University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

Marcin Waniek

Hiding in Social Networks

PhD dissertation

Supervisor
dr hab. Piotr Faliszewski
Department of Computer Science
AGH University of Science and Technology

Auxiliary Supervisor
dr Tomasz Michalak
Institute of Informatics
University of Warsaw

March 2017

Author's declaration:
I hereby declare that this dissertation is my own work.

March 15, 2017 ....
Marcin Waniek

Supervisor's declaration:
The dissertation is ready to be reviewed.

March 15, 2017 ....
dr hab. Piotr Faliszewski

Auxiliary Supervisor's declaration:
The dissertation is ready to be reviewed.

March 15, 2017 ....
dr Tomasz Michalak

Abstract

The Internet and social media have fuelled enormous interest in social network analysis. New tools continue to be developed and used to analyse our personal connections. This raises privacy concerns that are likely to exacerbate in the future. With this in mind, we ask the question: *Can individuals or groups actively manage their connections to evade social network analysis tools?* By addressing this question, the general public may better protect their privacy, oppressed activist groups may better conceal their existence, and security agencies may better understand how terrorists escape detection.

In this dissertation we consider hiding from three different types of social network analysis tools. First, we study how both an individual and a group of nodes can evade analysis utilizing centrality measures, without compromising ability to participate in network's activities. In the second study, we investigate how a community can avoid being identified by a community detection algorithm as a closely cooperating group of nodes. In the third study, we analyse how a presence of a particular edge in a network can be hidden from link prediction algorithms.

For considered problems we analyse their computational complexity and prove that they are usually NP-hard. However, we also provide polynomial-time heuristic solutions that turn out to be effective in practice. We test our algorithms on a number of real-life and artificially generated network datasets.

Keywords: Social networks; Hiding in networks; Centrality measures; Influence models; Community detection; Link prediction; Computational complexity; Heuristic solutions.

ACM classification: Security and privacy → Social network security and privacy. Theory of computation → Social networks. *Theory of computation → Problems, reductions and completeness.*

Streszczenie

Internet oraz media społecznościowe spowodowały ogromny wzrost zainteresowania metodami analizy sieci społecznych. Coraz bardziej zaawansowane narzędzia służą do analizy naszych powiązań z innymi ludźmi. Rodzi to poważne obawy związane z prywatnością. Mając to na uwadze, rozważamy następujące pytanie: *Czy członek lub grupa członków sieci społecznej może aktywnie zarządzać swoimi połączeniami tak, aby uniknąć wykrycia przez narzędzia analizy sieci społecznych?* Odpowiedź na to pytanie pozwoliłaby użytkownikom Internetu lepiej chronić swoją prywatność, grupom aktywistów lepiej ukrywać swoją działalność, a agencjom bezpieczeństwa lepiej rozumieć w jaki sposób organizacje terrorystyczne i kryminalne mogą unikać wykrycia.

W tej pracy rozważamy ukrywanie się przed trzema różnymi narzędziami analizy sieci społecznych. Po pierwsze, badamy jak pojedynczy węzeł lub ich grupa może uniknąć wykrycia przez miary centralności (*ang.* centrality measures), wciąż pozostając zdolnym do brania udziału w działalności sieci. Po drugie, analizujemy jak grupa węzłów może uniknąć zidentyfikowania przez algorytmy wykrywania społeczności (*ang.* community detection algorithms). Po trzecie wreszcie, badamy jak można ukryć istnienie określonej krawędzi w sieci przed algorytmami przewidywania połączeń (*ang.* link prediction algorithms).

Analizujemy złożoność obliczeniową rozważanych zagadnień oraz udowadniamy, że większość z nich to problemy NP-trudne. Tym niemniej prezentujemy również wielomianowe rozwiązania heurystyczne, które okazują się efektywne w praktyce. Nasze algorytmy testujemy na szeregu różnych sieci, tak prawdziwych, jak i wygenerowanych losowo.

Słowa kluczowe: Sieci społeczne; Ukrywanie się w sieciach; Miary centralności; Modele wpływu; Wykrywanie społeczności; Przewidywanie połączeń; Złożoność obliczeniowa; Rozwiązania heurystyczne.

Klasyfikacja ACM: Bezpieczeństwo i prywatność → Bezpieczeństwo i prywatność w sieciach społecznych. Teoria obliczeń → Sieci społeczne. *Teoria obliczeń → Preoblemy, redukcje i zupełność.*

Acknowledgments

I would like to express my deepest gratitude to Tomasz Michalak, my mentor and supervisor, who taught me almost all I know about research (with the rest coming from other people mentioned on this page). I would not be where I am today without him.

I am thankful to Piotr Faliszewski, who helped me put together this dissertation and greatly improved the quality of my writing.

I am grateful to Talal Rahwan, discussions with whom influenced many parts of this dissertation and whose hospitality made far away country feel like home.

I would also like to thank all my other co-authors in this and other lines of research: Michael Wooldridge, Long Tran-Thanh, Agata Nieścieruk, Nicholas Jennings, Marcin Bielecki, Joanna Tyrowicz and Krzysztof Makarski. I have learned a lot from all of them.

Finally, I would like to give thanks to my family and my friends, especially to my mom, my dad, and my brother Paweł. Finishing this dissertation would not be possible without their constant love and support.

My work was supported by the Polish National Science Centre grant *Hiding in Social Networks*, no. 2015/17/N/ST6/03686.

Contents

Chapter 1

Introduction

In this chapter we introduce the issue of hiding in social networks and we outline the organization of this dissertation. Next, we present the body of related literature. Finally, we list our publications that contain elements of this dissertation.

1.1 Motivation

Recent years have lead to an increased interest in the analysis of social networks [131]. Applications of this wide body of research vary from organizing massive viral-marketing campaigns [31], through the analysis of outbreaks of infectious diseases [56], all the way up to fighting global criminal and terrorist organizations [98, 141].

Such a variety of problems gave rise to various social network analysis tools and techniques. Examples of prominent types of tools include centrality measures, community detection algorithms, and link prediction algorithms. Centrality measures [12] are tools to identify key nodes in the network, being it the leader of a terrorist cell [80], the financial institution responsible for the financial crisis [82] or the most important airport in the world-wide air transportation structure [60]. Community detection algorithms [51] can extract communities of closely cooperating individuals from the network data, *e.g.*, associates of a known criminal [49], climate indices related to seasonal rainfall [14] or protein complexes in the interaction diagram [112]. Link prediction algorithms [57] can asses probabilities of existence of hidden or not-yet-formed edges, either to detect missing connections between mafia members [15], to discover the interactions between proteins in biological networks [26], or to recommend to customer a product that she might be interested in buying [37, 88].

Clearly, many situations, where social network analysis tools are being applied can be described as adversarial settings. In other words, members of the networks that are being analysed may be interested in falsifying the results of such analysis, *e.g.*, by modifying the structure of their social connections. At the same time, however, nearly all widely used social network analysis tools treat the objects of their analysis as completely oblivious entities.

In this dissertation we assume the role of a strategic member (or a group of members) of a social network and investigate whether and how they can evade various social network analysis tools. As tools that we consider take into account only the topology of the network, we assume that possible strategies of members that seek to hide themselves consist of adding and removing edges from the network. We investigate the problem both from the computational complexity point of view, *i.e.*, we ask *how hard it is to find an optimal way of hiding*, and from a more practical perspective, *i.e.*, we study the settings where, on one hand, either the knowledge or the set of possible strategies of hiding nodes become limited by the external factors and, on the other hand, imperfect, heuristic, easy to implement solutions suffice.

On a broader note, the questions that this dissertation addresses can be generalized as follows:

1. *Given a seeker who is running a social network analysis tool t^* and some evader(s) whose goal is to evade t^* under certain conditions and constraints, which set of actions should the evader(s) take to achieve that goal?*

2. *How hard would it be to compute such an optimal set?*

3. *And how effective would this set be against t^*?*

The motivation of our work is twofold. The first aspect is privacy protection. Social connections of millions of users of such social media sites as *Facebook*, *Twitter* or *Google Plus* can easily be used to uncover information that most of their users would prefer to remain private [40]. The way we use the Web is under constant surveillance by Internet service providers, governments, and private companies. The situation is even more dire in authoritarian regimes, where Internet content is the object of direct censorship and people spreading opinions that are not consistent with government's ideology are subject to serious repressions, including incarceration. King *et al.* performed a study of Internet censorship in China [76, 77]. It indicated that the main goal of the Chinese Internet police is not as much preventing the publication of undesirable content, as it is blocking any form of society self-organization. Under such and similar circumstances, any simple hiding techniques that can be utilised without knowledge about the structure of an entire network and without the need to perform extensive computation may be valuable to anti-government activists and ordinary Internet users alike.

The second aspect of hiding in networks is the potential use of presented (or similar) techniques by malevolent groups, such as terrorist and criminal organisations. The need of secrecy is one of the driving forces of their day-to-day activities. It seems to be a well-motivated assumption that over the course of years and decades of operating outside the law they developed various ways of concealing their actions. Since social network analysis tools are a part of analytic software used by police forces and intelligence agencies for many years now [64], members of the dark networks might have already sought means of remaining undetected by them. Investigating such venues of possible avoidance detection may help law enforcement agencies in identifying true ringleaders of malevolent groups.

Unfortunately, at the moment we lack sufficient understanding of evasion techniques that can falsify the results of social network analysis tools. What is more, currently existing tools do not even have the ability to internalize such evasion techniques. The reason of this is most social network analysis tools are built around the assumption that members of the network do not act strategically to evade these tools. Even the more advanced tools dedicated to analysing covert networks [117] typically assume that the investigated network is not subject to strategic manipulation. Given this, we want to direct attention towards the strategic evasion of social network analysis tools, as discussed by Michalak *et al.* [102].

Our work can be considered as a first step in the strategic analysis of hiding in social networks. To the best of our knowledge, we are the first to propose this kind of adversary setting in the social network analysis, where the strategic behaviour of network actors is explicitly considered.

1.2 Organization and Results

The remainder of the dissertation is organized as follows:

- In Chapter 2, we describe basic notation and network datasets that are used throughout the whole dissertation. We also provide a short summary of computational complexity concepts required for the theoretical part of this dissertation, including the notion of a Turing machine, P and NP complexity classes, NP-hardness and NP-completeness. We also define the NP-complete problems used in our reductions.

- In Chapter 3, we analyse the problem of hiding a single node in a network, by reducing its centrality value without hurting its influence over the network. Most defined problems turn out to be NP-hard, *i.e.*, finding the optimal solutions is an extremely demanding task. Given this difficulty, we design a heuristic solution that is scalable and easy to implement even for a lay person. Surprisingly, it proves to be efficient in practice and allows the network members who wish to maintain their privacy to do so.

- In Chapter 4, we extend the setting to hiding a group of nodes (network leaders) by ensuring their low positions in centrality rankings. Contrary to simple intuition, the problem proves to be intractable even for degree centrality—perhaps computationally simplest centrality measure. Next, we consider how, instead of modifying already existing network, one could construct a network from scratch so that the leaders would occupy low positions in the rankings. Our analysis aims to shed some light on how leaders of criminal and terrorist organizations design their networks to increase secrecy and safety of the leadership element.

- In Chapter 5, we approach the problem of group hiding from a different perspective. We investigate, how a group of nodes can avoid identification as a separate entity by community detection algorithms. We describe an optimal way of adding or removing a single edge from the network to lower the modularity of a given community structure. We also design and test a simple heuristic that allows a group of nodes to conceal themselves as a community, irrespectively of which community detection algorithm is used against them.

- In Chapter 6, we analyse the ways of hiding an edge in a network, by preventing its identification by a link prediction algorithm. We show that even for the simplest class of these algorithms, *i.e.*, local similarity indices, the problem of hiding edges is NP-complete. To address this hardness results in a practical way, we propose two polynomial time heuristic solutions—one focused on adding edges to the network and one focused on removing them—that allow to conceal a chosen set of connections by hindering their detection by link prediction algorithms.

- Finally, in Chapter 7, we summarize our main findings. We also describe the limitations of our approach and suggest potential directions for future work.

1.3 Related Work

In this section we present existing body of literature that can be related to this dissertation. First, we describe works concerning the problem of link recommendation, *i.e.*, adding to the network edges that maximize specific properties. Second, we outline the sensitivity analysis literature that investigates the robustness of centrality measures to random network changes. Third, we describe works concerning the analysis of the dark networks, *i.e.*, criminal and terrorist organisations. We present classical studies focused on identifying specific characteristics of the dark networks, works that model the activities of the networks using multi-agent systems, as well as literature that uses game theory to analyse covert organizations.

1.3.1 Link Recommendation

Our work, although motivated in a very different fashion, can be seen as an extension of the literature on link recommendation. Link recommendation systems intend to modify some characteristics of the network by adding edges to it [89, 155, 87]. Typical applications of such systems are recommending items to buy based on the history of purchases [130] or recommending future collaborations based on citation networks [120]. Some of the works in this body of literature focus directly on centrality measures.

Our analysis differs from this body of literature in a number of ways. While existing works typically focus solely on increasing a certain statistic, we take into consideration additional features that we want to maintain (*e.g.*, influence in the network in Chapter 3 or particular connections that are forbidden to modify in all chapters). While most works consider only a single network analysis tool, we analyse a range of most popular tools (three different centrality measures, seven different community detection algorithms and nine different link prediction algorithms). Finally, while authors typically assume a complete information setting, we design algorithms that can be used by network members without the knowledge about the structure of the entire network.

Parotsidis *et al.* [115] investigate the way of choosing edges to add to the network in order to maximize expected closeness centrality. The edges are chosen from the set of results of another link recommendation system, where each edge is given the probability of adoption, hence the optimization of *expected* centrality. Motivation of the authors is maximizing the spread of information in the network, *i.e.*, they treat closeness centrality in a similar way to influence measure. Authors find the problem to be NP-hard and they propose a greedy approximation algorithm.

A similar problem is considered by Crescenzi *et al.* [36]. They investigate how a node can improve its standing in the network, again, according to closeness centrality measure, but adding some edges that are incident to it, *i.e.*, by connecting to some new neighbours. Authors prove that the problem is hard to approximate, but they provide a greedy optimization algorithm, which is then evaluated on a number of different networks. The work shows that a node can improve its centrality ranking by adding only a small number of edges to the network.

Other network characteristics that are either minimized or maximized by various algorithms include eigenvalue of the network's adjacency matrix [142, 127], the diameter of the network [16, 53], average length of shortest-path between any two nodes [99, 113, 114], the number of closed triads [41], and eccentricity [118].

1.3.2 Sensitivity Analysis

Viewed from a different perspective, our work can be seen as an extension of the sensitivity analysis of centrality measures [35] and community detection algorithms [110].

However, while such analyses typically consider random network alterations, we focus on the effects of strategic changes. Rather than representing incidental information distortion in data, edge modifications in our case are result of a coordinated effort to falsify the results of the analysis.

Borgatti *et al.* [21] investigate the robustness of centrality measures under different kinds of random network errors, *i.e.*, removal and addition of either nodes or edges. Authors analyse how accuracy of the centrality measures depends on error rate. As it turns out, accuracy declines smoothly with increasing error. Interestingly, there is no notable difference in robustness between all considered centrality measures, *i.e.*, degree, closeness, betweenness, and eigenvector centralities. Another surprising result is that small kinds of network changes result in a very similar decrease in centrality accuracy.

Frantz *et al.* [52] study the impact of network's topology on the robustness of centrality measures in case of random errors. Authors find out that the type of topology has a significant effect on the robustness of centrality measure, far greater than either density of the network or its size. Scale-free and core-peripheral networks turn out to be more robust in terms of accuracy of the centrality measures than small-world and cellular networks.

Karrer *et al.* [71] claim that robustness of a community structure to small perturbations should be a measure of its significance, rather than more widely used modularity. Authors randomly change edges of the network and use the variation of information as a measure of distance between the original network and its modified variant. Using this approach they compare how changing the structure of the network affects results of community detection algorithms. This method enables one to discern between strong community structures and those that are artefacts of the algorithm's nature.

Yang and Leskovec [154] consider a similar setting, but in their analysis they use only real-life networks where the ground truth is known. This way they can evaluate how reliable are certain community detection algorithms in the presence of noise. As it turns out, robustness of an algorithm is highly dependant on the community structure scoring function it uses.

1.3.3 Dark Networks Analysis

We now describe the body of literature related to the analysis of terrorist and criminal networks. We divide it according to methods used in the analysis.

While this body of literature study the same structures as we do, *i.e.*, networks members of which may hope to remain undetected, it treats them in a very different way. Authors of this type of works usually assume that the object of their analysis is static and that it correctly represents underlying reality. On the other hand, we treat members of the network as strategic players, who may change the structure of their organization in order to achieve their goals.

Social Network Analysis Tools

The application of social network analysis techniques, such as centrality measures and community detection algorithms, to criminal organizations is a well-established research theme [137].

Carley [27] points out limitations of traditional social network analysis tools and emphasizes the importance of taking into account the changing structure of the network, the ability of networks to heal themselves, and the ability to act without full information. The author applies this approach to building a model of evolution and destabilization of social networks.

Ressler [124] investigates the ways in which social network analysis can be helpful in understanding *modus operandi* of terrorist organizations. The author notices increased interest in applying the social network analysis tools to terrorist networks after the attacks of 9/11.

Xu and Chen [153] analyse statistically a number of covert networks, including terrorist groups, organizations of drug traffickers, criminals involved in gang-related crimes and hyperlinks between terrorist websites. They gather characteristics such as average path length, clustering coefficient, degree distribution and link density. The results show striking similarities between dark networks and small-world structures.

Paulo *et al.* [116] presents Organization, Relationship, and Contact Analyser (ORCA), software for the analysis of organizations of gang-related criminals. Authors develop an algorithm that identifies the most influential members of the group using the contagion tipping model. The system can also decompose street gangs into components by maximizing modularity and determine the degree to which each individual is a member of a gang.

Multi-agent Systems

Carley *et al.* [29] consider the use of different tools and approaches to destabilizing networks: social network measures, pattern location and multi-agent simulations. Authors addresses the problem of complexity of analysing networks with hundreds of agents. They consider connections between agents not only as a social network, but as a meta-matrix of networks—incorporating knowledge possessed by individuals, goals that they can accomplish and information they have about others. Authors investigate the effects of removal of the network's leader on the redistribution of tasks and changes in agents' hierarchy.

Tsvetovat and Carley [144] present NetWatch—a multi-agent model of covert network surveillance and destabilization. It consists of two networks, representing a terrorist organization and a law enforcement agency. Every task carried out by a terrorist requires some level of knowledge. Terrorists use connections of the network to exchange information, improving their performance. At the same time, law enforcement agents intend to gain the same knowledge by intercepting messages between terrorists. At later stages, they use collected information to destabilize the network by isolating certain nodes. Authors analyse effects of different wiretap strategies on the learning process of law enforcement network and of different destabilization strategies on terrorist network's performance. Results show that wiretapping most central individuals is more effective than attempts to analyse all communication or listening to only a certain number of channels. Surprisingly, isolating gatekeeper agents is not the most efficient strategy of destabilizing network, as it exhibits an emergent healing behaviour. At the same time, removal of the most knowledgeable individuals results in permanent damage.

Tsvetovat and Carley [143] also address a problem of obtaining viable datasets of covert networks. There exists a limited number of open-source databases providing empirical data. Authors propose an algorithm for generating cellular networks of terrorists, as well as generating task and knowledge structures. Networks generated by the algorithm show similar characteristics to what we know about real-life networks [80, 126].

Liu *et al.* [93] develop a dynamic criminal network formation model using data about illegal behaviours of adolescents in the United States. As it turns out, the decision about joining a criminal network and the amount of effort contributed to its activities depend on the closest friends of an individual. The model is used to determine who is the key player, the removal of whom generates biggest reduction in aggregated crime level. Results show that being the key player rarely corresponds to high value of centrality measure or effort put into criminal activities, but rather to the influence on other members of the network.

Spezzano *et al.* [138] create a model of organizational network, taking into account different properties of its members. They analyse repercussions of removing a subset of nodes from the network and assuming their functions by other nodes. Their model allows to predict how network will reorganize itself after removal of some of its members. Authors present a greedy heuristic algorithm for finding the set of vertices that will maximize the decrease in organization's performance. The algorithm is tested on both synthetic and real-life terrorist networks and is able to predict new structure of the network with high accuracy.

Game Theory

Qin *et al.* [121] use a number of game-theoretic and social network analysis methods to analyse the Global Salafi Jihad network. The authors manage to identify leaders and most important members of four different subsets of the network. Authors also introduce the Web structural mining technique to the field. They employ PageRank algorithm [111] to rate importance of terrorists and find local leaders of terrorist organizations.

Enders and Su [43] present two different approaches to modelling terrorist networks. The rational-actor approach states that the whole network carries out rational optimization process, designed to maximize expected utility. The structural approach concentrates on actions of individual members of the group. Authors study how the density of connections in the networks affects efficiency of communication and risk of getting caught. They also predict lowering density of terrorist groups.

Shaikh *et al.* [132] use graph structural mining to analyse terrorist groups and argue that members with high centrality measure should be targeted.

Lindelauf *et al.* [91] analyse how the need of secrecy affects the communication structure of covert networks. They model the optimal network structure on different stages of organization life cycle. The authors show that during early stages of network development in friendly environment all-to-all communication is optimal, while adopting a star network is optimal in a hostile environment. For established organizations acting in hostile environment cellular network structure proved to be optimal solution.

Lindelauf *et al.* [92] introduce a game theoretic approach to identify most important members in terrorist networks. Authors use three different kinds of standard centrality measures—degree centrality, closeness centrality and betweenness centrality—-and present a new game theoretic centrality measure. In this new measure centrality of a node is its Shapley value in a modified version of a connectivity game [7]. With the use of a weighted connectivity measure authors managed to identify the key members of Jemaah Islamiyah involved in Bali bombings and Al Qaeda terrorists involved in 9/11 attacks. Michalak *et al.* [101] propose an improved general algorithm for computing Shapley value of connectivity games, as well as an approximation algorithm that allows to consider networks of much greater size.

1.4 Publications

Most elements of this dissertation are effects of cooperation with other researchers, and are being prepared to be published or have been presented at international conferences and world-class journals.

- Chapter 3 is based on the fragments of article *Hiding Individuals and Communities in a Social Network* [148], co-authored by Marcin Waniek, Tomasz Michalak, Talal Rahwan and Michael Wooldridge. The article is available on arXiv and it is currently in revision in the Nature Human Behaviour journal. Content of that chapter was also presented at Connected Life 2015 conference in Oxford.

- Chapter 4 is based on the paper *On the Construction of Covert Networks* [149], co-authored by Marcin Waniek, Tomasz Michalak, Talal Rahwan and Michael Wooldridge. The work is accepted to the 2017 International Conference on Autonomous Agents & Multiagent Systems (AAMAS 2017) in São Paulo. Its fragment concerning the construction of the captain network (namely Section 4.4) was presented at Connected Life 2015 conference in Oxford.

- Chapter 5 is based on the fragments of article *Hiding Individuals and Communities in a Social Network* [148], mentioned in the first point. It was also presented at Connected Life 2016 conference in Oxford.

- Chapter 6 is based on the article *Hiding Relationships in a Social Network* [147], co-authored by Marcin Waniek, Tomasz Michalak and Talal Rahwan. The article is prepared to be submitted to Artificial Intelligence journal. It will be presented at Third Annual Conference on Network Science and Economics (part of the conference series "Network Science in Economics" organised by Myrna Wooders and Matthew Jackson) in April 2017 at Washington University in St. Louis, as well as at Adversarial Reasoning in Multi-agent Systems (ADVERSE) workshop in May 2017 in São Paulo.

All theorems, lemmas, algorithms and simulation results have been obtained by the author. The ROAM algorithm (presented in Section 3.5.1) and the CTR algorithm (presented in Section 6.5.3) have been designed in cooperation with Dr Talal Rahwan.

Chapter 2

Preliminaries

In this chapter we present some basic notations, problems and concepts that are used throughout the dissertation. We specify the network notation that we use to describe all our theoretic results. Next, we present the random network models and real-life datasets that we test our algorithms on. We also describe the computational complexity concepts that w make use of, including Turing machines, P and NP complexity classes, as well as NP-hardness and NP-completeness. Finally, we list the NP-complete problems that we use in our reductions.

Most of the following chapters also include sections on preliminaries, where we define concepts specific for those chapter.

2.1 Basic Network Notation

Let $\mathbb{G}$ denote the set of all graphs. Because of the considered domain, in the remainder of this dissertation we use the term *network* when referring to a graph. Let $G = (V, E)$ denote a network, where $V = \{v_1, \ldots, v_n\}$ is a set of n nodes and $E \subseteq V \times V$ is a set of edges. By $\bar{E} = V \times V \setminus (\bigcup_{v_i \in V} \{(v_i, v_i)\} \cup E)$ we denote the set of all *non-existing edges*—we refer to them as *non-edges*. We denote the edge between nodes v_i and v_j by (v_i, v_j). In case of an *undirected* network, E is a set of unordered pairs, *i.e.*, we do not discern between edges (v_i, v_j) and (v_j, v_i). Otherwise the network is said to be *directed*, *i.e.*, E is a set of ordered pairs. Unless stated otherwise, in this dissertation we consider undirected networks. We also assume, that networks do not contain self-loops, *i.e.*, $\forall_{v_i \in V} (v_i, v_i) \notin E$, as well as $\forall_{v_i \in V} (v_i, v_i) \notin \bar{E}$.

A path in a network $G = (V, E)$ is an ordered sequence of distinct nodes, $p = \langle v_{i_1}, \ldots, v_{i_k} \rangle$, such that each two consecutive nodes are connected by an edge from E. When referring to a node v_i belonging to a path p, we use the set notation, *i.e.*, $v_i \in p$ means that v_i is present in path p. We consider the length of a path to be the number of edges in that path. We denote the set of all shortest paths between a pair of nodes $v_i, v_j \in V$ by $\Pi_G(v_i, v_j)$. The distance between two nodes $v_i, v_j \in V$, *i.e.*, the length of a shortest path between them, is denoted by $d_G(v_i, v_j)$. Furthermore, an undirected network G is said to be *connected* (*strongly connected* in case of a directed network) if and only if there exists a path between every pair of nodes in G. A directed network G is said to be connected if and only if an undirected network with the same sets of nodes and edges is connected.

We denote the set of *predecessors* of v_i in G by $P_G(v_i)$, *i.e.*, $P_G(v_i) = \{v_j \in V : (v_j, v_i) \in E\}$. On the other hand, we denote the set of *successors* of v_i in G by $S_G(v_i)$, *i.e.*, $S_G(v_i) = \{v_j \in V : (v_i, v_j) \in E\}$. We denote the set of *neighbours* of v_i in G by $N_G(v_i)$, *i.e.*, $N_G(v_i) = P_G(v_i) \cup S_G(v_i)$. Notice that in the case of an undirected network we have that $N_G(v_i) = P_G(v_i) = S_G(v_i)$. We denote by $\delta_G(v_i)$ the number of neighbours (the *degree*) of a node v_i, *i.e.*, $\delta_G(v_i) = |N(v_i)|$. Finally, we denote by $N_G(v_i, v_j)$ the set of common neighbours of nodes v_i and v_j, *i.e.*, $N_G(v_i, v_j) = N_G(v_i) \cap N_G(v_j)$.

To make the notation clearer, we will often denote two arbitrary nodes by v and w, instead of v_i and v_j. Moreover, we will often omit the network itself from the notation whenever it is clear from the context, *e.g.*, by writing $d(v, w)$ instead of $d_G(v, w)$. This applies not only to the notation presented thus far, but also to all future notation.

2.2 Datasets

To test the algorithms introduced in this dissertation, we use both synthetic and real-life networks. As for the former ones, we use the following standard models:

- *Scale-free* networks generated using the Barabási-Albert model [10]. In this model nodes are added to a network one by one. If there are m or less already existing nodes (where m is the parameter of the network creation process), we add edges between the new node and all already existing nodes. Otherwise, we connect the new node with m already existing nodes, with probability proportional to the degrees of these nodes. More formally, the probability $p_{i,j}$ of choosing an already existing node v_j as the one to be connected to the new node v_i in each of m rounds is:

  $$p_{i,j} = \frac{\delta(v_j)}{\sum_{v_k : k < i \wedge v_k \notin N(v_i)} \delta(v_k)}.$$

  We denote such a network by ScaleFree(n, m), where n is the number of nodes and m is the number of edges added with each node. The name comes from the fact that networks created using this model have scale-free degree distribution, *i.e.*, distribution that follows the power-law. Figure 2.1 presents an example of a network generated using the Barabási-Albert model with $n = 10$ and $m = 2$.

- *Small-world* networks generated using the Watts-Strogatz model [150]. The creation process is as follows. We start with nodes forming a ring, each node connected to $\frac{d}{2}$ previous nodes and $\frac{d}{2}$ following nodes. Then each edge is rewired with fixed probability p. A rewired edge (v_i, v_j), where $i < j$, is replaced with an edge (v_i, v_k), where v_k is chosen uniformly at random from nodes in $V \setminus (N(v_i) \cup \{v_i, v_j\})$. We denote such a network by SmallWorld(n, d, p), where n is the number of nodes, d is the average degree and p is the rewiring probability. The name comes from the fact that networks created using this model exhibit small-world properties, such as short average distance between nodes. Figure 2.2 presents an example of a network generated using the Watts-Strogatz model with $n = 10$, $d = 4$ and $p = \frac{1}{4}$.

- *Random graphs* generated using the Erdős-Rényi model [44]. In this model we add an edge between each pair of nodes with some fixed probability, independently of the other edges. We denote such a network by RandomGraph(n, d), where n is the number of nodes and d is the expected average degree, *i.e.*, $d = p(n-1)$, where p is the probability of adding and edge between two nodes. Figure 2.3 presents an example of a network generated using the Erdős-Rényi model with $n = 10$ and $d = 4$.

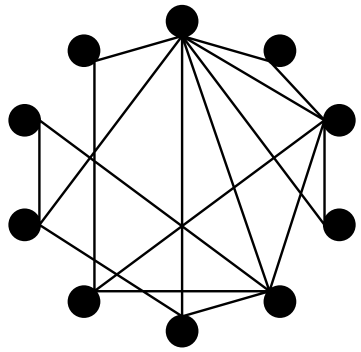

Figure 2.1: An example of a scale-free network.

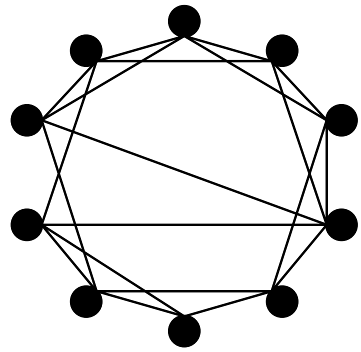

Figure 2.2: An example of a small-world network.

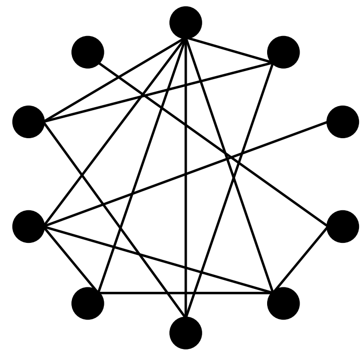

Figure 2.3: An example of a random graph network.

We now describe the real-life datasets used in our experiments:

- Facebook [86]—this group of datasets consists of three fragments of the Facebook social network, containing 61 nodes, 272 edges (small fragment), 333 nodes, 2523 edges (medium fragment) and 786 nodes, 14027 edges (large fragment) respectively. Each anonymised dataset represents what authors call an *ego-network*, a network of connections between all friends of a surveyed Facebook user. Authors intend to detect different *social circles* in the data, but the only information we use in our experiments is the structure of the network.

- Madrid terrorist network [62]—it is the network of terrorists behind the 2004 Madrid bombing, consisting of 70 nodes and 98 edges. The network includes people involved in the organisation of the attack in different ways, *e.g.*, terrorists who actually carried out the bombing, bomb makers, people who sold them the explosives, and financial supporters. Edges in the network represent both strong connections, such as childhood friendships or being involved in previous attacks, and weaker ones, such as casual encounters or financial transactions. The network was created based on the data revealed during the investigation.

- Bali terrorist network [79]—it is the network of terrorists behind the 2002 Bali attack, consisting of 17 nodes and 63 edges. Nodes represent members of the terrorist cell that planned and conducted the attack. Edges map communication channels of the network and represent either phone conversations, e-mail exchanges, or face-to-face conversations.

- WTC terrorist network [80]—it is the network of terrorists behind the 9/11 attacks, consisting of 36 nodes and 64 edges. The network was mapped a few weeks after the attack, based on publicly available information. The set of nodes consists of both 19 hijackers, who were aboard the plane, as well as their known direct contacts, financial supporters and instructors. The connections include history of past meetings, family ties, and common training.

- Zachary's Karate Club [156]—it the social network of participants of a university karate club, consisting of 34 nodes and 78 edges. The data was gathered between year 1970 and 1972 and represents friendship network between club members, *i.e.*, the history of social interactions outside the regular activities of the club. At one point the club fell apart into two separate organizations as a result of the conflict between the main instructor and the club president. We, however, consider the structure of the club as before this fission.

- Les Misérables [78]—it is the network of co-occurrences of characters in Victor Hugo's novel "Les Misérables", consisting of 77 nodes and 254 edges. Two nodes, representing characters from the book, are connected with an edge if and only if they encounter each other in a chapter.

- Greek blogs [157]—it is a network of Greek political blogs, consisting of 142 nodes and 354 edges. The dataset includes blogs discussing the five Greek parliamentary parties, as of November 2010. Edges represent hyperlinks between blogs websites.

2.3 Computational Complexity

We now describe the basic concepts and problems of the computational complexity theory used in this dissertation.

2.3.1 The Turing Machine

The central concept of the computational complexity theory is the *Turing machine* [145]. Turing machine is an abstract machine, capable of performing computation. Formally, Turing machine is a tuple $(Q, \mathbb{A}, \flat, \mathbb{A}_{in}, f, q_0, q_A)$, where:

- Q is a finite set of states;

- $\mathbb{A}$ is a set of alphabet symbols;

- $\flat \in \mathbb{A}$ is the blank symbol;

- $\mathbb{A}_{in} \subseteq \mathbb{A} \setminus \{\flat\}$ is the set of input symbols;

- $f : (Q \setminus \{q_A\}) \times \mathbb{A} \to 2^{Q \times \mathbb{A} \times \{\blacktriangleleft, \blacktriangleright\}}$ is the transition function;

- q_0 is the initial state;

- q_A is the accepting state.

The Turing machine operates on an infinite tape divided into cells. Each cell contains a single symbol from the alphabet $\mathbb{A}$. Initially the tape contains a finite input encoded with the symbols from $\mathbb{A}_{in}$, all other cells of the infinite tape contain blank symbol $\flat$.

The Turing machine posses a head that can read one tape cell at a time, and stores a single state from set Q. Initially, the head is positioned over the first symbol of the input, and stores the initial state q_0.

The Turing machine operates in consecutive, discrete rounds. In each round it performs an action encoded with the transition function f, based on the content of the cell that the head is currently over and its currently stored state. Let $(q', a', d) \in f(q, a)$. In this situation the Turing machine changes the content of the cell that its head is positioned over from a to a', it changes its stored state from q to q', and it shifts its head either one cell to the left (if $d =\blacktriangleleft$), or one cell to the right (if $d =\blacktriangleright$). The Turing machine ends its operation when its stored state is changed to the accepting state q_A. We call the complete run of a Turing machine a *computation*.

We call Turing machine the *deterministic* Turing machine if for every possible situation it encodes exactly one instruction, *i.e.*, $\forall_{q \in Q \setminus \{q_A\}} \forall_{a \in \mathbb{A}} |f(q, a)| = 1$. Otherwise it is called the *non-deterministic* Turing machine and we assume that it chooses the action to perform uniformly at random from set $f(q, a)$. When analysing computations of a non-deterministic Turing machine we consider all possible ways of choosing the next action from set $f(q, a)$.

2.3.2 Complexity Classes and Reduction Process

Turing machines can be used to solve computational problems. We say that the Turing machine M *outputs* symbol a for input x if there exists a computation of M such that after it is finished, the head of the machine is positioned over a cell with symbol a.

Let language L be a set of finite words over the symbols from $\mathbb{A}_{in}$, *i.e.*, $L \subseteq \mathbb{A}_{in}^*$. We say that the Turing machine M, such that its alphabet contains symbols 0 and 1, *i.e.*, $\{0, 1\} \subset \mathbb{A}$, *recognizes* language L if:

- for every $x \in L$ there exists a computation of machine M such that M outputs 1 for input x;
- for every $x \notin L$ machine M outputs 0 for input x for every computation.

Due to the nature of considered questions, we will call L *decision problem* (or simply *problem*) instead of language, and we will say that machine M *solves* problem L, rather than that M recognizes it.

We say that Turing machine M solves problem L *in polynomial time* if there exists a polynomial w such that for any input x there exists a computation of M that ends in a number of rounds smaller than $w(|x|)$, where $|x|$ is the length of input x, *i.e.*, the number of tape cells that it is encoded on.

We say that the decision problem L is in the complexity class P (or that it is simply *in P*) if there exists a deterministic Turing machine M that solves L in polynomial time. Intuitively, those are the problems that we know an effective solution to.

We say that the decision problem L is in the complexity class *NP* (or that it is simply *in NP*) if there exists a non-deterministic Turing machine M that solves L in polynomial time. Intuitively, those are also the problems that we do not know an effective solution to and we check every possible answer to find the solution. Note that $P \subset NP$, although we do not know whether $P = NP$ or not [33] (it is one of the fundamental questions in the computational complexity theory).

We say that problem A can be *reduced* to problem B if there exist a function f that can be computed in polynomial time such that for every input x we have $x \in A \iff f(x) \in B$. Intuitively, A can be reduced to B if we can effectively solve A by using procedure solving B as a subroutine.

We say that the computational problem L is *NP-hard* if any problem L' in NP can be reduced to L. We say that the computational problem L is *NP-complete* if L is NP-hard and L is in NP.

Typical method of proving that a computational problem L in NP-hard is by reducing a known NP-complete problem L' to L. This way any problem in NP can be first reduced to L' and subroutines computing L' can be reduced to L. We use this technique in all computational complexity proofs in this dissertation.

2.3.3 NP-Complete Problems

We now describe well-known NP-complete problems [70] used in the NP-hardness proofs throughout this dissertation:

- *Set Cover* problem—an instance of this problem is defined by a universe $U = \{u_1, \ldots, u_l\}$, a collection of sets $S = \{S_1, \ldots, S_m\}$ such that $\forall_j S_j \subset U$, and an integer $k \leqslant m$. The goal is to determine whether there exist k elements of S the union of which equals U.

- *3-Set Cover* problem—this is the version of Set cover problem where every element of S is of size 3.

- *Finding k-Clique* problem—an instance of this problem is defined by a network $G = (V, E)$ and an integer $k \leqslant n$. The goal is to determine whether there exist k nodes in G that form a clique, *i.e.*, there exists $V' \subseteq V$ such that $|V'| = k$ and $\forall_{v_i, v_j \in V'} (v_i, v_j) \in E$.

- *Finding Hamiltonian Cycle* problem—an instance of this problem is defined by a network $G = (V, E)$. The goal in to determine whether there exists a cycle that visits each node exactly once.

It is worth noting that, even though their NP-completeness, there exist effective approximation algorithms for these problems, *i.e.*, algorithms computing solutions that are close to optimal [67]. Many approximation algorithms use techniques such as linear programming [146], others assume a greedy approach [19].

Some solutions of the network problems are designed specifically for a certain class of networks, *e.g.*, planar networks [9] (intuitively, a planar network is a network that can be drawn in a way such that no edges cross each other).

Chapter 3

Disguising Centrality of a Node

In this chapter we discuss the problem of disguising the centrality of a given single node, while preserving its influence on the network activities. We formally define the problems, investigate their computational complexity, and propose heuristic solutions that can be realistically run for massive social networks.

3.1 Introduction

The on-going process of datafication continues to turn many aspects of our lives into massive amounts of computerised data [97]. This data is being collected and analysed for various applications by both public and private institutions. One particular type of data that has received significant attention over the past decade, mainly due to dynamic growth of social media, concerns the structure of social connections. A number of tools have been advocated for social network analysis, with one of the key problems being the identification of key individuals within a network [20].

Although the process of datafication has undeniably enhanced our lives in various ways [69], it has also given rise to many legitimate privacy concerns. Critics point out that it contributes towards the ever growing digital surveillance of our lives by both business enterprises and public institutions. For instance, Mislove *et al.* [103] demonstrated that by analysing the structure of Facebook's social network, as well as the attributes of its users, it is possible to infer otherwise-private information about other Facebook users. This highlighted that sharing seemingly harmless information may expose substantial knowledge about ourselves.

To tackle such privacy issues, various countermeasures have been proposed, ranging from strict legal regulations [1], through algorithmic solutions [72], to market-like mechanisms that allow participants to monetize their personal information [84]. However, to date, only a few such countermeasures have been implemented, leaving the privacy issue largely unresolved, *e.g.*, as is evident from the very recent release of Facebook's *Global Government Requests Report* [2], which revealed a global increase in government requests to secretly access user data. Through its social mapping program, the US government's National Security Agency (NSA) reportedly created a sophisticated web of social connections of some US citizens [135]. Furthermore, it is unlikely that effective legal mechanisms will be introduced in countries with authoritarian regimes, where social networking sites and other internet content are striclty controlled, and anti-governmental blogs and activities are censored [76, 77].

Against this background, we ask the following question: can individuals proactively manage their social connections so that their privacy is less exposed to the workings of network analysis tools? To put it differently, *can we disguise our standing in the network to escape detection?* This matters because, on one hand, it helps the general public to protect their privacy against intrusion from government and corporate interests, and, on the other hand, it gives counterterrorism units and law-enforcement agencies an insight into how criminals and terrorists could try to escape detection, especially given the increasing reliance of terrorists on social-media survival strategies [109, 68]. To date, however, this fundamental question has received little attention in the literature, as most research efforts have focused on developing ever more sophisticated network analysis tools, without considering how such tools can be evaded. We argue that when assessing the effectiveness of a social network analysis mechanism, one should take into consideration the difficulty of fooling this mechanism.

To address the above question from an individual's viewpoint, we analyse the possible ways of avoiding centrality measure analysis. Centrality measure is a function that, using the structure of the network, evaluates the importance of every node. We focus on three fundamental centrality measures, namely the degree centrality, the closeness centrality, and the betweenness centrality, and study how an individual can avoid being identified as a key player by those measures without compromising her influence. Since, from a graph-theoretic perspective, this is fundamentally an optimization problem, we analyse its computational complexity to study the theoretical limits of one's capability to disguise importance in a social network.

Although we show that an optimal solution is hard to compute, we demonstrate the effectiveness of a surprisingly simple heuristic, whereby the rewiring of social connections is restricted to the individual's immediate network neighbourhood. Specifically, it involves two actions that are already available on popular social-media platforms, namely "unfriending" a certain friend, and introducing two friends to each other. Our results demonstrate, surprisingly, that disguising individuals is in practice possible using a simple, readily implementable heuristic. We quantify the cost-benefit profile of using these heuristics in various empirical networks.

The remainder of this chapter is organized as follows. In Section 3.2 we present formal definitions of centrality measures and influence models. In Section 3.3 we define the problems of Disguising Centrality and Influence Recovery. In Section 3.4 we show the computational complexity of said problems. In Section 3.5 we present our ROAM heuristic that can be used to effectively hide one's importance in a social network, without sacrificing one's influence. In Section 3.6 summarize and discuss our findings.

3.2 Preliminaries

We now formally define three most popular centrality measures, *i.e.*, the degree centrality, the closeness centrality and the betweenness centrality, as well as two influence models, *i.e.*, Independent Cascade and Linear Threshold.

3.2.1 Centrality Measures

The concept of *centrality* in human organizations was introduced by Bavelas [12]. Intuitively, a centrality measure expresses the importance of a node relatively to all other nodes in a network. Formally, it is a function $c : \mathbb{G} \times V \to \mathbb{R}$. Three best-known centrality measures are the *degree centrality*, the *closeness centrality* and the *betweeness centrality* [55].

Degree centrality was introduced by Shaw [133]. It assumes that the importance of a node is proportional to the number of its neighbours. In particular, normalized (to have values between 0 and 1) degree centrality of a node $v_i \in V$ in a network G is defined as follows:

$$c_{degr}(G, v_i) = \frac{\delta(v_i)}{n-1}.$$

Closeness centrality, introduced by Beauchamp [13], quantifies importance of a node in terms of shortest distances from this node to all other nodes in the network. In particular, the most important node is the one with the shortest average path length to all other nodes. Formally, normalized (to have values between 0 and 1) closeness centrality of a node $v_i \in V$ in a (strongly) connected network G can be expressed as:

$$c_{clos}(G, v_i) = \frac{n-1}{\sum_{v_j \in V} d(v_i, v_j)}.$$

Betweenness centrality was developed independently by Anthonisse [8] and Freeman [54]. This measure quantifies the importance of a given node in the context of a network flow. In more detail, if we consider all the shortest paths in the network, then the more such paths traverse through a given node, the greater is the role of that node in the network. Formally, normalized (to have values between 0 and 1) betweenness centrality of a node $v_i \in V$ in a (strongly) connected network G can be expressed as:

$$c_{betw}(G, v_i) = \frac{2}{(n-1)(n-2)} \sum_{v_j, v_k \in V \setminus \{v_i\}} \frac{|\{p \in \Pi(v_j, v_k) : v_i \in p\}|}{|\Pi(v_j, v_k)|}.$$

To illustrate the meaning of centrality measures, consider the network G depicted in Figure 3.1. The degree centrality of node v_3 is $c_{degr}(G, v_3) = \frac{4}{5}$, as it has 4 neighbours out of 5 possible. The closeness centrality of node v_3 is $c_{clos}(G, v_3) = \frac{5}{4 \times 1 + 1 \times 2} = \frac{5}{6}$, as there are 4 nodes in distance 1 to v_3 (namely v_1, v_2, v_4 and v_5), and one node in distance 2 to v_3 (namely v_6). The betweenness centrality of node v_3 is $c_{betw}(G, v_3) = \frac{2}{5 \times 4}(6 \times 1 + 1 \times \frac{1}{2}) = \frac{13}{20}$, as v_3 controls all shortest paths between any node in $\{v_1, v_2\}$ and any node in $\{v_3, v_5, v_6\}$ (there are 6 such pairs), as well as one of two shortest paths between v_4 and v_5 (the other shortest path being controlled by v_6). Node v_3 is the most central node according to all three centrality measures.

3.2.2 Models of Influence

The propagation of influence through the network is often described in terms of node activation. When a certain node is sufficiently influenced by its neighbours, it becomes "active". It then starts to influence any "inactive" neighbours, and so on. To initiate this

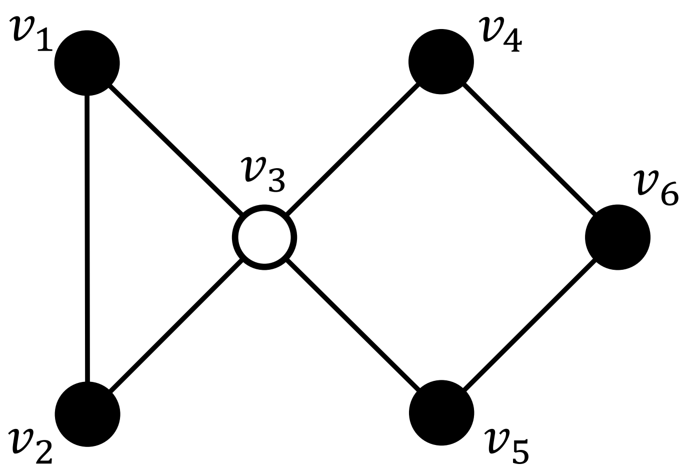

Figure 3.1: A sample network to illustrate centrality measures.

propagation process, a set of nodes (*seed set*) needs to be activated right from the start. Assuming that time is discrete, we denote by $I(t) \subseteq V$ the set of nodes that are active at round t, implying that $I(1)$ is the seed set. The way influence propagates to inactive nodes depends on the influence model under consideration. The two main models of influence are:

- *Independent Cascade* [58]: In this model, every pair of nodes is assigned an activation probability, $p : V \times V \rightarrow [0, 1]$. Then, in every round, $t > 1$, every node $v_i \in V$ that became active in round $t-1$ activates every inactive successor, $v_j \in S(v_i) \setminus I(t-1)$, with probability $p(v_i, v_j)$. The process ends when there are no new active nodes, *i.e.*, when $I(t) = I(t-1)$.

  The intention of this model is to resemble the process of spreading rumour in a social network. Each member of the network passes the news with probability depending on how often she contacts with others. However, after some time the rumour becomes outdated and the process stops.

- *Linear Threshold* [73]: In this model, every node $v_i \in V$ is assigned a *threshold value* t_{v_i} which is sampled (according to some probability distribution) from the set $\{0, \ldots, |P(v_i)|\}$. Then, in every round, $t > 1$, every inactive node v_i becomes active, *i.e.*, becomes a member of $I(t)$, if the number of her active predecessors meets or exceeds her threshold, *i.e.*, if $|I(t-1) \cap P(v_i)| \geqslant t_{v_i}$. The process ends when there are no new active nodes, *i.e.*, when $I(t) = I(t-1)$.

  The intuition behind this model is the analysis of situations where costs and benefits of adopting a particular behaviour depends on how many other members of the network adopt it [59], *e.g.*, people who smoke cigarettes just because their friends and co-workers do so.

In either model, the influence of a node v_i on another node v_j is denoted by $\imath_G(v_i, v_j)$ and is defined as the probability that v_j gets activated given the seed set is $\{v_i\}$. We assume that $\imath_G(v_i, v_i) = 0$ for all $v_i \in V$. The influence of v_i over the entire network G is then defined as $\imath_G(v_i) = \sum_{v_j \in V} \imath_G(v_i, v_j)$.

3.3 Problem Definitions

The goal of the disguising process is to hide the importance of the chosen *source node*, $v^\dagger$, without sacrificing its influence on the network. To achieve this goal we either add or remove edges from the network. We assume a limited budget, b, specifying the maximum number of edges that are allowed to be added or removed. To make the process more tractable, we divide it into two separate steps.

In the first step we rewire the network so as to minimize each centrality score of $v^\dagger$ separately. In the second step we attempt to compensate for any influence that $v^\dagger$ might have lost during the centrality minimization phase. We consider two variants of the Influence Recovery problem, the first variant focuses on the influence of $v^\dagger$ on every single node separately, whereas the second focuses on the aggregated influence of $v^\dagger$ over the network.

We also introduce two additional elements, the set $\hat{A}$ of edges allowed to be added and the set $\hat{R}$ of edges allowed to be removed. Their purpose is twofold. Firstly, we use them to avoid overwriting changes made by the previous steps, *e.g.*, adding during influence recovery the same edge that was previously removed during the centrality minimization step. Secondly, we consider the possibility that some edges are impossible to add or remove due to objective reasons, *e.g.*, we want to preserve a particular communication structure within the network or we want to avoid adding edges—or cannot add edges—between certain individuals.

We now formulate the formal definitions of our computational problems.

Definition 1 (Disguising Centrality)**.** *The problem is defined by a tuple* $(G, v^\dagger, b, c, \hat{R}, \hat{A})$*, where* $G = (V, E) \in \mathbb{G}$ *is a network,* $v^\dagger \in V$ *is the source node (whose centrality is to be minimized),* $b \in \mathbb{N}$ *is the budget specifying the maximal number of edges that can be added or removed,* $c : \mathbb{G} \times V \to \mathbb{R}$ *is a centrality measure,* $\hat{R} \subseteq E$ *is the set of edges allowed to be removed, and* $\hat{A} \subseteq \bar{E}$ *is the set of edges allowed to be added. The goal is then to identify two sets of edges,* $R^* \subseteq \hat{R}$ *and* $A^* \subseteq \hat{A}$*, such that:* $|A^*| + |R^*| \leqslant b$ *and* $G^* = (V, (E \cup A^*) \setminus R^*)$ *is connected (strongly connected if* G *is directed) and* G^* *is in:*

$$\underset{G' \in \left\{(V,(E\cup A)\setminus R) : R\subseteq \hat{R}, A\subseteq \hat{A}, |A|+|R|\leqslant b\right\}}{\arg\min} c(G', v^\dagger).$$

The intuition behind this problem is as follows. We try to find the sets of edges to add and remove from the network (taking into consideration restrictions represented by sets $\hat{A}$ and $\hat{R}$, as well as limited budget) such that the chosen centrality measure c gets as low as possible. We repeat this process for each centrality that we wish to lower, excluding just removed edges from set $\hat{A}$ and excluding just added edges from the set $\hat{R}$, to avoid overwriting our changes. Each centrality has its own budget, and after they are spent, we wish to use whatever budget we have left to rebuild our influence on the network. To this end, we define the following problems.

