DOCTORAL SCHOOL
MEDITERRANEA UNIVERSITY OF REGGIO CALABRIA

DEPARTMENT OF INFORMATION ENGINEERING, INFRASTRUCTURES
AND SUSTAINABLE ENERGY
(DIIES)

PHD IN
INFORMATION ENGINEERING

S.S.D. MAT/05 AND MAT/09
XXXVI CYCLE

Trust and Reputation Systems: Detection of Malicious Agents and a Novel Equilibrium Problem

CANDIDATE
ATTILIO MARCIANO'

ADVISORS
Prof. SOFIA GIUFFRE'

COORDINATOR
Prof. ANTONELLA MOLINARO

REGGIO CALABRIA, FEBRUARY 2024

Finito di stampare nel mese di **XXX 2024**

Edizione

Quaderno N. XX

Collana *XXXXXXXX*
Curatore *XXX*

ISBN XXX-XX-XXXXX-XX-X

Università degli Studi *Mediterranea* di Reggio Calabria
Via dell'Università 25. Reggio Calabria

ATTILIO MARCIANO'

Trust and Reputation Systems: Detection of Malicious Agents and a Novel Equilibrium Model

Abstract

This thesis is devoted to the analysis and development of trust and reputation systems, focusing on the study of accurate models for the computation of reputation, especially in virtual contexts such as social networks. By analysing what causes users to assign trust and how reputation is perceived, we have proposed advanced models that aim at accurately capturing the dynamics that influence reputation formation. We have examined the Eigentrust algorithm, one of the most effective solution to measure the reputation in a set of social agents. We have highlighted a possible limitation in the Eingentrust algorithm, observing that it increases the reputation of all the pre-trusted agents, regardless of their reliability. To solve this problem, as first algorithm, we have proposed a different strategy that introduces the advantage, with respect to Eigentrust, of estimating the reputation values of the honest actors in a manner more close to the actual reliability of these agents. Instead, the second proposed algorithm was designed for the detection of colluding agents divided into multiple clusters. It combines the EigenTrust algorithm with a clustering procedure, grouping agents based on their reputation scores. Finally, we apply the theory of variational inequalities in a virtual environment, in which users evaluated specific objects. We demonstrated that the equilibrium conditions of the system can be formulated as a variational inequality problem. We explored the robustness of our model across different conditions, introducing significant variations in both user trustworthiness and initial reputation of objects.

Contents

1 Introduction 1
1.1 Motivation 1
1.2 Thesis structure 2

Part I Collusion in Trust and Reputation System

2 Trust and Reputation in Multi-Agent System 9
2.1 Trust and Reputation Systems 9
2.2 Reputation network architectures 11
2.3 Agents and Multi Agent Systems 12
2.3.1 An introduction to Agents 13
2.3.2 Multi-Agent Systems 14
2.4 Agent organization 16
2.5 Malicious Agents 18

3 Identifying Colluding Agents in Social Communities by Reputation Measures 21
3.1 Related work 23
3.1.1 Pagerank model 23
3.1.2 Eigentrust Algorithm 24
3.2 Effective reputation computation 26
3.2.1 The Recursive Reputation Model 26
3.2.2 Detection of potential malicious users 28
3.2.3 Our algorithm to compute the reputation 29
3.3 An illustrative example 30
3.4 Error measure 33
3.5 Experimental Results 36
3.6 Benefits and Limitations 39

4 A Graph Clustering algorithm in Trust and Reputation Systems 41
4.1 Literature review on Spectral Clustering Algorithms 42
4.1.1 Similarity graphs 43
4.1.2 Graph Laplacians 44
4.1.3 Spectral Clustering Algorithms 44
4.2 Computing the reputation 46
4.2.1 The model 46
4.2.2 Detection of malicious users through graph clustering 47
4.3 An explicative example 50
4.4 Experimental Results 52
4.5 Benefits and Limitations 56

Part II A Variational Approach in Trust and Reputation Systems

5 Variational Problems 61
5.1 Model-optimizing approach 61
5.2 Optimization models 62
5.3 A brief recall to Lagrange Theory 64
5.4 Variational Inequality Theory 65

6 A Variational Formulation for a Trust and Reputation System 71
6.1 Literature Review 72
6.2 The Mathematical model (for users and objects) 73
6.3 An equivalent formulation 77
6.4 Numerical simulations 79
6.4.1 Simulation 1 (with Lagrange multipliers) 79
6.4.2 Simulation 2 (with a realistic community) 82
6.4.3 Simulation 3: Sensitivity analysis 83
6.5 Observations 85

7 Conclusions and Future Works 87

List of Figures

3.1 T, $\widetilde{T}$, $\widehat{T}$ matrices corresponding to the Example in Section 4.3. 34

3.2 Reputation vectors obtained with the two different approaches........ 34

3.3 Average errors in 2-norm with 10% of malicious users for different values of the matrix size. 37

3.4 Average errors in 2-norm for a social network with $n = 5000$ and different ratios of malicious users 38

3.5 Average errors in 2-norm for a social network with $n = 25000$ and different ratios of malicious users 38

4.1 Reputations obtained by using and not using malicious 51

4.2 Precision of Cluster and ER-EigenTrust methods as agents increase. ... 55

4.3 Precision of Cluster and ER-EigenTrust methods as malicious agents increase. 56

5.1 Geometric interpretation of $VI(F,K)$ 67

6.1 Range of Optimal Solutions 83

List of Tables

3.1 Notation 26

4.1 Results of common evaluation metrics for small communities. 53
4.2 Results of common evaluation metrics for medium communities. 54

6.1 Parameters used in Simulation 1 79
6.2 Optimal solutions of Simulation 1 81
6.3 Difference between the utility and the sum of costs: $u_j(w) - c_{ij}(w) - c_j(w)$. 81
6.4 Optimal trust values 82
6.5 Sensitivity analysis on a_i 84

1 Introduction

The first chapter of this thesis is devoted to explore in detail the motivations that constitute the driving force of our research over these years. Furthermore, crucial preliminary concepts related to multi-agent systems will be provided. We highlight the advantages (and disadvantages) of the trust and reputation system, in order to establish a solid conceptual basis for understanding the study context. Finally, the overall structure of the thesis will be outlined, providing an advance overview of the topics covered in the subsequent chapters. In particular, we focus our attention in a first part on Trust and Reputation System in peer to peer network, in a second part on Trust and Reputation System between agents and objects.

1.1 Motivation

The modern paradigm of social network represents the human need for a communication space, different from the traditional physical places as home and work, where people can talk, write and express their opinions, make several kinds of activities as business transactions, learning courses, challenging research etc., generally giving the possibility to the community members who interact to share knowledge, insight and ideas. This paradigm has been implemented in generalist social networks as Facebook and Twitter, and in thematic social network as Linkedin, Flickr and Researchgate, but also in e-commerce platforms as Èbay and Amazon, or in comparison shopping website as Tripadvisor. In all these different domains, a key issue is represented by the possibility to introduce a sufficient level of trustworthiness associated with the community members, generally called *agents*, since they usually perform some kind of activities and they could be either humans or software entities acting on behalf of humans, and where by the term trustworthiness we usually mean the possibility for each agent of the community to have trust in the other agents. This is particularly important both for competitive and collaborative agents [1, 2], since in both cases an agent has to accurately select its interlocutors, because both its incomes and outcomes depend on this choice, considering the possibility that malicious agents could be present in the community with the purpose to realize personal advantages engaging in deceptive behavior and making frauds. This makes it difficult to make decisions about which resources can be relied upon and which entities it is safe to interact with. **Trust and Reputation systems** are aimed at solving this problem by enabling service consumers to reliably assess both the quality of services and the reliability of entities, before deciding to use a particular service or to interact with or to depend on a given entity. Such systems should also allow serious service providers and online players to correctly represent the reliability of themselves and the quality of their services. In the case of reputation systems, the basic idea is to let parties rate each other, for example after the completion of a transaction, and use the aggregated ratings about a given party to derive its reputation score. In the case of trust systems, the basic idea is to analyse and combine paths and networks of trust relationships in order to derive measures of trustworthiness of specific nodes. Reputation scores and trust measures can assist other parties in deciding whether or not to transact with a given party in the future, and whether it is safe to depend on a given resource or entity. This represents an incentive for good behaviour and for offering reliable resources, which thereby tends to have a positive effect on the quality of online markets and communities.

Thus, we can state that dealing with trust and reputation systems is important for several reasons. These systems are crucial in online environments to build trust and security among users. In online commercial transactions, they help evaluate the reputation of sellers and products. In sharing and collaborative contexts, such as car-sharing platforms, reputation becomes crucial to encourage reliable behavior. They also contribute to preventing online fraud and maintaining the quality of user-generated content. In cases of disputes, trust systems facilitate fair resolution. Additionally, they incentivize positive behavior and allow for the evaluation of skills in online professional environments. Overall, these systems contribute to creating safe, reliable, and collaborative online environments.

1.2 Thesis structure

The thesis is divided into two distinctive parts. In the first part, we adopt a traditional approach, in accordance with the principles of information engineering and data processing. We focus on the study of trust reputation systems in modern social networks, especially common in e-commerce and e-learning, analysing the fraudulent behaviour of a specific category of agents. Infact, a particularly dangerous type of fraud is represented by *collusion*, i.e. a deceitful agreement or secret cooperation between two or more agents to limit open competition by deceiving, misleading or defrauding others of their legal right. We use established methods to thoroughly investigate and understand these dynamics, drawing on a classical perspective typical of the field.

In the second part, we take an innovative direction, embracing the theory of variational inequalities and Lagrange multipliers. This state-of-the-art methodology allows us to explore the systems in question in a more in-depth and advanced manner, introducing a more sophisticated level of analysis. The variational approach offers a unique perspective to address the complex dynamics of such trust reputation systems, enriching understanding and opening new perspectives for the management and optimisation of such systems in real-world contexts.

In detail, in chapter 2, the concept of trust and reputation in multi-agent systems is introduced. It provides in-depth definitions and analyzes crucial aspects of both concepts. The two main network architectures, centralized and distributed, are also explored, outlining their advantages and disadvantages. In particular, these systems find application in the context of multi-agent systems. The concept of an agent is defined, outlining its objectives and how it can be organized in collaboration or competition with other users. The dynamics of interactions between agents are analyzed, focusing on how they can cooperate or compete to achieve established goals. A crucial aspect that emerges is the presence of malicious agents within these organizations. The activities of these agents are explored, whose actions are aimed at completely disrupting the reputation of agents within a social network. This raises significant challenges for trust and reputation management in multi-agent systems, requiring specific strategies to mitigate the risk from harmful behaviors.

In this respect, in Chapter 3 we propose a strategy which allows to preliminarily detect the best candidates as malicious (colluding) agents directly from the trust matrix. We present an overview of the state of the art in this field, then we focus on the main algorithm used in this specific situation: the Eigentrust algorithm. It represents an efficient variant of PageRank and, in addition, stands out for its efficiency in calculating and dynamically updating users' reputations. However, it has an important disadvantage: it requires prior knowledge about user characteristics, including potential malicious users. We overcome this drawback proposing a new algorithm to compare the effectivness of our results with that generated bu Eigentrust. We introduce a measure of effectiveness when computing reputation in presence of malicious agents, defining a metric of error useful to quantitatively determine how much an algorithm for the identification of malicious modifies the correct reputation values of the honest agents.

In Chapter 4 we propose an alternative strategy to the previous one to detect potential malicious colluding agents by combining the Eigentrust algorithm with clustering techniques. Our approach makes use of the spectral properties of a specific matrix, known as the Lagrangian. In the course of chapter, we provide a broad overview of these techniques, with a particular focus on the similarity metrics used to cluster data, or in our case, agents. Through the results obtained, we compare this new strategy with the previous approach, showing that, in the presence of various groups of malicious colluding agents, our new algorithm proves to be significantly more efficient, with high precision.

In Chapter 5 we introduce an innovative methodological framework, the variational inequality theory, that will be used in the second part of the thesis. We not only offer practical indications on how to approach an optimization model typical of applied mathematics, but we also delve into the context by underlining the interesting link between the optimization problem and the theory of variational inequalities. In particular, we explore the possibility of expressing the latter as a minimization problem, thus highlighting the connection between these two fundamental concepts. In addition, we provide an overview of the Lagrangian theory, elucidating how Lagrange multipliers play a crucial role in the analysis of solutions. We offer a detailed explanation of how these multipliers represent fundamental elements in the context of optimization, contributing significantly to the understanding and interpretation of problem solutions.

In Chapter 6 we present a detailed mathematical model of a trust and reputation system, where users express their votes to indicate satisfaction or dissatisfaction with a particular product. We further delve into the context by introducing the concept of user weight and the initial reputation of objects, key elements that dynamically influence the system. To better understand the dynamics, we consider the equilibrium conditions of the model, evaluating how the utility function and the cost functions converge towards a point of equilibrium. Next, we introduce constraints on the model solution, providing an equivalent variational formulation. This formulation, based on optimization concepts, allows us to obtain a more in-depth perspective on the behavior of the system. We present existence and uniqueness results derived from this formulation, highlighting the mathematical robustness of the proposed model with several simulations.

Finally, in chapter 7 we provide a brief summary of the results obtained through the use of strategies to identify malicious users, referencing the advantages and disadvantages of each strategy. Concisely, we outline the innovative aspects arising from the application of Lagrangian theory and variational inequalities in the contexts of trust and reputation systems. We conclude with a mention of future research, offering brief insights and future plans. These could include further developments in detection strategies, an in-depth exploration of the practical implications of Lagrangian theories and variational inequalities in real-world scenarios, and consideration of broader application scenarios. In this way, we emphasise the continuity of the research and the potential for further innovative contributions to the field.

Part I

Collusion in Trust and Reputation System

2

Trust and Reputation in Multi-Agent System

Virtual interactions between people and services without any previous real world relationship have experienced an exponential increase with the availability of interactive on line sites, including the so called social networks. Despite the diverse range of online communities, they all share a common feature: a large number of users interacting under virtual identities. These emerging social media platforms enable users to establish explicit or implicit social connections and utilize the network as a global platform to share and disseminate products, services, information, opinions, and recommendations. In these digital media scenarios, evaluating the credibility of information poses a more complex challenge compared to traditional media due to its inherently anonymous and open nature, characterized by a lack of robust governance structures. This anonymity creates a conducive environment for malicious users to disseminate misinformation, viruses, or files, especially in Peer-to-Peer networks (P2P). Therefore, it becomes essential to implement mechanisms that facilitate the selection of interaction peers and efficiently identify and isolate malicious actors. The ideal solution is the development of ***Trust, Reliability and Reputation system*** **(TRRs)**, i.e. a network of reputation and trust, either at a local level (individual websites for example) or across the whole web, that allows users to express and propagate trust on others to the entire network to allow other users to assess the quality of the information or service provided even without a prior interaction with the agent in question.

2.1 Trust and Reputation Systems

Trust, Reliability and Reputation systems are tools designed to identify the trustworthiness of an actor, be it a human, a software or a device, in order to detect malicious actors and marginalize or expel them from the community.

Several general purpose and context specific TRRs have been developed and, on the one hand malicious actors can perform various misleading behaviors [3–5], even simultaneously, for deceiving TRRs and obtaining some undue benefits; on the other hand, TRRs are designed to be resilient to these attacks [6, 7]. In this setting, the interactions between the community members are considered as e-services that a provider agent (the trustee) gives to a client agent (the trustor), and trust can be formulated [8]:

Definition 2.1 (Trust). *Trust is the quantified belief by a trustor with respect to the competence, honesty, security and dependability of a trustee within a specified context.*

It is important to highlight that the concept of trust involves several different dimensions, depending on the viewpoint under which the interaction between the two agents is considered, and for each of these issues, trust increases the spectrum of interactions between two agents. Some common dimensions of trust are:

- **reliability**: ability of effectively and efficiently performing requested activities. In various contexts, reliability is associated with the ability of a system, person, or thing to consistently perform a specific function or deliver results with a high degree of accuracy and consistency over time. In interpersonal relationships, reliability may be related to the trustworthiness and consistency of an individual's actions and commitments. In the context of machines or systems, reliability often implies a low probability of failure or malfunction over a specified period, indicating their consistent and dependable performance.
- **honesty**: willing to operate a truthful behavior, avoiding fraudulent or misleading activities.Honesty is a quality characterized by truthfulness, integrity, and straightforwardness in one's actions, communication, and interactions. An honest individual (or entity) is someone who adheres to ethical principles, refrains from deceit or misleading behavior, and is sincere in expressing thoughts, feelings, and information. Honesty involves transparency, openness, and a commitment to upholding the truth even when faced with challenges or difficult situations. Honest behavior builds trust, fosters credibility, and promotes a positive and ethical environment.
- **security**: ability to protect assets, information, systems, and individuals from various forms of threats, risks, and unauthorized access. It encompasses a wide range of strategies, technologies, and protocols designed to ensure the confidentiality, integrity, and availability of resources.

We can say that trust is a key element in interpersonal relationships, commercial transactions, social competitions and online interactions. Building it requires time, consistency and the demonstration of reliable and respectful behavior.

While trust is a subjective measure, evaluated by a trustor with respect to a trustee (in a peer-to-peer interaction), *reputation* is instead a measure of the trustworthiness that the whole community evaluated with respect to a given agent. The concept of reputation is closely linked to that of trustworthiness, but it is evident that there is a clear and important difference. For the purpose of this study, we will define reputation according to the Concise Oxford dictionary.

Definition 2.2 (Reputation). *Reputation is what is generally said or believed about a person's or thing's character or standing.*

Reputation can also be interpreted as the expectation and perception that a trusting entity holds about a trusted entity from its past actions and behaviours with other entities.

The concept of reputation assumes a relevant role in all those situations where an agent x is not provided with a sufficient knowledge about another agent y. In those cases, x might use the y's reputation to decide if y can be considered as a reliable interlocutor. Reputation can be considered as a collective measure of trustworthiness (in the sense of reliability) based on the referrals or ratings from members in a community. An individual's subjective trust can be derived from a combination of received referrals and personal experience. In order to avoid dependence and loops it is required that referrals be based on first hand experience only, and not on other referrals. It is possible to abandon this principle for example when the weight of the trust referral is normalised or divided by the total number of referrals given by a single entity, and the latter principle is applied in Google's PageRank algorithm [9] described in more detail in next chapter.

2.2 Reputation network architectures

In a reputation system, once a transaction between two agents is completed the agents are required to rate the quality of the transaction (service). The architecture of the these systems determines the way in which the ratings and reputation scores are collected, stored and shared between the members of the system. There exist two main types of reputation network architectures: centralised and distributed.
In a centralised reputation system a central authority collects all the ratings and constantly updates each agent's reputation score as a function of the rating the agent received. This type of system, as reported in [10], requires

(i) a centralised communication protocol in charge of keeping the central authority updated from all the ratings;
(ii) a reputation calculation method for the central authority to estimate and update the reputation of each agent.

In many practical situations, the social networks, in which reputation measures are computed, are composed by a very large number of agents, and the computation activity is too loud to be implemented in a unique centralized unit, since such a centralization would imply obvious problems in terms of efficiency and robustness of the communication activities. In these situations, the computation of the reputation values is delegated to a distributed architecture, composed of local computing nodes, each one associated to a group of agents and which is capable of computing the reputation values of only the agents belonging to that group. These local values are then transmitted to higher level nodes in a hierarchical organization, and each node at a given level collects reputation values from a set of nodes belonging to the lower level, combining these values to compute reputation values at the given level, and then sending in its turn these values at the higher level.

In a distributed reputation system each agent individually collects and combines the ratings from the other agents. That is, an agent A, who wants to transact with another target agent B, has to demand for ratings to the other community members who have directly interacted with agent B. Consequently, given the distributed nature of the information, obtaining the ratings from all interactions with a given agent may be too expensive (time consuming) and so, only a subset of the interactions, usually from the relying agents' network are considered to calculate the reputation score. This type of system requires

(i) a distributed communication protocol to allow agents to get information from others agent they are considering to transact with;
(ii) a reputation computation method to estimate and update the reputation given the values of other agents (neighbours).

A well known example of distributed architecture are P2P networks in which each agent acts as both client and server. It is noted that these networks may introduce a security threat since they could be used to propagate malicious software or to bypass firewalls. Therefore the role of reputation in this particular case is crucial to determine which nodes in the network are most reliable and which ones should be avoided.

2.3 Agents and Multi Agent Systems

One of the fields that most use the concepts of Trust and Reputation is the field of ***multi-agent systems*** (**MAS**). These systems are composed of autonomous agents that need to interact to each other to achieve their goals. The parallelism with human societies is obvious, and also the problems, specially when we are talking about open MAS.

2.3.1 An introduction to Agents

Before approaching the study of multi-agent systems, it is essential to rigorously define the agent itself and its characteristics ([11]).

Definition 2.3 (Agent). *An Agent is an entity which is placed in an environment and senses different parameters that are used to make a decision based on the goal of the entity. The entity performs the necessary action on the environment based on this decision.*

The above definition comprises four keywords which can be further elaborated:

- In the given context, **entities** or agents can take on various forms, each characterized by unique attributes. Software agents, for instance, may include daemon security agents that operate in the background, diligently monitoring and safeguarding systems against security threats.
  On the hardware front, entities like thermostats serve as tangible components regulating the temperature in environments, particularly within heating and cooling systems. Sensors, another hardware type, detect and quantify diverse physical parameters, ranging from temperature to brightness and pressure.
  In the virtual realm, entities like virtual assistants demonstrate the prowess of software in providing virtual support to users. Siri, Google Assistant, and Alexa are prime examples of these virtual agents. Additionally, simulations represent software entities replicating the intricate behaviors of complex systems for analytical purposes, be it in traffic management or market dynamics.
- The term **environment** refers to the specific context or setting where an agent operates, such as a network or software environment. The information the agent gathers from its environment is crucial for decision-making in agent-based systems.
  Key features affecting these systems include accessibility, determining how accurately agents can collect data. Determinism relates to outcome predictability, with deterministic environments providing precise outcomes, while non-deterministic ones yield less predictable results influenced by various factors.
  Dynamism addresses changes in the environment independent of agent actions. Dynamic environments require agents to detect and update information, incurring more overhead compared to static environments where initial information remains relevant.
  Continuity classifies the MAS environment as continuous or discrete. In continuous environments, the agent's state is influenced by a continuous function,

such as movement in a physical environment. Discrete environments confine the agent to predetermined states.

- In the domain of agent-based systems, **parameters** refer to the varied data agents can perceive, contingent on the application. Examples include transportation systems where agents assess traffic conditions, smart homes monitoring room-specific data, and production systems overseeing machinery and inventory. Security systems involve agents detecting intrusions, while medical applications focus on vital signs. In multiplayer games, agents consider player status, and environmental management involves monitoring air quality. In e-commerce, agents analyze user behavior, and financial applications track market trends. These instances exemplify the adaptable nature of agents in diverse contexts.
- Each agent can perform an **action** that results in some changes in the environment. Actions refer to the varied functionalities agents can perform based on specific applications. For instance, in transportation, agents optimize vehicle trajectories and traffic flow. In smart homes, they regulate temperature and manage appliances. In production systems, agents control machinery and adjust production quantities. Security agents respond to intrusions and emergencies. Healthcare agents administer medication and respond to patient needs. In gaming, agents navigate virtual environments and collaborate with other players. Environmental agents activate purification systems and monitor climate shifts. E-commerce agents personalize recommendations and manage inventory. Financial agents execute transactions and adapt portfolios. These examples illustrate the adaptability of agents across diverse applications.

The goal of each agent is to solve its allocated task with some additional constraints, e.g. a deadline. To achieve this aim, the agent first senses parameters from the environment. Empowered with this data, the agent can build up knowledge about the environment. An agent might also use the knowledge of its neighbors. This knowledge along with the history of the previous actions taken and the goal are fed to an inference engine which decides on the appropriate action to be taken by the agent. While an agent working by itself is capable of taking actions (based on autonomy), the real benefit of agents can only be harnessed when they work collaboratively with other agents. Multiple agents that collaborate to solve a complex task are known as Multi-Agent Systems (MAS).

2.3.2 Multi-Agent Systems

A multi-agent system is a sophisticated extension of agent technology, in which the dynamics take place within an environment populated by a collective of autonomous and freely connected agents. These agents, similar to independent actors on a dynamic stage, engage in purposeful actions within a shared environment, all directed towards the achievement of a common goal. It is a digital theatre in which each agent plays a role, contributing to the overall plot through cooperation or competition.

The success of Multi-Agent Systems (MAS) across diverse sectors can be attributed to a set of significant advantages that contribute to their widespread adoption. These systems, representing an advanced extension of agent technology, excel in coordinating autonomous actions within a shared environment to achieve a common goal. A key aspect defining MAS is their ability to enhance the speed and efficiency of system operations. This is achieved through the use of parallel computation and asynchronous operations, creating a synergy of agents working simultaneously for quicker responses and increased overall agility.

A distinctive feature is the MAS's capability to address challenges and agent failures with "graceful degradation." Instead of experiencing a complete collapse in the event of a failure, the system dynamically adapts, maintaining functionality with reduced capacity. This flexibility contributes to improving the overall robustness of the system.

Scalability and flexibility represent another key advantage of MAS. The ability to dynamically add agents to the system allows for gradual growth and adjustments without the need for radical changes in the structure. This feature results in a system that can evolve with changes in requirements and scale. A pragmatic element is the cost-efficient management facilitated by the decentralized architecture of MAS. Autonomous agents, operating individually, often incur lower costs compared to centralized solutions. This makes MAS an economically attractive choice.

Moreover, the modular structure of agents in MAS promotes reusability. The replacement or upgrade of agents can be executed seamlessly, contributing to greater flexibility and long-term adaptability of the system.

MAS are composed of autonomous agents that need to interact to each other to achieve their goals. The parallelism with human societies is obvious, and also the problems, specially when we are talking about open MAS. The main feature that characterizes open multi-agent systems is that the intentions of the agents are unknown. Hence, due to the uncertainty of their potential behavior we need mechanisms to control the interactions among the agents, and protect good agents from fraudulent entities. Traditionally, as reported in [12], three approaches have been followed to solve such problems:

1. **Security Approach**: At this level, basic structural properties are guaranteed, like authenticity and integrity of messages, privacy, agents' identities, etc. They can be secured by means of cryptography, digital signatures, electronic certificates

etc. However, this approach does not tell anything about the quality of the information, although the established control is more than valuable.

2. **Institutional Approach**: This approach assumes a central authority that observes, controls or enforces agents' actions, and might punish them in case of non-desirable behaviors. It is indisputable that this approach ensures a high control in the interactions, but it requires a centralized hub. Moreover, the control is bounded to structural aspects of the interactions: allowed, forbidden and obliged actions can be checked and controlled. However, the quality of the interactions is left apart, in part, because a good or bad interaction has a subjective connotation that depends on the current goals of each individual agent.
3. **Social Approach**: Reputation and trust mechanisms are placed at this level. In this approach agents themselves are capable of punishing non-desirable behaviors, by for instance, not selecting certain partners. To achieve such distributed control agents must model other agents' behaviors, and following the similitude with human societies, trust and reputation mechanisms arise as a good solution. This requires however the development of computational models of trust and reputation, which must cover not only the generation of social evaluations in all the dimensions, but knowledge on how agents use reputation information to select partners, how agents communicate and spread reputation, and how agents handle communicated reputation information. It is important to remark that these three approaches are complementary and that each one covers a different typology of problems, all related to the control of interactions on open MAS.

