionary Methods*, pp. 10–15, 2009.

[48] O. Simeone, U. Spagnolini, Y. Bar-Ness, and S. H. Strogatz, “Distributed Synchronization in Wireless Networks,” *IEEE Signal Processing Magazine*, vol. 25, no. 5, pp. 81–97, 2008.

[49] V. Volterra, *Leçons sur la théorie mathématique de la lutte pour la vie.* Gauthier-Villars, 1931.

[50] G. Di Caro, F. Ducatelle, and L. M. Gambardella, “AntHocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks,” *European Transactions on Telecommunications*, vol. 16, no. 5, pp. 443–455, 2005.

[51] C. Gershenson, *Design and Control of Self-Organizing Systems.* CopIt Arxives, 2007.

[52] M. Resnick, *Turtles, termites, and traffic jams: Explorations in massively parallel microworlds.* Mit Press, 1997.

[53] A. Tonmukayakul and M. B. Weiss, “A Study of Secondary Spectrum use Using Agent-Based Computational Economics,” *NETNOMICS: Economic Research and Electronic Networking*, vol. 9, no. 2, pp. 125–151, 2008.

[54] G. Tononi, G. M. Edelman, and O. Sporns, “Complexity and Coherency: Integrating Information in the Brain,” *Trends in Cognitive Sciences*, vol. 2, no. 12, pp. 474–484, 1998.

[55] G. Tononi, O. Sporns, and G. M. Edelman, “Measures of Degeneracy and Redundancy in Biological Networks,” *Proceedings of the National Academy of Sciences*, vol. 96, no. 6, 1999.

[56] J. Goldstein, “Emergence as a Construct: History and Issues,” *Emergence*, vol. 1, no. 1, pp. 49–72, 1999.

[57] J. M. Whitacre, “Degeneracy: A Link between Evolvability, Robustness and Complexity in Biological Systems,” *Theoretical Biology and Medical Modelling*, vol. 7, no. 1, 2010.

[58] O. Sporns, G. Tononi, and G. M. Edelman, “Connectivity and Complexity: The Relationship between Neuroanatomy and Brain Dynamics,” *Neural Networks*, vol. 13, no. 8-9, pp. 909–922, 2000.

[59] M. Randles, D. Lamb, E. Odat, and A. Taleb-Bendiab, “Distributed Redundancy and Robustness in Complex Systems,” *Journal of Computer and System Sciences*, vol. 77, no. 2, pp. 293–304, 2011.

[60] D. P. Feldman, C. S. McTague, and J. P. Crutchfield, “The Organization of Intrinsic Computation: Complexity-Entropy Diagrams and the Diversity of Natural Information Processing,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 18, no. 4, 2008.

[61] W. Bialek, I. Nemenman, and N. Tishby, “Complexity through Nonextensivity,” *Physica A: Statistical Mechanics and its Applications*, vol. 302, no. 1-4, pp. 89–99, 2001.

[62] D. P. Feldman and J. P. Crutchfield, "Structural Information in Two-Dimensional Patterns: Entropy Convergence and Excess Entropy," *Physical Review E*, vol. 67, no. 5, pp. 1–9, 2003.

[63] W. Warren, "Science and Complexity," *American Scientist*, no. 36, pp. 536–544, 1948.

[64] S. Lloyd, "Measures of Complexity: A Nonexhaustive List," *Control Systems, IEEE*, vol. 21, no. 4, pp. 7–8, 2001.

[65] G. Tononi, O. Sporns, and G. M. Edelman, "A Complexity Measure for Selective Matching of Signals by the Brain," *Proceedings of the National Academy of Sciences*, vol. 93, no. 8, 1996.

[66] D. H. Wolpert and W. Macready, "Self-Dissimilarity: An Empirically Observable Complexity Measure," *Unifying Themes in Complex Systems*, pp. 626–643, 1997.

[67] A. Shiryayev, *Selected Works of AN Kolmogorov: Volume III: Information Theory and the Theory of Algorithms.* Springer Science & Business Media, 2013, vol. 27.

