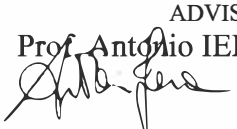
DOCTORAL SCHOOL
UNIVERSITA' *MEDITERRANEA* DI REGGIO CALABRIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE, DELLE INFRASTRUTTURE E DELL'ENERGIA SOSTENIBILE (DIIES) PHD IN INFORMATION ENGINEERING
S.S.D. ING-INF/03
XXXIII CICLO

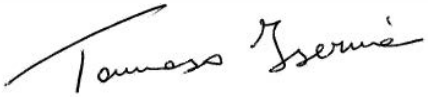
USING SOCIAL OBJECTS IN DISTRIBUTED ONLINE SOCIAL NETWORKS TO BRIDGE SEPARATED COMMUNITIES WITH COMMON INTERESTS

CANDIDATE
David GAROMPOLO

ADVISOR
Prof. Antonio IERA



COORDINATOR
Prof. Tommaso ISERNIA



REGGIO CALABRIA, APRIL 2021

Finito di stampare nel mese di **Aprile 2021**

Edizione 

Quaderno N. 48
Collana *Quaderni del Dottorato di Ricerca in Ingegneria dell'Informazione*
Curatore *Prof. Tommaso Isernia*

ISBN 978-88-99352-48-6

Università degli Studi *Mediterranea* di Reggio Calabria
Salita Melissari, Feo di Vito, Reggio Calabria

DAVID GAROMPOLO

USING SOCIAL OBJECTS IN DISTRIBUTED ONLINE SOCIAL NETWORKS TO BRIDGE SEPARATED COMMUNITIES WITH COMMON INTERESTS

The Teaching Staff of the PhD course in
INFORMATION ENGINEERING
consists of:

Tommaso ISERNIA (coordinator)
Pier Luigi ANTONUCCI
Giuseppe ARANITI
Francesco BUCCAFURRI
Salvatore COCO
Giuseppe COPPOLA
Lorenzo CROCCO
Dominique DALLET
Claudio DE CAPUA
Francesco DELLA CORTE
Giuliana FAGGIO
Fabio FILIANOTI
Patrizia FRONTERA
Sofia GIUFFRE'
Giorgio GRADITI
Voicu GROZA
Antonio IERA
Gianluca LAX
Aime' LAY EKUAKILLE
Giacomo MESSINA
Antonella MOLINARO
Andrea MORABITO
Giacomo MORABITO
Rosario MORELLO
Domenico ROSACI
Giuseppe RUGGERI
Domenico URSINO

To my family

Alla mia famiglia

Related Papers

- Chiara Suraci, Sara Pizzi, **David Garompolo**, Giuseppe Araniti, Antonella Molinaro, Antonio Iera, "Trusted and secured D2D-aided communications in 5G networks", Ad Hoc Networks, 102403.
- **David Garompolo**, Antonella Molinaro, Antonio Iera, "Bridging Separate Communities with Common Interest in Distributed Social Networks through the Use of Social Objects", Future Generation Computer Systems, 2021, to appear.

Abstract

In light of the growing privacy violations of users of centralized social networks, the need to define effective platforms for decentralized online social networks (DOSNs) is deeply felt. Interesting solutions have been proposed in the past, which own the necessary mechanisms to allow users maintaining control over their personal information and set the rules to regulate the access of other users. Unfortunately, the effectiveness of this type of solutions is severely reduced by the fact that different user communities with a shared interest could be disconnected/separated from each other. This translates into a reduced ability in effectively spreading data of common interest towards the involved users, as it currently happens in centralized social networks. In order to overcome the cited limitation, this thesis proposes a disruptive approach, which exploits the availability of a new class of Internet of Things (IoT) devices with autonomous social behaviors and cognitive abilities. Such devices can be leveraged as intermediaries of friendship between devices' owners with the same interest who are connected to a DOSN platform. We will demonstrate that clear advantages can be achieved in terms of increased percentage of reachable interested nodes (a specific measure of Delivery Ratio) in distributed social networks among humans, when enhanced with so called *Mediator Objects* following the rules of the well-known social IoT (SIoT) concept.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Scandals such as Cambridge Analytica have clearly shown that technology giants (Facebook, Google & Co, etc.) have serious problems in protecting user data. To guarantee users greater privacy and greater control over their data, researchers all over the world have recently proposed an increasing number of decentralized solutions to implement Decentralized Online Social Networks (DOSNs) [35] either based on Peer-to-Peer (P2P) architectures [3, 14, 25, 34] or with a Web-based nature [94, 96].

In this context, an interesting decentralized platform for Social Web applications, Solid [78], was proposed by Tim Berners Lee. It focuses on decoupling user data from applications, making sure they have a simple, generic and well-defined way to access data stored in the users' Web-accessible personal online datastore (POD) [78]. This platform, in essence, allows users to maintain control over their personal information and to decide where it is stored, who can view it, and which applications can access it.

Web-oriented Decentralized Online Social Networks (DOSNs) can be developed by leveraging the Solid platform (for example Solid Social) that a user (Source) can use to diffuse her own content in a targeted manner to all people interested in receiving it. In Solid, the social graph visible to each user consists of the contacts stored in her PODs, the contacts of these contacts, and so on. Contacts can be seen as an interface to manage the user's distributed social graph. Interestingly, a user can either mark her contacts as public or make them accessible to a specific individual or group of people [56, 78]. While with a Centralized Online Social Network the application knew the complete social graph, in this case it will only know a partial graph consisting of the contacts contained in the PODs of the Source

willing to disclose the content and the contacts contained in the PODs of other users (her contacts) to which the Source is authorized to access to.

Obviously, the authorization mechanisms in the use of the node's contacts can be leveraged to guarantee the diffusion of a given content to the interested nodes only, This allows to reduce as much as possible the number of unnecessary nodes visited and to make the information spreading as efficient as possible [30, 48, 49, 59].

Unfortunately, such an approach, while being very effective with a view to ensuring extreme trust and security in the access to information, at the same time amplifies the inherent limitations of decentralized social networks solutions: different communities with the same interest could be disconnected/separated from each other with a higher probability and some nodes potentially interested in a content could be isolated.

In other words, nodes that share the same generic interest cannot always discover and mutually exchange contents related to that interest. This reduces the extent to which a Source manages to spread its content of interest to the highest possible number of interested nodes.

This might bring to a possible consequent decrease in interest in this type of solution by users who increasingly play the role of "prosumers" (i.e., both producers and consumers) of contents. As consumers, they could see the possibility of accessing content of their interest reduced due to a discovery hindered by fragmentation and possible isolation of the communities of interest. Even worse is the situation with a view to a user producing content. In fact, the latter would have difficulty in spreading its content to a large number of interested users and, if a business is based on the delivery of such content, the problem translates into a reduction in possible revenues from commercial activities that operate on distributed platforms.

We currently have a new class of devices available that could help in addressing some of these highlighted issues. In fact, objects that show autonomous social behaviors and that have cognitive abilities have been studied and their use begins to spread. The idea is to allow devices with these characteristics to act as "facilitators" of contact and friendship between users with the same interests and, therefore, as potential "bridges" between communities with similar interests but which are not connected in a distributed social network.

The function of a device acting as intermediary of friendships is a concept that could be seen as the implementation, through ICT technologies, of the well-known concept of

"social object" introduced, by referring to an object creating a human connection between two people, in sociological studies on object-centered sociality [16].

The objective of our work is to implement this concept through a solution that involves the notion of social devices and social relationships among devices. In particular, we want to study whether the introduction of these concepts can bring advantages in terms of increased percentage of reachable interested nodes (IRN Percentage, a specific measure of Delivery Ratio).

More specifically, the main contributions of the thesis are listed in the following.

We make the well-known Social Internet of Things (SIoT) concept [7, 8] evolve into its empowered version, named here *Enhanced SIoT*, enabling a further increase in the number of interested node that can be reached by a given content. In it, a key role is played by the introduced so called *Mediator Object*, a device with a social and cognitive nature that allows to mediate the propagation of a content towards other interested devices/users, otherwise unreachable.

We conduct a set of analysis, by referring to Enhanced SIoT, thanks to which we observe the advantages offered in terms of Interested Reachable Nodes (IRN) percentage in a decentralized social network compared to the traditional case in which only friendship (among humans) is leveraged. In particular, we observe the variation in the percentage of interested reachable nodes, when varying the percentage of nodes that during the Discovery phase contribute to spread the Source interests and that authorize access to their own PODs and when exploiting different combinations of social object relationships. We also evaluate the measure in which any social object relationship contributes to increase the number of reachable interested nodes.

Obviously, the information diffusion among social devices can take place by exploiting the mechanisms of management and control of the devices' trustworthiness proposed in the literature for a Social Internet of Things [17, 19, 54, 71]. In addition we propose new improvements in Trustworthiness models for Social Internet of Things Scenarios.

Finally, because today there are no SIoT tracks, nor datasets that contain all the information we needed, we tried to model the Scenario from real human tracks in Social Networks. After studying the current state of the art, we propose a new procedure to generate realistic SIoT tracks. This to the best of our knowledge is the first job to get all this information

(co-locations, interest profiles, Place of Interests, Home-Points, etc.) from realistic datasets (Brightkite [21], Foursquare [91–93]). We will describe the entire step-by-step mechanism.