An essential aspect of multi-agent systems is their organization, a topic that will be explored in the next section

2.4 Agent organization

An integral facet of Multi-Agent Systems lies in the orchestrated arrangement of agents, where the organizational dynamics play a pivotal role in shaping the collaborative or competitive interactions within the system. The strategic deployment and coordination of autonomous agents contribute significantly to the overall effectiveness and adaptability of MAS in diverse application domains. The prevailing approaches for organizing MAS, as reported in [11], are given below.

In the **flat** organizational structure, all agents are considered equals, with no designated leader. Communication takes place directly between each agent and its neighbors, creating a decentralized network.

In **hierarchical** organization, agents have tree-like relations. Leaf agents, communicate with other agents using their parents. Parents control their leaf agents, i.e., children, and may have their own parents. At the highest level, there is one agent known as root agent. Hierarchical organization may lead to delay or create a bottleneck, particularly at the root agent (or parents) as it is responsible for processing communications of all leaf agents. Based on the number of authoritative agents, i.e. those who have control over other agents, the hierarchical organization can be divided into two types namely simple and uniform. In the simple approach, the root agent has exclusive authority and controls all communications. In the uniform approach, there are more than one authoritative agents in the hierarchy meaning that in addition to the root, all or particular parents can also control their children.

In **coalition** organization, agents are temporarily grouped based on their goal. Consequently, the agents can reach their own goal with lower (processing delay and communication) overhead compared to the organization where there is no such grouping. Each agent can be part of more than one coalition. By reaching their goal, the agents destroy the coalition. The internal organization of a coalition is normally flat; however, other organizations, e.g., hierarchical organization, can be used to further reduce overheads or apply control over agents. Finding and grouping agents with the same goal incurs processing and communication overhead on agents. Thus, there is a trade-off between the decreased overhead resulting from the coalition and the incurred overhead for finding agents with the same goal and forming them in coalition. This organization is suited when a collection of agents with similar goals exists in MAS which their collaboration associates them in reaching their goal.

In **team** organization, the agents create a group (team) and define a group goal which differs with their own goal. Depending on the time required to reach the team goal, a team may be short-time or longtime. The agents in a team collaborate to reach the team goal. The team goal can be updated which leads to change in the responsibilities, roles and authorities of agents in the team. Each team can request information from agents in other teams to improve its own decision-making process. A team can have an internal organization (e.g. hierarchical) to improve the performance and efficiency in reaching the team goal. The final decision of the team is less challenging in small teams, however, the data used by small teams is limited. Unlike coalition where agents are grouped to reach their own goal, in a team agents attempt reaching the team goal. Thus, this organization is suited when multiple agents attempt reaching the same goal.

In **congregation** organization, agents in a location form a congregation to achieve their requirements that they cannot achieve alone. Each agent can leave or join congregations but should be part of only one congregation at each point of time. The satisfaction of an agent in congregation, i.e., the degree in which an agent fulfills its requirements, depends on other agents in the congregation. A congregation should always have at least one member. This organization is affective where each agent requires the resources of other agents to achieve its goal or perform its tasks.

Given the vast diversity in which agents can be organized within multi-agent communities, there is a compelling need to employ appropriate trust and reputation systems tailored to the specific context. The complexity arising from the structural and functional variations among agents calls for a strategic approach to assess mutual trust and reputation within the system. In this context, the selection and design of such systems become pivotal aspects to ensure optimal functioning, as trust dynamics must adapt to the peculiarities of each organizational setting. Establishing a trustworthy environment and managing reputations thus become fundamental cornerstones to support effective and collaborative relationships among agents, fostering seamless interaction within multi-agent communities. Namely, the goal of trust and reputation systems is to mitigate the risks associated with interactions involving potential malicious agents.These systems focus not only on performance assessment but also on establishing robust mechanisms to identify and address potential threats posed by agents that may act maliciously. In this context, the priority is to implement advanced strategies and algorithms that can effectively evaluate and manage the trust and reputation of agents, providing a secure foundation for interactions within a multi-agent environment.

In the next section, we will introduce the key attributes that define malicious agents and provide a brief explanation of the potential threats such agents may present.

2.5 Malicious Agents

The interest in implementing appropriate trust and reputation models becomes particularly acute when considering the potential presence of **malicious agents**, emphasizing the importance of developing robust and adaptive mechanisms that can effectively mitigate the risk of unethical or fraudulent behavior within the dynamics of multi-agent communities. While global reputation is efficient and helps quickly detect misbehavior in the system, it is vulnerable to false ratings. On the other hand, reputation ratings directly derived from firsthand experience are highly reliable, but do not help blacklist malicious peers for others. Also, firsthand information only proves effective if a peer locates honest service providers with which to repeatedly transact [13]. Since global reputations provide significantly more information than firsthand reputations, reputation systems predominantly employ them. We enlist, as reported in [14], some types of ratings misbehavior commonly observed due to global reputation aggregation.

- **Dishonest agents**. An honest agent is one that is truthful in its ratings of other agents. A dishonest agent, on the other hand, attempts to subvert a system by falsely rating a bad transaction as good, and vice versa. Such biased ratings presented due to jealousy, competition, or other malicious reasons adversely affect the quality of reputation scores generated by a reputation system.
- **Dynamic personalities**. Some agents exhibit a dynamic personality, switching between honest and dishonest behavior. Behavior changes can be based on the type or value of the transaction or the party involved at the other end. Reputation milkers, or oscillating agents, attack a reputation system by first building a good reputation and then taking advantage of it to do harm.
- **Collusion**. Collusion occurs when two or more agents collaboratively enhance each other's reputations or conspire against one or more agents in the network [15]. In the form of collusion known as *ballot stuffing*, a colluding group artificially inflates each other's reputations, enabling them to exploit the positive reputation to launch attacks on other system agents. Another manifestation of collusion is *bad-mouthing*, wherein a malicious collective conspires against one or more agents in the network by assigning unfairly low ratings to the target agents, thereby damaging their reputation. Lastly, positive (and negative) *discrimination* emerges when agents provide excellent (and poor) service selectively to a few targeted peers.
- **Sybil-based collusion**. The Sybil attack occurs in the absence of a centrally trusted party, when an agent with sufficient resources can establish a potentially unbounded number of distinct online identities (or Sybils) [16][17][18]. Recently it was shown that users can use these identities to collude and artificially inflate their own reputations in order to monopolize service, lure users into scams, or otherwise gain performance benefits from the system [19].
- **Churn attacks**. While reputations have been deployed in online marketplaces suchas *eBay.com*, they are not necessarily a natural fit for the dynamic nature of P2P overlay networks. Since reputations assess a peer's trustworthiness using historical feedback of its past interactions, longer peer lifetimes lead to more interactions, and a more accurate reputation. Distributed communities like P2P file sharing networks, however, experience significant churn (or peer turnover) which means a high percentage of peers will have relatively "short-term" reputations accrued from a small number of past interactions. For instance, malicious peers penalized by reputation systems for poor performance have the ability to rejoin the network with newly acquired identities and a clean history. Such churn attacks result in erroneous or misleading reputations for malicious peers.

In the upcoming two chapters, we will delve into the examination of adaptive trust and reputation systems in the presence of colluding malicious agents. This exploration will involve a detailed analysis of strategies and mechanisms designed to adapt and respond effectively to the challenges posed by agents who engage in collusion with malicious intent.

3

Identifying Colluding Agents in Social Communities by Reputation Measures

A particularly dangerous type of fraud in modern social networks is represented by *collusion*. Collusion consists in a deceptive agreement or secret cooperation between two or more agents to restrict free competition by cheating, misleading or defrauding others of their legal right.

Many strategies to deal with malicious agents in social networks have been proposed in the literature.

More in general, among these proposals, one of the most effective is represented by PageRank [9], the well-known algorithm used by the Google search engine for determining result rankings. In PageRank, the importance (reputation) of a web page is measured, in particular, according to the number of links to that specific web page.

An important alternative to PageRank is Eigentrust, which has the primary focus on determining sureness and authenticity of traffic in P2P Networks. Thanks to its effectiveness, Eigentrust is also one of the most highly considered benchmarks to evaluate/train other systems identifying malicious agents.

These two algorithms, are used in our model and then we will analyse them in detail in Sections 3.1 together with a briefly overview of other strategies.

In a nutshell, the Eigentrust algorithm aims at calculated the reputations of the agents an detecting the malicious users. To this end, it uses some additional information about agents that can be a-priori considered particularly trustworthy (i.e. the pre-trusted agents), rewarding them in terms of reputation, while the non pre-trusted agents are penalized. It applies its rewarding operation to all the pre-trusted agents, regardless of their reputation values.

The method is thus capable to effectively detect malicious agents (that are identified as the worst reputed agents among the non pre-trusted agents). Unfortunately, Eigentrust generates the **side effect** to artificially modify the reputation of the other agents, which are all indiscriminately awarded. As a consequence, the differences, in terms of reliability, between honest agents are flattened. We highlight that, since the reputation values are upper bounded by the value 1, the operation of indiscriminately rewarding all the honest agents tends to increase the reputation values of these latter towards the upper bound, thus flattening the differences between those agents.

Moreover, Eigentrust needs to yield as input the information about pre-trusted agents, that in the case of collusion is not trivial to be found.

Given the observations above, in this chapter we provide the following contributions, also included in [20] [21]. First, we propose a simple strategy which allows to preliminary detect the best candidates as malicious (colluding) agents directly from the trust matrix $T = [t_{ij}]$. We use these agents as agents. Second, we introduce a different strategy to compute the agents reputation with respect to that used by Eigentrust. In other words, instead of indiscriminately rewarding pre-trusted agents, we artificially decrease the fraudulent trust values that colluding agents mutually exchange. This way, besides of detecting the malicious agents with the same effectiveness of Eigentrust, we do not modify the real reputation values of the honest agents. Finally, we introduce a measure of effectiveness in computing reputation in presence of malicious agents. The consequential metric of error is useful to quantitatively determine how much an algorithm for the identification of malicious agents modifies the correct reputation values of the honest agents. We have used such a metric to compare the effectiveness of our result with that generated by Eigentrust.

We have performed an experimental campaign of mathematical simulations on a dynamic multi-agent environment, showing that our method is more effective than Eigentrust in determining reputation values. Our method presents an error, defined as above, which is about a thousand times lower than the error produced by Eigentrust on medium-sized social networks. It is important to highlight that our experiments are exclusively devoted to improving the effectiveness of the methodology used by Eigentrust in terms of precision of the computed reputation values. In fact, we are not trying to improve the capability of Eigentrust to detect malicious agents. In particular, the robustness of a reputation system is not an absolute property, but depends on several aspects such as model characteristics, application scenario, and class of attack. Although context changes may not be performance-neutral, benchmarks on major reputation systems confirm that, in the considered context, Eigentrust is the best performing approach against collusion attacks [6]. This is also true for the presence of pre-trusted peers, although with the side effect of ranking other peers lower despite their honesty. In any case, comparison with other reputation systems is outside our scope, which is only to minimize the aforementioned Eigentrust side effect.

3.1 Related work

Intelligent software agents are largely exploited in many areas for their autonomous, adaptive, learning, proactive and social capabilities (e.g., in decision support systems [22], conversational tools [23], hazard scenarios [24], smart factories [25], neural models [26], transportation [27], control systems [28] and many others). Also Trust and Reputation Systems (TRSs) exploit agents to simulate a variety of complex, uncertain, dynamic human and agent-to-agent behaviors at different levels of detail and abstraction.

A number of studies compared the robustness of TRSRs to a variety of misbehaviors and contexts. The earliest comparisons [7, 29, 30] did not perform quantitative evaluations on strategies, weaknesses, and strengths. Later, several testbeds where reliable and unreliable actors compete with each other to gain some advantage were used. Well known is ART (Agent Reputation and Trust) [31] but other remarkable testbeds are described in [32–34]. Usually, testbeds are unable to autonomously identify the worst conditions and may lead to errors when TRSs are moved to different contexts. In contrast, mathematical or analytical approaches [35, 36] are more complex, more effective but lack of generality.

The simplicity of the eBay RS made it popular, although it is exposed to many threats [37–39]. This RS increases or decreases reputation (starting from 0) based on feedback issued by users that also have to interpret the reputation scores. Over time, this RS has been updated to improve its resilience to malicious but without changing its basic nature. PeerTrust [40] is a distributed agent transaction-based RS, over a peer-to-peer overlay network, to identify peers to collaborate. It considers direct peers' feedback, number and context of peer' transactions, and credibility of indirect feedback sources to improve resilience against threats. Also Hypertrust [41] adopts a distributed approach to discover and allocate trusted resources into competitive, large federations of utility infrastructures. Its metric considers reliability and reputation (from opinions) sources. A decentralized procedure builds an overlay network with all the federation nodes to implement an efficient finding process for computational resources.

3.1.1 Pagerank model

The early web search engines such as Altavista simply presented every web page that matched the key words entered by the user, which often resulted in too many and irrelevant pages being listed in the search results. Altavista's proposal for handling this problem was to offer advanced ways to combine keywords based on binary logic. This was too complex for users, and therefore did not provide a good solution. PageRank proposed by Page et al. [9] represents a way of ranking the best search results based on a page's score according to a specific metric. Roughly speaking, PageRank computes the score for any Web page as the sum of the normalised weights of hyperlinks pointing to it, where a normalised hyperlink weight is determined by the score of the page containing the hyperlink, divided by the total number of hyperlinks from that page. In other words, it is possible to represent the World Wide Web as a directed graph, in which Web pages are nodes and an edge from a page $P1$ to a page $P2$ means that there is an hyperlink to $P2$ on $P1$. From the description of the PageRank algorithm, we can see that the probability of going from $P1$ to $P2$ is described exactly by the formula used for computing the weights a_{P1P2} of a linear neural network. In this way, the final probabilities, i.e. the ranks of the different pages, form an eigenvector of the corresponding matrix of the weights.

This can be described as a trust system, because the total set of hyperlinks form transitive trust chains that can be used as a basis for deriving a relative trust measures for each page. A single hyperlink to a given web page can be seen as a unidirectional trust relationship between the source and the target page. Google's search engine15 is based on the PageRank algorithm, and the rapidly rising popularity of Google at the cost of Altavista was obviously caused by the superior search results that the PageRank algorithm delivered. The increasing popularity and economic importance of search engines has also lead to more damaging methods for artificially boosting the score of Web pages. Such an example is the phenomenon called *link spam* which consists of placing many hyperlinks to the same Web page in open web forums such as online discussion boards, guest books, weblogs and wikis. The motivation behind this attack is that search engines will give an increased score for the Web page that these hyperlinks point to. In order to counter the link spam attacks Google announced in early 2005 that hyperlinks marked with the attribute `rel="nofollow"` would not influence the hyperlink target's score in the search engine's index. PageRank had a significant impact on the evaluation of the quality of web pages in search engines. Moreover, it is important to note that over the years, search engine page evaluation algorithms have become more complex, incorporating a number of factors, such as content relevance, user experience, content freshness and more. However, PageRank continues to be an important component of search engine ranking algorithms.

3.1.2 Eigentrust Algorithm

The Eigentrust algorithm is a trust and reputation algorithm used in peer-to-peer (P2P) systems to assess the trustworthiness of users or peers in the network. Eigentrust is a variation of the PageRank algorithm [42]. It is based on the concept of distributed trust, where each peer assigns a trust score to the other peers it interacts with. Eigentrust can be effectively used to detect malicious agents in a social agent network. Eigentrust [43] has been designed for file sharing peer-to-peer networks. Here, each peer builds its own local trust representation about the whole peer community. All the normalized peers' trust reputations, weighted by the trustworthiness of the peer, are collected in a matrix, called global reputation. The authors show how such values asymptotically converge to the matrix eigenvalues. The malicious discrimination uses an arbitrary reputation threshold. To minimize the influence of colluding activities, some peers are assumed as trusted. This RS is susceptible for peers' feedback manipulations [44], and several changes have been introduced to increase its robustness [45–47] or adapt it to new scenarios [48, 49].

Due to its performance, Eigentrust is a popular benchmark for evaluating other TRRs [6, 50, 51]. In particular, in Eigentrust, a *trust value* t_{ij} is assigned from each agent i to each other agent j, while a *reputation* r_i is assigned from the whole community to each agent i. In this respect, the reputation of an agent i, is computed as a weighted mean of all the opinions about i provided by the other agents. Therefore, reputation has to be considered as a synthetic evaluation about the opinion that the whole community, and not only a single agent, has with regard to i.

We highlight that Eigentrust, in addition to being equivalent to linear neural networks, can be usefully applied also to make the training activity of any neural network more effective and efficient. Indeed, often a neural network has to be trained with data coming from unreliable or even malicious sources. The aforementioned case of providing rankings of scientific papers based on their importance is an example of such a type of situation. In fact, since some authors may adopt unethical behavior in using citations, for example by flattering themselves in citing each other (collusion). Eigentrust could be fruitfully exploited to identify such types of malicious sources, eliminating them from the training set or reducing their impact on the training.

However, two important issues should be highlighted to apply this algorithm, namely:

1. Eigentrust needs to receive additional information about agents to consider a-priori particularly trustworthy as inputs. The task to determine such agents is not trivial in the case of collusion.
2. The application of Eigentrust to a social agent community presents the side-effect of artificially modifying the reputation values of the non colluded agents. This is not a desired effect when reputation analysis is a main target of the system, besides of the detection of the colluded agents. Indeed, we highlight that the final goal of any reputation mechanism in presence of malicious agents is

not only to detect malicious agents. In fact, a main goal is also to determine the reputation values that the agents would assume in absence of malicious agents in the system. These values can be considered as "ideal reputation values". Eigentrust, although it is very efficient to detect malicious agents, adopts a strategy that leads to assign to the honest agents reputation values that are relatively lower than their ideal reputation values.

3.2 Effective reputation computation

In this section we present the recursive reputation model. Before developing the model, we introduce our notation.

Notation	
n	# of agents
$T = [t_{ij}]$	$n \times n$ *original trust matrix* where
	t_{ij} is the trust perceived by member j
	with respect to the member i
$\widetilde{T}$	$n \times n$ *Eigentrust matrix*
$\widehat{T}$	$n \times n$ *matrix obtained with our strategy*
r	$n \times 1$ *reputation vector*
v	$n \times 1$ *teleportation vector*
u	$n \times 1$ vector of ones
d	$n \times 1$ row-sums vector of T
$\mathcal{P}$	Set of *pre-trusted agents*
$\mathcal{F}$	Set of *potential colluded agents*
$\mathcal{M}$	Set of *malicious agents*

Table 3.1: Notation

3.2.1 The Recursive Reputation Model

In our scenario, we suppose to have a social network composed by n members, each member being uniquely identified by an integer i, $1 \leq i \leq n$.

The *reputation* r_i of each member i of the social network can be seen as the sum of all the trust values t_{ij}, $j = 1, \ldots, n$, corresponding to a trustor j, weighted by the reputation r_j of j. Without loss of generality, we will consider evaluations over the interval $[0,1]$, i.e. $T = [t_{ij}] \in [0,1]^{n \times n}$ and the reputation vector $r \in [0,1]^n$. Moreover, we assume that each member j may distribute an overall sum of the trust values t_{ij} equal to 1, namely $\sum_{i=1}^{n} t_{ij} = 1$, then the matrix T will be column-stochastic.

The reputation r_i can also be viewed as the barycenter of all the trust values expressed for i by the trustors. Formally:

$$r_i = \frac{\sum_{j=1}^{n} t_{ij}\, r_j}{\sum_{j=1}^{n} r_j}, \quad i = 1, \dots, n. \tag{3.1}$$

It is worthwhile to remark that it can be seen as the transpose of the weighted adjacency matrix $A = [a_{ij}]$ of a directed graph $\mathcal{G}$ where each node is associated with a user and a non negative value a_{ij} is assigned to the edge (i, j), representing the trust score assigned by the member i to the member j.

The *reputation vector* $r = (r_1, \dots, r_n)^T$, whose elements satisfy (6.2), can be computed as solution to the system of linear equations:

$$Tr = r \tag{3.2}$$

Since the matrix T is column-stochastic (equivalently it is a transition or a Markov matrix), as it is well-known, if we further assume that its entries are positive, the solution to (4.2) is guaranteed by the Perron-Frobenius theorem. In particular, it follows that $\lambda = 1 = \rho(T)$ is a simple eigenvalue of T (the other ones being in modulus less than 1), and there exists a corresponding unique eigenvector $r \in \mathbb{R}^n$, $\|r\|_1 = 1$, that is a unique positive reputation vector whose elements sum up to 1.

This vector (also called the steady-state vector) represents the stable equilibrium distribution of the Markov chain represented by the transition matrix T.

A modified version of the eigensystem (4.2) is represented by the well-known *PageRank model*:

$$\widetilde{T} r = r \tag{3.3}$$

where

$$\widetilde{T} = \alpha T + (1 - \alpha) v u^T$$

represents the new trust matrix, $0 \leq \alpha \leq 1$, u is the vector of all ones, v is a non-negative vector with unitary 1-norm, that is $u^T v = 1$.

The vector v is usually called *teleportation vector*.

PageRank's original algorithm adopts the uniform choice $v = \frac{1}{n} u$. In such a case, when $\alpha \neq 0, 1$, the existence and uniqueness of the solution to (3.3) are guaranteed.

In the 2003 paper by Kamvar et al. [43] a modified model is proposed, known as *Eigentrust algorithm*. Its *basic* version consists in solving the same system as in (3.3), by choosing v in a way useful to mitigate the final reputation of malicious users. In particular, by denoting with $\mathcal{P}$ the set of the so-called *pre-trusted* agents, i.e. agents that can be a-priori considered particularly trustworthy, the entry v_i of the vector v is equal to $1/|\mathcal{P}|$, if the i-th user belongs to $\mathcal{P}$, 0 otherwise.

In this approach, the pre-trusted users are known a priori. They coincide, for example, with the first few members to join the network.

The detection of potential malicious users, on the other hand, could be better exploited for a more realistic computation of the reputation vector. In the next subsection, we propose a strategy that allows to “read" such information straight from the matrix T.

3.2.2 Detection of potential malicious users

We aim at investigating collusion in social networks, so we first provide a definition of malicious agents and, then, a threshold strategy for concretely detecting them.

We stress that by means of this approach we detect the best candidates as malicious agents directly from the matrix T.

Once the malicious are identified, we can use this information in two ways: either considering the non-colluding agents as pre-trusted in the Eigentrust model (3.3) or employing an alternative algorithm which specifically makes use of such data.

We classify a pair of agents i, j as *malicious* if they *collude*, i.e. when the two conditions listed below are verified at the same time:

- the trusts t_{ij} and t_{ji} take high values (and they are similar to each other);
- the sum of the residual trust scores belonging to the rows i and j takes a low value.

We thus check in the matrix T for high values t_{ij} corresponding to high values t_{ji}, associated with users which receive low values from the most part of the members.

We observe that the collusion relationship is not transitive, since the collusion between two agents A and B and the collusion between the agents B and C do not necessarily imply a collusion between A and C, and the transitive property is not requested by our model.

For a proper mathematical formulation of the above procedure, we introduce the vector d of the node out-degrees of the graph $\mathcal{G}$, or, equivalently, the vector of the row-sums of the matrix T, i.e. $d = Tu$. We furthermore fix two threshold values δ_i, $i = 1, 2$.

We first identify the quasi-symmetric high-valued elements in the matrix T by constructing an auxiliary matrix C with elements:

$$c_{ij} = \begin{cases} t_{ij}, & \text{if } t_{ij} \geq \delta_1 \\ 0 & \text{otherwise} \end{cases} . \tag{3.4}$$

It corresponds to the weighted adjacency matrix of the (undirected) sub-graph of $\mathcal{G}$ with edges connecting potential colluded agents.

We then construct the vector $\widetilde{d} = d - Cu$ and classify an agent i, for each $i = 1, \dots, n$, as malicious if:

$$\widetilde{d_i} \leq \delta_2.$$

A proper selection of the threshold values δ_1 and δ_2 is crucial for achieving an accurate detection of the malicious agents.

The parameter δ_1 must be chosen sufficiently large to identify all possible candidates (agents which assign high trust scores to each other). So, in order not to exclude any of them, we fix it as:

$$\delta_1 = \min_{1 \leq j \leq n} \max_{1 \leq i \leq n} t_{ij}.$$

As to the threshold δ_2, in order to exclude from the first selection honest agents which receive anyway high trust values from the rest of the honest agents, we set it as:

$$\delta_2 = \frac{\sum_{i,j \in \mathcal{F}} (t_{ij} - c_{ij})}{|\mathcal{F}|}$$

where $\mathcal{F}$ is the index set of the potential colluded agents identified after the thresholding step (3.4) and c_{ij} are the elements of the matrix C. Such information can then be used to set the proper initial reputation vector in the Eigentrust model. Indeed, let $\mathcal{M}$ be the index set of the malicious users identified as above. The teleportation vector v has then elements v_i which are set to 0 if $i \in \mathcal{M}$, and to $1/(n - |\mathcal{M}|)$ otherwise.

It has to be noted that, in such a way, we assign the same amount of trust to the non-malicious agents, irrespective of their initial actual trust values.

3.2.3 Our algorithm to compute the reputation

The strategy of Eigentrust is to penalize all the agents that are not pre-trusted (i.e., all the potential colluding agents), with the side-effect of computing an incorrect final reputation for those agents that are not in the pre-trusted set but also are not colluded. With the aim to avoid the above strong penalization of all the agents that are not pre-trusted, we propose an alternative approach to Eigentrust. The idea is to pre-process the set of the agents to determine a sub-set of suitable candidates to be considered as colluded, and then assign a low value of trustworthiness only to these "suspects", rather than to all the agents that are not in the pre-trusted set.

In particular, once the malicious are identified, we modify the matrix T into a new matrix $\widehat{T}$ as follows:

- Fix a value $\epsilon > 0$;
- Let $\widehat{t_{ij}} = \begin{cases} \epsilon, & i, j \in \mathcal{M} \\ t_{ij}, & \text{otherwise} \end{cases}$
- Make the matrix $\widehat{T}$ column stochastic by normalizing to 1 each column.

In the choice of ϵ the size of the matrix has to be taken into account, so to guarantee that it keeps a low value as n increases. This is achieved, for example, by setting $\epsilon = \beta/n$, with $\beta \leq 1$.

In our experiments, for computational purpose, we adopt the choice $\beta = 2 \cdot 10^{-3}$.

For the sake of completeness, the scheme describing the entire procedure (including the steps for the detection of malicious) is illustrated in Algorithm 1.

Numerical and experimental examples, as shown later, confirm that, in addition to detect colluding agents, this strategy allows to keep the reputations of all the honest agents unchanged.

3.3 An illustrative example

We propose an example to illustrate our approach and to compare it to the Eigentrust approach in the particular situation of a social network of 14 agents. We consider the trust matrix T given in Table 1. For the choice of α, we follow the traditional Eigentrust approach, where the value is fixed to 0.85.