Definition 2 (Global Influence Recovery)**.** *This problem is defined by a tuple* $(G, v^\dagger, \inf, \hat{R}, \hat{A}, \Phi)$*, where* $G = (V, E) \in \mathbb{G}$ *is a network,* $v^\dagger \in V$ *is the source node (whose influence is to be recovered),* $\inf : V \times V \to \mathbb{R}$ *is an influence measure,* $\hat{R} \subseteq E$ *is the set of edges allowed to be removed,* $\hat{A} \subseteq \bar{E}$ *is the set of edges allowed to be added, and* $\Phi \in \mathbb{R}$ *is the total influence to be recovered. The goal is then to identify two sets of edges,*

$R^* \subseteq \hat{R}$ and $A^* \subseteq \hat{A}$, such that $G^* = (V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and G^* is in:

$$\underset{G' \in \left\{(V,(E\cup A)\setminus R): R\subseteq \hat{R}, A\subseteq \hat{A}, \inf_{G'}(v^\dagger) \geqslant \Phi\right\}}{\arg\min} |A| + |R|.$$

In this problem we intend to rebuild the aggregated influence on the network that the source node might have lost during the process of lowering centrality. Note that this time we do not specify the budget, but rather we try to find the smallest set of edges that allows to regain the influence.

Finally, we present an alternative version of the problem, where we do not rebuild the total influence over the entire network, but rather where we aim to regain the previous level of influence on each separate node.

Definition 3 (Individual Influence Recovery)**.** *This problem is defined by a tuple $(G, v^\dagger, \inf, \hat{R}, \hat{A}, \phi)$, where $G = (V, E) \in \mathbb{G}$ is a network, $v^\dagger \in V$ is the source node (whose influence is to be recovered), $\inf : V \times V \to \mathbb{R}$ is an influence measure, $\hat{R} \subseteq E$ is the set of edges allowed to be removed, $\hat{A} \subseteq \bar{E}$ is the set of edges allowed to be added, and $\phi : V \to \mathbb{R}$ specifies the influences to be recovered (i.e., for every $v_i \in V$ we want the influence of $v^\dagger$ over v_i to be at least $\phi(v_i)$). The goal is then to identify two sets of edges, $R^* \subseteq \hat{R}$ and $A^* \subseteq \hat{A}$, such that $G^* = (V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected, if G is directed) and G^* is in:*

$$\underset{G' \in \left\{(V,(E\cup A)\setminus R): R\subseteq \hat{R}, A\subseteq \hat{A}, \forall_{v_i\in V} \inf_{G'}(v^\dagger, v_i) \geqslant \phi(v_i)\right\}}{\arg\min} |A| + |R|.$$

Note that for both the Independent Cascade and the Linear Threshold influence models, the only way to increase a node's influence is to add edges to the network, *i.e.*, removing edges always lowers influence. Nonetheless, we allow for the removal of edges as some other models of influence might not have this property.

We now move to the complexity analysis of the just defined problems.

3.4 Complexity Analysis

Our findings regarding the complexity of the problems defined in the previous section are summarized in Table 3.1.

From the computational point of view, disguising the degree centrality of $v^\dagger$ is easy. The only way to decrease this centrality is to remove edges connecting $v^\dagger$ to its neighbours. The decrease in centrality is only affected by the number of removed edges, not by their choice. Hence, the problem of Disguising Degree Centrality is clearly in P.

Next, we study the problems of Disguising Closeness Centrality and Disguising Betweenness Centrality, followed by the problem of Influence Recovery under the Independent Cascade model and under the Linear Threshold model. As it turns out, all of these problems are either NP-complete or NP-hard.

Theorem 1. *Disguising Closeness Centrality is NP-complete.*

Problem	Complexity
Disguising Centrality (Degree)	P
Disguising Centrality (Closeness)	NP-complete
Disguising Centrality (Betweenness)	NP-complete
Individual Influence Recovery (IC)	NP-hard
Individual Influence Recovery (LT)	NP-hard
Global Influence Recovery (IC)	NP-hard
Global Influence Recovery (LT)	NP-hard

Table 3.1: Summary of our computational hardness results, where LT denotes Linear Threshold influence model and IC denotes Independent Cascade influence model.

Proof. The decision version of the optimization problem is the following: given a network $G = (V, E)$, a source node $v^{\dagger}$, two sets $\hat{R} \subseteq E$, $\hat{A} \subseteq \bar{E}$, a budget $b \in \mathbb{N}$ and a value $x \in \mathbb{R}$, does there exist two sets, $R^* \subseteq \hat{R}$ and $A^* \subseteq \hat{A}$ such that $|A^*| + |R^*| \leqslant b$, and the network $(V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and $c_{clos}((V, (E \cup A^*) \setminus R^*), v^{\dagger}) \leqslant x$.

This problem clearly is in NP, as given a solution, *i.e.*, two sets A^* and R^*, we can verify whether $|A^*|+|R^*| \leqslant b$ and $c_{clos}((V, (E \cup A^*) \setminus R^*), v^{\dagger}) \leqslant x$ in polynomial time. This only requires computing the closeness centrality of node $v^{\dagger}$ in network $(V, (E \cup A^*) \setminus R^*)$.

The main idea of the NP-hardness proof is as follows. We show reduction from the NP-complete Finding Hamiltonian Cycle problem. We modify the network from the Finding Hamiltonian Cycle problem instance and use it as an input for the minimizing closeness centrality problem. We know that the optimal network structure in terms of minimizing closeness centrality is a path for undirected networks and a cycle for directed networks. From such a solution to minimizing closeness centrality we can easily obtain a Hamiltonian cycle in the original network.

We will now show that the decision version of our problem is NP-hard. To this end, let us denote by $q \in \mathbb{R}$ the smallest possible closeness centrality of $v^{\dagger}$ in any (strongly) connected network whose set of nodes is V. Let us also denote by Q the network with the smallest possible number of edges such that the closeness centrality of $v^{\dagger}$ in Q is q. One can see that $q = \frac{2}{n}$ in the case of an undirected network (Q is then a path of which $v^{\dagger}$ is an end), and $q = (\sum_{i=1}^{n-1} \frac{1}{i})/(n-1)$ in the case of a directed network (Q is then a directed cycle).

Let us comment on the structure of Q. Clearly, in order to achieve the lowest possible closeness centrality, the spanning tree of undirected Q rooted in $v^{\dagger}$ has to be a path. Any other edge added to Q creates a shorter path from $v^{\dagger}$ to at least one other node, hence Q must be a path in the undirected case. In the directed case the argument is similar, although to ensure strong connectivity it is necessary to add an edge between end of the path and $v^{\dagger}$.

With this in mind, the proof involves a reduction from the Finding Hamiltonian Cycle problem, as defined in Section 2.3.3 (*i.e.*, the problem of determining whether there exists a cycle that visits each node exactly once) to the decision problem of determining whether it is possible to reduce the closeness centrality of $v^{\dagger}$ to a value smaller than, or equal to q.

To this end, given some arbitrary network, $G' = (V', E')$, be it directed or undirected, let us modify G' so as to obtain a new network, $G = (V, E)$, as illustrated in Figures 3.2 and 3.3. Formally, we do so by choosing some arbitrary node, $v' \in V'$, and then setting:

$$V = V' \cup \{v^{\dagger}, w_1, w_2\}, \text{and } E = E' \cup \{(v^{\dagger}, v'), (w_1, w_2)\} \cup \{(v, w_1) : v \in N_{G'}(v')\}$$

in the case of an undirected network, or setting:

$$V = V' \cup \{v^{\dagger}, w_1\}, \text{and } E = E' \cup \{(v^{\dagger}, v'), (w_1, v^{\dagger})\} \cup \{(v, w_1) : v \in P_{G'}(v')\}$$

in the case of a directed network.

We will now show that the Finding Hamiltonian Cycle problem in G' is equivalent to the following decision problem: Given network G and budget $b = |E'| - |V'| + |P_{G'}(v')|$, where $\hat{A} = \emptyset$ and $\hat{R} = E$, determine whether it is possible to reduce the closeness centrality of $v^{\dagger}$ to a value lesser than or equal to q, by removing at most b edges from G. Throughout the remainder of the proof, the edges and nodes in G that were present in G' will be referred to as *original*.

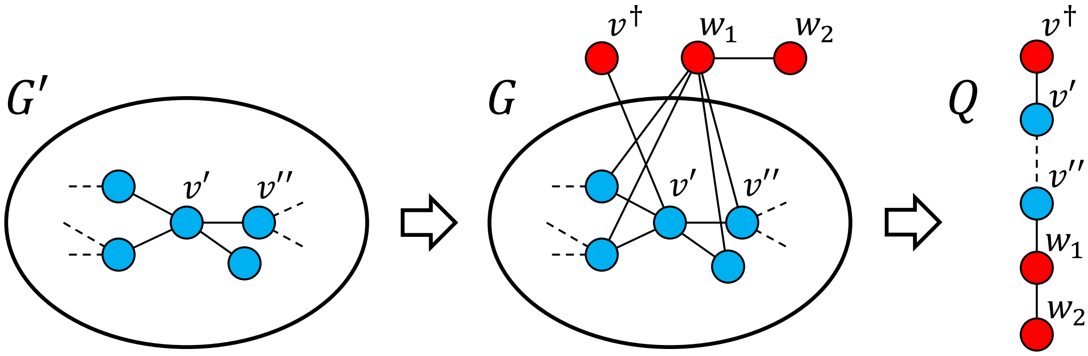

Figure 3.2: The main steps of reducing the Finding Hamiltonian Cycle problem to the Disguising Closeness Centrality problem in an undirected network.

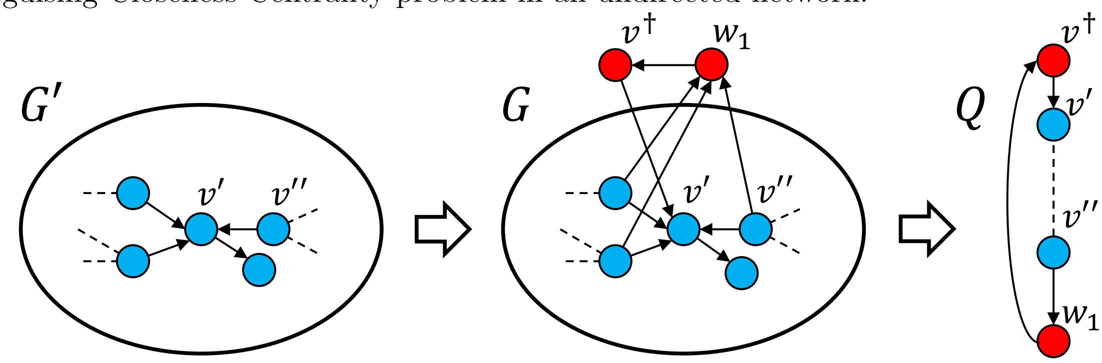

Figure 3.3: The main steps of reducing the Finding Hamiltonian Cycle problem to the Disguising Closeness Centrality problem in a directed network.

Firstly, we will show that if G' has a Hamiltonian cycle then it is possible to obtain Q by removing $|E'| - |V'| + |P_{G'}(v')|$ edges from G. To this end, fix a Hamiltonian cycle of G', then:

- remove from G all original edges that are not in the Hamiltonian cycle; there are exactly $|E'| - |V'|$ such edges;

- in the Hamiltonian cycle, there are exactly two edges of which v' is an end; remove any of those edges in the undirected network, or the one pointing to v' in the directed network; let us denote the removed edge as (v'', v');

- remove all edges from all predecessors of v' to w_1, with the exception of (v'', w_1); there are exactly $|P_{G'}(v')| - 1$ such edges.

In so doing, we have obtained the network Q by removing a total of $|E'| - |V'| + |P_{G'}(v')|$ edges from G (see figures 3.2 and 3.3).

Secondly, we show that if it is possible to obtain Q by removing $|E'| - |V'| + |P_{G'}(v')|$ edges from G, then there exists a Hamiltonian cycle in G'. We will first deal with the undirected case, before moving on to the directed one.

In the undirected case, observe that nodes $v^\dagger$ and w_2 each have a degree of 1 in G, since their only neighbours are v' and w_1, respectively. Now, since Q must be connected, and since we obtained Q by only removing (rather than adding) edges from G, the nodes $v^\dagger$ and w_2 must each have degree equal to 1 in Q. Consequently, they must be the two ends of Q. This, in turn, implies that since w_1 must have exactly two neighbours in Q and one of them is w_2, the other one must be one of the original nodes, let us call it v''. This, as well as the fact that $v^\dagger$ is the only node connected to v', implies that the segment of Q between v' and v'' contains all the original nodes from G' and only original edges from G' (recall that we did not add any edges between the original nodes). Finally, by adding to that segment the original edge between v'' and v' (that was present in the original network G', but was removed during the creation of G), we obtain a Hamiltonian cycle in G'.

As for the directed case, we observe that node $v^\dagger$ has only one successor in Q, namely v', and only one predecessor in Q, namely w_1. We also know that w_1 has only one predecessor in Q, let us call that predecessor v''. These facts imply that the segment of Q between v' and v'' contains all original nodes from G' and only original edges from G' (again, recall that we did not add any edges between the original nodes). By adding to that segment the original edge between v'' and v' (that was present in the original network G', but was removed during the creation of G), we obtain a Hamiltonian cycle in G'.

We have shown that a Hamiltonian cycle in G' exists if and only if it is possible to reduce the closeness centrality of $v^\dagger$ to q by removing exactly $|E'| - |V'| + |P_{G'}(v')|$ edges from G, which concludes the proof. □

Notice that the proof takes advantage of the fact that the optimal structure in terms of minimizing closeness centrality is either a path (in case of an undirected network) or a cycle (in case of a directed network). Due to this, we are able to use the network being part of a Finding Hamiltonian Cycle problem instance with only minimal modifications.

Having proven the NP-completeness of Disguising Closeness Centrality problem, we move to the proof for Disguising Betweenness Centrality. Both it and the following proofs in this section use a slightly different approach. Instead of performing minor changes on a given network, we encode the structure of a Set Cover problem instance in an entirely new network and take advantage of the correspondence between optimal solutions to our problem and to the problem of Set Cover.

Theorem 2. *Disguising Betweenness Centrality is NP-complete.*

Proof. The decision version of the optimization problem is the following: given a network $G = (V, E)$, a source node $v^\dagger$, two sets $\hat{R} \subseteq E$, $\hat{A} \subseteq \bar{E}$, a budget $b \in \mathbb{N}$ and a value $x \in \mathbb{R}$, does there exist two sets $R^* \subseteq \hat{R}$ and $A^* \subseteq \hat{A}$ such that $|A^*| + |R^*| \leqslant b$, and the network $(V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and $c_{betw}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leqslant x$.

This problem clearly is in NP, as given a solution, *i.e.*, two sets A^* and R^*, we can verify whether $|A^*|+|R^*| \leqslant b$ and $c_{betw}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leqslant x$ in polynomial time. This only requires computing the betweenness centrality of node $v^\dagger$ in network $(V, (E \cup A^*) \setminus R^*)$.

The main idea of the NP-hardness proof is as follows. We will show a reduction from the NP-complete Set Cover problem. We build a network that reflects the structure of a given Set Cover problem instance and use it as an input for the minimizing betweenness centrality problem. We allow the addition of edges that corresponds to choosing sets in the Set Cover problem instance. Finally, we show that the optimal solution of the minimizing betweenness centrality problem corresponds to the solution of the given instance of the Set Cover problem.

We will now show that the decision version is NP-hard. To this end, we give a reduction from the NP-complete Set Cover problem, as defined in Section 2.3.3. To remind the reader, $U = \{u_1, \ldots, u_l\}$ denotes the universe, $S = \{S_1, \ldots, S_m\}$ denotes the set of subsets of the universe, and the goal is to determine whether there exist k elements of S the union of which equals U.

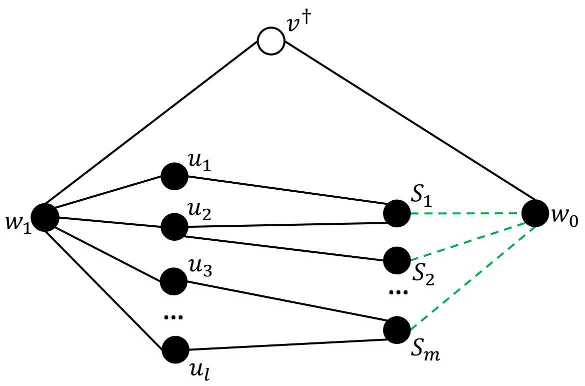

Figure 3.4: An undirected network used to reduce the Set Cover problem to Disguising Betweenness Centrality of $v^\dagger$. To solve both problems, we consider adding (some of) the dashed (green) edges.

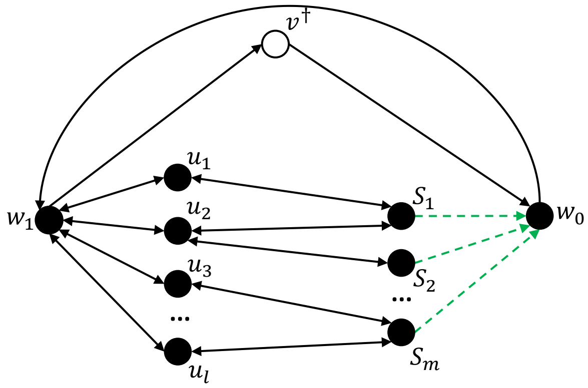

Figure 3.5: A directed network used to reduce the Set Cover problem to Disguising Betweenness Centrality of $v^\dagger$. To solve both problems, we consider adding (some of) the dashed (green) edges.

First, let us create a network G as shown in figures 3.4 and 3.5. More specifically, we create one node for every $S_j \in S$, one node for every $u_i \in U$, and three additional nodes, $v^\dagger$, w_0 and w_1. Next, we add (either undirected or directed) edges as follows. We add the edges $(v^\dagger, w_0)$, $(w_1, v^\dagger)$, and for every node $u_i \in U$ we add an edge (u_i, w_1), as well as the edges (S_j, u_i) for every S_j where $u_i \in S_j$. In case of a directed network, we also add the edges (w_1, u_i) for $u_i \in U$ and edges (u_i, S_j) for every $u_i \in S_j$, as well as the edge (w_0, w_1).

Now, consider the problem of Disguising Betweenness Centrality of $v^\dagger$ in G given $\hat{R} = \emptyset$, $\hat{A} = \{(S_1, w_0), \ldots, (S_m, w_0)\}$, and budget $b = k$. Note that $v^\dagger$ *controls* (*i.e.*, lays on) every shortest path to w_0, and does not control any shortest path between any other pair of nodes. As such, to minimize the betweenness centrality of $v^{\dagger}$, we need to create alternative shortest paths to w_0. This should be done by adding (some of) the edges in $\{(S_1, w_0), \ldots, (S_m, w_0)\}$, since no other edge can be added, and no edge can be removed (following the definitions of $\hat{R}$ and $\hat{A}$). To be more precise, we can add at most k of edges $\{(S_1, w_0), \ldots, (S_m, w_0)\}$, since we cannot exceed the budget. After this process, the betweenness centrality of $v^{\dagger}$ may drop to as low a value as $q = \frac{2}{(n-1)(n-2)}$ in the undirected case (where $v^{\dagger}$ controls shortest paths between w_1 and w_0), or $q = \frac{1}{(n-1)(n-2)}$ in the directed case (where $v^{\dagger}$ controls the shortest path from w_1 to w_0). This happens when $v^{\dagger}$ no longer controls any of the shortest paths to w_0 except for the one from w_1 to w_0. Note that adding an edge (S_j, w_0) creates a new shortest path from every node $u_i \in S_j$ to w_0, as well as a new shortest path from every node $S_{j'}$ such that $u_i \in S_{j'}$ to w_0 (either directly to w_0 in case of S_j, or running through u_i and S_j in case of other $S_{j'}$ such that $u_i \in S_{j'}$). This implies that the betweenness centrality of $v^{\dagger}$ can be reduced to q if and only if we create such a new shortest path to w_0 for every $u_i \in U$, *i.e.*, if there exists at most k elements of S the union of which equals U. This concludes the proof. □

Having proven the NP-completeness of Disguising Betweenness Centrality problem, we now move to the proof of NP-hardness of Influence Recovery problem under the Independent Cascade model.

Interestingly, while we are able to prove NP-completeness of the problems of Disguising Closeness Centrality and Disguising Betweenness Centrality, in case of the Influence Recovery problems we are only able to prove NP-hardness, *i.e.*, we do not show that they are in NP. The reason is that a solution to any of the Disguising Centrality problems is easy to verify, *i.e.*, given a solution we are able to confirm that it is correct by performing a polynomial computation. At the same time, however, verifying a solution to any of the Influence Recovery problems requires an exponential time to perform. Intuitively, this is because the influence in a network can spread via exponential number of different paths and thus even computing the value of influence of one node on another is an exponential-time task [73].

Theorem 3. *Both Global and Individual Influence Recovery problems are NP-hard under the Independent Cascade model.*

Proof. The main idea of the NP-hardness proof is as follows. We will show a reduction from the NP-complete Set Cover problem. We built a network that reflects the structure of a given Set Cover problem instance and use it as an input for the Influence Recovery problem with the Independent Cascade model. We allow the addition of edges that corresponds to choosing sets in the Set Cover problem instance. Finally, we show that the optimal solution of the Influence Recovery problem corresponds to the solution of the given instance of the Set Cover problem.

We show a reduction from the NP-complete Set Cover problem, as defined in Section 2.3.3. To remind the reader, $U = \{u_1, \ldots, u_l\}$ denotes the universe, $S = \{S_1, \ldots, S_m\}$ denotes the set of subsets of the universe, and the goal is to determine whether there exist k elements of S the union of which equals U.

To this end, let us create a network G as illustrated in figures 3.6 and 3.7. In more detail, we start by creating one node for every $S_j \in S$, one node for every $u_i \in U$, and one additional node $v^{\dagger}$. After that, for every $S_j \in S$ and every $u_i \in S_j$, we add the edge

(S_j, u_i) (either directed or undirected). In the directed case we additionally add an edge $(u_i, v^\dagger)$ for every $u_i \in U$ to ensure strong connectivity.

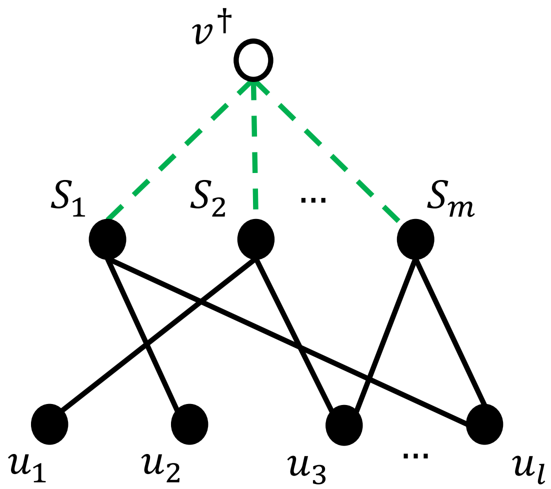

Figure 3.6: Undirected network used to reduce the Set Cover problem to the Independent Cascade Influence Recovery problem. To solve both problems, we consider adding (some of) the dashed (green) edges.

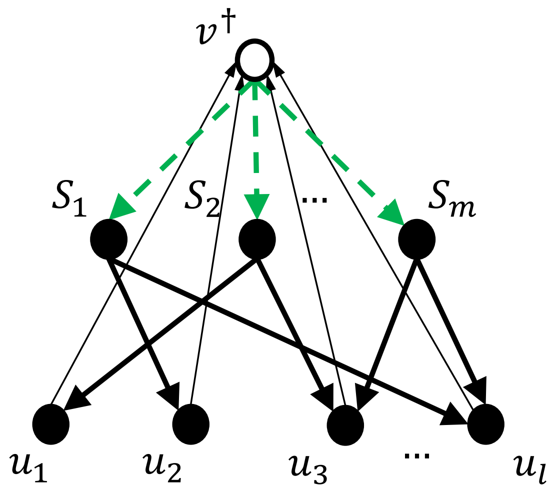

Figure 3.7: A directed network used to reduce the Set Cover problem to the Influence Recovery problem. To solve both problems, we consider adding (some of) the dashed (green) edges.

Consider the Influence Recovery problem in G under the Independent Cascade model, where:

- $\hat{R} = \emptyset$;
- $\hat{A} = \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$;
- $p : V \times V \to [0, 1]$ such that $\forall_{S_j \in S}\ p(v^\dagger, S_j) = 1$ and $\forall_{S_j \in S} \forall_{u_i \in S_j}\ p(S_j, u_i) = 1$, and $p(v, w) = 0$ for every other pair of nodes;
- for Individual Influence Recovery, $\forall_{u_i \in U} \phi(u_i) = 1$ and $\phi(v) = 0$ for every other node;
- for Global Influence Recovery, $\Phi = k + l$.

The goal is then to identify the smallest subset of edges to be added to the network, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that either $\inf_{(V,E \cup A)}(v^\dagger) \geqslant \Phi$ in the global variant of the problem, or $\forall_{v \in V} \inf_{(V,E \cup A)}(v^\dagger, v) \geqslant \phi(v)$ in the individual variant of the problem.

Recall that the influence of $v^\dagger$ is measured by setting the seed set as $\{v^\dagger\}$ and calculating the probability that other nodes get activated. Also recall that under the Independent Cascade model an active node, v, activates any of its successors, w, with probability $p(v, w)$. Importantly, with the p function defined as above, adding an edge $(v^\dagger, S_j)$ for some $S_i \in S$ makes the influence of $v^\dagger$ on every $u_i \in S_j$ equal to 1. Furthermore, the above definitions of Φ and ϕ imply that our goal (in both the individual and the global variants of the problem) is achieved if and only if the influence of $v^\dagger$ on every node $u_i \in U$

equals 1. Consequently, our goal is achieved if and only if we add to G a set of edges, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that:

$$\bigcup_{(v^\dagger, S_j) \in A} S_j = U.$$

Since we are interested in finding the smallest such subset, a solution to the above instance of the Influence Recovery problem gives us a solution to the Set Cover problem. □

Having proven NP-hardness of the Influence Recovery problem under the Independent Cascade model, we now move to the Linear Threshold model.

Interestingly, in case of directed networks, we use the exactly same network as in the proof for Independent Cascade influence model. However, in case of undirected networks we have to modify the network structure. This is because for the Independent Cascade model we are able to prevent "backward" activation of nodes S_j as a result of the activation of nodes u_i by setting the value of $p(S_j, u_i)$ to 0. In case of the Linear Threshold model we have to use a "gadget" (consisting of nodes T_i and $s_{i,j}$) in order to achieve the same result.

Theorem 4. *Both Global and Individual Influence Recovery problems are NP-hard under the Linear Threshold model.*

Proof. The main idea of the NP-hardness proof is as follows. We will show a reduction from the NP-complete Set Cover problem. We built a network that reflects the structure of a given Set Cover problem instance and use it as an input for the Influence Recovery problem for the Linear Threshold model. We allow the addition of edges that corresponds to choosing sets in the Set Cover problem instance. Finally, we show that the optimal solution of the Influence Recovery problem corresponds to the solution of the given instance of the Set Cover problem.

We show a reduction from the NP-complete Set Cover problem, as defined in Section 2.3.3. To remind the reader, $U = \{u_1, \ldots, u_l\}$ denotes the universe, $S = \{S_1, \ldots, S_m\}$ denotes the set of subsets of the universe, and the goal is to determine whether there exist k elements of S the union of which equals U.

For the directed case, we create a network G as illustrated earlier in Figure 3.7 and described in the proof of Theorem 3. As for the undirected case, we create G as illustrated in Figure 3.8. In more detail, for every $S_j \in S$, we create two nodes, namely S_j and T_j, as well as l additional nodes, namely $s_{j,1}, \ldots, s_{j,l}$. We also create one node for every $u_i \in U$, and finally add the source node, $v^\dagger$. As for the edges, for every $S_j \in S$ and every $u_i \in S_j$, we add the edge (T_j, u_i). Furthermore, for every node $s_{j,i}$, we add the edges $(S_j, s_{j,i})$ and $(s_{j,i}, T_j)$. The purpose of adding nodes T_i and $s_{i,j}$ is to avoid "backward" activation of nodes S_j (*i.e.*, activation of S_i by nodes in U), as would happen in the network shown in Figure 3.6. Note that if there exists a node S_j connected to all nodes in U, it can still be "backward" activated, but only in case when we restore influence on all nodes in U.

Now consider the Influence Recovery problem in G under the Linear Threshold model, where:

- $\hat{R} = \emptyset$;
- $\hat{A} = \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$;

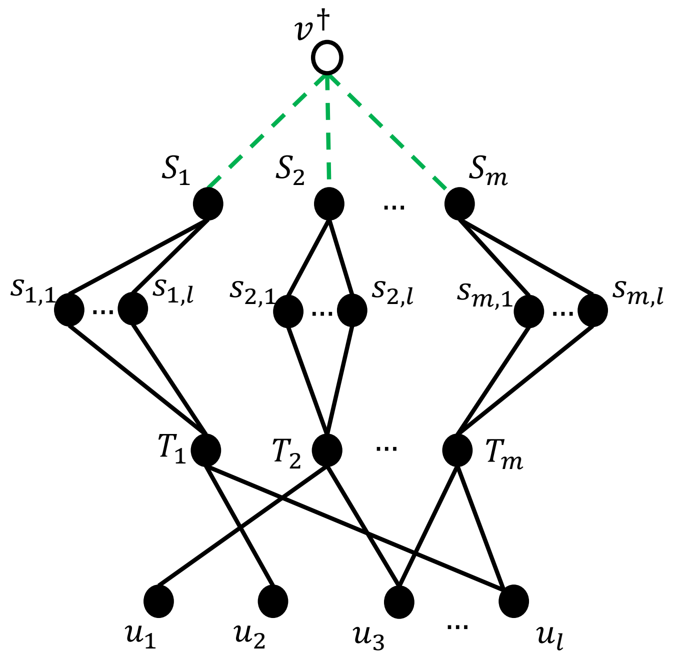

Figure 3.8: Undirected network used to reduce the Set Cover problem to Influence Recovery problem under the Linear Threshold model. To solve both problems, we consider adding (some of) the green dashed edges.

- $t_v = l$ for every node $v \in \{T_1, \ldots, T_m\}$ and $t_v = 1$ for every other node;
- for Individual Influence Recovery, $\forall_{u_i \in U} \phi(u_i) = 1$ and $\phi(v) = 0$ for every other node;
- for Global Influence Recovery, $\Phi = k + l$ for the directed case, and $\Phi = k(l+2) + l$ for the undirected case.

The goal is then to identify the smallest subset of edges to be added to the network, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that either $\inf_{(V,E\cup A)}(v^\dagger) \geqslant \Phi$ in the global variant of the problem, or $\forall_{v\in V} \inf_{(V,E\cup A)}(v^\dagger, v) \geqslant \phi(v)$ in the individual variant of the problem.

Recall that the influence of $v^\dagger$ is measured by setting the seed set as $\{v^\dagger\}$ and calculating the probability that other nodes get activated. Also recall that under the Linear Threshold model a node, v, gets activated if the number of its active predecessors exceeds t_v. Note that, with t_v defined as above, adding an edge $(v^\dagger, S_j)$ in the undirected case leads to the activation of nodes $s_{i,j}$ and T_i, which in turn leads to the activation of every $u_i \in S_j$ (see Figure 3.8). Likewise, in the directed case, adding $(v^\dagger, S_j)$ leads to the activation of every $u_i \in S_j$ (see Figure 3.7). To put it differently, when adding $(v^\dagger, S_j)$, the influence of $v^\dagger$ on every $u_i \in S_j$ equals 1. Importantly, the above definitions of Φ and ϕ imply that our goal (in both the individual and the global variants of the problem) is achieved if and only if the influence of $v^\dagger$ on every node $u_i \in U$ equals 1. Those observations imply that our goal is achieved if and only if we add to G a set of

Algorithm 1 The ROAM heuristic

Input: A network $G = (V, E)$, budget $b \in \mathbb{N}$, source node $v^\dagger \in V$
Output: Sets of edges to be added A^* and to be removed R^* from the network
    Choose $v^* \in N(v^\dagger)$
    $R^* = \{(v^\dagger, v^*)\}$
    $A^* = \emptyset$
    for $i = 1, \ldots, b-1$ **do**
        Choose $v \in N(v^\dagger) \setminus N(v^*)$
        $A^* = A^* \cup \{(v, v^*)\}$

edges, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that:

$$\bigcup_{(v^\dagger, S_j) \in A} S_j = U.$$

Since we are interested in finding the smallest such subset, a solution to the above instance of the Influence Recovery problem gives us a solution to the Set Cover problem. □

Having proven the NP-hardness of the Influence Recovery problem for the Linear Threshold model, we now move to designing and testing a simple heuristic solution.

3.5 Heuristic Solution

Knowing that for most of our problems it is NP-hard to compute the optimal solution, we now describe a simple polynomial time heuristic that seems to perform well in practice.

3.5.1 The ROAM heuristic

Typically, one has very limited knowledge of the social ties beyond his or her immediate friends, and maybe friends of friends. However, even if one was able to somehow acquire information about the entire network structure, our theoretical results from the previous section suggest that it is extremely unlikely for such an individual to have the necessary computational power to optimally disguise himself or herself.

Against this background, we investigate the possibility of disguising one's centrality adequately (albeit not optimally) while restricting one's attention to only his or her immediate neighbourhood, and without requiring massive computational power nor expertise in sophisticated optimization techniques. With this in mind, we propose a heuristic whose instructions are simple enough for an average user of social-networking services to understand and use, regardless of their technical background. The pseudocode of our heuristic, called ROAM—*Remove One, Add Many*—is presented as Algorithm 1. An illustration of how it works is presented in Figure 3.9. Specifics of choosing specific nodes in the ROAM heuristic are described in the next subsection.

Let us now comment on this heuristic, starting with the construction of the set R^*, *i.e.*, with the removal of a connection between $v^\dagger$ and v^*. As far as decreasing the centrality of $v^\dagger$ is concerned, this step can only be beneficial. More specifically, cutting off $v^\dagger$ from one of its neighbours is the only way to reduce the degree of $v^\dagger$. Likewise, this operation can only

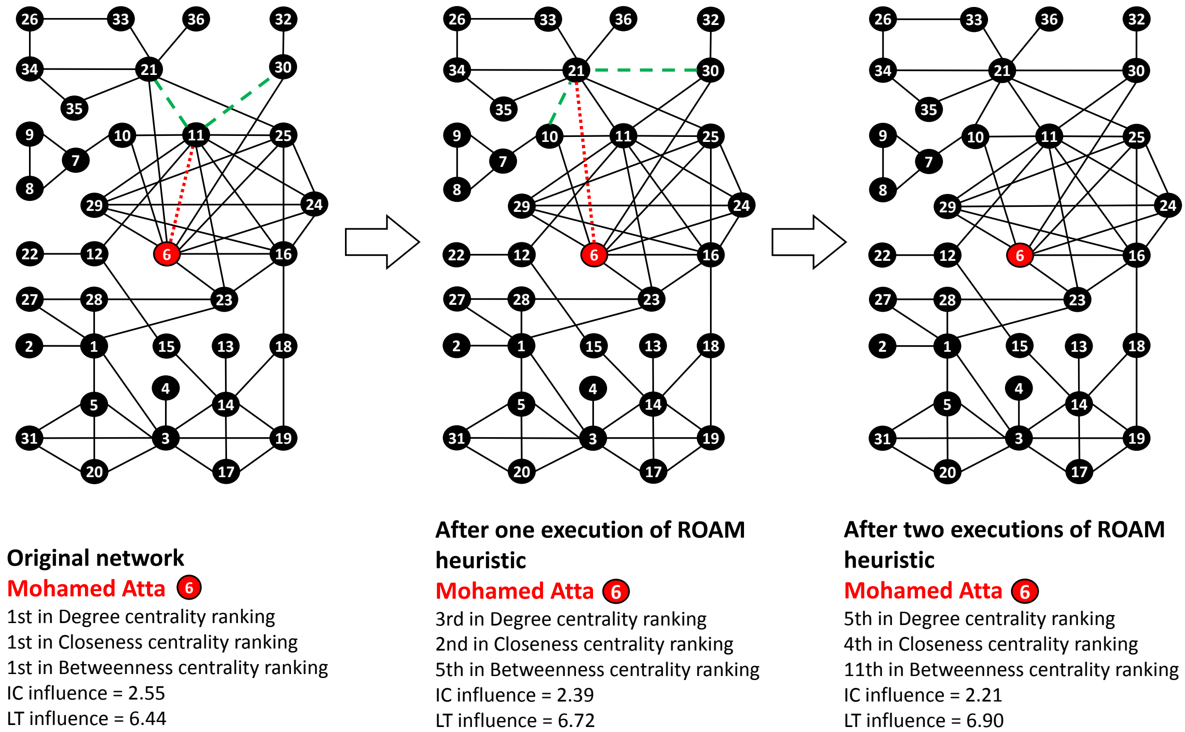

Figure 3.9: Executing the ROAM heuristic with budget 3 twice on the 9/11 terrorist network to hide Mohamed Atta—one of the ringleaders of the attack [80]. The dotted red edge is the one to be to removed by the algorithm, and the dashed green edges are the ones to be added.

decrease the closeness of $v^\dagger$ (this happens when all shortest paths between $v^\dagger$ and some other node run through the removed link), and can only decrease the betweenness of $v^\dagger$ (this happens when some of the shortest paths going through $v^\dagger$ contain the removed link). However, as far as the influence of $v^\dagger$ is concerned, this operation may be detrimental, as it deprives $v^\dagger$ of its direct influence over v^*.

Moving on to the construction of A^*, this step is primarily designed to compensate for any influence that $v^\dagger$ may have lost during the previous step. Specifically, it creates new, indirect connections between $v^\dagger$ and v^* (that was previously a neighbour of $v^\dagger$) to compensate for the direct one that was removed earlier. As far as the centrality of $v^\dagger$ is concerned, while this step does not affect the degree of $v^\dagger$, it increases the degrees of some of its neighbours, which in turn contributes towards concealing the relative importance of $v^\dagger$ within the network. Furthermore, the addition of an edge, (v, v^*)—where v is some neighbour of $v^\dagger$—cannot increase the closeness centrality of $v^\dagger$ beyond its state before running the ROAM heuristic altogether. This is because any path containing (v, v^*) and $(v^\dagger, v)$ is certainly longer than any original path in which (v, v^*) and $(v^\dagger, v)$ were replaced with $(v^*, v^\dagger)$. Likewise, the addition of this link cannot increase the betweenness centrality of $v^\dagger$ beyond its original state, because replacing a direct connection between $v^\dagger$ and v^* with an indirect one cannot increase the percentage of shortest paths going through $v^\dagger$.

We now specify how ROAM should be modified to work on directed networks. First of all, v^* is not chosen among the neighbours of $v^\dagger$, but rather among the successors of $v^\dagger$. This is mainly because removing a successor of $v^\dagger$ reduces its closeness centrality, whereas removing a predecessor has no such impact. As for the $b-1$ neighbours of $v^\dagger$ to be connected to v^*, they are chosen among the predecessors of $v^\dagger$; for each such predecessor, v, we add the edge (v, v^*). This is mainly because it could potentially rebuild the influence of $v^\dagger$ on v^*, which was hampered by the removal of the edge $(v^\dagger, v^*)$. Furthermore, for every shortest path that contains the edge $(v^\dagger, v)$, the addition of (v, v^*) could create a new alternative shortest path that does not pass through $v^\dagger$, thus further reducing the betweenness centrality of $v^\dagger$.

3.5.2 Configuring the ROAM Heuristic

The ROAM heuristic involves choosing v^* (the neighbour of $v^\dagger$ whom the heuristic will disconnect from $v^\dagger$), and choosing the $b-1$ neighbours of $v^\dagger$ whom the heuristic will connect to v^*. We conducted a number of experiments to determine whether it is more beneficial to choose v^* as the neighbour of $v^\dagger$ with the least connections or the most connections. Likewise, we wanted to determine whether it is more beneficial to choose the $b-1$ neighbours of $v^\dagger$ (who will be connected to v^*) as the ones with the least connections or the most connections.

In particular, Figure 3.10 compares the different settings given 50 randomly generated scale-free networks consisting of 100 nodes each, where 3 edges are added with each step of the generation process (for more details, see [10]). We chose scale-free networks as they resemble real-life networks in many way, *e.g.*, in terms of degree distribution. As for the source node, it is chosen to be the one with the lowest sum of positions in centrality rankings (ties are broken uniformly at random). As for the Independent Cascade model, we set the activation probability to be $p(v, w) = 0.15$ for every pair of nodes, $v, w \in V$. As for the Linear Threshold model, for every node, $v \in V$, the threshold value, t_v, is sampled uniformly at random from the set $\{0, \ldots, |P(v)|\}$. For both models, the influence values are approximated using the Monte-Carlo method. In the figure, we write ROAM-x-y(b), where x can either be "*max*" or "*min*" (indicating that v^* is the neighbour with the *highest* degree or the *lowest* connections, respectively) and y can either be "*max*" or "*min*" (indicating that the $b-1$ neighbours are chosen to be the ones with the *highest* degree or the *lowest* degree, respectively), whereas b represents the budget (which is set to 3 in this experiment). Since the results are averaged over 50 random networks, the coloured areas in the figure represent the 95% confidence intervals. For each network, the ROAM heuristic is executed multiple, consecutive times; the x-axis in each subfigure represents the number of executions.

As can be seen, while there is no setting that dominates the others, the best overall performance seems to be achieved by ROAM-max-min(3). Based on this, in all subsequent experiments on ROAM, we choose v^* as the neighbour of $v^\dagger$ with the *highest* degree, and we connect v^* to the $b-1$ neighbours of $v^\dagger$ with the *lowest* degree.

3.5.3 Experimental Results

We now describe experimental results of using ROAM heuristic to hide in the network. The setting for each of our experiments consists of a network, a budget and a source node. More specifically, we experiment with a budget of 2, 3 and 4. The source node is

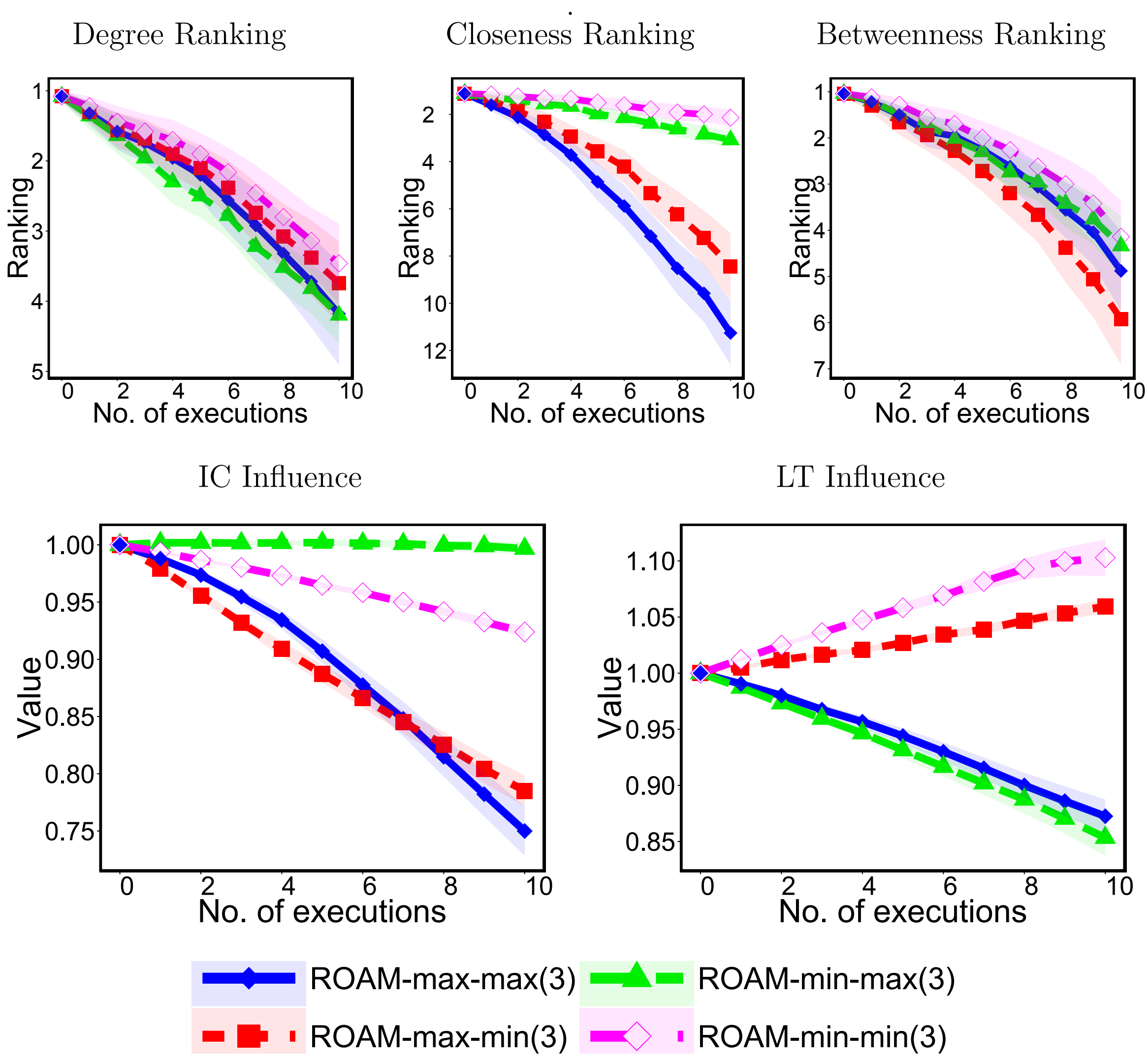

Figure 3.10: Comparing different settings of ROAM on 50 randomly generated scale-free network consisting of 100 nodes, with 3 edges added in each step of the generation process. For each such network, ROAM is executed multiple, consecutive times (the x-axis represents the number of executions). The subfigures show the source node's ranking (according to different centrality measures), and the relative change in its influence value (according to different influence models).

assumed to be the one with the lowest sum of positions in the centrality rankings (ties are broken uniformly at random). Whenever the Independent Cascade model is used, activation probability of 0.15 is assumed on each link. On the other hand, whenever the Linear Threshold model is used, the uniform distribution of thresholds is assumed. For both models, the influence values are approximated using the Monte-Carlo method. In each of these experiment, the ROAM heuristic is executed multiple, consecutive times. All datasets are described in Section 2.2.

Figures 3.11 and 3.12 shows example results of our experiments. The centrality plots depict the ranking of the source node, whereas the influence plots depict its relative influence value (compared to the *original* influence value before executing the heuristic

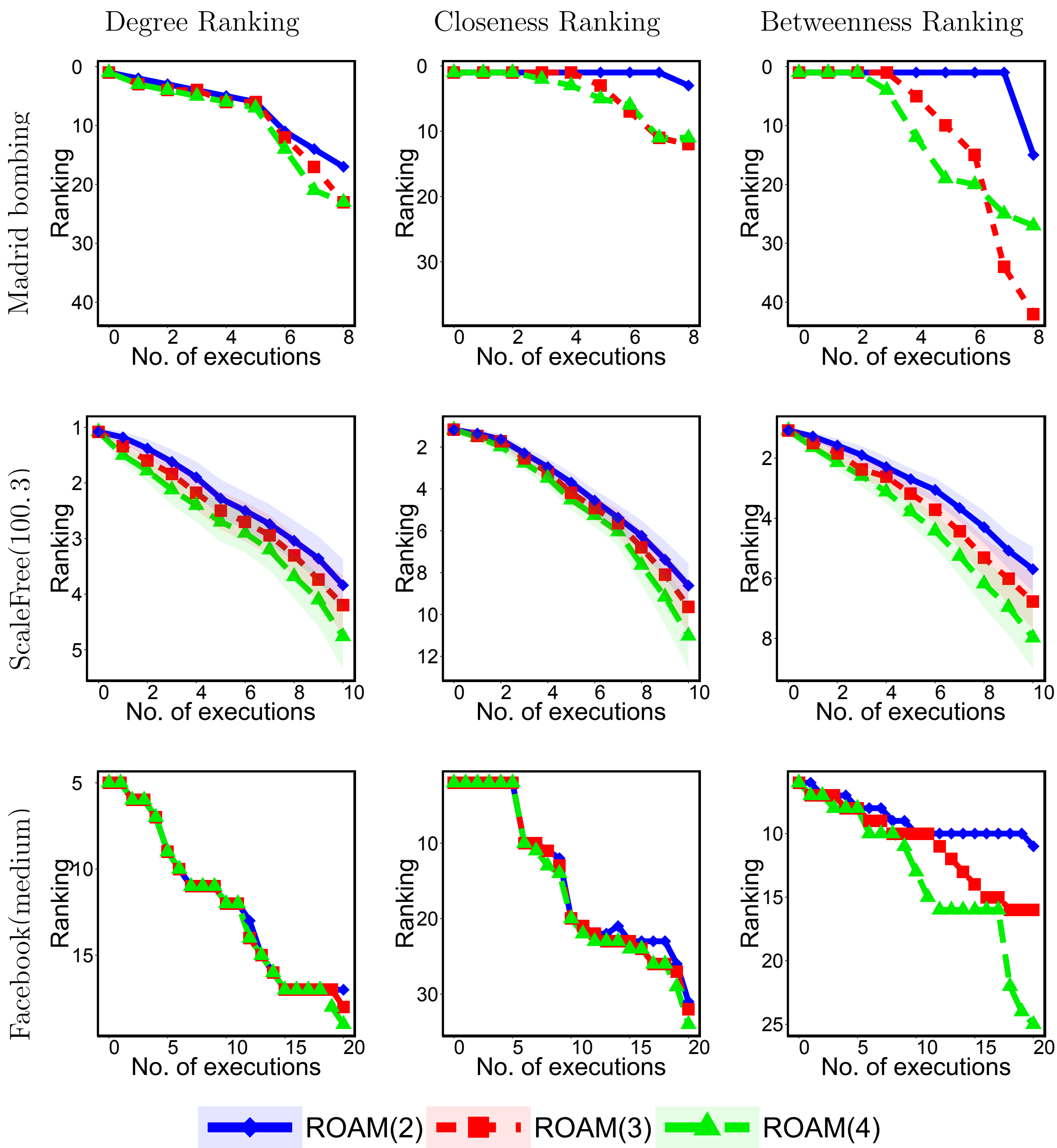

Figure 3.11: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

altogether). As can be seen, the heuristic is effective in decreasing the source node's ranking, and this effectiveness increases with the budget spent on rewiring the network. As for influence, the performance of the heuristic varies depending on the network, the influence model, and the budget. Overall, the greater the budget, the greater the influence, *e.g.*, a budget of 4 manages to maintain (or even increase) the influence in 4 out of 6 cases.

All experimental results for the ROAM heuristic are presented in Appendix A in Figures A.1, A.2, A.3, A.4, A.5, A.6, A.7, A.8, A.9,A.10, A.11 and A.12.

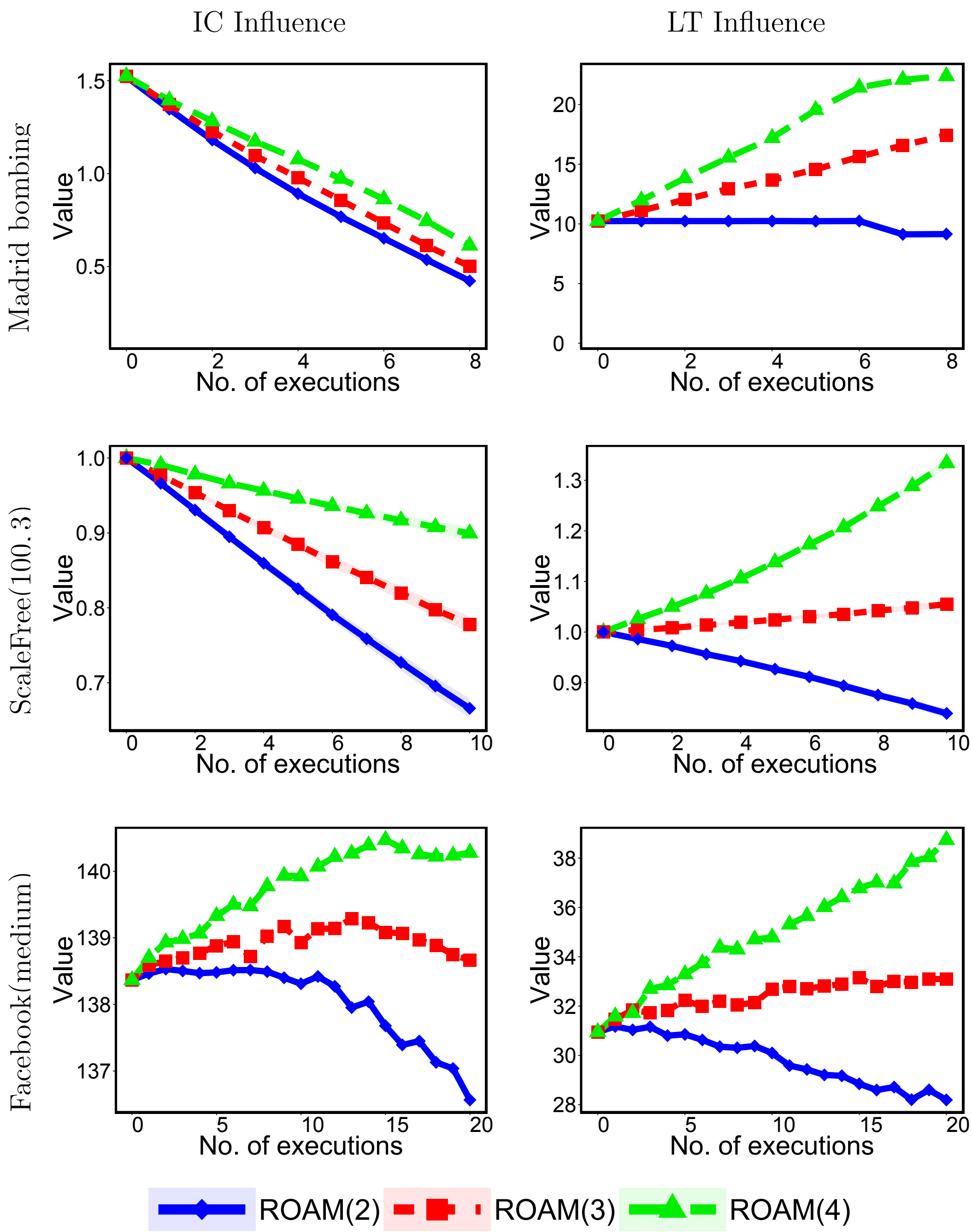

Figure 3.12: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

3.6 Concluding Remarks

Our goal was to understand the practical limits of disguising individuals, *i.e.*, the ability to increase the likelihood of being overlooked by various centrality measures. Our main result is that, despite the hardness of finding optimal solutions, disguising oneself can be surprisingly easy in practice, and can be achieved with the use of simple heuristics that can be easily implemented even by lay people.