[68] G. J. Chaitin, *Information, Randomness & Incompleteness: Papers on Algorithmic Information Theory.* World Scientific, 1990, vol. 8.

[69] S. Lloyd and H. Pagels, "Complexity as Thermodynamic Depth," *Annals of Physics*, vol. 188, no. 1, pp. 186–213, 1988.

[70] M. Gell-Mann, "What is Complexity?" *Complexity and Industrial Clusters*, pp. 13–24, 2002.

[71] H. V. Ribeiro, L. Zunino, E. K. Lenzi, P. A. Santoro, and R. S. Mendes, "Complexity-Entropy Causality Plane as a Complexity Measure for Two-Dimensional Patterns," *PloS one*, vol. 7, no. 8, pp. 1–9, 2012.

[72] O. Rosso, L. Zunino, D. Pérez, A. Figliola, H. Larrondo, M. Garavaglia, M. Martín, and A. Plastino, "Extracting Features of Gaussian Self-Similar Stochastic Processes via the Bandt-Pompe Approach," *Physical Review E*, vol. 76, no. 6, 2007.

[73] G. M. Edelman and J. A. Gally, "Degeneracy and Complexity in Biological Systems," *Proceedings of the National Academy of Sciences*, vol. 98, no. 24, pp. 13 763–13 768, 2001.

[74] R. Thanki, "The Economic Significance of Licence-Exempt Spectrum to the Future of the Internet," *White Paper*, 2012.

[75] N. McBride, J. Bulava, C. Galiotto, N. Marchetti, I. Macaluso, and L. Doyle, "Degeneracy Estimation in Interference Models on Wireless Networks," *Physica A: Statistical Mechanics and its Applications*, vol. 469, pp. 540–550, 2017.

[76] R. Frei and J. Whitacre, "Degeneracy and Networked Buffering: Principles for Supporting Emergent Evolvability in Agile Manufacturing Systems," *Natural Computing*, vol. 11, no. 3, pp. 417–430, 2012.

[77] J. Fromm, "Ten Questions about Emergence," *arXiv*, 2005.

[78] C. Hooker, "Philosophy of Complex Systems," *Handbook of the Philosophy of Science*, vol. 10, 2011.

[79] M. A. Bedau, “Weak Emergence,” *appearing in book "Philosophical Perspectives: Mind, Causation and World", Blackwell Publishers*, vol. 11, pp. 375–399, 1997.

[80] ——, “Downward Causation and the Autonomy of Weak Emergence,” *Principia: An International Journal of Epistemology*, vol. 6, no. 1, pp. 5–50, 2002.

[81] V. Darley, “Emergent Phenomena and Complexity,” *Artificial Life*, vol. 4, pp. 411–416, 1994.

[82] J. Deguet, Y. Demazeau, and L. Magnin, “Elements About the Emergence Issue: A Survey of Emergence Definitions,” *Complexus*, vol. 3, no. 1-3, pp. 24–31, 2006.

[83] T. O’Connor, “Emergent Properties,” *American Philosophical Quarterly*, vol. 31, no. 2, pp. 91–104, 1994.

[84] R. K. Sawyer, “Simulating Emergence and Downward Causation in Small Groups,” *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pp. 49–67, 2000.

[85] C. Prehofer and C. Bettstetter, “Self-Organization in Communication Networks: Principles and Design Paradigms,” *IEEE Communications Magazine*, vol. 43, no. 7, pp. 78–85, 2005.

[86] S. H. Strogatz, “Exploring Complex Networks,” *Nature*, vol. 410, no. 6825, p. 268, 2001.

[87] J. Klinglmayr, C. Bettstetter, M. Timme, and C. Kirst, “Convergence of Self-Organizing Pulse-Coupled Oscillator Synchronization in Dynamic Networks,” *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1606–1619, 2017.

[88] C. Grabow, S. M. Hill, S. Grosskinsky, and M. Timme, “Do Small Worlds Synchronize Fastest?” *Europhysics Letters*, vol. 90, no. 4, 2010.