Algorithm 1 Algorithm for malicious detection and reputation computation

```
Require: $T$, $\beta$, $\alpha$
 1: $v = [0,0,\dots,0]^T$
 2: $d \leftarrow T[1,1,\dots,1]^T$
 3: $\delta_1 \leftarrow \min\{\max\{t_{ij}, i = 1,\dots,n\}, j = 1,\dots,n\}$
 4: $\mathcal{F} = \emptyset$
 5: for $i = 1$ to $n$ do
 6:   for $j = 1$ to $n$ do
 7:     if $t_{ij} \geq \delta_1$ then
 8:       $c_{ij} \leftarrow t_{ij}$
 9:       $\mathcal{F} \leftarrow \mathcal{F} \cup \{i\}$ ($i$ added only once)
10:     else
11:       $c_{ij} \leftarrow 0$
12:     end if
13:   end for
14: end for
15: $d \leftarrow d - C[1,1,\dots,1]^T$
16: $\delta_2 \leftarrow \sum_{i,j} |t_{ij} - c_{ij}| / |\mathcal{F}|$
17: for $i = 1$ to $n$ do
18:   if $d_i > \delta_2$ then
19:     $v_i = 1$
20:     $\mathcal{M} \leftarrow \mathcal{M} \cup \{i\}$
21:   end if
22: end for
23: $v \leftarrow v / \sum_i v_i$
24: for $i, j = 1$ to $n$ do
25:   if $i, j \in \mathcal{M}$ then
26:     $t_{ij} \leftarrow \beta/n$
27:   end if
28: end for
29: normalize $T$
30: solve $(\alpha T + (1-\alpha)v[1,1,\dots,1]^T)r = r$
31: return $r$
```

The suspected malicious users are the ones corresponding to the agents 8, 9 and 10. Moreover, the members 1, 2 and 3 are good agents and receive high values of trust from all honest agents, whereas the remaining honest members receive only low trust values from all other agents.

We assume that $t_{ii} = 0$, $i = 1,\dots,n$, so that the trust self assigned by a member is not considered.

Table 1 Matrices used for our example in Section 4

$$T = \begin{pmatrix}
0 & 0.4 & 0.39 & 0.33 & 0.26 & 0.23 & 0.2 & 0.001 & 0.002 & 0.003 & 0.25 & 0.19 & 0.23 & 0.24 \\
0.36 & 0 & 0.43 & 0.28 & 0.3 & 0.189 & 0.22 & 0.002 & 0.003 & 0.002 & 0.22 & 0.2 & 0.25 & 0.223 \\
0.42 & 0.41 & 0 & 0.19 & 0.28 & 0.195 & 0.23 & 0.001 & 0.005 & 0.001 & 0.23 & 0.21 & 0.196 & 0.196 \\
0.08 & 0.0351 & 0.07 & 0 & 0.1 & 0.14 & 0.11 & 0.003 & 0.001 & 0.009 & 0.1 & 0.095 & 0.12 & 0.09 \\
0.04 & 0.07 & 0.06 & 0.1 & 0 & 0.09 & 0.091 & 0.006 & 0.003 & 0.008 & 0.09 & 0.08 & 0.093 & 0.086 \\
0.02 & 0.019 & 0.01 & 0.06 & 0.02 & 0 & 0.1 & 0.009 & 0.01 & 0.006 & 0.08 & 0.1 & 0.055 & 0.076 \\
0.025 & 0.015 & 0.003 & 0.02 & 0.03 & 0.125 & 0 & 0.019 & 0.03 & 0.006 & 0.01 & 0.11 & 0.035 & 0.066 \\
0.01 & 0.0049 & 0.006 & 0.002 & 0.003 & 0.006 & 0.009 & 0 & 0.46 & 0.46 & 0.002 & 0.002 & 0.006 & 0.009 \\
0.009 & 0.006 & 0.003 & 0.006 & 0.001 & 0.002 & 0.001 & 0.42 & 0 & 0.42 & 0.003 & 0.001 & 0.005 & 0.003 \\
0.015 & 0.019 & 0.008 & 0.001 & 0.002 & 0.009 & 0.002 & 0.48 & 0.43 & 0 & 0.008 & 0.002 & 0.003 & 0.005 \\
0.009 & 0.009 & 0.005 & 0.003 & 0.001 & 0.008 & 0.006 & 0.01 & 0.009 & 0.001 & 0 & 0.002 & 0.002 & 0.002 \\
0.002 & 0.003 & 0.005 & 0.005 & 0.001 & 0.001 & 0.009 & 0.02 & 0.01 & 0.06 & 0.003 & 0 & 0.003 & 0.001 \\
0.004 & 0.008 & 0.001 & 0.002 & 0.001 & 0.002 & 0.01 & 0.009 & 0.02 & 0.02 & 0.003 & 0.004 & 0 & 0.003 \\
0.006 & 0.001 & 0.009 & 0.001 & 0.001 & 0.003 & 0.012 & 0.02 & 0.017 & 0.004 & 0.001 & 0.004 & 0.002 & 0
\end{pmatrix}$$

$$\widetilde{T} = \begin{pmatrix}
0.1925 & 0.2725 & 0.2705 & 0.2585 & 0.2445 & 0.2385 & 0.2325 & 0.1927 & 0.1929 & 0.1931 & 0.2425 & 0.2305 & 0.2385 & 0.2405 \\
0.2612 & 0.1892 & 0.2752 & 0.2452 & 0.2492 & 0.2270 & 0.2332 & 0.1896 & 0.1898 & 0.1896 & 0.2332 & 0.2292 & 0.2392 & 0.2338 \\
0.2650 & 0.2630 & 0.1810 & 0.2190 & 0.2370 & 0.2200 & 0.2270 & 0.1812 & 0.1820 & 0.1812 & 0.2270 & 0.2230 & 0.2202 & 0.2202 \\
0.0833 & 0.0743 & 0.0813 & 0.0673 & 0.0873 & 0.0953 & 0.0893 & 0.0679 & 0.0675 & 0.0691 & 0.0873 & 0.0863 & 0.0913 & 0.0853 \\
0.0657 & 0.0717 & 0.0697 & 0.0777 & 0.0577 & 0.0757 & 0.0759 & 0.0589 & 0.0583 & 0.0593 & 0.0757 & 0.0737 & 0.0763 & 0.0749 \\
0.0439 & 0.0437 & 0.0419 & 0.0519 & 0.0439 & 0.0399 & 0.0599 & 0.0417 & 0.0419 & 0.0411 & 0.0559 & 0.0599 & 0.0509 & 0.0551 \\
0.0399 & 0.0379 & 0.0355 & 0.0389 & 0.0409 & 0.0599 & 0.0349 & 0.0387 & 0.0409 & 0.0361 & 0.0369 & 0.0569 & 0.0419 & 0.0481 \\
0.0062 & 0.0052 & 0.0054 & 0.0046 & 0.0048 & 0.0054 & 0.0060 & 0.0042 & 0.0962 & 0.0962 & 0.0046 & 0.0046 & 0.0054 & 0.0060 \\
0.0046 & 0.0040 & 0.0034 & 0.0040 & 0.0030 & 0.0032 & 0.0030 & 0.0868 & 0.0028 & 0.0868 & 0.0034 & 0.0030 & 0.0038 & 0.0034 \\
0.0082 & 0.0090 & 0.0068 & 0.0054 & 0.0056 & 0.0070 & 0.0056 & 0.1012 & 0.0912 & 0.0052 & 0.0068 & 0.0056 & 0.0058 & 0.0062 \\
0.0065 & 0.0065 & 0.0057 & 0.0053 & 0.0049 & 0.0063 & 0.0059 & 0.0067 & 0.0065 & 0.0049 & 0.0047 & 0.0051 & 0.0051 & 0.0051 \\
0.0091 & 0.0093 & 0.0097 & 0.0097 & 0.0089 & 0.0089 & 0.0105 & 0.0127 & 0.0107 & 0.0207 & 0.0093 & 0.0087 & 0.0093 & 0.0089 \\
0.0069 & 0.0077 & 0.0063 & 0.0065 & 0.0063 & 0.0065 & 0.0081 & 0.0079 & 0.0101 & 0.0101 & 0.0067 & 0.0069 & 0.0061 & 0.0067 \\
0.0069 & 0.0059 & 0.0075 & 0.0059 & 0.0059 & 0.0063 & 0.0081 & 0.0097 & 0.0091 & 0.0065 & 0.0059 & 0.0065 & 0.0061 & 0.0057
\end{pmatrix}$$

$$\widehat{T} = \begin{pmatrix}
0 & 0.4000 & 0.3900 & 0.3300 & 0.2600 & 0.2300 & 0.2000 & 0.0100 & 0.0182 & 0.0250 & 0.2500 & 0.1900 & 0.2300 & 0.2400 \\
0.3600 & 0 & 0.4300 & 0.2800 & 0.3000 & 0.1890 & 0.2200 & 0.0200 & 0.0273 & 0.0167 & 0.2200 & 0.2000 & 0.2500 & 0.2230 \\
0.4200 & 0.4100 & 0 & 0.1900 & 0.2800 & 0.1950 & 0.2300 & 0.0100 & 0.0454 & 0.0083 & 0.2300 & 0.2100 & 0.1960 & 0.1960 \\
0.0800 & 0.0351 & 0.0700 & 0 & 0.1000 & 0.1400 & 0.1100 & 0.0300 & 0.0091 & 0.0750 & 0.1000 & 0.0950 & 0.1200 & 0.0900 \\
0.0400 & 0.0700 & 0.0600 & 0.1000 & 0 & 0.0900 & 0.0910 & 0.0600 & 0.0273 & 0.0667 & 0.0900 & 0.0800 & 0.0930 & 0.0860 \\
0.0200 & 0.0190 & 0.0100 & 0.0600 & 0.0200 & 0 & 0.1000 & 0.0900 & 0.0909 & 0.0500 & 0.0800 & 0.1000 & 0.0550 & 0.0760 \\
0.0250 & 0.0150 & 0.0030 & 0.0200 & 0.0300 & 0.1250 & 0 & 0.1900 & 0.2727 & 0.0500 & 0.0100 & 0.1100 & 0.0350 & 0.0660 \\
0.0100 & 0.0049 & 0.0060 & 0.0020 & 0.0030 & 0.0060 & 0.0090 & 0 & 0.0001 & 0.0001 & 0.0020 & 0.0020 & 0.0060 & 0.0090 \\
0.0090 & 0.0060 & 0.0030 & 0.0060 & 0.0010 & 0.0020 & 0.0010 & 0.0001 & 0 & 0.0001 & 0.0030 & 0.0010 & 0.0050 & 0.0030 \\
0.0150 & 0.0190 & 0.0080 & 0.0010 & 0.0020 & 0.0090 & 0.0020 & 0.0001 & 0.0001 & 0 & 0.0080 & 0.0020 & 0.0030 & 0.0050 \\
0.0090 & 0.0090 & 0.0050 & 0.0030 & 0.0010 & 0.0080 & 0.0060 & 0.1000 & 0.0818 & 0.0083 & 0 & 0.0020 & 0.0020 & 0.0020 \\
0.0020 & 0.0030 & 0.0050 & 0.0050 & 0.0010 & 0.0010 & 0.0090 & 0.2000 & 0.0909 & 0.4999 & 0.0030 & 0 & 0.0030 & 0.0010 \\
0.0040 & 0.0080 & 0.0010 & 0.0020 & 0.0010 & 0.0020 & 0.0100 & 0.0900 & 0.1818 & 0.1666 & 0.0030 & 0.0040 & 0 & 0.0030 \\
0.0060 & 0.0010 & 0.0090 & 0.0010 & 0.0010 & 0.0030 & 0.0120 & 0.2000 & 0.1545 & 0.0333 & 0.0010 & 0.0040 & 0.0020 & 0
\end{pmatrix}$$

In our example we fix $\delta_1 = 0.21$ and $\delta_2 = 0.96$.

Applying our threshold strategy, the agents 8, 9 and 10 are actually considered as the only malicious agents, so that $\mathcal{M} = \{8, 9, 10\}$.

Following the Eigentrust model, the teleportation vector is

$$v = \frac{1}{11}(1,1,1,1,1,1,1,0,0,0,1,1,1,1)^T$$

and the trust matrix $\widetilde{T}$, that is used in (3.3), is given in Table 1.

To construct the matrix $\widehat{T}$ following our new method, we fix $\epsilon = 0.00014$, taking into account the size of matrix T. For $i, j \in \mathcal{M} = \{8, 9, 10\}$, let $\widehat{t_{ij}} = \epsilon$ and by normalizing to 1 each column we get the column stochastic matrix $\widehat{T}$ in Table 1.

In Fig. 3.1 we illustrate, for comparison, the original trust matrix T and the $\widetilde{T}$, $\widehat{T}$ matrices obtained with the two different methods, where trust values are differentiated by color tone. It is evident that in the original trust matrix the rows 8, 9 and 10 exhibit unusual high values of trust and are classified as malicious users. We highlight that, unlike the Eigentrust matrix, the trust values of the honest agents remain unchanged in the matrix obtained by applying our method.

The respective corresponding reputation vectors, obtained by solving the eigensystem (4.2) with $\widetilde{T}$, $\widehat{T}$ matrices are plotted in Fig. 3.2. We note that the method we propose does not modify the real reputation scores of the honest agents unlike Eigentrust.

3.4 Error measure

To compare the results obtained by using different methods, we propose an error metric to measure how each method affects the reputation of the non colluded (pre-trusted) agents (once they are identified with our strategy) while lowering the reputation of malicious ones.

The idea is to evaluate the distance between the reputation values associated only with the pre-trusted users obtained with each algorithm and the solution to (4.2) once in the original matrix T the trust values associated with malicious agents are deleted.

More formally, let $\mathcal{P}$ be the set of indexes associated to the pre-trusted users, obtained as $\mathcal{P} = \{1, 2, \ldots, n\} \backslash \mathcal{M}$ and let $p = |\mathcal{P}|$. Let E be the $n \times p$ matrix whose columns are the standard unit coordinate (column) vectors $e^j = [0, \ldots, 1, \ldots, 0]^T$, $j \in \mathcal{P}$, with the only nonzero coefficient 1 at the j-th entry. Premultiplication by the transpose of such a matrix allows to extract only the rows associated to the pre-trusted-users, while postmoltiplication by E perfoms a similar extraction on the columns.

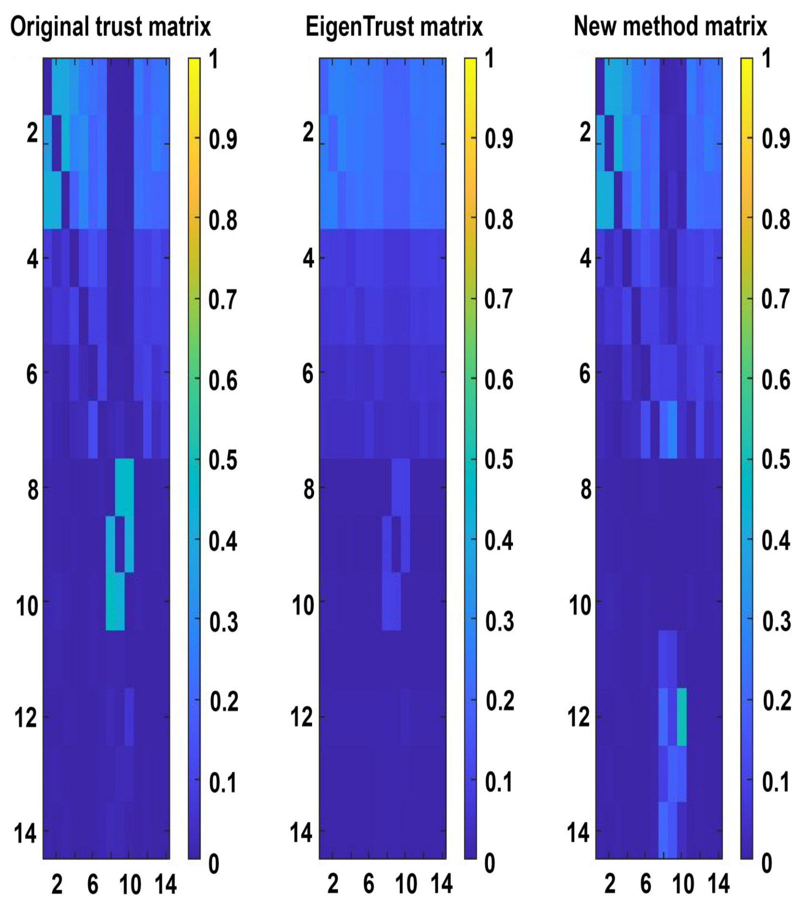

Fig. 3.1: T, $\widetilde{T}$, $\widehat{T}$ matrices corresponding to the Example in Section 4.3.

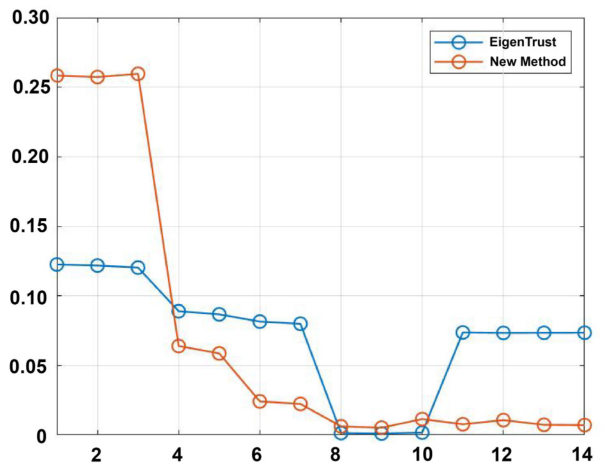

Fig. 3.2: Reputation vectors obtained with the two different approaches.

Table 2 Errors with Eigentrust and our method, as functions of the number of agents and the percentage of malicious users

n	% malicious	Eigentrust		Our method	
		e_2	e_∞	e_2	e_∞
50	10%	$6.2 \cdot 10^{-1}$	$1.3 \cdot 10^{-2}$	$1.2 \cdot 10^{-3}$	$2.3 \cdot 10^{-3}$
	20%	$7.2 \cdot 10^{-2}$	$1.4 \cdot 10^{-1}$	$2.1 \cdot 10^{-3}$	$4.0 \cdot 10^{-3}$
	30%	$8.1 \cdot 10^{-2}$	$1.5 \cdot 10^{-1}$	$3.2 \cdot 10^{-3}$	$7.1 \cdot 10^{-3}$
	40%	$8.1 \cdot 10^{-2}$	$1.5 \cdot 10^{-1}$	$4.8 \cdot 10^{-3}$	$8.8 \cdot 10^{-3}$
	50%	$8.9 \cdot 10^{-2}$	$1.6 \cdot 10^{-1}$	$6.9 \cdot 10^{-3}$	$1.2 \cdot 10^{-2}$
100	10%	$4.8 \cdot 10^{-2}$	$1.1 \cdot 10^{-1}$	$4.2 \cdot 10^{-4}$	$9.5 \cdot 10^{-4}$
	20%	$4.9 \cdot 10^{-2}$	$1.1 \cdot 10^{-1}$	$6.7 \cdot 10^{-4}$	$1.5 \cdot 10^{-3}$
	30%	$5.5 \cdot 10^{-2}$	$1.2 \cdot 10^{-1}$	$1.0 \cdot 10^{-3}$	$2.2 \cdot 10^{-3}$
	40%	$5.9 \cdot 10^{-2}$	$1.2 \cdot 10^{-1}$	$1.6 \cdot 10^{-3}$	$3.2 \cdot 10^{-3}$
	50%	$6.4 \cdot 10^{-2}$	$1.3 \cdot 10^{-1}$	$2.3 \cdot 10^{-3}$	$4.2 \cdot 10^{-2}$
200	10%	$3.3 \cdot 10^{-2}$	$9.1 \cdot 10^{-2}$	$1.5 \cdot 10^{-4}$	$3.8 \cdot 10^{-4}$
	20%	$3.4 \cdot 10^{-2}$	$9.5 \cdot 10^{-2}$	$2.6 \cdot 10^{-4}$	$7.9 \cdot 10^{-4}$
	30%	$3.8 \cdot 10^{-2}$	$9.6 \cdot 10^{-2}$	$3.8 \cdot 10^{-4}$	$1.1 \cdot 10^{-3}$
	40%	$4.2 \cdot 10^{-2}$	$1.0 \cdot 10^{-1}$	$5.5 \cdot 10^{-4}$	$1.3 \cdot 10^{-3}$
	50%	$4.5 \cdot 10^{-2}$	$1.1 \cdot 10^{-1}$	$8.0 \cdot 10^{-4}$	$1.8 \cdot 10^{-3}$
300	10%	$2.7 \cdot 10^{-2}$	$7.5 \cdot 10^{-2}$	$8.4 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
	20%	$2.9 \cdot 10^{-2}$	$7.9 \cdot 10^{-2}$	$1.4 \cdot 10^{-4}$	$3.6 \cdot 10^{-4}$
	30%	$3.0 \cdot 10^{-2}$	$8.6 \cdot 10^{-2}$	$1.9 \cdot 10^{-4}$	$5.5 \cdot 10^{-4}$
	40%	$3.5 \cdot 10^{-2}$	$9.9 \cdot 10^{-2}$	$3.0 \cdot 10^{-4}$	$8.7 \cdot 10^{-4}$
	50%	$3.8 \cdot 10^{-2}$	$1.0 \cdot 10^{-1}$	$4.2 \cdot 10^{-4}$	$1.1 \cdot 10^{-3}$
400	10%	$2.4 \cdot 10^{-2}$	$6.8 \cdot 10^{-2}$	$5.3 \cdot 10^{-5}$	$1.6 \cdot 10^{-4}$
	20%	$2.5 \cdot 10^{-2}$	$7.0 \cdot 10^{-2}$	$9.0 \cdot 10^{-5}$	$2.5 \cdot 10^{-4}$
	30%	$2.6 \cdot 10^{-2}$	$7.2 \cdot 10^{-2}$	$1.2 \cdot 10^{-4}$	$3.7 \cdot 10^{-4}$
	40%	$3.0 \cdot 10^{-2}$	$8.5 \cdot 10^{-2}$	$2.0 \cdot 10^{-4}$	$5.2 \cdot 10^{-4}$
	50%	$3.3 \cdot 10^{-2}$	$8.9 \cdot 10^{-2}$	$2.8 \cdot 10^{-4}$	$7.2 \cdot 10^{-4}$
500	10%	$2.1 \cdot 10^{-2}$	$6.9 \cdot 10^{-2}$	$3.7 \cdot 10^{-5}$	$1.1 \cdot 10^{-4}$
	20%	$2.3 \cdot 10^{-2}$	$7.1 \cdot 10^{-2}$	$6.6 \cdot 10^{-5}$	$1.8 \cdot 10^{-4}$
	30%	$2.3 \cdot 10^{-2}$	$7.2 \cdot 10^{-2}$	$9.5 \cdot 10^{-5}$	$2.6 \cdot 10^{-4}$
	40%	$2.6 \cdot 10^{-2}$	$7.4 \cdot 10^{-2}$	$1.3 \cdot 10^{-4}$	$3.9 \cdot 10^{-4}$
	50%	$2.9 \cdot 10^{-2}$	$8.0 \cdot 10^{-2}$	$2.1 \cdot 10^{-4}$	$6.1 \cdot 10^{-4}$
1000	10%	$1.5 \cdot 10^{-2}$	$4.9 \cdot 10^{-2}$	$1.3 \cdot 10^{-5}$	$4.2 \cdot 10^{-5}$
	20%	$1.6 \cdot 10^{-2}$	$5.3 \cdot 10^{-2}$	$2.2 \cdot 10^{-5}$	$7.4 \cdot 10^{-5}$
	30%	$1.7 \cdot 10^{-2}$	$6.0 \cdot 10^{-2}$	$3.4 \cdot 10^{-5}$	$1.1 \cdot 10^{-4}$
	40%	$1.8 \cdot 10^{-2}$	$5.7 \cdot 10^{-2}$	$4.8 \cdot 10^{-5}$	$1.4 \cdot 10^{-4}$
	50%	$2.0 \cdot 10^{-2}$	$6.8 \cdot 10^{-2}$	$7.3 \cdot 10^{-5}$	$2.1 \cdot 10^{-4}$
2500	10%	$0.9 \cdot 10^{-2}$	$3.4 \cdot 10^{-2}$	$3.4 \cdot 10^{-6}$	$1.2 \cdot 10^{-5}$
	20%	$1.0 \cdot 10^{-2}$	$3.4 \cdot 10^{-2}$	$5.7 \cdot 10^{-6}$	$2.1 \cdot 10^{-5}$
	30%	$1.0 \cdot 10^{-2}$	$3.6 \cdot 10^{-2}$	$8.6 \cdot 10^{-6}$	$2.8 \cdot 10^{-5}$
	40%	$1.2 \cdot 10^{-2}$	$4.0 \cdot 10^{-2}$	$1.2 \cdot 10^{-5}$	$4.1 \cdot 10^{-5}$
	50%	$1.3 \cdot 10^{-2}$	$4.4 \cdot 10^{-2}$	$1.8 \cdot 10^{-5}$	$5.9 \cdot 10^{-5}$
5000	10%	$6.9 \cdot 10^{-3}$	$2.5 \cdot 10^{-2}$	$1.2 \cdot 10^{-6}$	$4.5 \cdot 10^{-6}$
	20%	$7.3 \cdot 10^{-3}$	$2.7 \cdot 10^{-2}$	$2.0 \cdot 10^{-6}$	$7.7 \cdot 10^{-6}$
	30%	$7.8 \cdot 10^{-3}$	$2.8 \cdot 10^{-2}$	$3.0 \cdot 10^{-6}$	$1.2 \cdot 10^{-5}$
	40%	$8.4 \cdot 10^{-3}$	$2.8 \cdot 10^{-2}$	$4.4 \cdot 10^{-6}$	$1.8 \cdot 10^{-5}$
	50%	$9.2 \cdot 10^{-3}$	$3.2 \cdot 10^{-2}$	$6.5 \cdot 10^{-6}$	$2.1 \cdot 10^{-5}$
7500	10%	$5.7 \cdot 10^{-3}$	$2.2 \cdot 10^{-2}$	$6.5 \cdot 10^{-7}$	$2.3 \cdot 10^{-6}$
	20%	$6.0 \cdot 10^{-3}$	$2.2 \cdot 10^{-2}$	$1.1 \cdot 10^{-6}$	$4.2 \cdot 10^{-6}$
	30%	$6.4 \cdot 10^{-3}$	$2.4 \cdot 10^{-2}$	$1.6 \cdot 10^{-6}$	$6.5 \cdot 10^{-6}$
	40%	$6.9 \cdot 10^{-3}$	$2.5 \cdot 10^{-2}$	$2.4 \cdot 10^{-6}$	$9.3 \cdot 10^{-6}$
	50%	$7.5 \cdot 10^{-3}$	$2.5 \cdot 10^{-2}$	$3.5 \cdot 10^{-6}$	$1.3 \cdot 10^{-5}$
10000	10%	$4.9 \cdot 10^{-3}$	$1.9 \cdot 10^{-2}$	$4.2 \cdot 10^{-7}$	$1.7 \cdot 10^{-6}$
	20%	$5.1 \cdot 10^{-3}$	$1.9 \cdot 10^{-2}$	$7.1 \cdot 10^{-7}$	$2.5 \cdot 10^{-6}$
	30%	$5.5 \cdot 10^{-3}$	$1.9 \cdot 10^{-2}$	$1.0 \cdot 10^{-6}$	$3.5 \cdot 10^{-6}$
	40%	$6.0 \cdot 10^{-3}$	$2.1 \cdot 10^{-2}$	$1.5 \cdot 10^{-6}$	$5.5 \cdot 10^{-6}$
	50%	$6.5 \cdot 10^{-3}$	$2.4 \cdot 10^{-2}$	$2.3 \cdot 10^{-6}$	$8.2 \cdot 10^{-6}$
25000	10%	$3.8 \cdot 10^{-3}$	$1.5 \cdot 10^{-2}$	$1.1 \cdot 10^{-7}$	$4.2 \cdot 10^{-7}$
	20%	$4.0 \cdot 10^{-3}$	$1.5 \cdot 10^{-2}$	$1.9 \cdot 10^{-7}$	$5.9 \cdot 10^{-7}$
	30%	$4.3 \cdot 10^{-3}$	$1.6 \cdot 10^{-2}$	$3.0 \cdot 10^{-7}$	$7.8 \cdot 10^{-7}$
	40%	$4.7 \cdot 10^{-3}$	$1.9 \cdot 10^{-2}$	$4.4 \cdot 10^{-7}$	$9.1 \cdot 10^{-7}$
	50%	$5.1 \cdot 10^{-3}$	$1.9 \cdot 10^{-2}$	$5.8 \cdot 10^{-7}$	$2.1 \cdot 10^{-6}$

Suppose we have found a solution to (4.2), where T is replaced either by the matrix $\widetilde{T}$ as in (3.3) in the case of the Eigentrust method, with teleportation vector $v = \frac{1}{n}u$, or the matrix $\widehat{T}$ as in our strategy.