On one hand, our findings contribute towards charting the limits of protecting privacy in social networks. On the other hand, they expose potentially serious limitations of using generic social network analysis tools in security applications. The fact that such tools can be easily misled underlines the need for developing specialized tools that account for the nature of links and nodes in the network, and not just its topology.

As a first step in the study of how to strategically evade node-centrality analysis, we focused on a basic model, designed to shed light on the workings of some of the most fundamental social network analysis tools. Nevertheless, our model has a number of limitations which will be discussed next.

First, when identifying key nodes, the seeker is assumed to use three simple yet fundamental measures of centrality—degree, closeness, and betweenness—as these measures can be used even with large social networks. Given this assumption, the evader can focus on reducing her centrality without compromising her influence, knowing that the seeker is unable to use influence measures as means of identifying leaders. This assumption is based on the fact that it is intractable to measure influence according to the Independent Cascade or the Linear Threshold models, especially when the network is large. Indeed, even industrial social network analysis software packages do not typically allow for efficient and accurate approximation of influence in large networks [64]. However, in cases where the seeker has extensive processing power, and is handling small networks, then the leader can be identified by ranking the nodes not based on their centrality scores but rather based on their influence. For such cases, our study can serve as a step towards understanding the trade-off between the two measures of node importance, *i.e.*, centrality measures and influence measures. In fact, in our analysis, we divided the process of disguising the source node into two phases: the centrality minimization phase, and the influence recovery phase. As such, if the source node suspects that the seeker will also use influence as a measure of centrality, then the source node can focus solely on the centrality minimization phase.[1]

Second, our model assumes that the evader's actions are automatically successful. While this kind of assumption is common when considering theoretical settings, real-life applicability often involves probability of a failure of an action. An interesting possible direction of future work is taking this possibility into account both in theoretical analysis, as well as in designing the heuristic solutions. Such algorithms with built-in fail-safes in case of an action failure would be more suitable to use in real-life noisy settings, where the possibility of adding and removing edges from the network might depend on arbitrary decisions of network members.

Third, although our heuristic algorithm—ROAM—appears to be effective in practice, it does not provide any worst-case guarantees on the solution quality. This is because our primary goal was to develop algorithms that are scalable and applicable by lay individuals who want to protect their privacy; such individuals typically do not know the topology of the entire network, nor do they have the ability the rewire links between two complete strangers. Undoubtedly, however, scalability and applicability are achieved at the expense

[1]Such a study would probably be interesting only if there are additional constraints that the source node wants to satisfy. Otherwise, the source node would be better off removing as many of its neighbours as possible, leading to an optimal solution where the source node is cut off entirely from the network.

of solution quality. As such, there is room to develop more advanced algorithms that compute near-optimal solutions (although, of course, there is still a limit to what can be achieved in polynomial time).

Fourth, in our experiments we assume that the evaders do not know the exact social network analysis tool that the seeker is using. As such, we focus on developing a heuristic algorithm that performs reasonably well on a wide range of social network analysis tools, taken from the rich repository of node-centrality tools. Alternatively, one can focus on developing dedicated algorithms for each centrality-measure. Every such specialized algorithm would have a narrower scope, but in return would likely outperform the more generic one that was developed in this dissertation.

To summarize, we believe that our work is an important first step in the study of the issue of concealing one's importance in a social network. At the same time, further research is necessary to better understand the limits of how well the problem can be solved. Above we mentioned just a few of many possible venues of future work.

Chapter 4

Hiding Leaders

In this chapter we discuss the problem of lowering the positions in centrality ranking of multiple nodes at once, allowing to effectively hide leaders of a network. We formally define the problem, prove its NP-completeness for three most widely used centrality measures, and show an algorithm for constructing a network where leaders are well-hidden.

4.1 Introduction

Mapping terrorist networks and analysing their structure is a vitally important part of counter-terrorism efforts. Not only does it help to understand their operational methods and the way of thinking, but also it plays a key role in designing and implementing destabilization strategies [28, 47, 104]. Common approach to devising such strategies requires identifying individuals that are playing central roles in the organization [45, 46]. Typically used tools in such situations are centrality measures [92, 100, 61]. In the previous chapter we analysed the problem avoiding centrality-based detection from the perspective of a single network member. We now intend to expand this analysis for multiple nodes.

Understanding how criminals organize themselves into a network structure is challenging at various levels [80, 137]: the data may be incomplete, the nature of the relationships between the criminals may be unclear, and the network may evolve continuously. The literature on this research problem agrees that criminals in general, and terrorists in particular, face a trade-off between secrecy and efficiency [105], though the way in which both factors are modelled differs.

Overall, the literature can be divided into two general categories, which we now briefly discuss.

In the first category, researchers study known topologies of historical or contemporary criminal networks, with the aim to understand why particular structures have emerged [38, 42, 75]. One of the most comprehensive studies in this category is due to Kilberg [75], who analysed an extensive dataset of more than 240 terrorist networks, and provided a classification of those networks based on their structure and functionality. Furthermore, using regression analysis, the author tried to quantify the degree to which the shapes of terrorist networks are influenced by such variables as the GDP level of the target country, the political rights and civil liberties therein, and the inclination to attack police and military targets in that country.

Our study contributes to the second category of the literature, which is more theoretical in nature and aims to explain the structural properties of covert networks by developing explicit models of the terrorists' preferences and the different choices they face [43, 66, 91]. With such analyses, certain network topologies typically emerge as the result of modelling the terrorists as rational decision makers. A notable example of such a model is that of Lindelauf *et al.* [91], who consider the trade-off between secrecy and operational efficiency of a terrorist network, and borrows concepts from both game theory and graph theory to identify more fitting topologies. Arguably, it is less efficient if a message has to be passed many times from one person to another (*i.e.*, the shortest path from the sender to the receiver is relatively long). Based on this, Lindelauf *et al.* defined efficiency as the reciprocal of the total distance of the graph (*i.e.*, the sum of shortest distances between any two nodes in the network). Secrecy, in turn, is defined for each node and is proportional to the fraction of the network that remains unexposed when this node is detected. The secrecy of the network is the sum of the secrecy scores over all nodes.

In this chapter, we propose a theoretical model to study the secrecy-efficiency trade-off that differs from previous ones in a number of ways. Firstly, inspired by studies of real-life covert networks [30, 95], we take a leader-centric approach, *i.e.*, we focus on the role played in a terrorist network by its leadership. In more detail, we investigate how the topology of the network can be deliberately designed to keep the leaders identities hidden. In this context, while the previous literature on identifying leaders of terrorist networks typically assumed that such leaders are not aware of the techniques and methods used by law-enforcement agencies, we assume that this is not the case, *i.e.*, in our model the terrorist leaders strategically shape their network to protect themselves from detection by the centrality measures. In fact, recent media reports and academic studies of criminal and terrorist organizations suggest that members of such organizations are becoming increasingly tech-savvy [109, 68]. Hence, their obliviousness with respect to the available social network analysis techniques should not be taken for granted.

As already argued, secrecy is not the only objective that the leaders of a terrorist network may have. Indeed, if they were concerned only with hiding themselves, they would simply cut most (if not all) of their connections in the network. This, however, would clearly impair the leaders' efficiency. In our model, the efficiency of the leaders is defined as their influence over the network. In other words, the leaders in our model face the trade off between hiding from centrality measures, and influencing the network members.

In the first part of the chapter we focus on the computational aspects of modifying an existing network so as to shield a group of leaders from centrality analysis. In more detail, we analyse the hardness of rewiring the network so that the ranking of all leaders (based on one of the three main centrality measures) drops below a certain threshold. At first glance, this problem may appear to be easy at least for the degree centrality, which is mathematically uninvolved. Surprisingly, however, we find that this is not the case. In comparison with the previous chapter, we now consider a group of nodes that wish to be hidden (while in Chapter 3 we consider only a single node), and we are interested in relative position in centrality ranking (while in Chapter 3 we analysed the complexity of lowering absolute centrality value), and these differences in the setting lead to computational hardness. Our results are in line with the literature on modifying a network to increase centrality [36].

Given the hardness of modifying an existing network, we turn out attention to a different question, which is how a covert network can be built from scratch so that the leaders are hidden and, at the same time, have a reasonable influence over the network members. Here the main idea is for the leaders to surround themselves with an "inner circle" of trustees, called "captains", whose role is to conceal the leaders, and to pass on their communicates to the rest of the network. We identify one such network structure, and prove that every captain is guaranteed to be ranked higher than any of the leaders (according to the three standard centrality measures). In fact, "inner circles" have been identified in various real-life terrorist networks, *e.g.*, in Al-Quaeda [11] and IRA [139]. While we do not have access to data that confirms that those real-life "inner circles" have similar structure to the ones obtained in this dissertation, we hope that our results shed more light on why such circles may exist in covert networks. In this context, charting the topology of covert networks became one of the key research directions.

The remainder of this chapter is organized as follows. In Section 4.2 we present the definition of the main problem considered in this chapter, *i.e.*, hiding a group of leaders. In Section 4.3 we prove this problem to be NP-complete for three most widely used centrality measures. In Section 4.4 we present the way of building a new communication structure that keeps leaders well-hidden and influential. In Section 4.5 we summarize and discuss our findings. Note that many concepts we use, such as centrality measures and influence models, are defined in the previous chapter, in Section 3.2.

4.2 Problem Definition

We assume that the covert network is composed of two types of agents: the *leaders* and the *followers*. Furthermore, we assume that the leaders are aware that hostile agencies may use centrality analysis to identify them. Thus, the leaders would like to strategically modify the existing network so that their centrality rankings becomes lower than a certain predefined threshold $d \in \mathbb{N}$ that we refer to as *safety margin*. To achieve this objective, no more than $b \in \mathbb{N}$ modifications can be made to the network (b will be referred to as the *budget*).

Following the arguments in Section 3.3, we also introduce a set of edges allowed to be added $\hat{A}$ and a set of edges allowed to be removed $\hat{R}$. These sets may be used to preserve the critical communication channels within the organisation or, *e.g.*, to avoid connecting two individuals belonging to conflicted parties within the network.

Formally, we define *Hiding Leaders* as follows:

Definition 4 (Hiding Leaders). *This problem is defined by a tuple* $(G, L, b, c, d, \hat{A}, \hat{R})$, *where* $G = (V, E) \in \mathbb{G}$ *is a network,* $L \subset V$ *is a set of leaders to be hidden,* $b \in \mathbb{N}$ *is a budget specifying the maximum number of edges that can be added or removed from the network,* $c : \mathbb{G} \times V \to \mathbb{R}$ *is a centrality measure,* $d \in \mathbb{N}$ *is a chosen safety margin,* $\hat{A} \subseteq \bar{E}$ *is a set of edges allowed to be added, and* $\hat{R} \subseteq E$ *is a set of edges allowed to be removed. Then, if we denote by* $F = V \setminus L$ *the set of "followers", the goal is to identify a set of edges to be added to the network,* $A^* \subseteq \hat{A}$, *and a set of edges to be removed from the network,* $R^* \subseteq \hat{R}$, *such that* $|A^*| + |R^*| \leqslant b$ *and the resulting network* $G' = (V, (E \cup A^*) \setminus R^*)$ *contains at least* d *followers that each has a centrality score higher than that of any leader,* *i.e.:*

$$\exists_{F' \subseteq F}(|F'| \geqslant d \wedge \forall_{f \in F'} \forall_{l \in L} c(G', f) > c(G', l))$$

In the following section we perform complexity analysis of the Hiding Leaders problem for degree, closeness and betweenness centralities.

4.3 Complexity Analysis

Intuitively, the Hiding Leaders problem should be easy to solve for the degree centrality measure. Indeed, adding an edge between any two disconnected followers increases their degree centrality with respect to all the leaders. However, we prove below that the problem is in fact NP-complete for the degree centrality measure.

Theorem 5. *The problem of Hiding Leaders is NP-complete given the degree centrality measure.*

Proof. The problem is trivially in NP, since after the addition of a given set of edges A^* and the removal of a given set of edges R^* it is possible to compute the degree centrality for all nodes in polynomial time.

We now prove that the problem is NP-hard. To this end, we give a reduction from the NP-complete problem of Finding k-Clique, as defined in Section 2.3.3. To remind the reader, the goal is to determine whether there exist k nodes in G that form a clique.

Given an instance of the problem of Finding k-Clique, defined by some $k \in \mathbb{N}$ and a network $G = (V, E)$, let us construct a network, $H = (V', E')$, as follows:

- **The set of nodes:** For every node, $v_i \in V$, we create a single node, v_i, as well as $|N_G(v_i)|$ other nodes, denoted by $X = \{x_{i,1}, \dots, x_{i,|N_G(v_i)|}\}$. Additionally, we create one node called y, as well as $n + k - 1$ other nodes, namely $L' = l_1, \dots, l_{n+k-1}$.
- **The set of edges:** We create an edge between two nodes $v_i, v_j \in V$ if and only if this edge was not present in G, *i.e.*, $(v_i, v_j) \in E' \iff (v_i, v_j) \notin E$. Additionally, for every v_i we create an edge (v_i, y) as well as an edge $(v_i, x_{i,j})$ for every $x_{i,j}$. We also create an edge (l_i, l_j) between every pair of nodes $l_i, l_j \in L'$, except for the edge (l_1, l_2). Finally, we create two additional edges, (l_1, y) and (l_2, y).

An example of such a network H is illustrated in Figure 4.1.

Now, consider instance $(H, L, b, c, d, \hat{A}, \hat{R})$ of the problem of Hiding Leaders, where:

- $H = (V', E')$ is the network we just constructed;
- $L = V' \setminus V$;
- $b = \frac{k(k-1)}{2}$;
- c is the degree centrality measure;
- $d = k$;
- $\hat{A} = E$;

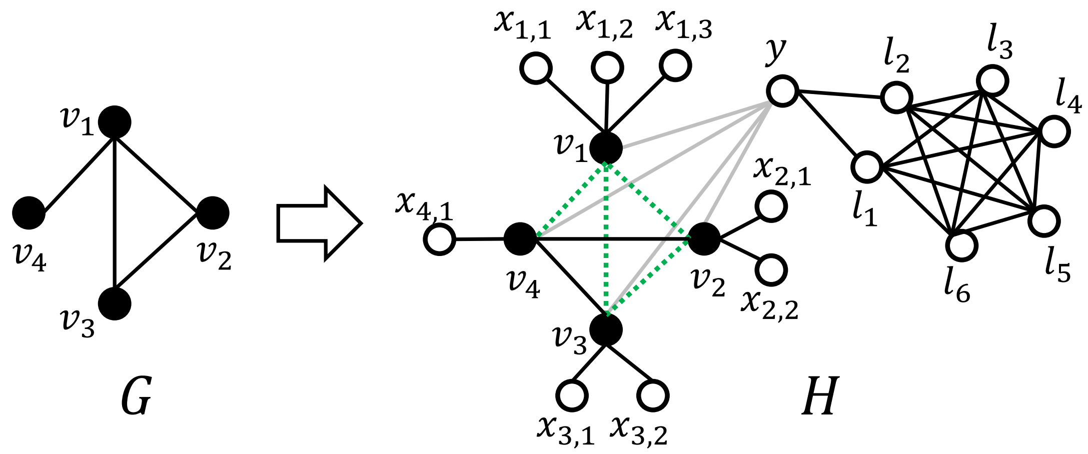

Figure 4.1: An illustration of the network used in the NP-completeness proof of the problem of Hiding Leaders given the degree centrality.

- $\hat{R} = \emptyset$.

From the definition of the problem of Hiding Leaders, we know that the edges to be added to H must be chosen from E, *i.e.*, edges from the network in the Finding k-Clique problem. Out of those edges, we need to choose subset, $A^* \subseteq E$, as a solution to the Hiding Leaders problem. In what follows, we will show that a solution to the above instance of the Hiding Leaders in H corresponds to a solution to the problem of Finding k-Clique in G.

First, note that each of the k nodes with the highest degree centrality in H must be a member of L'. This is because there are more than k nodes in L', each of which has a degree of $n+k-2$, while the degree of every node in $V' \setminus L'$ is smaller than $n+k-2$. Thus, in order for A^* to be a solution to the problem of Hiding Leaders, the addition of A^* to H must increase the degree of at least k nodes in V such that each of them has a degree of at least $n+k-1$ (note that the addition of A^* only increases the degrees of nodes in V, as we already established that $A^* \subseteq E$). Now since in H the degree of every node in V equals n (because of the way H is created), then in order to increase the degree of k such nodes to $n+k-1$, each of them must be an end of at least $k-1$ edges in A^*. But since the budget in our problem instance is $\frac{k(k-1)}{2}$, then the only possible choice of A^* is the one that increases the degree of exactly k nodes in V by exactly $k-1$. If such a choice of A^* is available, then surely those k nodes would form a clique in G, since all $\frac{k(k-1)}{2}$ edges in A^* are taken from G. □

Having proven the NP-completeness of the problem given the degree centrality, we next prove its NP-completeness given the closeness centrality.

Both proof for the closeness centrality and the following proof for the betweenness centrality are based on a similar idea. We create a network with two nodes pretending to the first place in centrality ranking, namely nodes t and z. Node z is one of the leaders, who we intend to hide. However, in the initial state of the network it is also the node with the highest centrality. We show that increasing the centrality of t—in order for it to achieve higher centrality than z—requires adding the edges corresponding to an optimal solution of the underlying Set Cover problem instance.

Theorem 6. *The problem of Hiding Leaders is NP-complete given the closeness centrality measure.*

Proof. The problem is trivially in NP, since after the addition of a given A^*, and the removal of a given R^*, it is possible to compute the closeness centrality for all nodes in polynomial time.

Next, we prove that the problem is NP-hard. To this end, we propose a reduction from the NP-complete 3-Set Cover problem, as defined in Setion 2.3.3. To remind the reader, $U = \{u_1, \ldots, u_l\}$ denotes the universe, $S = \{S_1, \ldots, S_m\}$ denotes the set of subsets of the universe, where for every S_i we have that $|S_i| = 3$, and the goal is to determine whether there exist k elements of S the union of which equals U.

Given an instance of the 3-Set Cover problem, let us construct a network G as follows:

- **The set of nodes:** For every $S_i \in S$, we create a single node denoted by S_i, and for every $u_i \in U$, we create two nodes denoted by u_i and w_i. We denote the set of all S_i nodes by S, the set of all u_i nodes by U, and the set of all w_i nodes by W. In addition, we create $l+m+1$ nodes denoted by $X = \{x_1, \ldots, x_{l+m+1}\}$, and m nodes denoted by $Y = \{y_1, \ldots, y_m\}$. Lastly, we create two additional nodes, denoted by t and z.

- **The set of edges:** First, we create the edge (t, z). Then, for every node x_i we create an edge (x_i, t), for every node y_i we create an edge (y_i, z), for every node w_i we create the edges (w_i, z) and (w_i, u_i), and every node S_i we create an edge (S_i, u_j) for every $u_j \in S_i$. After that, we create k edges, $(z, S_1), \ldots, (z, S_k)$, *i.e.*, we connect k initial elements of S to z. Finally, we create edges such that the nodes in S form a clique, and those in U also form a clique. That is, we create an edge (u_i, u_j) for every $u_i, u_j \in U$ and an edge (S_i, S_j) for every $S_i, S_j \in S$.

An example of the resulting network, G, is illustrated in Figure 4.2.

Now, consider the following instance of the problem of Hiding Leaders, $(G, L, b, c, d, \hat{A}, \hat{R})$, where:

- G is the network we just constructed;
- $L = \{z\} \cup U \cup W \cup X \cup Y$;
- $b = k$, where k is the parameter of the 3-Set Cover problem (where the goal is to determine whether there exist k elements of S the union of which equals U);
- c is the closeness centrality measure, *i.e.*, $c(G, v) = \frac{n-1}{\sum_{w \in V \setminus \{v\}} d(v,w)}$;
- $d = 1$;
- $\hat{A} = \{(t, S_i) : S_i \in S\}$;
- $\hat{R} = \emptyset$.

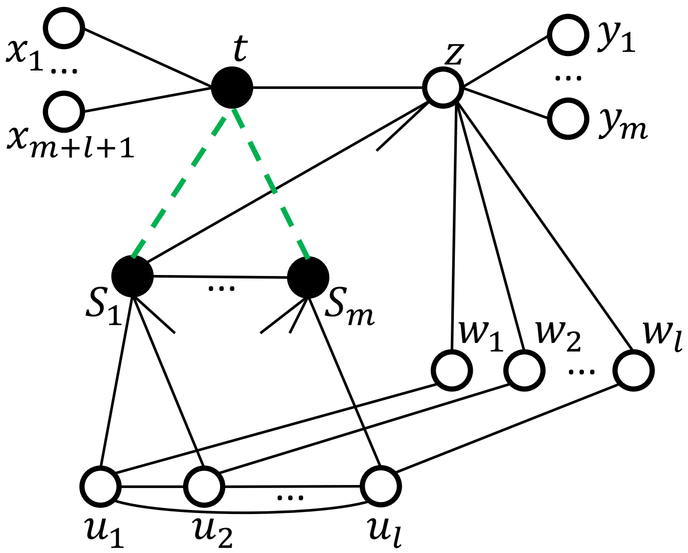

Figure 4.2: An illustration of the network used in the NP-completeness proof of the problem of Hiding Leaders given the closeness centrality.

From the definition of the problem of Hiding Leaders, we see that the only edges that can be added to the graph are the edges between t and the members of S, *i.e.*, $A^* \subseteq \hat{A}$, where $\hat{A} = \{(t, S_1), \ldots, (t, S_m)\}$. Notice that any such choice of A^* corresponds to selecting a subset of $|A^*|$ elements of S in the 3-Set Cover problem. In what follows, we will show that a solution to the above instance of Hiding Leaders corresponds to a solution to the 3-Set Cover problem.

First, we show that for every $v \in V \setminus \{t, z\}$ and every $A^* \subseteq \hat{A}$ we either have $c(G', v) < c(G', t)$ or $c(G', v) < c(G', z)$, where $G' = (V, E \cup A^*)$. To this end, let $D(G', v)$ denote the sum of distances from v to all other nodes, *i.e.*, $D(G', v) = \sum_{w \in V \setminus \{v\}} d(v, w)$. Note that $D(G', v) = \frac{n-1}{c(G',v)}$. We show that the following holds:

$$\forall_{v \in V \setminus \{t,z\}} \forall_{A^* \subseteq \hat{A}} (D(G', v) > D(G', t)) \vee (D(G', v) > D(G', z))$$

Let d_t denote $\sum_{u_i \in U} d(t, u_i) + \sum_{S_i \in S} d(t, S_i)$. Table 4.1 presents computation of distances between nodes in G'. In what follows, we compute $D(G', v)$ for different types of node v. Given these values we have that:

- $D(G', z) = 5l + 5m - k + 3$;
- $D(G', t) = 3l + 3m + 2 + d_t$;
- $D(G', x_i) = 6l + 6m + 3 + d_t > D(G', t)$;
- $D(G', y_i) = 8l + 8m - k + 4 > D(G', z)$;
- $D(G', w_i) \geqslant 7l + 7m + 3 > D(G', z)$ because we have $d(w_i, S_j) \geqslant 2$;
- $D(G', u_i) \geqslant 6l + 7m + 5 > D(G', z)$ because we have $d(u_i, t) \geqslant 2$ and $d(u_i, S_j) \geqslant 1$;
- $D(G', S_i) \geqslant 6l + 5m > D(G', z)$; because we have $d(S_i, z) \geqslant 1$ and $d(S_i, t) \geqslant 1$.

v	$d(v,z)$	$d(v,t)$	$\sum_{x_i \in X} d(v,x_i)$	$\sum_{y_i \in Y} d(v,y_i)$	$\sum_{w_i \in W} d(v,w_i)$	$\sum_{u_i \in U} d(v,u_i)$	$\sum_{S_i \in S} d(v,S_i)$	$D(G',v)$
z	0	1	$2(m+l+1)$	m	l	$2l$	$k+2(m-k)$	$5l+5m-k+3$
t	1	0	$m+l+1$	$2m$	$2l$	$\sum_{u_i \in U} d(t,u_i)$	$\sum_{S_i \in S} d(t,S_i)$	$3l+3m+2+d_t$
x_i	2	1	$2m+2l$	$3m$	$3l$	$l+\sum_{u_i \in U} d(t,u_i)$	$m+\sum_{S_i \in S} d(t,S_i)$	$6l+6m+3+d_t$
y_i	1	2	$3(m+l+1)$	$2(m-1)$	$2l$	$3l$	$2k+3(m-k)$	$8l+8m-k+4$
w_i	1	2	$3(m+l+1)$	$2m$	$2(l-1)$	$1+2(l-1)$	$\geqslant 2m$	$\geqslant 7l+7m+3$
u_i	2	$\geqslant 2$	$3(m+l+1)$	$3m$	$1+2(l-1)$	$l-1$	$\geqslant m$	$\geqslant 6l+7m+5$
S_i	$\geqslant 1$	$\geqslant 1$	$2(m+l+1)$	$2m$	$2l$	$3+2(l-3)$	$m-1$	$\geqslant 6l+5m$

Table 4.1: Distances between nodes in the graph constructed for the closeness centrality proof.

Therefore, either t or z has the highest closeness centrality. Since $z \in L$ and $t \in F$, then $A^* \subseteq \hat{A}$ is a solution to the problem of problem of Hiding Leaders if and only if $D(G', t) < D(G', z)$. This is the case when:

$$d_t < 2l + 2m - k + 1.$$

Let $U_A = \{u_i \in U : \exists_{S_j \in S} u_i \in S_j \wedge (t, S_j) \in A^*\}$. We have that $d_t = |A^*| + 2(m - |A^*|) + 2|U_A| + 3(l - |U_A|)$ which gives us:

$$d_t = 3l - |U_A| + 2m - |A^*|$$

Since by definition $|U_A| \leqslant l$ and $|A^*| \leqslant k$, it is possible that $d_t < 2l + 2m - k + 1$ only when $|U_A| = l$ and $|A^*| = k$, *i.e.*, $\forall_{u_i \in U} \exists_{S_j \in S} u_i \in S_j \wedge (t, S_j) \in A^*$. This solution to the problem of Hiding Leaders corresponds to a solution to the given instance of the 3-Set Cover problem, which concludes the proof. □

Having proven the NP-completeness of the problem for the case of the closeness centrality, we next prove its NP-completeness for the case of the betweenness centrality.

As mentioned before, the proof for the betweenness centrality case follows a very similar reasoning to the proof for the closeness centrality case. This time we add edges to the network in order to create alternative shortest paths omitting node z, thus reducing its betweenness centrality and making it lose the first place in the centrality ranking.

Theorem 7. *The problem of Hiding Leaders is NP-complete given the betweenness centrality measure.*

Proof. The problem is trivially in NP, since after the addition of a given set of edges A^*, and the removal of a given set of edges R^*, it is possible to compute the betweenness centrality for all nodes in polynomial time.

Next, we prove that the problem is NP-hard. To this end, we propose a reduction from the NP-complete 3-Set-Cover problem, as defined in Setion 2.3.3. To remind the reader, $U = \{u_1, \ldots, u_l\}$ denotes the universe, $S = \{S_1, \ldots, S_m\}$ denotes the set of subsets of the universe, where for every S_i we have that $|S_i| = 3$, and the goal is to determine whether there exist k elements of S the union of which equals U.

Given an instance of the 3-Set Cover problem, let us construct a network G as follows:

- **The set of nodes:** For every $S_i \in S$, we create a single node denoted by S_i, and for every $u_i \in U$, we create a single node denoted by u_i. We denote the set of all S_i nodes by S, and the set of all u_i nodes by U. In addition, we create α nodes denoted by $X = \{x_1, \ldots, x_\alpha\}$, where $\alpha = m^2 l(m + l + 2)$ and β nodes denoted by $Y = \{y_1, \ldots, y_\beta\}$, where $\beta = m^2 l(k + l + 2)$. Lastly, we create four additional nodes, denoted by t, z, w_1 and w_2.

- **The set of edges:** First, we create the edges (t, z) and (w_1, w_2). Then, for every node x_i we create an edge (x_i, t), for every node y_i we create an edge (y_i, z). For every node $S_i \in S$ we create an edge (S_i, z), an edge (S_i, w_1), and an edge (S_i, u_j) for every $u_j \in S_i$. For every node $u_i \in U$ we create an edge (u_i, w_2). We also create edges such that the nodes in X form a clique, *i.e.*, we create an edge (x_i, x_j) for every $x_i, x_j \in X$, as well as edges such that the nodes in Y form a clique, *i.e.*, we create an edge (y_i, y_j) for every $y_i, y_j \in Y$.

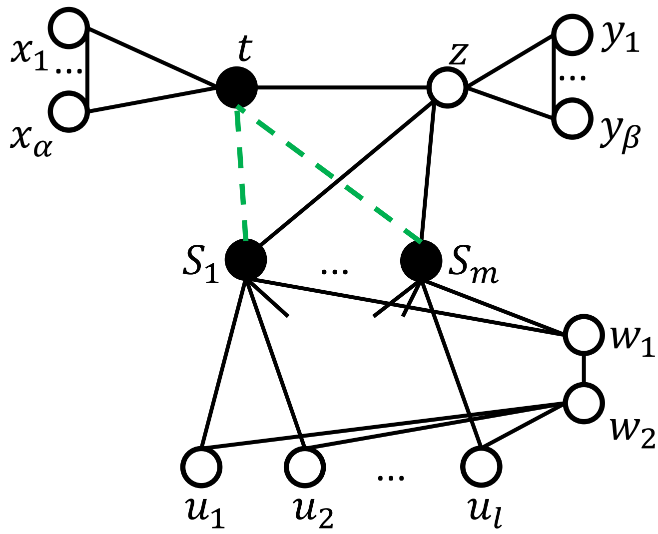

Figure 4.3: An illustration of the network used in the NP-completeness proof of the problem of Hiding Leaders given the betweenness centrality.

An example of the resulting network, G, is illustrated in Figure 4.3.
Now, consider instance $(G, L, b, c, d, \hat{A}, \hat{R})$ of the problem of Hiding Leaders, where:

- G is the network we just constructed;
- $L = \{z, w_1, w_2\} \cup U \cup X \cup Y$;
- $b = k$, where k is the parameter of the 3-Set Cover problem (where the goal is to determine whether there exist k elements of S the union of which equals U);
- c is the betweenness centrality measure, *i.e.*, we have that:

$$c(G, v) = \frac{2}{(n-1)(n-2)} \sum_{v_j, v_k \in V \setminus \{v_i\}} \frac{|\{p \in \Pi(v_j, v_k) : v_i \in p\}|}{|\Pi(v_j, v_k)|};$$

- $d = 1$;
- $\hat{A} = \{(t, S_i) : S_i \in S\}$;
- $\hat{R} = \emptyset$.

From the definition of the problem of Hiding Leaders, we see that the only edges that can be added to the graph are the edges between t and the members of S, *i.e.*, $A^* \subseteq \hat{A}$, where $\hat{A} = \{(t, S_1), \ldots, (t, S_m)\}$. Notice that any such choice of A^* corresponds to selecting a subset of $|A^*|$ elements of S in the 3-Set Cover problem. In what follows, we will show that a solution to the above instance of Hiding Leaders corresponds to a solution to the 3-Set Cover problem.

First, we show that for every $v \in V \setminus \{t, z\}$ and every $A^* \subseteq \hat{A}$ we have $c(G', v) < c(G', t)$, where $G' = (V, E \cup A^*)$. To this end, let $B(G', v)$ denote the sum of percentages of all short paths between all other pairs of nodes that are controlled by v, *i.e.*, $B(G', v) = \sum_{w,w' \in V\setminus\{v\}} \frac{|\{p \in \Pi(w,w') : v \in p\}|}{|\Pi(w,w')|}$. Note that $B(G', v) = \frac{(n-1)(n-2)}{2} c(G', v)$ We show that the following holds:

$$\forall_{v \in V\setminus\{t,z\}} \forall_{A^* \subseteq \hat{A}} B(G', v) < B(G', t)$$

Note that since t controls all shortest paths between nodes in X and nodes in $\{z, w_1, w_2\} \cup Y \cup S \cup U$, we have that:

$$B(G', t) \geqslant \alpha(\beta + m + l + 3) \geqslant m^4 l^3 (m + l + 2) + m^2 l (m + l + 2)^2$$

Also, notice that since $\alpha = m^2 l(m+l+2)$, $\beta = m^2 l(k+l+2)$, and $k < m$, we have that:

$$\alpha + \beta < 2m^2 l(m + l + 2)$$

For nodes other than t we have:

- $B(G', x_i) = B(G', y_i) = 0 < B(G', t)$, as nodes in $X \cup Y$ do not control any shortest paths.
- $B(G', w_1) \leqslant (\alpha + \beta + m + 2) + \frac{m(m-1)}{2} + ml \leqslant 2m^2 l(m + l + 2) + m^2 + m + ml < (2m^2 l + m)(m + l + 2) < B(G', t)$, because w_1 controls some shortest paths between w_2 and nodes in $\{t, z\} \cup X \cup Y \cup S$ (there are $\alpha + \beta + m + 2$ such pairs of these nodes), some shortest paths between pairs of nodes in S (there are at most $\frac{m(m-1)}{2}$ such pairs of these nodes), and some shortest paths between nodes in U and nodes in S (there are at most ml such pairs of these nodes).
- $B(G', w_2) \leqslant \frac{l(l-1)}{2} + l + ml < \frac{l^2+l}{2} + ml < B(G', t)$, because w_2 controls some shortest paths between pairs of nodes in U (there are at most $\frac{l(l-1)}{2}$ such pairs of these nodes), some shortest paths between nodes in U and w_1 (there are at most l such pairs of these nodes), and some shortest paths between nodes in U and nodes in S (there are at most ml such pairs of these nodes)/
- $B(G', u_i) \leqslant (\alpha + \beta + m + 2) + \frac{m(m-1)}{2} < B(G', t)$, because u_i controls some shortest paths between w_2 and nodes in $\{t, z\} \cup X \cup Y \cup S$ (there are $\alpha + \beta + m + 2$ such pairs of these nodes), and some shortest paths between pairs of nodes in S (there are at most $\frac{m(m-1)}{2}$ such pairs of these nodes). Proof follows the same as for $B(G', w_1)$/
- $B(G', S_i) \leqslant 3(\alpha + \beta + l + m + 2) + l + 2(\alpha + \beta + 2) \leqslant 5(\alpha + \beta + l + m + 2) \leqslant (10m^2 l + 5)(m + l + 2) < B(G', t)$, because S_i controls some shortest paths between nodes in U connected to S_i and nodes in $\{t, z\} \cup X \cup Y \cup S \cup U$ (there are at most $3(\alpha + \beta + l + m + 2)$ such pairs of these nodes), some shortest paths between w_1 and nodes in U (there are at most l such pairs of these nodes), and some of the shortest paths between nodes in $\{w_1, w_2\}$ and nodes in $\{t, z\} \cup X \cup Y$ (there are at most $2(\alpha + \beta + 2)$ such pairs of these nodes).

Therefore, either t or z has the highest betweenness centrality. Since $z \in L$ and $t \in F$, then $A^* \subseteq \hat{A}$ is a solution to the problem of Hiding Leaders if and only if $B(G', t) > B(G', z)$. We now compute the values of $B(G', t)$ and $B(G', z)$.

We have that:

$$B(G',t) = \alpha(\beta+m+l+3) + \sum_{\substack{S_i,S_j\in S:\\(t,S_i)\in E\wedge(t,S_j)\in E}} \frac{1}{|N(S_i,S_j)|} + \sum_{S_i\in N(t)} \sum_{u_j\in U\setminus N(S_i)} \frac{|N(t,u_j)|}{|N(t,u_j)|+|N(z,u_j)|+1}$$

as t controls all shortest paths between nodes in X and all other nodes (there are $\alpha(\beta+m+l+3)$ such pairs), one shortest path between each pair of neighbouring nodes in S, and paths between neighbouring nodes in S and nodes in U that T has common neighbours with (other paths run through z and nodes in S, or through w_1 and w_2).

On the other hand, we have that:

$$B(G',z) = \beta(\alpha+m+l+3) + \sum_{S_i,S_j\in S} \frac{1}{|N(S_i,S_j)|} + \sum_{S_i\in S} \sum_{u_j\in U\setminus N(S_i)} \frac{|N(z,u_j)|}{|N(t,u_j)|+|N(z,u_j)|+1} + \sum_{S_i\notin N(t)} (\alpha+1) + \sum_{u_i\in U:N(t,u_i)=\emptyset} (\alpha+1)$$

as z controls all shortest paths between nodes in Y and all other nodes (there are $\beta(\alpha+m+l+3)$ such pairs), one shortest path between each pair of nodes in S, paths between nodes in S and nodes in U, and all shortest paths between $\{t\}\cup X$ and nodes $\{S_i\in S: S_i\notin N(t)\}\cup\{u_i\in U: N(t,u_i)=\emptyset\}$.

Therefore we have that:

$$B(G',z)-B(G',t) = (\beta-\alpha)(m+l+3) + \sum_{\substack{S_i,S_j\in S:\\(t,S_i)\notin E\vee(t,S_j)\notin E}} \frac{1}{|N(S_i,S_j)|} + \Delta SU + \sum_{S_i\notin N(t)} (\alpha+1) + \sum_{u_i\in U:N(t,u_i)=\emptyset} (\alpha+1)$$

where:

$$\Delta SU = \sum_{S_i\in S} \sum_{u_j\in U\setminus N(S_i)} \frac{|N(z,u_j)|}{|N(t,u_j)|+|N(z,u_j)|+1} - \sum_{S_i\in N(t)} \sum_{u_j\in U\setminus N(S_i)} \frac{|N(t,u_j)|}{|N(t,u_j)|+|N(z,u_j)|+1}.$$

Notice that $B(G',z)$ gets lower with the number of edges in A^* and with the number of $u_i\in U$ such that $\exists_{S_j\in N(t)} u_i\in S_j$. We will now prove that:

1. If $|A^*|=k$ and for every $u_i\in U$ there exists $S_j\in N(t)$ such that $u_i\in S_j$, then $B(G',z)<B(G',t)$;

2. If $|A^*|=k$ and there exists $u_i\in U$ such that for every $S_j\in N(t)$ we have $u_i\notin S_j$, then $B(G',z)>B(G',t)$.

To prove the first inequality, we have that in this case:

$$B(G',z) - B(G',t) =(\beta-\alpha)(m+l+3) + \sum_{\substack{S_i,S_j \in S: \\ (t,S_i)\notin E \vee (t,S_j)\notin E}} \frac{1}{|N(S_i,S_j)|} + \Delta SU + (m-k)(\alpha+1)$$

Since $|\{S_i, S_j \in S : (t, S_i) \notin E \vee (t, S_j) \notin E\}| = \frac{m(m-1)-k(k-1)}{2} = \frac{(m-k)(m+k-1)}{2}$, and $|N(S_i, S_j)| \geqslant 2$ we have that:

$$B(G',z) - B(G',t) \leqslant (\beta-\alpha)(m+l+3) + (m-k)(\alpha + 1 + \frac{\Delta SU}{m-k} + \frac{m+k-1}{4}).$$

By substituting values of α and β, and observing that $\Delta SU < ml$ and $k < m$, we have that:

$$B(G',z) - B(G',t) < m^2 l(k-m)(m+l+3) + (m-k)(m^2 l(m+l+2) + 1 + ml + 2m - 1),$$

that gives us:

$$B(G',z) - B(G',t) < (k-m)m^2 l + (m-k)(ml+2m) = (k-m)m(ml-l-2) < 0.$$

Hence, we have that if $|A^*| = k$ and for every $u_i \in U$ there exists $S_j \in N(t)$ such that $u_i \in S_j$, then $B(G',z) < B(G',t)$.

To prove the second inequality, since there exists $u_i \in U$ such that for every $S_j \in N(t)$ we have $u_i \notin S_j$, we have that :

$$B(G',z) - B(G',t) \geqslant (\beta-\alpha)(m+l+3) + \sum_{\substack{S_i,S_j \in S: \\ (t,S_i)\notin E \vee (t,S_j)\notin E}} \frac{1}{|N(S_i,S_j)|} + \Delta SU + (m-k)(\alpha+1) + (\alpha+1).$$

Since $\sum_{\substack{S_i,S_j \in S: \\ (t,S_i)\notin E \vee (t,S_j)\notin E}} \frac{1}{|N(S_i,S_j)|} > 0$ and $\Delta SU > 0$, we have that:

$$B(G',z) - B(G',t) > (\beta-\alpha)(m+l+3) + (m-k+1)(\alpha+1).$$

By substituting values of α and β we have that:

$$B(G',z) - B(G',t) > m^2 l(k-m)(m+l+3) + (m-k+1)(m^2 l(m+l+2) + 1),$$

that gives us:

$$B(G',z) - B(G',t) > m^2 l(k-m) + m^2 l(m+l+2) = m^2 l(k+l+2) > 0.$$

Hence, we have that if $|A^*| = k$ and there exists $u_i \in U$ such that for every $S_j \in N(t)$ we have $u_i \notin S_j$, then $B(G',z) > B(G',t)$.

Therefore the solution to the problem of Hiding Leaders corresponds to a solution to the given instance of the 3-Set Cover problem, which concludes the proof. $\square$

We now move to a setting, where instead of modifying an existing network, we want to build a new structure, already with satisfying properties.

4.4 Constructing a Network

In the previous section we proved the NP-completeness of modifying an existing network in order to hide its leaders. However, in certain cases the leaders might prefer to develop a new covert network (*e.g.*, a subnetwork in a new region) rather than to modify an existing one.

In this section we show that it is possible to efficiently create a network designed specifically to hide its leaders without limiting their ability to influence the other nodes. We call this structure the *captain network*.

The intuition of its main principles is as follows. First the leader nodes, L, form a clique, to provide the best possible communication among them. Each leader $l_i \in L$ is then assigned a group of k *captains*, $C_i = \{c_{i,1}, \ldots, c_{i,k}\}$, that are connected to that leader. All captains are then connected into a complete $|L|$-partite graph. A captain, $c_{i,j}$ servers two purposes: the first is to conceal the leaders in L, by ensuring that it is ranked higher than each of them (according to the rankings of the three standard centrality measures); the second purpose of $c_{i,j}$ is to pass on the influence of l_i to the rest of the network. The remaining nodes, the set of which is denoted $X = \{x_1, \ldots, x_m\}$, are each connected to one captain from each group. Note that the set of followers in this network is $F = X \cup C_1 \cup \ldots \cup C_{|L|}$ Figure 4.4 illustrates a sample captain network with $|L| = 3$.

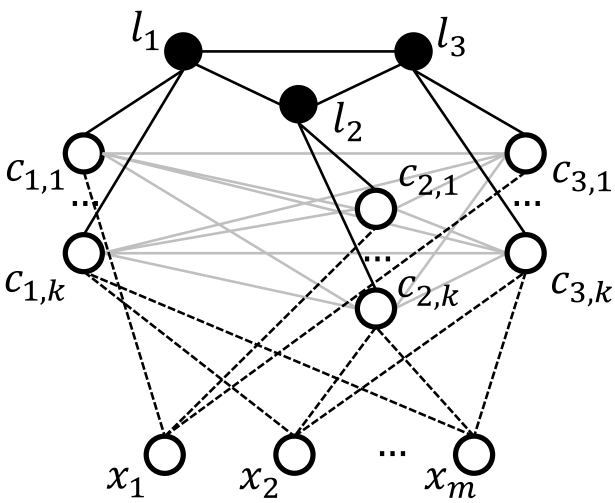

Figure 4.4: An illustration of the captain network with $|L| = 3$. Edges including leaders are depicted as solid black lines; edges between captains are depicted as gray lines; edges between captains and other nodes are depicted as dashed lines.

Note that if the above steps are followed for the case of a single leader, the result would be a tree structure. While a tree is a fairly common organizational structure, it does not provide adequate hiding of the leader, as the leader can be identified as a root of the tree. With this in mind, whenever there is a single leader, we create two groups of captains to avoid the tree structure. The resulting structure is illustrated in Figure 4.5. Algorithm 2 presents formally the steps required to create a captain network for a given number of leaders.

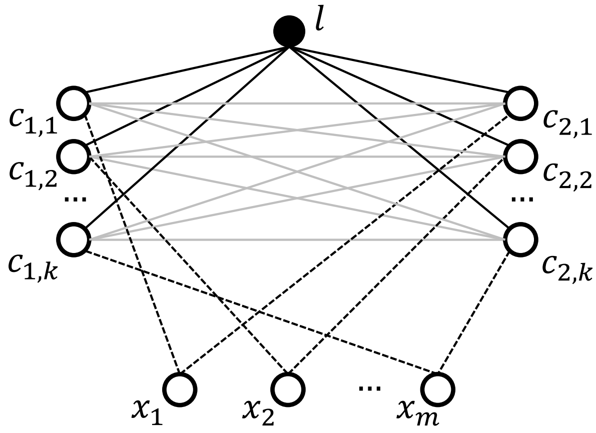

Figure 4.5: An illustration of the captain network with one leader. Edges including leaders are depicted as solid black lines; edges between captains are depicted as gray lines; edges between captains and other nodes are depicted as dashed lines.

We now prove that every captain has higher centrality values than each of the leaders.

Theorem 8. *Given a captain network, let* $r = \left\lfloor \frac{m}{k} \right\rfloor$ *denote the minimal number of connections that a captain,* $c_{i,j}$*, has with nodes from* X*. Then, if either (*$|L| \geqslant 2$ *and* $r \geqslant 1$*), or (*$|L| = 1$ *and* $k < r+1$*), then every captain has greater degree, closeness and betweenness centrality values than each of the leader nodes.*

Proof. Starting with degree centrality and multiple leaders, the degree centrality of a leader node, l, is $c_{degr}(G,l) = \frac{|L|+k-1}{n-1}$, since it is only connected to other leaders and captains from its group. On the other hand, the degree centrality of a captain, $c_{i,j}$, is $c_{degr}(G,c_{i,j}) \geqslant \frac{1+k(|L|-1)+r}{n-1}$, since it is connected to one of the leader nodes, to all captains from other groups, and to at least r other nodes from X. As such, we have:

$$c_{degr}(G,c_{i,j}) - c_{degr}(G,l) \geqslant \frac{1+k(|L|-1)+r-(|L|+k-1)}{n-1},$$

that gives us:

$$c_{degr}(G,c_{i,j}) - c_{degr}(G,l) \geqslant \frac{(|L|-2)(k-1)+r}{n-1}$$

Therefore, since $|L| \geqslant 2$, $k \geqslant 1$, and $r \geqslant 1$, we have that $c_{degr}(G,c_{i,j}) > c_{degr}(G,l)$ for any $c_{i,j}$.

As for the case with a single leader, the degree of the leader node, l, is $c_{degr}(G,l) = \frac{2k}{n-1}$, since it is only connected to captains from both groups. On the other hand, the degree of a captain $c_{i,j}$, is $c_{degr}(G,c_{i,j}) \geqslant \frac{1+k+r}{n-1}$, since it is connected to the leader node, to all captains from other group, and to at least r members. As such, we have:

$$c_{degr}(G,c_{i,j}) - c_{degr}(G,l) \geqslant \frac{1+k+r-2k}{n-1},$$

that gives us:

$$c_{degr}(G,c_{i,j}) - c_{degr}(G,l) \geqslant \frac{1+r-k}{n-1}$$

Algorithm 2 The construction of a captain network

Input: The set of leaders $L = \{l_1, \dots, l_{|L|}\}$, the set of followers $F = \{f_1, \dots, f_{|F|}\}$, the number of captains in each group k (where $1 \leqslant k \leqslant \frac{|F|}{|L|}$).

Output: The set of edges E that constitutes the captain network

```
g ← max(2, |L|)
for i = 1, ..., g do
    for j = 1, ..., k do
        c_{i,j} ← f_{(i−1)k+j}
        C_i ← C_i ∪ {c_{i,j}}
X ← F \ ⋃_{i=1}^{g} C_i
for l_i, l_j ∈ L do
    E ← E ∪ {(l_i, l_j)}
for l_i ∈ L do
    for c_{i,j} ∈ C_i do
        E ← E ∪ {(l_i, c_{i,j})}
if |L| = 1 then
    for c_{2,j} ∈ C_2 do
        E ← E ∪ {(l_1, c_{2,j})}
for C_i ≠ C_j do
    for c ∈ C_i do
        for c' ∈ C_j do
            E ← E ∪ {(c, c')}
j ← 0
for x ∈ X do
    for i = 1, ..., g do
        E ← E ∪ {(x, c_{i,j})}
    j ← (j + 1) mod k
```

Therefore, we have that $c_{degr}(G, c_{i,j}) > c_{degr}(G, l)$ for $k < r + 1$.

Moving on to the closeness centrality, for any given node v, this centrality depends inversely on the sum of the lengths of shortest paths from v to every other node, *i.e.*, $\sum_{w \in V} d(v, w)$. For every leader and every captain, the distance to every other node is either 1 or 2. More precisely, for every $v \in V$, we have: $\sum_{w \in V} d(v, w) = 1|N(v)| + 2(n - |N(v)|) = 2n - |N(v)|$. Consequently, whenever all captains have greater degree centrality than all leaders, they must also have greater closeness centrality. Since we have already proven this fact for the degree centrality, then this implies that $c_{clos}(G, c_{i,j}) > c_{clos}(G, l)$.

Finally, regarding the betweenness centrality, let $B(v)$ denote $\sum_{u,w \in V \setminus \{v\}} \frac{|\{p \in \Pi(u,w) : v \in p\}|}{|\Pi(u,w)|}$. Then the betweenness centrality of a node $v \in V$ can be written as:

$$c_{betw}(G, v) = \frac{2}{(n-1)(n-2)} B(v).$$

For a network with multiple leaders, every leader node l belongs to one of $(|L|-1)k+1$ shortest paths between the pairs of captains from its group (alternative shortest paths run through captains from other groups), as well as one of $k + 1$ shortest paths between each captain from its group and all other leaders (alternative shortest paths run through captains from the group of the chosen leader). Since the leader node l belongs to no other shortest paths, we have:

$$B(l) = \frac{k(k-1)}{2((|L|-1)k+1)} + \frac{k(|L|-1)}{k+1}$$

Having analysed $B(l)$, let us now analyse $B(c_{i,j})$ for captain $c_{i,j}$. In particular, since $c_{i,j}$ belongs to one of $(|L|-1)k+1$ shortest paths between pairs of captains from all other groups, as well as one of $k+1$ shortest paths between each of the captains from the other groups and the leader of its group, and some shortest paths between neighbouring nodes from X and the leader of its group, we have:

$$B(c_{i,j}) > \frac{(|L|-1)k(k-1)}{2((|L|-1)k+1)} + \frac{k(|L|-1)}{k+1}$$

Therefore, we have that $B(c_{i,j}) > B(l)$, which results in $c_{betw}(G, c_{i,j}) > c_{betw}(G, l)$.

For a network with a single leader, the leader node l belongs to one of the $k+1$ shortest paths between the pairs of captains from each group (alternative shortest paths run through captains from the other group). Since the leader node l belongs to no other shortest paths, we have:

$$B(l) = \frac{k(k-1)}{(k+1)}$$

Having analysed $B(l)$, let us now analyse $B(c_{i,j})$ for captain $c_{i,j}$. In particular, since $c_{i,j}$ belongs to one of the $k+1$ shortest paths between pairs of captains from the other group, and to the only shortest path between the member nodes connected to it and the captains from the other group, we have:

$$B(c_{i,j}) > \frac{k(k-1)}{2(k+1)} + rk = \frac{k(k-1) + 2rk(k+1)}{2(k+1)}$$

Therefore, since $2rk(k+1) > k(k-1)$, we have that $B(c_{i,j}) > B(l)$, which implies that $c_{betw}(G, c_{i,j}) > c_{betw}(G, l)$. □

As stated in Theorem 8, a captain network can indeed conceal its leaders as far as centrality is concerned. On the other hand, as far as influence is concerned, we evaluate the network empirically to see how the different parameters affect the influence of the leaders. To this end, given a captain network with 400 nodes, we varied the parameters of the network, namely k (the size of each captain group) and $|L|$ (the number of leaders) for a network with multiple leaders. For every pair of parameters, we measure the difference in centrality between a leader node, and each captain (the greater the difference, the greater the leaders' disguise), and measured the influence of a leader to see how this influence is affected by the disguising process. When measuring the influence, we used the Independent Cascade model with probability 0.15 on each edge, and the Linear Threshold model with the threshold value sampled uniformly at random.