Specifically, the thesis is organized as follows. In Chapter 2 we provide the background information needed to understand the context and scenario we are dealing with. In particular, we briefly illustrate the concepts behind Solid and behind the Social Internet of Things (SIoT). We also give an overview of the main works in the literature addressing research issues similar to ours. In Chapter 3 we: define the Enhanced Social Internet of Things, illustrate the concept of community in our scenario and the role of the Mediator Object, introduce a new Social Object Relationships, and describe the related establishment mechanisms. Eventually, we show how discovery and diffusion take place in the Enhanced SIoT version. In Chapter 4, we provide basic information about the concept of trustworthiness (or trust in brief) in general and how trust can be established. We present the status of art of the main models of trustworthiness. We propose new improvements, proposing a new trustworthiness model for Social Internet of Things comparing its performances with the closest trust models in literature. In Chapter 5.3, we suggest a new procedure to generate realistic SIoT tracks and present all the steps that have been taken during implementation. In Chapter 6 we analyze the performance achieved by the proposed system compared to a traditional social network.

Chapter 2

Background

2.1 Online Social Network (OSN)

2.1.1 What they are and why we treat OSNs

In recent years, Online Social Networks have changed the way people can communicate, interact and exchange information with each other. They have attracted a huge number of users and they are one of the most popular applications in the Internet. Just think that at the beginning of this year the number of people around the world using the Internet was about the 59% of the total world population and the number of active social media users was about the 49%. Today Facebook can be considered the most popular OSN with around 2.5 billion active users, followed by YouTube and WhatsApp with around 2 billion and Instagram with 1 billion. Facebook also has the highest number of daily user connections. Through these platforms, users can build a public profile, establish relationships, share content and look up new friends. Among the most important social networks, in addition to the aforementioned Facebook, Youtube, WhatsApp and Instagram, we can mention Twitter, Linkedin, TikTok, Pinterest, Snapchat and many others. Among the different services offered by these platforms, they also allow the possibility of sharing information within user groups and of building user communities characterized by common interests [35].

2.1.2 Limits of the current Centralized OSNs

The privacy of user's data is undoubtedly one of the main problems of current centralized OSNs. In May 2015 Belgian privacy commission study concluded that Facebook's use of user data violated privacy and data protection laws. More recently scandals such as Cambridge Analytica have clearly shown that technology giants (Facebook, Google & Co, etc.) have serious problems in protecting user data. These and many other legal issues involving not only Facebook, but also other OSNs (Google, Twitter) have highlighted that users' social data (including personal sensitive data) may have been sold without any consent from their legitimate owners. In short, the companies that manage the OSNs use this data for commercial purposes, not protecting user data. The main problem is that all the user's data are stored in centralized provider's repository and so users risk to loss of control over ownership of their data. Another important problem of Centralized OSNs is the performance bottleneck due to the very high number of user requests and the huge amount of social data (all data exchanged in Social Networks regarding both user information and generated content). Other drawbacks related to the centralized nature of the OSN are the single point of failure, the need to be online for every transaction and the lack of locality. Finally, centralized OSNs can suffer from other problems such as limited scalability and high maintenance costs to manage the data of so many users. These problems led researchers to consider and develop decentralized alternatives [35, 95].

2.2 Decentralized Online Social Network (DOSN)

2.2.1 DOSN: definition and advantages

In order to solve the main problems of Centralized OSNs, researchers have increasingly moved towards decentralized solutions. A Decentralized Online Social Network (DOSN) is an online social network implemented on a distributed platform. Distributed in the sense that all computing, storage and communication resources are provided by the users. While in the Centralized case the single service provider had control over all user data and could change the existing terms of service, in the Decentralized case there is a set of nodes that cooperate to guarantee all the functionalities. This gives to the users more control over their privacy

[24]. Indeed in a DOSN the user data will be stored locally in devices and controlled by end users instead of OSN central entity (OSN provider). Consequently the DOSNs can mitigate the privacy control issues, the problem of security and scalability and increase the flexibility and the ability to deal with big data problems [27, 95]. They can also alleviate the problem of performance bottleneck and avoid the single point of failure and the single point of attack. Finally the shifting of the implementation of the infrastructure, and the privacy and security control to the user allow effectively lowering the operational cost [13, 33].

2.2.2 DOSN: problems

We have seen that the decentralization of OSNs allows to increase the level of privacy of user data. Unfortunately DOSNs also present a series of problems still to be solved concerning both the design of the DOSN architecture and the management of social data [35, 36].

A first fundamental problem concerns how to support efficient service discovery (ability to locate different and multi-source social resources) and diffusion of social data [95].

A second issue is represented by the presence of dynamism. In particular, there are two types of dynamism in DOSNs: social dynamism and infrastructure dynamism or churn. The first concerns changes in social relationships according to the variation (establishment or breakdown) of relations between users. The second is related to the fact that users arbitrarily decide when to be online or offline in the system. As a result the nodes (users) in the underlying network can appear or disappear, by modifying the overlay structure in terms of active links and nodes. Basically in DOSNs while social dynamism influences the structure of the Social Overlay, infrastructure dynamism influences the underlying overlay network. Of the two types of dynamism in DOSN, the second is more dangerous, being much more frequent than the first.

Another fundamental problem linked to the high level of dynamism is that of guaranteeing the availability of social data (data availability or persistence). In the absence of a central entity, new storage/archiving techniques must be used to ensure that the data are always available. One of the most used is replication. A current trend is to use trustworthiness to choose replica nodes, this allows to guarantee a high level of privacy in the system. In this context, one possibility is to choose the replica nodes taking into consideration only the friend nodes.

Another important issue concerns the development of techniques for the efficient propagation of social updates (user-generated content). In fact, in a DOSN it is essential to make the service as real-time as possible to provide the latest information to users. In centralized OSNs it is normal to have an effective Information Diffusion mechanism thanks to the uniqueness of its repository. On the contrary, in DOSN users have a limited knowledge of the network and communications between users' devices take place on the overlay. In particular in DOSNs based on Social Overlay users can directly communicate with each other only if there is a social link that connects them and since data can travel only through social links, there is the need for specific diffusion strategies to spread the information.

As for Scalability, mapping all the links of the social network into the links of the distributed network can be really expensive as well as inefficient (since many of the social links are inactive/unused). In fact, a user often interacts only with a certain number of friends, not with everyone.

Considering the Topology in which the nodes are connected to their friends in the overlay network if on the one hand it could favor information diffusion and data storage, on the other it would limit the availability and solidity of data access if a user has only a few online friends.

Finally, new privacy issues must be resolved in DOSNs, such as detecting trusted nodes that can act as replica nodes [35]. In the centralized case, privacy was guaranteed by the service provider. Unfortunately it has been seen that it itself can maliciously violate users' privacy by exploiting their data, for example by selling them to third parties. In DOSNs this is no longer possible because each user decides who can access his data [36].

2.3 Solid

Proposed by Tim Berners-Lee, Solid is an existing web-based open source platform that allows users to maintain control over their personal information and to decide where it is stored, who can view it and which applications can access it. Thanks to it, companies like Facebook will be allowed to use only part of the data, but this permission can also be withdrawn at any time. Solid's core is the "Solid POD" which we can see as "a private website with data interoperable with all apps". It stores all the user's personal information

that will be linked from the outside in order to be used. In this only the user will check their own information [56, 78]. This new vision, which allows to decouple data from applications, brings with it new problems to be addressed in reference to Social Network applications for Solid (eg Solid Social). While in the centralized Social Networks the application (e.g. Facebook, Instagram, etc.), possessing all the data (in the data Silos), knows the complete graph of the contacts of all users, in this new distributed vision this is no longer possible. In particular, each user will know, in addition to their contacts contained in their Solid PODs, only the contacts contained in the Solid PODs of the users who have granted them permission to access their PODs. Consequently we will have a graph with many separate components (Local Knowledge). Let's imagine that a user wants to disclose content in a targeted manner to all the nodes that are interested in receiving it. In this case the Social Network application is distributed, no longer knowing the complete graph, it will not be able during Discovery to discover all the nodes interested in receiving that content, as was the case in the centralized case. Consequently, even during the targeted Diffusion phase (diffusion only to the interested nodes) there will be repercussions and the node may not be able to spread the content to all the interested nodes.

2.3.1 Solid application: "Contacts"

The "Contacts" application manages a list of contacts stored on a User POD. In Solid the social graph of a user consists of the contacts stored on his PODs, the accessible contacts of these contacts and so on, where each user is identified by a WebID. In addition, contacts can be viewed as an interface to manage a user's distributed social graph. The "Contacts" application maintains a set of vCards for a user's contacts using the vCard ontology. Each vCard is a resource with a single URI and can contain the user's WebID which it represents, in addition to other fields such as name and email. A user can mark a vCard as public or allow access to the vCard only to a single individual or to a specific group of people (identified by their WebID). One of the most interesting social features in "Contacts" application, enabled by Solid, is the ability to search for Contacts of your Contacts using the link-following SPARQL. A user can search for a vCard in his pods, by matching a search criterion such as name, email or address. In addition, the "Contacts" application can use the link-following SPARQL query to search for a contact in public contacts on pods that can be reached by

WebID in the user's vCards (via link-following). The user obtains a list of vCards that match the search criteria and the URI of each vCard obtained in response indicates the source of this card. This search capability provides an example of the innovative social features supported by a decentralized social platform such as Solid [56].

2.4 Social Internet of Things (SIoT)

The Internet of Things (IoT) is a paradigm that in recent years has received much attention from the scientific world. It is based on the pervasive presence around us of a variety of things or objects that, through a unique addressing scheme, are able to interact with each other and cooperate with their neighbors to reach common goals [32]. These connected objects, at the base of the Internet of Things, are named as "smart objects". The IoT will greatly influence behavior and many everyday-life aspects of users in the working and domestic fields. In reality only the resolution of the main issues of the IoT will allow the full interoperability of devices, while ensuring trust, privacy and security. One of the main problems will be related to the fact that the things that make up the IoT will have low resources in terms of computation and energy capacity. As a result, the new solutions must pay particular attention to the efficiency of resources, in addition to the obvious problems of scalability [5].