Once the reputation vector r has been computed by one of the two methods, we extract only the values associated to the pre-trusted users, that is we compute $\widehat{r} = E^T \widehat{r}$.

We now use the original trust matrix T and solve the system

$$E^T T E \widetilde{r} = \widetilde{r},$$

which is associated to the "ideal" situation, where only the pre-trusted agents are taken into account.

After normalizing the vectors $\widehat{r}$ and $\widetilde{r}$, we evaluate the error both as

$$e_2 = \frac{\|\widetilde{r} - \widehat{r}\|_2}{\|\widetilde{r}\|_2} = \sqrt{\frac{\sum_{j=1}^{n} (\widetilde{r}_j - \widehat{r}_j)^2}{\sum_{j=1}^{n} \widetilde{r}_j^2}} \tag{3.5}$$

and

$$e_\infty = \frac{\|\widetilde{r} - \widehat{r}\|_\infty}{\|\widetilde{r}\|_\infty} = \frac{\max\limits_{1 \le j \le n} |\widetilde{r}_j - \widehat{r}_j|}{\max\limits_{1 \le j \le n} \widetilde{r}_j} \tag{3.6}$$

In Section 4.4 we perform several mathematical simulations, illustrating the results obtained for different values of the matrix size and different ratios of malicious users in the network. We make a quantitative analysis of the results by means of the error measures (3.5) and (3.6).

3.5 Experimental Results

We present the results of an experimental campaign of simulations for testing our method, comparing it with the Eigentrust algorithm.

We stress that we obtain the information on the malicious users straight from the matrix T in the pre-processing phase in both cases. In fact, the information on the pre-trusted agents needed by Eigentrust relates to the agents that are not identified as malicious in such a phase. In other words, in order to make the comparison between the two approaches meaningful, we use the same information both in Eigentrust and in our algorithm, i.e. we inform Eigentrust that the agents which are not identified as malicious could be considered honest, and then pre-trustable.

Our numerical code has been implemented using MATLAB Release 2022a following the steps summarized in our Algorithm 1. Our numerical experiments were run on a 64-bit workstation with a Intel(R) Core(TM) i7-10875H CPU @ 2.30 GHz and 128 GB of RAM.

The major computational load in the entire procedure corresponds to the computation of the solution to the system of linear equations (line 30). Our code uses the MATLAB function `eig` to solve it by finding the eigenvector corresponding to the eigenvalue 1, with a cost corresponding to $O(n^3)$. The running time of the larger experiment (corresponding to $n = 25.000$) was equal to 1821 seconds.

Our algorithm, if implemented by a centralized architecture, generates running times that are reasonable for small networks (10.000-50.000 nodes). We only refer to a centralized architecture, where a recomputation of the reputation values is needed whenever some users are added or removed from the system. Generally, on large-scale communities centralized architectures [10] are more expensive than distributed ones in terms of costs for data storage, retrieval tasks, computational and communication overheads [25]. In contrast, distributed architectures [40, 52, 53] may perform better than centralized approaches in terms of computational costs, although they are more complex to implement and more vulnerable to malicious attacks. It is of course possible to implement the algorithm in a distributed architecture fashion, to improve efficiency.

We considered different cases, for different values of n, i.e. the dimension of the social network, and different ratios of malicious users in the network. For each case we have performed several simulations. We report in Table 2 the average error obtained by both algorithms.

As shown in the table, our method is more effective than Eigentrust for each value of the dimension and for each value of the ratio, presenting a lower error than the one produced by Eigentrust.

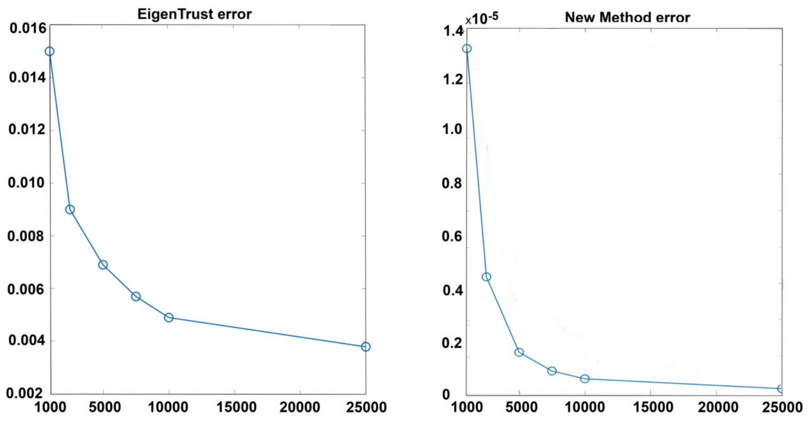

Fig. 3.3: Average errors in 2-norm with 10% of malicious users for different values of the matrix size.

As n increases, both errors decrease, but with our method the error becomes very small: for example, for $n = 25000$, with 20% of malicious members, the 2-norm error order is 10^{-7} against 10^{-3} of the Eigentrust algorithm.

Finally, we may observe that, as the ratio of malicious agents increases, the error increases for both methods, but our algorithm still produces a very small error (for $n = 25000$, the 2-norm error order with 10% of malicious members is 10^{-7} and it increases with 50% of malicious users but with the same 10^{-7}) magnitude.

In Fig. 3.3, we compare the 2-norm average errors obtained with the two methods, varying the number of users from 1000 to 25000 with a fixed ratio of 10% of malicious agents and in Fig. 3.4 and 3.5 we compare the 2-norm average errors obtained with the two methods in a social network of 5000 and 25000 users, varying the ratios of malicious users from 10% to 50%. The graphics in the other cases are analogous. Let us note that the behavior of the ∞-norm error is analogous to the one of 2-norm error. The significant improvement of the precision in determining

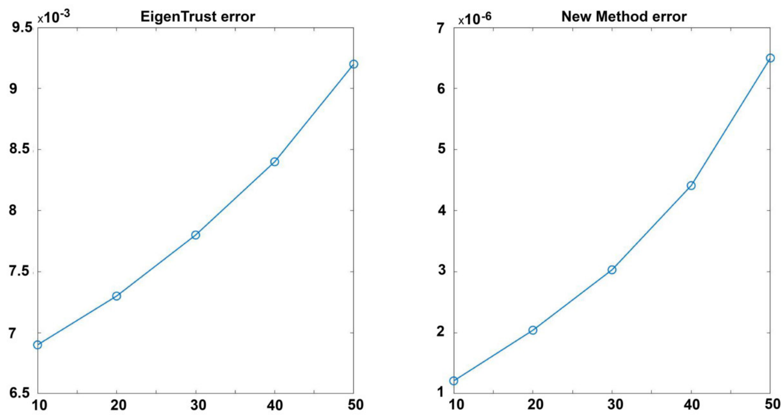

Fig. 3.4: Average errors in 2-norm for a social network with $n = 5000$ and different ratios of malicious users

.

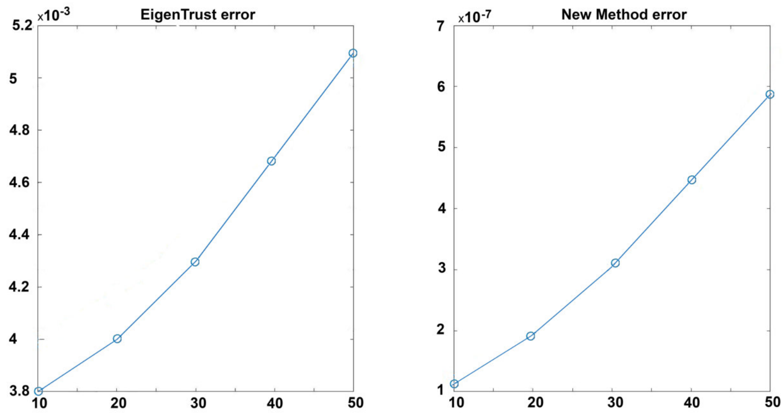

Fig. 3.5: Average errors in 2-norm for a social network with $n = 25000$ and different ratios of malicious users

.

the real reputation of the agents (i.e., that reputation that would be obtained if all the colluded agents were removed from the community) is due to the fact that our algorithm, differently from Eigentrust, avoids a strong penalization of pre-trusted (non-colluding) agents, while lowering the final reputation of malicious ones. This is possible since our algorithm performs a specific pre-processing phase (not present in EigentTrust) to detect potential colluded candidates. This then allows to assign a low value of trustworthiness only to these colluded candidates, without penalizing all the non pre-trusted agents (which is the strategy adopted by EigentTrust and that leads to the side-effect of generating incorrect reputation values for these non pre-trusted agents).

3.6 Benefits and Limitations.

In this chapter, we have focused on the problem of identifying, in a social network, the trustworthiness of an agent (human or software entity), in order to detect malicious actors and marginalize or expel them from the community. We have considered the Eigentrast algorithm, recognized as one of the most effective solution both to measure the reputation in a set of social agents and as benchmark or training assistant. But we have highlighted a possible limitation in the Eingentrust algorithm, observing that it increases the reputation of all the pre-trusted agents, regardless of their reliability. To address the problem above, in this chapter we have proposed a different strategy, based on a decrement of the fraudulent trust values that colluding agents mutually exchange. Our strategy introduces the advantage, with respect to Eigentrust, of estimating the reputation values of the honest actors in a manner more adherent to the actual reliability of these agents. This elevated precision of our method is particularly important, when the reputation of the agents is computed in a distributed environment, when different reputation values are continuously combined from different sources.

As limitations, in this work we have considered only one category of malicious agents, precisely colluding agents, as if they were organised in a single group. We have designed a strategy to determine malicious candidates in a pre-processing phase, before computing the reputation values of the agents. We thus highlight that our approach is limited to environments presenting only the collusion as malicious activities. We notice that the results of our algorithm in terms of effectiveness strictly depends on the richness of the information contained in the network, that is maximum for a complete network. Conversely, for other types of less rich and more sparse network it would inevitably lead to a deterioration of performances. Finally, our algorithm is currently limited to be applied on static networks.

In the next chapter, we propose an alternative strategy to detection of malicious colluding agents, considering the case of several malicious cluster in coalition.

4

A Graph Clustering algorithm in Trust and Reputation Systems

As it has been previously discussed, a common and crucial issue in all social network environments is the need to ensure a high level of trustworthiness, a fundamental prerequisite for fostering authentic connections and ensuring a secure and positive digital experience for members of communities.

The main challenge arises due to the presence of malicious agents engaging in deceptive behaviors or fraud. A prevalent form of malicious behavior is collusion, characterized by clandestine cooperation between multiple agents providing reciprocal positive feedback to manipulate perceptions and gain undeserved advantages. As highlighted in Chapter 3, the EigenTrust algorithm is an effective method for detecting malicious agents in social networks.

Although EigenTrust can be fruitfully used to detect agents performing several different malicious behaviours in a social network, however three important issues arise which limit its effectiveness when the particular malicious behaviour is the collusion:

- (*i*) EigenTrust algorithm needs to yield as inputs the information about those agents that can be a-priori considered particularly trustworthy, and detecting such agents is not a trivial task in the particular case of collusion;
- (*ii*) Eigentrust detects malicious agents by artificially rewarding, in terms of trust, non honest agents, without considering the reliability of these agents, i.e. the effectiveness in performing their tasks; this generates an artificial modification of the actual reputation values;
- (*iii*) finally, EigenTrust is not designed to face the situation in which we have several, different groups of colluded agents.

In this latter case, the presence of different clusters of malicious agents leads Eigentrust to confuse malicious agents, which have low reputation, with non-malicious but not very effective ones, which also have low reputation. This confusion generates, in these particular cases, a significant number of false positive agents. In particular, we highlight an important issue arising when it is necessary to compute reputation measures in presence of several, distinct groups of colluded agents. In this situation, the agents of each group operate independently from the colluded agents of each other group, with the result that the effect of the malicious activities of a given group of colluded agents could also falsify the reputation of another group of colluded agents, making harder to detect all the malicious agents. We have faced the first two issues in the previous Chapter, introducing an automatic procedure to provide to EigenTrust the necessary inputs mentioned in (*i*) and adopting a strategy that detects malicious agents without modifying reputation values of the other agents. Here we refer to the method introduced in Chapter 3 as **E**fficient **R**eputation-Eigentrust (ER-EingenTrust).

In this Chapter, we provide a contribution to face the issue (*iii*) (see [54]). To this purpose, we will introduce an apposite algorithm that combines EigenTrust with a clustering procedure, which groups agents based on their reputation scores.

4.1 Literature review on Spectral Clustering Algorithms

There are a number of studies and surveys of TR systems in the literature that analyze the robustness of TRs to a variety of malicious behaviors, although sometimes it can be complicated to compare systems designed for different specific application contexts.

In section 3.1 we introduced the main Trust and Reputation systems, as well as the mechanisms found in the literature that focus on identifying malicious users or restricting their fraudulent activities. In this section, we will mainly focus on a general overview on the clustering techniques, used for the classification of objects, data or, as in our specific context, agents. In particular we analyze in detail the Spectral clustering technique (see [55]), a particular algorithm that exploits the spectral properties of a matrix called Laplacian.

Spectral clustering goes back to Donath and Hoffman (1973), who first suggested to construct graph partitions based on eigenvectors of the adjacency matrix. In the same year, Fiedler (1973) discovered that bi-partitions of a graph are closely connected with the second eigenvector of the graph Laplacian, and he suggested to use this eigenvector to partition a graph. In the machine learning community, spectral clustering has been made popular by the works of Shi and Malik (2000), Ng et al. (2002), Meila and Shi (2001), and Ding (2004).

The success of spectral clustering is mainly based on the fact that it does not make strong assumptions on the form of the clusters. Spectral clustering can solve very general problems like intertwined spirals, whereas other clustering tecniques require strongly assumption, for example k-means, where the resulting clusters form convex sets (or, to be precise, lie in disjoint convex sets of the underlying space). Moreover, spectral clustering can be implemented efficiently even for large data sets, as long as we make sure that the similarity graph is sparse. Once the similarity graph is chosen, we just have to solve a linear problem, and there are no issues of getting stuck in local minima or restarting the algorithm for several times with different initializations.

4.1.1 Similarity graphs

Given a set of data points $x_1,\dots,x_n$ and some notion of similarity $s_{ij} \geq 0$ between all pairs of data points x_i and x_j, the intuitive goal of clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other. If we do not have more information than similarities between data points, a nice way of representing the data is in form of the similarity graph $G = (V,E)$. We assume that the graph G is weighted and undirect with vertex set $V = \{v_1,\dots,v_n\}$. Each vertex v_i in this graph represents a data point x_i. Two vertices are connected if the similarity s_{ij} between the corresponding data points x_i and x_j is positive or larger than a certain threshold, and the edge is weighted by s_{ij} . The weighted adjacency matrix of the graph is the matrix $W = (w_{ij})_{i,j=1,\dots,n}$. If $w_{ij} = 0$ this means that the vertices v_i and v_j are not connected by an edge. As G is undirected we require $w_{ij} = w_{ji}$. The degree of a vertex $v_i \in V$ is defined as

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

Note that, in fact, this sum only runs over all vertices adjacent to v_i, as for all other vertices v_j the weight w_{ij} is 0. The degree matrix D is defined as the diagonal matrix with the degrees $d_1,\dots,d_n$ on the diagonal. There are several popular constructions to transform a given set $x_1,\dots,x_n$ of data points with pairwise similarities s_{ij} or pairwise distances d_{ij} into a graph, for example:

- **The ϵ-neighborhood graph:**: We connect all points whose pairwise distances are smaller than ϵ. The ϵ-neighborhood graph is usually considered as an unweighted graph.
- ***k*-nearest neighbor graphs:** We connect vertex v_i with vertex v_j if v_j is among the k-nearest neighbors of v_i. However, this definition leads to a directed graph, as the neighborhood relationship is not symmetric. To solve this problem we can simply ignore the directions of the edges, that is we connect v_i and v_j with an undirected edge if v_i is among the k-nearest neighbors of v_j or if v_j is among the k-nearest neighbors of v_i.

- **The fully connected graph:** We simply connect all points with positive similarity with each other, and we weight all edges by s_{ij}. An example for such a similarity function is the Gaussian similarity function

  $$s(x_i, x_j) = e^{-(\|x_i - x_j\|/(2\sigma^2))},$$

  where the parameter σ controls the width of the neighborhoods. This parameter plays a similar role as the parameter ϵ in case of the ϵ-neighborhood graph.

4.1.2 Graph Laplacians

The Laplacian of a graph, also known as the graph Laplacian or Laplacian matrix, is a matrix associated with an undirected graph. This matrix is defined as the difference between the adjacency matrix of the graph and a diagonal matrix representing the degrees of the nodes, $L = D - W$. An important property of the Laplacian is its symmetry. The eigenvalues of the Laplacian matrix are all non-negative, and the smallest eigenvalue is always 0, with a multiplicity equal to the number of connected components in the graph. This relationship implies that the graph is connected if and only if its minimum eigenvalue is 0. The eigenvalues of the Laplacian matrix can provide information about the graph's structure, such as the number of connected components and bipartite structures. Moreover, there are two matrices which are called normalized graph Laplacians in the literature. Both matrices are closely related to each other and are defined as

$$L_{sym} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$

$$L_{rw} := D^{-1} L = I - D^{-1} W.$$

We denote the first matrix by L_{sym} as it is a symmetric matrix, and the second one by L_{rw} as it is closely related to a random walk.

As it is the case for the unnormalized graph Laplacian, the multiplicity of the eigenvalue 0 of the normalized graph Laplacian is related to the number of connected components. The properties of the Laplacian make it a fundamental tool for analyzing the structure and topological properties of graphs, with applications in various fields, including machine learning, graph theory, and complex network analysis.

4.1.3 Spectral Clustering Algorithms

We state the most common spectral clustering algorithms:

Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described above. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- Compute the first k eigenvectors $u_1, \dots, u_k$ of L.
- Let $U \in \mathbb{R}^{n \times n}$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns.
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-throw of U.
- Cluster the points $(y_i)_{i=1,\dots,n}$ in $\mathbb{R}^k$ with k-means algorithm into clusters $A_1, \dots, A_k$

Output: Clusters $C_1, \dots, C_k$ with $C_i = \{j \mid y_j \in A_i\}$.

There are two different versions of normalized spectral clustering, depending which of the normalized graph Laplacians is used.

Normalized spectral clustering according to Shi and Malik (2000)

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described above. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- Compute the first k generalized eigenvectors $u_1, \dots, u_k$ of the generalized eigenproblem $Lu = \lambda Du$.
- Let $U \in \mathbb{R}^{n \times n}$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns.
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-throw of U.
- Cluster the points $(y_i)_{i=1,\dots,n}$ in $\mathbb{R}^k$ with k-means algorithm into clusters $A_1, \dots, A_k$

Output: Clusters $C_1, \dots, C_k$ with $C_i = \{j \mid y_j \in A_i\}$.

Note that this algorithm uses the generalized eigenvectors of L which correspond to the eigenvectors of the matrix L_{rw}. The next algorithm also uses a normalized Laplacian, but this time the matrix L_{sym} instead of L_{rw}.

Normalized spectral clustering according to Ng, Jordan and Weiss (2002)

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described above. Let W be its weighted adjacency matrix.
- Compute the normalized Laplacian L_{sym}.
- Compute the first k eigenvectors $u_1, \dots, u_k$ of L_{sym}.
- Let $U \in \mathbb{R}^{n \times n}$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns.

- Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1, that is set $t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2}$.
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-throw of T.
- Cluster the points $(y_i)_{i=1,\dots,n}$ in $\mathbb{R}^k$ with k-means algorithm into clusters $A_1, \dots, A_k$

Output: Clusters $C_1, \dots, C_k$ with $C_i = \{j \mid y_j \in A_i\}$.

All three algorithms stated above look rather similar, apart from the fact that they use three different graph Laplacians. A fundamental question related to spectral clustering is related to which of the three graph Laplacians should be used to compute the eigenvectors. One should always look at the degree distribution of the similarity graph. If the graph is very regular and most vertices have approximately the same degree, then all the Laplacians are very similar to each other, and will work equally well for clustering. However, if the degrees in the graph are very broadly distributed, then the Laplacians differ considerably and it would be appropriate to use normalized rather than non-normalized spectral clustering. To use these algorithms, as evident, it is essential to have an availability of a similarity matrix and the appropriate number of clusters. Constructing the similarity graph for spectral clustering is not a trivial task, and little is known on theoretical implications of the various constructions. Choosing the number k of clusters is a general problem for all clustering algorithms, and a variety of more or less successful methods have been devised for this problem. In model-based clustering settings there exist well-justified criteria to choose the number of clusters from the data. Those criteria are usually based on the log-likelihood of the data, which can then be treated in a frequentist or Bayesian way.

4.2 Computing the reputation

4.2.1 The model

For the sake of clarity we recall now the model described in subsection 3.2.1. Suppose to have a social network composed by n members, each member being uniquely identified by an integer i, $1 \le i \le n$.

The *trust* perceived by the member j with respect to the member i is represented by the real number t_{ij}, $0 \le t_{ij} \le 1$, $\forall i, j = 1, \dots, n$.

The *reputation* r_i of each member i of the social network can be seen as the sum of all the trust values t_{ij}, $j = 1, \dots, n$, corresponding to a trustor j, weighted by the reputation r_j of j. In other words, r_i can be viewed as the barycenter of all the trust values expressed for i by the trustors. Formally:

$$r_i = \frac{\sum_{j=1}^n t_{ij}\, r_j}{\sum_{j=1}^n r_j}, \quad i = 1,\ldots,n. \tag{4.1}$$

Let us denote with $T = [t_{ij}]$ the *trust matrix*.

The *reputation vector* $r = (r_1,\ldots,r_n)^T$ can then be computed by solving the following eigenvector problem:

$$Tr = r, \quad \|r\|_1 = 1, \tag{4.2}$$

where the 1-norm is defined as the sum of the values of the vector components and with the assumption

$$\sum_{i=1}^{n} t_{ij} = 1, \tag{4.3}$$

which is equivalent to say that the matrix T is column-stochastic.

To mitigate the final reputation of malicios users, a modified model, known as *Eigentrust algorithm* has been proposed [43]. Nevertheless, in this approach, the pretrusted users have to be known a priori and Eigentrust does not suggest any kind of procedure for a priori discriminating honest and malicious agents.

Such a detection can lead to a more realistic computation of the reputation vector. In Chapter 3 a strategy has been proposed to "read" such information straight from the matrix T and to use it in the computation of the final reputation vector.

To this purpose, a proper definition of malicious user has been conveniently given in the following way:

We classify a group of agents as *malicious* if they *collude*, i.e. if they have a bidirectional exchange of high trust values and, at the same time, they are given low trust values by the other members.

For such a definition of malicious, it is natural to think of clustering techniques for identifying groups and to use such information to compute the final actual reputation, as described in the next subsection.

4.2.2 Detection of malicious users through graph clustering

An appropriate model for a social network can be given in terms of a graph $G = (V,E)$, where each vertex (or node) v_i corresponds to a user. In our scenario G is a directed weighted graph, the value t_{ji} (i.e. the trust value assigned by the member i to the member j) representing the weight of the edge (i,j).

Note that the adjacency matrix A of such graph corresponds to the transpose of the trust matrix T.

A common feature of social networks is the presence of community structure properties, so that the graph can be often considered organized into subgraphs, each of them representing a *cluster*.

In our scenario, the maliciousness property of a pair of agents can be reviewed in terms of *similarity* and an appropriate definition of similarity is the crucial aspect for every clustering process. Roughly speaking, two nodes belonging to a certain cluster are much more "similar" than two nodes belonging to different clusters.

Therefore, the concept of similarity is fundamental for grouping together items that share similar characteristics or neighbouring behaviours. Similarity can be defined through different metrics, depending on the type of data or characteristics being considered. Then, to precisely define the similarity between users we base it on their distinguishing characteristics. For example, good honest users are characterised by exchanging high trust values with each other and receiving high trust values. On the other hand, malicious colluding users are identified by the practice of exchanging high trust values with each other and receiving low trust values from other users.

Our aim is to use techniques for graph clustering for detecting malicious behaviours in order to use such information for a more reliable computation of the final reputation.

The problem of identifying clusters over a network have been addressed in several contributions (see, for example, [56, 57] and references therein). However, the case of undirected graphs has been mainly considered. Finding clusters in directed graphs is a more challenging task and is the topic of an increasing research activity (see [58] for a review).

For the purpose of our work, we found out that the use of spectral clustering techniques, originally designed for undirected graphs, can be adopted by means of a proper reformulation of the problem.

Since spectral clustering only operates on symmetric adjacency matrices, a proper redefinition of such matrix must be given.

Typical transformations to the adjacency matrix in order to obtain a symmetric one, include $\tilde{A} = AA^T$, $\tilde{A} = A^T A$.

Nevertheless, due to its simplicity, most of the algorithms use $\tilde{A} = A + A^T$. This approach almost ignores the directionality of the edges, except that in the case of pairs of nodes with directed edges in both directions, which is indeed the situation we are facing in this work (malicious pairs). On the other hand, the information about the low trust values received by malicious agents by the other members risks to be lost.

We thus propose to adopt the following transformation for the elements in the new adjacency matrix:

$$\tilde{a}_{ij} = \frac{a_{ij} + a_{ji}}{2(0.1 + |a_{ij} - a_{ji}|)}. \tag{4.4}$$

This transformation allows us to create an adjacency matrix with optimal attributes for our goals. Dividing the average of elements a_{ij} and a_{ji} for their difference, with the addition of a constant 0.1, we obtain the connection between users. A larger disparity between elements produces a value of $\tilde{a}_{ij}$ lower, while a smaller difference generates a higher value for $\tilde{a}_{ij}$. Then, by construction, $\tilde{A} = [\tilde{a}_{ij}]$ is a symmetric matrix, whose elements $\tilde{a}_{ij}$ attain high values if the users i and j are "similar", low values if the users i and j are "not similar", thus satisfying the definition of similarity given above.

Spectral clustering is a strategy for clustering graph data which operates on the Laplacian matrix L, or on some normalized version of it, performing dimensionality reduction before using a standard clustering procedure, as k-means.