The results are depicted in Figures 4.6 and 4.7. Both figures should be read as follows. The x-axis represents the number of captains in each group. The y-axis represents the number of leaders of the network. The more intense the color in Figure 4.6, the higher

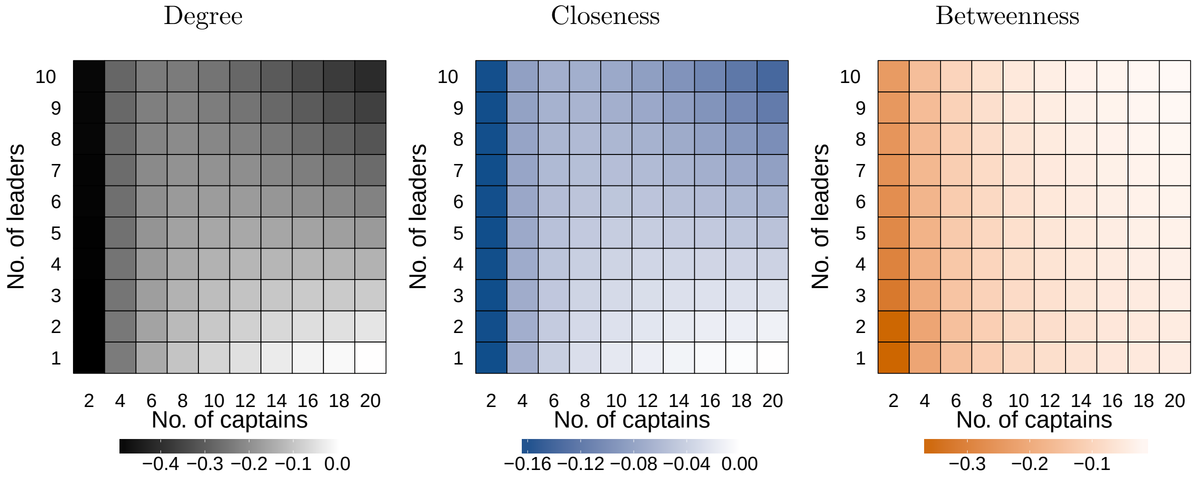

Figure 4.6: Given a captain network of 400 nodes, with different number of captains in each group (the x-axis) and number of leaders (the y-axis), the figure depicts the difference in centrality between a leader and a captain.

the difference in centrality between a leader node and a captain, and the safer the leader. The more intense the color in Figure 4.7, the higher the influence of a leader node.

Roughly speaking, the results can be categorized into three groups:

- *small k*: This yields relatively high levels of disguise in terms of degree, closeness, and betweenness centralities. On the other hand, it yields rather low levels of Independent-Cascade influence and Linear-Threshold influence.

- *large k and small $|L|$*: This yields relatively low levels of disguise in terms of degree, closeness and betweenness centralities. On the other hand, it yields relatively high levels of Linear-Threshold influence, but not Independent-Cascade influence.

- *large k and large $|L|$*: This yields relatively high levels of disguise in terms of degree and closeness centralities, but not the betweenness centrality. On the other hand, it yields relatively high levels of Independent-Cascade influence, but not Linear-Threshold influence.

4.5 Concluding Remarks

The model studied in this chapter offers new insights into the secrecy-efficiency trade-off faced by the covert organizations. The novelty of our approach comes from our definition of secrecy, which assumes that the members of a terrorist network act strategically to evade detection by centrality measures. Indeed, it is well established that centrality measures belong to the key social network analysis tools used to analyse covert networks. Unfortunately, centrality measures—like most other social network analysis tools—were designed to analyse social networks among members of the general public, rather than among members of covert organizations who are well aware of the possibility of attracting unwanted attention from the authorities.

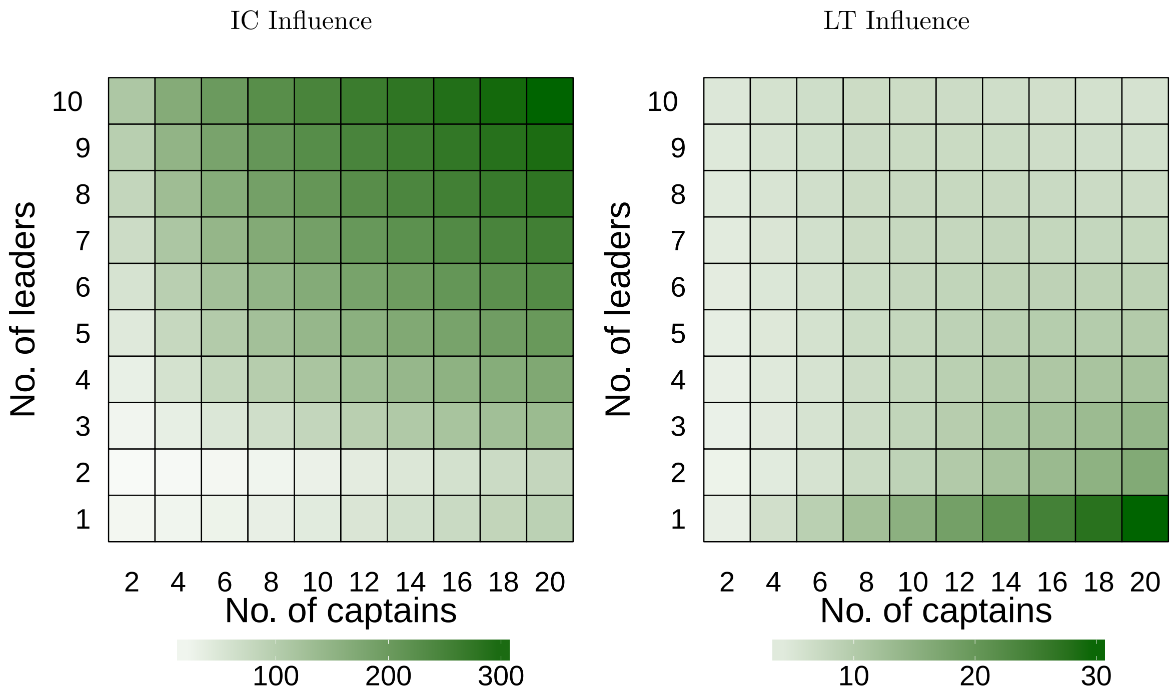

Figure 4.7: Given a captain network of 400 nodes, with different number of captains in each group (the x-axis) and number of leaders (the y-axis), the figure depicts the influence value of a leader.

However, recent findings—especially with the respect to the tech-savvy ISIS—clearly demonstrate that such an assumption may be incorrect. The known evading techniques used by ISIS range from changing aliases and keeping own profile private [109] to using encrypted communication platforms [74] and staging an entire group disappearance from the social media only to pop up again in a different place and under alternative aliases [109]. In fact, it has been claimed that the evasion capabilities of ISIS significantly increased after Edward Snowden's disclosure of classified information on the social network analysis techniques used by US intelligence [129].

We now describe a number of limitations of our study that also show directions in which our model can be extended. First, we only consider possible strategies of the evaders, *i.e.*, the members of covert organization, and we assume that the seeker, *i.e.*, the party who is using centrality measures to identify key terrorists, is unaware of any potential strategic efforts of the evaders. It would be interesting to see new social network analysis tools, and centrality measures in particular, that are immune (at least to some extent) against such evasion techniques. That includes considering an opposite problem, *i.e.*, assuming the perspective of the seeker who intends to identify the nodes that try to maintain low positions in centralities rankings. For example, nodes whose centrality can be greatly increased by modifying just a few edges may be suspected of having performed hiding process in the past.

Second, our model assumes that the seeker's knowledge is restricted to the topology of the network. The motivation behind this assumption is twofold: (a) the most fundamental social network analysis tools—including the ones studied in this dissertation—are all based solely on the topology of the network; and (b) the exclusion of domain knowledge makes the model more general, as it can be applied to any network. Nevertheless, there may be cases where the seeker has additional, domain-dependent information that might be used in conjunction with social network analysis tools, *e.g.*, as in the case of covert networks [100]. In such cases, further investigation is needed to understand the extent to which the evaders can protect themselves against the seeker.

Third, in the construction our captain networks we focus solely on the matter of assuring safety of the leaders in terms of low positions in centrality rankings. Interesting direction for future research is development of network structures that incorporate also other features, *e.g.*, topologies that not only provide low centrality of the leaders, but are also resilient to potential attacks. Another possible desired network quality is for it to be similar to another network structure, *e.g.*, to manifest small-world qualities. This type of communication structure would be easier to hide in plain sight as a part of another social network.

Finally, another interesting direction is to investigate whether there exist special classes of networks for which the problem of Hiding Leaders can be easily solved, *e.g.*, trees or networks with small treewidth.

Chapter 5

Hiding Communities

In this chapter we analyse the problem of preventing a group of nodes to be identified as a community by a community detection algorithm. We present the formal definition of the problems, analyse the complete information setting, introduce the first measure of how well the community is hidden in a community structure and provide heuristic solution to the hiding problem.

5.1 Introduction

While there exists no precise definition of a community, typically it is understood as a group of nodes who are more densely connected among themselves than with the rest of the network.[1] Various dedicated community detection algorithms have been proposed in the literature (see Section 5.2 for an overview). Such algorithms return a community structure, *i.e.*, a partition of the set of nodes into typically-disjoint communities. In this context, the modularity index is a well-known measure of the quality of a particular community structure. As such, any community detection algorithm searches through the space of community structures to select the one that has a particularly high modularity.

In this chapter, we investigate whether and how a community could hide itself within a network so that it becomes difficult to identify by various community-detection algorithms. Our motivation is twofold. Firstly, a community may be simply interested in privacy—a value that seems to be increasingly violated by the process of datafication. Consider, for instance, police units that are involved in undercover operations against street gangs. The private life of this community should not be easily traceable on Facebook or Twitter. What this lack of privacy might lead to has been recently shown by various Ukrainian activists groups who have identified Russian military units in Donbass by analysing social networking sites [3]. As yet another example, staying "below the radar" is imperative for communities of opposition blogers in authoritarian regimes, because such regimes actively monitor internet content [77]. Our second motivation is related to security. In particular, methods of social network analysis are becoming increasingly used in the fight against criminal organisations [152]. Hence, it is important to understand how various community detection algorithms could be "fooled" by such covert organizations.

[1] See the work by Fortunato [51] for a detailed discussion of alternative ways to define communities.

We study how a community can conceal itself to increase the likelihood of being overlooked by community-detection algorithms. To this end, we propose a measure of concealment designed to quantify the degree to which a group of individuals is hidden. Using this measure, we demonstrate the effectiveness of yet another simple heuristic, whereby members of the community either "unfriend" certain other members, or "befriend" some non-members, in order to blend in with the surrounding web of social connections.

The remainder of this chapter is organised as follows. In Section 5.2 we present community detection algorithms and basic concepts used in this chapter. In Section 5.3 we describe the problem of lowering modularity of a chosen community structure. In Section 5.4 we propose the first measure of how well a community is hidden within a community structure. In Section 5.5 we present and test a simple heuristic solution, allowing community members to better hide the fact of their cooperation in a social network. We conclude with discussion of the results and ideas for future work in Section 5.6.

5.2 Preliminaries

We consider a *community structure*, $\Gamma = \{C_1, \ldots, C_k\}$, to be a partition of the set of nodes into disjoint subsets, or *communities*.[2] Formally, it satisfies the following three conditions: $\forall_{C_i \in \Gamma} C_i \subseteq V$, $\bigcup_{C_i \in \Gamma} C_i = V$, and $\forall_{C_i, C_j \in \Gamma} C_i \cap C_j = \emptyset$.

We denote the set of edges between the members of the communities C_i and C_j by $E_G(C_i, C_j)$, *i.e.*, $E_G(C_i, C_j) = \{(v, w) \in E : v \in C_i, w \in C_j\}$. We denote the set of edges between the members of C_i by $E_G(C_i)$, *i.e.*, $E_G(C_i) = E_G(C_i, C_i)$. Finally, we denote the sum of degrees of nodes in C_i by $\delta_G(C_i)$, *i.e.*, $\delta_G(C_i) = \sum_{v \in C_i} \delta_G(v)$; this is called the *degree of community* C_i. Notice, that $\delta_G(C_i) = 2|E(C_i)| + \sum_{C_j \neq C_i} |E(C_i, C_j)|$.

Modularity [108] is the most widely used measure of quality of a given community structure. Modularity of a community structure Γ in a network G can be expressed as [23]:

$$Q_G(\Gamma) = \sum_{C_i \in \Gamma} \left(\frac{|E_G(C_i)|}{|E|} - \left(\frac{\delta_G(C_i)}{2|E|} \right)^2 \right).$$

The intuition behind the modularity measure is the preference of community structures, where the most edges are between the nodes in the same community (component $\frac{|E_G(C_i)|}{|E|}$). However, this criterion is always maximized by a community structure consisting of a single community, *i.e.*, $\Gamma = \{V\}$. Hence the need of balancing the modularity value by subtracting the sum of squares of communities degrees (component $\left(\frac{\delta_G(C_i)}{2|E|}\right)^2$).

A *community detection algorithm* is an algorithm that, given a network G, returns a community structure of the nodes of G. Formally, it is a function $\rho : \mathbb{G} \to 2^{2^V}$, where $\rho(G)$ satisfies the three conditions of being a community structure for any G, *i.e.*, $\forall_{C_i \in \Gamma} C_i \subseteq V$, $\bigcup_{C_i \in \Gamma} C_i = V$, and $\forall_{C_i, C_j \in \Gamma} C_i \cap C_j = \emptyset$. The problem of finding a community structure with maximal modularity is NP-complete [24], and there exist many algorithms designed to find a solution of acceptable quality in a reasonable time.

In this dissertation we consider seven community detection algorithms implemented in the igraph [39] package of the R programming language, version 1.0.1. These are:

[2]Some works consider overlapping community structures [151]. However, as is common in the literature, we restrict our attention to disjoint ones.

- Betweenness [108], based on iterative removal of edges with largest number of shortest paths running through them. Communities correspond to connected components of the resulting network.

- Greedy [32], based on computing local modularity optima, starting with each node as a separate community and merging communities that provide the highest modularity gain.

- Walktrap [119], based on a tendency of random walks to stay within a single community, rather than to move to another.

- Eigenvector [107], recursively splitting nodes into two groups. Each split is determined by the signs in the eigenvector corresponding to the highest eigenvalue of the adjacency matrix of the network.

- Louvain [18], based on the multi-level modularity optimization. When a local optimum is achieved by greedy merging, each community is treated as a single node and the process continues for this new network.

- Infomap [125], based on compressing a description of the probability flow, intended to model the information flow in the network.

- Spinglass [123], based on physical interpretation of the community structure, where each node is an atom in a disordered magnet and the assignment to the particular community is its spin. The algorithm intends to minimize the energy of such system.

Lancichinetti and Fortunato [83] provide a comparison of various community detection algorithms. According to their analysis, Infomap [125] algorithm has the best performance of the considered set of algorithms. Their benchmark is the set of random graphs, where the underlying community structure is known.

5.3 Minimizing Modularity

In this section, we focus on the case where the community $C^\dagger$ knows the exact community-detection algorithm that the adversary will use, and knows that the algorithm will return a particular community structure Γ such that $C^\dagger \in \Gamma$. Knowing this, the goal of $C^\dagger$ is to rewire the network such that the modularity of Γ is minimized, with the hope that if the aforementioned algorithm is used, Γ will not be detected. Here, we assume that $C^\dagger$ has a budget, $b \in \mathbb{N}$, which specifies the maximum number of edges that $C^\dagger$ can afford to modify, *i.e.*, add or remove. More formally, our problem is defined as follows.

Definition 5 (Minimizing Modularity)**.** *Let $G = (V, E)$ be a network, and let Γ be a community structure. Then, given a community $C^\dagger \in \Gamma$ and a budget $b \in \mathbb{N}$, the Minimizing Modularity problem is to find a set of edges to be added $A^* \subseteq \bar{E}$ and another to be removed $R^* \subseteq E$ such that $|A^*| + |R^*| \leqslant b$ and $G^* = (V, (E \cup A^*) \setminus R^*)$ is in:*

$$\operatorname*{arg\,min}_{G' \in \left\{(V,(E\cup A)\setminus R): A\subseteq\bar{E}, R\subseteq E, |A|+|R|\leqslant b\right\}} Q_{G'}(\Gamma).$$

In our analysis, we study the problems of adding edges and removing edges separately.

As it turns out, when considering the problem of adding a single edge to lower modularity in an optimal way, the best choice is adding an edge between two communities with highest degrees.

Theorem 9. *Let $G = (V, E)$ be a network, and $\Gamma = \{C_1, \dots, C_k\}$ be a community structure with communities sorted in decreasing order accordingly to their degree, i.e., $\forall_{1 \leqslant i < k} \delta_G(C_i) \geqslant \delta_G(C_{i+1})$. Then, for any two unconnected nodes, $v_i \in C_1$ and $v_j \in C_2$, we have:*

$$(v_i, v_j) \in \arg\min_{e \in \bar{E}} Q_{\{V, E \cup \{e\}\}}(\Gamma).$$

Proof. Let us first introduce the following notation: $\zeta_G(\Gamma) = \sum_{C_i \in \Gamma} |E_G(C_i)|$, and $\lambda_G(\Gamma) = \sum_{C_i \in \Gamma} \delta_G(C_i)^2$. Let m denote the number of edges in the graph, *i.e.*, $m = |E|$. With this notation in hands, modularity $Q_G(\Gamma)$ can be expressed as follows:

$$Q_G(\Gamma) = \frac{\zeta_G(\Gamma)}{m} - \frac{\lambda_G(\Gamma)}{4m^2}.$$

Now, let G_i^+ denote the network that results from adding to G an edge $(v, w) : v, w \in C_i$. Likewise, let G_{ij}^+ denote the network that results from adding to G an edge $(v, w) : v \in C_i, w \in C_j$, where $i \neq j$. Next, we show how to compute the modularity of Γ in each of these networks.

Let us start by analysing the effect of adding to G an edge $(v, w) : v, w \in C_i$. This increases the number of intra-community edges in C_i by 1 (implying that $\zeta_{G_i^+}(\Gamma) = \zeta_G(\Gamma) + 1$), and increases the sum of degrees of the members of C_i by 2 (thus $\delta_{G_i^+}(C_i) = \delta_G(C_i) + 2$ and $\lambda_{G_i^+}(\Gamma) = \lambda_G(\Gamma) - \delta_G(C_i)^2 + (\delta_G(C_i) + 2)^2$). Consequently, we have:

$$Q_{G_i^+}(\Gamma) = \frac{\zeta_G(\Gamma) + 1}{m + 1} - \frac{\lambda_G(\Gamma) + 4\delta_G(C_i) + 4}{4(m + 1)^2}.$$

As can be seen, $Q_{G_i^+}(\Gamma)$ decreases with $\delta_G(C_i)$, implying that: $C_1 \in \arg\min_{C_i \in \Gamma} Q_{G_i^+}(\Gamma)$.

Let us move on to analysing the effect of adding to G an edge (v, w) where $v \in C_i$, $w \in C_j$, and $i \neq j$. This causes an increase in the sum of degrees of the members of C_i and the members of C_j by 1 each (*i.e.*, $\delta_{G_{ij}^+}(C_i) = \delta_G(C_i) + 1$ and $\delta_{G_{ij}^+}(C_j) = \delta_G(C_j) + 1$), which in turn implies that: $\lambda_{G_{ij}^+}(\Gamma) = \lambda_G(\Gamma) - \delta_G(C_i)^2 + (\delta_G(C_i) + 1)^2 - \delta_G(C_j)^2 + (\delta_G(C_j) + 1)^2$. Consequently:

$$Q_{G_{ij}^+} = \frac{\zeta_G(\Gamma)}{m + 1} - \frac{\lambda_G(\Gamma) + 2\delta_G(C_i) + 2\delta_G(C_j) + 2}{4(m + 1)^2}.$$

Since $M_{G_{ij}^+}(\Gamma)$ decreases with $\delta_G(C_i)$ and $\delta_G(C_j)$, then: $\{C_1, C_2\} \in \arg\min_{\{C_i, C_j\} \subseteq \Gamma : i \neq j} Q_{G_{ij}^+}(\Gamma)$.

Having analysed G_i^+ and G_{ij}^+, let us now compare the two. In particular, we have

$$Q_{G_i^+}(\Gamma) - Q_{G_{ij}^+}(\Gamma) = \frac{2m + 1 - \delta_G(C_i) + \delta_G(C_j)}{2(m + 1)^2}.$$

Since $\delta_G(C_i) \leqslant 2m$, we have that: $Q_{G_i^+}(\Gamma) > Q_{G_{ij}^+}(\Gamma)$. In other words, adding an edge between members of two communities decreases the modularity of Γ more than adding an edge between members of the same community. Furthermore, we showed that adding an edge between C_1 and C_2 decreases the modularity of Γ more than adding an edge between any other pair of communities. This concludes the proof. □

Having analysed the minimization of modularity when adding an edge, we now analyse it during the edge removal. As it turns out, when considering the problem of removing a single edge to lower modularity in an optimal way, the best choice is removing an edge from inside the community with minimal degree.

Theorem 10. *Let $G = (V, E)$ be a network, and $\Gamma = \{C_1, \ldots, C_k\}$ be a community structure with communities sorted in decreasing order accordingly to their degree, i.e., $\forall_{1 \leqslant i < k} \delta_G(C_i) \geqslant \delta_G(C_{i+1})$. Then, for any two nodes, $v, w \in C_k$, we have:*

$$(v, w) \in \arg\min_{e \in E} Q_{\{V, E \setminus \{e\}\}}(\Gamma).$$

Proof. We will follow a similar reasoning as in the proof of Theorem 9. In more detail, let G_i^- denote the network that results from removing an edge $(v, w) : v, w \in C_i$ from G. Likewise, let G_{ij}^- denote the network that results from removing an edge (v, w) : $v \in C_i, w \in C_j$ from G, where $i \neq j$. Next, we analyse each of these networks, before comparing them against each other. In what follows ζ, λ and m are defined as in the proof of Theorem 9.

Starting with G_i^-, the removal of an edge (v, w), such that $v, w \in C_i$, decreases the number of intra-community edges by 1 (implying that $\zeta_{G_i^-}(\Gamma) = \zeta_G(\Gamma) - 1$), and decreases the sum of degrees of the members of C_i by 2 (thus $\delta_{G_i^-}(C_i) = \delta_G(\Gamma)(C_i) - 2$ and $\lambda_{G_i^-}(\Gamma) = \lambda_G(\Gamma) - \delta_G(C_i)^2 + (\delta_G(C_i) - 2)^2$). As a result, we have:

$$Q_{G_i^-(\Gamma)} = \frac{\zeta_G(\Gamma) - 1}{m - 1} - \frac{\lambda_G(\Gamma) - 4\delta_G(C_i) + 4}{4(m - 1)^2}.$$

As can be seen, $Q_{G_i^-}(\Gamma)$ increases with $\delta_G(C_i)$, implying that:

$$C_k \in \arg\min_{C_i \in \Gamma} Q_{G_i^-}(\Gamma).$$

Having discussed G_i^-, let us now discuss G_{ij}^-. Here, the removal of an edge (v, w), such that $v \in C_i, w \in C_j$, decreases the sum of degrees of the members of C_i and the members of C_j by 1, *i.e.*, $\delta_{G_{ij}^-}(C_i) = \delta_G(C_i) - 1$ and $\delta_{G_{ij}^-}(C_j) = \delta_G(C_j) - 1$. Thus: $\lambda_{G_{ij}^-}(\Gamma) = \lambda_G(\Gamma) - \delta_G(C_i)^2 + (\delta_G(C_i) - 1)^2 - \delta_G(C_j)^2 + (\delta_G(C_j) - 1)^2$. As a result, we have:

$$Q_{G_{ij}^-}(\Gamma) = \frac{\zeta_G(\Gamma)}{m - 1} - \frac{\lambda_G(\Gamma) - 2\delta_G(C_i) - 2\delta_G(C_j) + 2}{4(m - 1)^2}.$$

Since $Q_{G_{ij}^-}(\Gamma)$ increases with both $\delta_G(C_i)$ and $\delta_G(C_j)$, we have:

$$\{C_k, C_{k-1}\} \in \arg\min_{\{C_i, C_j\} \subseteq \Gamma : i \neq j} Q_{G_{ij}^-}(\Gamma).$$

Having analysed G_i^- and G_{ij}^-, let us now compare the two. In particular:

$$Q_{G_i^-}(\Gamma) - Q_{G_{ij}^-}(\Gamma) = \frac{-2m + 1 + \delta_G(C_i) - \delta_G(C_j)}{2(m-1)^2}.$$

Since $\delta_G(C_i) \leqslant 2m$, then $Q_{G_i^-}(\Gamma) < Q_{G_{ij}^-}(\Gamma)$. In other words, the removal of an edge between members of the same community decreases the modularity of Γ more than the removal of an edge between members of two different communities. Furthermore, we have shown that, out of all communities, the removal of an edge between members of C_k decreases the modularity of Γ the most. This concludes the proof. $\square$

Having dealt with the cases of adding and removing an edge, we now compare them against each other. The following corollary summarizes the findings of two previous theorems.

Corollary 1. *Let $G = (V, E)$ be a network, and $\Gamma = \{C_1, \ldots, C_k\}$ be a community structure with communities sorted in decreasing order accordingly to their degree, i.e., $\forall_{1 \leqslant i < k} \delta_G(C_i) \geqslant \delta_G(C_{i+1})$. Then, out of all single modifications of G (i.e., addition or removal of a single edge), the single modification that minimizes the modularity of Γ is the addition of (v, w), such that $v \in C_1, w \in C_2$, if $Q_{G_{12}^+}(\Gamma) > Q_{G_k^-}(\Gamma)$. Otherwise, it is the removal of (v, w), such that $v, w \in C_k$.*

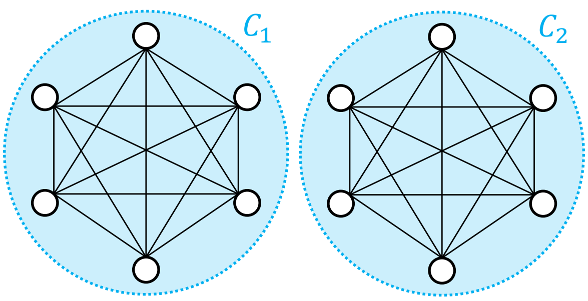

Figure 5.1: A coarse-grained community structure.

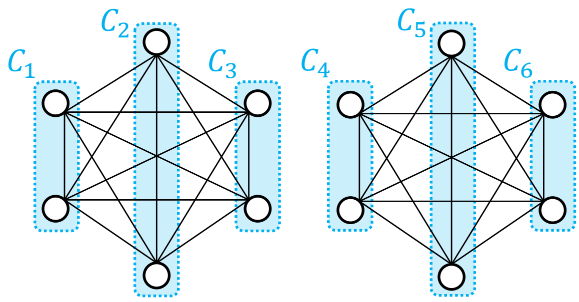

Figure 5.2: A fine-grained community structure.

Figures 5.1 and 5.2 present a sample network with two different community structures, one consisting of two communities (Figure 5.1) and another consisting of six communities (Figure 5.2). Let us first comment on Figure 5.1. Here, the community structure, Γ, satisfies: $Q_{G_{12}^+}(\Gamma) > Q_{G_k^-}(\Gamma)$. Thus, out of all single modifications of the network (*i.e.*, addition or removal of a single edge), the one that minimizes the modularity of Γ is the addition of an edge between the two communities therein (see Corollary 1). On the other hand, in Figure 5.2, the depicted community structure Γ satisfies: $Q_{G_{12}^+}(\Gamma) < Q_{G_k^-}(\Gamma)$. Thus, the single modification that minimizes the modularity of Γ is the removal of an edge from within the community C_i whose sum of degrees $\delta_G(C_i)$ is minimal (here, any community would be suitable, since the sum of degrees is equal in all of them).

5.4 Measure of Concealment

We now propose a measure of concealment, designed specifically to reflect how well a given group $C^\dagger$ is hidden in a particular community structure Γ. Importantly, in this section $C^\dagger$ is not necessarily a member of Γ. As such, when measuring how well $C^\dagger$ is hidden in Γ, it may well be the case that the members of $C^\dagger$ are spread out across multiple communities in Γ.

To this end, we will first propose two measures, denoted by μ'_G and μ''_G, which capture different aspects of concealment, and then we merge them into a single measure. More specifically, μ'_G is defined for every community structure Γ and every community $C^\dagger \subseteq V$ as follows:

$$\mu'_G(C^\dagger, \Gamma) = \frac{|\{C_i \in \Gamma : C_i \cap C^\dagger \neq \emptyset\}| - 1}{\max(|\Gamma| - 1, 1) \max_{C_i \in \Gamma}(|C_i \cap C^\dagger|)}.$$

Basically, this measure focuses on how well the members of $C^\dagger$ are spread out across the communities in Γ. In more detail, we have that $\mu'_G(C^\dagger, \Gamma) \in [0, 1]$, and the greater $\mu'_G(C^\dagger, \Gamma)$, the more $C^\dagger$ is concealed in Γ. Note that the numerator grows linearly with the number of communities that $C^\dagger$ is distributed over. Subtracting 1 from both the numerator and the $|\Gamma|$ term of the denominator is meant to handle the worst case, where all members of $C^\dagger$ appear in a single (possibly larger) community in Γ; in this case, we have: $\mu'_G(C^\dagger, \Gamma) = 0$. In contrast, the term $\max_{C \in \Gamma}(|C \cap C^\dagger|)$ is meant to promote community structures in which the members of $C^\dagger$ are more evenly distributed across the communities in Γ. As such, the maximum concealment is achieved when the members of $C^\dagger$ are uniformly distributed, with each member appearing in its own separate community; in this case, we have: $\mu'_G(C^\dagger, \Gamma) = 1$.

While μ'_G focuses on how well the members of $C^\dagger$ are distributed across communities, μ''_G focuses on how well $C^\dagger$ is "hidden in the crowd". More specifically, μ''_G is defined as:

$$\mu''_G(C^\dagger, \Gamma) = \sum_{C_i \in \Gamma : C_i \cap C^\dagger \neq \emptyset} \frac{|C_i \setminus C^\dagger|}{\max(n - |C^\dagger|, 1)}.$$

Just like the case with μ'_G, we have $\mu''_G(C^\dagger, \Gamma) \in [0, 1]$, and the greater $\mu''_G(C^\dagger, \Gamma)$, the more $C^\dagger$ is concealed in Γ. As can be seen, the value of this measure grows linearly with the number of non-members of $C^\dagger$ that appear in a communities containing members of $C^\dagger$.

As we have described, the measures μ'_G and μ''_G view the concealment of a community from two different perspectives. With this in mind, we combine them into a single measure, denoted by μ_G, whereby the trade-off between the two measures is controlled by parameter $\alpha \in [0, 1]$. More formally, our proposed measure of concealment of a community $C^\dagger$ in a community structure Γ is defined as follows:

$$\mu_G(C^\dagger, \Gamma) = \alpha \mu'_G(C^\dagger, \Gamma) + (1 - \alpha) \mu''_G(C^\dagger, \Gamma).$$

Naturally, for all $C^\dagger$ and Γ we have $\mu_G(C^\dagger, \Gamma) \in [0, 1]$, where greater values indicate greater levels of concealment. Unless stated otherwise, in the rest of this dissertation we will consider our concealment measure with $\alpha = \frac{1}{2}$.

To illustrate how our measure works, we present in Figures 5.3, 5.4 and 5.5 a sample network with three different community structures, where the goal is to measure the

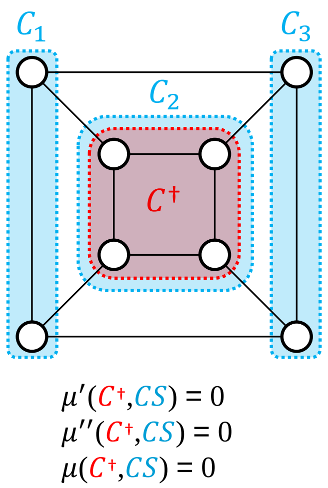

Figure 5.3: Example of a community structure, where $\mu(C^\dagger, \Gamma) = 0$.

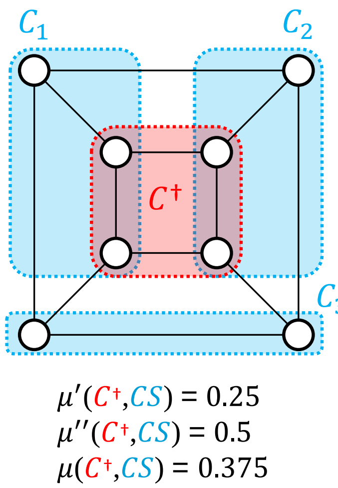

Figure 5.4: Example of a community structure, where $\mu(C^\dagger, \Gamma) = \frac{3}{8}$.

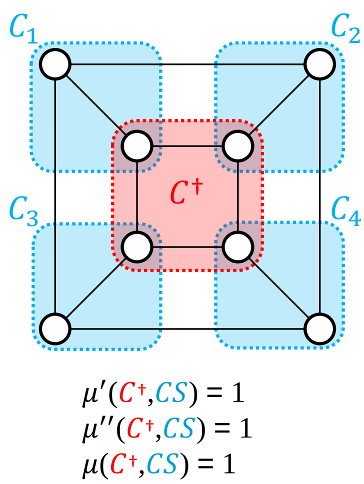

Figure 5.5: Example of a community structure, where $\mu(C^\dagger, \Gamma) = 1$.

concealment of $C^\dagger$. The concealment of $C^\dagger$ in Figure 5.3 is $\mu(C^\dagger, \Gamma) = 0$, which is because the members of $C^\dagger$ are completely exposed as a separate community. In contrast, given the community structure in Figure 5.4 the concealment of $C^\dagger$ is $\mu(C^\dagger, \Gamma) = \frac{3}{8}$. Eventually, in Figure 5.5 we have $\mu_G(C^\dagger, \Gamma) = 1$.

Finally, we define a new variant of the problem of disguising a community, where the the members of community $C^\dagger$ intend to rewire the network so that $\mu(C^\dagger, \Gamma)$ is maximized. More formally:

Definition 6 (Concealing Community)**.** *This problem is defined by a tuple, $(G, C^\dagger, \rho, b)$, where $G = (V, E)$ is a network, $C^\dagger \subseteq V$ is the community to be hidden, $\rho : \mathbb{G} \to 2^{2^V}$ is a community-detection algorithm, and $b \in \mathbb{N}$ is a budget specifying the maximum number of edges that can be added or removed. The goal is then to find two sets of edges A^* and R^* (where edges from A^* will be added and edges from R^* will be removed), such that $|A^*| + |R^*| \leqslant b$ and $G^* = (V, (E \cup A^*) \setminus R^*)$ is in:*

$$\underset{G' \in \left\{(V,(E\cup A)\setminus R) : A \subseteq \bar{E}, R \subseteq E, |A|+|R| \leqslant b\right\}}{\arg\max} \mu_{G'}(C^\dagger, \rho(G')).$$

5.5 Heuristic Solution

We now describe a simple heuristic solution to the Concealing Community problem.

5.5.1 The DICE Heuristic

We set to develop a simple heuristic that can be applied by any group of people regardless of their technical background or their knowledge of the network topology. After all, it would be of limited use to have an exact algorithm that can only be understood or applied by optimization experts armed with enormous processing power. Likewise, exact algorithms that require knowing the entire network topology may prove to be impractical since such knowledge is rarely available. Our heuristic, called *DICE—Disconnect Internally, Connect Externally*—is presented as Algorithm 3.

Algorithm 3 The DICE heuristic

Input: A network $G = (V, E)$, budget $b \in \mathbb{N}$, number of edges to remove $d \leqslant b$, community $C^{\dagger} \subseteq V$ to be hidden
Output: Sets of edges to be added A^* and to be removed R^* from the network

```
for i = 1, ..., d do
    Choose random (v, w) ∈ E(C†)
    R* = R* ∪ {(v, w)}
for i = 1, ..., b − d do
    Choose random (v, w) ∈ Ē, where v ∈ C† and w ∉ C†
    A* = A* ∪ {(v, w)}
```

This heuristic is inspired by the analysis provided in Section 5.3, particularly in Corollary 1. As community-detection algorithms are typically designed to search for a structure that maximizes modularity, they promote structures that have dense connections within communities and sparse connections between them. With this in mind, the first loop of our heuristic decreases the density of the connections within $C^{\dagger}$, whereas the second loop increases the connections between $C^{\dagger}$ and other communities. By doing so, a community-detection algorithm is more likely to overlook $C^{\dagger}$, *i.e.*, it may fail to recognize $C^{\dagger}$ as a community, and may instead assign its members to multiple communities. Parameter d allows the group of evaders to control the trade-off between concealment and connectedness. Increasing d would sacrifice the connectivity within the group in return for a better camouflage.

Let us comment on how DICE can be applied in practice. On Facebook, for example, the first loop requires some members to "unfriend" other members, which is rather straightforward. As for the second loop, members must send a friendship request to non-members. These could be classmates, coworkers, neighbours living next door, or even random people (it is possible to try multiple random friendship requests, hoping that some of them would be successful).

5.5.2 Experimental Results

We now describe experimental results of using DICE heuristic to conceal community in the network. For each network, we experiment with seven community-detection algorithms described in Section 5.2, namely: Eigenvector [107], Betweenness [108], Walktrap [119], Louvain [18], Greedy [32], Infomap [125] and Spinglass [123].

Every experiment consists of a community-detection algorithm and a network. The experiment starts by running the algorithm to obtain a community structure Γ. After that, the group of evaders, *i.e.*, $C^{\dagger}$, is chosen to be the element in Γ whose size is the median of the sizes of all communities in Γ (ties are broken uniformly at random). Although $C^{\dagger}$ does not necessary have to be an element of Γ, we choose it this way in order to study the worst case scenario in which $C^{\dagger}$ is initially exposed completely by the algorithm. The experiment then proceeds in rounds, each involving the execution of DICE followed by

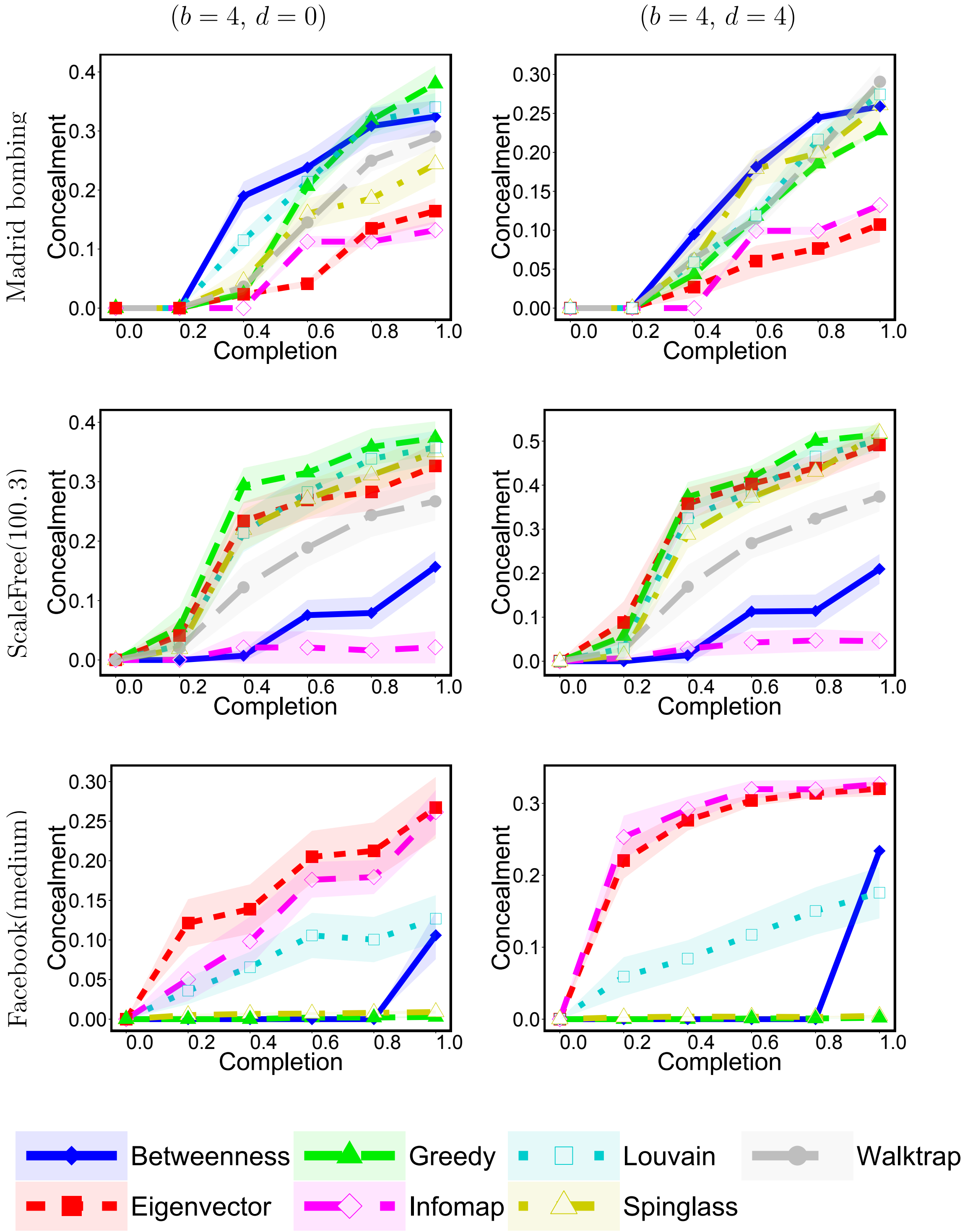

Figure 5.6: Consecutive execution of DICE $\left\lceil \frac{|C^{\dagger}|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

the execution of the community-detection algorithm, to measure how well $C^\dagger$ is hidden in the new outcome of the algorithm. We set the number of rounds to be $\left\lceil \frac{|C^\dagger|}{b} \right\rceil$. In each round, we disconnect d links from within $C^\dagger$ (chosen uniformly at random), and then connect $b - d$ members of $C^\dagger$ to $b - d$ non-members of $C^\dagger$ (again chosen uniformly at random). Due to this randomness in our implementation, DICE may yield different results in different executions. Therefore, we repeat each experiment 50 times, and report the 95% confidence interval. All datasets are described in Section 2.2.

Figure 5.6 shows the results of some of our experiments. As can be seen, DICE is able to hide the group of evaders with varying levels of success, depending on the community-detection algorithm being used. Importantly, the performance does not appear to be overly-sensitive to the parameter d. This is important because it provides the members of $C^\dagger$ with the ability to control this parameter as needed (*i.e.*, control the trade-off between the number of internal links being removed, and the number of external links being added). For example, the members of $C^\dagger$ might be interested in hiding as much as possible, while removing as few internal links as possible (after all, the added external links are fake, serving no purpose other than disguising the group of evaders, whereas the removed internal links are real; they existed within the group for a reason). In such a case, since the addition of an external link is not entirely under the control of $C^\dagger$ (as it requires the consent of a non-member), the number of newly-added external links may be insufficient for providing a satisfactory level of concealment, in which case the members can compensate for this by sacrificing more internal links, *i.e.*, by increasing the parameter d.

Figures 5.7 and 5.8 illustrate the average value of our concealment measure, μ. In particular, each row represents a community-detection algorithm, each column represents a network, and the intensity of the colour in each cell represents the average value of μ, taken over 50 simulations, either by generating a new random network in each simulation, or by re-running the simulation on the same real-life network (recall that the DICE heuristic is non-deterministic, and may yield different results on the same network). Columns and rows are sorted according to the increasing average value. As can be seen, the Infomap [125] algorithm seems to be the most difficult to fool (see how the row representing Infomap has, on average, the lightest colour shades). Interestingly, this is also the algorithm that Lancichinetti and Fortunato [83] find to be the most effective in practice.

All experimental results for the DICE heuristic are presented in Appendix B in Figures B.1, B.2, B.3, B.4 and B.5.

5.6 Concluding Remarks

In this chapter we introduced the subject of hiding communities in social networks. We analysed how the modularity of a given community structure is affected by modifying the network iteratively, where each modification involves the addition or removal of an edge. We also proposed a measure of concealment, designed to quantify how well a group of nodes is hidden in a given community structure. Moreover, we presented simple heuristic that can realistically be applied by actual communities; it is scalable and does not require any knowledge on the topology of the entire network, nor does it require any technical

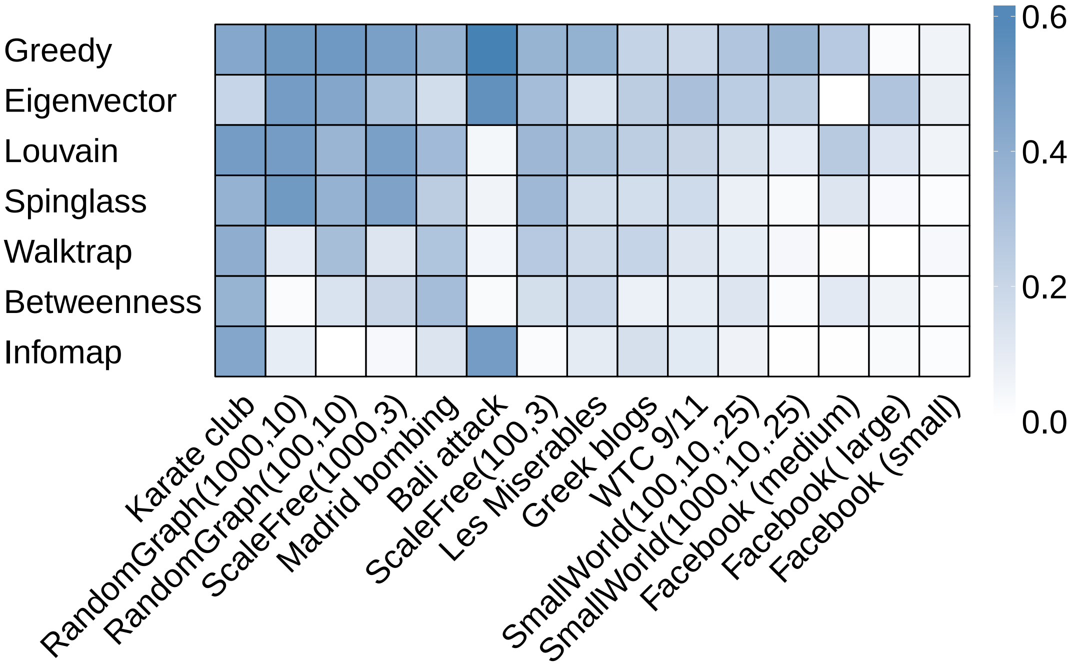

Figure 5.7: Average concealment-measure value in each experiment for DICE with $(b = 4, d = 0)$, where b is the budget in each execution, and d is the number of removed internal edges.

knowledge or massive processing power. We evaluated the effectiveness of our heuristic in both real-life and randomly-generated networks, using a wide variety of community-detection algorithms.

We now describe a number of limitations.of our study. All of these limitations can be considered potential venues of future research. First, in Section 5.3 we consider only iterative changes of the network's structure in order to minimize modularity. One might investigate the effects of adding or removing multiple edges at once, and seek for optimal algorithms of Minimizing Modularity with higher budget. We expect this to be an NP-hard problem, however some effective heuristic solutions might be found.

Second, we considered a group of community detection algorithms realising one particular approach to community detection, *i.e.*, finding community structures where each node is a part of exactly one community. Nonetheless, there exists a whole body of literature on an alternative approach, *i.e.*, one where we seek overlapping communities. Hiding communities in such setting may give rise to new problems and research questions, *e.g.*, considering the process of hiding from the perspective of a single node. Since each node can be a part of multiple communities, a given member of the network might be interested in being associated with a particular subset of communities. Alternatively, a node might seek an optimal way to become a part of a chosen social circle.

Finally, we presented a heuristic solution that works surprisingly well against a wide variety of community detection algorithms. However, some of them proved to be more difficult to hide against than the others. A potential direction of future work is developing

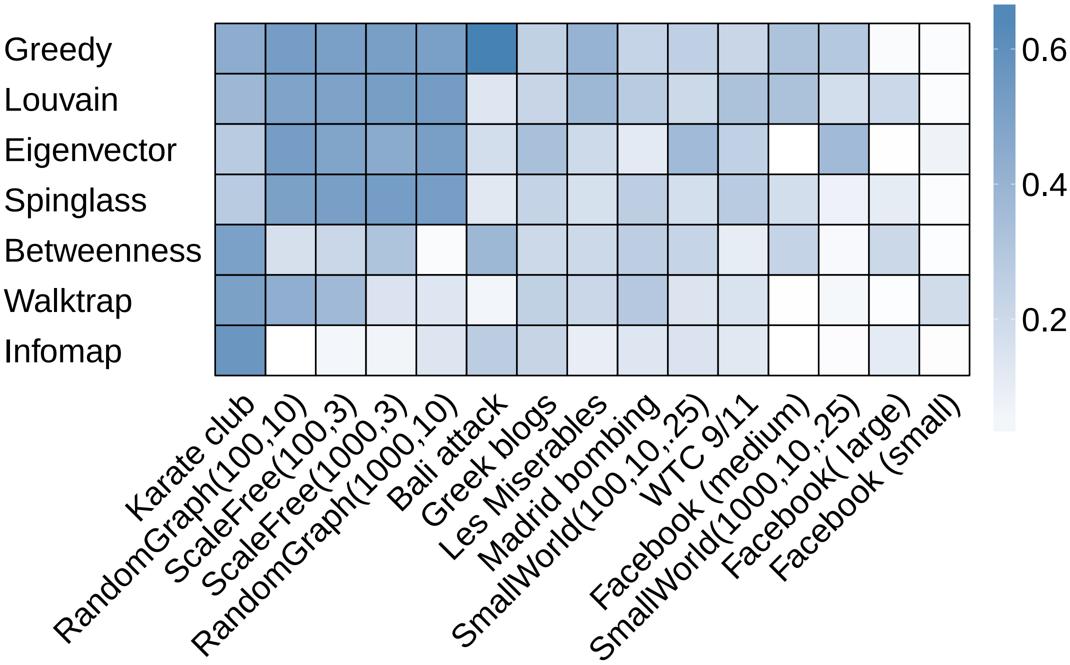

Figure 5.8: Average concealment-measure value in each experiment for DICE with ($b = 4, d = 4$), where b is the budget in each execution, and d is the number of removed internal edges.

specialized hiding algorithms, focusing on the community structure that will be returned by a particular algorithm.

Chapter 6

Evading Link Prediction

In this chapter we investigate the problem of preventing identification of chosen edges by link prediction algorithms. In what follows, we introduce the formal definition of the problems, investigate its computational complexity, and provide simple heuristic solution that can be run in polynomial time.

6.1 Introduction

One of the key research challenges in social network analysis is the link prediction problem [89, 94]. Intuitively, based on the current structure of the network, this problem involves predicting the connections that are most likely to be created in the future. An alternative interpretation of this problem is to identify the connections that are hidden from the observer, either due to scarcity of data, or due to the deliberate concealment of information [25]. Link prediction has many applications, such as providing recommendations to customers in e-commerce [37], discovering the interactions between proteins in biological networks [26], and finding hidden connections between terrorists [5] or criminals [140].

A plethora of link prediction algorithms have been proposed in the literature (see [89, 94, 6] for an extensive overview). In essence, the aim of all such algorithms is to estimate the likelihood that there exists a not-yet-discovered edge between two seemingly-disconnected nodes, or the likelihood that an edge will be formed between those two nodes in the future [57]. The mainstream class of link-prediction algorithms is based on similarity indices [94]. As the name suggests, such indices measure the similarity between any two disconnected nodes in a network by analysing its topology. The underlying assumption behind similarity-based algorithms is that the greater the similarity between two nodes, the greater the likelihood of having a link between them.

If used with malicious or mischievous intent, social network analysis tools—and link prediction algorithms in particular—may constitute a serious threat to both privacy and security of the general public. Importantly, not only can such tools use the data that has been willingly disclosed by its owners, but they can also use private data that has not been disclosed at all. In particular, it has been shown that such private data can be inferred from the publicly disclosed data, and that is by performing an attribute inference attack. As the name suggests, such an attack infers the missing or partial attributes of network nodes [159]. Mislove [103] showed that it is possible to infer otherwise-private information about other Facebook users by analysing the topology of this social network combined with the attributes of some users. It has been shown that the effectiveness of an inference attack can be significantly improved if the attacker starts with a pre-processing stage in which a link prediction algorithm is used to infer the missing links [81]. In this chapter, we are particularly interested in a special type of such attacks, called the link reconstruction attack, whereby a link prediction algorithm is used to identify missing or hidden links [50].