[89] D. J. Watts and S. H. Strogatz, “Collective Dynamics of Small-World Networks,” *Nature*, vol. 393, pp. 440–442, 1998.

[90] M. Barahona and L. M. Pecora, “Synchronization in Small-World Systems,” *Physical Review Letters*, vol. 89, no. 5, 2002.

[91] A. Roxin, H. Riecke, and S. A. Solla, “Self-Sustained Activity in a Small-World Network of Excitable Neurons,” *Physical Review Letters*, vol. 92, no. 19, 2004.

[92] M. Niazi and A. Hussain, “Agent-Based Tools for Modeling and Simulation of Self-Organization in Peer-to-Peer, Ad Hoc, and Other Complex Networks,” *IEEE Communications Magazine*, vol. 47, no. 3, pp. 166–173, 2009.

[93] A. Tonmukayakul and M. B. Weiss, “An Agent-Based Model for Secondary use of Radio Spectrum,” *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, pp. 467–475, 2005.

[94] 3GPP, “3GPP TS 32.500 Version 15.0.0,” 2018.

[95] R. Mathar and J. Mattfeldt, “Pulse-Coupled Decentral Synchronization,” *SIAM Journal on Applied Mathematics*, vol. 56, no. 4, pp. 1094–1106, 1996.

[96] A. Patel, J. Degesys, and R. Nagpal, “Desynchronization: The theory of Self-Organizing Algorithms for Round-Robin Scheduling,” *International Conference on Self-Adaptive and Self-Organizing Systems*, pp. 87–96, 2007.

[97] R. Leidenfrost and W. Elmenreich, “Firefly Clock Synchronization in an 802.15.4 Wireless Network,” *EURASIP Journal on Embedded Systems*, 2009.

[98] A. Tyrrell, G. Auer, and C. Bettstetter, “Emergent Slot Synchronization in Wireless Networks,” *IEEE Transactions on Mobile Computing*, vol. 9, no. 5, pp. 719–732, 2010.

[99] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, “A Knowledge Plane for the Internet,” *Applications, technologies, architectures, and protocols for computer communications*, pp. 3–10, 2003.

[100] P. Leitão, A. W. Colombo, and S. Karnouskos, “Industrial Automation Based on Cyber-Physical Systems Technologies: Prototype Implementations and Challenges,” *Computers in Industry*, vol. 81, pp. 11–25, 2016.

[101] N. J. Kaminski, “Social Intelligence for Cognitive Radios,” *Virginia Polytechnic Institute and State University, PhD Dissertation*, 2014.

[102] M. Newman, “Networks: An Introduction,” *Oxford University Press, Inc.*, 2010.

[103] M. Fiedler, “Algebraic Connectivity of Graphs,” *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

[104] B. Markovsky, J. Skvoretz, D. Willer, and M. J. Lovaglia, “The Seeds of Weak Power: An Extension of Network Exchange Theory,” *Am.J. Sociol*, no. 58, 1993.

[105] S. Banerjee and S. Khuller, “A Clustering Scheme for Hierarchical Control in Wireless Networks,” *Proceedings of IEEE INFOCOM 2001*, pp. 1028–1037, 2001.

[106] A. A. Abbasi and M. Younis, “A survey on clustering algorithms for wireless sensor networks,” *Computer Communications*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[107] M. M. Afsar and M. H. Tayarani-N, “Clustering in Sensor Networks: A Literature Survey,” *Journal of Network and Computer Applications*, vol. 46, pp. 198–226, 2014.

[108] M. Aslam, N. Javaid, A. Rahim, U. Nazir, A. Bibi, and Z. A. Khan, “Survey of extended LEACH-based clustering routing protocols for wireless sensor networks,” *Proceedings of the 14th IEEE International Conference on High Performance Computing and Communications, HPCC-2012 - 9th IEEE International Conference on Embedded Software and Systems, ICESS-2012*, pp. 1232–1238, 2012.