The best definition that can be given at the moment for the Internet of Things is the following: *"a conceptual framework that leverages the availability of heterogeneous devices and interconnect solutions, as well as augmented physical objects that provide a shared global information base to support the design of applications that involve both people and representations of objects".*

Accordingly, every person and thing in IoT has a virtual counterpart that can be localized, addressed and readable in the Internet (in the Cloud, the Fog, or at the network Edge). Here the objects are prosumers of services and collaborate with other counterparts to reach common goals. This led to design a new generation of "smart objects", that will have to operate in an extremely complex context. It's unlikely that the single object will have the capacity to deal with such complexity alone. Like some species of animals that, to cope with complexity and the difficulties of the environment in which they live, have created a dense network of social relationships, in the same way a new generation of social objects as been

envisaged. This led to an augmented IoT, called Social Internet of Things (SIoT), that applies the typical concepts, solutions, and technologies of social networks [81]. Considering the evolution of objects, an important first step is that from res sapiens (smart object) to what we call res agens (an acting object). The res agens is able to translate the awareness of causal relationships into actions. A further important step is that to res socialis, an object that is part of and acts in a social community of objects and devices. In this context the social networks of objects are built between objects that are owned by human beings that may have no connection between them [7].

The application of the social networking principles to the IoT (SIoT) can lead to different advantages in network navigability, in scalability, in discovery of objects and services. Moreover, powerful models of trustworthiness management designed for social networks can be reused to address IoT-related issues. In SIoT a basic set of inter-device relationships are defined:

- "Parental object relationship" (POR): established between objects belonging to the same production batch.
- "Co-location object relationship" (C-LOR): established between objects used in the same location.
- "Co-work object relationship" (C-WOR): established between objects that often cooperate together to provide a common IoT application.
- "Ownership object relationship" (OOR): established between objects that belong to the same user.
- "Social object relationship" (SOR): established between objects that often come into contact because their owners come into contact with each other during their lives.

The creation and management of such relationships can take place without human intervention [4], but always strictly complying to rules set by the devices' owners. To them several further types of friendship have been added over the time, driven by specific applicative environments (e.g., in Social Internet of Vehicles).

2.5 Related Works

2.5.1 DOSN classification

DOSNs are considered in many works in the literature as a possible solution to give users greater control over their data and, at the same time, to overcome typical problems of centralized online social networks, such as privacy, performance bottleneck, single point of failure, need to be online in all transactions, unexploited locality [35, 36, 61, 95].

In [74] DOSNs are classified into two categories: Web-based and P2P-based DOSNs. The first category, which is considered as a basis in our work, is characterized by a distributed web server infrastructure, and includes systems such as Diaspora [96] and "Friend-of-a-Friend" (FoaF) [94]. These solutions need web space or deploying web servers. Users can publish their profiles in their Web space and manage access control rules (access authorizations) locally, to specifically allow the recovery of attributes and resources reserved for the selected users. Web links to other users' profiles are used to represent the Contact List and thus recreate the social graph.

The second category includes systems such as Likir [3], Peerson [14], Safebook [25], which exploit the advantages of the P2P principle to allow the publication, search, and retrieval of profiles and their attributes, very similarly to conventional P2P file-sharing systems. In them, the resources are kept locally and the profiles are stored in the local devices instead of the Web, and controlled by the users themselves. In P2P-based DOSNs one of the main challenges concerns the replication of user profiles, which must always be available online even when the user is offline. The issue of data availability/persistence is addressed in [35].

In summary, while web-based systems rely on a dedicated web space where user profiles can be stored and retrieved, P2P-based systems exploit only the local and shared resources of the P2P overlay. Obviously, exploiting the rather unreliable storage services of other peers, which are subject to churn (i.e., nodes enter and leave the network, or they change state from online to offline continuously), requires more sophisticated means to keep the data available, which in turn causes overhead and higher implicit costs shared among the participants. Differently, in our research we follow a Solid-like web-based system design

and, since the PODs in which the profiles are stored are always available online, there is no need for replicas of the same profile.

2.5.2 Content sharing and diffusion

Our reference scenario is characterized by the presence of Interest Communities (i.e., sets of nodes that share the same interest) that are separated, disconnected, and unreachable from each other. This represents the problem we aim to solve.

The reasons for which the concerned nodes can be unreachable are the most disparate and vary according to the scenario considered. They can be related *(i)* to the overlay topology; for example, in Friend-to-friend (F2F) networks, communications can only take place between "friends"; *(ii)* to the type of content diffusion; for example, in the case of targeted diffusion, the content is sent only to the interested nodes without any bridging performed by non-interested intermediate nodes; or *(iii)* to the fact that the interested nodes are located in different network Partitions (i.e., areas of a network that are connected in the presence of a mobile node and disconnected in its absence) caused by churn phenomena, for example. This is not an exclusive feature of DOSNs, but it may also occur in other systems, ranging from mobile ad-hoc to opportunistic networks.

In general, the goal of a content-sharing system is to move content items to devices owned by users who want to access such a content [73]. In our reference context the goal is very similar, as a Source node wants to spread its content to the highest possible percentage of reachable interested nodes. The main issue in such as Scenario is that, when diffusing the information relevant to Source interests both to the interested and not interested nodes during the Discovery phase, not all nodes will be equally cooperative and reachable. Also, they won't be equally willing to authorize access to their PODs and to re-forward information on behalf of the Source to their Contacts (otherwise unreachable directly by the source). It shall be considered that while the Contacts are stored in the User PODs and therefore are always available online, the SIoT Contacts are stored locally on the devices themselves and, thus, are not always reachable. It is important to understand which nodes to send the information on the Source's interests during Discovery and which nodes to spread the content of the Source in the Diffusion phase.

The solutions proposed in the literature to spread contents to interested users belonging to different Communities of Interest/Partitions are manifold and usually depend on the type of scenario and related assumptions. In [73] an ad-hoc content-sharing system for mobile devices is proposed within an opportunistic network scenario. In it, mobile devices share contents and interests and can store-carry-forward contents on behalf of other nodes, based on interests, therefore connecting otherwise disconnected devices. In [73], Haggle introduces a content delegation mechanism that allows to selflessly disseminate a given number of items based on the interests of other nodes (third-party nodes). This is particularly important in the presence of network Partitions. The mobile nodes that provide this type of connectivity are called "data mules", reflecting the idea that they carry data between otherwise partitioned areas of the network, [11]. In [73] it is clearly explained that depending on the network structure and the users' interests, a content may not reach the interested nodes without exploiting data mules that carry the content although they are not interested in it. Although the scenario is very different from ours, it presents a problem very similar to the one we intend to address. Here too, the goal is to spread the contents to devices owned by users who need such contents [73]. In our scenario, however, we assume that the content can be diffused *only to the interested nodes* and, consequently, there will be no uninterested data mules that connect the separate Communities of Interest (Partitions). On the contrary, during the Discovery phase the information on the Source's interests will be sent (anonymized) both to interested and not interested nodes. In this case we are not talking about "data mules", but about intermediaries that allow the Source to reach nodes that otherwise it would not be able to directly reach.

Several studies aimed to design valid mechanisms to diffuse the content as much as possible to the interested nodes. For example, [49] analyses how maximizing the total weight of the "content receivers", which is measured in proportion both to how much the users themselves are interested in the content and based on their ability to connect with other interested users. There are more reasons why many works decide to spread the information only to the interested nodes. First, they are more active during the diffusion process [48]. The second important reason is the guarantee of greater privacy and security. So a possible research goal is to minimize the number of nodes not interested that are involved in the diffusion process [30], [59]. In our case, even in the Discovery phase, when information

on the Source interests is diffused, we are able to guarantee privacy by sending it totally anonymously.

In [12] the goal is to make the information available in those regions where there are interested users without overusing the resources, i.e., avoiding flooding. Third, it must be considered that the interested nodes are also the desired receivers, and therefore they are motivated to participate in the diffusion process to receive the content of interest. Finally, users with similar interests also have higher frequency and probability of communicating with each other, which allows a more efficient diffusion [48]. This feature is exploited to improve content diffusion in [22]. In particular, a user will download the content from another user she meets if and only if *(i)* the topic of the content is of her own interest, *(ii)* it interests her friends, or *(iii)* it interests the users she meets. In our framework the content of the Source is diffused only to the interested nodes. Differently, during Discovery intermediaries (even if not interested) spread information on the Source interests to allow the Source to reach many more interested nodes. We will demonstrate and quantify this advantage in Section 6.7 through different simulations at varying of different parameters. As a result, on the one hand, during Discovery, the Source is able to indirectly reach a higher number of interested nodes, thanks to the presence of intermediaries. On the other hand, during Diffusion, the Source manages to send its content as efficiently as possible to the interested nodes only. Privacy is guaranteed during the Discovery phase thanks to the anonymity of the information disseminated, and during the Diffusion phase by sending the content directly to the desired receivers only.

Due to the churn in the Social Overlay (SO), a limited set of links may be available for reconfiguration and cause transient network partitions, which are responsible of long unacceptable delays in the content diffusion phase [61, 60]. As a solution to this issue, a hybrid architecture is proposed in [61] that on the one hand exploits the SO for fast, decentralized and friend-to-friend communications, but occasionally exploits access to the cloud to overcome the high delays caused by the purely decentralized solution. Comparing it with our work, we see the analogy between the transient network partitions and the separate Communities of Interest, but the focus is slightly different because, while in our case a Source node wants to spread a content to all the interested network nodes, the goal in [61] is to allow efficient profile-based communication between direct friends. In both cases, however, the

problem lies in the limited set of available links. Unlike the solution presented in [61], in which the concept of purely distributed architecture is lost, in our case we want to show the advantages in terms of reachability that are obtained by extending the set of friendships through social object relationships (SIoT Contact List), while still maintaining a distributed approach to reach any interested node belonging to other communities.