In particular, we make use of the *normalized spectral clustering* procedure, in according to Ng, Jordan and Weiss, described in [59]. Our input data consist in: the similarity matrix $\tilde{A}$ as in (4.4) and the number k of clusters.

Thus the process consists of the following steps:

1. Construct the *normalized graph Laplacian* matrix

$$L = D^{-1/2}(D - \tilde{A})D^{1/2}$$

   where D is the *degree matrix* defined as the diagonal matrix with diagonal elements $d_{ii} = \sum_{j=1}^{n} \tilde{a}_{ij}$
2. Evaluate the k eigenvectors corresponding to the k smallest eigenvalues of the L. Obtain a reduced-dimensional representation of the data arranging such eigenvectors in the columns of a matrix V of size $n \times k$
3. normalize the each row $v_i \in \mathbb{R}^k$ of V so that its 2-norm is equal to 1
4. Use k-means to cluster the rows v_i, $i = 1, \dots, n$, of V, into the clusters $C_1, \dots, C_k$

An important role in the above described procedure is played by the parameter k connected to the number of clusters, which must be judiciously chosen according to the number of users and the type of social network. A possible automatic choice can be performed in terms of *eigengap heuristic* in the following way: choose k such that the first k eigenvalues of L are very small and the following one is relatively large. This idea works well in case of very distinct clusters, but fails in case there is some kind of "overlapping". If we consider very large communities, an effective approach present in literature is to approximate the number of clusters k with $\log(n)$, where n is the number of users. In our case, with small to medium communities, we have heuristically determined this number by an analysis of sensitivity, testing different values and using the value leading to the best results.

Once the clusters $\mathcal{C} = \{C_1, \dots, C_k\}$ have been obtained, we proceed recursively as follows

1. Compute the reputation vector r as in (4.2)
2. Compute the j-th cluster reputation as
$$\bar{r}_j = \frac{\sum_{i \in C_j} r_i}{|C_j|}, \quad \forall j = 1, \ldots, k$$
3. Fixing $\delta > 0$, identify as "colluding clusters" all those that possess a reputation less or equal to δ
4. Remove from the network all the agents belonging to such clusters, i.e. delete the corresponding rows and columns from the matrix T. Let $\mathcal{C} \leftarrow \mathcal{C} \setminus \{C_j : \bar{r}_j \leq \delta\}$, $k = |\mathcal{C}|$
5. Repeat from step 1 and stop when all clusters in $\mathcal{C}$ have a reputation greater than δ.

4.3 An explicative example

We propose an example to illustrate our approach in the particular situation of a social network of 8 agents. We consider the original trust matrix

$$T = \begin{pmatrix} 0 & 0.9 & 0.7 & 0.7 & 0.2 & 0.2 & 0.8 & 0.9 \\ 0.9 & 0 & 0.8 & 0.7 & 0.3 & 0.1 & 0.9 & 0.8 \\ 0.25 & 0.25 & 0 & 0.9 & 0.95 & 0.95 & 0.2 & 0.2 \\ 0.25 & 0.25 & 0.9 & 0 & 0.95 & 0.95 & 0.15 & 0.2 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0.9 & 0.3 & 0.25 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.9 & 0 & 0.25 & 0.3 \\ 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0 & 0.4 \\ 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0 \end{pmatrix}.$$

The suspected malicious colluding group are the ones corresponding to the clusters (3,4) and (5,6).

We assume that $t_{ii} = 0, \forall i = 1, \ldots, 8$, so that the trust assigned by member to itself is not consideredand compute similarity matrix S.

$$S = \begin{pmatrix} 0 & 9 & 0.86 & 0.86 & 0.75 & 0.75 & 1.2 & 1.08 \\ 9 & 0 & 0.8 & 0.86 & 0.66 & 1 & 1.08 & 1.2 \\ 0.86 & 0.8 & 0 & 9 & 0.55 & 0.55 & 1 & 1 \\ 0.86 & 0.86 & 9 & 0 & 0.55 & 0.55 & 0.78 & 1 \\ 0.75 & 0.66 & 0.55 & 0.55 & 0 & 9 & 1.75 & 1.3 \\ 0.75 & 1 & 0.55 & 0.55 & 9 & 0 & 1.3 & 1.75 \\ 1.2 & 1.08 & 1 & 0.78 & 1.75 & 1.3 & 0 & 4 \\ 1.08 & 1.2 & 1 & 1 & 1.3 & 1.75 & 4 & 0 \end{pmatrix}.$$

We apply *Spectral Clustering*, after choosing the appropriate number k of clusters, in this case $k = 4$.

Agents $i = 1, \dots, 8$ are clustered as follows:

$$C_1 = (1,2),\ C_2 = (3,4),\ C_3 = (5,6),\ C_4 = (7,8).$$

By computing the reputation score of users, we obtain the average reputation $\tilde{r}(C_k)$ associated with each cluster:

$$\tilde{r}(C_1) = 0.20 \qquad \tilde{r}(C_2) = 0.12$$
$$\tilde{r}(C_3) = 0.06 \qquad \tilde{r}(C_4) = 0.12$$

In corrispondence of $\delta = 0.11$, we have $\tilde{r}(C_3) \le \delta$, then the users 5 and 6 are classified as malicious colluding agents and removed from the network.

We recalculate the new reputation score of the remaining users then the new average reputation of the remaining clusters:

$$\tilde{r}(C_1) = 0.25 \quad \tilde{r}(C_2) = 0.10 \quad \tilde{r}(C_4) = 0.14$$

We note that $\tilde{r}(C_2) \le \delta$, then the users 3 and 4 are classified as malicious colluding agents and removed from the network.
By iterating the process, we observe that the two remaining clusters (C_1 and C_4) have a greater average reputation than δ, so we stop the algorithm.

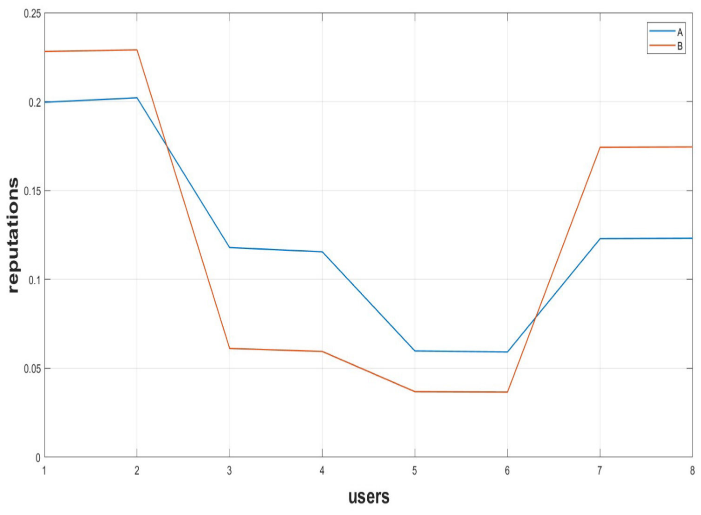

Fig. 4.1: Reputations obtained by using and not using malicious

In Figure 4.1, we present, for information purposes only, the user reputations obtained through the use of ER-EigenTrust. In case A, no malicious users are taken into account, while in case B we incorporate data on malicious users obtained through the Cluster Method.

It is evident how the presence of malicious users in a social network impacts all reputations, thus underlining the importance of a method that accurately identifies malicious users at a prior phase to the calculation of the final reputations. This would ensure the provision of correct information to users, preserving the integrity of the reputational assessment process.

4.4 Experimental Results

We present the results of an experimental campaign of simulations for testing our Clustering method, comparing it with our previous algorithm. We stress that we obtain the information on the malicious colluding cluster (namely users) straight from the matrix T. Our numerical code has been implemented using MATLAB Release 2023a following the steps summarized in section 4.2.2. Our numerical experiments were run on a 64-bit workstation with a Intel(R) Core(TM) i7-10875H CPU @ 2.30 GHz and 128 GB of RAM. We considered different cases, for different values of n, i.e. the dimension of the social network, and different ratios of malicious users in the network. For each case we have performed several simulations. As shown in Table 1 and Table 2, to evaluate the performance of our algorithm, we used the well-know metrics *precision*, *recall* and *F-score*, making a comparison with the values obtained with the threshold strategy in our previous algorithm.

$$precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

$$recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

$$F - score = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

To clarify and make more accessible the fundamental concepts used in the above-mentioned evaluation metrics, we recall that:

- *True Positives* are the positive elements that have been correctly classified as positive by the model, in our case the malicious colluding agents;
- *False Positives* are the negative elements that have been wrongly classified as positive by the model, in our case the honest agents;
- *True Negatives* are the negative elements that have been correctly classified as negative by the model;
- *False Negatives* are the positive elements that have been misclassified as negative by the model.

In particular, *precision* measures how much of the cases classified as positives are actually positives, while *recall* measures how much of the true positives have been identified by the model. Instead, the *F-score* provides a single number that balances *precision* and *recall*. This metric is particularly useful in situations where classes are unbalanced, as it takes into account both type I errors (false positives) and type II errors (false negatives). A high *F-score* indicates a good balance between *precision* and *recall*.

n	**% of malicious**	**Cluster method**			**ER-EigenTrust method**		
		precision	*recall*	*F-score*	*precision*	*recall*	*F-score*
100	5%	1	1	1	0.13	1	0.23
	10%	1	1	1	0.2	1	0.3
	15%	0.94	1	0.96	0.27	1	0.42
	20%	1	0.8	0.88	0.34	1	0.51
	25%	1	0.91	0.95	0.38	1	0.55
200	5%	1	1	1	0.16	1	0.27
	10%	1	1	1	0.34	1	0.51
	15%	1	0.8	0.88	0.44	1	0.61
	20%	1	0.85	0.91	0.55	1	0.71
	25%	1	0.92	0.95	0.51	1	0.68
300	5%	1	0.87	0.93	0.12	1	0.21
	10%	1	0.8	0.88	0.19	1	0.32
	15%	1	0.82	0.9	0.38	1	0.55
	20%	1	0.83	0.9	0.4	1	0.57
	25%	1	0.82	0.9	0.57	1	0.73
400	5%	1	1	1	0.13	1	0.23
	10%	1	0.8	0.88	0.17	1	0.29
	15%	1	1	1	0.26	1	0.41
	20%	1	0.87	0.93	0.39	1	0.56
	25%	0.98	0.94	0.96	0.46	1	0.63
500	5%	1	0.92	0.96	0.11	1	0.20
	10%	1	0.88	0.93	0.2	1	0.33
	15%	0.98	0.97	0.98	0.32	1	0.49
	20%	0.98	0.91	0.94	0.42	1	0.60
	25%	0.99	0.91	0.95	0.46	1	0.63

Table 4.1: Results of common evaluation metrics for small communities.

n	% of malicious	Cluster method			ER-EigenTrust method		
		precision	*recall*	*F-score*	*precision*	*recall*	*F-score*
1000	5%	1	1	1	0.16	1	0.28
	10%	1	0.96	0.97	0.25	1	0.40
	15%	0.98	1	0.99	0.31	1	0.47
	20%	0.96	0.89	0.92	0.42	1	0.59
	25%	0.96	0.94	0.95	0.51	1	0.68
2500	5%	0.92	1	0.96	0.14	1	0.25
	10%	0.96	0.99	0.97	0.27	1	0.42
	15%	0.97	0.98	0.98	0.34	1	0.51
	20%	0.96	0.66	0.78	0.28	1	0.43
	25%	0.95	0.95	0.93	0.61	1	0.54
5000	5%	0.92	1	0.96	0.13	1	0.23
	10%	0.96	0.98	0.97	0.26	1	0.42
	15%	0.97	0.99	0.98	0.37	1	0.54
	20%	0.96	0.85	0.90	0.78	1	0.87
	25%	0.93	0.59	0.72	0.55	1	0.71
7500	5%	0.94	0.99	0.97	0.14	1	0.24
	10%	0.97	0.99	0.98	0.26	1	0.42
	15%	0.98	0.97	0.97	0.4	1	0.57
	20%	0.98	0.88	0.92	0.86	1	0.92
	25%	0.99	0.89	0.94	0.87	1	0.93
10000	5%	0.89	1	0.94	0.15	1	0.27
	10%	0.98	0.98	0.98	0.27	1	0.43
	15%	0.98	0.97	0.97	0.35	1	0.52
	20%	0.93	0.37	0.53	0.75	1	0.86
	25%	0.97	0.74	0.84	0.77	1	0.87

Table 4.2: Results of common evaluation metrics for medium communities.

It is very clear that our method having a very high *recall* maintains the same effectiveness as Eigentrust in detecting malicious agents (also considering the improvement introduced by ER-EigentTrust). But the advantage is mainly in terms of *precision*: it is significantly better at avoiding false positives.

For example, as shown in the Figure 4.2, as *n* increase, leaving the percentage of malicious stable at 15%, the *precision* of Cluster Method is high, in many cases with values of 1 for small communities. Instead, the *precision* of ER-EigenTrust Method is low, varying with values between 0.27 and 0.55.

In Figure 4.3 we analyse the *precision* obtained in a social network with 500 users, varying the percentage of malicious from 5% to 25%. Regarding the Cluster Method,

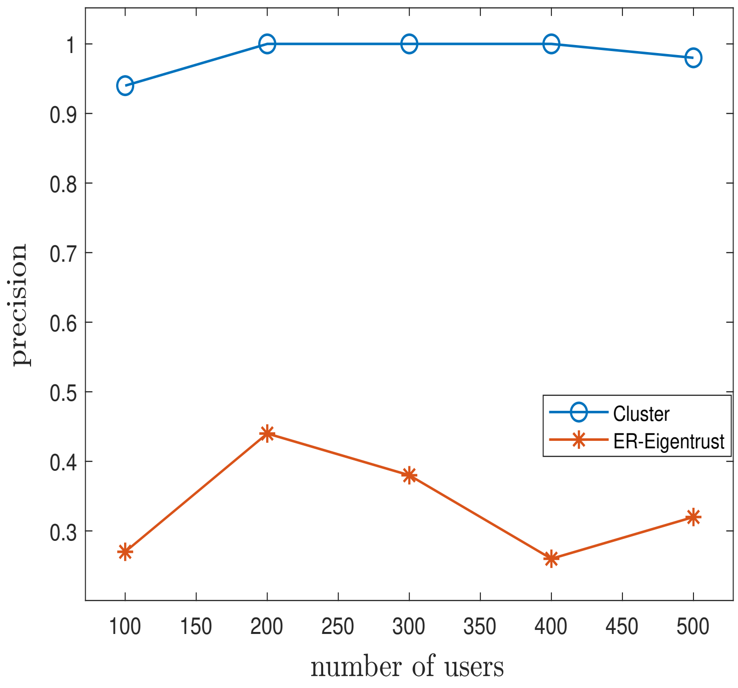

Fig. 4.2: Precision of Cluster and ER-EigenTrust methods as agents increase.

we note that as the percentage of maliciousness varies, the *precision* remains stable at high values, while as regards ER-EigenTrust Method, the *precision* is very low but increase as the percentage of malicious increase.

The significant improvement in *precision* achieved through the application of the Cluster Method is attributable to its ability to aggregate malicious users into appropriate clusters and honest users into others. This subdivision facilitates a distinct disparity in average reputations between individual clusters, mitigating the risk of misclassifying honest users as malicious. This is particularly relevant since honest users are generally expected to maintain a higher level of reputation.

In contrast, the ER-EigenTrust Method directly classifies malicious users based on the trust matrix, determining that all users, including honest ones, who receive low trust ratings are tagged as malicious. This process leads to a situation in which any honest users who receive low trust ratings and, consequently, possess a low reputation, are also wrongly identified as malicious. This procedure increases the number of false positives and ultimately reduces the *precision* of the model.

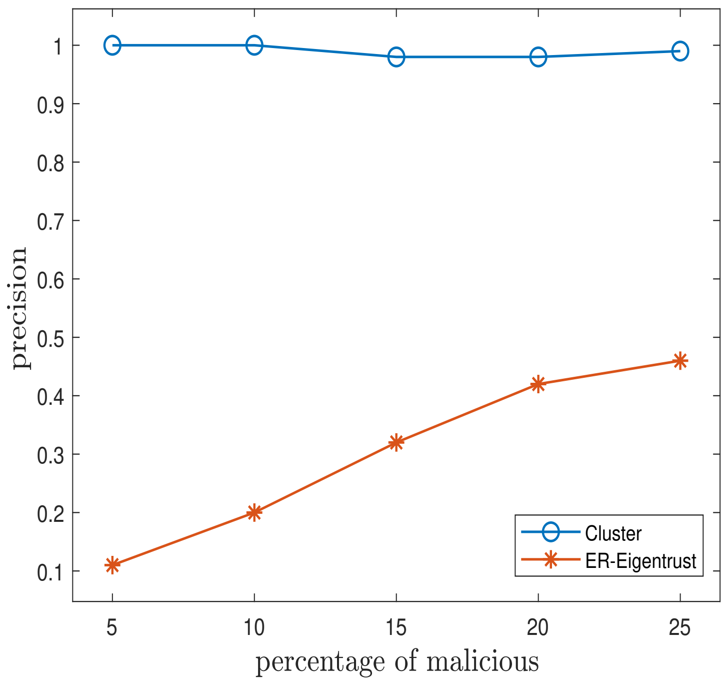

Fig. 4.3: Precision of Cluster and ER-EigenTrust methods as malicious agents increase.

4.5 Benefits and Limitations

In this Chapter, we have focused on the problem of detecting colluded agents in a social networks, trought cluster tecniques, and marginalize or expel them from the community.

In this context, several TRR systems have been proposed in the literature, and here we have examined the particular case represented by the well-known algorithm Eigentrust, that is recognized as one of the most effective solution to measure the reputation in a set of social agents. We had already faced, in Chapter 3, two important problems affecting Eigentrust, i.e. the use of some additional information about agents that can be a-priori considered particularly trustworthy, and the strategy that EigenTrust uses of rewarding these trustworthy agents in terms of trust, while the other agents are penalized, producing the side effect to flattening the differences, in terms of reliability, between honest agents. However, we have highlighted as ER-EigenTrust, similarly to the original version of EigenTrust, presents an important limitation in terms of false positives that are generated when colluded agents are partitioned in different groups. We have proposed a new algorithm for detecting colluded agents, which combines EigenTrust with a clustering procedure, grouping agents based on their reputation scores. Our experimental campaign, described in the previous section, shows that our method, besides of maintaining the same effectiveness of Eigentrust in detecting malicious agents (also considering the improvement introduced by ER-EigentTrust), is significantly more capable of avoiding the presence of false positives. This advantage, in terms of precision, increases from a 55 percent achieved by ER-EigenTrust to a 89 percent reached by our proposed method an a social networl having 10.000 users and a 25 percent of malicious agents, and this difference is even higher when the percentage of malicious agents is smaller.

It is important to highlight two current limitations of our approach. First, the reputation threshold under which a group should be considered as colluded is arbitrarily decided from the human administrator of the social network; although this can be reasonable to take into account the personal opinion of the administrator about the level of reputation that a group of agents must have to be considered as colluded, however we argue that some more deep considerations should be made about the possibility to help the administrator with additional information automatically extracted from the social network. Secondly, the clustering algorithm we have used needs as input the number of clusters to be formed. Currently, we are heuristically determining this number by an analysis of sensitivity, testing different values and using that value leading to the best results. However, an apposite procedure to automatically determining the reputation threshold in a more efficient way would increase the efficiency of the approach. Moreover the computational cost of our algorithm is $O(n^3)$, making it difficult to scale. In our experimental campaign, we adopted a centralized approach with acceptable execution times. However, for larger communities, it is advisable to move towards a distributed approach that may perform better than centralized approaches in terms of computational costs, although they are more complex to implement and more vulnerable to malicious attacks.

Part II

A Variational Approach in Trust and Reputation Systems

5 Variational Problems

Throughout the research on trust and reputation in information systems, this thesis aims to explore new perspectives and deepen the understanding of these concepts by adopting an innovative methodological framework in its second part. Specifically, while the first part examined trust and reputation systems from a well-established perspective, the subsequent part of the thesis distinguishes itself by adopting an approach based on variational inequalities and the application of Lagrange multipliers.

5.1 Model-optimizing approach

Mathematical models are often applied to real phenomena or situations and they are used in many fields, such as the natural sciences (physics, biology, earth science, meteorology), the engineering disciplines (artificial intelligence, mechanics, computer science), in the social sciences (economics, psychology, sociology, management science and political science) and other important areas. This is because a model can be useful to explain what we are describing and to make predictions about the future, allowing us to adopt the best strategy and make the correct decision. Therefore, mathematical modelling is one of the main approaches that mathematicians use to describe real situations.
Over the years, numerous results have been achieved in the area of Operational Research thanks to the contribution of several scholars: J.L. von Neumann, who developed a series of models for studying economic grow thin conditions of competitive equilibrium, decision-making in a multidisciplinary envoronment, and how to run computing programs; P.M.S. Blackett and T.C. Koopmans studying models for logistic applications; G.B. Dantzing who contributed to the development of linear programming with the introduction of a resolution operating method known as the "simplex method". Around 1950 the first results in the area of network optimization algorithms began to appear and in the 1960s the researchers focused the problem of assessing the efficiency of solving decision-making algorithms.

Thus, the central contribution of Operations Research consists of the introduction of the so-called *model-optimizing approach* for the solution of a decision problem. In this approach, the analysis of a real problem is organized into two phases:

- representation of the problem means of a mathematical model that should extract the essential aspects and outline the interrelations existing between the different aspects of the phenomenon under examination;
- development of efficient mathematical methods to determine an optimal solution of the problem.

To build a mathematical-optimizing model which represents a particular phenomenon, the significant control parameters must be identified. Having determined the correct model, the OR is responsible for providing an explicit procedure to determine a solution to a problem. This procedure can be represented by analytical mathematical methods or numerical methods which determine the solution of the problem through specific calculation algorithms.

The modeling approach is achieved through different phases:

- Problem analysis,
- Model construction,
- Model analysis,
- Numerical solution,
- Validation of the model.

The first phase consists in *analyzing the structure of the problem* to identify the logical-functional links and objectives, to collect the data. In the subsequent construction phase of the model, also called *formulation*, the main characteristics of the problem are described in mathematical terms. Then follows the *analysis of the model* which provides the analytical deduction of some important properties, such as the existence and uniqueness of the optimal solution, the optimality conditions, that is: the analytical characterization of the optimal solution and the stability of the solutions when the data or any parameters change.
The next phase of *numerical solution* takes place by means of suitable calculation algorithms and the numerical solution thus obtained must then be interpreted from an applicative point of view. This *model validation* can be carried out by experimental verification or simulation methods.

5.2 Optimization models

Many real problems, studied in different disciplines, consist in finding the maximum or the minimum value of a determined function. The branch of study dealing with such problems takes the name of Optimization Theory. Among the various optimization problems there are those where it is necessary to determine the optimal values of a function, whose decision-making variables are subject to constraints expressed by equalities and/or inequalities. Given a function $f : \mathbb{R}^n \to \mathbb{R}$ and the set $K \subseteq \mathbb{R}^n$, an *optimization problem*, called also *mathematical programming problem* (term introduced by Robert Dorfman in 1949), can be formulated as follows:

$$\min_{x \in K} f(x).$$

Therefore, an optimization problem consists in determining, if it exists, a minimum point of the function f between the points of the set K.
The function f is called *objective function* and K is the *feasible set*, that is the set of all possible solutions to the problem. A point $x \in K$ is called feasible solution or candidate solution.
The feasible set K is a subset of $\mathbb{R}^n$ and therefore $x = (x_1, \dots, x_n)^T$ is a n-dimensional vector and the objective function f is a function of n real variables.
We underline that it is possible to speak indifferently about problems of maximum or minimum because the following relation is valid:

$$\min_{x \in K} f(x) = -\max_{x \in K}(-f(x)).$$

Definition 5.1. *An optimization problem is said to be infeasible if $K = \emptyset$.*

Definition 5.2. *It is said that the optimization problem admits an optimal solution if there exists a point $x^* \in K$ such that it results $f(x^*) \leq f(x)$ for all $x \in K$. Point x^* is called optimal solution or global minimum and the corresponding value $f(x^*)$ is called optimal value.*

A first classification of the optimization problems is based on the structure of the feasible set K. If $K = \mathbb{R}^n$ then the problem is said to be *unconstrained*. If the set K is described by a finite number of inequalities and/or equalities, $K = \{x \in \mathbb{R}^n : g(x) \leq 0, h(x) = 0\}$, then the problem is said to be *constrained* and has the following formulation:

$$\begin{cases} \min f(x) \\ g_i(x) \leq 0 \ \ \forall i = 1, \dots, m \\ h_j(x) = 0 \ \forall j = 1, \dots, p \\ x \in \mathbb{R}^n. \end{cases} \tag{5.1}$$

Moreover, the Mathematical Programming problems can be classified also according to the nature of the functions that define them:

- *Linear* Programming Problem (LP) if the objective function f and all the functions that define the constraints are linear;

- *Nonlinear* Programming problem (NLP) if at least one of the functions defining the problem is not linear.

Finally, depending upon the values permitted for the variables, optimization problems can be classified as *integer* (if all the variables can only take integer values), *mixed integer* (if only some of the variables are constrained to take integer value) or *real valued*, and *deterministic* or *stochastic*.

Thanks to the generality of the models, an outsize number of real problems can be represented by programming models.

The kind of model that must be studied and the most suitable approach shape the choice of the mathematical tools to be used, such as dynamic systems, variational inequalities, game theory and many others.
Furthermore, the main role in the formulation of a mathematical model and in decision-making processes is played by the definition of the network on which the whole model is based. Indeed, the network allows the definition the different layers of decision-makers involved in the whole process. The related flows and the methods of analysis are applicable not only to physical networks, such as transport and energy networks, production and logistics, but also to complex networks such as supply chains, financial, social and economic networks.

5.3 A brief recall to Lagrange Theory

Modern optimization problems originated towards the end of the last century, but the history of mathematical programming dates back the end of the 1700s, although limited to the case of equality constraints. Indeed, in the second half of the 18th century, G.L. Lagrange studied mathematical programming problems which consisted in minimizing (or maximizing) a given function, subject to a system of constraints expressed by equality.
The well known multiplier method was introduced by Lagrange in 1788, in the first part of his book titled *Mècanique Analytique*, as a tool to determine the stable equilibrium configuration in a specific issue of Mechanics. The Lagrange multipliers were presented for general optimization problems, not referring to any mechanical system, in *Thèorie de fonctions analytiques* (1797).

Given the optimization problem 5.1, the function

$$L(x,\lambda,\mu) = f(x) + \sum_{i=1}^{m} \lambda g(x) + \sum_{j=1}^{p} \mu h(x)$$

is called the *Lagrangian function* of the problem.