[109] C. Jiang, D. Yuan, and Y. Zhao, “Towards Clustering Algorithms in Wireless Sensor Networks-A Survey,” *IEEE Wireless Communications and Networking Conference (WCNC2009)*, pp. 1–6, 2009.

[110] X. Liu, “A survey on clustering routing protocols in wireless sensor networks,” *Sensors*, vol. 12, no. 8, 2012.

[111] R. Nagpal and D. Coore, “An Algorithm for Group Formation in an Amorphous Computer,” *Proc. 10th International Conference on Parallel and Distributed Computing Systems*, 1998.

[112] A. Ephremides, J. E. Wieselthier, and D. J. Baker, "Design Concept for Reliable Mobile Radio Networks With Frequency Hopping Signaling." *Proceedings of the IEEE*, vol. 75, no. 1, pp. 56–73, 1987.

[113] K. X. K. Xu and M. Gerla, "A Heterogeneous Routing Protocol Based on a New Stable Clustering Scheme," *MILCOM 2002 Proceedings*, vol. 2, pp. 838–843, 2002.

[114] D. J. Baker and A. Ephremides, "The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm," *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1694–1701, 1981.

[115] C. R. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, pp. 1265–1275, 1997.

[116] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications.*, pp. 1713–1723, 2003.

[117] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

[118] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 1, 2004.

[119] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: Recent developments and deployment challenges," *IEEE Network*, vol. 20, no. 3, pp. 20–25, 2006.

[120] S. Basagni, "Distributed clustering for ad hoc networks," *Proceedings Fourth International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN'99)*, pp. 1–15, 1999.

[121] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-Efficient Routing Protocols in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 551–591, 2013.

[122] K. Pattanayak, A. Chatterjee, M. Dzaferagic, S. S. Das, and N. Marchetti, "A Functional Complexity Framework for Dynamic Resource Allocation in VANETs," *International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 458–463, 2018.

[123] 3GPP, "3GPP TS 32.450 Version 9.1.0," 2010.

[124] P. H. Mason, "Degeneracy: Demystifying and Destigmatizing a Core Concept in Systems Biology," *Complexity*, vol. 20, no. 3, 2015.

[125] V. Shah, B. K. Dey, and D. Manjunath, "Network Flows for Function Computation," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 4, pp. 714–730, 2013.

[126] S. Kannan and P. Viswanath, "Multi-Session Function Computation and Multicasting in Undirected Graphs," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 4, pp. 702–713, 2013.

[127] J. Liu, C. H. Xia, N. B. Shroff, and X. Zhang, "On Distributed Computation Rate Optimization for Deploying Cloud Computing Programming Frameworks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 63–72, 2013.

[128] P. Vyavahare, N. Limaye, D. Manjunath, P. Vyavahare, N. Limaye, and D. Manjunath, "Optimal Embedding of Functions for In-Network Computation: Complexity Analysis and Algorithms," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 4, pp. 2019–2032, 2016.

[129] Y. Yang, S. Kar, and P. Grover, "Graph Codes for Distributed Instant Message Collection in an Arbitrary Noisy Broadcast Network," *IEEE Transactions on Information Theory*, vol. 63, no. 9, 2017.

[130] E. Di Pascale, I. Macaluso, A. Nag, M. Kelly, and L. Doyle, "The Network as a Computer: a Framework for Distributed Computing over IoT Mesh Networks," *IEEE Internet of Things Journal*, 2018.

[131] A. Giridhar and P. R. Kumar, "Toward a Theory of In-Network Computation in Wireless Sensor networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 98–107, 2006.

[132] A. Jindal and M. Liu, "Networked Computing in Wireless Sensor Networks for Structural Health Monitoring," *Networking, IEEE/ACM Transactions on*, vol. 20, no. 4, pp. 1203–1216, 2012.

[133] F. Zeng, C. Li, and Z. Tian, "Distributed Compressive Spectrum Sensing in Cooperative Multihop Cognitive Networks," *IEEE Journal on Selected Topics in Signal Processing*, vol. 5, no. 1, pp. 37–48, 2011.