Chapter 3

Enhanced Social Internet of Things

3.1 Community

3.1.1 Community definition

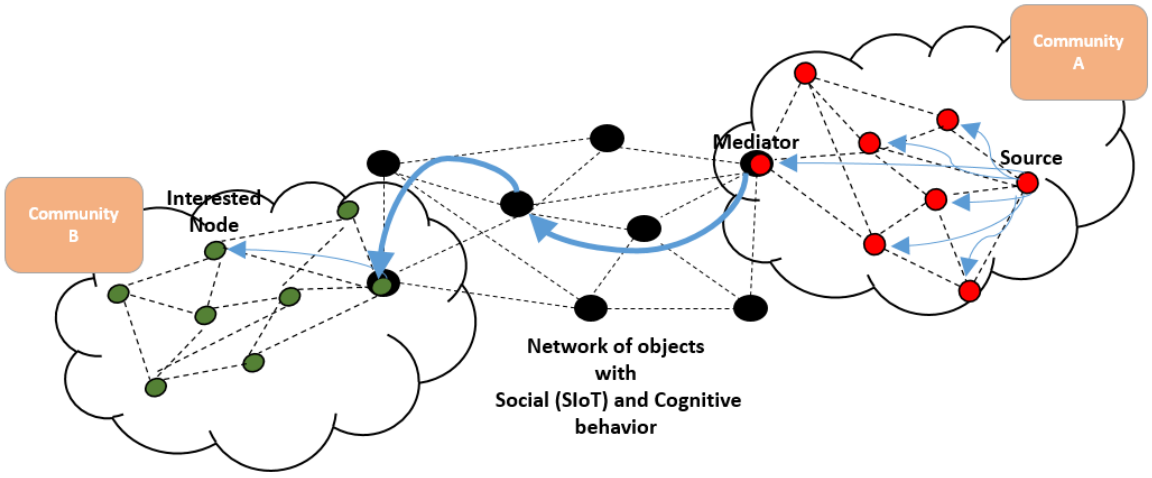


Fig. 3.1 Example of communities interconnected by the SIoT network of cognitive objects.

To understand what is a community in our scenario, we use Fig. 3.1.

A community *A* is made up of nodes that share at least one common interest, e.g. in Fig. 3.1 the nodes in red share the "Football" Interest. We will assign the same color to nodes belonging to the same community. This does not mean that nodes belonging to a community

cannot have other interests in common with nodes belonging to other communities, but, to simplify, we in Fig. 3.1 do not illustrate this case.

Each direct dashed line between a pair of nodes indicates that both nodes can be mutually reached by performing a Search for that specific interest. They can be reached using the "Contacts of Contacts" mechanism, as they are authorized to access the contacts of their contacts. For more accuracy, we should also mark the direction of the dashed line (one-way or two-way) on who can reach the other by doing the search. If two nodes within the same community are not connected by a dotted line, it means that they cannot be reached via direct search (if one of the two nodes searches for that interest, the other's vCard will not be returned). If two nodes that share the same interest are in two different communities, it means that none of the nodes in the former community can reach any node in the second community.

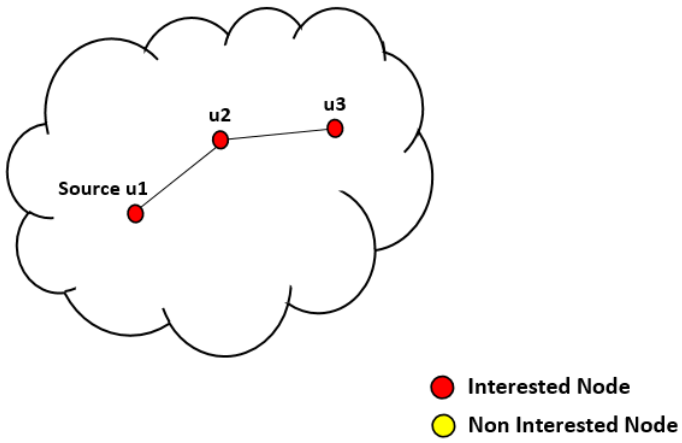


Fig. 3.2 Scenario where $u2$ is a contact of $u1$, and $u3$ is a contact of $u2$.

For clarity, we consider just the $u1$ search. Let's say $u2$ is a contact of $u1$, and $u3$ is a contact of $u2$ in Fig. 3.2. We assume that $u2$ gives to $u1$ the authorization to see its own contacts. The dashed arrows between $u1$ and $u2$ and between $u1$ and $u3$ will be established (in Fig. 3.3). The arrow is directed from $u1$ to $u2$, and from $u1$ to $u3$ because it's $u1$ that through the search can reach $u2$ and $u3$. So $u1$, after searching the specific interest, will receive the vCards of $u2$ and $u3$ as a result.

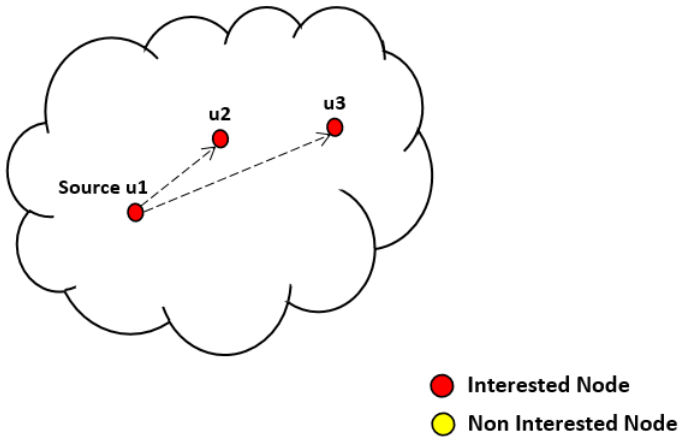


Fig. 3.3 Search for $u1$ in case $u2$ grants authorization to $u1$ to see its own contacts.

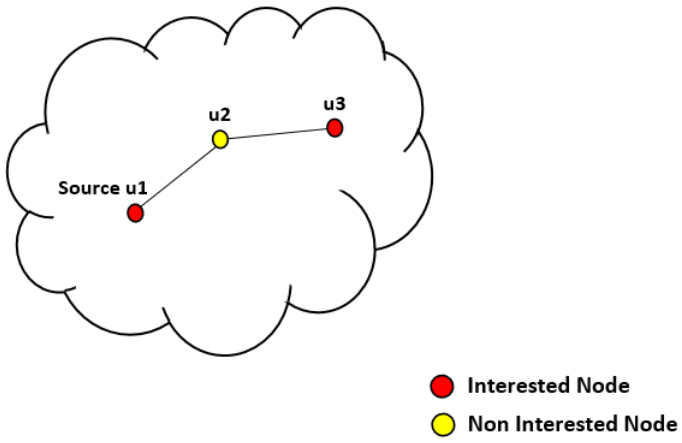


Fig. 3.4 Scenario where $u2$ is a contact of $u1$, $u3$ is a contact of $u2$ and $u2$ is not interested in that specific interest.

Also in the following scenario in Fig. 3.4, where $u2$ is not interested in the specific interest, assuming that $u2$ grants authorization to $u1$ to see its own contacts, a dashed line between $u1$ and $u3$ will be established because $u1$ will be able to reach $u3$ through search (in Fig. 3.5).

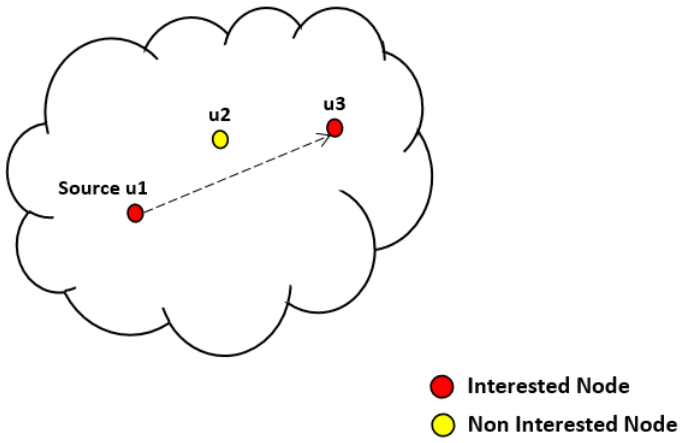


Fig. 3.5 Search for $u1$ in case $u2$ is not interested in the specific interest and $u2$ grants authorization to $u1$ to see its own contacts.

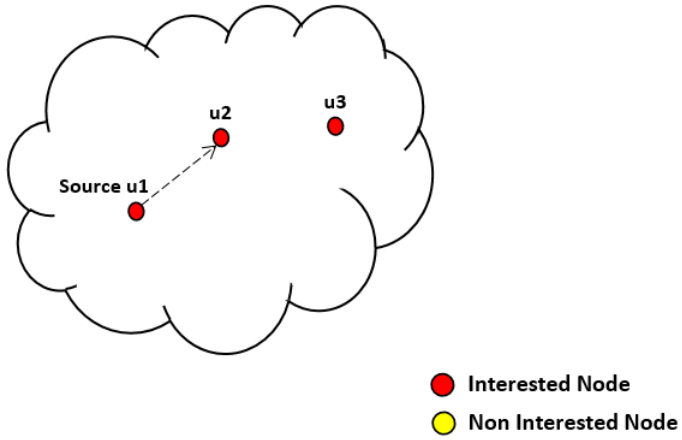


Fig. 3.6 Search for $u1$ if $u2$ does not grant authorization to $u1$ to see its own contacts.

If we were in scenario in Fig. 3.2, but $u2$ did not grant authorization to $u1$ to see its own contacts, we will not have the dashed line between $u1$ and $u3$, because $u1$ will not be able to reach through search $u3$ (in Fig. 3.6). Each user can decide which of his Contacts to make visible and which users to make them visible to. A user can also decide to make his contacts visible to all other users indiscriminately (public mode).