Given such concerns, a number of studies recommended that social network users conceal some of their attributes, as a countermeasure against attribute inference attacks [90, 63]. Such recommendations include also concealing links. In this context, although a number of studies in the literature have argued *why* there is a need to conceal one's private links, they unfortunately did not specify *how* this should be done.

Against this background, given a "*seeker*" who is running a link prediction algorithms, and "*evaders*" who want to hide some of their connections, we study how the evaders can make those connections harder for the seeker to identify. This research question matters because, on one hand, it may assist the general public in protecting their privacy from intrusion by private and public entities; on the other hand, it may mitigate (at least to some extent) the threats posed by cyber criminals. It may also assist law-enforcement agencies in understanding how criminals and terrorists could evade social network analysis tools, especially given their increasing reliance on social-media survival strategies [109, 68].

The remainder of this chapter is organized as follows. Necessary notation and definitions are introduced in Section 6.2. After that, in Section 6.3, we formally introduce the problem of Evading Link Prediction. We analyse its complexity in Section 6.4. In Section 6.5 we propose our heuristic for Evading Link Prediction problem, and perform its empirical evaluation. Concluding remarks follow in Section 6.6.

6.2 Preliminaries

Link prediction algorithms evaluate how likely it is that there exists a not-yet-discovered edge between a pair of nodes or how likely it is to be formed in the future [57]. Many link prediction algorithms are based on *similarity indices*, also called *kernels* [134]. More formally, given a network $G = (V, E)$, a similarity index is a function that assigns a score to each non-edge in G, *i.e.*, $\sigma : \bar{E} \to \mathbb{R}$.

An important class of link prediction algorithms are *local* similarity indices, *i.e.*, indices that account only for the direct network vicinity of the non-edge in question. As such, the algorithms based on local similarity indices are typically computationally tractable and can be used for efficient analysis of even large networks [94]. In this dissertation, we study all local similarity indices for general networks presented in the survey of Lü and Zhou [94]:[1] Common Neighbours [106], Salton [128], Jaccard [65], Sørensen [136], Hub Promoted [122], Hub Depressed [122], Leicht-Holme-Newman [85], Adamic-Adar [4] and Resource Allocation [160]. Table 6.1 presents the defining formula for each index. We denote the set of all those similarity indices by $\mathbb{S}$, *i.e.*, $\mathbb{S} = \{\sigma^{\mathrm{CN}}, \sigma^{\mathrm{Sal}}, \sigma^{\mathrm{Jac}}, \sigma^{\mathrm{Sør}}, \sigma^{\mathrm{HPI}}, \sigma^{\mathrm{HDI}}, \sigma^{\mathrm{LHN}}, \sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$.

[1]The only local index that we exclude from our analysis is the Preferential Attachment since it is based on the assumption that the network has scale-free properties.

Index name	Value
Common Neighbours	$\sigma^{\mathrm{CN}}(v,w) = \lvert N(v,w)\rvert$
Salton	$\sigma^{\mathrm{Sal}}(v,w) = \frac{\lvert N(v,w)\rvert}{\sqrt{\delta(v)\delta(w)}}$
Jaccard	$\sigma^{\mathrm{Jac}}(v,w) = \frac{\lvert N(v,w)\rvert}{\lvert N(v)\cup N(w)\rvert}$
Sørensen	$\sigma^{\mathrm{Sør}}(v,w) = \frac{2\lvert N(v,w)\rvert}{\delta(v)+\delta(w)}$
Hub Promoted	$\sigma^{\mathrm{HPI}}(v,w) = \frac{\lvert N(v,w)\rvert}{\min(\delta(v),\delta(w))}$
Hub Depressed	$\sigma^{\mathrm{HDI}}(v,w) = \frac{\lvert N(v,w)\rvert}{\max(\delta(v),\delta(w))}$
Leicht-Holme-Newman	$\sigma^{\mathrm{LHN}}(v,w) = \frac{\lvert N(v,w)\rvert}{\delta(v)\delta(w)}$
Adamic-Adar	$\sigma^{\mathrm{AA}}(v,w) = \sum_{u\in N(v,w)} \frac{1}{\log(\delta(u))}$
Resource Allocation	$\sigma^{\mathrm{RA}}(v,w) = \sum_{u\in N(v,w)} \frac{1}{\delta(u)}$

Table 6.1: Formulas of local similarity indices.

The value of all considered similarity indices increases with the number of common neighbours of a pair of nodes. The main difference between the indices is the way the number of common neighbours is scaled. Two of the indices, namely Adamic-Adar and Resource Allocation, take into consideration also the degree of the common neighbours, again, scaling it in different manners.

The most common ways of evaluating the performance of a similarity measure are *Area under Receiver Operating Characteristic curve (AUC)* [48] and *Area under Precision-Recall curve (PR)* [96] statistics. To compute either of these statistics for a similarity index σ, we partition the set of edges E into two disjoint sets: the training set T and the probe set H, *i.e.*, $T \cup H = E$ and $T \cap H = \emptyset$. Network (V,T) serves as an input for similarity index σ, which produces the ranking of elements in $H \cup \bar{E}$. AUC and PR statistics express the quality of this ranking with a single number.

In what follows, let σ_k denote the set of top k elements from $H \cup \bar{E}$ in the ranking of values of σ and let $m = |H \cup \bar{E}|$.

- *Area under Receiver Operating Characteristic curve*, denoted by $AUC(E,H)$, for given T and H is the area under the plot consisting of points:

$$\left\{\left(\frac{|\sigma_k \cap \bar{E}|}{|\bar{E}|}, \frac{|\sigma_k \cap H|}{|H|}\right)\right\}_{k=1}^{m}.$$

More intuitive interpretation of $AUC(E,H)$ is that it is the probability that σ assigns higher score to a randomly chosen edge from H than to a randomly chosen non-edge from $\bar{E}$ (ties are broken at random). Let $Q^{>} \subseteq H \times \bar{E}$ denote the set of pairs of edges where the edge from H has higher σ score than the edge from $\bar{E}$, *i.e.*,

$Q^{>} = \{(e_1, e_2) \in H \times \bar{E} : \sigma(e_1) > \sigma(e_2)\}$. Analogically, let $Q^{=} \subseteq H \times \bar{E}$ denote the set of pairs of edges where the edge from H and the edge from $\bar{E}$ have equal σ scores, *i.e.*, $Q^{=} = \{(e_1, e_2) \in H \times \bar{E} : \sigma(e_1) = \sigma(e_2)\}$. Now, we have:

$$AUC(E, H) = \frac{1}{|H||\bar{E}|}\left(|Q^{>}| + \frac{1}{2}|Q^{=}|\right).$$

- *Area under Precision-Recall curve*, denoted by $PR(E, H)$, for given T and H is the area under the plot consisting of points:

$$\left\{\left(\frac{|\sigma_k \cap H|}{|H|}, \frac{|\sigma_k \cap H|}{k}\right)\right\}_{k=1}^{m}.$$

  Since this value is not well-defined for plots that are not continuous, we use the average precision estimator AP, described as one of the most robust by Boyd *et al.* [22]. The value of AP is the average precision of a classifier predicting existence of all edges that have higher value of σ than a chosen edge from H. Let $W^{>}(e_0)$ denote the edges from $H \cup \bar{E}$ with σ scores higher than the score of e_0, *i.e.*, $W^{>}(e_0) = \{e \in H \cup \bar{E} : \sigma(e) > \sigma(e_0)\}$. Let $W^{=}(e_0)$ denote the edges from $H \cup \bar{E}$ other than e_0 with σ scores equal to the score of e_0, *i.e.*, $W^{=}(e_0) = \{e \in (H \cup \bar{E}) \setminus \{e_0\} : \sigma(e) = \sigma(e_0)\}$. Taking into account the possibility of equal scores, average precision value is:

$$AP(E, H) = \frac{1}{|H|} \sum_{e_0 \in H} \frac{|W^{>}(e_0) \cap H| + 1 + \frac{1}{2}|W^{=}(e_0) \cap H|}{|W^{>}(e_0)| + 1 + \frac{1}{2}|W^{=}(e_0)|}.$$

6.3 Problem Definition

In this section, we formally introduce the problem of *Evading Link Prediction.*

Definition 7 (Evading Link Prediction)**.** *This problem is defined by a tuple $(G, s, f, H, b, \hat{A}, \hat{R})$, where $G = (V, E) \in \mathbb{G}$ is a network, $\sigma : \bar{E} \to \mathbb{R}$ is a similarity index, $f \in \{AUC, AP\}$ is a performance evaluation metric, $H \subset E$ is the set of edges to be hidden, $b \in \mathbb{N}$ is a budget specifying the maximum number of edges that can be modified (i.e., added or removed), $\hat{A} \subseteq \bar{E}$ is the set of edges that can be added, and $\hat{R} \subseteq E \setminus H$ is the set of edges that can be removed. The goal is then to identify two sets of edges, $A^* \subseteq \hat{A}$ and $R^* \subseteq \hat{R}$, such that the resulting set, $E^* = (E \cup A^*) \setminus R^*$, is in:*

$$\underset{E' \in \left\{(E \cup A) \setminus R \ : \ A \subseteq \hat{A}, \ R \subseteq \hat{R}, \ |A|+|R| \leqslant b\right\}}{\arg\min} f(E', H).$$

Let us comment on the above definition. The goal is to find edges to be added and edges to be removed from the network (taking into consideration sets $\hat{A}$ and $\hat{R}$, as well as limited budget), such that the chosen set of edges H is well hidden. Our measure of performance is lowering either *Area under ROC curve* or *Average Precision* statistic of the ranking of the chosen similarity index σ.

Note that, from the perspective of the evaders, H is a subset of E. In contrast, from the perspective of the seeker, we have $H \subseteq \bar{E}$. We generally assume the point of view of the evaders.

We introduced the sets $\hat{A}$ and $\hat{R}$ to model scenarios in which the evaders can only modify the network in a limited manner. This could be the case, for example, when certain connections are more costly to establish than others, or when the evaders want to avoid removing certain critical edges, or avoid connecting to certain individuals. Likewise, we introduced the "budget", b, to model scenarios in which the evaders can only perform a limited number of modifications.

We now move to the complexity analysis of the presented problem.

6.4 Complexity Analysis

In what follows, we prove that the problem of Evading Link Prediction is NP-complete for all the similarity indices described in Section 6.2 and for both the AUC and AP metrics. To this end, we need to first define a certain network, denoted by $\Gamma(c, P)$, which will be used later on in our NP-completeness proof.

Note that in the NP-completeness proof we use the universe U and the set of its subsets S from the 3-Set Cover problem (as defined in Section 2.3.3) to construct the network $\Gamma(c, S)$.

Definition 8 (The $\Gamma(c, S)$ Network)**.** *Let $U = \{u_1, \ldots, u_m\}$ be a set of m elements, and let $S = \{S_1, \ldots, S_q\}$ be a cover of U containing q subsets that are each smaller than U. That is, $\forall_i S_i \subset U$ and $\bigcup_{S_i \in S} S_i = U$. Then, given a constant, $c \in \mathbb{N}$, the network $\Gamma(c, P)$ is created as follows:*

- ***The set of nodes:*** *For every $S_i \in S$, we create a single node, denoted by S_i. Moreover, for every $u_i \in U$, we create a node denoted by u_i, c nodes denoted by $a_{i,1}, \ldots, a_{i,c}$, and $q - |S(u_i)|$ nodes denoted by $d_{i,1}, \ldots, d_{i,q-|S(u_i)|}$, where $S(u_i) = \{S_j \in S : u_i \in S_j\}$. Finally, we create three nodes, v_0, v_1, and u_0, as well as c nodes, $a_{0,1}, \ldots, a_{0,c}$, and q nodes, $d_{0,1}, \ldots, d_{0,q}$.*
- ***The set of edges:*** *First, we create the edge (u_0, v_0). After that, for every $S_j \in S$ we create the edge (S_j, v_1), as well as the edges (S_j, u_i) for every $u_i \in S_j$. Moreover, for every $u_i \in U$ we create the edge (u_i, v_1), as well as the edges (u_i, u_j) for every $u_j \in \{u_{i+1}, \ldots, u_m\}$ (this way, the nodes in $\{u_0, \ldots, u_m\}$ form an $(m+1)$-clique). Furthermore, for every $d_{i,j}$ we create the edges $(d_{i,j}, u_i)$ and $(d_{i,j}, v_1)$. Finally, for every $a_{i,j}$ we create the edges $(a_{i,j}, u_i)$, $(a_{i,j}, v_0)$ and $(a_{i,j}, v_1)$.*

Figure 6.1 provides an illustration of the $\Gamma(c, S)$ network. The following lemma is built around this network.

Intuitively, the lemma claims that by adding to the network a set of edges corresponding to the choice of a subset of S, we increase the σ scores of non-edges corresponding to the covered elements of the universe U from the value equal to the score of (u_0, v_0) to the value higher than the score of (u_0, v_0). Moreover, the relation of σ scores (*i.e.*, whether they are lower, equal, or higher) of all other non-edges to the score of (u_0, v_0) remains the same.

Lemma 1. *Consider the network $(V, E) = \Gamma(c, S)$ for which $m \geqslant 5$ and $|S_i| = 3$ for all $S_i \in S$. Let $\hat{A} = \{(S_i, v_0) : S_i \in S\}$, and for every $A \subseteq \hat{A}$, let $S_A = \{S_i \in S :$*

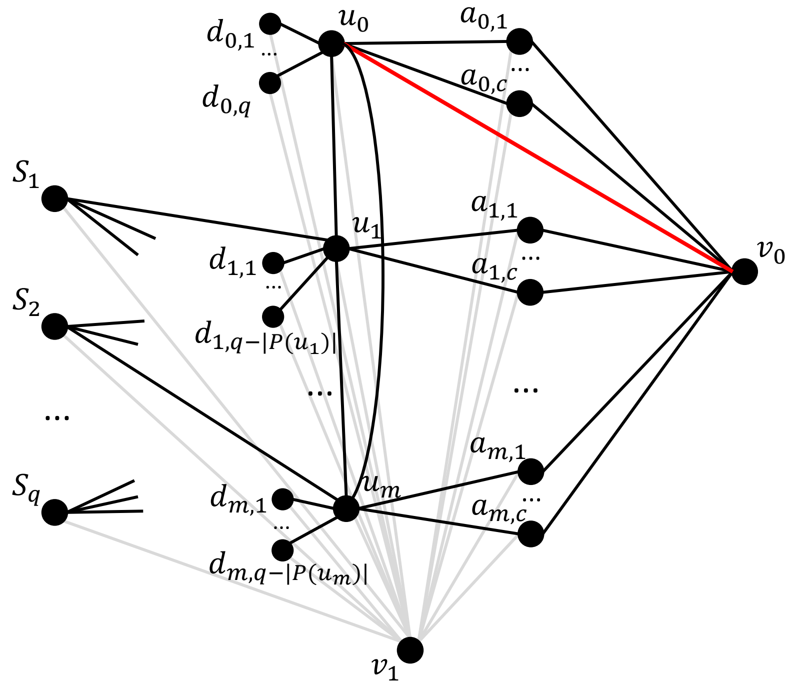

Figure 6.1: An illustration of the $\Gamma(c, S)$ network. Edges connecting v_1 with other nodes are grayed out to improve readability. The red edge (u_0, v_0) is the one to be hidden.

$(S_i, v_0) \in A\}$, and let $S_A(u_j) = \{S_i \in S_A : u_j \in S_i\}$. Furthermore, let Γ_A denote a set of all possible networks created by adding a subset of $\hat{A}$ to the network (V, E), i.e., $\Gamma_A = \{(V, E \cup A) : A \subseteq \hat{A}\}$. For every similarity index, $\sigma \in \mathbb{S}$, there exists some constant, $c \in \mathbb{N}$, such that:

(a) for every non-edge of the form $(u_i, v_0) : i \in \{0, \dots, m\}$ and for every network in Γ_A we have:

- *$\sigma(u_i, v_0) = \sigma(u_0, v_0)$ if $S_A(u_i) = \emptyset$, and*
- *$\sigma(u_i, v_0) > \sigma(u_0, v_0)$ otherwise.*

(b) for every non-edge, $(S_i, v_0) : i \in \{0, \dots, q\}$ and for every network in Γ_A where $(S_i, v_0) \notin A$ we have:

- *$\sigma(S_i, v_0) < \sigma(u_0, v_0)$.*

(c) for every other non-edge $e \in \bar{E} \setminus \{(u_0, v_0), \dots, (u_m, v_0), (S_1, v_0), \dots, (S_q, v_0)\}$ either:

- *for every network in Γ_A we have $\sigma(e) > \sigma(u_0, v_0)$, or*
- *for every network in Γ_A we have $\sigma(e) = \sigma(u_0, v_0)$, or*
- *for every network in Γ_A we have $\sigma(e) < \sigma(u_0, v_0)$.*

Proof. First, note that the following holds:

- for each S_i and for every network in Γ_A where $(S_i, v_0) \notin A$ we have $N(S_i, v_0) = \emptyset$;

- for each $d_{i,j}$ and for every network in Γ_A we have $N(v_0, d_{i,j}) = \emptyset$.

This implies that for every similarity index, $\sigma \in \mathbb{S}$, we have:

- for each S_i and for every network in Γ_A where $(S_i, v_0) \notin A$ we have $\sigma(S_i, v_0) = 0$;
- for each $d_{i,j}$ and for every network in Γ_A we have $\sigma(v_0, d_{i,j}) = 0$.

One can also verify that for every $\sigma \in \mathbb{S}$ and for every network in Γ_A it holds that $\sigma(u_0, v_0) > 0$.

This implies that point (b) of the Lemma 1 holds, and that point (c) holds for every non-edge of the form $(v_0, d_{i,j})$. We still need to prove the correctness of point (a), as well as the correctness of point (c) for every non-edge of the form:

(i) (v_0, v_1)

(ii) (u_i, S_j) for $u_i \notin S_j$

(iii) $(u_i, a_{j,l})$ for $i \neq j$

(iv) $(u_i, d_{j,l})$ for $i \neq j$

(v) (S_i, S_j) for $i \neq j$

(vi) $(S_i, a_{j,l})$

(vii) $(S_i, d_{j,l})$

(viii) $(a_{i_1,j_1}, a_{i_2,j_2})$

(ix) $(a_{i_1,j_1}, d_{i_2,j_2})$

(x) $(d_{i_1,j_1}, d_{i_2,j_2})$

To this end, first note that the following holds for every network in Γ_A and every $a_{i,j}$, $d_{i,j}$, S_i in that network:

- $\delta(a_{i,j}) = 3$ (because $a_{i,j}$ is connected to v_0, v_1 and u_i);
- $\delta(d_{i,j}) = 2$ (because $d_{i,j}$ is connected to v_1 and u_i);
- $4 \leqslant \delta(S_i) \leqslant 5$ (because S_i is connected to v_1 and to every $u_j \in S_i$, where we assumed that $|S_i| = 3$; also, if $S_i \in A$, then S_i is connected to v_0).

Also note that $u_0 \notin S_i$ for every $S_i \in S$. Therefore, for each given $A \subseteq \hat{A}$, we have: $S_A(u_0) = 0$. In what follows, we will use the aforementioned facts without referring back to them.

Next, we present the proof for only one similarity index, namely *Common Neighbours*, σ^{CN}, which simply counts the number of nodes that are neighbours to both v and w. The proofs for all the similarity indices in $\mathbb{S}$ follow a similar reasoning and can be found in Appendix C.

In particular, for σ^{CN} we choose network $\Gamma(c, S)$ with $c = 6$. Then, to prove the correctness of point (a) it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\ldots,u_m\}} \sigma^{\mathrm{CN}}(u_j, v_0) = 6 + |S_A(u_j)|.$$

Moving on to point (c), note that for every network in Γ_A we have $\sigma^{\mathrm{CN}}(u_0, v_0) = 6$ and that the following holds:

(i) $\sigma^{\mathrm{CN}}(v_0, v_1) = 6(m+1) + |A| > \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of u_0 and v_1 are all the nodes $a_{i,j}$ and all the nodes S_i where $(S_i, v_0) \in A$.

(ii) $\sigma^{\mathrm{CN}}(u_i, S_j) \leqslant 4 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of u_i and S_j consist of v_1 and every $u_l \in S_j : l \neq i$ (note that we assumed that $|S_j| = 3$, and u_i may or may not be an element of S_j).

(iii) $\sigma^{\mathrm{CN}}(u_i, a_{j,l}) = 2 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of u_i and $a_{j,l}$ are v_1 and u_j.

(iv) $\sigma^{\mathrm{CN}}(u_i, d_{j,l}) = 2 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of u_i and $d_{j,l}$ are v_1 and u_j.

(v) $\sigma^{\mathrm{CN}}(S_i, S_j) \leqslant 5 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of S_i and S_j consist of v_1, and possibly v_0 (if $\{(S_i, v_0), (S_j, v_0)\} \subseteq A$), as well as the every element in $S_i \cap S_j$ (there can be at most 3 such elements, since we assumed that $|S_i| = |S_j| = 3$, and we place no restrictions on having $S_i = S_j$).

(vi) $\sigma^{\mathrm{CN}}(S_i, a_{j,l}) \leqslant 3 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of S_i and $a_{j,l}$ consist of v_1, and possibly v_0 (if $(S_i, v_0) \in A$) and possibly u_j (if $i = j$).

(vii) $\sigma^{\mathrm{CN}}(S_i, d_{j,l}) \leqslant 2 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of S_i and $d_{j,l}$ consist of v_1 and possibly u_j (if $i = j$).

(viii) $\sigma^{\mathrm{CN}}(a_{i_1,j_1}, a_{i_2,j_2}) \leqslant 3 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of a_{i_1,j_1} and a_{i_2,j_2} consist of v_1 and v_0 and possibly u_{i_1} (if $i_2 = i_1$).

(ix) $\sigma^{\mathrm{CN}}(a_{i_1,j_1}, d_{i_2,j_2}) \leqslant 2 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of a_{i_1,j_1} and d_{i_2,j_2} consist of v_1 and possibly u_{i_1} (if $i_2 = i_1$).

(x) $\sigma^{\mathrm{CN}}(d_{i_1,j_1}, d_{i_2,j_2}) \leqslant 2 < \sigma^{\mathrm{CN}}(u_0, v_0)$, because the common neighbours of d_{i_1,j_1} and d_{i_2,j_2} consist of v_1 and possibly u_{i_1} (if $i_2 = i_1$).

This concludes the the proof of Lemma 1. □

Having defined the $\Gamma(c, S)$ network, and having proven the correctness of Lemma 1, we are now ready to present our main theorem of this chapter.

The intuition behind the proof of this theorem is as follows. The values of AUC and AP depend solely on the order of non-edges in the ranking of the similarity index, rather than on the values of the said index. From Lemma 1 we know that the only change in the ranking of the similarity index is that non-edges corresponding to covered elements of the universe increase their similarity index values from equal to the value of the edge we intend to hide, to higher than this value. Therefore, we show that the optimal solution that minimizes the values of AUC and AP is the one that covers all elements of the universe, hence the one corresponding to the optimal solution of the underlying Set Cover problem instance.

Theorem 11. *The problem of Evading Link Prediction is NP-complete for both AUC (the Area under ROC curve) and AP (the Average Precision), and for every similarity index in $\mathbb{S}$, i.e., Common Neighbours, Salton, Jaccard, Sørensen, Hub Promoted, Hub Depressed, Leicht-Holme-Newman, Adamic-Adar and Resource Allocation.*

Proof. The problem is trivially in NP, since computing AUC and AP before and after the addition of a given set of edges $A^* \subseteq \hat{A}$ and the removal of a given set of edges $R^* \subseteq \hat{R}$ can be done in polynomial time for every similarity index in $\mathbb{S}$.

Next, we prove that the problem is NP-hard. To this end, we give a reduction from the NP-complete 3-Set Cover problem, as defined in Section 2.3.3 To remind the reader, $U = \{u_1, \dots, u_l\}$ denotes the universe, $S = \{S_1, \dots, S_m\}$ denotes the set of subsets of the universe, and the goal is to determine whether there exist k elements of S the union of which equals U.

Let us assume that $m \geqslant 5$, as all other cases can be easily solved in polynomial time.

Now, for any given similarity index, $\sigma \in \mathbb{S}$, consider the following instance of the problem of Evading Link Prediction $(G, \sigma, f, H, b, \hat{A}, \hat{R})$, where:

- $G = (V, E) = \Gamma(c, S)$, where $c \in \mathbb{N}$ is chosen to be a constant that satisfies Lemma 1 (the lemma states that such a constant exists);
- σ is the similarity index under consideration;
- f is either the AUC or the AP metric;
- $H = \{(u_0, v_0)\}$;
- b is the parameter of the 3-Set Cover problem (where the goal is to determine whether there exist b elements of S the union of which equals U);
- $\hat{A} = \{(S_i, v_0) : S_i \in S\}$;
- $\hat{R} = \emptyset$.

Let us also introduce the following notation:

- $\Upsilon^{<}(G) = \{e \in \bar{E} : \sigma(e) < \sigma(u_0, v_0)\}$ in network G;
- $\Upsilon^{=}(G) = \{e \in \bar{E} : \sigma(e) = \sigma(u_0, v_0)\}$ in network G;
- $\Upsilon^{>}(G) = \{e \in \bar{E} : \sigma(e) > \sigma(u_0, v_0)\}$ in network G.

Observe that $\bar{E}$ is the set of non-edges in (V, E), whereas $\bar{E} \setminus A$ is the set of non-edges in a network of the form $(V, E \cup A)$. For every network of the form $G' = (V, E \cup A)$ where $A \subseteq \hat{A}$, we know from the definition of AUC in Section 6.2 that:

$$AUC(E \cup A, H) = \frac{|\Upsilon^{<}(G')| + \frac{1}{2}|\Upsilon^{=}(G')|}{|\bar{E} \setminus A|}. \tag{6.1}$$

We also know from the definition of AP in Section 6.2 that:

$$AP(E \cup A, H) = \frac{1}{|\Upsilon^{>}(G')| + 1 + \frac{1}{2}|\Upsilon^{=}(G')|}. \tag{6.2}$$

Now, let $U_A = \{u_i : \exists_{S_j \in S_A} u_i \in S_j\}$. Point (b) of Lemma 1 implies that:

$$|\Upsilon^{<}(G')| = |\Upsilon^{<}(G)| - |A|. \tag{6.3}$$

On the other hand, point (a) of Lemma 1 implies that:

$$|\Upsilon^{=}(G')| = |\Upsilon^{=}(G)| - |U_A|, \tag{6.4}$$

as well as:

$$|\Upsilon^{>}(G')| = |\Upsilon^{>}(G)| + |U_A|, \tag{6.5}$$

Equations (6.1), (6.3) and (6.4) imply that:

$$AUC(E \cup A, H) = \frac{|\Upsilon^{<}(G)| - |A| + \frac{1}{2}(|\Upsilon^{=}(G)| - |U_A|)}{|\bar{E}| - |A|} \tag{6.6}$$

On the other hand, equations (6.2) and (6.5) imply that:

$$AP(E \cup A, H) = \frac{1}{|\Upsilon^{>}(G)| + |U_A| + 1 + \frac{1}{2}(|\Upsilon^{=}(G)| - |U_A|)} \tag{6.7}$$

that gives us:

$$AP(E \cup A, H) = \frac{1}{|\Upsilon^{>}(G)| + 1 + \frac{1}{2}(|\Upsilon^{=}(G)| + |U_A|)}. \tag{6.8}$$

Equations (6.6) and (6.8) imply that both AUC and AP decrease with $|U_A|$. Thus, for each of these two metrics an optimal choice of A is one that maximizes $|U_A|$. This happens when $U_A = U$. For any choice of A such that $U_A = U$, the following holds: $\forall_{u_j \in U} \exists_{(S_i, v_0) \in A} u_j \in S_i$. Such an optimal choice of A constitutes a solution to the problem of Evading Link Prediction where $\hat{R} = \emptyset$. It also corresponds directly to a solution to the 3-Set Cover problem. $\square$

Having proven the NP-completeness of the problem of Evading Link Prediction, in the following section we propose an efficient heuristic solution, designed to work well in practice.

6.5 Heuristic Solution

Since finding the optimal solution to the problem of Evading Link Prediction turned out to be intractable, we now focus our efforts on developing heuristic solutions that can be computed in polynomial time. Our heuristic algorithms are based on our analysis of how adding or removing a single edge affects the scores of non-edges.

6.5.1 The Effects of Adding or Removing an Edge

By looking at the formulae of the different similarity indices in $\mathbb{S}$ (see Table 6.1), one can see that the score of every non-edge, $(v, w) \in \bar{E}$, depends solely on the following:

(i) the number of common neighbours of both ends of the non-edge. More formally, $\sigma(v, w)$ depends on $|N(v, w)|$. This observation affects every similarity index in $\mathbb{S}$. More precisely, for every $\sigma \in \mathbb{S}$, the score $\sigma(v, w)$ *increases* with $|N(v, w)|$;

(ii) the degree of each end of the non-edge, but only if both ends have at least one common neighbour. More formally, $\sigma(v, w)$ depends on $\delta(v)$ and $\delta(w)$ if $N(v, w) \neq \emptyset$. This observation does not affect the similarity indices σ^{CN} (*Common Neighbours*), σ^{AA} (*Adamic-Adar*) and σ^{RA} (*Resource Allocation*). As for every other similarity index, *i.e.*, $\sigma \in \mathbb{S} \setminus \{\sigma^{\mathrm{CN}}, \sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$, the score $\sigma(v, w)$ *decreases* with $\delta(v)$ and $\delta(w)$ if $N(v, w) \neq \emptyset$.[2]

(iii) the degree of every common neighbour of the non-edge. More formally, $\sigma(v, w)$ depends on $\delta(x) : x \in N(v, w)$. This observation only affects the indices σ^{AA} (*Adamic-Adar*) and σ^{RA} (*Resource Allocation*). To be more precise, for $\sigma \in \{\sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$, the score $\sigma(v, w)$ *decreases* with $\delta(x) : x \in N(v, w)$.

Based on the above three observations, the *addition* of an edge (u_1, u_2) can only affect the scores of three types of non-edges:

1. A non-edge $(u_1, x) : x \in N(u_2) \setminus N(u_1)$ (such a non-edge is affected by the addition of (u_1, u_2), which adds a new common neighbour of x and u_1, namely u_2), or analogously a non-edge $(u_2, x) : x \in N(u_1) \setminus N(u_2)$. For every such non-edge, the addition of (u_1, u_2) *increases* every $\sigma \in \mathbb{S}$, see observation (i).

2. A non-edge $(u_1, x) : N(u_1, x) \neq \emptyset$ (such a non-edge is affected by the addition of (u_1, u_2), which increases the degree of one end of the non-edge, namely u_1), or analogously a non-edge $(u_2, x) : N(u_2, x) \neq \emptyset$. For every such non-edge, the addition of (u_1, u_2) *decreases* every $\sigma \in \mathbb{S} \setminus \{\sigma^{\mathrm{CN}}, \sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$, see observation (ii).

3. A non-edge $(x, y) : x, y \in N(u_1)$ (such a non-edge is affected by the addition of (u_1, u_2), which increases the degree of a common neighbour of x and y, namely u_1), or analogously a non-edge $(x, y) : x, y \in N(u_2)$. For every such non-edge, the addition of (u_1, u_2) *decreases* every $\sigma \in \{\sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$, see observation (iii).

Conversely, the *removal* of an edge, (u_1, u_2), can only affect the scores of:

1. A non-edge $(u_1, x) : x \in N(u_2) \setminus N(u_1)$, or a non-edge $(u_2, x) : x \in N(u_1) \setminus N(u_2)$. For every such non-edge, the removal of (u_1, u_2) *decreases* every $\sigma \in \mathbb{S}$.

2. A non-edge $(u_1, x) : N(u_1, x) \neq \emptyset$, or a non-edge $(u_2, x) : N(u_2, x) \neq \emptyset$. For every such non-edge, the removal of (u_1, u_2) *increases* every $\sigma \in \mathbb{S} \setminus \{\sigma^{\mathrm{CN}}, \sigma^{\mathrm{AA}}, \sigma^{\mathrm{RA}}\}$.

[2]Regarding σ^{Jac} note that: $|N(v) \cup N(w)| = \delta(v) + \delta(w) - |N(v, w)|$.

3. A non-edge $(x, y) : x, y \in N(u_1)$, or a non-edge $(x, y) : x, y \in N(u_2)$. For every such non-edge, the removal of (u_1, u_2) *increases* every $\sigma \in \{\sigma^{\text{AA}}, \sigma^{\text{RA}}\}$.

Figure 6.2 illustrates a sample network, where the dashed edges are non-edges (*i.e.*, they do not belong to the sample network) whose scores may change as a result of adding the edge (u_1, u_2).

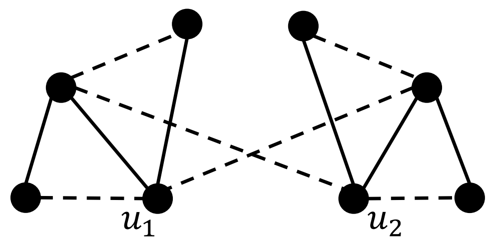

Figure 6.2: A sample network whose edges are drawn as solid lines, along with the non-edges whose scores may change as a result of adding the edge (u_1, u_2) drawn as dashed lines.

In the following subsections, we use the above observations as corner stones based on which we build our heuristic algorithms.

6.5.2 The OTC Heuristic

Based on our analysis in the previous section, we propose a heuristic algorithm that "hides" the edges in H by increasing the similarity scores of the non-edges in $\bar{E}$. This way, if the seeker were to run a link-prediction algorithm, whereby all the non-edges in $\bar{E}$ are ranked according to some similarity index, then our heuristic would increase the ranking of the non-edges in $\bar{E}$, thereby reducing the likelihood that a non-edge $e \in H$ is discovered.

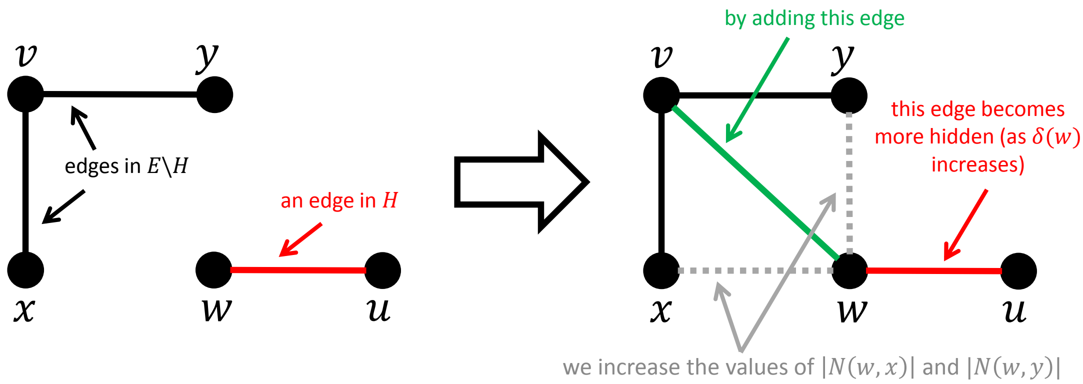

Figure 6.3: An illustration of the main idea behind the OTC heuristic.

In particular, we propose what we call the *Open-Triad-Creation* (OTC) algorithm, which is based on *adding* edges to the network. The purpose behind every such addition is to increase the number of *open triads*, thereby increasing the number of common neighbours of the missing edge in each such triad. The rationale comes from the fact that by increasing the number of common neighbours of a non-edge, its score increases according to *every similarity index* in $\mathbb{S}$ (see Section 6.5.1). Moreover, every time an edge is added to the network, the degree of each end increases. Now if any such end, $w \in V$, happens to also be an end of some edge $(w, u) \in H$, then the increase in $\delta(w)$ would decrease the score of (w, u) according to every similarity index in $\mathbb{S} \setminus \{\sigma^{\text{CN}}, \sigma^{\text{AA}}, \sigma^{\text{RA}}\}$ (again see Section 6.5.1). An example is illustrated in Figure 6.3.

Algorithm 4 The Open-Triad-Creation (OTC) algorithm

Input: A network (V, E), a budget $b \in \mathbb{N}$, a set of edges that can be added $\hat{A} \subseteq \bar{E}$, and a set of edges to be hidden $H \subset E$.

Output: Updated network (V, E).

1: $A' \leftarrow \Big\{(v, w) \in \hat{A} : \Big(\exists u \in N(v) : (u, v) \in H\Big) \vee \Big(\exists u \in N(w) : (u, w) \in H\Big)\Big\}$
2: **for** $i = 1, \dots, b$ **do**
3:     **for** $(v, w) \in A'$ **do**
4:         **if** $\exists_{u \in V}((v, u) \in E \setminus H \wedge (w, u) \in H) \vee ((w, u) \in E \setminus H \wedge (v, u) \in H)$ **then**
5:             $\theta_{(v,w)} \leftarrow -\infty$
6:         **else**
7:             $\theta(v, w) \leftarrow \Big|\Big(N_{(V, E \setminus H)}(v) \ \cup \ N_{(V, E \setminus H)}(w)\Big) \setminus N_{(V, E \setminus H)}(v, w)\Big|$
8:     $(v^*, w^*) \leftarrow \ \arg\max_{(v,w) \in A'} \theta(v, w)$
9:     **if** $\theta(v^*, w^*) > -\infty$ **then**
10:         $E \leftarrow E \cup (v^*, w^*)$

The pseudo-code of the heuristic is presented in Algorithm 4. Specifically, in Line 1 out of all the non-edges that can be added (*i.e.*, all the edges in $\hat{A}$), the algorithm narrows the search to only the subset $A' \subseteq \hat{A}$ in which every non-edge has at least one end that belongs to some edge in H. In lines 3 to 7, the algorithm computes for every non-edge $(v, w) \in A'$ a score $\theta_{(v,w)}$ which reflects the gain from adding (v, w) to the network. Here, in lines 4 and 5 the algorithm ensures that we do not increase the number of common neighbours of some edge in H. Line 7 is responsible for counting the non-edges whose number of common neighbours will increase as a result of adding (v, w). In lines 8 to 10 we choose the non-edge with the highest score and add it to the network. This entire process is repeated until the budget b runs out.

The complexity of a naive implementation of OTC is $\mathcal{O}(b|H||V|^2)$. In more detail, computing score $\theta_{(v,w)}$ for each non-edge $(v, w) \in A'$ can be done in time linear in $|V|$ for each of the $|H||V|$ non-edges. Searching for a non-edge in A' with the maximal score takes $b|H||V|$ operations. Finally, updating the scores after adding each of the b edges can be done in time linear in $|V|$. In Appendix D we show a more efficient (although less intuitive) implementation with complexity $\mathcal{O}(|H||V|^2 + b|H||V|)$ or $\mathcal{O}(|H||V|^2 + b|V| \log(|V|))$ when using a priority queue.

6.5.3 The CTR Heuristic

We now propose an alternative heuristic that focuses on decreasing the scores of the edges in H rather than on increasing the scores of the non-edges in $\bar{E}$ (which was the case with OTC). In particular, we propose what we call the *Closed-Triad-Removal* (CTR) algorithm, which is based on *removing* edges from the network (unlike OTC, which was based on *adding* edges). The purpose behind every such removal is to decrease the number of *closed triads* that contain edges from H, thereby decreasing the number of common neighbours of (v, w). The rationale comes from the fact that by decreasing the number of common neighbours of (v, w), its score decreases according to *every similarity index* in $\mathbb{S}$ (see Section 6.5.1). However, this comes at a cost; it reduces the degree of one end of (v, w), which increases its score according to every similarity index in $\mathbb{S} \setminus \{\sigma^{\text{CN}}, \sigma^{\text{AA}}, \sigma^{\text{RA}}\}$. An example is presented in Figure 6.4.

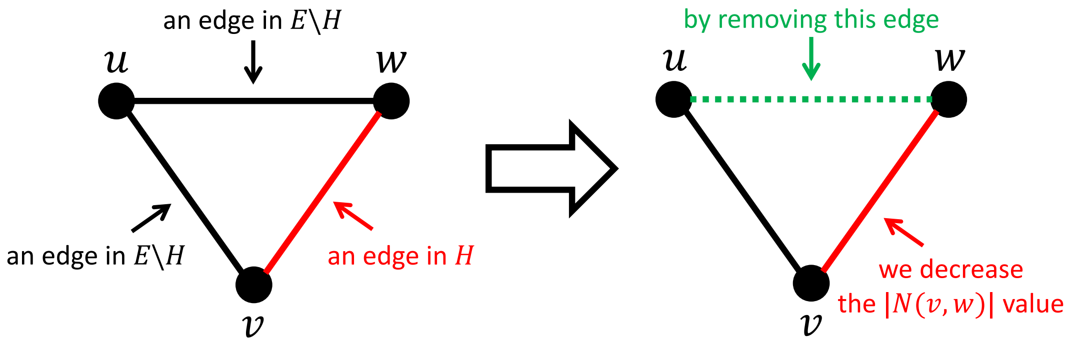

Figure 6.4: An illustration of the main idea behind the CTR heuristic.

The pseudo-code of the CTR heuristic is given in Algorithm 5. Specifically, in Line 1, out of all the edges that can be removed (*i.e.*, all the edges in $\hat{R}$), the algorithm narrows the search to only the subset $R' \subseteq \hat{R}$ in which every edge has at least one end that belongs to some edge in H. After that, in lines 3 to 9, the algorithm computes for every edge $(x, y) \in R'$ a score $\theta_{(x,y)}$ which reflects the gain from removing (x, y) from the network. This is done by simply counting the number of closed triads that contain (x, y) and two other edges, one of which is in H. The edge with the greatest gain is chosen in line 10, and removed from the network in line 12.

The complexity of a naive implementation of CTR (which uses a hash table) is $\mathcal{O}(b|H||V|)$. This is because for every $(v, w) \in H$ the algorithm considers updating the score of every $(v, u) : u \in N(v)$ (there are at most $|V|$ such edges) and every $(w, u) : u \in N(w)$ (again there are at most $|V|$ such edges). This process is repeated b times.

Notice that when $|H| = \omega(\log(|V|))$, an implementation using a *priority queue* is faster, with a complexity of $\mathcal{O}(|H||V| + b|V|\log(|H||V|))$. Such an implementation can be found in Appendix E.

6.5.4 Experimental Design

We now describe experiments performed with OTC and CTR heuristics. They are designed to assess the efficiency of our heuristics in hiding a chosen set of edges.

Setting of every experiment consists of a network, a link prediction algorithm, and a set of edges to be hidden, H. The network is either randomly generated using a certain model, or taken from our dataset of real-life networks. The algorithm is based on one of the similarity indices in $\mathbb{S}$ (see Section 6.2 for a formal definition of these indices). The

Algorithm 5 The Closed-Triad-Removal (CTR) algorithm

Input: A network (V, E), a budget $b \in \mathbb{N}$, a set of edges that can be removed $\hat{R} \subseteq \bar{E}$, and a set of edges to be hidden $H \subset E$.
Output: Updated network (V, E).

$R' \leftarrow \Big\{(v,w) \in \hat{R} : \Big(\exists u \in N(v) : (u,v) \in H\Big) \vee \Big(\exists u \in N(w) : (u,w) \in H\Big)\Big\}$
for $i = 1, \dots, b$ **do**
  for $(x,y) \in R'$ **do**
    $\theta_{(x,y)} \leftarrow 0$
  for $(v,w) \in H$ **do**
    for $u \in N(v,w)$ **do**
      if $(v,u) \in E \setminus H \wedge (w,u) \in E \setminus H$ **then**
        if $(v,u) \in R'$ **then** $\theta_{(v,u)} \leftarrow \theta_{(v,u)} + 1$
        if $(w,u) \in R'$ **then** $\theta_{(w,u)} \leftarrow \theta_{(w,u)} + 1$
  $(v^*, w^*) \leftarrow \arg\max_{(v,w) \in R'} \theta_{(v,w)}$
  if $\theta_{(v^*,w^*)} > 0$ **then**
    $E \leftarrow E \setminus (v^*, w^*)$

set of edges to be hidden, *i.e.*, H, consists of 10% of network's edges, chosen uniformly at random (we chose 10% as it is a typical size of the probe set when evaluating link predication algorithms [158, 94]). Finally, we run either OTC or CTR heuristc algorithm, with the budget being $b = |H|$. In each step of the algorithm, *i.e.*, at the end of every iteration in the main loop, we record the value of the AUC and AP metrics. Each such experiment is repeated 50 times, and the average results are reported along with the 95% confidence intervals.

6.5.5 Simulation Results

Figures 6.5 and 6.6 presents the relative change in the AUC and AP value during the execution of the heuristics (results are shown for only some of the networks under consideration; the remaining results can be found in Appendix F in Figures F.1, F.2, F.3, F.4, F.5, F.6, F.7, F.8, F.9 and F.10. Every point in a subplot represents the average of 50 simulations, with coloured areas representing the 95% confidence intervals. As can be seen, the heuristics are able to reduce the AUC and AP values, with varying levels of success.

Figures 6.7, 6.8, 6.9, and 6.10 evaluate the resilience of each algorithm given different networks and different heuristics, in terms of AUC and AP. More specifically, in each figure columns represent algorithms, rows represent networks, and the color of each cell represents the relative change in the evaluation metric (be it AUC or AP) after running one of our heuristics (be it OTC or CTR). The color represents the average result taken over 50 experiments. The darker the color, the more effective our heuristic turns out to be. Rows and columns are sorted ascendingly based on the sum of the values therein. As can be seen, the degree to which an algorithm can be fooled depends heavily on the heuristic being used. For example, the similarity index *Leicht-Holme-Newman*[85], is the easiest to fool when using the OTC, but the hardest to fool when using CTR.

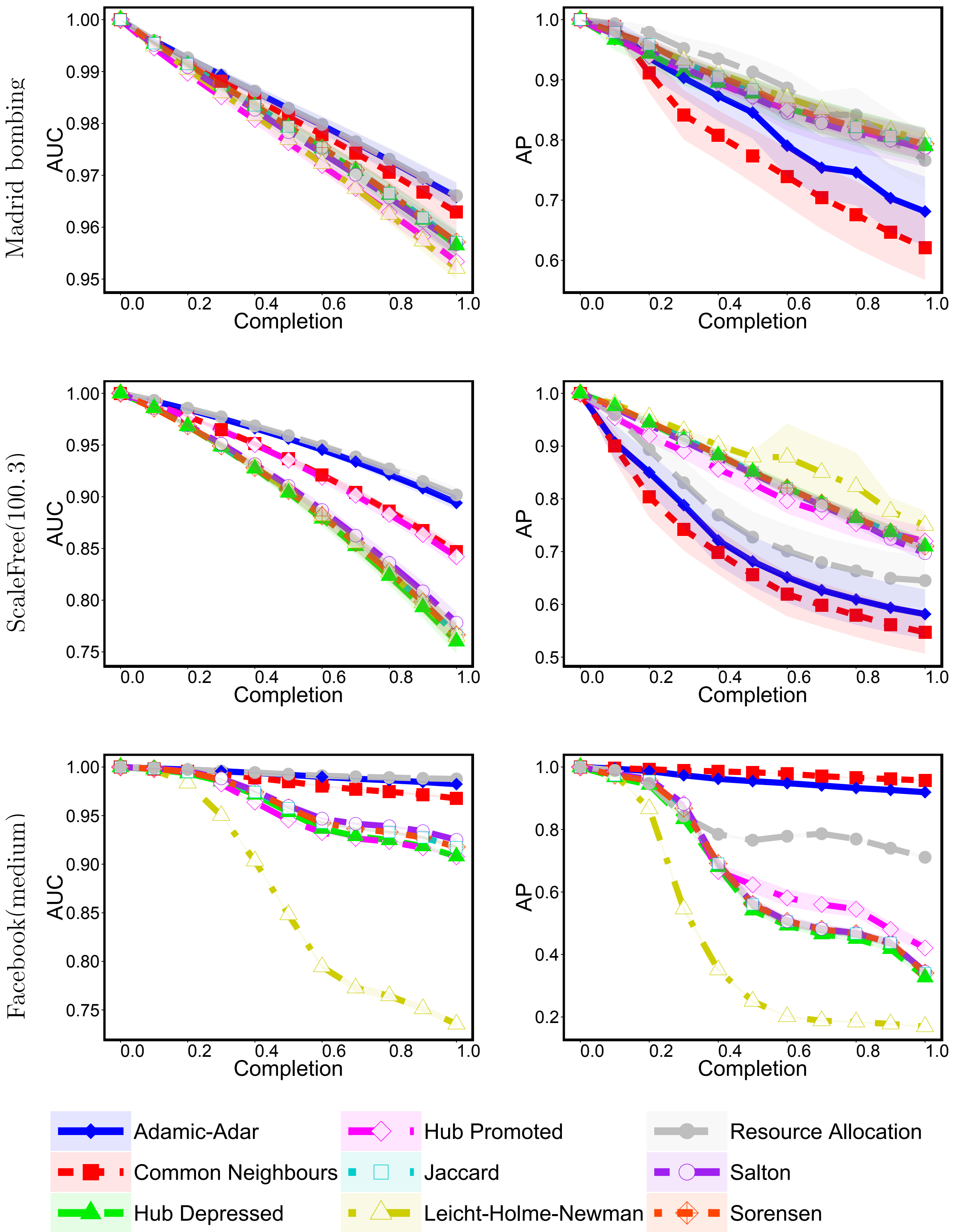

Figure 6.5: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

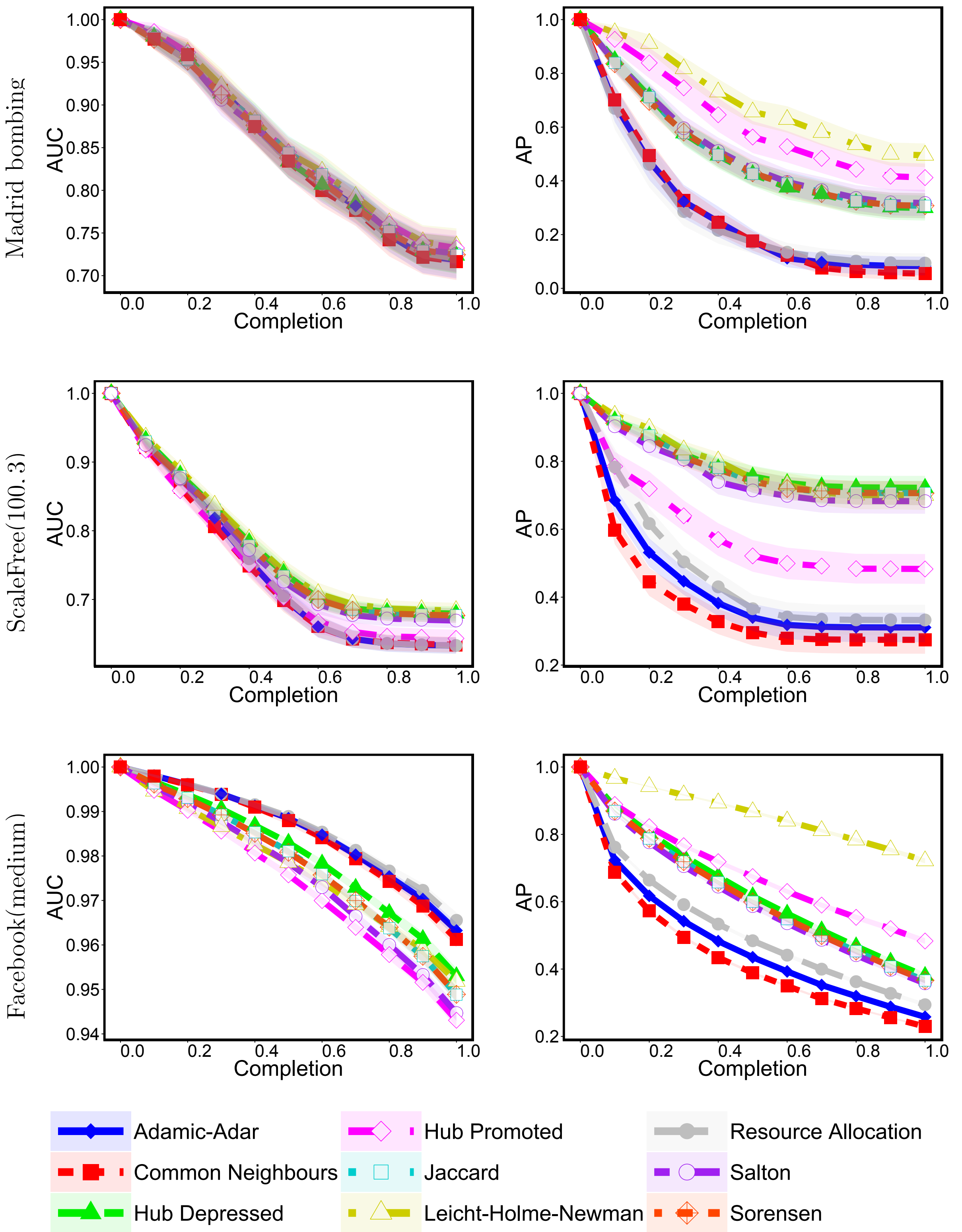

Figure 6.6: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

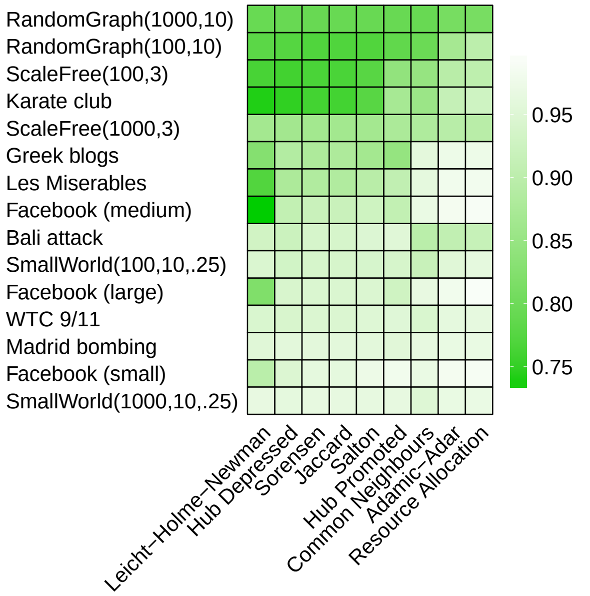

Figure 6.7: Relative changes in the AUC values after running the OTC heuristic.