Assume $x^* = (x_1^*, x_2^*, \dots, x_n^*)$ minimizes $f(x)$ subject to the constraints $g_i(x) \leq 0$, for $i = 1, 2, \dots, m$ and $h_j(x) = 0$, for $j = 1, 2, \dots, p$.
Then either:

(i) the vectors $\nabla g_1(x^*), \dots, \nabla g_m(x^*), \nabla h_1(x^*), \dots, \nabla h_p(x^*)$ are linearly independent, or

(ii) there exists a vector $\lambda^* = (\lambda_1^*, \dots, \lambda_m^*)$ and $\mu^* = (\mu_1^*, \dots, \mu_p^*)$ such that

$$\begin{aligned}
&\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \mu_j^* \nabla h_j(x^*) = 0,\\
&g_i(x^*) \leq 0 \quad \forall i = 1, 2, \dots, m,\\
&h_j(x^*) = 0 \quad \forall j = 1, 2, \dots, p,\\
&\lambda_i^* g_i(x^*) = 0 \ \ \forall i = 1, 2, \dots, m, \quad \text{(Complementarity)},\\
&\lambda_i^* \geq 0 \quad \forall i = 1, 2, \dots, m.
\end{aligned}$$

The above conditions are called the **Karush-Kuhn-Tucker conditions** (see [60] and [61]) and are the necessary conditions for the solution of a nonlinear programming problem.
This is a generalization of the *Lagrange multiplier method*, applied to problems in which there are also inequality constraints. The first condition is that of the cancellation of the gradient of the Lagrangian function associated with the problem. The second and third conditions are the constraints of the admissibility of point x^*, while the fourth condition is called a complementarity condition or a "complementary deviation", since the multiplier of an inactive constraint must be null. Finally, the last condition is the non-negativity condition of the multiplier associated with the inequality constraints.

5.4 Variational Inequality Theory

A great variety of problems in the real world can be traced back to variational models that are much closer to reality, when their equilibrium condition is expressed as a solution to a system of Variational Inequalities (VI).
The scientific life of the Variational Inequalities Theory has immediately proved to be eventful and surprising. This theory was developed in the seventies as an innovative and effective method to solve a series of equilibrium problems. It was advanced by mathematical physicists to solve, for example, the problem of Signorini (1959), the problem of the obstacle and that of elasto-plastic twisting.
Therefore, the first variational inequality problem was the problem known as the Signorini problem (see [62]). His student, Gaetano Fichera, dedicated the name to

him and resolved it in 1963. In 1964, Guido Stampacchia, a 20th-century Italian mathematician, known for his work on the theory of variational inequalities, generalized the Lax-Milgram theorem (see [63]) in order to study the regularity problem for partial differential equations and the name "variational inequality" was coined by him for all the problems involving inequalities of this kind. Further in-depth analyses of previous studies date back to 1966 by Hartman and Stampacchia (see [64]) and to 1967 by Stampacchia and Jacques-Louis Lions (see [65]).
An explanation of infinite-dimensional variational inequalities and numerous references can be found in the text by Kinderlehrer and Stampacchia (see [66]).
After an intense period of successes and fundamental results obtained with the Variational Inequalities theory, the interest waned, perhaps because of the early death of Stampacchia in 1979, and it seemed that this theory had nothing more to communicate.
On the contrary, in 1980, the breakthrough in finite-dimensional theory occurred when S. Dafermos recognized that the problem of traffic equilibrium, as stated by M.J. Smith (1979), could be formulated in terms of a finite dimensional inequality and, moreover, in this way it is possible to study the existence, uniqueness and stability of the traffic equilibrium problem and calculate the solutions. So began the use of this methodology for the study of problems in economics, management science/-operations research, and also in engineering, with a focus on transportation.

At the end of the nineties, researchers started to investigate optimization problems, through a variational approach, by considering also time-dependence. Daniele, Maugeri and Oettli, in [67] and [68] (see also [69]), first studied and analyzed the traffic network equilibrium problem with feasible path flows which have to satisfy capacity constraints dependent from time and traffic demands.
As a result of this, the last decades have witnessed an exceptional interest in Variational Inequalities both in the development of VI theory and its application to equilibrium problems arising in many different contexts and an enormous amount of papers and books have been dedicated to this topic.

For the analysis of economic phenomena, equilibrium is a central concept, therefore, various problems from the world of economics, such as those of spatially distributed economic and oligopolistic markets, migration, pollution and many other problems, have been formulated in terms of a finite dimensional variational inequality and, by means of this theory, they have been solved.

Recently, a lot of problems coming from fields of applied sciences such Operation Reasearch, Physics, Engineering, Biology and Economics are studied as optimization problems with a variational approach.

Now, we introduce the definition of Variational Inequality:

Definition 5.3 (Variational Inequality Problem). *The finite-dimensional variational inequality problem, $VI(F,K)$, is the problem to find a vector $x^* \in K \subset \mathbb{R}^n$, such that*

$$\left\langle F(x^*)^T, x - x^* \right\rangle \geq 0, \quad \forall x \in K, \tag{5.2}$$

where F is a given continuous function from K to $\mathbb{R}^n$, K is a given closed convex nonempty set and $\langle \, . \, , \, . \, \rangle$ denotes the inner product in n-dimensional Euclidean space.

In geometric terms (see Figure 5.1), the variational inequality 5.2 states that $F(x^*)^T$ is "orthogonal" to the feasible set K at the point x^*.
We recall that, for two vectors $u, v \in \mathbb{R}^n$, the inner product $\left\langle u^T, v \right\rangle = \|u\|\|v\| cos\theta$, where θ is the angle between the vectors u and v. Hence, for θ in the range: $0 \leq \theta \leq 90°$, we have that $\left\langle u^T, v \right\rangle \geq 0$. Thus, one can see from Figure 5.1 that x^* is a solution of $VI(F,K)$ if and only if the angle between the vectors $F(x^*)^T$ and $x - x^*$, with x and x^* both in K, is a non-obtuse angle, that is: less than or equal to 90°.
This formulation is particularly convenient because it permits a unified treatment of equilibrium problems and optimization problems.

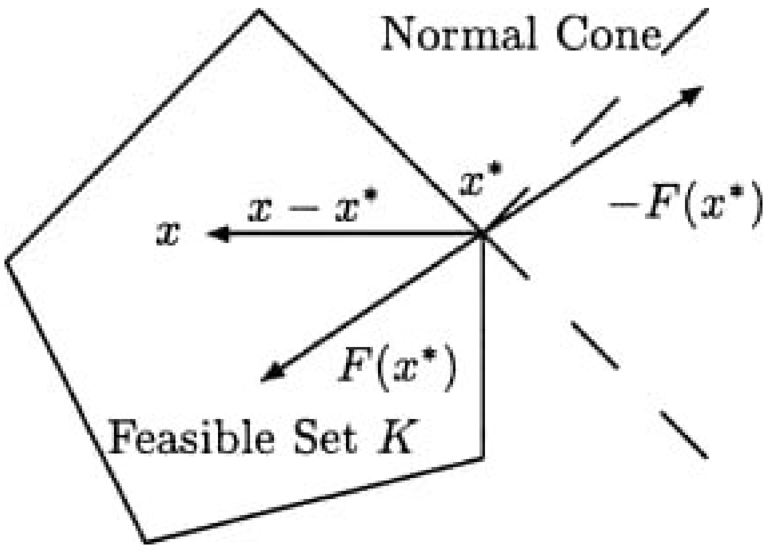

Fig. 5.1: Geometric interpretation of $VI(F,K)$

We may formalize this observation using the concept of the normal cone.
Specifically, associated with the set K and any vector x' belonging to K, we define the *normal cone* to K at x' the following:

$$\mathcal{N}(x', K) = \{d \in \mathbb{R}^n : d^T(x - x') \leq 0, \, \forall x \in K\}.$$

Therefore, the Variational Inequality 5.2 affirms that a vector $x^* \in K$ solves the $VI(F,K)$ if and only if $-F(x^*)$ is a normal vector to K at x^*.

It is interesting to outline the relationships between variational inequalities and optimization problems. Indeed, a variational inequality is related to an optimization problem when the objective function is a primitive of the operator involved in the inequality itself and optimization problems, constrained and not, can be formulated as Variational Inequality Problems.
The connections between minimum problems and the variational inequalities have been widely studied in the case in which K is a convex set and the objective function $f : \mathbb{R}^n \to \mathbb{R}$, defined and differentiable on a open set containing K, is a primitive of F, that is $f'(x) = F(x)$.
The precise connection between the $VI(F,K)$ and the Optimization Problem in described in the following results.

Proposition 5.4. *Let x^* be a solution to the optimization problem:*

$$\begin{aligned} &\min f(x) \\ &\textit{subject to: } x \in K, \end{aligned} \tag{5.3}$$

where f is continuously differentiable and K is closed and convex. Then x^* is a solution to the variational inequality problem:

$$\left\langle \nabla f(x^*)^T, x - x^* \right\rangle \geq 0, \quad \forall x \in K. \tag{5.4}$$

Proof. Let $\phi(t) = f(x^* + t(x - x^*))$, for $t \in [0,1]$. Since $\phi(t)$ achieves its minimum at $t = 0, 0 \leq \phi'(0) = \nabla f(x^*)^T \cdot (x - x^*)$, that is, x^* is a solution to (5.4).

Proposition 5.5. *If $f(x)$ is a convex differentiable function and x^* is a solution to $VI(\nabla f, K)$, then x^* is a solution to the optimization problem (5.3).*

Proof. Since $f(x)$ is convex,

$$f(x) \geq f(x^*) + \nabla f(x^*)^T \cdot (x - x^*), \quad \forall x \in K. \tag{5.5}$$

But $\nabla f(x^*)^T \cdot (x - x^*) \geq 0$, since x^* is a solution to $VI(\nabla f, K)$. Therefore, from (5.5) one concludes that

$$f(x) \geq f(x^*), \quad \forall x \in K,$$

that is, x^* is a minimum point of the mathematical programming problem (5.3).

In the upcoming chapter, our focus will shift towards crafting an advanced model for trust and reputation systems, employing approaches based on variational inequalities. This methodology proves crucial in navigating the intricate dynamics of trust and reputation assessment in digital contexts, where agents interact with diverse entities. We will delve into the application of variational inequalities as a key mathematical tool to model trust relationships, ensuring flexibility in evaluating trust between agents and the entities they engage with. Simultaneously, we will utilize Lagrange multipliers to gain a more profound understanding of the model's solutions, enhancing our ability to analyze and manage the complexities of trust and reputation. The aim is to develop a model that accurately reflects the trust agents place in various entities, contributing to fostering a more reliable and secure online environment.

6

A Variational Formulation for a Trust and Reputation System

The concept of Equilibrium is central in several disciplines, for examples Mathematics, Economics, Engineering and so on. Many methodologies have been applied to formulate the model describing an equilibrium problem, among these the optimization theory, the fixed point theory, the complementarity theory.

Recently, there has been a sharp increase in interest in variational inequalities ([70, 71]), that have become one of the most challenging and dynamic topics of mathematics and represent an excellent tool in the study of real-world problems. Indeed, they cover a large variety of applications of extreme importance related to computer science, mathematical physics, engineering, statistics, economics, financial networks, and generalized complementarity problems.

Variational inequalities theory is a powerful instrument, that has been used to study a large variety of equilibrium problems, to analyze them in terms of existence, uniqueness and stability of the solution to provide a sensitivity analysis and algorithms for the calculus of the solution.

The kind of equilibrium, we are dealing with in the thesis, is not only the one coming from the minimization of a functional such as energy or utility, but also the one coming from another concept of equilibrium, which, in the economic case, is called "user equilibrium". This type of equilibrium usually leads to a variational inequality, or, to a quasi-variational inequality. Examples of such a formulation, that have been already partially reached, are the financial equilibrium problem, the economic markets, the traffic equilibrium problem, the spatial price equilibrium problem, oligopolistic market problem, pollution emission price problem, obstacle problem, Signorini problem, elastic-plastic torsion problem, Walrasian, migration problem and many others (see [72–82]).

Then, in the thesis our contribution is to develop an effective equilibrium model in trust and reputation systems by means of the variational inequalities theory.

In today's digital age, consumers are no longer passive recipients of products and services; they have a significant role to play in shaping the market. One of the most impactful ways they can contribute is by voting for products. Whether it's a simple star rating or a detailed review, user votes have become a driving force behind the success and improvement of products and services. Indeed, a user vote and review can help potential buyers make informed decisions, ensuring they get the best value for their money. In such a way, the user becomes a trusted source of information for others. On the other hand, by expressing a personal satisfaction or disappointment, users contribute to maintaining high standards in the market. Companies often pay close attention to user feedback to make necessary improvements. Moreover, when users vote for products and point out their strengths and weaknesses, businesses have the opportunity to refine and enhance their offerings.

6.1 Literature Review

As already recalled in the introduction of the thesis, Trust and Reputation systems have been implemented in social networks, thematic social networks, e-commerce platforms, comparison shopping websites.

They also find applications in many other different domains, as wireless sensor networks [83], energy optimization [84], security for vehicular networks [85], Internet of Things [86] and so on.

Several techniques have been applied to further understand and improve these systems. In [87] the authors propose to use fuzzy techniques in the design of reputation systems based on collecting and aggregating peers' opinions. Such techniques are compared with probabilistic approaches. In [88] an iterative computing of the reputation in terms of weighted trust is introduced together with an algorithm, that linearly converges to the unique vector of reputations.

A game theory approach is provided in [89–91] in different frameworks.

In [92] the author describes a new paradigm for modeling traffic levels on the world wide web (WWW), using a maximization method. The objective function is the famous entropy function and a fast iterative algorithm for computing the variables is provided. Tomlin specifically applies the primal and dual solutions of this model to the ranking (reputation) of web sites. The first of these uses an imputed measure of total traffic through a web page, the second provides an analogy of local "temperature", allowing to quantify the "HOTness" of a page (see also [93]).

Finally, we refer to [94, 95] for Trust and Reputation Model Based on Bayesian Networks.

Our contributions to the literature lie in proposing a different approach to trust and reputation systems, in which n users vote m objects. In particular, to the best of our knowledge, for the first time we formulate the problem as a variational one, allowing to apply the variational inequality theory to analyze it.

6.2 The Mathematical model (for users and objects)

In this Section, we present the mathematical model of a trust and reputation system, where users provide votes or ratings to express their satisfaction or dissatisfaction with products.

Assume we have:

- n users, where the typical one is denoted by i, and
- m objects, where the typical one is denoted by j.

In a trust and reputation model, users can indeed provide feedback or ratings using either stars (symbolic representation) or real numbers (numeric representation). This dual approach allows users to express their opinions and evaluations in a way that suits their preferences and the platform's design. Both methods serve the same purpose of collecting user feedback and assessing trust or reputation. Using stars can make it simpler and more intuitive for users to provide quick feedback, while numeric ratings allow for more granularity and precision in expressing opinions. In our model we assume that users can give a numerical rating, typically on a scale (e.g., 0 to 1), where a higher number signifies a higher level of trust or reputation, and a lower number indicates the opposite.
Therefore, let:

- $v_{ij} \in [0,1]$ be the trust value assigned by user i to object j;
- a_i be the weight associated with user i; it is a kind of reliability of user i in order to weight the trusts he has to assign; such a weight could be equal for all users, as in Ebay, or can be higher for the founders of the social network, as in Eigentrust;
- $w_{ij} = a_i v_{ij}$ be the weighted trust assigned by i to j; we remark that, since each object may obtain a trust value rate from zero to one, we have the following capacity constraints:

$$0 \le w_{ij} \le a_i, \quad \forall i = 1,\dots,n,\ \forall j = 1,\dots,m; \tag{6.1}$$

- b_{ij} be a boolean parameter which indicates whether object j is rated by user i ($b_{ij} = 1$ if user i rates object j, $b_{ij} = 0$ otherwise). Observe that a user i is taken into account if she/he assigns a rating to at least one object and, similarly, an object j is considered if it receives at least one rating from users.

As previously mentioned, trust (or weighted trust) and reputation are strictly connected. Such a connection lies in how the trust values, especially when weighted differently, can impact and contribute to an object's overall reputation within a network or community. Indeed, high levels of trust values, especially from sources with significant weight or credibility in a network, can positively impact an object's reputation. Recommendations or endorsements from highly trusted sources are likely to enhance an object's reputation more significantly than those from less credible sources. Moreover, reputation, being a collective evaluation, affects how much trust an object might garner from users. A positive reputation implies that the object is reliable, that is, it conforms to the description and expectations of the user who purchases it, leading others to be more likely to purchase it. Therefore, if an object is "new" in the system (that is, it has never been rated before), the reputation of object j, denoted by r_j, is related to the weighted trust values it receives, and it is defined as the (arithmetic) average of all the weighted trust values obtained by j:

$$r_j = \frac{\sum_i w_{ij}}{\sum_i a_i b_{ij}}. \tag{6.2}$$

Observe that, since we assumed that each object is rated by at least one user, the denominator is never zero.
On the other hand, if $\hat{r}_j$ is the initial reputation of object j, obtained by the valuation of N_j users, and denoting by $B_j = \sum\limits_i b_{ij}$ the number of new ratings, the final reputation R_j is given by the average of all the weighted trust values (both, the previous ones and the current ones), as follows:

$$R_j = \frac{\hat{r}_j N_j + r_j B_j}{N_j + B_j}. \tag{6.3}$$

For our purposes, we will assume that the initial reputation of each individual object is non-null, as assigning a zero initial reputation to an object would be of limited significance within our system. Moreover, we now introduce the utility and cost functions. Let:

- $\hat{u}_j$ be the utility function of object j in terms of the final reputation R_j, in terms of the variable r_j (see (6.3)), that is $\hat{u}_j = \hat{u}_j(r_j)$; moreover, since the reputation is obtained by the weighted trust values (see (6.2)), we may express the utility of j in terms of w, namely, $\hat{u}_j(r_j) = u_j(w)$, where $w = (w_{ij})_{\substack{i=1,\dots,n \\ j=1,\dots,m}}$;
- c_{ij} be the cost of i to valuate j; such a cost can be interpreted as an expense in terms of time to buy an object or to go to a restaurant/hotel to evaluate it, or as the payment of a fee to access evaluation tools or it can be the damage incurred by object j depending on a low trust from i. In any case object j aims at a low value of c_{ij} in order to have a trust from i or to have a low damage. We assume that such a cost depends on w: $c_{ij} = c_{ij}(w)$;
- $\hat{c}_j(r_j)$ be the cost payed by object j to increase its initial reputation. It depends on the difference between the final and initial reputation $(R_j - \hat{r}_j)$, therefore, it

depends on the reputation r_j. It could be interpreted as an economic cost, in terms of advertisements, taxes or a cost in terms of time and efforts, which allows the object j to improve its reputation. Also in this case, since equation (6.2) holds, we have that $\hat{c}_j(r_j) = c_j(w)$.

We assume that the utility function $\hat{u}_j$ and the cost functions c_{ij} and $\hat{c}_j(r_j)$ are all continuous.

In order to define the equilibrium conditions for trust and reputation systems (which allow us to determine the equilibrium weighted trust values), we introduce the Lagrange multipliers λ_{ij}, $i = 1,\dots,m$, $j = 1,\dots,n$, associated with the capacity constraints $w_{ij} \leq a_i$, and α_{ij}, $i = 1,\dots,m$, $j = 1,\dots,n$, associated with the capacity constraints $w_{ij} \geq 0$, respectively.

Then, we provide the following definition.

Definition 6.1 (Equilibrium Definition).

*A vector $(w^*, \lambda^*, \alpha^*) \in K^1 \equiv \mathbb{R}_+^{n\times m} \times \mathbb{R}_+^{n\times m} \times \mathbb{R}_+^{n\times m}$ is an **equilibrium** if it satisfies the following equilibrium conditions:*

$$u_j(w^*) - c_{ij}(w^*) - c_j(w^*) \begin{cases} \leq \lambda_{ij}^* - \alpha_{ij}^* & \text{if } w_{ij}^* = 0 \\ = \lambda_{ij}^* - \alpha_{ij}^* & \text{if } 0 < w_{ij}^* < a_i \\ \geq \lambda_{ij}^* - \alpha_{ij}^* & \text{if } w_{ij}^* = a_i \end{cases} \quad \forall i = 1,\dots,n \text{ and } j = 1,\dots,m, \tag{6.4}$$

$$\lambda_{ij}^* \begin{cases} \geq 0 & \text{if } w_{ij}^* = a_i \\ = 0 & \text{if } w_{ij}^* < a_i \end{cases} \quad \forall i = 1,\dots,n \text{ and } j = 1,\dots,m \tag{6.5}$$

and

$$\alpha_{ij}^* \begin{cases} \geq 0 & \text{if } w_{ij}^* = 0 \\ = 0 & \text{if } w_{ij}^* > 0 \end{cases} \quad \forall i = 1,\dots,n \text{ and } j = 1,\dots,m. \tag{6.6}$$

Conditions (6.4)-(6.6) mean that:
if j gets a null rating ($w_{ij}^* = 0$), then j has non-positive net gain, that is:

$$u_j(w^*) - c_{ij}(w^*) - c_j(w^*) \leq -\alpha_{ij}^* \leq 0, \tag{6.7}$$

since $\alpha_{ij}^* \geq 0$ and $\lambda_{ij}^* = 0$; if j gets a good rating from i, which is not the case corresponding to a zero or maximum vote, then j has neither losses nor gains:

$$u_j(w^*) = c_{ij}(w^*) + c_j(w^*); \tag{6.8}$$

lastly, if j gets the maximum rating from i ($w_{ij}^* = a_i$), then j has a higher gain in utility, that is:

$$u_j(w^*) \geq c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} \tag{6.9}$$

and $\lambda^*_{ij} \geq 0$ if j obtains from $i = 1, \dots, n$ the highest rate.

We now present the variational inequality formulation of the equilibrium conditions (6.4)-(6.6).

Theorem 6.2. *A vector $(w^*, \lambda^*, \alpha^*) \in K^1$ is an equilibrium if and only if it satisfies the following variational inequality*

$$\begin{aligned}&\sum_{i=1}^{n}\sum_{j=1}^{m}(-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij}) \times (w_{ij} - w^*_{ij})\\ &+\sum_{i=1}^{n}\sum_{j=1}^{m}(a_i - w^*_{ij}) \times (\lambda_{ij} - \lambda^*_{ij}) + \sum_{i=1}^{n}\sum_{j=1}^{m}(w^*_{ij}) \times (\alpha_{ij} - \alpha^*_{ij}) \geq 0, \quad \forall (w, \lambda) \in K^1\end{aligned} \tag{6.10}$$

Proof. $\Rightarrow$) Let $(w^*, \lambda^*) \in K^1$ be an equilibrium, then $\forall i = 1, \dots, n$ and $\forall j = 1, \dots, m$ we have

$$(-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij}) \times (w_{ij} - w^*_{ij}) \geq 0.$$

Indeed, for a fixed i, j we obtain that:

- if $w^*_{ij} = 0 \quad \Rightarrow$ then $-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij} \geq 0$ and $w_{ij} - w^*_{ij} = w_{ij} \geq 0$, namely
$$(-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij}) \times (w_{ij} - w^*_{ij}) \geq 0;$$
- if $0 < w^*_{ij} < a_i \quad \Rightarrow$ then
$$-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij} = 0;$$
- if $w^*_{ij} = a_i \quad \Rightarrow$ then $-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij} \leq 0$ and $w_{ij} - w^*_{ij} = w_{ij} - a_i \leq 0$, namely
$$(-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda^*_{ij} - \alpha^*_{ij}) \times (w_{ij} - w^*_{ij}) \geq 0.$$

If we sum, $\forall i = 1, \dots, n$ and $\forall j = 1, \dots, m$ we have:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(-u_j(w^*) + c_{ij}(w^*) + c_j(w^*) + \lambda_{ij} - \alpha^*_{ij}) \times (w_{ij} - w^*_{ij}) \geq 0, \quad \forall w \in \mathbb{R}^{n \times m}_+, \quad 0 \leq w_{ij} \leq a_i.$$

Moreover, since $(w^*, \lambda^*, \alpha^*) \in K^1$ is an equilibrium, for a fixed i, j we obtain that:

- if $w^*_{ij} - a_i < 0 \quad \Rightarrow \lambda^*_{ij} = 0 \quad \Rightarrow (a_i - w^*_{ij}) \times (\lambda_{ij} - \lambda^*_{ij}) \geq 0$;
- if $a_i - w^*_{ij} = 0 \Rightarrow (a_i - w^*_{ij}) \times (\lambda_{ij} - \lambda^*_{ij}) = 0$.

If we sum, $\forall i = 1, \dots, n$ and $\forall j = 1, \dots, m$ we obtain that:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(a_i - w^*_{ij}) \times (\lambda_{ij} - \lambda^*_{ij}) \geq 0.$$

Furthermore, we have that:

- if $w^*_{ij} > 0 \quad \Rightarrow \alpha^*_{ij} = 0 \quad \Rightarrow (w^*_{ij}) \times (\alpha_{ij} - \alpha^*_{ij}) \geq 0$;
- if $w^*_{ij} = 0 \Rightarrow (w^*_{ij}) \times (\alpha_{ij} - \alpha^*_{ij}) = 0$.

If we sum, $\forall i = 1,\dots,n$ and $\forall j = 1,\dots,m$ we obtain that:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(w^*_{ij}) \times (\alpha_{ij} - \alpha^*_{ij}) \geq 0,$$

and then, variational inequality (6.10) holds.

$\Leftarrow$)

Now, suppose that (w^*, λ^*) is a solution to variational inequality (6.10). Set $\lambda_{ij} = \lambda^*_{ij}$ and $\alpha_{ij} = \alpha^*_{ij}$ $\forall i,j$, and $w_{ij} = w^*_{ij}$, $\forall i,j$ except for $i = \bar{i}$ e $j = \bar{j}$. Variational inequality (6.10) reduces to:

$$(-u_{\bar{j}}(w^*) + c_{\bar{i}\bar{j}}(w^*) + c_{\bar{i}}(w^*) + \lambda^*_{\bar{i}\bar{j}} - \alpha^*_{\bar{i}\bar{j}}) \times (w_{\bar{i}\bar{j}} - w^*_{\bar{i}\bar{j}}) \geq 0$$

and equilibrium condition (6.4) holds.

If we set $w_{ij} = w^*_{ij}$ and $\alpha_{ij} = \alpha^*_{ij}$ $\forall i,j$, and $\lambda_{ij} = \lambda^*_{ij}$, $\forall i,j$ except for $i = \bar{i}$ e $j = \bar{j}$, then variational inequality (6.10) reduces to:

$$(a_{\bar{i}} - w^*_{\bar{i}\bar{j}}) \times (\lambda_{\bar{i}\bar{j}} - \lambda^*_{\bar{i}\bar{j}}) \geq 0,$$

and equilibrium condition (6.5) holds.

Similarly, if we set $w_{ij} = w^*_{ij}$ and $\lambda_{ij} = \lambda^*_{ij}$ $\forall i,j$, and $\alpha_{ij} = \alpha^*_{ij}$, $\forall i,j$ except for $i = \bar{i}$ e $j = \bar{j}$, then variational inequality (6.10) reduces to:

$$(w^*_{\bar{i}\bar{j}}) \times (\alpha_{\bar{i}\bar{j}} - \alpha^*_{\bar{i}\bar{j}}) \geq 0,$$

and equilibrium condition (6.6) holds.