Now we consider the scenario in Fig. 3.2 and we assume that all three nodes are going to search and that all three allow others to see their own contacts. Assuming that $u1$ and $u3$ are contacts of $u2$, we'll get the situation shown in Fig. 3.7.

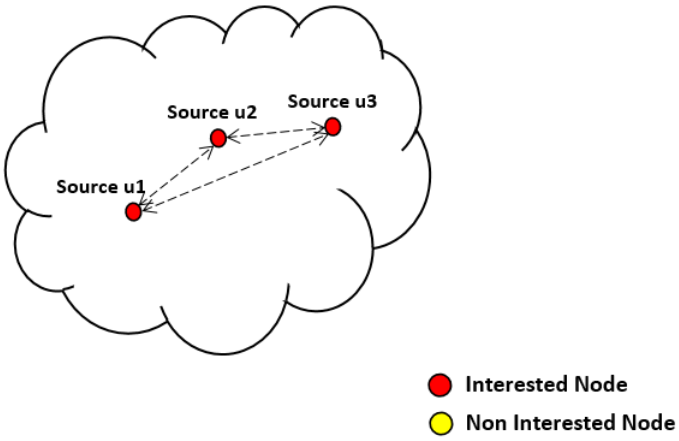


Fig. 3.7 Search for all the three nodes if all the three nodes allow others to see their own contacts. Assuming that $u1$ and $u3$ are contacts of $u2$.

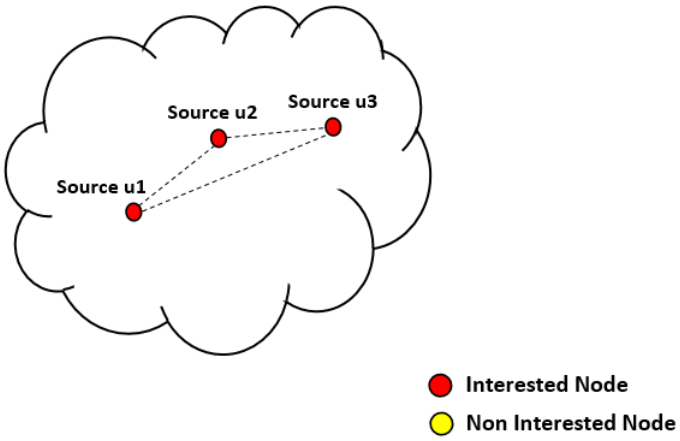


Fig. 3.8 Search for all the three nodes if all the three nodes allow others to see their own contacts with a different symbology. Assuming that $u1$ and $u3$ are contacts of $u2$.

Or with a different symbology in Fig. 3.8.

In brief, a *Community* given a Source that wants to diffuse its content belonging to a specific interest is constituted by:

- the Source node;
- the set of all the nodes interested to that specific interest reachable (through search based on the "Contacts of contacts" mechanism, considering all contacts' authorizations

to view their own contacts) from the Source node. For simplicity, we call them *Interested Reachable Contacts (IRC)*. In this first case, then, they will be the Interested Reachable Contacts from Source (IRCs of Source);

- the set of all the IRCs obtained by considering as Source every node belonging to IRCs of Source;
- recursively the set of all the IRCs obtained by considering as Source each node belonging to the set of all reachable IRCs.

3.1.2 Creating a Community: an example

In this example, we want to examine how a community is created. We suppose that Source $u1$ wants to diffuse its content belonging to a specific interest to all the interested nodes (in red) in Fig. 3.9.

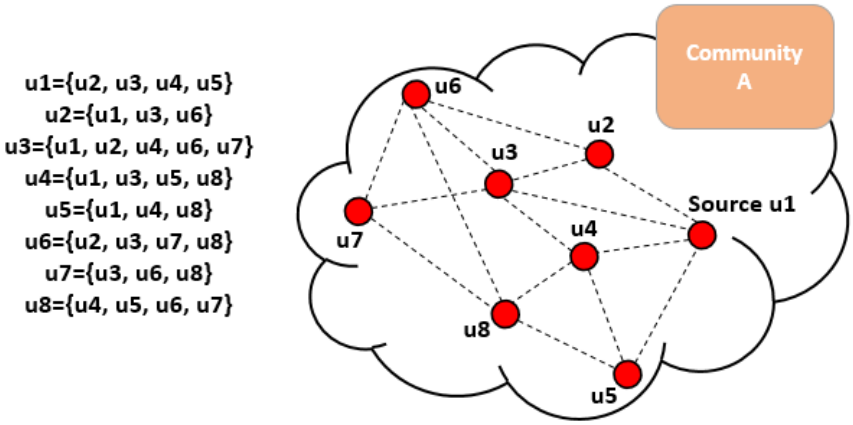


Fig. 3.9 Starting scenario.

Interested Reachable Contacts della Source (u2, u3, u4, u5)

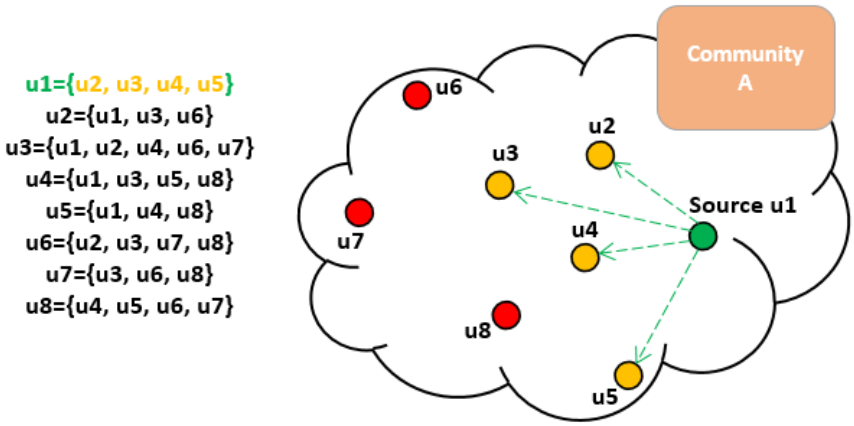


Fig. 3.10 Step 1 - IRCs of Source.

In Fig. 3.10 we represent in orange the set of all the nodes interested to that specific interest reachable (through search based on the "Contacts of contacts" mechanism, consid-

ering all contacts' authorizations) from the Source (IRCs of Source). In particular, we use the following symbols for simplicity: $u1 = \{u2, u3, u4, u5\}$, where $u1$ is the Source node (in green) that performs the search and in curly brackets there are the names of IRCs of Source (in orange), that are the interested nodes reachable through search from the Source $u1$.

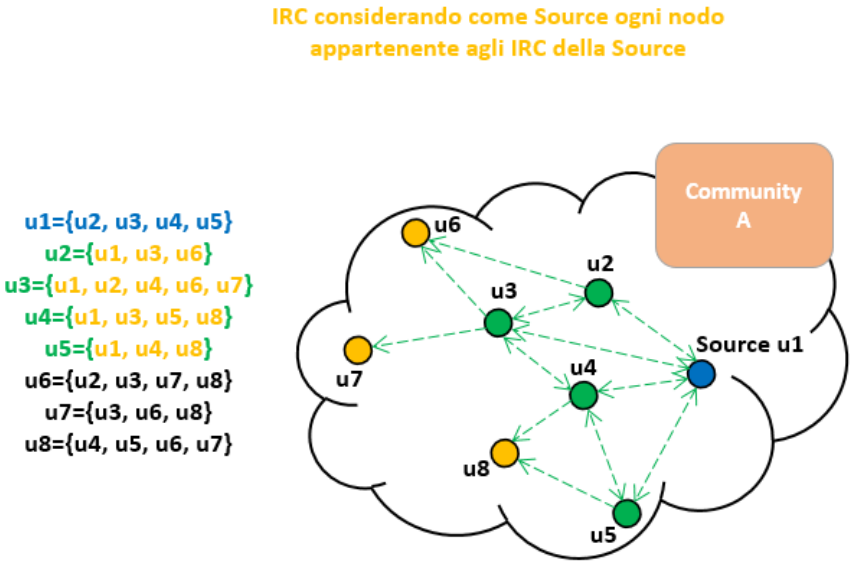


Fig. 3.11 Step 2 - IRCs obtained by considering as Source every node belonging to IRCs of Source.

In turn each of the IRCs of Source ($u2$, $u3$, $u4$, $u5$) can reach a certain set of interested nodes via search. In Fig. 3.11 we represent that $u2$ (in green) will be able to reach through search the set of interested nodes consisting of nodes $u1$, $u3$, $u6$, using the symbology seen above $u2 = \{u1, u3, u6\}$. In turn $u3 = \{u1, u2, u4, u6, u7\}$ and so on. Obviously we do not consider the nodes previously reached. In Fig. 3.11 therefore we represent the set of all the IRCs obtained by considering as Source every node belonging to IRCs of Source. The IRCs of Source are represented in green, their IRCs are represented in orange, the Source node $u1$ considered in the previous Step is represented in blue.

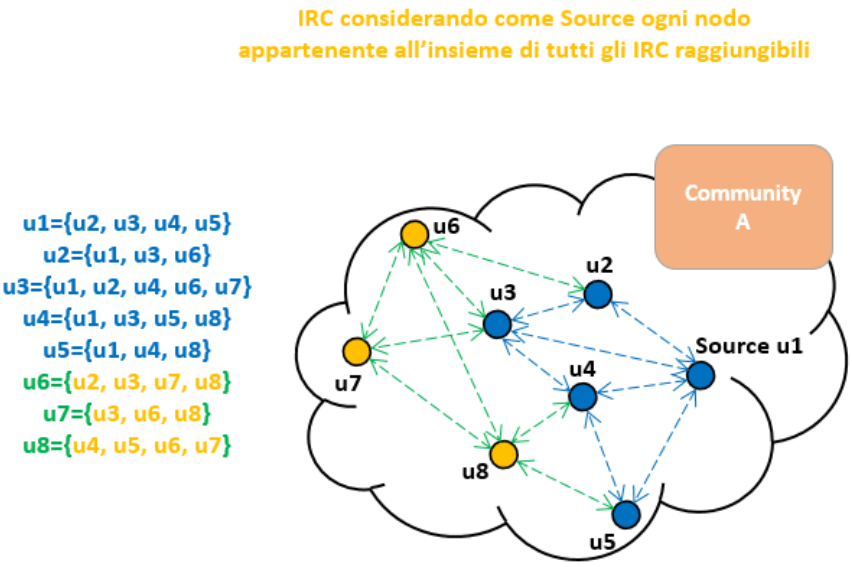


Fig. 3.12 Step 3 - IRCs obtained by considering as Source each node belonging to the set of all reachable IRCs.