[134] R. Olfati-Saber, "Distributed Kalman Filtering for Sensor Networks," *IEEE Conference on Decision and Control*, pp. 5492–5498, 2007.

[135] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: An Acquisitional Query Processing System for Sensor Networks," *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 122–173, 2005.

[136] A. Giridhar and P. R. Kumar, "Computing and Communicating Functions Over Sensor Networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 755–764, 2005.

[137] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-Network Aggregation Techniques for Wireless Sensor Networks: A Survey," *IEEE Wireless Communication*, pp. 70–87, 2007.

[138] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, "Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks," *IEEE Workshop on Mobile Computing Systems and Applications*, pp. 49–58, 2002.

[139] O. Diallo, J. J. P. C. Rodrigues, M. Sene, and J. Lloret, "Distributed Database Management Techniques for Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 604–620, 2015.

[140] J. Cheriyan and M. R. Salavatipour, "Packing Element-Disjoint Steiner Trees," *ACM Transactions on Algorithms (TALG)*, vol. 3, no. 4, 2007.

[141] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, “Virtual network embedding: A Survey,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[142] S. Bhardwaj, L. Jain, and S. Jain, “Cloud Computing: A Study of Infrastructure as a Service (IAAS),” *International Journal of engineering and information Technology*, vol. 2, no. 1, pp. 60–63, 2010.

[143] A. Haider, R. Potter, and A. Nakao, “Challenges in Resource Allocation in Network Virtualization,” *20th ITC Specialist Seminar*, vol. 18, 2009.

[144] A. Belbekkouche, M. M. Hasan, and A. Karmouch, “Resource Discovery and Allocation in Network Virtualization,” *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 1114–1128, 2012.

[145] J. Inführ and G. R. Raidl, “Introducing the Virtual Network Mapping Problem with Delay, Routing and Location Constraints,” *International Conference on Network Optimization*, pp. 105–117, 2011.

[146] W.-L. Yeow, C. Westphal, and U. Kozat, “Designing and Embedding Reliable Virtual Infrastructures,” *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, pp. 33–40, 2010.

[147] J. F. Botero, X. Hesselbach, A. Fischer, and H. De Meer, “Optimal Mapping of Virtual Networks with Hidden Hops,” *Telecommunication Systems*, vol. 51, no. 4, pp. 273–282, 2012.

[148] R. E. Korf, “Depth-First Iterative-Deepening: An Optimal Admissible Tree Search,” *Artificial Intelligence*, vol. 27, no. 1, pp. 97–109, 1985.

[149] J. B. Kruskal, “On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem,” *Proceedings of the American Mathematical Society*, vol. 7, no. 1, pp. 48–50, 1956.

[150] M. Faloutsos, P. Faloutsos, and C. Faloutsos, “On Power-Law Relationships of the Internet Topology,” *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4, pp. 251–262, 1999.

[151] C. G. S. M. M. P. Peter Bodik, Wei Hong and R. Thibaux, “Intel Berkeley Research Lab Data (v. 2004-04-28),” http://db.csail.mit.edu/labdata/labdata.html, Apr. 2004.

[152] 3GPP, “3GPP TS 123 501 Version 15.3.0 Release 15,” 2018.

[153] ——, “3GPP TR 23.791 Version 1.0.0 Release 16,” 2018.

[154] M. Patzold, “5G Readiness on the Horizon [Mobile Radio],” *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 6–13, 2018.

[155] Huawei, “4G Wireless Broadband Industry White Paper,” 2017.

[156] P. Alvarez, C. Galiotto, J. van de Belt, D. Finn, H. Ahmadi, and L. DaSilva, “Simulating Dense Small Cell Networks,” *IEEE Wireless Communications and Networking Conference*, pp. 1–6, 2016.

[157] J. C. Ikuno, M. Wrulich, and M. Rupp, “System Level Simulation of LTE Networks,” *IEEE Vehicular Technology Conference*, pp. 1–5, 2010.

[158] 3GPP, “3GPP TS 23.401 Version 11.3.0 Release 11,” 2012.