6.3 An equivalent formulation

In this Section, we provide an alternative variational inequality formulation, in which the Lagrange multipliers do not appear.
Let us consider the feasible set:

$$K^2 = \left\{ w \in \mathbb{R}^{n\times m}_{+} : \; w_{ij} \geq 0, \; w_{ij} \leq a_i, \quad \forall i = 1,\dots,n, \text{ and } \forall j = 1,\dots,m \right\},$$

and the following variational inequality:

Find $w^{**} \in K^2$ such that

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(-u_j(w^{**}) + c_{ij}(w^{**}) + c_j(w^{**})) \times (w_{ij} - w^{**}_{ij}) \geq 0, \quad \forall w \in K^2. \tag{6.11}$$

Theorem 6.3. *The solution w^{**} to variational inequality (6.11) also satisfies equilibrium conditions (6.4), (6.5), and (6.6), where λ^*_{ij} is the Lagrange multiplier associated with constraints $a_i - w^*_{ij} \geq 0$, and α^*_{ij} is the Lagrange multiplier associated with constraints $w^*_{ij} \geq 0$, $\forall i, j$.*

Proof. If we set

$$f(w) = \sum_{i=1}^{n} \sum_{j=1}^{m} (-u_j(w^{**}) + c_{ij}(w^{**}) + c_j(w^{**})) \times (w_{ij}),$$

since w^{**} is the solution to (6.11), then

$$f(w) \geq f(w^{**}) \quad \forall w \in K^2,$$

namely, w^{**} is a minimum point for f in K^2.

Since the constraints are linear, from KKT conditions it follows:

$$\begin{cases} -u_j(w^{**}) + c_{ij}(w^{**}) + c_j(w^{**}) - \ddot{\alpha}_{ij} + \lambda^*_{ij} = 0 & \forall i, j \\ \ddot{\alpha}_{ij}(-w^{**}_{ij}) = 0, & \ddot{\alpha}_{ij} \geq 0 \\ \lambda^*_{ij}(w^{**}_{ij} - a_i) = 0, & \lambda^*_{ij} \geq 0. \end{cases}$$

and, then, the equilibrium conditions hold.

The converse part also holds true and, then, the two formulations are equivalent.

Following the classical theory of variational inequalities (see, for instance, [96]), the existence of a solution to variational inequality (6.11) is guaranteed since the underlying feasible set K^2 is compact and the functions u_j, c_{ij} and c_j are assumed continuous. Furthermore, it follows that, if the function $F(w) = -u_j(w) + c_{ij}(w) + c_j(w)$ is strictly monotone, then the solution is unique. It is worth noting that the sum of a monotone function and a strictly monotone one is also strictly monotone. Hence, not all the functions in variational inequality (6.11) need to be strictly monotone for uniqueness of the equilibrium to hold. For the safe of clarity we recall the following existence and uniqueness results ([96]).

Theorem 6.4 (Existence). *Variational inequality (6.11) admits at least a solution if u_j, c_{ij} and c_j are continuous functions (since K^2 is a compact and convex set).*

Theorem 6.5 (Uniqueness). *Under the assumptions of Theorem 6.4, if the function $F(w) = -u_j(w) + c_{ij}(w) + c_j(w)$ in* (6.11) *is strictly monotone on K^2, that is:*

$$\langle (F(w_1) - F(w_2))^T, w_1 - w_2 \rangle > 0, \quad \forall w_1,\ w_2 \in K^2, w_1 \neq w_2,$$

then variational inequality (6.11) *admits a unique solution.*

6.4 Numerical simulations

We now provide some numerical simulations, in order to study the trust and reputation system and to gain insights into the behavior of the proposed model. We first analyze some numerical simulations paying particular attention to the optimal values assumed by the Lagrange multipliers; then we report the optimal solutions obtained by solving a more realistic simulation in terms of problem size (that is, with a higher number of users and objects). Furthermore, we also performed some sensitivity analysis. The optimal results are computed by solving the variational inequality (6.10), given in the previous section, implementing the projection method in Matlab on an LG laptop with a 12th Gen Intel(R) Core(TM) i7-1260P, 16 GB RAM.

6.4.1 Simulation 1 (with Lagrange multipliers)

The first simulation consists of $n = 3$ users and $m = 4$ objects.
The small number of users and objects in this simulation is justified in order to clearly illustrate the results. In particular, in this first simulation we focus on the optimal values of the Lagrange multipliers. Therefore, we solve Variational Inequality (6.10).

All the used parameters are reported in Table 6.1.

Users' weight	$a_1 = 0.6, a_2 = 0.4, a_3 = 0.3$
Boolean parameters	$b_{11} = b_{13} = 1,\ b_{12} = b_{14} = 0$ $b_{2j} = 1,\ \forall j = 1,\dots,4$ $b_{32} = 1,\ b_{31} = b_{33} = b_{34} = 0$
Initial reputations	$\hat{r}_1 = 0.9, \hat{r}_2 = 0.6, \hat{r}_3 = 0.2, \hat{r}_4 = 0.3$
Number of initial rates	$N_1 = 2, N_2 = 1, N_3 = 3, N_4 = 4,$

Table 6.1: Parameters used in Simulation 1

The chosen utility and cost functions are the following (but any type of expression which satisfies the assumption could be used):

- the utility functions $\hat{u}_j(r_j) = u_j(w)$ are assumed to be increasing by varying the reputation $r_j, \forall j = 1,\dots,m$:

$$\hat{u}_j(r_j) = \rho_j r_j + k = \rho_j \frac{\sum_i w_{ij}}{\sum_i a_i b_{ij}} + k,$$

  where $\rho_j, k > 0$.
- the costs $c_{ij}(w)$ are assumed to be decreasing by varying the weighted trust:

$$c_{ij}(w) = \alpha_{ij} - \beta_{ij} w_{ij},$$

where $\alpha_{ij} = D_{ij} > 0$, $\beta_{ij} = \frac{D_{ij}}{\hat{r}_j} > 0$ and D_{ij} represents the maximum damage incurred by object j when receives the minimum trust from i, namely if $w_{ij} = 0$, then $c_{ij} = D_{ij}$. Moreover, if object j receives from i the same trust as its initial reputation, the damage is null: if $w_{ij} = \hat{r}_j$, then $c_{ij} = 0$. Clearly, $w_{ij} < \hat{r}_j$ means that object j incurs in a damage (since $c_{ij}(w) > 0$); otherwise (if $w_{ij} > \hat{r}_j$) object j gets a profit.

- the cost payed by each object to increase its initial reputation $\hat{c}_j(r_j) = c_j(w)$ depends on the difference between the final and initial reputation, that is $R_j - \hat{r}_j = \dfrac{(r_j - \hat{r}_j)B_j}{N_j - B_j}$:
$$\tilde{c}_j(R_j - \hat{r}_j) = \tilde{c}_j\left(\frac{(r_j - \hat{r}_j)B_j}{N_j - B_j}\right) = \delta_j\left(\frac{(r_j - \hat{r}_j)B_j}{N_j - B_j}\right) + \gamma_j,$$
where $\delta_j, \gamma_j > 0$ and we assume that such costs are all positive.

We established all the parameters of the functions, for all the simulations, so that all the assumptions described above are satisfied (positive coefficients, positive function $\hat{c}_j(r_j)$) and the function $F(w)$ in (6.11) is strictly monotone on K^2. Furthermore, for the first simulation S1.1, we used high values for the parameters of the utility function: $\rho_j = 10$, $\forall j = 1,\ldots,4$, and $k = 7$. On the contrary, we used low values for simulations S1.2 and S1.3 ($\rho_j = 0.5$, $\forall j = 1,\ldots,4$ for both the two simulations, and $k = 1$ for S1.2 and $k = 0.7$ for S1.3). Simulations S1.1, S1.2 and S1.3 have the same following parameters (related to the cost functions): $D_{ij} = 0.7$, $\forall i, \forall j$ (hence, $\alpha_{ij} = D_{ij} = 0.7$, $\forall i, \forall j$ and $\beta_{i1} = \frac{D_{ij}}{\hat{r}_1} = 0.78$, $\beta_{i2} = \frac{D_{ij}}{\hat{r}_2} = 1.17$, $\beta_{i3} = \frac{D_{ij}}{\hat{r}_3} = 3.50$ and $\beta_{i4} = \frac{D_{ij}}{\hat{r}_4} = 2.33$); $\gamma_1 = 0.8192$, $\gamma_2 = 0.5444$, $\gamma_3 = 3.5800$, $\gamma_4 = 1.2600$; $\delta_1 = 1.2385$, $\delta_2 = 0.5167$, $\delta_3 = 8.4500$, $\delta_4 = 5.3000$.

Simulation 1 consists of 36 variables and the optimal solutions are obtained in less than one second. The optimal solutions are reported in Table 6.2.

We can see that in Simulation 1.1 all the optimal solutions assume their maximum value, that is $w^*_{ij} = \bar{s}_i = a_i$. In accordance with the equilibrium condition (6.6), since $w^*_{ij} > \underline{s}$, we obtain that $\alpha^*_{ij} = 0$, $\forall i = 1,\ldots,3$ and $\forall j = 1,\ldots,4$. Moreover, we obtain that $\lambda^*_{ij} = 0.5 > 0$, $\forall i = 1,\ldots,3$ and $\forall j = 1,\ldots,4$, as established by condition (6.5). Lastly, according to the first condition (6.4) of the Equilibrium Definition, we obtain that since all the optimal solutions assume their maximum value, all the objects have a positive gain in utility (see Table 6.3 for the difference between the utility and the costs).

In Simulation 1.2, some optimal variables take on minimum values and others take on maximum values (see Table 6.2). In this case, if the optimal solution is the minimum ($w^*_{ij} = 0$), then $\lambda^*_{ij} = 0$ and the object has a loss (that is a negative value in Table 6.3). On the contrary, if the optimal solution is the maximum ($w^*_{ij} = a_i$), then $\alpha^*_{ij} = 0$

		$i=1$				$i=2$				$i=3$			
		$j=1$	$j=2$	$j=3$	$j=4$	$j=1$	$j=2$	$j=3$	$j=4$	$j=1$	$j=2$	$j=3$	$j=4$
	w^*_{ij}	0.60	0.60	0.60	0.60	0.40	0.40	0.40	0.40	0.30	0.30	0.30	0.30
S1.1	λ^*_{ij}	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	α^*_{ij}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	w^*_{ij}	0.60	0.60	0.00	0.00	0.00	0.40	0.00	0.00	0.30	0.30	0.00	0.00
S1.2	λ^*_{ij}	0.23	0.50	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.50	0.00	0.00
	α^*_{ij}	0.00	0.00	0.50	0.39	0.00	0.00	0.50	0.40	0.00	0.00	0.50	0.45
	w^*_{ij}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
S1.3	λ^*_{ij}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	α^*_{ij}	0.07	0.02	0.50	0.43	0.20	0.21	0.50	0.43	0.05	0.16	0.50	0.45
	w^*_{ij}	0.50	0.50	0.50	0.50	0.30	0.30	0.30	0.30	0.20	0.20	0.20	0.20
S1.4	λ^*_{ij}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	α^*_{ij}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 6.2: Optimal solutions of Simulation 1

		$j=1$	$j=2$	$j=3$	$j=4$
	$i=1$	70.7008	98.7497	66.1020	166.0400
S1.1	$i=2$	70.5470	98.5941	65.4020	165.4800
	$i=3$	70.4700	98.5163	65.0520	165.2000
	$i=1$	0.3985	0.8211	-2.6040	-0.6950
S1.2	$i=2$	-0.0630	0.6656	-2.6040	-0.6950
	$i=3$	0.1677	0.5878	-2.6040	-0.6950
	$i=1$	-0.2557	-0.2344	-2.9040	-0.9950
S1.3	$i=2$	-0.2557	-0.2344	-2.9040	-0.9950
	$i=3$	-0.2557	-0.2344	-2.9040	-0.9950
	$i=1$	0.00	0.00	0.00	0.00
S1.4	$i=2$	0.00	0.00	0.00	0.00
	$i=3$	0.00	0.00	0.00	0.00

Table 6.3: Difference between the utility and the sum of costs: $u_j(w) - c_{ij}(w) - c_j(w)$.

and the object has a gain.

In Simulation 1.3, we can observe that all the optimal weighted trusts are null, as well as all the optimal Lagrange multipliers related to the upper bound constraints, while the optimal Lagrange multipliers related to the lower constraints are non-zero. Finally, in Simulation 1.4, we can note that all the optimal values are neither zero nor maximum and all the Lagrange multipliers are null. We obtain this case, according to the conditions of Equilibrium Definition (particularly, the second condition of (6.4), (6.5) and (6.6)), because the utility equals the sum of costs (see Table 6.3).

		$i = 1$				$i = 2$				$i = 3$			
		$j = 1$	$j = 2$	$j = 3$	$j = 4$	$j = 1$	$j = 2$	$j = 3$	$j = 4$	$j = 1$	$j = 2$	$j = 3$	$j = 4$
S1.1	v^*_{ij}	1	1	1	1	1	1	1	1	1	1	1	1
S1.2	v^*_{ij}	1	1	0	0	0	1	0	0	1	1	0	0
S1.3	v^*_{ij}	0	0	0	0	0	0	0	0	0	0	0	0
S1.4	v^*_{ij}	0.83	0.83	0.83	0.83	0.75	0.75	0.75	0.75	0.67	0.67	0.67	0.67

Table 6.4: Optimal trust values

In Table 6.4 the optimal trust values are reported. Clearly, we obtain the maximum trust value when the weighted trust is maximum, zero when the weighted trust is zero, otherwise we obtain a value between 0 and 1.

6.4.2 Simulation 2 (with a realistic community)

As an additional simulation, we choose to apply our mathematical model to the study of trust and reputation systems in a realistic and restricted community of 100 users that evaluate 60 objects. A small community can enhance the overall experience by promoting positive behavior, fostering trust, and providing a mechanism for community members to identify and engage with reliable contributors. Knowing that their actions are being tracked and will contribute to their overall reputation, community members are likely to engage in positive behaviors. This can include providing helpful information, contributing to discussions, and supporting others. Moreover, communication could be more direct and users could have more influence on community dynamics by facilitating quality control of discussions and management of the misleading or fraudulent behavior (these behaviors will be the subject of future studies).

Due to the manageable size, we assume that each user evaluates all objects. The weight a_i associate with each user i, $\forall i = 1, \dots, 100$ ranges from 0.1 to 0.7. The initial reputations $\hat{r}_j$ of objects $j = 1, \dots, 60$ are variable between 0.1 and 0.6, instead the number of users N_j, that initially rated the objects, ranges from 30 to 50. Finally, the number of variables is 6000, the number of constraints is 12000, the number of parameters used is 18402 and all of them satisfy the assumptions described previously. Figure (6.1) shows the optimal solutions of the system.

In particular, we obtained 1100 solutions equal to zero, 700 solutions equal to 0.1, i.e. the minimum reliability value, instead 580 solutions equal to 0.7 i.e. the maximum reliability value possessed. Finally, the remaining solutions vary between intermediate values. The run times were about 100 seconds.

In a broader context, it is worth noting that the ability to address equilibrium problems in scenarios with a substantial number of users and objects can be achieved

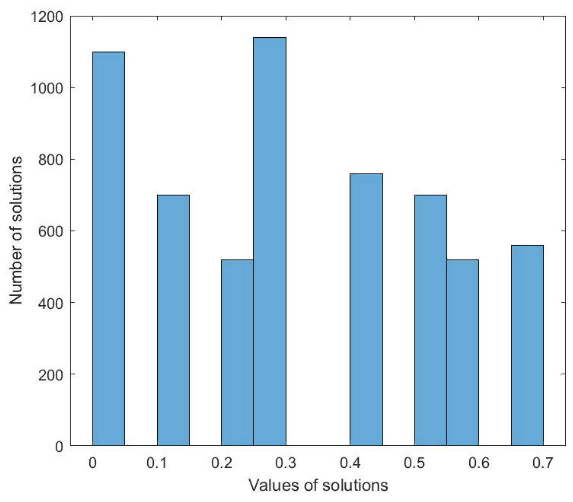

Fig. 6.1: Range of Optimal Solutions

by leveraging devices with larger memory count while still maintaining reasonable computation times.

6.4.3 Simulation 3: Sensitivity analysis

In order to evaluate the impact of changes in input parameters on the outcomes of our system, we performed some sensitivity analysis. Particularly, we first investigated on changes on the weight associated with the third user, then on the initial reputations of the objects.

Simulation 3.1 (sensitivity analysis on a_i)

We performed seven additional simulations, in order to investigate how sensitive the results are to variations in a_i, that is the weight associated with each user i, $\forall i = 1,\dots,n$. We used the same values for parameters and functions as in Simulation 1. In the different simulations we kept the value relating to the weight associated with the first two users constant, whereas we changed the value of the weight associated with the third user, in an increasing manner, as follows: $a_1 = 0.6$, $a_2 = 0.4$ and $a_3 \in \{0.1, 0.4, 0.8, 1.2, 2, 2.8, 3\}$.

We reported the optimal solutions (w^*_{ij}) in Table 6.5. The results show that the optimal values of the variables related to $i = 1$ and $i = 2$ do not change. On the contrary, the optimal values of the variables related to $i = 3$ could change. More specifically, we note that:

- w_{31} is always zero;

w^*_{ij}	$i=1$				$i=2$				$i=3$			
	$j=1$	$j=2$	$j=3$	$j=4$	$j=1$	$j=2$	$j=3$	$j=4$	$j=1$	$j=2$	$j=3$	$j=4$
S3.1	0.00	0.00	0.60	0.60	0.00	0.40	0.40	0.40	0.00	0.10	0.10	0.10
S3.2	0.00	0.60	0.60	0.60	0.00	0.40	0.40	0.40	0.00	0.40	0.40	0.40
S3.3	0.00	0.60	0.60	0.60	0.00	0.40	0.40	0.40	0.00	0.80	0.80	0.80
S3.4	0.00	0.60	0.60	0.60	0.00	0.40	0.40	0.40	0.00	0.73	1.20	1.20
S3.5	0.00	0.00	0.60	0.60	0.00	0.00	0.40	0.40	0.00	0.00	2.00	0.00
S3.6	0.00	0.00	0.60	0.60	0.00	0.00	0.40	0.40	0.00	0.00	2.80	2.80
S3.7	0.00	0.00	0.60	0.60	0.00	0.00	0.40	0.40	0.00	0.00	3.00	3.00

Table 6.5: Sensitivity analysis on a_i

- $w^*_{32} = a_3$ for the first three simulations (that is, until $a_i \leq 0.80$), $w^*_{32} = 0.73$ in simulation S3.4 (that is, the optimal value is between the maximum and the minimum, excluded) and $w^*_{32} = 0$ in the last three simulations;
- $w^*_{33} = a_3$ for all the simulations;
- $w^*_{34} = a_3$ for all the simulations, except S3.5 where the optimal solution is zero.

Simulation 3.2 (sensitivity analysis on $\hat{r}_j$)

We also performed a sensitivity analysis on $\hat{r}_j$, that is the initial reputation of each object. The results show that if the initial reputation of the first object $\hat{r}_1$ is less than or equal to 0.13, then the optimal solutions w^*_{i1} are zero, otherwise (namely, if $\hat{r}_1 \geq 0.135$) the optimal solutions w^*_{i1} equal the maximum (the weight associated with user i). This equivalently holds for all the users. It is obvious that, if the initial reputations of objects $j = 2, 3, 4$ do not change, the values of the related variables remain constant.
Similarly, if the initial reputation of the second object $\hat{r}_2$ is less than or equal to 0.155, then the optimal solution w^*_{12} is zero, otherwise (namely, if $\hat{r}_2 \geq 0.16$) the optimal solution w^*_{12} equals the maximum. Observe that such a threshold holds only for the first user $i = 1$, while $w^*_{i2} = a_i$, $\forall i = 2, 3$.
The results also show that if the initial reputation of the third object $\hat{r}_3$ is less than or equal to 0.14, then the optimal solutions w^*_{i3} are zero, otherwise (namely, if $\hat{r}_3 \geq 0.145$) the optimal solutions w^*_{i3} equal the maximum (the weight associated with user i, for all the users).
Finally, if the initial reputation of the last object $\hat{r}_4$ is less than or equal to 0.040, then the optimal solutions w^*_{i4} are zero, otherwise (namely, if $\hat{r}_4 \geq 0.045$) the optimal solutions w^*_{i4} equal the maximum (the weight associated with user i).

Therefore, the performed sensitivity analysis on the initial reputations of all the objects allows us to identify the threshold at which a change in such an input parameter causes a significant shift in the results. More specifically, the results of such simulations mean that if the initial reputation of an object is very small (that is, less than a certain threshold), the object does not have a positive gain, but has a loss and even trying to increase the reputation (or, equivalently, the weighted trust) does not lead to a profit, hence, it is convenient not to invest and keep the weighted trust null. On the contrary, if the initial reputation is greater than a certain value, the object has a profit and it is convenient to invest in order to put the weighted trust at their maximum values (obtaining the maximum gain).

6.5 Observations

The establishment of trust and reputation systems where users actively contribute by voting or rating products plays a pivotal role in shaping the modern digital landscape. These systems not only assist consumers in making informed decisions but also cultivate accountability among product or service providers. The democratization of opinions through user-generated ratings empowers individuals to influence and navigate their choices in an increasingly interconnected world.

Variational formulations emerge as a fundamental framework in understanding and modeling trust and reputation systems due to their inherent flexibility and ability to capture intricate dynamics. Leveraging variational methods allows for a comprehensive representation of user behaviors, product evaluations, and network interactions within a unified mathematical framework.

Moreover, the application of variational inequalities within trust and reputation systems provides a formalism that enables the modeling of interactions among users and products, considering the competitive nature of influence and reputation dynamics.

The development and presentation of the variational formulation for a trust and reputation system in this paper offer a promising framework for enhancing the reliability and effectiveness of online interactions. By leveraging variational methods, we have provided a novel approach that integrates trust and reputation mechanisms into a unified framework, allowing for a more robust assessment of trustworthiness in complex networked environments.

This research contributes to addressing the challenges associated with trust and reputation systems by introducing a mathematically grounded model that considers both individual behavior and network dynamics. The proposed formulation offers a solid foundation for further exploration and application in diverse domains, including e-commerce, social networks, and decentralized systems.

7

Conclusions and Future Works

This thesis is devoted to the analysis and development of trust and reputation systems, focusing on the development of accurate models for the computation of reputation, especially in virtual contexts such as social networks. In these environments, where users share feedback based on their experiences, we investigated the dynamics that influence reputation formation. By analysing what causes users to assign trust and how reputation is perceived, we have proposed advanced models that aim at accurately capturing the complexity of these interactions. Due to vastness of the topic, we have divided the thesis into two distinctive parts. In the first part, we focused on model development with an engineering approach, adopting a data processing engineering perspective. After a brief introduction to the state of the art and the main definitions, such as trust, reputation, multi-agents system and agents organization, we delved into the field by presenting two key algorithms. These algorithms were specifically designed to identify malicious and colluding users present in the virtual environment. Through these tools, we were able to more precisely identify users who act maliciously and who could compromise the integrity of virtual environments. A crucial result of this first part was the ability to calculate the actual reputation of honest users, taking into account the fraudulent actions and interactions present in such contexts. In this way, our research has contributed to providing a more accurate and reliable picture of user reputation in online environments, improving the understanding and management of the complex dynamics related to trust and virtual reputation.

In particular, in the first chapter of the thesis, we provided a general background on the topic discussed, presenting preliminary information and illustrating the motivations behind the decision to explore this specific research topic. Moreover, we emphasised the importance of addressing the field of trust and reputation systems in detail.

In the second chapter, we focused on the state of the art, exploring in detail how trust and reputation systems are used and analysing the main scenarios in which they are applied. In particular, we examined multi-agent systems, providing an extensive analysis of their relevance and the situations in which they are employed. In this context, we examined the organisational structure of agents, with a specific focus on malicious agents, who operate in secret to take advantage at the cost of honest agents. We analysed the nature of such agents and the challenges associated with recognising and managing fraudulent activities, thus contributing to a better insight into the contexts in which trust and reputation systems develop.

In Chapter 3, we have focused on the problem of identifying, in a social network, the trustworthiness of an agent (human or software entity), in order to detect malicious actors and marginalize or expel them from the community. In this context, several TRR systems have been proposed in the literature, and here we have examined the particular case represented by the well known Eigentrust algorithm. This algorithm is recognized as one of the most effective solution both to measure the reputation in a set of social agents and as benchmark or training assistant. However, Eigentrust, in order to detect malicious agents, uses some additional information about agents that can be a priori considered particularly trustworthy, rewarding them in terms of reputation, while the other agents are penalized. In this setting, we have highlighted a possible limitation in the Eingentrust algorithm, observing that it increases the reputation of all the pre-trusted agents, regardless of their reliability. This way, the method is capable to effectively recognize colluding agents, but producing the side effect to flattening the differences, in terms of reliability, between honest agents. To address the problem above, we have proposed a different strategy, based on a decrement of the fraudulent trust values that colluding agents mutually exchange. Our strategy introduces the advantage, with respect to Eigentrust, of estimating the reputation values of the honest actors in a manner more adherent to the actual reliability of these agents. This elevated precision of our method is particularly important, when the reputation of the agents is computed in a distributed environment, when different reputation values are continuously ombined from different sources. In order to measure this improvement of effectiveness we have introduced a metric of error to quantitatively determine how much an algorithm for the identification of malicious agents modifies the correct reputation values of the honest agents. We have used such a metric in an experimental simulation, in which we have compared the effectiveness of our result with repsect to the one generated by Eigentrust. We have shown that our method is more effective than Eigentrust in determining reputation values, presenting an error, which is about a thousand times lower than that produced by Eingentrust on medium-size social networks.

In Chapter 4 we have focused on the problem of detecting clusters of colluded agents in a social networks and marginalize or expel them from the community. We have already faced, in previous Chapter, two important problems affecting Eigentrust, i.e. the use of some additional information about agents that can be a-priori considered particularly trustworthy, and the strategy that EigenTrust uses of rewarding these trustworthy agents in terms of trust, while the other agents are penalized, producing the side effect to flattening the differences, in terms of reliability, between honest agents. However, we have highlighted as ER-EigenTrust (our first algorithm), similarly to the original version of EigenTrust, presents an important limitation in terms of false positives that are generated when colluded agents are partitioned in different groups. So, we have proposed a new algorithm for detecting colluded agents, which combines EigenTrust with a clustering procedure, grouping agents based on their reputation scores. Our experimental campaign showed that our Cluster method, besides of maintaining the same effectiveness of Eigentrust in detecting malicious agents (also considering the improvement introduced by ER-EigentTrust), is significantly more capable of avoiding the presence of false positives. The experiments conducted on communities of users with medium-high dimensions produced highly satisfactory results that were very similar to the cases with small communities. The Cluster method also proved adaptable to large communities, showing high precision and recall both as the size of the community increased and as the percentage of malicious users increased. At the same time, the ER EigenTrust Method continues to prove effective especially in the presence of a high percentage of malicious users, showing a significant improvement in precision as the size of communities increases. The execution times of the algorithms used to detect malicious users vary considerably. The Cluster method takes around 1700 seconds to complete the process while ER EigenTrust method takes just over 100 seconds for a community of 10000 users. The significantly different execution times between the two detection methods should not be misinterpreted as a direct indicator of the overall efficiency of the two approaches. Despite the longer time involved, the cluster method demonstrates significantly higher precision in detecting malicious users and in not generating false positives, underlining the trade-off between execution time and the accuracy of the results obtained.

In the second part of the thesis, we dedicated to study trust and reputation models by adopting both a different methodological approach and examining a different context. We have embraced a mathematical approach, employing the theory of variational inequalities as an analysis tool. This methodology was applied in a virtual environment, in which users evaluated specific objects. Our analysis focused on how these mathematical models could be effectively applied to understand and improve the concept of reputation in digital environments, thus adding a level of detail and precision to the evaluation of the objects in question.