In Fig. 3.12 we represent the set of all the IRCs obtained by considering as Source each node belonging to the set of all reachable IRCs. In particular, in blue we represent the nodes already reached in the previous Steps, in orange their IRCs not yet reached.

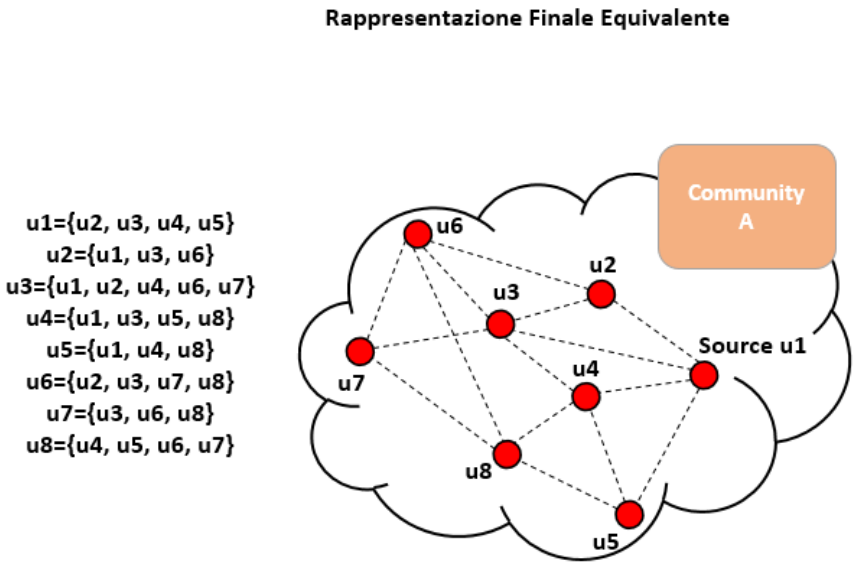


Fig. 3.13 Equivalent Final Representation.

Finally in Fig 3.13 we represent the Community thus obtained (in red).

The Problem (represented in Fig. 3.14) is that none of the nodes in the *A* community can reach the Interested Node (in the *B* Community) through Search. Otherwise the Interested Node would belong to the *A* community.

So:

- two nodes belonging to the same Community:
  - can reach each other Directly through Search (if they are directly connected from dashed line) *Direct IRC (D-IRC)*;
  - can reach each other Indirectly, through other nodes (if they are connected through a dashed line path) *Indirect IRC (I-IRC)*. The Source can reach through intermediary nodes along the path other nodes otherwise unreachable;
- two nodes belonging to different Communities:
  - cannot reach either Directly or Indirectly.

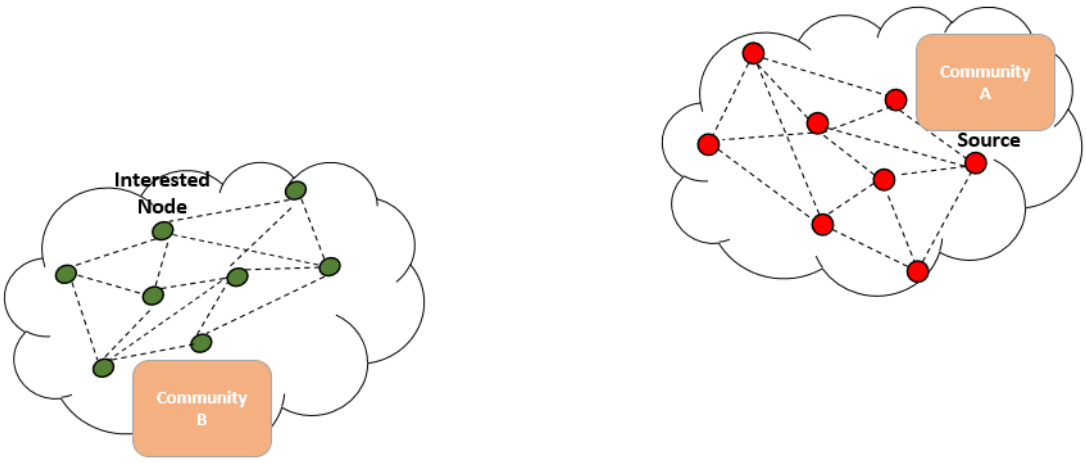


Fig. 3.14 Scenario with two Separate Communities of Interest.

Briefly summarizing, a node of a community that intends to spread content belonging to a specific interest is called Source. This node can reach with its contents all the nodes interested in that specific interest, directly thanks to the permissions received from its contacts to view their contacts (defined as *Direct Interested Reachable Contacts (D-IRC) of*

the Source. Furthermore, its Contacts may relaunch the search reaching further interested nodes (otherwise not reachable from the Source) by operating as Sources. This hopefully allows the Source's content to reach all the nodes interested in that specific interest (the respective IRCs), which for the Source are to be considered Indirect IRCs (I-IRCs). Therefore, a Source node will be able to disclose its content to all Interested Nodes of the Community *A* (D-IRC and I-IRC). Obviously all the nodes reachable in the described way constitute the community *A*. We assume that if two nodes that share the same interest are in two different communities, it means that none of the nodes in the former community can reach any node in the second community. In Fig. 3.1 the node marked as "interested node" is a node that cannot be reached by search from any node belonging to the community *A*, otherwise it would belong to it, and therefore it is part of the community *B*. To diffuse the content also to Interested Nodes belonging to other communities, it is necessary to exploit the SIoT.

So the idea is that through SIoT network an *A* community Source will be able to reach an Interested Node belonging to another community and will therefore be able to diffuse the content to it.

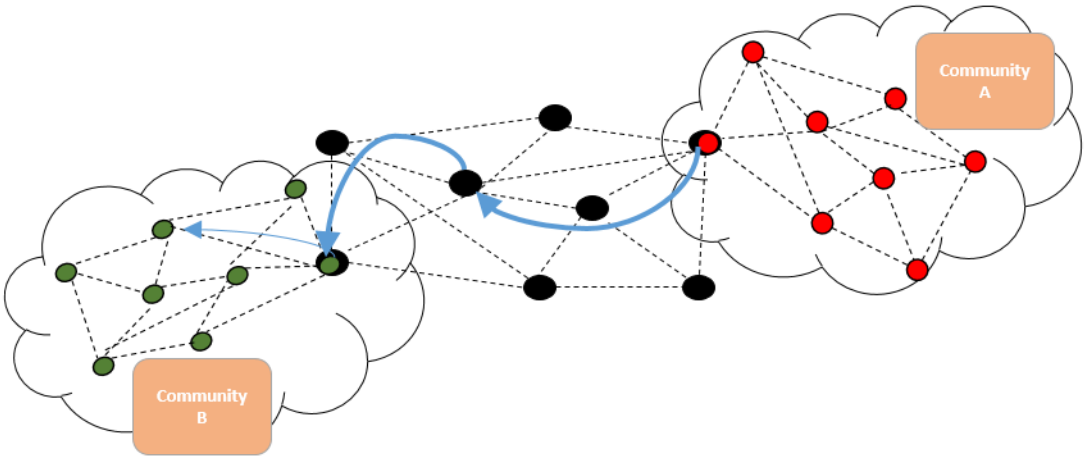


Fig. 3.15 Scenario with two separate communities of interest and the SIoT network that connects them.

So it's true that an *A* node that makes the search (discovery) will not obtain as result all the interested nodes belonging to the *A* Community, it will only get the D-IRC (Direct IRC) of the Source *A*. However it will be able to diffuse its content to all the Interested Nodes in the *A* Community (D-IRC + I-IRC). The *A* node can't diffuse the content to the interested

nodes belonging to the *B* community. So in short, the *A* community is made up of all the interested nodes to which the node *A* can diffuse its content (directly or indirectly).

We can use SIoT network to diffuse content also to Interested Nodes belonging to other communities. Let's look at the general mechanisms at the basis of SIoT network that we're talking about.

3.2 Mediator Object and Cognitive Object

Suppose we consider a distributed social network made up of several separate communities. In Fig. 3.16 we only consider two communities, which for simplicity we call *A* community and *B* community.

In general, the content can only be spread within the specific community to which its Source belongs, but also in another community may be other devices interested in it.

The idea of the "Mediator Object" is defined in this context and its objective is precisely to "mediate" the propagation of the content from community *A*, in which it is generated by the Source, to community *B*, where the node interested in receiving is located.

To achieve this, between the two communities it is necessary that there are objects that have a "Social and Cognitive" behavior and interact with one another. In Fig. 3.16, in fact, between the two communities we consider a SIoT Network made up of "Cognitive" objects.