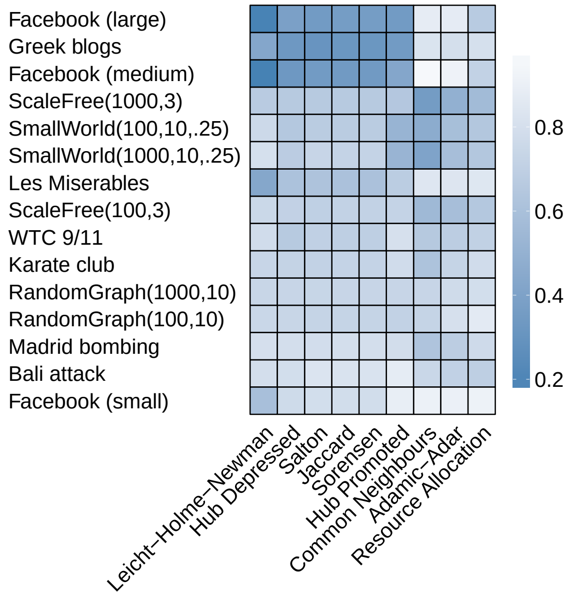

Figure 6.8: Relative changes in the AP values after running the OTC heuristic.

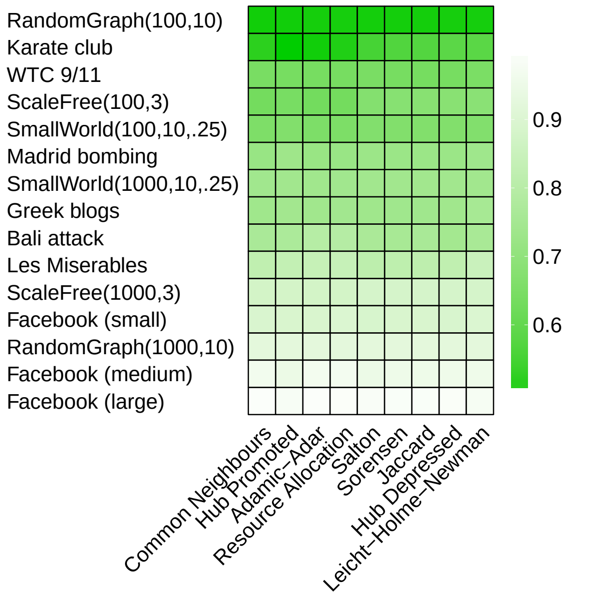

Figure 6.9: Relative changes in the AUC values after running the CTR heuristic.

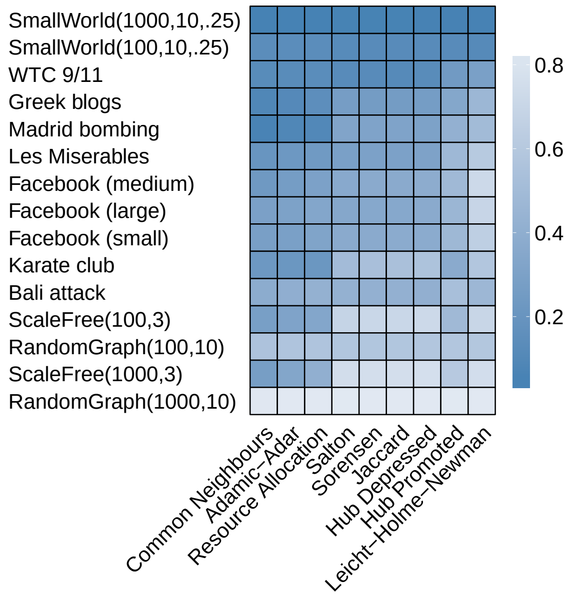

Figure 6.10: Relative changes in the AP values after running the CTR heuristic.

6.6 Concluding Remarks

In this chapter, we introduced and analysed the problem of Evading Link Prediction. We proved that this problem is NP-complete for nine local link prediction algorithms that are widely studied in the literature. Given these hardness results, we focused our efforts on developing scalable heuristics that can be applied by members of the general public. The heuristics do not require any involved technical knowledge nor massive processing power, and can readily be implemented on existing social media.

Our work should be treated as the first step towards a more advanced analysis of evading link prediction algorithms. We now discuss some of the assumptions we made and describe how changing them may lead to potential future work on the subject.

We limited our analysis to local similarity indices, as the most tractable class of link prediction algorithms that are viable for massive social networks. However, analyser with either massive computational power, or interest only in networks of limited size, might also use other types of link prediction algorithms, such as global and semi-local indices, as well as probabilistic models [94]. We identify developing tools for evading this more general class of algorithms as one of possible directions for future work. It's worth noticing that since Evading Link Prediction problem turned out to be NP-complete even for local similarity indices, more complex analysis methods will probably prove at least as much computationally challenging.

Both of our heuristic solutions focus either only or adding edges (in case of the OTC algorithm), or only on removing edges (in case of the CTR algorithm). It might be beneficial to design a heuristic algorithm that makes use of both of those actions. This way potential loss in communication efficiency caused by removing edges from the network can be compensated by creating new connections in another place. At the same time, such a heuristic might result in network modifications that seem more natural, while strategy that only adds edges to the network in the vicinity of hidden connections might actually draw unwanted attention to them.

Other potential venues of expanding presented work include developing settings in which analyser is aware of attempts to hide some of the edges and might take into consideration history of the network structure (not only snapshot of current connections, as we assume in this chapter), as well as analysing networks where nodes have additional attributes, that can indicate their similarity (*e.g.*, political views). As already shown by Bird *et al.* [17], this type of additional data can be used to reveal personal information that is otherwise confidential.

Chapter 7

Conclusions

In this dissertation we considered various aspects of hiding in social networks and preventing detection by social network analysis tools. In Chapter 3 we analysed the problem of reducing one's centrality in a network. However, since simply being connected to a network is not necessarily enough to participate in the activities of the organization in an efficient way, we set additional goal of preserving one's influence on the network. The posed computational problems proved to be intractable, hence finding an optimal solution would be an extremely demanding task. To solve this conundrum we provided a heuristic solution that turns out to be surprisingly effective in practice in a vast majority of considered scenarios. The main goal in Chapter 4 was to prevent detection of a group of leaders of a social network by a centrality measure, *i.e.*, to ensure that no leader would appear in top positions of the centrality's ranking. Again, computational problems turned out to be NP-hard. Notably, this time even considering the simplest centrality measure, *i.e.*, degree centrality, did not make the problem tractable. However, we successfully designed an easy to build and maintain network structure that provides a group of leaders with low positions in centrality rankings, as well as efficient contact with each other and with the rest of the network. In Chapter 5 we investigated the matter of hiding a group of nodes from a different perspective, *i.e.*, we considered the problem of avoiding detection by a community detection algorithms. To evaluate the quality of hiding, we introduced the first measure of how well a group of nodes is concealed within a given community structure. The measure incorporates two notions of hiding, *i.e.*, distributing members of the group across many communities and hiding in the crowd. We used the measure to show effectiveness of a surprisingly simple heuristic that allows a group of nodes to improve their concealment with ease. In Chapter 6 we analysed the issue of preventing detection of a certain set of edges by link prediction algorithms. Again, the problem turned out to be NP-hard from the computational point of view, even when considering a very simple class of link prediction algorithms, *i.e.*, local similarity indices. However, also in this case we developed polynomial time heuristic solutions that allow to hide a chosen set of edges. Contrary to simple intuition, our algorithms worked well even in a setting without complete information.

The following main conclusions can be drawn based on our analysis. Firstly, when considered from the point of view of computational complexity theory, most hiding problems turn out to be NP-hard. Consequently, finding an optimal solution to a given instance is a computationally intensive task, and cannot be effectively solved for most networks.

In many cases even for simplest social network analysis tools, such as degree centrality and common neighbours similarity index, considered problems prove to be intractable (see Theorems 5 and 11). It suggests that hardness of the hiding questions is the result of a structure of the problem itself, and not only of the complexity of the social network analysis tools that we try to evade. Since in all considered settings we aimed to use the most basic tools (that can be used even to analyse massive social networks), we expect the problems of hiding from more complex social network analysis techniques to be even less tractable.

Secondly, even though the conclusion drawn in the previous paragraph may suggest that the network analysers have nothing to be worried about, this is certainly not the case. For every considered problem we managed to find a simple polynomial-time heuristic solution that turned out to be effective in practice, for both artificial and real-life datasets. Each of these algorithms takes into account several factors that are crucial while developing solutions that are implementable in practice. Our heuristic solutions do not need complete knowledge about entire structure of the network, as such information is rarely available in practice. Typically, we only know the structure of our direct network vicinity, *i.e.*, connections of our neighbours and sometimes neighbours of our neighbours. Our heuristic algorithms do not require extensive computations, *i.e.*, they are all polynomial time (for low-ordered polynomials). As real-life network can be massive, it is important to develop solutions that do not require extensive computational power. Most of our heuristic solutions are also simple enough that they can be used even by lay people, without specialized knowledge about algorithm design. Consequently, they can be implemented by any social media users. All things considered, even though finding the optimal solutions to the hiding problems is a very hard task from a computational point of view, coming up with a solution that provides an acceptable level of concealment is often relatively straightforward.

Third conclusion that can be drawn from this dissertation is the need to extend the approaches used to analyse social networks, especially when considering the dark networks. Nearly all social network analysis tools treat the nodes of the network as oblivious entities. While this might be true in many cases, when considering criminal and terrorist organizations, their members have clear incentive to falsify the results of the analysis. As we have shown in this dissertation, this goal can be achieved even with relatively simple means, *e.g.*, by modifying the structure of connections between members of the organization or by reorganizing its communication channels. This indicates, that when performing the analysis on this particular type of networks, nodes should be considered as strategic players. Since today's criminal and terrorist organizations are aware of being under constant surveillance, they can modify their behaviour in order to minimize the damage. This possibility should be brought to attention of police forces and intelligence agencies utilizing social network analysis tools.

Fourth conclusion is the need to develop new social network analysis tools, that are aware of the potential attempts to avoid and falsify the analysis. This dissertation describes flaws of many of the currently used techniques and risks that are the result of commonly accepted assumptions. Next generations of social network analysis tools should adapt to the fact that their object of analysis is not static and unaware of their existence. One of possible ways to achieve this, is developing techniques taking into account more than just the structure of connections between network members, but also specific information about each of them. While current software is usually just visualising this type of data, its usage should be built into analytic algorithms. This may lead to more effective analysis of dark networks and, as a result, more effective ways of fighting criminal and terrorist organizations.

We now describe a number of limitations of our study. First limitation is that even though we consider the evader who is strategic, the analyser of the network is still not. We believe that such a setting accurately describes the current use of social network analysis tools. More often than not, they are applied without any consideration of the possible reaction of the analysed network's members. Hence, we investigate possible strategies of the evaders, while we assume that the action of the analyser is fixed, *i.e.*, she is only using a particular social network analysis tools. Our work should be treated as a first step in a more advanced analysis of the problem of hiding in social networks. The next step should be introducing strategic analyser, who is aware that evaders may intend to falsify the results of the analysis. Approach to this kind of problems should be more game-theoretic, possibly similar to the Stackelberg games (Leader-Follower models).

Second limitation is the number and nature of considered real-life network datasets. Unfortunately, there is a very limited number of dark networks datasets that are available to the public. Further, many of them contain fairly few nodes, representing only a single cell, in which case it is difficult to obtain significant simulation results. Others are massive networks, composed of many different organizations, where computing evaluative characteristics presented as part of results, *e.g.*, influence values, is an extremely demanding computational task. In our simulations we used a variety of dark networks, ranging from small-scale (9/11 terrorist network), to medium-sized organizations (Madrid bombing network). We also performed tests on other real-life networks, such as fragments of social media networks (Facebook, Twitter, Google Plus) and artificially generated datasets, to provide a wide variety of tested subjects. Another problem is the nature of datasets, in that most of them consist only of the network structure. There are very few available datasets containing information about nodes that can be used as input for social network analysis tools. Hence, in our simulations we used techniques that require only connections between nodes to perform analysis.

Third limitation is the range of considered social network analysis tools. In all chapters we intended to apply the most widely used techniques. It turns out that it is usually the simplest class that is the most popular and investigated in the literature. Part of the reason is that it enables the user to analyse even large networks. While dark networks with a few dozen nodes are a viable subject of analysis, datasets based on social media sites often consist of thousands or tens of thousands of nodes. In such cases, tools that are used to analyse the networks have to be both effective and computationally tractable. It may turn out that more complicated social network analysis techniques are also more difficult to evade. However, due to their limited use in case of larger networks, we decided to concentrate on more popular and simpler tools, leaving the analysis of their more advanced counterparts as a possible venue of future work.

We now describe other possible directions of future work. In this dissertation we considered the problem of hiding from three different social network analysis tools, namely centrality measures, community detection algorithms, and link prediction algorithms. However, there exists a plethora of other kinds of tools and algorithms, the evasion of which can be potentially beneficial for analysed parties. Good example is the class of role assignment algorithms that recognize role that each node plays in an organization. This is yet another way to identify key nodes in the network, and falsifying its results may be of interest to the ringleaders of various organizations for the same reasons that we listed for centrality measures. Another potential objects of analysis are network comparison measures that express similarity between network structures. Dark networks may wish to appear less like other criminal and terrorist organizations to avoid being classified as one. Other networks may wish to hide their true robustness, or to strategically falsify values of other network statistics, such as eccentricity, communication cost, or clustering coefficient.

Another potential venue of future work is investigating the hiding problems in different types of networks. One of those types is the class of multi-layered networks. If examined more closely, our social connections can be actually divided into many categories, *e.g.*, accordingly to the platform of communication (face-to-face meetings, telephone conversations, different types of social media), nature of relation (blood relatives, associates, casual acquaintances), or time frame of contacts. Such a distinction can be modelled as a network with multiple layers. This adds another dimension to the problems of hiding. Examples of research questions are how should one distribute the budget of network modifications among the layers (is it more beneficial to focus on a single layer, or rather make small changes in each of them), or does considering multiple layers make hiding easier or more complicated (as analyser has to choose one of the multiple possible ways of implementing social network analysis tools for multi-layered networks).

Another potential question is the analysis of dynamic networks. In this dissertation we consider static network structure, *i.e.*, we assume that analyser performs her activities on a snapshot of a social network in a given moment. We showed that in this setting identification of nodes that intend to hide is an extremely demanding task. Potential hope for analyser is considering the history of changes in networks structure, *i.e.*, a dynamic version of the network. Since many edge modifications being part of hiding process happen locally, the analyser may focus her efforts on areas of the network with many changes, thus making it easier to identify hiding entities. At the same time, time dimension of the domain allows to design even more complex hiding strategies, where edge changes are intentionally distributed over long time or performed further away in the network to avoid suspicions.

Different type of networks (although somewhat similar to the dynamic networks) are the temporal networks. Edges of temporal network are active only in particular periods of time, *e.g.*, may represent a history of contacts between organisation members. Hiding problems in such a network give yet another dimension of complexity, with possible actions including changing the hours of contact between members, or fine-tuning frequency of sending messages in an attempt to conceal importance of a connection.

Bibliography

[1] European Data Protection Supervisor, Meeting the Challenges of Big Data, Opinion 7/2015.

[2] Global Government Requests Report. `https://govtrequests.facebook.com/`. Accessed: 16-02-2017.

[3] Selfie Soldiers: Russia Checks in to Ukraine. `https://news.vice.com/video/selfie-soldiers-russia-checks-in-to-ukraine`. Accessed: 16-02-2017.

[4] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[5] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.

[6] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[7] R. Amer and J. M. Giménez. A connectivity game for graphs. *Mathematical Methods of Operations Research*, 60(3):453–470, 2004.

[8] J. M. Anthonisse. The rush in a graph. *Amsterdam: University of Amsterdam Mathematical Centre*, 1971.

[9] B. S. Baker. Approximation algorithms for np-complete problems on planar graphs. *Journal of the ACM (JACM)*, 41(1):153–180, 1994.

[10] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[11] V. Barber. The evolution of al qaeda's global network and al qaeda core's position within it: A network analysis. *Perspectives on Terrorism*, 9(6), 2015.

[12] A. Bavelas. A mathematical model for group structures. *Human organization*, 7(3):16–30, 1948.

[13] M. A. Beauchamp. An improved index of centrality. *Behavioral Science*, 10(2):161–163, 1965.

[14] G. A. Bello, M. Angus, N. Pedemane, J. K. Harlalka, F. H. Semazzi, V. Kumar, and N. F. Samatova. Response-guided community detection: application to climate index discovery. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 736–751. Springer, 2015.

[15] G. Berlusconi, F. Calderoni, N. Parolini, M. Verani, and C. Piccardi. Link prediction in criminal networks: A tool for criminal intelligence analysis. *PloS one*, 11(4):e0154244, 2016.

[16] D. Bilò, L. Gualà, and G. Proietti. Improved approximability and non-approximability results for graph diameter decreasing problems. *Theoretical Computer Science*, 417:12–22, 2012.

[17] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *Proceedings of the 2006 international workshop on Mining software repositories*, pages 137–143. ACM, 2006.

[18] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[19] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer, 1999.

[20] S. P. Borgatti. Identifying sets of key players in a social network. *Computational & Mathematical Organization Theory*, 12(1):21–34, 2006.

[21] S. P. Borgatti, K. M. Carley, and D. Krackhardt. On the robustness of centrality measures under conditions of imperfect data. *Social networks*, 28(2):124–136, 2006.

[22] K. Boyd, K. H. Eng, and C. D. Page. Area under the precision-recall curve: Point estimates and confidence intervals. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 451–466. Springer, 2013.

[23] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing modularity is hard. *arXiv preprint physics/0608255*, 2006.

[24] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. *On modularity-np-completeness and beyond*. Citeseer, 2006.

[25] P. L. Brantingham, M. Ester, R. Frank, U. Glässer, and M. A. Tayebi. Co-offending network mining. In *Counterterrorism and Open Source Intelligence*, pages 73–102. Springer, 2011.

[26] C. V. Cannistraci, G. Alanis-Lobato, and T. Ravasi. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific reports*, 3, 2013.

[27] K. M. Carley. Dynamic network analysis. In *Dynamic social network modeling and analysis: Workshop summary and papers*, pages 133–145. Citeseer, 2003.

[28] K. M. Carley. Destabilization of covert networks. *Computational & Mathematical Organization Theory*, 12(1):51–66, 2006.

[29] K. M. Carley, J.-S. Lee, and D. Krackhardt. Destabilizing networks. *Connections*, 24(3):31–34, 2001.

[30] J. T. Chatagnier, A. Mintz, and Y. Samban. The decision calculus of terrorist leaders. *Perspectives on Terrorism*, 6(4-5), 2012.

[31] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1029–1038. ACM, 2010.

[32] A. Clauset, M. E. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[33] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[34] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*, volume 6. MIT press Cambridge, 2001.

[35] C. D. Correa, T. Crnovrsanin, and K.-L. Ma. Visual reasoning about social networks using centrality sensitivity. *Visualization and Computer Graphics, IEEE Transactions on*, 18(1):106–120, 2012.

[36] P. Crescenzi, G. D'angelo, L. Severini, and Y. Velaj. Greedily improving our own closeness centrality in a network. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(1):9, 2016.

[37] S. F. Crone and D. Soopramanien. Predicting customer online shopping adoption-an evaluation of data mining and market modelling approaches. In *DMIN*, pages 215–221, 2005.

[38] N. Crossley, G. Edwards, E. Harries, and R. Stevenson. Covert social movement networks and the secrecy-efficiency trade off: The case of the {UK} suffragettes (1906–1914). *Social Networks*, 34(4):634 – 644, 2012.

[39] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[40] B. Debatin, J. P. Lovejoy, A.-K. Horn, and B. N. Hughes. Facebook and online privacy: Attitudes, behaviors, and unintended consequences. *Journal of Computer-Mediated Communication*, 15(1):83–108, 2009.

[41] S. Dehghani, M. A. Fazli, J. Habibi, and S. Yazdanbod. Using shortcut edges to maximize the number of triangles in graphs. *Operations Research Letters*, 43(6):586–591, 2015.

[42] F. Demiroz and N. Kapucu. Anatomy of a dark network: the case of the turkish ergenekon terrorist organization. *Trends in organized crime*, 15(4):271–295, 2012.

[43] W. Enders and X. Su. Rational terrorists and optimal network structure. *Journal of Conflict Resolution*, 51(1):33–57, 2007.

[44] P. Erdős and A. Rényi. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[45] S. F. Everton. Network topography, key players and terrorist networks. *Connections*, 2009.

[46] S. F. Everton and N. Roberts. Strategies for combating dark networks. *Paper presented at the Sunbelt XXIX: The Annual Meeting of the International Network of Social Net work Analysis.*, 2011.

[47] J. D. Farley. Breaking al qaeda cells: A mathematical analysis of counterterrorism operations (a guide for risk assessment and decision making). *Studies in Conflict & Terrorism*, 26(6):399–411, 2003.

[48] T. Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[49] E. Ferrara, P. De Meo, S. Catanese, and G. Fiumara. Detecting criminal organizations in mobile phone networks. *Expert Systems with Applications*, 41(13):5733–5750, 2014.

[50] M. Fire, G. Katz, L. Rokach, and Y. Elovici. Links reconstruction attack. In *Security and Privacy in Social Networks*, pages 181–196. Springer, 2013.

[51] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[52] T. L. Frantz, M. Cataldo, and K. M. Carley. Robustness of centrality measures under uncertainty: Examining the role of network topology. *Computational and Mathematical Organization Theory*, 15(4):303, 2009.

[53] F. Frati, S. Gaspers, J. Gudmundsson, and L. Mathieson. Augmenting graphs to minimize the diameter. In *International Symposium on Algorithms and Computation*, pages 383–393. Springer, 2013.

[54] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[55] L. C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[56] J. L. Gardy, J. C. Johnston, S. J. H. Sui, V. J. Cook, L. Shah, E. Brodkin, S. Rempel, R. Moore, Y. Zhao, R. Holt, et al. Whole-genome sequencing and social-network analysis of a tuberculosis outbreak. *New England Journal of Medicine*, 364(8):730–739, 2011.

[57] L. Getoor and C. P. Diehl. Link mining: a survey. *ACM SIGKDD Explorations Newsletter*, 7(2):3–12, 2005.

[58] J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

[59] M. Granovetter. Threshold models of collective behavior. *American journal of sociology*, pages 1420–1443, 1978.

[60] R. Guimera, S. Mossa, A. Turtschi, and L. N. Amaral. The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences*, 102(22):7794–7799, 2005.

[61] I. Hamed, M. Charrad, and N. B. Ben Saoud. *Which Centrality Metric for Which Terrorist Network Topology?*, pages 195–208. Springer International Publishing, Cham, 2016.

[62] B. Hayes. Connecting the dots can the tools of graph theory and social-network studies unravel the next big plot? *American Scientist*, 94(5):400–404, 2006.

[63] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Preventing private information inference attacks on social networks. *IEEE TKDE*, 25(8):1849–1862, 2013.

[64] I2. Analyst's Notebook 8, Social Network Analysis Whitepaper. `www.i2group.com`, 2010.

[65] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[66] R. Janssen and H. Monsuur. Stable network topologies using the notion of covering. *European Journal of Operational Research*, 218(3):755–763, 2012.

[67] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of computer and system sciences*, 9(3):256–278, 1974.

[68] N. F. Johnson, M. Zheng, Y. Vorobyeva, A. Gabriel, H. Qi, N. Velasquez, P. Manrique, D. Johnson, E. Restrepo, C. Song, and S. Wuchty. New online ecology of adversarial aggregates: Isis and beyond. *Science*, 352(6292):1459–1463, 2016.

[69] A. M. Kaplan and M. Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.

[70] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[71] B. Karrer, E. Levina, and M. E. Newman. Robustness of community structure in networks. *Physical Review E*, 77(4):046119, 2008.

[72] M. Kearns, A. Roth, Z. S. Wu, and G. Yaroslavtsev. Private algorithms for the protected in social network search. *Proceedings of the National Academy of Sciences*, page 201510612, 2016.

[73] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[74] M. Khayat. Jihadis Shift To Using Secure Communication App Telegram's Channels Service. *Inquiry & Analysis Series*, (1198), 2015.

[75] J. Kilberg. A basic model explaining terrorist group organizational structure. *Studies in Conflict & Terrorism*, 35(11):810–830, 2012.

[76] G. King, J. Pan, and M. E. Roberts. How censorship in china allows government criticism but silences collective expression. *American Political Science Review*, 107(02):326–343, 2013.

[77] G. King, J. Pan, and M. E. Roberts. Reverse-engineering censorship in china: Randomized experimentation and participant observation. *Science*, 345(6199):1251722, 2014.

[78] D. E. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*, volume 37. Addison-Wesley Reading, 1993.

[79] S. Koschade. A social network analysis of jemaah islamiyah: The applications to counterterrorism and intelligence. *Studies in Conflict & Terrorism*, 29(6):559–575, 2006.

[80] V. E. Krebs. Mapping networks of terrorist cells. *Connections*, 24(3):43–52, 2002.

[81] A. Kumar and N. Rathore. Improving attribute inference attack using link prediction in online social networks. In *Recent Advances in Mathematics, Statistics and Computer Science*, pages 494–503. 2016.

[82] T. U. Kuzubaş, I. Ömercikoğlu, and B. Saltoğlu. Network centrality measures and systemic risk: An application to the turkish financial crisis. *Physica A: Statistical Mechanics and its Applications*, 405:203–215, 2014.

[83] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical review E*, 80(5):056117, 2009.

[84] J. I. Lane, V. Stodden, S. Bender, and H. Nissenbaum, editors. *Privacy, big data, and the public good: frameworks for engagement.* 2014.

[85] E. A. Leicht, P. Holme, and M. E. Newman. Vertex similarity in networks. *Physical Review E*, 73(2):026120, 2006.

[86] J. Leskovec and J. J. Mcauley. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pages 539–547, 2012.

[87] D. Li, Z. Xu, S. Li, X. Sun, A. Gupta, and K. Sycara. Link recommendation for promoting information diffusion in social networks. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 185–186. ACM, 2013.

[88] X. Li and H. Chen. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. *Decision Support Systems*, 54(2):880–890, 2013.

[89] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[90] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Inferring private information using social network data. In *Proceedings of the 18th international conference on World wide web*, pages 1145–1146. ACM, 2009.

[91] R. Lindelauf, P. Borm, and H. Hamers. The influence of secrecy on the communication structure of covert networks. *Social Networks*, 31(2):126–137, 2009.

[92] R. Lindelauf, H. Hamers, and B. Husslage. Cooperative game theoretic centrality analysis of terrorist networks: The cases of jemaah islamiyah and al qaeda. *European Journal of Operational Research*, 229(1):230–238, 2013.

[93] X. Liu, E. Patacchini, Y. Zenou, and L.-F. Lee. Criminal networks: Who is the key player? 2012.

[94] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[95] J. Magouirk, S. Atran, and M. Sageman. Connecting terrorist networks. *Studies in Conflict & Terrorism*, 31(1):1–16, 2008.

[96] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[97] V. Mayer-Schnberger. *Big Data: A Revolution That Will Transform How We Live, Work and Think. Viktor Mayer-Schnberger and Kenneth Cukier.* John Murray Publishers, UK, 2013.

[98] R. M. Medina. Social network analysis: a case study of the islamist terrorist network. *Security Journal*, 27(1):97–121, 2014.

[99] A. Meyerson and B. Tagiku. Minimizing average shortest path distances via shortcut edge addition. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 272–285. Springer, 2009.

[100] T. Michalak, T. Rahwan, O. Skibski, and M. Wooldridge. Defeating terrorist networks with game theory. *IEEE Intelligent Systems*, 2015.

[101] T. P. Michalak, T. Rahwan, N. R. Jennings, P. L. Szczepański, O. Skibski, R. Narayanam, and M. J. Wooldridge. Computational analysis of connectivity games with applications to the investigation of terrorist networks. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 293–301. AAAI Press, 2013.

[102] T. P. Michalak, T. Rahwan, and M. Wooldridge. Strategic social network analysis. In *AAAI*, 2017.

[103] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel. You are who you know: Inferring user profiles in online social networks. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 251–260, New York, NY, USA, 2010. ACM.

[104] I.-C. Moon. *Destabilization of adversarial organizations with strategic interventions.* ProQuest, 2008.

[105] C. Morselli, C. Giguère, and K. Petit. The efficiency/security trade-off in criminal networks. *Social Networks*, 29(1):143 – 153, 2007.

[106] M. E. Newman. Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102, 2001.

[107] M. E. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[108] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[109] A. Nordrum. Pro-ISIS Online Groups Use Social Media Survival Strategies to Evade Authorities, 2016.

[110] G. K. Orman and V. Labatut. A comparison of community detection algorithms on artificial networks. In *Discovery science*, pages 242–256. Springer, 2009.

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

[112] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[113] M. Papagelis, F. Bonchi, and A. Gionis. Suggesting ghost edges for a smaller world. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2305–2308. ACM, 2011.

[114] N. Parotsidis, E. Pitoura, and P. Tsaparas. Selecting shortcuts for a smaller world. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 28–36. SIAM, 2015.

[115] N. Parotsidis, E. Pitoura, and P. Tsaparas. Centrality-aware link recommendations. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 503–512. ACM, 2016.

[116] D. Paulo, B. Fischl, T. Markow, M. Martin, and P. Shakarian. Social network intelligence analysis to combat street gang violence. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1042–1049. ACM, 2013.

[117] A. Perliger and A. Pedahzur. Social network analysis in the study of terrorism and political violence. *PS: Political Science & Politics*, 44(01):45–50, 2011.

[118] S. Perumal, P. Basu, and Z. Guan. Minimizing eccentricity in composite networks via constrained edge additions. In *Military Communications Conference, MILCOM 2013-2013 IEEE*, pages 1894–1899. IEEE, 2013.

[119] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer, 2005.

[120] A. Popescul and L. H. Ungar. Statistical relational learning for link prediction. In *IJCAI workshop on learning statistical models from relational data*, volume 2003. Citeseer, 2003.

[121] J. Qin, J. J. Xu, D. Hu, M. Sageman, and H. Chen. Analyzing terrorist networks: A case study of the global salafi jihad network. In *Intelligence and security informatics*, pages 287–304. Springer, 2005.

[122] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, 2002.

[123] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[124] S. Ressler. Social network analysis as an approach to combat terrorism: Past, present, and future research. *Homeland Security Affairs*, 2(2):1–10, 2006.

[125] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, 2010.

[126] R. Rothenberg. From whole cloth: Making up the terrorist network. *New York Times*, 2001.

[127] S. Saha, A. Adiga, B. A. Prakash, and A. K. S. Vullikanti. Approximation algorithms for reducing the spectral radius to control epidemic spread. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 568–576. SIAM, 2015.

[128] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA, 1986.

[129] R. Scarborough. Islamic State using leaked Snowden info to evade U.S. intelligence, 2014.

[130] J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. In *Applications of Data Mining to Electronic Commerce*, pages 115–153. Springer, 2001.

[131] J. Scott. *Social network analysis*. Sage, 2012.

[132] M. A. Shaikh, J. Wang, Z. Yang, and Y. Song. Graph structural mining in terrorist networks. In *Advanced Data Mining and Applications*, pages 570–577. Springer, 2007.

[133] M. E. Shaw. Group structure and the behavior of individuals in small groups. *The Journal of Psychology*, 38(1):139–149, 1954.

[134] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[135] D. Simpson and P. Brown. NSA mines facebook for connections, including americans' profiles. *CNN*, 2013. Accessed: May 25, 2016.

[136] T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biol. Skr.*, 5:1–34, 1948.

[137] M. K. Sparrow. The application of network analysis to criminal intelligence: An assessment of the prospects. *Social networks*, 13(3):251–274, 1991.

[138] F. Spezzano, V. Subrahmanian, and A. Mannes. Stone: shaping terrorist organizational network efficiency. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 348–355. ACM, 2013.

[139] R. Stevenson and N. Crossley. Change in covert social movement networks: The 'inner circle'of the provisional irish republican army. *Social Movement Studies*, 13(1):70–91, 2014.

[140] M. A. Tayebi, L. Bakker, U. Glasser, and V. Dabbaghian. Locating central actors in co-offending networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 171–179. IEEE, 2011.

[141] R. W. Taylor, E. J. Fritsch, and J. Liederbach. *Digital crime and digital terrorism*. Prentice Hall Press, 2014.

[142] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 245–254. ACM, 2012.

[143] M. Tsvetovat and K. M. Carley. Generation of realistic social network datasets for testing of analysis and simulation tools. Technical report, DTIC Document, 2005.

[144] M. Tsvetovat and K. M. Carley. Structural knowledge and success of anti-terrorist activity: The downside of structural equivalence. *Institute for Software Research*, page 43, 2005.

[145] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265, 1937.

[146] V. V. Vazirani. *Approximation algorithms.* Springer Science & Business Media, 2013.

[147] M. Waniek, T. P. Michalak, and T. Rahwan. Hiding relationships in a social network. Mimeo, available on request.

[148] M. Waniek, T. P. Michalak, T. Rahwan, and M. Wooldridge. Hiding individuals and communities in a social network. *CoRR*, abs/1608.00375, 2016.

[149] M. Waniek, T. P. Michalak, T. Rahwan, and M. Wooldridge. On the construction of covert networks. In *Proceedings of the 2017 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

[150] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.

[151] J. Xie, S. Kelley, and B. K. Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys (csur)*, 45(4):43, 2013.

[152] J. Xu and H. Chen. Criminal network analysis and visualization. *Communications of the ACM*, 48(6):100–107, 2005.

[153] J. Xu and H. Chen. The topology of dark networks. *Communications of the ACM*, 51(10):58–65, 2008.

[154] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, 2015.

[155] Z. Yin, M. Gupta, T. Weninger, and J. Han. A unified framework for link recommendation using random walks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 152–159. IEEE, 2010.

[156] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.

[157] K. Zafiropoulos, V. Vrana, and D. Vagianos. Bloggers' community characteristics and influence within greek political blogosphere. *Future Internet*, 4(2):396–412, 2012.

[158] P. Zhang, X. Wang, F. Wang, A. Zeng, and J. Xiao. Measuring the robustness of link prediction algorithms under noisy environment. *Scientific reports*, 6, 2016.

[159] E. Zheleva and L. Getoor. To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 531–540, New York, NY, USA, 2009. ACM.

[160] T. Zhou, L. Lü, and Y.-C. Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.