In Chapter 5 we provided an overview of the theory of variational inequalities, examining its application to optimization problems. Furthermore, we have provided some insight into Lagrangian theory, highlighting the key role played by Lagrange multipliers in the context of solution analysis. These multipliers, with their interpretative value, have contributed to illuminating the underlying dynamics, providing a more complete and in-depth picture of the optimal search for solutions. We have tried to show the relevance of these theoretical concepts in the context of our analysis based on variational inequalities and optimization.

Finally, in Chapter 6 we introduce a variational formulation approach to model and analyze trust and reputation systems. By formulating trust and reputation as variational problems, this approach gives us a new perspective on understanding mechanisms that govern the creation of trust. In order to define the equilibrium conditions that allow to determine the equilibrium weighted trust values, we introduced the Lagrange multipliers associated with the capacity constraints. We demonstrate that the equilibrium conditions can be formulated as a variational inequality problem and we provide a novel alternative formulation. In the chapter we illustrate the applicability of this variational formulation through various simulations, demonstrating its effectiveness in modeling trust and reputation systems. Over the course of the simulation, we explored the robustness of our model across different conditions, introducing significant variations in both user trustworthiness and initial item reputation ratings. This sensitivity analysis allowed us to evaluate how the system responded to changes in key variables, providing valuable insights into its adaptability and reliability in diverse contexts. Furthermore, the limited but representative size of the simulated community and reasonable execution times contributed to providing concrete and applicable results, underlining the practical applicability of our approach in realistic scenarios.

In future research, we intend to devote additional efforts to the development of trust and reputation systems, focusing in particular on the creation of new algorithms suitable for less restrictive and more realistic contexts. This involves addressing limitations related to various types of malicious agents (only colluding) and variations in the networks used, that we exposed in section 3.6. Specifically, we aim to overcome current restrictions associated with the static and complete nature of networks, instead opting for dynamics networks and distributed approaches. Regarding the algorithm introduced in Chapter 4, we aim to construct an automated procedure to determine the actual number of clusters within the social community, without relying on approaches already documented in the literature. Furthermore, we aim to establish better parameters for classifying groups of malicious agents again without limiting ourselves only to the category of colluding malicious agents.

In addition, would like to continue the study of the variational formulation of trust and reputation systems, to provide an even more adherent model to real systems. The first step will be the formulation of a model with an iterative computing of the reputation in terms of a quasi-variational inequality. Moreover, we would like to provide a variational formulation for trust and reputation system with n agents, evaluating each other too. Since Game theory represents an excellent methodological framework for the investigation of decision-makers who compete amongst themselves (see [97–100]), and Nash equilibrium problems are naturally associated with variational inequalities ([101]), we would like to develop a a non cooperative model for trust and reputation systems, in which the underlying equilibrium concept is a generalized Nash equilibrium ([102, 103]).

Acknowledgment

It seems like yesterday when I started this journey, yet three years have passed since then. During this time, I have had the privilege of embarking on a research journey that has enriched me personally and professionally in ways I never imagined.
I have had the opportunity to meet numerous scholars and participate in stimulating conferences that have enriched my knowledge. In addition, I have been fortunate to visit beautiful places during my research journey, which have also allowed me to appreciate the beauty and diversity of the world around us.

I would like to express my deepest gratitude to all the people who contributed to the success of this research work and the completion of my doctoral thesis.
I would like to express my deepest gratitude to my supervisor, Prof. Sofia Giuffrè, who played a fundamental role in initiating me into research and generously donated her time even before I started my PhD. Her valuable knowledge and commitment played a fundamental role in my academic training and in the development of this research project.
I would also like to dedicate my heartfelt thanks to Prof. Mariantonia Cotronei, who has been a constant presence from the beginning. Her support has been crucial on many occasions. I sincerely hope to be able to continue collaborating with her in future work as well.
I would like to express my sincere gratitude to Professors Domenico Rosaci and Giuseppe M.L. Sarnè for the valuable contribution they have made to my academic career. Although they come from different fields of study, they have generously shared their knowledge and expertise, enriching me with new and stimulating perspectives. These were fundamental to the realisation of this research work.
I would like to express my sincere thanks to Prof. Patrizia Daniele and Dr Gabriella Colajanni. With their cooperation, expertise and availability they have contributed significantly to the realisation of this research.
I would like to express my sincere gratitude to the evaluators of this thesis. Thanks

to their valuable suggestions and constructive criticism, I was able to make significant improvements to the results.
I would like to extend sincere thanks to all my friends and those who have contributed in various ways to this work.
In particular, I would like to thank Mirko for his constant support, his willingness to listen to me and encourage me to always give my best. His participation in all my successes has been fundamental and appreciated more than I can express in words.
Special thanks also go to Guglielmo for his many suggestions and invaluable help, especially with regard to programming with Matlab. His advice has contributed significantly to the improvement of this research work. I wish them both a bright and successful future in their careers.
Finally, I would like to say a special thanks to my family for always being by my side, even when some of my choices were not shared. In particular, I want to dedicate an extraordinary thank you to my sister Caterina. Words are not enough to express the depth of my gratitude to her. Her trust, active support and constant encouragement during this research journey have been crucial to my success. Caterina has shown infinite love and support, and for this I will be eternally grateful.

Attilio

References

[1] Yaowen Zhang et al. “Multi-agent feature learning and integration for mixed cooperative and competitive environment”. In: *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE. 2020, pp. 9–14.

[2] Yara Rizk, Mariette Awad, and Edward W Tunstel. “Decision making in multiagent systems: A survey”. In: *IEEE Transactions on Cognitive and Developmental Systems* 10.3 (2018), pp. 514–529.

[3] Weidong Fang et al. “Trust-based attack and defense in wireless sensor networks: a survey”. In: *Wireless Communications and Mobile Computing* 2020 (2020), pp. 1–20.

[4] Félix Gómez Mármol and Gregorio Martínez Pérez. “Security threats scenarios in trust and reputation models for distributed systems”. In: *computers & security* 28.7 (2009), pp. 545–556.

[5] Behrouz Pourghebleh, Karzan Wakil, and Nima Jafari Navimipour. “A comprehensive study on the trust management techniques in the Internet of Things”. In: *IEEE Internet of Things Journal* 6.6 (2019), pp. 9326–9337.

[6] Amir Jalaly Bidgoly and Behrouz Tork Ladani. “Benchmarking reputation systems: A quantitative verification approach”. In: *Computers in Human Behavior* 57 (2016), pp. 274–291.

[7] Kevin Hoffman, David Zage, and Cristina Nita-Rotaru. “A survey of attack and defense techniques for reputation systems”. In: *ACM Computing Surveys (CSUR)* 42.1 (2009), pp. 1–31.

[8] Tyrone Grandison and Morris Sloman. “Trust management tools for internet applications”. In: *Trust Management: First International Conference, iTrust 2003 Heraklion, Crete, Greece, May 28–30, 2003 Proceedings 1*. Springer. 2003, pp. 91–107.

[9] Sergey Brin. “The PageRank citation ranking: bringing order to the web”. In: *Proceedings of ASIS, 1998* 98 (1998), pp. 161–172.

[10] Audun Jøsang, Roslan Ismail, and Colin Boyd. “A survey of trust and reputation systems for online service provision”. In: *Decision support systems* 43.2 (2007), pp. 618–644.

[11] Ali Dorri, Salil S Kanhere, and Raja Jurdak. “Multi-agent systems: A survey”. In: *Ieee Access* 6 (2018), pp. 28573–28593.

[12] Isaac Pinyol and Jordi Sabater-Mir. “Computational trust and reputation models for open multi-agent systems: a review”. In: *Artificial Intelligence Review* 40.1 (2013), pp. 1–25.

[13] Sergio Marti and Hector Garcia-Molina. "Identity crisis: anonymity vs reputation in P2P systems". In: *Proceedings Third International Conference on Peer-to-Peer Computing (P2P2003)*. IEEE. 2003, pp. 134–141.

[14] Gayatri Swamynathan, Kevin C Almeroth, and Ben Y Zhao. "The design of a reliable reputation system". In: *Electronic Commerce Research* 10 (2010), pp. 239–270.

[15] Chrysanthos Dellarocas. "Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior". In: *Proceedings of the 2nd ACM Conference on Electronic Commerce*. 2000, pp. 150–157.

[16] John R Douceur. "The sybil attack. Peer-to-peer Systems". In: *Lecture Notes in Computer Science* 2429 (2002), pp. 251–260.

[17] Haifeng Yu et al. "Sybilguard: defending against sybil attacks via social networks". In: *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. 2006, pp. 267–278.

[18] Haifeng Yu et al. "Sybilguard: defending against sybil attacks via social networks". In: *IEEE/ACM Transactions on networking* 16.3 (2008), pp. 576–589.

[19] Rajat Bhattacharjee and Ashish Goel. "Avoiding ballot stuffing in ebay-like reputation systems". In: *Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*. 2005, pp. 133–137.

[20] Mariantonia Cotronei et al. "Identifying Colluding Actors in Social Communities by Reputation Measures". In: *International Conference on Applied Intelligence and Informatics*. Springer. 2022, pp. 347–359.

[21] Mariantonia Cotronei et al. "Improving the Effectiveness of Eigentrust in Computing the Reputation of Social Agents in Presence of Collusion." In: *International Journal of Neural Systems* (2023), pp. 2350063–2350063.

[22] Anastasios Alexiadis et al. "A smarthome conversational agent performing implicit demand-response application planning". In: *Integrated Computer-Aided Engineering* 29.1 (2022), pp. 43–61.

[23] Mohammad Aghababaei and Maria Koliou. "Community resilience assessment via agent-based modeling approach". In: *Computer-Aided Civil and Infrastructure Engineering* 38.7 (2023), pp. 920–939.

[24] Amir Esmalian, Wanqiu Wang, and Ali Mostafavi. "Multi-agent modeling of hazard–household–infrastructure nexus for equitable resilience assessment". In: *Computer-Aided Civil and Infrastructure Engineering* 37.12 (2022), pp. 1491–1520.

[25] Giancarlo Fortino et al. "A trust-based team formation framework for mobile intelligence in smart factories". In: *IEEE Transactions on Industrial Informatics* 16.9 (2020), pp. 6133–6142.

[26] Sophie Hendrikse, Jan Treur, and Sander Koole. "Modeling emerging interpersonal synchrony and its related adaptive short-term affiliation and long-term bonding: a second-order multi-adaptive neural agent model". In: *International journal of neural systems* (2023).

[27] Maria Nadia Postorino, Francesco Alessandro Sarné, Giuseppe ML Sarné, et al. "An Agent-based Simulator for Urban Air Mobility Scenarios." In: *WOA*. Vol. 1613. 2020, p. 2.

[28] Mariantonieta Gutierrez Soto and Hojjat Adeli. "Multi-agent replicator controller for sustainable vibration control of smart structures". In: *Journal of Vibroengineering* 19.6 (2017), pp. 4300–4322.

[29] Audun Jøsang and Jennifer Golbeck. "Challenges for robust trust and reputation systems". In: *Proceedings of the 5th International Workshop on Security and Trust Management (SMT 2009), Saint Malo, France*. Vol. 5. 9. Citeseer. 2009.

[30] Sokratis Vavilis, Milan Petković, and Nicola Zannone. "A reference model for reputation systems". In: *Decision Support Systems* 61 (2014), pp. 147–154.

[31] Karen K Fullam et al. "A specification of the agent reputation and trust (art) testbed: experimentation and competition for trust in agent societies". In: *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. 2005, pp. 512–518.

[32] Anna A Adamopoulou and Andreas L Symeonidis. "A simulation testbed for analyzing trust and reputation mechanisms in unreliable online markets". In: *Electronic Commerce Research and Applications* 13.5 (2014), pp. 368–386.

[33] Reid Kerr and Robin Cohen. "Treet: the trust and reputation experimentation and evaluation testbed". In: *Electronic Commerce Research* 10 (2010), pp. 271–290.

[34] Félix Gómez Mármol and Gregorio Martínez Pérez. "TRMSim-WSN, trust and reputation models simulator for wireless sensor networks". In: *2009 IEEE International Conference on Communications*. IEEE. 2009, pp. 1–5.

[35] Amir Jalaly Bidgoly and Behrouz Tork Ladani. "Modelling and quantitative verification of reputation systems against malicious attackers". In: *The Computer Journal* 58.10 (2015), pp. 2567–2582.

[36] Seyed Asgary Ghasempouri and Behrouz Tork Ladani. "Modeling trust and reputation systems in hostile environments". In: *Future Generation Computer Systems* 99 (2019), pp. 571–592.

[37] Luis Cabral and Ali Hortacsu. "The dynamics of seller reputation: Evidence from eBay". In: *The Journal of Industrial Economics* 58.1 (2010), pp. 54–78.

[38] Stephen C Hayne, Haonan Wang, and Lu Wang. “Modeling reputation as a time-series: Evaluating the risk of purchase decisions on eBay”. In: *Decision Sciences* 46.6 (2015), pp. 1077–1107.

[39] Paul Resnick and Richard Zeckhauser. “Trust among strangers in Internet transactions: Empirical analysis of eBay’s reputation system”. In: *The Economics of the Internet and E-commerce*. Emerald Group Publishing Limited, 2002, pp. 127–157.

[40] Li Xiong and Ling Liu. “Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities”. In: *IEEE transactions on Knowledge and Data Engineering* 16.7 (2004), pp. 843–857.

[41] Fabrizio Messina et al. “A trust-aware, self-organizing system for large-scale federations of utility computing infrastructures”. In: *Future Generation Computer Systems* 56 (2016), pp. 77–94.

[42] Sergey Brin and Lawrence Page. “The anatomy of a large-scale hypertextual web search engine”. In: *Computer networks and ISDN systems* 30.1-7 (1998), pp. 107–117.

[43] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. “The eigentrust algorithm for reputation management in p2p networks”. In: *Proceedings of the 12th international conference on World Wide Web*. 2003, pp. 640–651.

[44] Rob Jansen et al. “A priori trust vulnerabilities in eigentrust”. In: *University of Minnesota, http://www-users. cs. umn. edu/jansen/papers/fet-csci5271. pdf, Tech. Rep* (2008).

[45] Zoë Abrams, Robert Mcgrew, and Serge Plotkin. “A non-manipulable trust system based on eigentrust”. In: *ACM SIGecom Exchanges* 5.4 (2005), pp. 21–30.

[46] Xinxin Fan et al. “EigenTrustp++: Attack resilient trust management”. In: *8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*. IEEE. 2012, pp. 416–425.

[47] Heba A Kurdi. “HonestPeer: An enhanced EigenTrust algorithm for reputation management in P2P systems”. In: *Journal of King Saud University-Computer and Information Sciences* 27.3 (2015), pp. 315–322.

[48] Sheng Gao et al. “T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm”. In: *China Communications* 16.12 (2019), pp. 111–123.

[49] Kun Lu, Junlong Wang, and Mingchu Li. “An Eigentrust dynamic evolutionary model in P2P file-sharing systems”. In: *Peer-to-Peer Networking and Applications* 9 (2016), pp. 599–612.

[50] Alanoud Alhussain, Heba Kurdi, and Lina Altoaimy. "Managing trust and detecting malicious groups in peer-to-peer iot networks". In: *Sensors* 21.13 (2021), p. 4484.

[51] Guangchi Liu et al. "Trust assessment in online social networks". In: *IEEE Transactions on Dependable and Secure Computing* 18.2 (2019), pp. 994–1007.

[52] Ferry Hendrikx, Kris Bubendorfer, and Ryan Chard. "Reputation systems: A survey and taxonomy". In: *Journal of Parallel and Distributed Computing* 75 (2015), pp. 184–197.

[53] Fabrizio Messina et al. "A trust-aware, self-organizing system for large-scale federations of utility computing infrastructures". In: *Future Generation Computer Systems* 56 (2016), pp. 77–94.

[54] Rosaci Domenico Marcianò Attilio and Sarnè Giuseppe. "A Strategy to Detect Colluding Groups by Reputation Measures". In: *CEUR Workshop Proceedings WOA 2023*. Vol. 3579. 2023.

[55] Ulrike Von Luxburg. "A tutorial on spectral clustering". In: *Statistics and computing* 17 (2007), pp. 395–416.

[56] Satu Elisa Schaeffer. "Graph clustering". In: *Computer science review* 1.1 (2007), pp. 27–64.

[57] Santo Fortunato. "Community detection in graphs". In: *Physics Reports* 486.3 (2010), pp. 75–174. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2009.11.002. URL: https://www.sciencedirect.com/science/article/pii/S0370157309002841.

[58] Fragkiskos D. Malliaros and Michalis Vazirgiannis. "Clustering and community detection in directed networks: A survey". In: *Physics Reports* 533.4 (2013). Clustering and Community Detection in Directed Networks: A Survey, pp. 95–142. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2013.08.002. URL: https://www.sciencedirect.com/science/article/pii/S0370157313002822.

[59] Andrew Ng, Michael Jordan, and Yair Weiss. "On Spectral Clustering: Analysis and an algorithm". In: *Advances in Neural Information Processing Systems*. Ed. by T. Dietterich, S. Becker, and Z. Ghahramani. Vol. 14. MIT Press, 2001. URL: https://proceedings.neurips.cc/paper_files/paper/2001/file/801272ee79cfde7fa5960571fee36b9b-Paper.pdf.

[60] William Karush. "Minima of functions of several variables with inequalities as side constraints". In: *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago* (1939).

[61] HW Kuhn and AW Tucker. "Proceedings of 2nd berkeley symposium". In: *Proceedings of 2nd Berkeley Symposium*. 1951, pp. 481–492.

[62] Antonio Signorini. "Questioni di elasticità non linearizzata e semilinearizzata". In: *Rend. Mat. Appl* 18.5 (1959), pp. 95–139.

[63] Guido Stampacchia. "Formes bilineaires coercitives sur les ensembles convexes". In: *Comptes Rendus Hebdomadaires Des Seances De L Academie Des Sciences* 258.18 (1964), p. 4413.

[64] Philip Hartman and Guido Stampacchia. "On some non-linear elliptic differential-functional equations". In: (1966).

[65] JL Lions and G Stampacchia. "Le disuguaglianze variazionali". In: *Comunicazioni su Matematica Pura e Applicata* 20 (1967), pp. 493–519.

[66] David Kinderlehrer and Guido Stampacchia. *An introduction to variational inequalities and their applications*. SIAM, 2000.

[67] Patrizia Daniele, Antonino Maugeri, and Werner Oettli. "Variational inequalities and time-dependent traffic equilibria". In: *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics* 326.9 (1998), pp. 1059–1062.

[68] P Daniele, A Maugeri, and W Oettli. "Time-dependent traffic equilibria". In: *Journal of Optimization Theory and Applications* 103 (1999), pp. 543–555.

[69] Terry L Friesz et al. "A variational inequality formulation of the dynamic network user equilibrium problem". In: *Operations research* 41.1 (1993), pp. 179–191.

[70] Anna Nagurney. *Network economics: A variational inequality approach*. Vol. 10. Springer Science & Business Media, 1998.

[71] Patrizia Daniele. *Dynamic networks and evolutionary variational inequalities*. Edward Elgar Publishing, 2006.

[72] Annamaria Barbagallo and Monica-Gabriela Cojocaru. "Dynamic equilibrium formulation of the oligopolistic market problem". In: *Mathematical and Computer Modelling* 49.5-6 (2009), pp. 966–976.

[73] P Daniele. "Evolutionary variational inequalities applied to financial equilibrium problems in an environment of risk and uncertainty". In: *Nonlinear Analysis: Theory, Methods & Applications* 63.5-7 (2005), e1645–e1653.

[74] Patrizia Daniele et al. "Duality theory and applications to unilateral problems". In: *Journal of Optimization Theory and Applications* 162 (2014), pp. 718–734.

[75] S Giuffrè. "Lagrange multipliers and non-constant gradient constrained problem". In: *Journal of Differential Equations* 269.1 (2020), pp. 542–562.

[76] Sofia Giuffrè and Attilio Marcianò. "Lagrange multipliers and nonlinear variational inequalities with gradient constraints". In: *Philosophical Transactions of the Royal Society A* 380.2236 (2022), p. 20210355.

[77] Maria Bernadette Donato, Monica Milasi, and Laura Scrimali. “Walrasian equilibrium problem with memory term”. In: *Journal of optimization theory and applications* 151 (2011), pp. 64–80.

[78] Anna Nagurney and Patrizia Daniele. “International human migration networks under regulations”. In: *European Journal of Operational Research* 291.3 (2021), pp. 894–905.

[79] Anna Nagurney, David Parkes, and Patrizia Daniele. “The Internet, evolutionary variational inequalities, and the time-dependent Braess paradox”. In: *Computational Management Science* 4 (2007), pp. 355–375.

[80] Antonino Maugeri and Laura Scrimali. “A new approach to solve convex infinite-dimensional bilevel problems: application to the pollution emission price problem”. In: *Journal of Optimization Theory and Applications* 169 (2016), pp. 370–387.

[81] Anna Nagurney et al. “Dynamic electric power supply chains and transportation networks: An evolutionary variational inequality formulation”. In: *Transportation Research Part E: Logistics and Transportation Review* 43.5 (2007), pp. 624–646.

[82] Laura Scrimali. “An inverse variational inequality approach to the evolutionary spatial price equilibrium problem”. In: *Optimization and Engineering* 13 (2012), pp. 375–387.

[83] Guangjie Han et al. “Management and applications of trust in Wireless Sensor Networks: A survey”. In: *Journal of Computer and System Sciences* 80.3 (2014), pp. 602–617.

[84] Ignacio Aransay et al. “A trust and reputation system for energy optimization in cloud data centers”. In: *2015 IEEE 8th International Conference on Cloud Computing*. IEEE. 2015, pp. 138–145.

[85] Na Fan and Chase Q Wu. “On trust models for communication security in vehicular ad-hoc networks”. In: *Ad Hoc Networks* 90 (2019), p. 101740.

[86] Pasquale De Meo et al. “A reputation framework to share resources into iot-based environments”. In: *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*. IEEE. 2017, pp. 513–518.

[87] Roberto Aringhieri et al. “Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems”. In: *Journal of the American Society for Information Science and Technology* 57.4 (2006), pp. 528–537.

[88] Cristobald de Kerchove and Paul Van Dooren. “Iterative filtering for a dynamical reputation system”. In: *arXiv preprint arXiv:0711.3964* (2007).

[89] Ujwala Ravale, Anita Patil, and Gautam M Borkar. “Trust Management: A Cooperative Approach Using Game Theory”. In: *The Psychology of Trust*. IntechOpen, 2022.

[90] Chengyi Xia, Zhengyang Hu, and Dawei Zhao. “Costly reputation building still promotes the collective trust within the networked population”. In: *New Journal of Physics* 24.8 (2022), p. 083041.

[91] Amira Bradai and Hossam Afifi. “Game theoretic framework for reputation-based distributed intrusion detection”. In: *2013 International Conference on Social Computing*. IEEE. 2013, pp. 558–563.

[92] John A Tomlin. “A new paradigm for ranking pages on the world wide web”. In: *Proceedings of the 12th international conference on World Wide Web*. 2003, pp. 350–355.

[93] Amy N Langville and Carl D Meyer. *Google’s PageRank and beyond: The science of search engine rankings*. Princeton university press, 2006.

[94] Hien Trang Nguyen, Weiliang Zhao, and Jian Yang. “A trust and reputation model based on bayesian network for web services”. In: *2010 IEEE International Conference on Web Services*. IEEE. 2010, pp. 251–258.

[95] Yao Wang and Julita Vassileva. “Trust and reputation model in peer-to-peer networks”. In: *Proceedings Third International Conference on Peer-to-Peer Computing (P2P2003)*. IEEE. 2003, pp. 150–157.

[96] Anna Nagurney. *Network economics: A variational inequality approach*. Vol. 10. Springer Science & Business Media, 2013.

[97] Georgia Fargetta and Laura RM Scrimali. “Generalized Nash equilibrium and dynamics of popularity of online contents”. In: *Optimization Letters* 15.5 (2021), pp. 1691–1709.

[98] Gabriella Colajanni et al. “Cybersecurity investments with nonlinear budget constraints and conservation laws: variational equilibrium, marginal expected utilities, and Lagrange multipliers”. In: *International Transactions in Operational Research* 25.5 (2018), pp. 1443–1464.

[99] Annamaria Barbagallo et al. “Variational approach for a general financial equilibrium problem: the deficit formula, the balance law and the liability formula. A path to the economy recovery”. In: *European Journal of Operational Research* 237.1 (2014), pp. 231–244.

[100] Anna Nagurney, Emilio Alvarez Flores, and Ceren Soylu. “A Generalized Nash Equilibrium network model for post-disaster humanitarian relief”. In: *Transportation research part E: logistics and transportation review* 95 (2016), pp. 1–18.

[101] Francisco Facchinei, Andreas Fischer, and Veronica Piccialli. “On generalized Nash games and variational inequalities”. In: *Operations Research Letters* 35.2 (2007), pp. 159–164.

[102] John F Nash Jr. “Equilibrium points in n-person games”. In: *Proceedings of the national academy of sciences* 36.1 (1950), pp. 48–49.

[103] Non-Cooperative Games. “JOHN F. NASH, JR.” In: *The Essential John Nash* (2002), p. 51.

Publications

1 S. Giuffrè, A. Marcianò: Duality Minimax and Applications, Minimax Theory and its Applications 06 (2021), No. 2, 353-364.

2 S. Giuffrè, A. Marcianò: On the existence of radial solutions to a Non constant gradient constraint problem, Symmetry, 2022, 14 (7), 1423.

3 S. Giuffrè, A. Marcianò: Lagrange multipliers and nonlinear variational inequalities with gradient constraints, Philosophical Transactions of the Royal Society A, 2022, 380 (2236), 202103355.

4 A. Marcianò: Accurate Colluding Agents Detection by Reputation Measures, CEUR Workshop Proceedings ISSN:1613-0073, 2022.

5 M. Cotronei, S. Giuffrè, A. Marcianò, D. Rosaci, GML Sarnè, Identifying Colluding Actors in Social Communities by Reputation Measures. In: Mahmud, M., Ieracitano, C., Kaiser, M.S., Mammone, N., Morabito, F.C. (eds) Applied Intelligence and Informatics. AII 2022. Communications in Computer and Information Science, vol 1724. Springer, Cham, 2022.

6 M. Cotronei, S. Giuffrè, A. Marcianò, D. Rosaci, GML Sarnè, (2022). Detecting Collusive Agents by Trust Measures in Social IoT Environments: A Novel Reputation Model. In Security, Trust and Privacy Models, and Architectures in IoT Environments (pp. 43-61). Cham: Springer International Publishing.

7 M. Cotronei, S. Giuffrè, A. Marcianò, D. Rosaci, GML Sarnè, Improving the Effectiveness of Eigentrust in Computing Reputation of Social Agents in Presence of Collusion, International journal of neural systems (2023).

8 A. Marcianò, D. Rosaci, G. M.L. Sarnè, A Strategy to Detect Colluding Groups by Reputation Measures, CEUR Workshop Proceedings, 2023, 3579, (pp. 92-105).

9 M. Cotronei, S. Giuffrè, A. Marcianò, D. Rosaci, G. M.L. Sarnè, Using Trust and Reputation for Detecting Groups of Colluded Agents in Social Networks, submitted.

10 G. Colajanni, P. Daniele, S. Giuffrè, A. Marcianò, A Variational Formulation for a Trust and Reputation System, submitted.