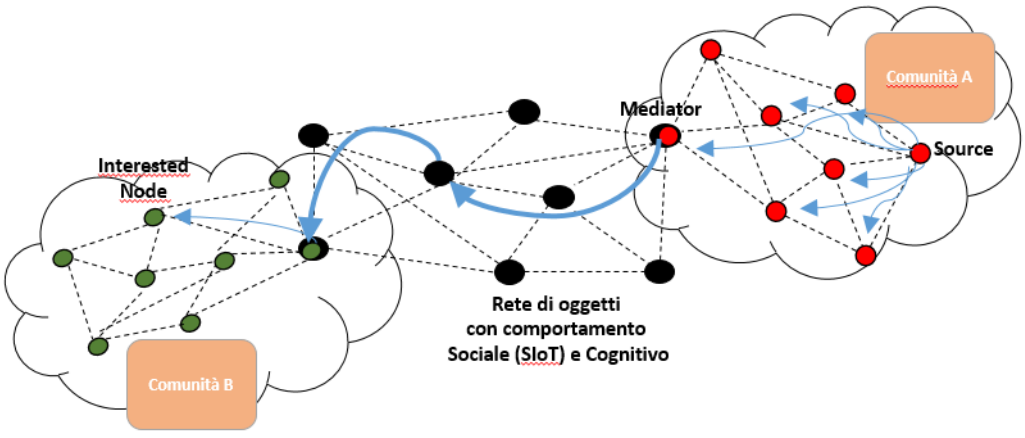


Fig. 3.16 Scenario with Mediator Object.

A "Cognitive Object" has the ability to proactively search the Social Network of Objects, which it belongs to, (through the use of a SIoT platform) and to understand from profiles/Interests Descriptors (defined in the following of this Section) and previous events whether "friends objects" from other communities may be interested in receiving a certain content. The way it happens will be described in the following.

An exemplary use case (in Fig. 3.17) is given by a cognitive object which understand, through the mechanism described below, that the news circulating in Community *A* of soccer fans may also interest another "friend" and "trusted" object whose owner belongs to Community *B* of soccer bettors, for example. We assume that a social object relationship was indeed created between the devices, according to the SIoT rules. The devices became friends, because, for example, they often came into contact in a soccer stadium, although their owners do not know each other.

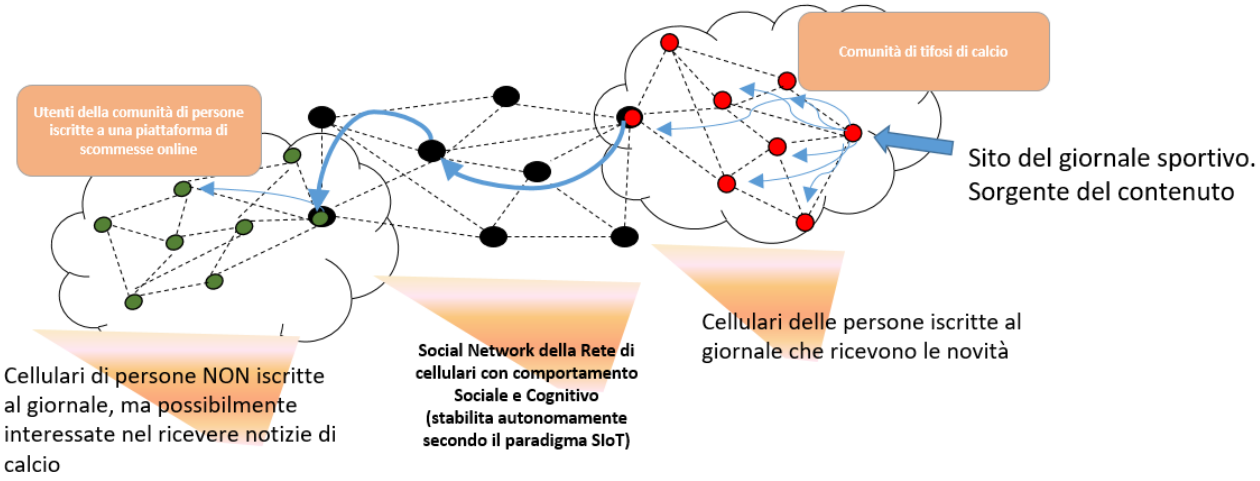


Fig. 3.17 Scenario with Cognitive Objects.

In this context, a first problem is linked to the way in which a cognitive object can understand if "friends objects" from other communities may be interested in receiving a certain content. To this end, each device is associated with an Interest Descriptor, i.e. a vector of words (keywords) that describes the interests of its owner.

There are several ways to derive the Interest Descriptor that depend on the information available. Without losing generality, we can refer to an exemplary solution based on the Visual User Interest Profile (VUIP), as in [97].

In order to understand what the VUIP is, it is necessary to introduce the concept of user profiling, which can be defined such as the process of identifying data relating to the user's domain of interest. A device can infer the owner's profile based on a set of images, accessed through that device, that describe her interests (as in Instagram, Facebook, etc.). For example, by leveraging deep learning techniques, from the images it is possible to obtain the corresponding VUIP (that is a vector of keywords derived from images) that can be used as an Interest Descriptor.

During the discovery process, better described in the remainder of the paper, the device itself can send its VUIP (coinciding with the VUIP of the owner) to its first social neighbors (i.e., nodes with which it has already established at least one social object relationship) in the Social Internet of Things. We assume that each device has such a capability of profiling the interests of its owner and creating the corresponding VUIP.

Obviously, during the whole process of browsing the SIoT, the Source's VUIP exchanged among the devices must remain anonymous.

3.2.1 Visual Feature Extraction

We suppose that each user is associated with a set of images describing her interests. To build the visual user interest profile of each individual it is necessary to extract the semantic content of her own images. Mining the semantic content to extract visual features is an application of content-based image retrieval (CBIR), which has been an active research field for decades [53]. Recently higher-level semantic extraction, typically based on deep learning, has gained favour [38]. Wang et al. [88] proposed a ranking model trained with deep learning methods, which is able to distinguish the differences between images within the same category. Babenko et al. [9] and Wan et al. [87] both proved that pre-trained deep CNNs (Convolutional Neural Networks) for image classification can be re-purposed to image retrieval problem. In [97] the authors want to model user visual interest, so their approach is similar to Babenko's and Wan's methods. They reused a pre-trained deep learning network to retrieve and rank hotels images. In [97] image features are extracted with a CNN (Convolutional Neural Network) where the distribution over classes from the output layer is used as the descriptor for each image. This CNN produces a distribution over 1000 visual object classes for the VUIP. Because each dimension of the feature vector is actually a

class in ImageNet, the descriptor helps to bridge the semantic gap between low-level visual features and high-level human perception. Instead of training a CNN by theirselves, in [97] they employ a pre-trained model on the ImageNet "ILSVRC-2012" dataset from the Caffe framework [38]. Every image is forward passed through the pre-trained network and a distribution over 1000 object classes from ImageNet is produced. This 1000 dimension vector is regarded as the descriptor of the image and saved in the VUIP. The VUIP can contain the descriptors of many images.

In our work we use VUIP in a different way. In particular, instead of looking for images of hotels whose descriptors are more similar to the individual's VUIP, we are looking for other users whose VUIP is as similar as possible to that of the considered individual. Obviously, a specific software that allows the extraction of the VUIP through deep learning must be installed on each device. Also in our work we use Cosine Similarity to establish the degree of similarity between VUIPs.

In Matlab we can load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into 1000 object categories. As a result, the network has learned rich feature representations for a wide range of images. Matlab gives us the possibility to use different pre-trained networks, for example the Inception-v3 convolutional neural network.

For further details on the visual feature extraction procedures, please refer to the articles mentioned above.

3.2.2 Software tools to extract the VUIP on smartphone

On-device deep learning engines are finding their way into smartphones. For example, Apple has introduced a neural engine as part of the A11 Bionic chip, and Huawei has introduced the Kirin 970 neural processing unit (NPU). The smartphone industry is also working toward dedicated processors to speed up on-device deep learning in contrast to cloud servers in order to cope with real-time implementation issues and the need for Internet connection. In addition, on-device deep learning helps to alleviate security or privacy concerns due to data storage on servers. Considering that smartphones are equipped with multi-core CPUs, multi-threading is used here to reduce computation time toward achieving real-time throughputs.

Furthermore, the open-source deep learning software tools have reached a maturation point in terms of libraries for on-device deep learning deployment [82].

In [82] is presented how to deploy DNN (Deep Neural Networks) models on Android and iOS smartphones using publicly and freely available software tools. The steps discussed in this article are aimed at turning DNN models into apps for smartphones.

The main publicly and freely available libraries that are widely used include Caffe [38] (developed by Berkeley AI Research), TensorFlow [1] (developed by Google), PyTorch [PyTorch] (developed by Facebook), and CNTK [Toolkit] (developed by Microsoft).

In [82] are selected TensorFlow, Keras [Keras], and CoreML [ML] based on (i) their easy portability to mobile devices and (ii) active support by their developers. The steps involved in turning DNN models to smartphone apps are applied to six popular convolutional neural networks: ResNet50, InceptionV3, SqueezeNet, MobileNet, Dense-Net, LeNet. The six popular CNNs are benchmarked based on the Modified National Institute of Standards and Technology database (MNIST) [MNIST] dataset, which is considered to be a gateway dataset for exploring deep learning, and the widely used ImageNet Large Scale Visual Recognition Competition (ILSVRC) [76] dataset. The benchmarking results have shown that the deep learning models are implemented on smartphones without any significant loss in accuracy compared to PCs. It has also been shown that the use of multi-threading leads to achieving real-time throughputs.

For more details, we refer you to reading the article [82].

3.3 C-IOR and mechanisms for its creation

3.3.1 Co-Interest Object Relationship (C-IOR) and operating principles of a mediator device

For our purposes, we also define a new social relationship between devices, the Co-Interest Object Relationship (C-IOR), whose establishment requires the presence of a social graph consisting of the social object relationships defined for the SIoT. A Co-Interest Object Relationship is established between two devices when the VUIPs (Interest Descriptors) of the device are sufficiently similar, i.e., when the degree of similarity between the VUIPs

owned by the devices exceeds a certain threshold. This similarity assessment is carried out if and only if the two devices are already connected to each other through other social object relationships, such as one among the mentioned SIoT relationships, such as C-LOR, SOR, etc. As a notation we will use C-IOR (Interest), e.g. C-IOR (Soccer), if the two profiles share the interest in the "Soccer" subject.