Appendix A

ROAM Simulation Results

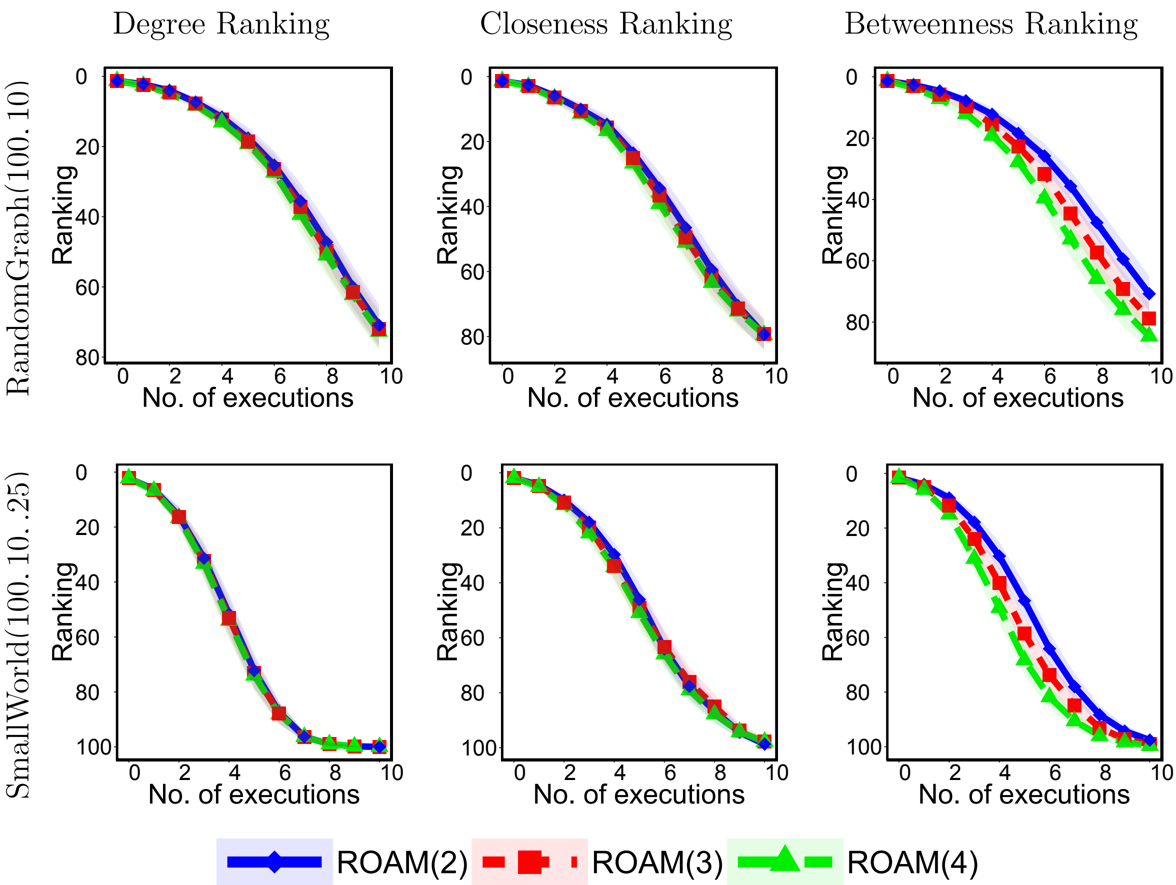

Figure A.1: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

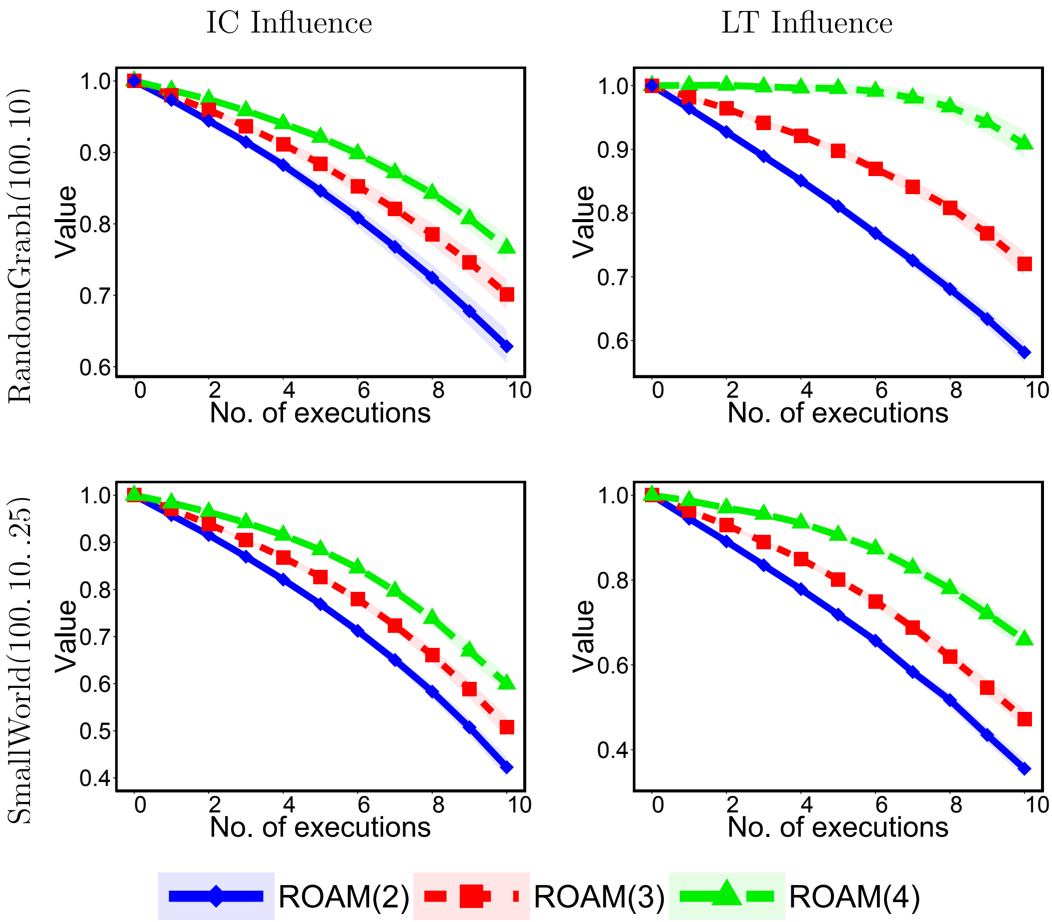

Figure A.2: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

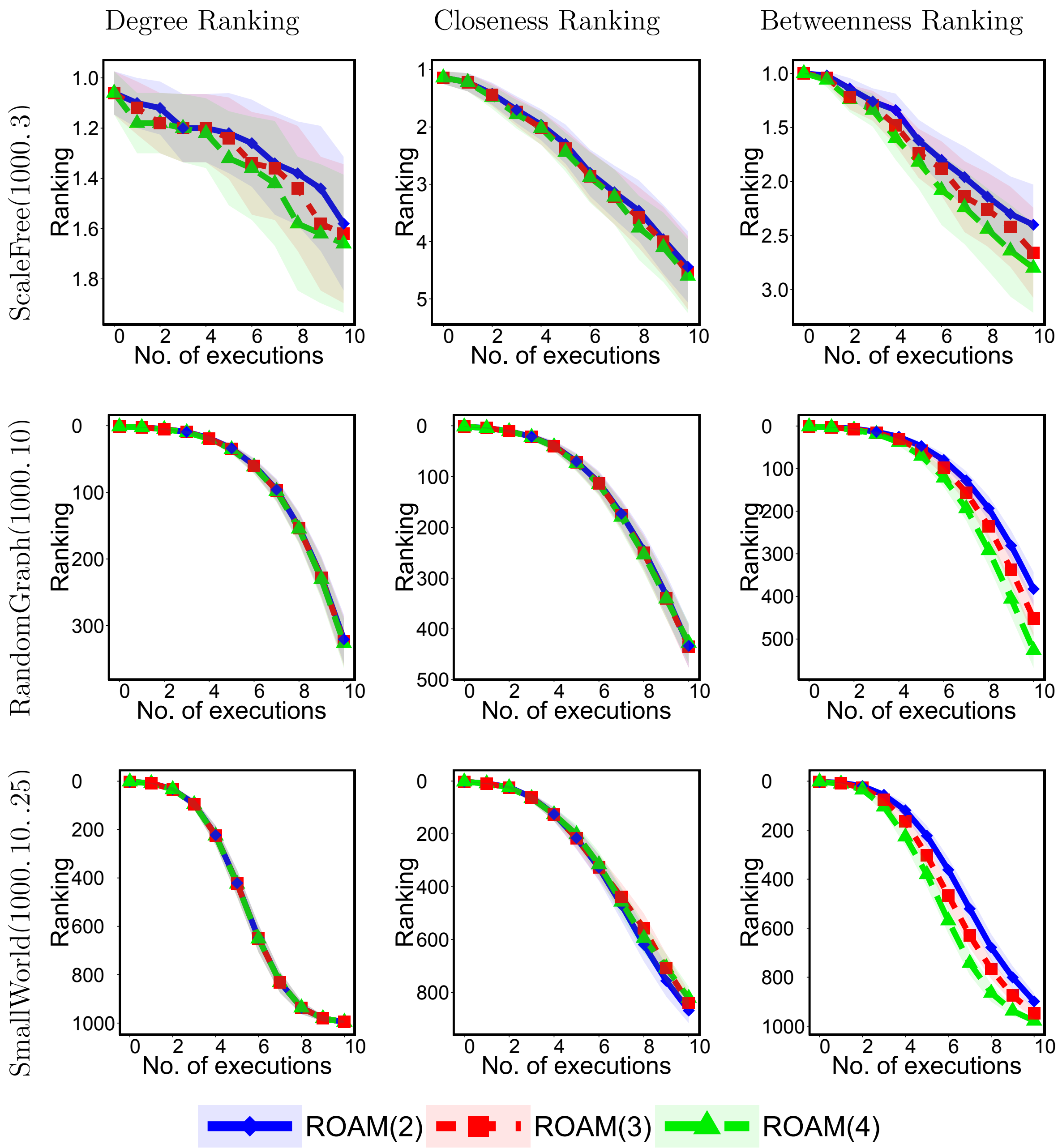

Figure A.3: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

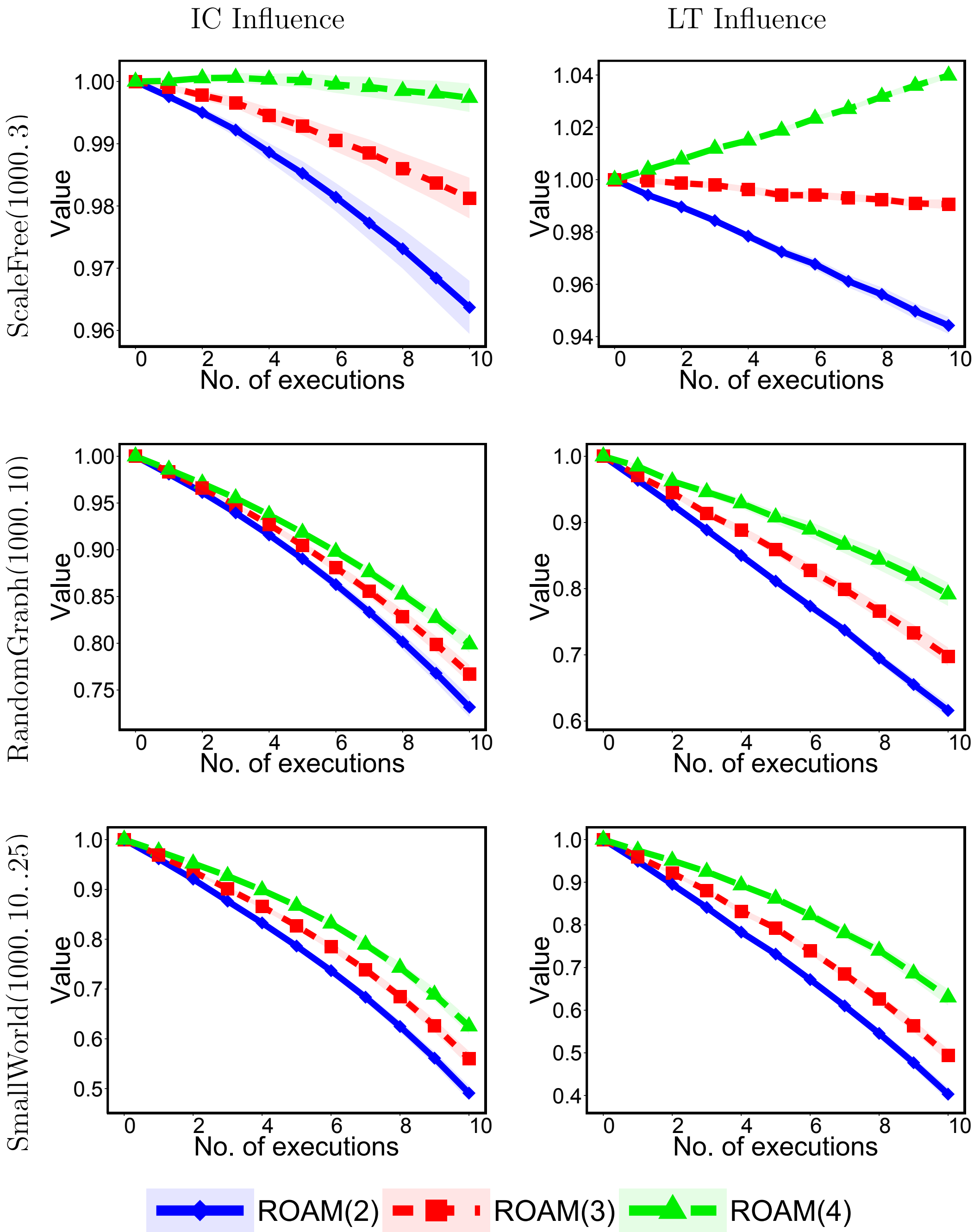

Figure A.4: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

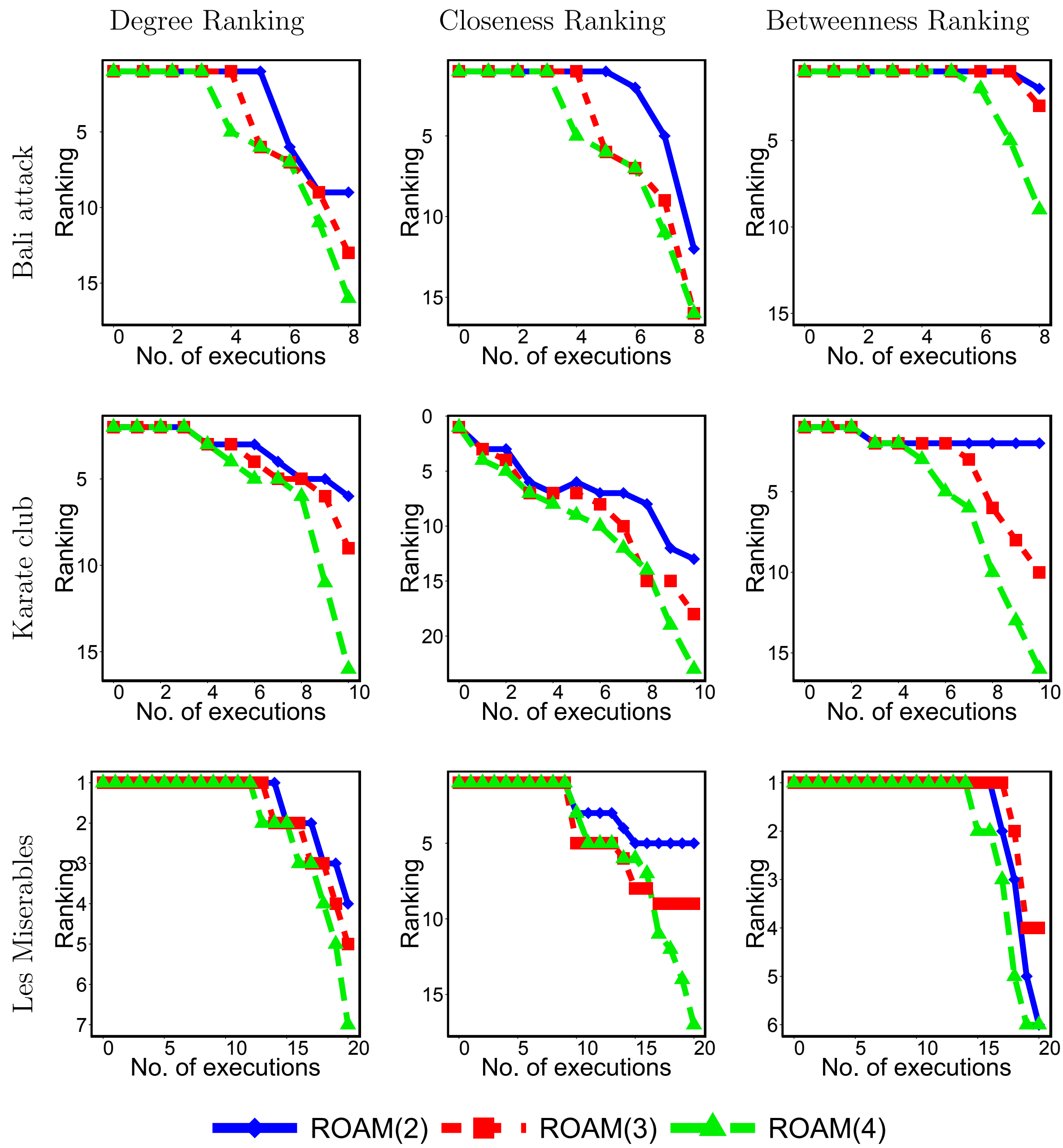

Figure A.5: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

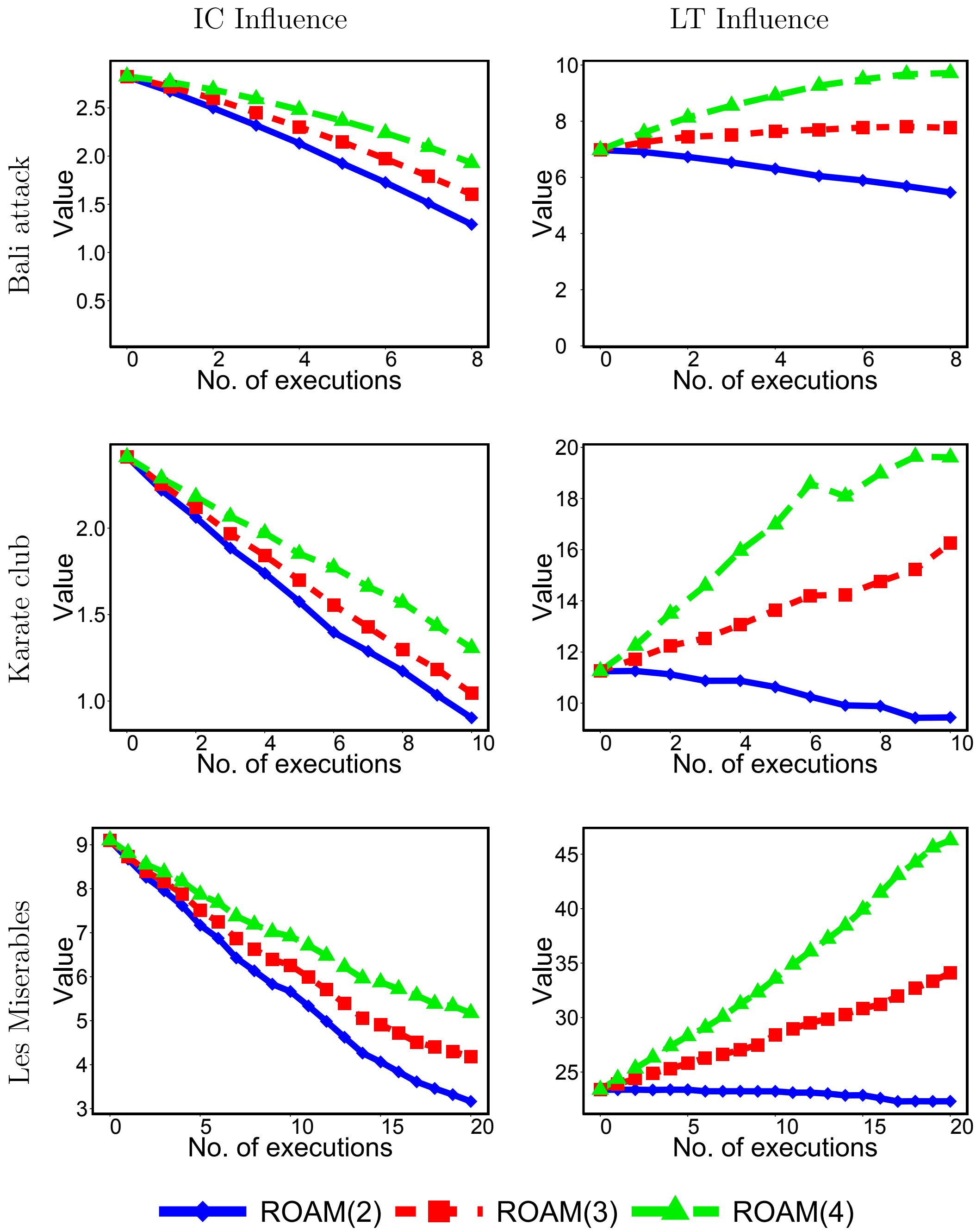

Figure A.6: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

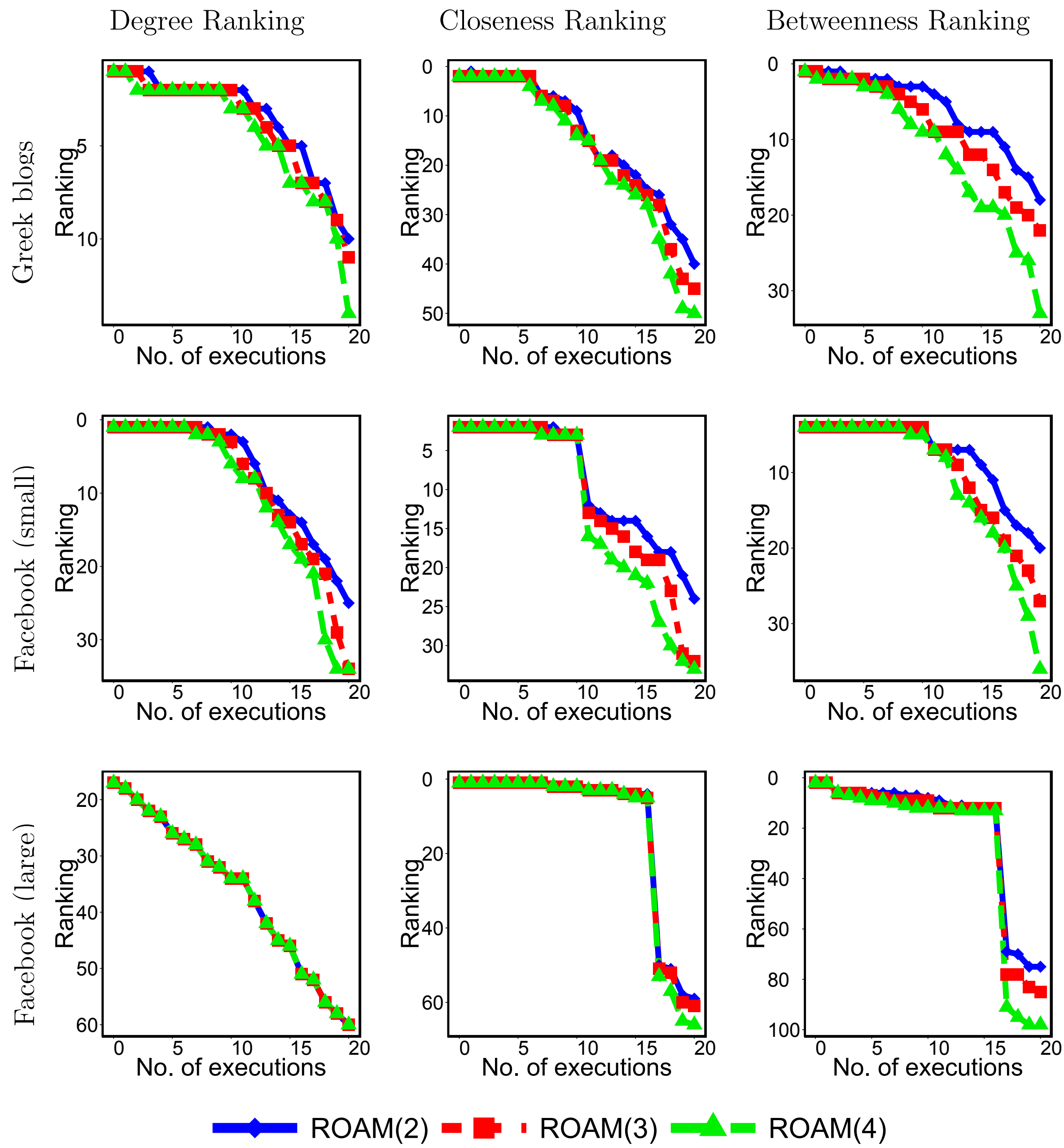

Figure A.7: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

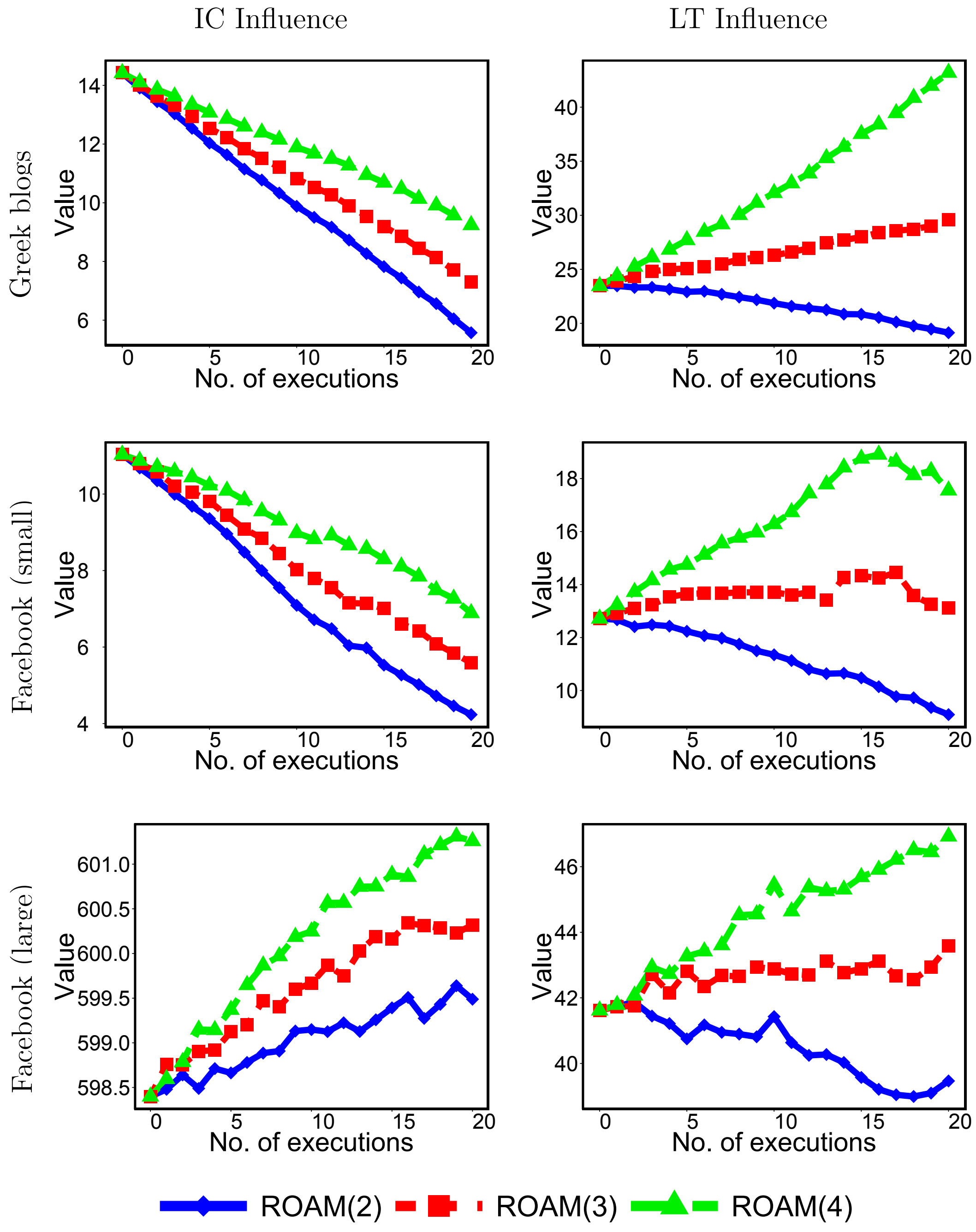

Figure A.8: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

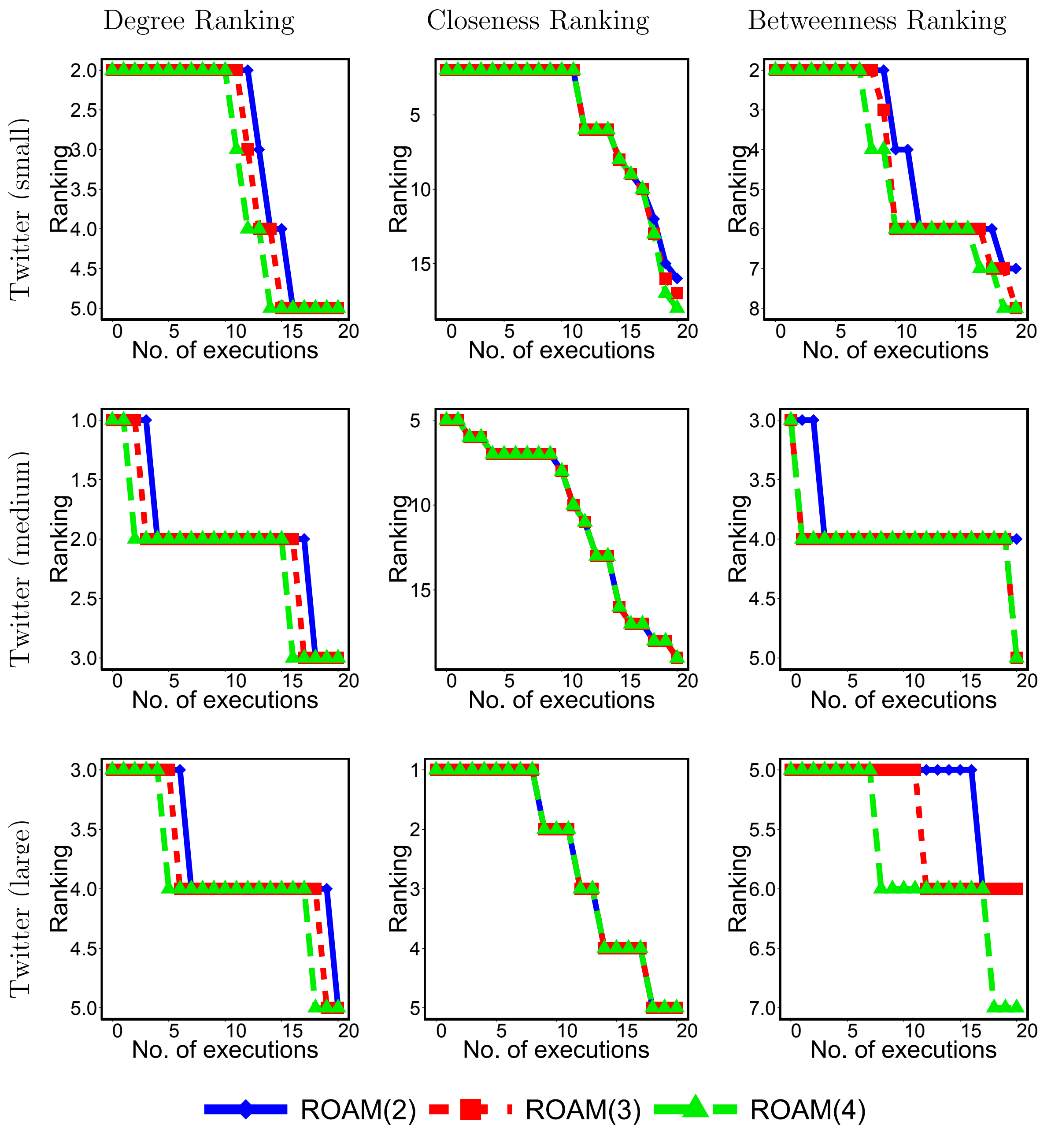

Figure A.9: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

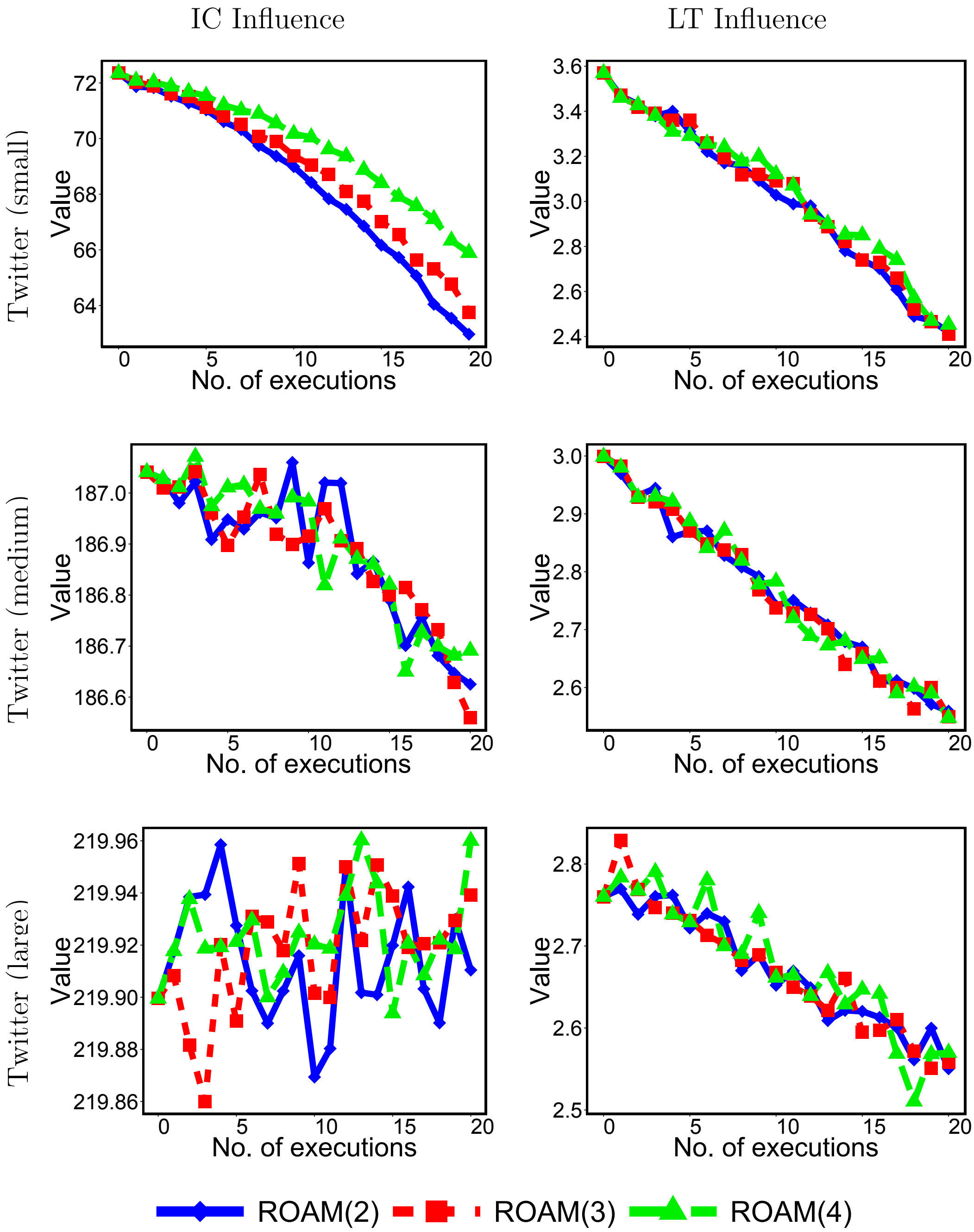

Figure A.10: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

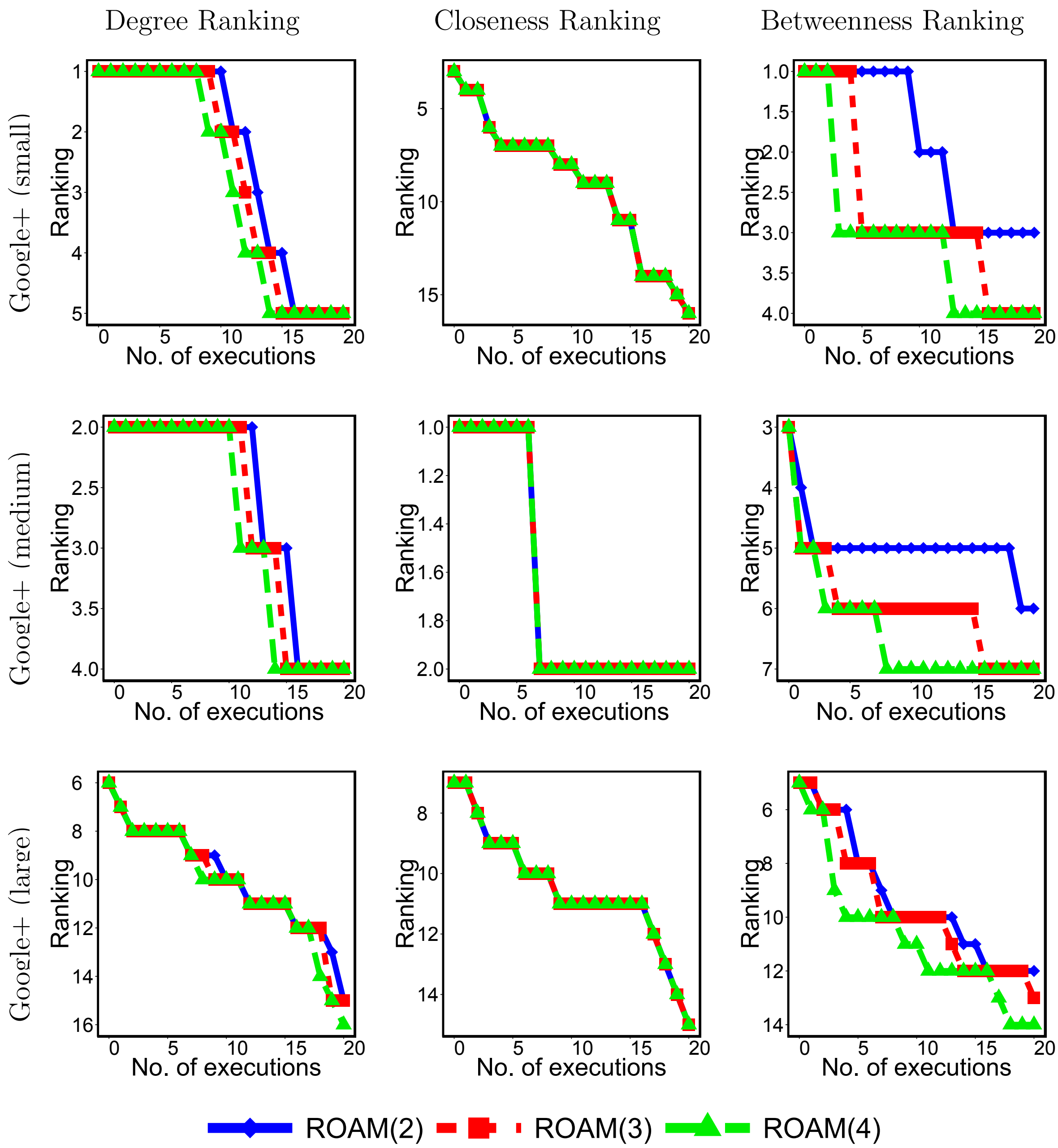

Figure A.11: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the source node's ranking (according to the centrality measures). Results are shown for ROAM(b), where b is the budget in each execution.

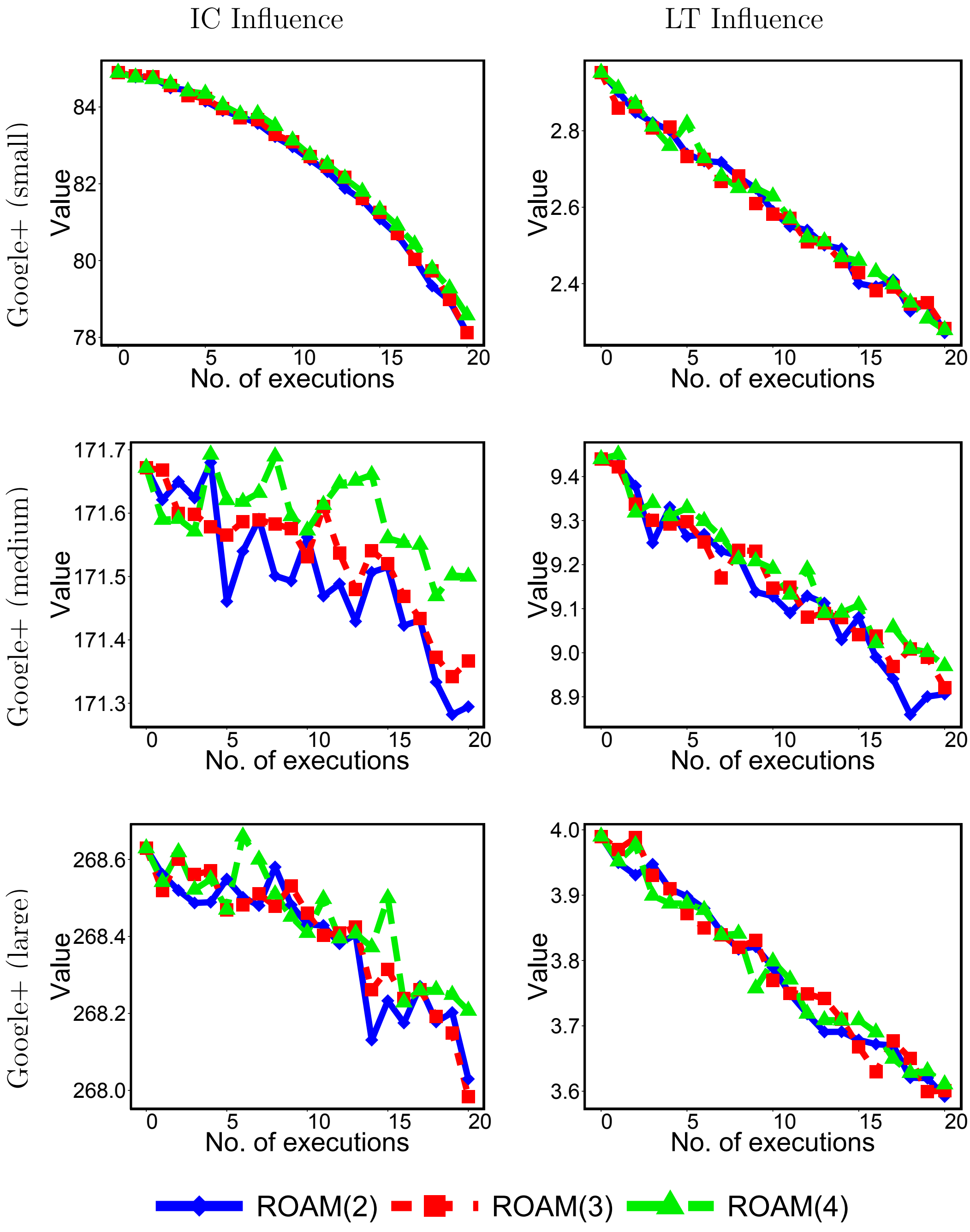

Figure A.12: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the source node's influence value (according to the influence models). Results are shown for ROAM(b), where b is the budget in each execution.

Appendix B

DICE Simulation Results

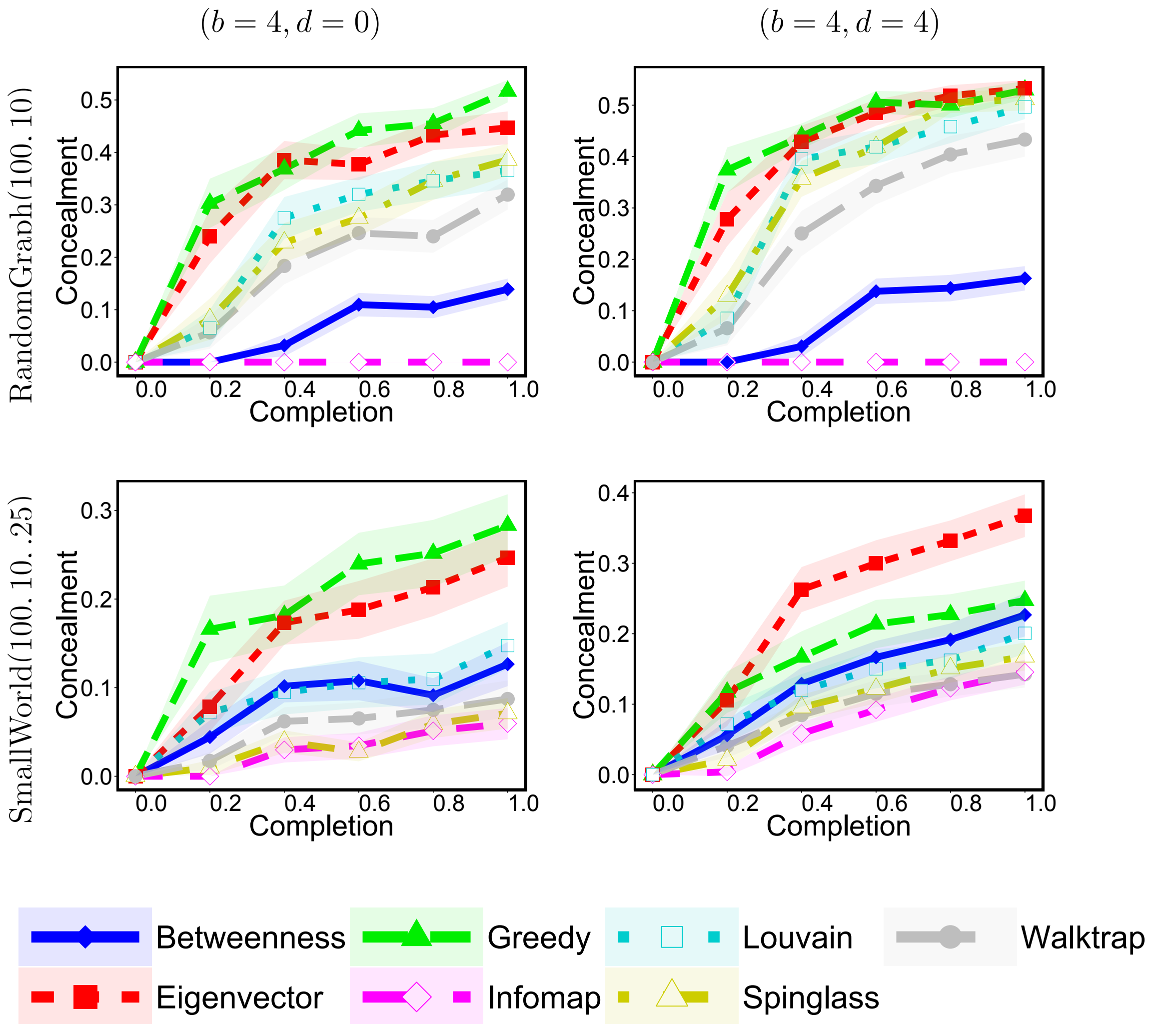

Figure B.1: Consecutive execution of DICE $\left\lceil \frac{|C^{\dagger}|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

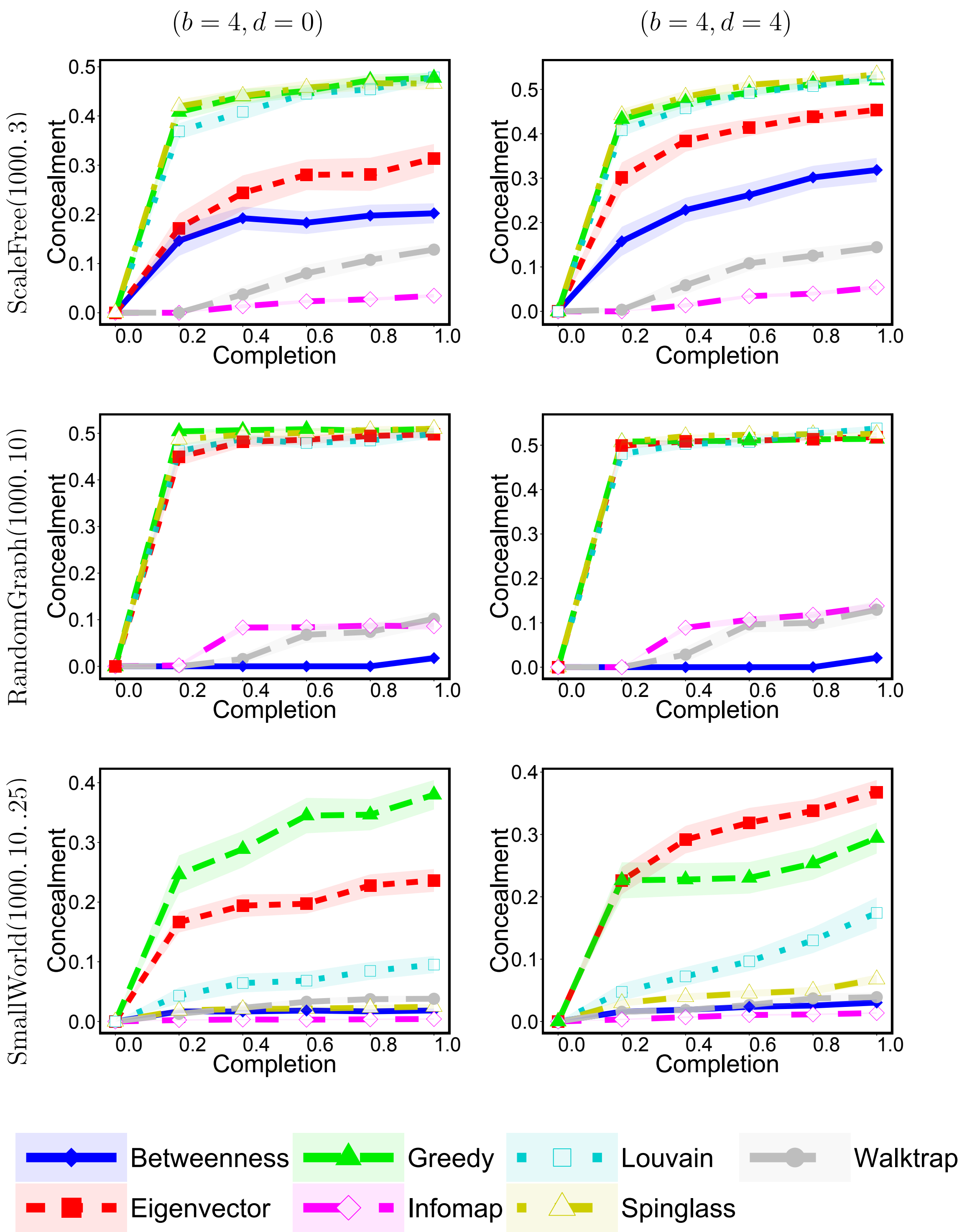

Figure B.2: Consecutive execution of DICE $\left\lceil \frac{|C^\dagger|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

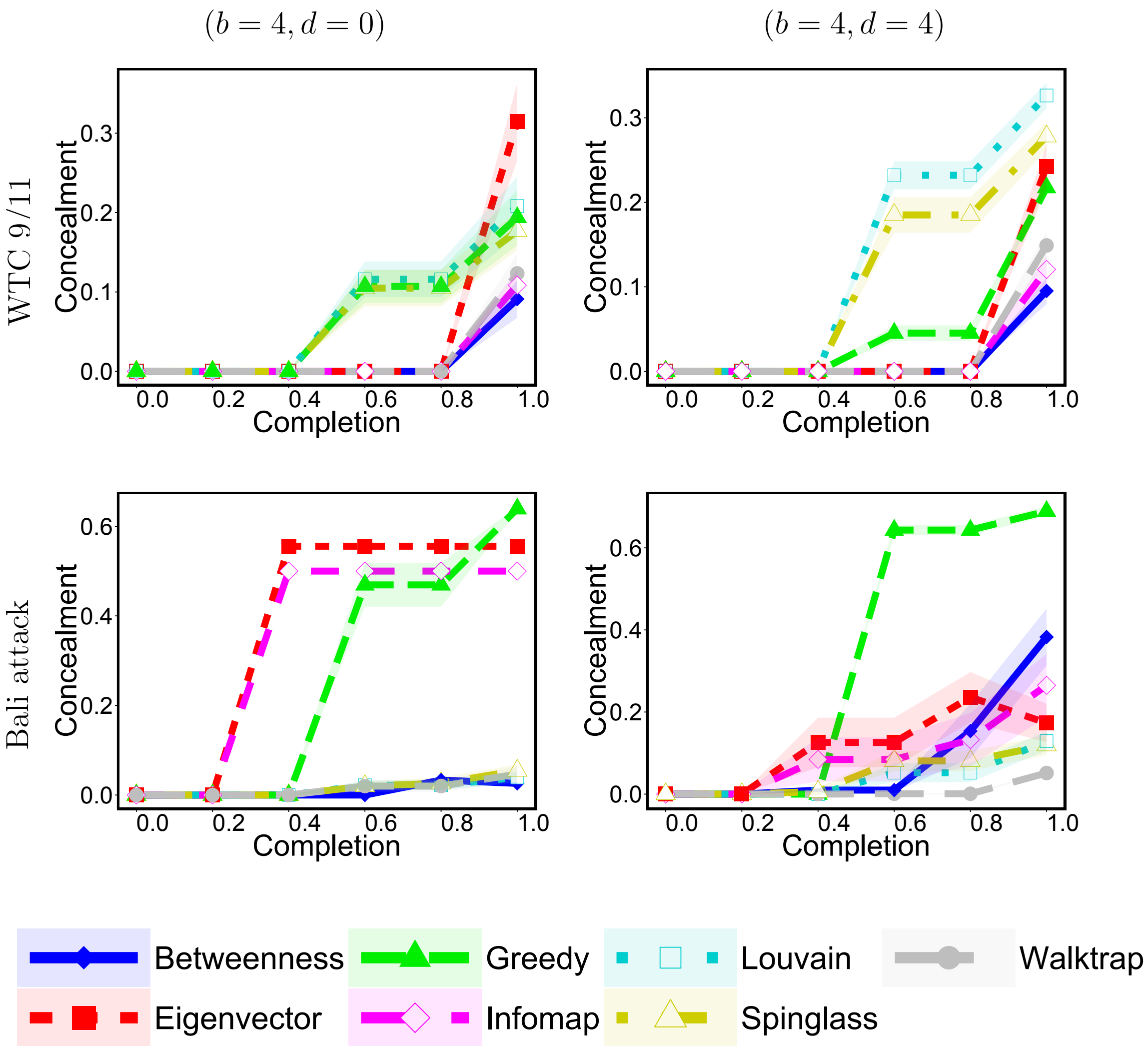

Figure B.3: Consecutive execution of DICE $\left\lceil \frac{|C^{\dagger}|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

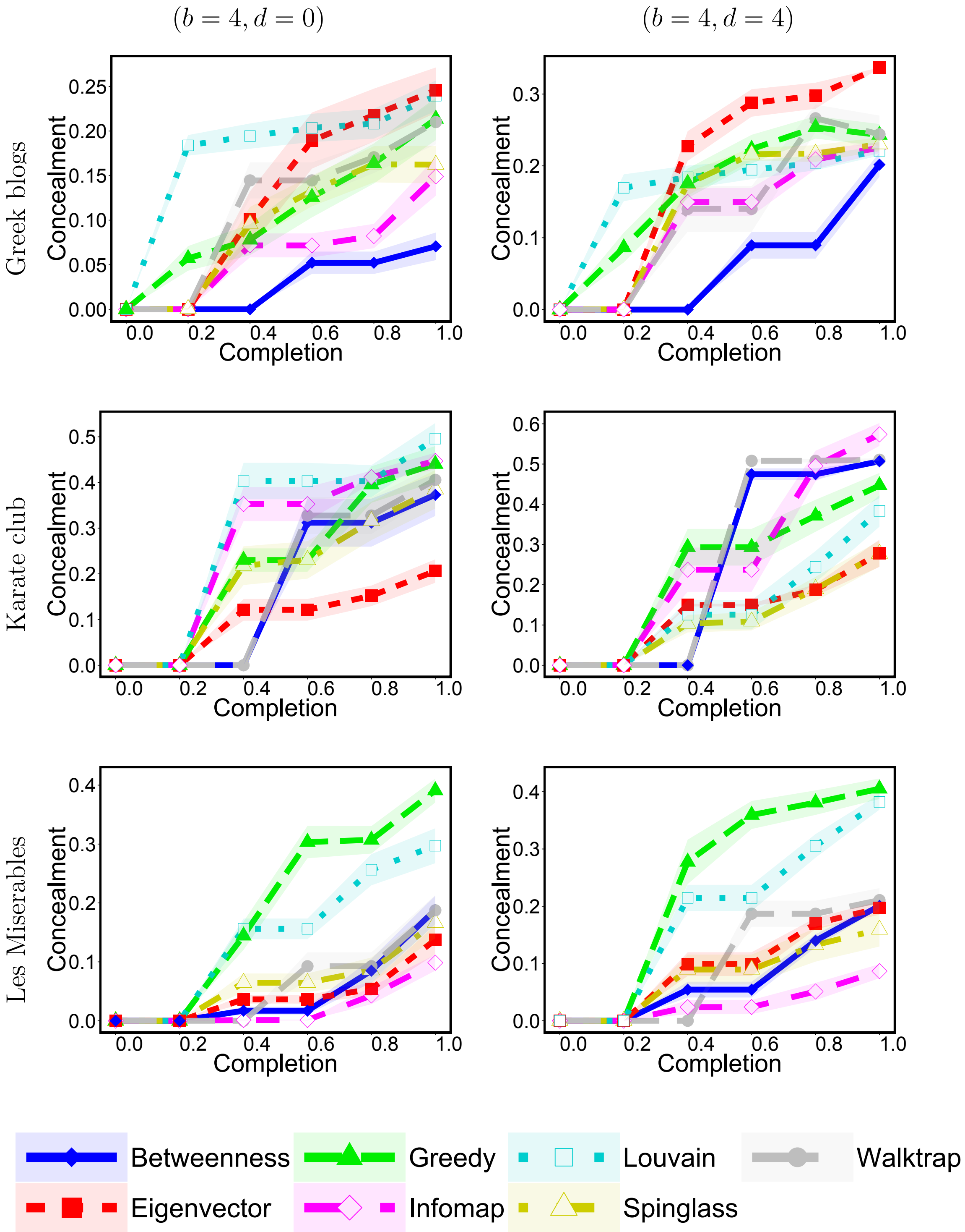

Figure B.4: Consecutive execution of DICE $\left\lceil \frac{|C^{\dagger}|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

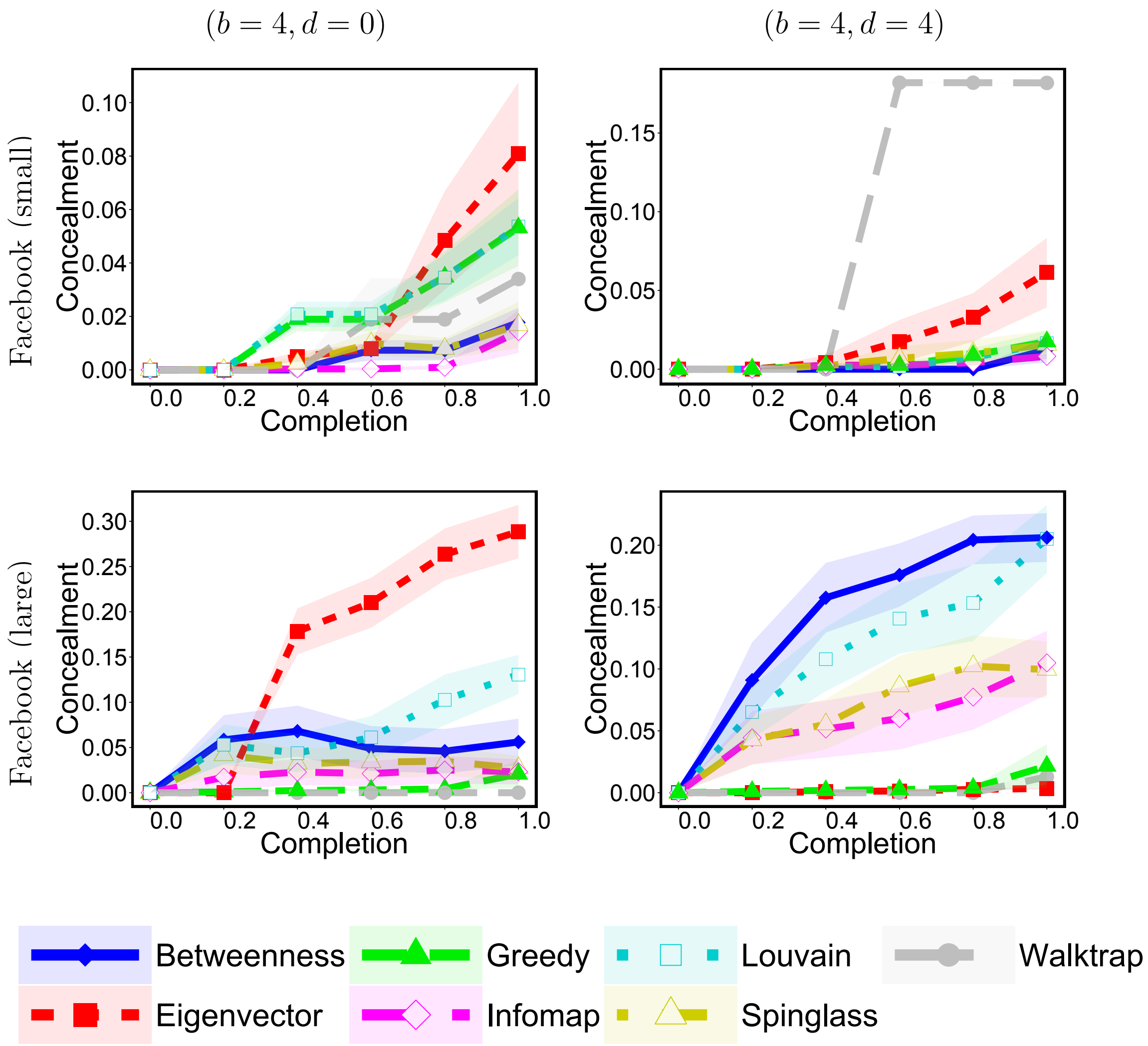

Figure B.5: Consecutive execution of DICE $\left\lceil \frac{|C^{\dagger}|}{b} \right\rceil$ times (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE (b, d), where b is the budget in each execution, and d is the number of removed internal edges.

Appendix C

Remainder of the Proof of Lemma 1

In the proof of Lemma 1 (which was presented earlier in Section 6.4), we proved that point (b) of the lemma holds, and that point (c) holds for every non-edge of the form $(v_0, d_{i,j})$. What remained was to prove the correctness of point (a), as well as the correctness of point (c) for every non-edge of the form:

(i) (v_0, v_1)

(ii) (u_i, S_j) for $u_i \notin S_j$

(iii) $(u_i, a_{j,l})$ for $i \neq j$

(iv) $(u_i, d_{j,l})$ for $i \neq j$

(v) (S_i, S_j) for $i \neq j$

(vi) $(S_i, a_{j,l})$

(vii) $(S_i, d_{j,l})$

(viii) $(a_{i_1,j_1}, a_{i_2,j_2})$

(ix) $(a_{i_1,j_1}, d_{i_2,j_2})$

(x) $(d_{i_1,j_1}, d_{i_2,j_2})$

We proved the above for only one similarity index, namely *Common Neighbours*, σ^{CN}. We will now do the same, but for each of the remaining similarity indices in $\mathbb{S}$. To this end, first recall how in the proof we showed that the following holds for every network in Γ_A and for every $a_{i,j}$, $d_{i,j}$, S_i in these network:

- $\delta(a_{i,j}) = 3$;
- $\delta(d_{i,j}) = 2$;
- $4 \leqslant \delta(S_i) \leqslant 5$.

We also showed that for any given $A \subseteq \hat{A}$, we have $S_A(u_0) = 0$. In what follows, we use those facts without referring back to them. We also use r to denote the number of $a_{i,j}$ nodes, *i.e.*, $r = c(m+1)$, and use h to denote the number of $d_{i,j}$ nodes, *i.e.*, $h = (m+1)q - 3q = mq - 2q$.