[159] V.-G. Nguyen, T.-X. Do, and Y. Kim, “SDN and Virtualization-Based LTE Mobile Network Architectures: A Comprehensive Survey,” *Wireless Personal Communications*, vol. 86, no. 3, pp. 1401–1438, 2016.

[160] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, “Towards an Elastic Distributed SDN Controller,” *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, pp. 7–12, 2013.

[161] 3GPP, “3GPP TS 36.423 Version 12.3.0 Release 12,” 2014.

[162] ——, “3GPP TS 36.413 Version 10.2.0 Release 10,” 2011.

[163] ——, “TS 28.533 Version 15.0.0 Release 15,” 2018.

[164] A. Kunz, T. Taleb, and S. Schmid, “On Minimizing Serving GW/MME Relocations in LTE,” *International Wireless Communications and Mobile Computing Conference*, pp. 960–965, 2010.

[165] T. Taleb and A. Ksentini, “Gateway Relocation Avoidance-Aware Network Function Placement in Carrier Cloud,” *ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*, pp. 341–346, 2013.

[166] M. Moradi, L. E. Li, and Z. M. Mao, “SoftMoW: A Dynamic and Scalable Software Defined Architecture for Cellular WANs,” *Workshop on Hot topics in software defined networking*, pp. 201–202, 2014.

[167] M. Moradi, W. Wu, L. E. Li, and Z. M. Mao, “SoftMoW: Recursive and Reconfigurable Cellular WAN Architecture,” *ACM International Conference on Emerging Networking Experiments and Technologies*, pp. 377–390, 2014.

[168] H. Qian, M. Brown, and C. Provost, “Optimized Home Evolved NodeB (eNB) Handover in an LTE Network,” Apr. 15 2014, US Patent 8,699,461.

[169] L. Xu, “Apparatus and Method for a Handover in Mobile Communication System,” Oct. 20 2011, US Patent App. 13/085,756.

[170] M. K. Velamati, J. Furry, J. R. Calippe, and M. Selvam, “System and Method for Load Balancing MMEs and MME Pools,” Feb. 3 2015, US Patent 8,948,014.

[171] K. Chowdhury, A. Gibbs, and R. Koodli, “Dynamic Load Balancing in a Communication Network,” Apr. 23 2013, US Patent 8,428,610.

[172] E. Aqeeli, A. Moubayed, and A. Shami, “Dynamic SON-Enabled Location Management in LTE Networks,” *IEEE Transactions on Mobile Computing*, vol. 17, no. 7, pp. 1511–1523, 2018.

[173] X. Wang, H. Zhang, B. Tang, Y. He, X. Shi, and S. Zhu, “Method, Apparatus and System for Optimizing and Updating Tracking Area,” Nov. 11 2014, US Patent 8,886,202.

[174] J. P. Singh, S. R. Vargantwar, M. K. Shah, and D. Rai, “Tracking Area Reconfiguration Based on Sector Load,” Aug. 12 2014, US Patent 8,804,566.

[175] V.-G. Nguyen, A. Brunstrom, K.-J. Grinnemo, and J. Taheri, “SDN/NFV-Based Mobile Packet Core Network Architectures: A Survey,” *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.

[176] 3GPP, “3GPP TS 36.133 Version 10.1.0 Release 10,” 2011.

[177] G. Karypis and V. Kumar, “A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1999.

[178] Uniwired Labs, “The World’s Largest Open Database of Cell Towers,” 2019. [Online]. Available: https://opencellid.org

[179] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, “CRAWDAD Dataset epfl/mobility (v. 2009-02-24),” Downloaded from https://crawdad.org/epfl/mobility/20090224, Feb. 2009.

[180] 3GPP, “TR 38.913 Version 14.2.0 Release 14,” 2017.

[181] ——, “3GPP TR 25.912 Version 13,” 2016.

[182] Z. Li and M. Wilson, “User Plane and Control Plane Separation Framework for Home Base Stations,” *Fujitsu Scientific and Technical Journal*, vol. 46, no. 1, pp. 79–86, 2010.