Salton similarity index ($\sigma^{\mathbf{Sal}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\text{Sal}}(u_j, v_0) = \frac{1 + |S_A(u_j)|}{\sqrt{(r+|A|)(m+q+2)}}.$$

Moving on to point (c), note that $\sigma^{\text{Sal}}(u_0, v_0) \leqslant \frac{1}{\sqrt{42}}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\text{Sal}}(v_0, v_1) = \frac{\sqrt{r+|A|}}{\sqrt{r+h+q+m+1}} > \sigma^{\text{Sal}}(u_0, v_0)$

(ii) $\sigma^{\text{Sal}}(u_i, S_j) \geqslant \frac{3}{\sqrt{(m+q+2)5}} > \sigma^{\text{Sal}}(u_0, v_0)$

(iii) $\sigma^{\text{Sal}}(u_i, a_{j,l}) \geqslant \frac{1}{\sqrt{(m+q+2)3}} > \sigma^{\text{Sal}}(u_0, v_0)$

(iv) $\sigma^{\text{Sal}}(u_i, d_{j,l}) \geqslant \frac{1}{\sqrt{(m+q+2)2}} > \sigma^{\text{Sal}}(u_0, v_0)$

(v) $\sigma^{\text{Sal}}(S_i, S_j) \geqslant \frac{1}{\sqrt{20}} > \sigma^{\text{Sal}}(u_0, v_0)$

(vi) $\sigma^{\text{Sal}}(S_i, a_{j,l}) \geqslant \frac{1}{2\sqrt{3}} > \sigma^{\text{Sal}}(u_0, v_0)$

(vii) $\sigma^{\text{Sal}}(S_i, d_{j,l}) \geqslant \frac{1}{\sqrt{10}} > \sigma^{\text{Sal}}(u_0, v_0)$

(viii) $\sigma^{\text{Sal}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{2}{3} > \sigma^{\text{Sal}}(u_0, v_0)$

(ix) $\sigma^{\text{Sal}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{\sqrt{6}} > \sigma^{\text{Sal}}(u_0, v_0)$

(x) $\sigma^{\text{Sal}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{2} > \sigma^{\text{Sal}}(u_0, v_0)$

Jaccard similarity index ($\sigma^{\mathbf{Jac}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\text{Jac}}(u_j, v_0) = \frac{1 + |S_A(u_j)|}{r+q+m+|A|+2-|S_A(u_j)|}.$$

Moving on to point (c), note that $\sigma^{\text{Jac}}(u_0, v_0) \leqslant \frac{1}{13}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\text{Jac}}(v_0, v_1) = \frac{r+|A|}{r+h+q+m+1} > \sigma^{\text{Jac}}(u_0, v_0)$

(ii) $\sigma^{\text{Jac}}(u_i, S_j) \geqslant \frac{3}{m+q+3} > \sigma^{\text{Jac}}(u_0, v_0)$

(iii) $\sigma^{\text{Jac}}(u_i, a_{j,l}) \geqslant \frac{1}{m+q+3} > \sigma^{\text{Jac}}(u_0, v_0)$

(iv) $\sigma^{\mathrm{Jac}}(u_i, d_{j,l}) \geqslant \frac{1}{m+q+2} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(v) $\sigma^{\mathrm{Jac}}(S_i, S_j) \geqslant \frac{1}{8} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(vi) $\sigma^{\mathrm{Jac}}(S_i, a_{j,l}) \geqslant \frac{1}{6} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(vii) $\sigma^{\mathrm{Jac}}(S_i, d_{j,l}) \geqslant \frac{1}{6} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(viii) $\sigma^{\mathrm{Jac}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{2}{4} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(ix) $\sigma^{\mathrm{Jac}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{4} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

(x) $\sigma^{\mathrm{Jac}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{3} > \sigma^{\mathrm{Jac}}(u_0, v_0)$

Sørensen similarity index ($\sigma^{\mathbf{Sør}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\mathrm{Sør}}(u_j, v_0) = \frac{2 + 2|S_A(u_j)|}{r + q + m + |A| + 2}.$$

Moving on to point (c), note that $\sigma^{\mathrm{Sør}}(u_0, v_0) \leqslant \frac{2}{13}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\mathrm{Sør}}(v_0, v_1) = \frac{2r+2|A|}{2r+h+q+m+1+|A|} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(ii) $\sigma^{\mathrm{Sør}}(u_i, S_j) \geqslant \frac{6}{m+q+7} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(iii) $\sigma^{\mathrm{Sør}}(u_i, a_{j,l}) \geqslant \frac{2}{m+q+5} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(iv) $\sigma^{\mathrm{Sør}}(u_i, d_{j,l}) \geqslant \frac{2}{m+q+4} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(v) $\sigma^{\mathrm{Sør}}(S_i, S_j) \geqslant \frac{2}{9} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(vi) $\sigma^{\mathrm{Sør}}(S_i, a_{j,l}) \geqslant \frac{2}{8} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(vii) $\sigma^{\mathrm{Sør}}(S_i, d_{j,l}) \geqslant \frac{2}{7} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(viii) $\sigma^{\mathrm{Sør}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{4}{6} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(ix) $\sigma^{\mathrm{Sør}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{2}{5} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

(x) $\sigma^{\mathrm{Sør}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{2}{4} > \sigma^{\mathrm{Sør}}(u_0, v_0)$

Hub Promoted similarity index ($\sigma^{\mathbf{HPI}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\mathrm{HPI}}(u_j, v_0) = \frac{1 + |S_A(u_j)|}{r + |A|}.$$

Moving on to point (c), note that $\sigma^{\mathrm{HPI}}(u_0, v_0) \leqslant \frac{1}{6}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\text{HPI}}(v_0, v_1) = \frac{r+|A|}{r+|A|} = 1 > \sigma^{\text{HPI}}(u_0, v_0)$

(ii) $\sigma^{\text{HPI}}(u_i, S_j) \geqslant \frac{3}{5} > \sigma^{\text{HPI}}(u_0, v_0)$

(iii) $\sigma^{\text{HPI}}(u_i, a_{j,l}) \geqslant \frac{1}{3} > \sigma^{\text{HPI}}(u_0, v_0)$

(iv) $\sigma^{\text{HPI}}(u_i, d_{j,l}) \geqslant \frac{1}{2} > \sigma^{\text{HPI}}(u_0, v_0)$

(v) $\sigma^{\text{HPI}}(S_i, S_j) \geqslant \frac{1}{4} > \sigma^{\text{HPI}}(u_0, v_0)$

(vi) $\sigma^{\text{HPI}}(S_i, a_{j,l}) \geqslant \frac{1}{3} > \sigma^{\text{HPI}}(u_0, v_0)$

(vii) $\sigma^{\text{HPI}}(S_i, d_{j,l}) \geqslant \frac{1}{2} > \sigma^{\text{HPI}}(u_0, v_0)$

(viii) $\sigma^{\text{HPI}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{2}{3} > \sigma^{\text{HPI}}(u_0, v_0)$

(ix) $\sigma^{\text{HPI}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{2} > \sigma^{\text{HPI}}(u_0, v_0)$

(x) $\sigma^{\text{HPI}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{2} > \sigma^{\text{HPI}}(u_0, v_0)$

Hub Depressed similarity index ($\sigma^{\textbf{HDI}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\text{HDI}}(u_j, v_0) = \frac{1 + |S_A(u_j)|}{m + q + 2}.$$

Moving on to point (c), note that $\sigma^{\text{HDI}}(u_0, v_0) \leqslant \frac{1}{7}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\text{HDI}}(v_0, v_1) = \frac{r+|A|}{r+h+q+m+1} > \sigma^{\text{HDI}}(u_0, v_0)$

(ii) $\sigma^{\text{HDI}}(u_i, S_j) \geqslant \frac{3}{m+q+2} > \sigma^{\text{HDI}}(u_0, v_0)$

(iii) Either $\sigma^{\text{HDI}}(u_i, a_{j,l}) = \frac{1}{m+q+2} = \sigma^{\text{HDI}}(u_0, v_0)$ (if $i = j$) or $\sigma^{\text{HDI}}(u_i, a_{j,l}) = \frac{2}{m+q+2} > \sigma^{\text{HDI}}(u_0, v_0)$ (otherwise)

(iv) Either $\sigma^{\text{HDI}}(u_i, d_{j,l}) = \frac{1}{m+q+2} = \sigma^{\text{HDI}}(u_0, v_0)$ (if $i = j$) or $\sigma^{\text{HDI}}(u_i, d_{j,l}) = \frac{2}{m+q+2} > \sigma^{\text{HDI}}(u_0, v_0)$ (otherwise)

(v) $\sigma^{\text{HDI}}(S_i, S_j) \geqslant \frac{1}{5} > \sigma^{\text{HDI}}(u_0, v_0)$

(vi) $\sigma^{\text{HDI}}(S_i, a_{j,l}) \geqslant \frac{1}{5} > \sigma^{\text{HDI}}(u_0, v_0)$

(vii) $\sigma^{\text{HDI}}(S_i, d_{j,l}) \geqslant \frac{1}{5} > \sigma^{\text{HDI}}(u_0, v_0)$

(viii) $\sigma^{\text{HDI}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{2}{3} > \sigma^{\text{HDI}}(u_0, v_0)$

(ix) $\sigma^{\text{HDI}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{3} > \sigma^{\text{HDI}}(u_0, v_0)$

(x) $\sigma^{\text{HDI}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{2} > \sigma^{\text{HDI}}(u_0, v_0)$

For Leicht-Holme-Newman similarity index ($\sigma^{\mathbf{LHN}}$): We choose $c = 1$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\mathrm{LHN}}(u_j, v_0) = \frac{1 + |S_A(u_j)|}{(r + |A|)(m + q + 2)}.$$

Moving on to point (c), note that $\sigma^{\mathrm{LHN}}(u_0, v_0) \leqslant \frac{1}{42}$ (since $m \geqslant 5$), and that the following holds for every network in Γ_A:

(i) $\sigma^{\mathrm{LHN}}(v_0, v_1) = \frac{1}{r+h+q+m+1} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(ii) $\sigma^{\mathrm{LHN}}(u_i, S_j) \geqslant \frac{3}{(m+q+2)5} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(iii) $\sigma^{\mathrm{LHN}}(u_i, a_{j,l}) \geqslant \frac{1}{(m+q+2)3} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(iv) $\sigma^{\mathrm{LHN}}(u_i, d_{j,l}) \geqslant \frac{1}{(m+q+2)2} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(v) $\sigma^{\mathrm{LHN}}(S_i, S_j) \geqslant \frac{1}{20} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(vi) $\sigma^{\mathrm{LHN}}(S_i, a_{j,l}) \geqslant \frac{1}{12} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(vii) $\sigma^{\mathrm{LHN}}(S_i, d_{j,l}) \geqslant \frac{1}{10} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(viii) $\sigma^{\mathrm{LHN}}(a_{i_1,j_1}, a_{i_2,j_2}) \geqslant \frac{2}{9} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(ix) $\sigma^{\mathrm{LHN}}(a_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{6} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

(x) $\sigma^{\mathrm{LHN}}(d_{i_1,j_1}, d_{i_2,j_2}) \geqslant \frac{1}{4} > \sigma^{\mathrm{LHN}}(u_0, v_0)$

For Adamic-Adar similarity index ($\sigma^{\mathbf{AA}}$): We choose $c = 3$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\mathrm{AA}}(u_j, v_0) = \frac{3}{\log(3)} + \frac{|S_A(u_j)|}{\log(5)}.$$

Moving on to point (c), note that $\sigma^{\mathrm{AA}}(u_0, v_0) = \frac{3}{\log(3)} > 6$ and that the following holds for every network in Γ_A:

(i) $\sigma^{\mathrm{AA}}(v_0, v_1) = \frac{r}{\log(3)} + \frac{|A|}{\log(5)} > \sigma^{\mathrm{AA}}(u_0, v_0)$

(ii) $\sigma^{\mathrm{AA}}(u_i, S_j) = \frac{1}{\log(r+h+q+m+1)} + \frac{3}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(iii) $\sigma^{\mathrm{AA}}(u_i, a_{j,l}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(iv) $\sigma^{\mathrm{AA}}(u_i, d_{j,l}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(v) $\sigma^{\mathrm{AA}}(S_i, S_j) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(r+|A|)} + \frac{3}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(vi) $\sigma^{\mathrm{AA}}(S_i, a_{j,l}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(r+|A|)} + \frac{1}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(vii) $\sigma^{\mathrm{AA}}(S_i, d_{j,l}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(q+m+4)} < \sigma^{\mathrm{AA}}(u_0, v_0)$

(viii) $\sigma^{\text{AA}}(a_{i_1,j_1}, a_{i_2,j_2}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(r+|A|)} + \frac{1}{\log(q+m+4)} < \sigma^{\text{AA}}(u_0, v_0)$

(ix) $\sigma^{\text{AA}}(a_{i_1,j_1}, d_{i_2,j_2}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(q+m+4)} < \sigma^{\text{AA}}(u_0, v_0)$

(x) $\sigma^{\text{AA}}(d_{i_1,j_1}, d_{i_2,j_2}) \leqslant \frac{1}{\log(r+h+q+m+1)} + \frac{1}{\log(q+m+4)} < \sigma^{\text{AA}}(u_0, v_0)$

For Resource Allocation similarity index ($\sigma^{\textbf{RA}}$): We choose $c = 3$. Then, to prove the correctness of point (a), it suffices to note that for every network in Γ_A we have:

$$\forall_{u_j \in \{u_0,\dots,u_m\}} \sigma^{\text{RA}}(u_j, v_0) = \frac{3}{3} + \frac{|S_A(u_j)|}{5}.$$

Moving on to point (c), note that $\sigma^{\text{RA}}(u_0, v_0) = 1$ and that the following holds for every network in Γ_A:

(i) $\sigma^{\text{RA}}(v_0, v_1) = \frac{r}{3} + \frac{|A|}{5} > \sigma^{\text{RA}}(u_0, v_0)$

(ii) $\sigma^{\text{RA}}(u_i, S_j) = \frac{1}{r+h+q+m+1} + \frac{3}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(iii) $\sigma^{\text{RA}}(u_i, a_{j,l}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(iv) $\sigma^{\text{RA}}(u_i, d_{j,l}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(v) $\sigma^{\text{RA}}(S_i, S_j) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{r+|A|} + \frac{3}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(vi) $\sigma^{\text{RA}}(S_i, a_{j,l}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{r+|A|} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(vii) $\sigma^{\text{RA}}(S_i, d_{j,l}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(viii) $\sigma^{\text{RA}}(a_{i_1,j_1}, a_{i_2,j_2}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{r+|A|} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(ix) $\sigma^{\text{RA}}(a_{i_1,j_1}, d_{i_2,j_2}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$

(x) $\sigma^{\text{RA}}(d_{i_1,j_1}, d_{i_2,j_2}) \leqslant \frac{1}{r+h+q+m+1} + \frac{1}{q+m+4} < \sigma^{\text{RA}}(u_0, v_0)$ □

Appendix D

An Efficient Implementation of the OTC algorithm

We now present a more efficient implementation of the OTC algorithm (see Algorithm 6). The complexity of this implementation is $\mathcal{O}(|H||V|^2 + b|H||V|)$. The $|H||V|^2$ term comes from computing an initial score $\theta_{(v,w)}$ for each non-edge $(v, w) \in A'$ (this can be done in time linear in $|V|$ for each of the $|H||V|$ non-edges). The $b|H||V|$ term comes from searching for a non-edge in $\hat{A}$ with the maximal score. Finally, updating the scores after adding each of the b edges can be done in time linear in $|V|$.

Notice that when $|H| = \omega(\log(|V|))$ and $b = \omega(|V|)$ an implementation utilizing a *priority queue* (*e.g.*, a heap [34]) is faster. The complexity of such an implementation is $\mathcal{O}(|H||V|^2 + b|V|\log(|V|))$. In more detail, a priority queue can be built in time $\mathcal{O}(|H||V|^2)$. The cost of all operations of extracting an element with maximal score is then $\mathcal{O}(b\log(|H||V|))$, which equals $\mathcal{O}(b\log(|V|))$, since $|H|$ is at most $\Theta(|V|^2)$. However, the cost of updating the scores is now $\mathcal{O}(b|V|\log(|V|))$, since it could involve either increasing or decreasing the scores (decreasing scores is realized by removing an element and adding it with a lower score).

Algorithm 6 The *Open-Triad-Creation* (OTC) algorithm

Input: A network (V, E), a budget $b \in \mathbb{N}$, a set of edges that can be added $\hat{A} \subseteq \bar{E}$, and a set of edges to be hidden $H \subset E$.
Output: Updated network (V, E).

$A' \leftarrow \Big\{(v, w) \in \hat{A} : \Big(\exists u \in N(v) : (u, v) \in H\Big) \vee \Big(\exists u \in N(w) : (u, w) \in H\Big)\Big\}$
for $(v, w) \in A'$ **do**
  if $\exists_{u \in V}\Big((v, u) \in E \setminus H \wedge (w, u) \in H\Big) \vee \Big((w, u) \in E \setminus H \wedge (v, u) \in H\Big)$ **then**
    $\theta_{(v,w)} \leftarrow -\infty$
  else
    $\theta_{(v,w)} \leftarrow \Big|\Big(N_{(V,E \setminus H)}(v) \cup N_{(V,E \setminus H)}(w)\Big) \setminus N_{(V,E \setminus H)}(v, w)\Big|$
for $i = 1, \ldots, b$ **do**
  $(v^*, w^*) \leftarrow \arg\max_{(v,w) \in A'} \theta_{(v,w)}$
  if $\theta_{(v^*,w^*)} > -\infty$ **then**
    $E \leftarrow E \cup (v^*, w^*)$
    for $u \in V \setminus \{v^*, w^*\}$ **do**
      if $(u, v^*) \in A'$ **then**
        if $(u, w^*) \in H$ **then** $\theta_{(u,v^*)} \leftarrow -\infty$
        if $(u, w^*) \in E \setminus H \wedge \theta_{(u,v^*)} > -\infty$ **then** $\theta_{(u,v^*)} \leftarrow \theta_{(u,v^*)} - 1$
        if $(u, w^*) \in \bar{E} \wedge \theta_{(u,v^*)} > -\infty$ **then** $\theta_{(u,v^*)} \leftarrow \theta_{(u,v^*)} + 1$
      if $(u, w^*) \in A'$ **then**
        if $(u, v^*) \in H$ **then**
          $\theta_{(u,w^*)} \leftarrow -\infty$
        if $(u, v^*) \in E \setminus H \wedge \theta_{(u,w^*)} > -\infty$ **then**
          $\theta_{(u,w^*)} \leftarrow \theta_{(u,w^*)} - 1$
        if $(u, v^*) \in \bar{E} \wedge \theta_{(u,w^*)} > -\infty$ **then**
          $\theta_{(u,w^*)} \leftarrow \theta_{(u,w^*)} + 1$

Appendix E

Priority Queue Implementation of the CTR algorithm

We now show an alternative implementation of the CTR algorithm (see Algorithm 7), that utilizes a priority queue (*e.g.*, a heap [34]). The complexity of such an implementation is $\mathcal{O}(|H||V| + b|V|\log(|V|))$. In more detail, a priority queue can be built in time $\mathcal{O}(|H||V|)$. The cost of all operations of extracting an element with maximal score is then $\mathcal{O}(b\log(|H||V|))$, which equals $\mathcal{O}(b\log(|V|))$, since $|H|$ is at most $\Theta(|V|^2)$. However, the cost of updating the scores becomes $\mathcal{O}(b|V|\log(|V|))$, since it can only involve decreasing the scores (decreasing scores is realized by removing an element and adding it with a lower score).

Algorithm 7 The *Open-Triad-Creation* (OTC) algorithm

Input: A network (V, E), a budget $b \in \mathbb{N}$, a set of edges that can be removed $\hat{R} \subseteq E$, and a set of edges to be hidden $H \subset E$.
Output: Updated network (V, E).

$R' \leftarrow \Big\{(v, w) \in \hat{R} : \Big(\exists u \in N(v) : (u, v) \in H\Big) \vee \Big(\exists u \in N(w) : (u, w) \in H\Big)\Big\}$
for $(x, y) \in R'$ **do**
  $\theta_{(x,y)} \leftarrow 0$
for $(v, w) \in H$ **do**
  for $u \in N(v, w)$ **do**
    if $(v, u) \in E \setminus H \wedge (w, u) \in E \setminus H$ **then**
      if $(v, u) \in R'$ **then** $\theta_{(v,u)} \leftarrow \theta_{(v,u)} + 1$
      if $(w, u) \in R'$ **then** $\theta_{(w,u)} \leftarrow \theta_{(w,u)} + 1$
for $i = 1, \ldots, b$ **do**
  $(v^*, w^*) \leftarrow \arg\max_{(v,w) \in R'} \theta_{(v,w)}$
  if $\theta_{(v^*,w^*)} > 0$ **then**
    $E = E \setminus (v^*, w^*)$
    for $u \in N(v^*) \cup N(w^*)$ **do**
      if $(v^*, u) \in H \wedge (w^*, u) \in R'$ **then** $\theta_{(w^*,u)} \leftarrow \theta_{(w^*,u)} - 1$
      if $(w^*, u) \in H \wedge (v^*, u) \in R'$ **then** $\theta_{(v^*,u)} \leftarrow \theta_{(v^*,u)} - 1$

Appendix F

OTC and CTR Simulation Results

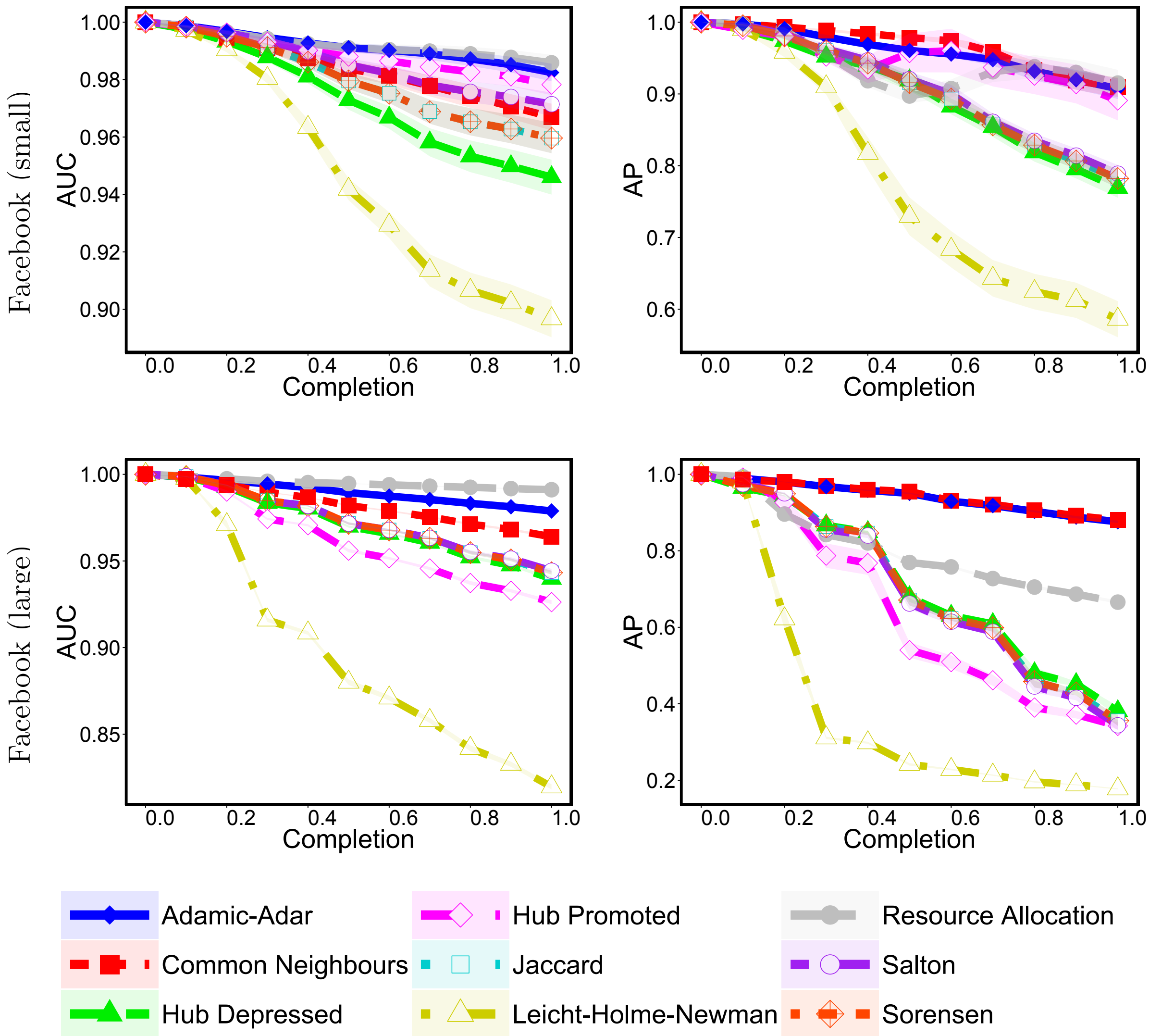

Figure F.1: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

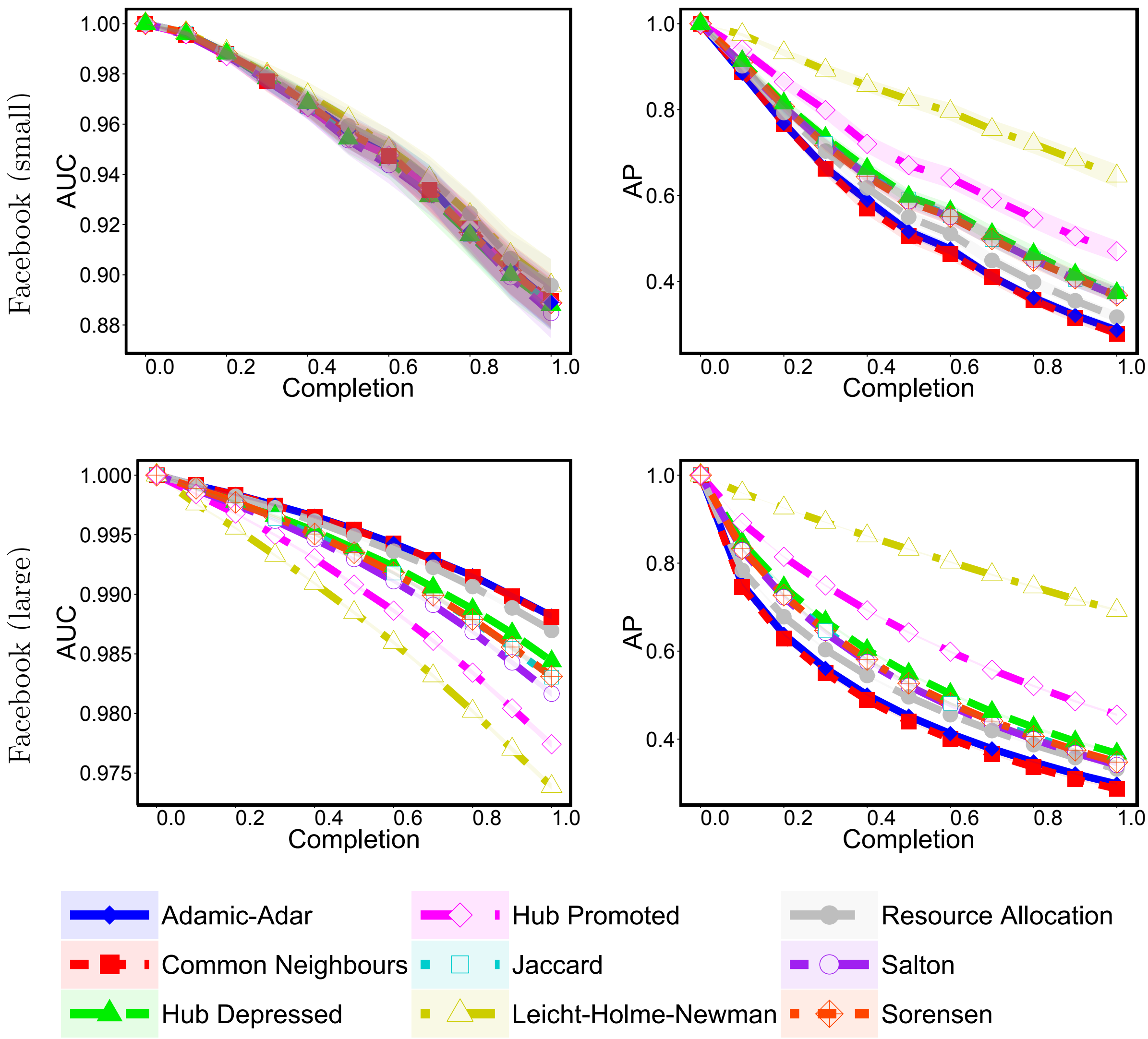

Figure F.2: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

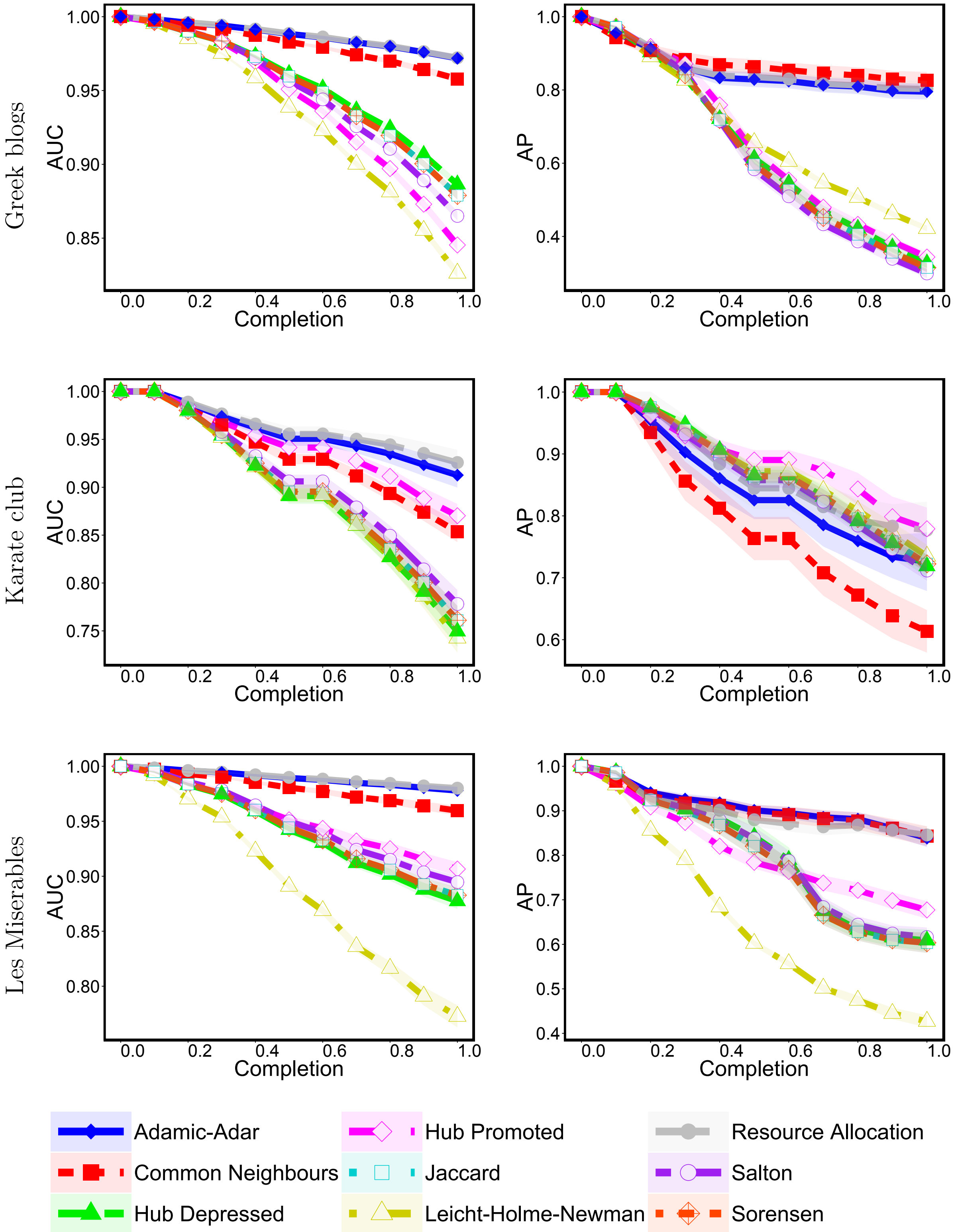

Figure F.3: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

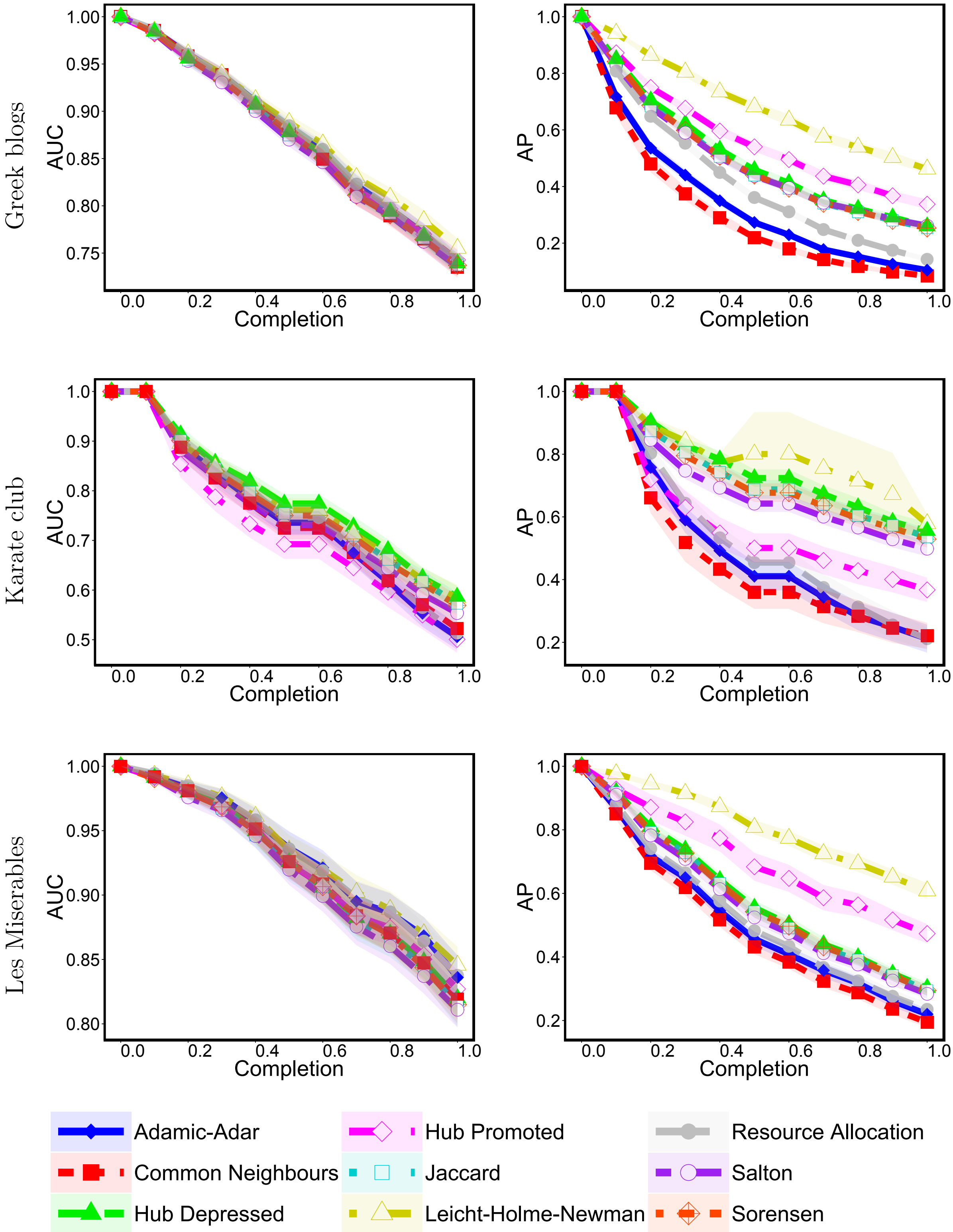

Figure F.4: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

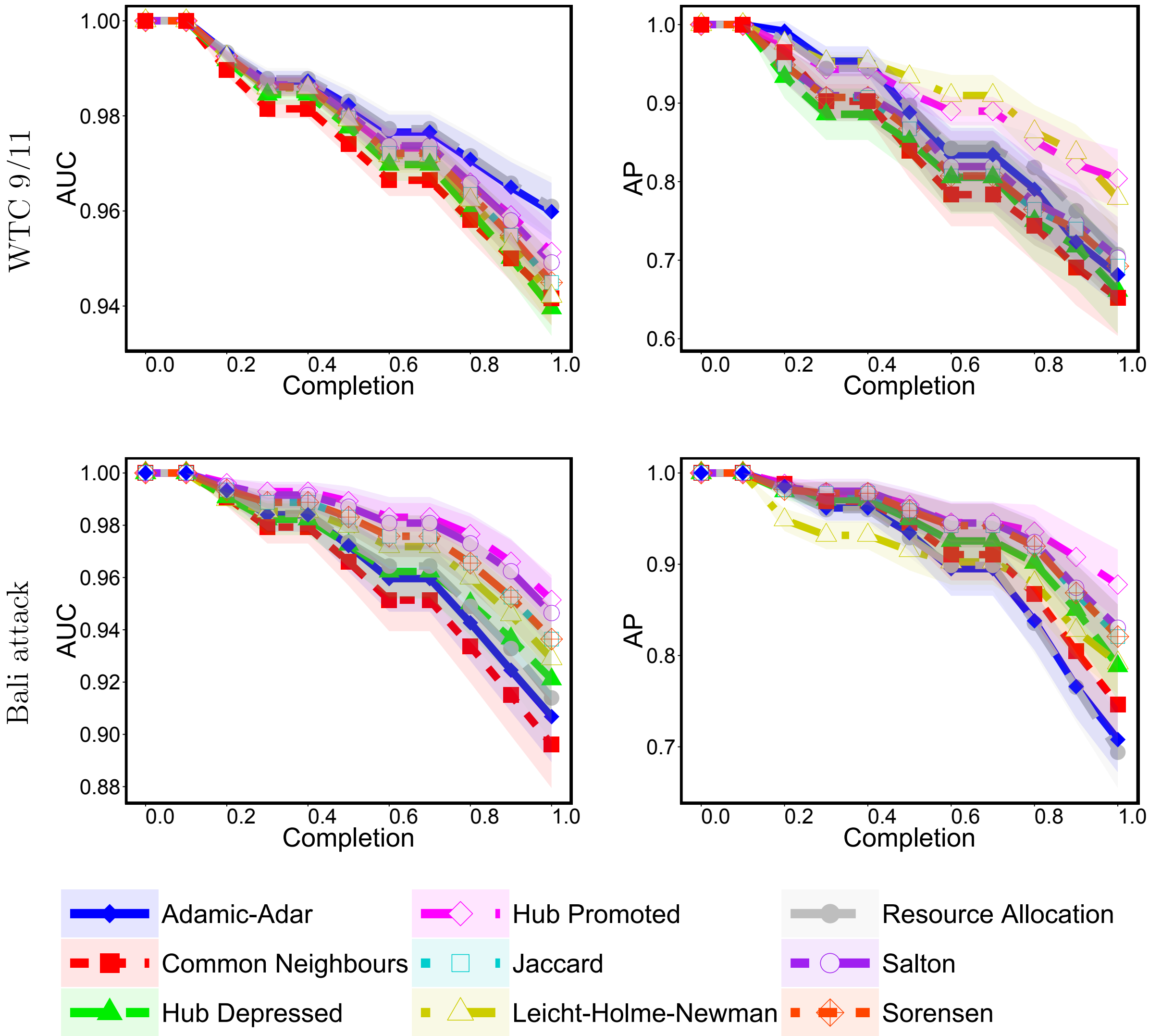

Figure F.5: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

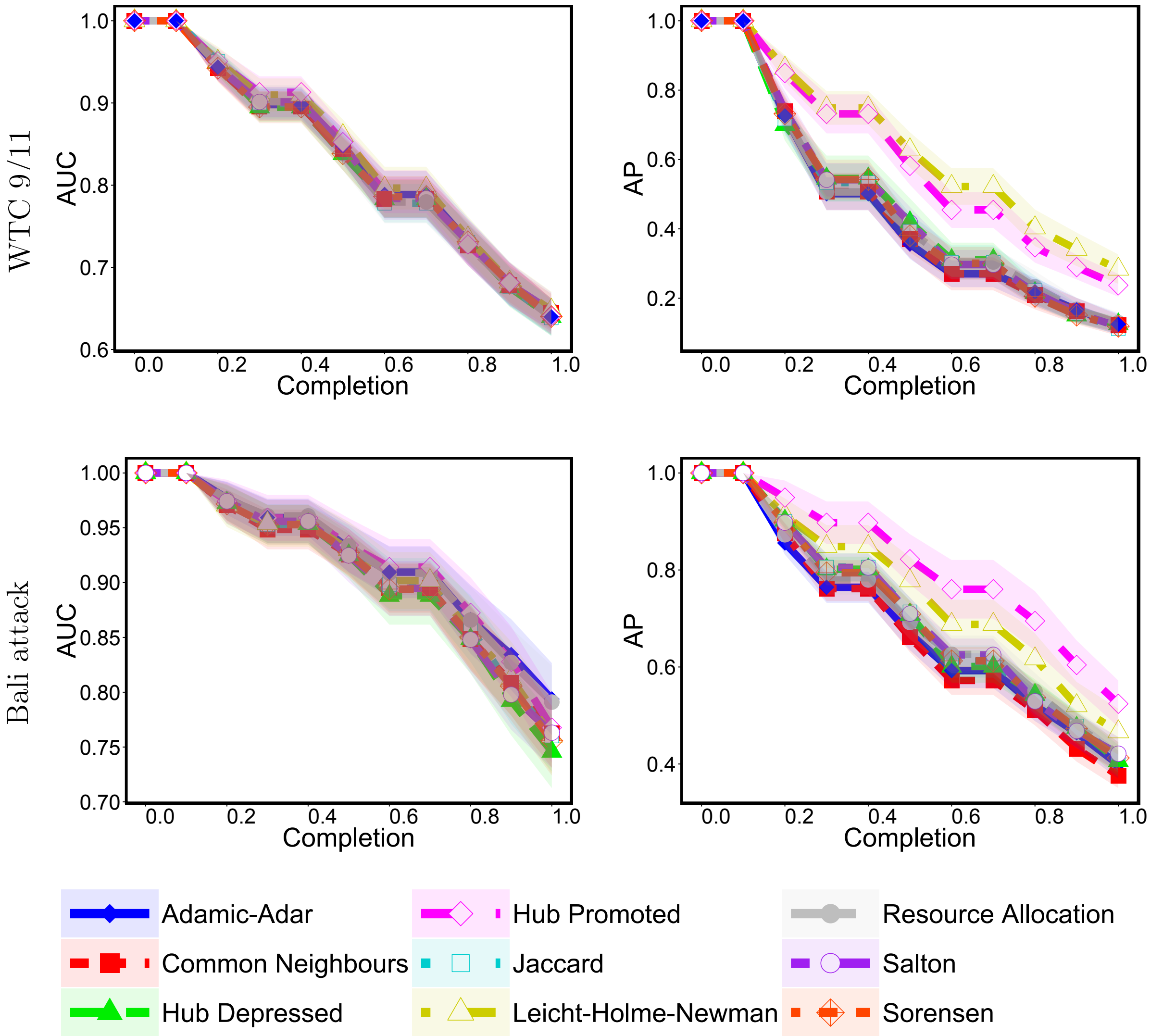

Figure F.6: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

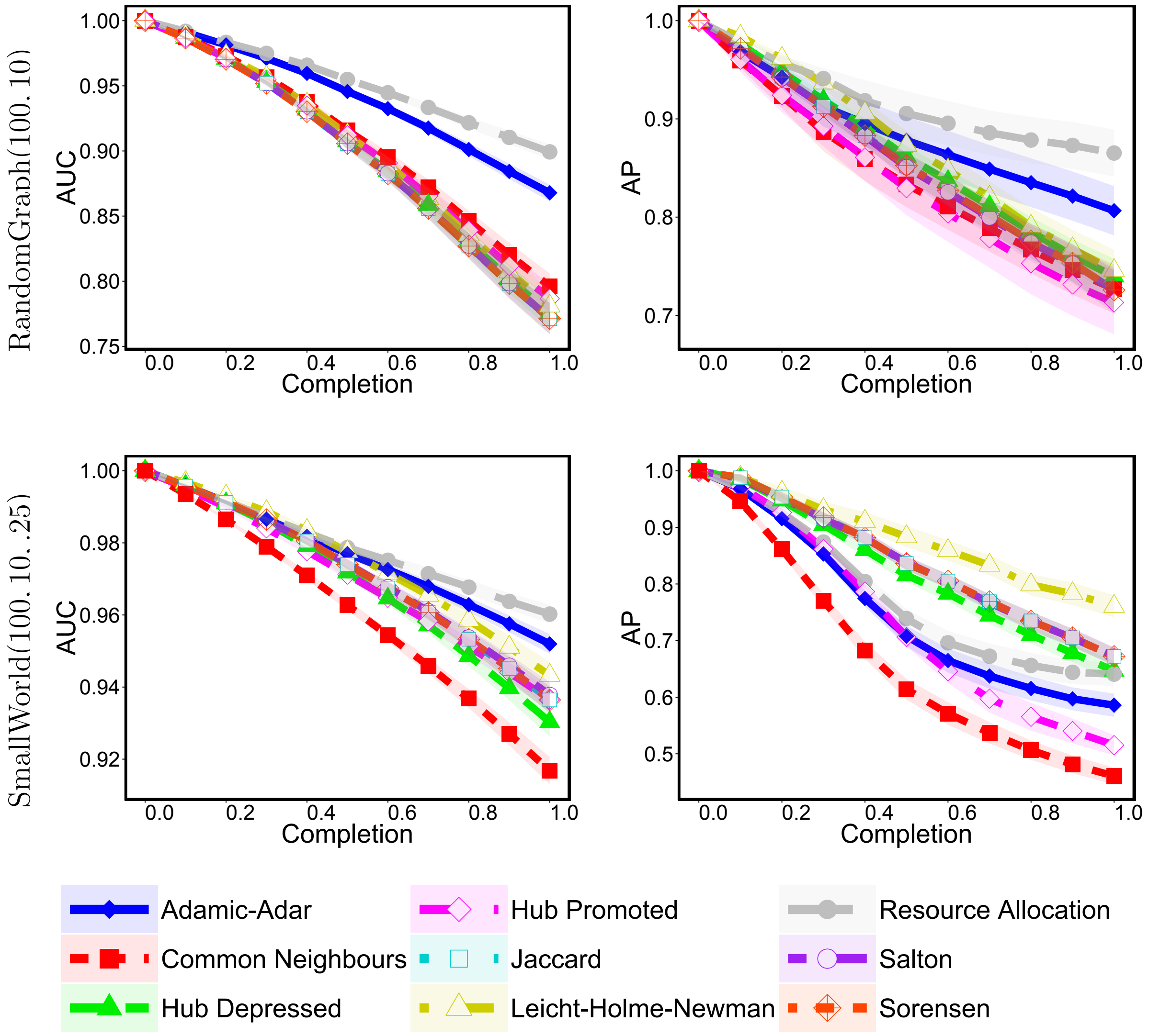

Figure F.7: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

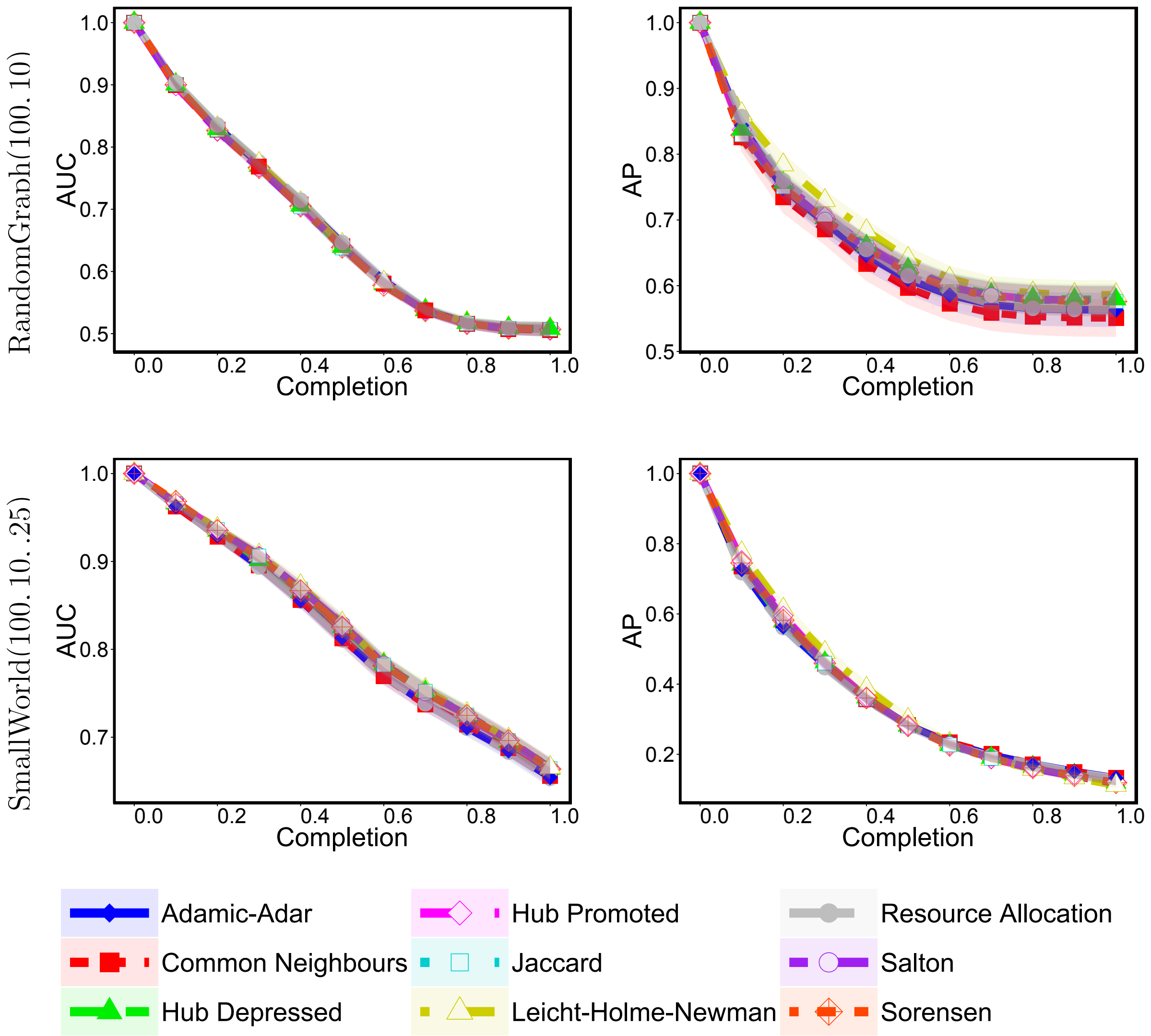

Figure F.8: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

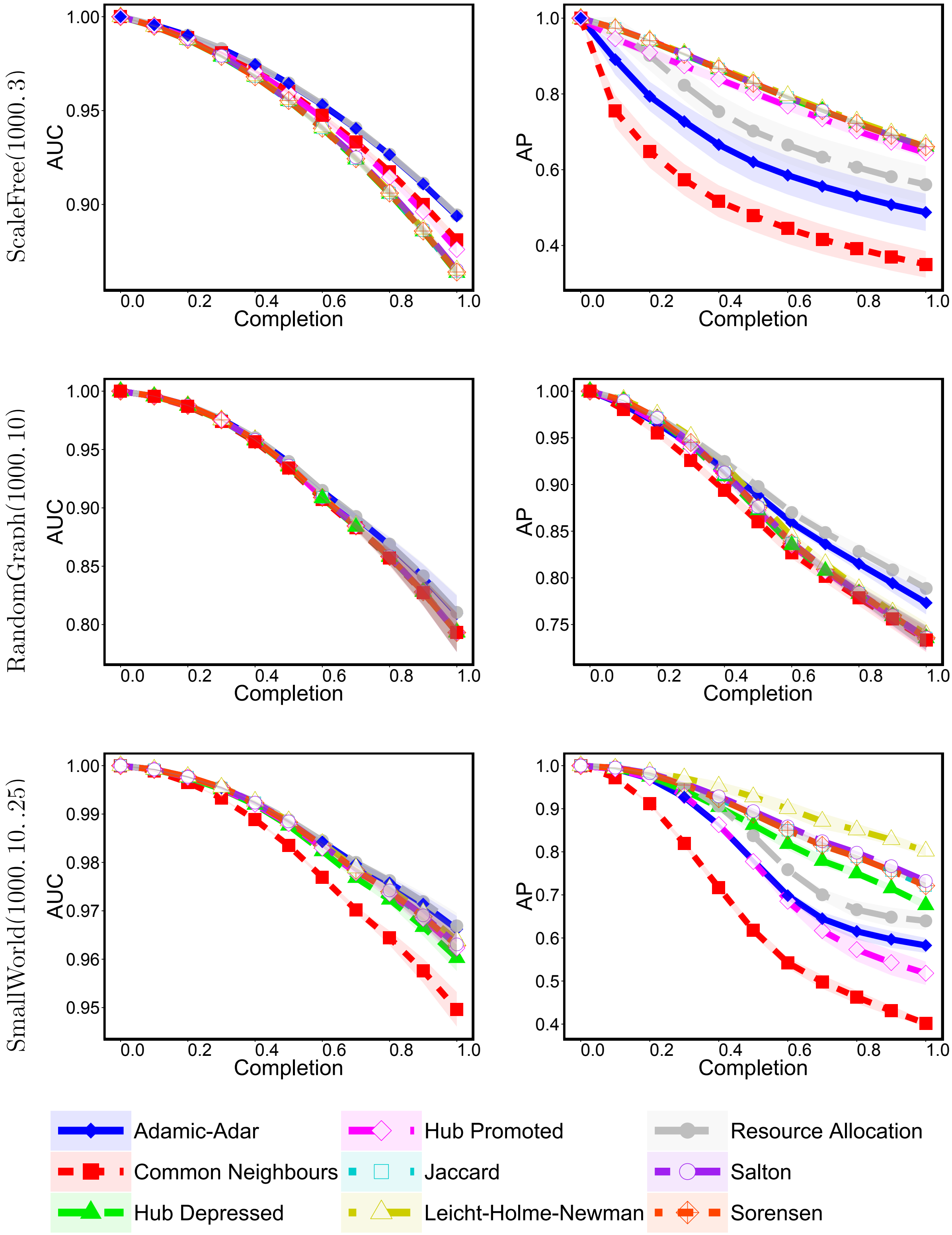

Figure F.9: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of OTC. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.

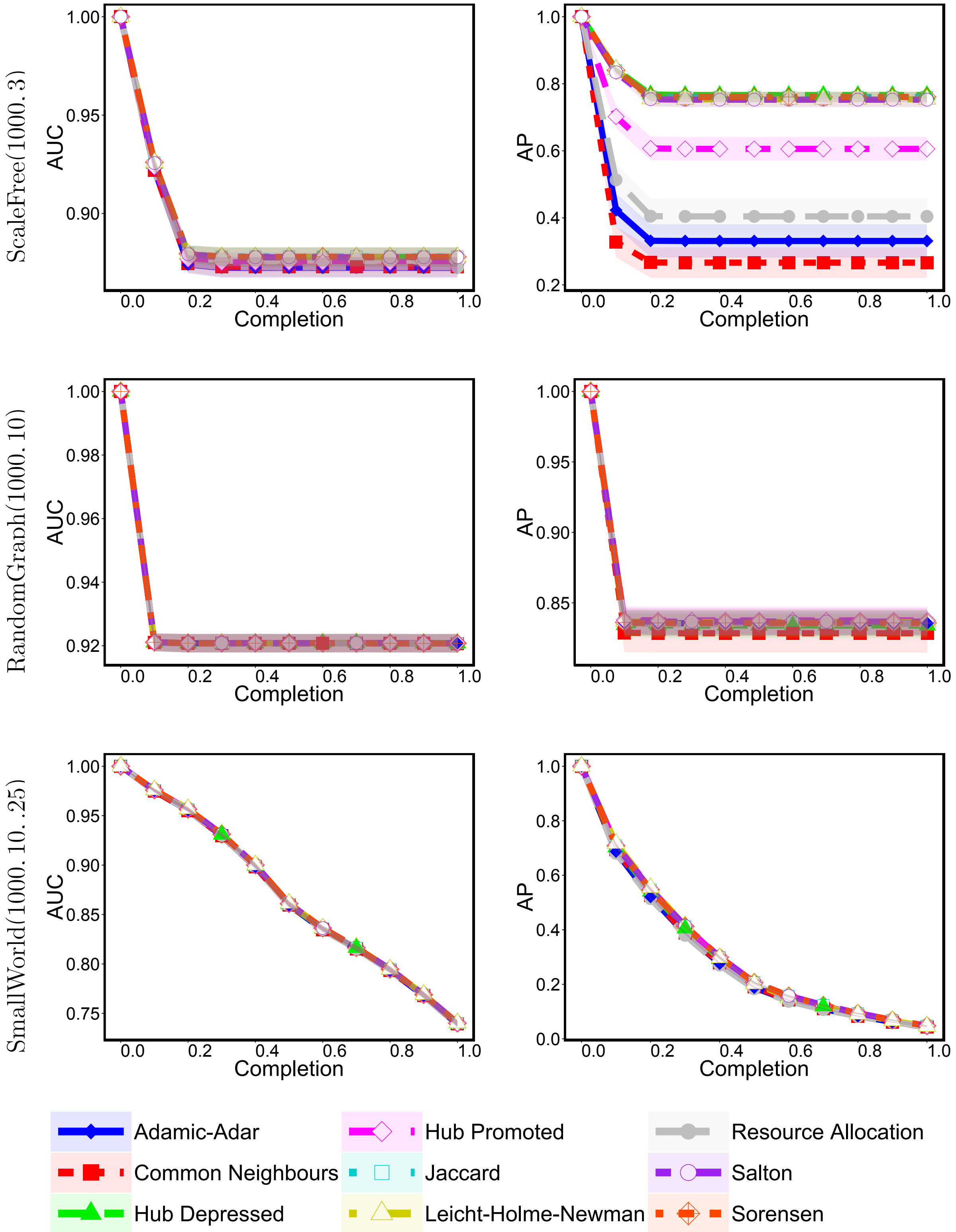

Figure F.10: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of CTR. Results are shown for an average of 50 executions, coloured areas representing the 95% confidence intervals.