3.3.2 Mechanisms for the C-IOR establishment

Basic mechanism for the C-IOR establishment

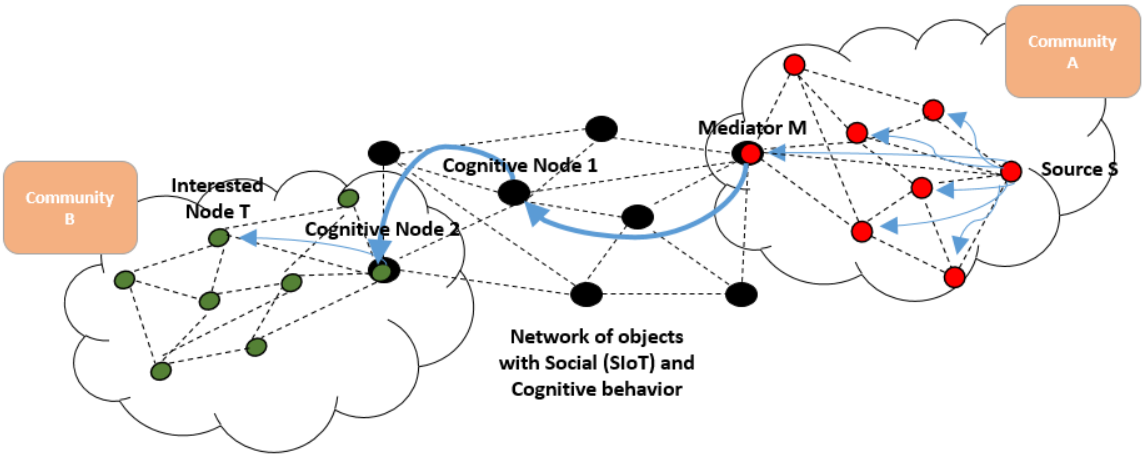


Fig. 3.18 Basic mechanism for the C-IOR establishment.

In general, an Interest Neighbor of a S device is a device connected via C-IOR to S, therefore it is a device sharing one or more interests with S. We describe the basic mechanism that characterizes the phases of a C-IOR discovery and establishment. For greater clarity we will refer to Fig. 3.18.

The S device derives (and then owns) its owner's VUIP. S sends its VUIP to all its first social neighbors. The first social neighbors of S are the nodes with which S has already established at least one SIoT relationship; among them there can be also cognitive objects, including the Mediator Object M. While normal social devices can disclose the VUIP of S exclusively within the community to which they belong, cognitive objects can disclose the VUIP of S also to cognitive objects belonging to other communities. This is because

all the first social neighbors of a normal social node belong to its own community, while the first social neighbors of a cognitive object can also belong to other communities. We assume that all the devices are cognitive. Every first social neighbor of S who receives the VUIP of S, including M, checks whether it is possible for it to establish a C-IOR with S (based on the degree of similarity between VUIPs) and sends in turn the S's VUIP to its own first social neighbors who have not yet received the S's VUIP. The VUIP of S is recursively delivered with a maximum Time to Live (TTL) of 6 hops (small world network property). The TTL is set by the Source and is decremented hop-by-hop. If an interested node in another community, e.g., the T node in Community B, based on the similarity between its VUIP and the received S's VUIP, decides to establish a C-IOR with S, then the C-IOR establishment request of T (which includes T's identity) goes backwards, forwarded by the intermediate nodes, until it gets to Source S. The reason why T cannot directly send a C-IOR establishment request to S but rather the establishment request of T is brought back to the Source by leveraging the intermediate nodes is that the VUIP of S must always remain anonymous in order not to infringe S's privacy. Furthermore, again to maintain the anonymity of S, the intermediate nodes do not have to know the value of the TTL, but only if it is different from zero to know if they must in turn forward the Source's VUIP to their first social neighbors. In fact, if the received TTL were equal to 5, the first neighbors would understand that the VUIP would belong to the previous node. In short, to ensure the privacy of the Source, the VUIP is anonymous and each intermediary knows only the identity of the "previous node" that forwarded the VUIP to it (to which it will have to forward any request of C-IOR establishment coming from the interested node T) and if the TTL is non-zero. After 6 hops, the VUIP expires and is no longer forwarded. Once the C-IOR establishment request of the concerned node reaches the Source, a C-IOR can be established that directly binds node T and node S.

Schematizing the basic mechanism for the C-IOR establishment we will have:

- Step 1. The S device owns its owner's VUIP.
- Step 2. S sends its VUIP exclusively to all its first social neighbors, including the Mediator Object M (cognitive object).

- Step 3. Every first social neighbor of S that receives the VUIP of S, including M, checks whether, based on the similarity between VUIPs, it is possible for him to establish a C-IOR with S.
- Step 4. Every first social neighbor of S that obtains the VUIP of S, including M sends the VUIP of S to its first social neighbors.
- Step 5. Each first social neighbor of M, to which the VUIP of S arrives, in turn repeats Step 3 and Step 4 with a maximum Time to Live of 6 hops (small world).
- Step 6. If a node, eg. the T node, based on the similarity between VUIPs, can establish a C-IOR with S, the establishment request of T goes backwards until get to S.
- Step 7. Once the C-IOR between S and T has been established, the two nodes will be linked directly through the C-IOR.

In short, the cognitive objects, the Mediator Objects in particular, by mediating the propagation of the VUIP (Interest Descriptor) of S from one community to another, and by enabling the establishment of the C-IOR between nodes belonging to different communities, allow the propagation of content across separate communities. Obviously, in implementing this mechanism the VUIP must always remain anonymous and it is not required to know which node it belongs to.

The only information necessary for the nodes to which the VUIP will be forwarded is: the node ID from which the VUIP has been received and if the TTL value is different from zero (to understand if they can in turn re-forward it), which will be decremented at each hop.

The described mechanism, based on the VUIP propagation, is obviously general and can be implemented with any type of Interest Descriptor obtainable for a given user.

Alternative Mechanism for the establishment of the C-IOR

The C-IOR may be Direct (Original) or Indirect (Derived). In the first case, the social object relationship can be established in accordance with the basic mechanism described before. In the second case, the social object relationship is deducted and proposed by the object M (mediator in the establishment of the Indirect C-IOR, not to be confused with the concept of "Mediator Object" seen earlier) in a new and alternative way. Let's look at the establishment

mechanism for the Indirect C-IOR in more detail. For greater clarity we will refer to Fig. 3.19, in which we represent the reference Scenario.

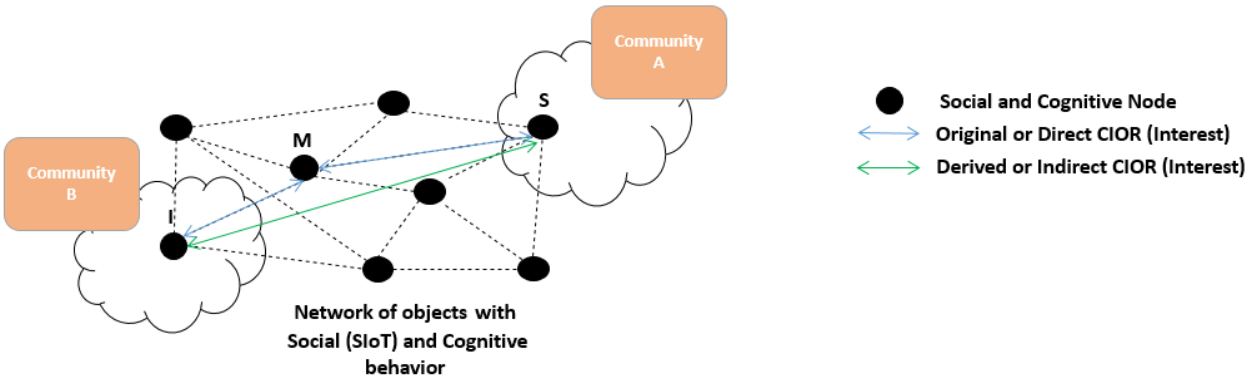


Fig. 3.19 Alternative Mechanism for the C-IOR establishment.

Let's assume that in the past a Direct Co-Interest Object Relationship (characterized by the "Football" interest) has been established between S and M, even if their owners were not friends. We also assume that a Direct Co-Interest Object Relationship, characterized by the same interest, has been already established in the past between M and I, even if their owners were not friends. Now the M device, just looking at its own social object relationships table, realizes that both S and I are linked to him by C-IOR relationships, characterized by the same Interest. As a result, S and I share a common interest. M can now propose a new C-IOR, this time Indirect, to both S and I. If both devices accept the establishment request, the new C-IOR will be established between the two devices. We remember that only the establishment mechanism for the new relationship is indirect, but once the C-IOR has been formed, it will link directly the two devices. Schematizing the Indirect C-IOR establishment mechanism we will have:

- Step 1. We assume that between S and M, a Direct C-IOR exists, characterized by a specific interest. We assume also that between M and I, a Direct C-IOR, characterized by the same interest, exists.
- Step 2. M infers from its social object relationship table that S and I share a common interest.
- Step 3. M proposes to S and to I the establishment of a new C-IOR with I.

- Step 4. If both *S* and *I* accept the *M* proposal, the Derived (Indirect) C-IOR is established between the two devices; otherwise if one of the two nodes does not accept the *M* proposal, the Derived (Indirect) C-IOR will not be established.

In Figs. 3.20 and 3.21 we show two Scenarios where the establishment for the Indirect C-IOR occurs in the way described above.

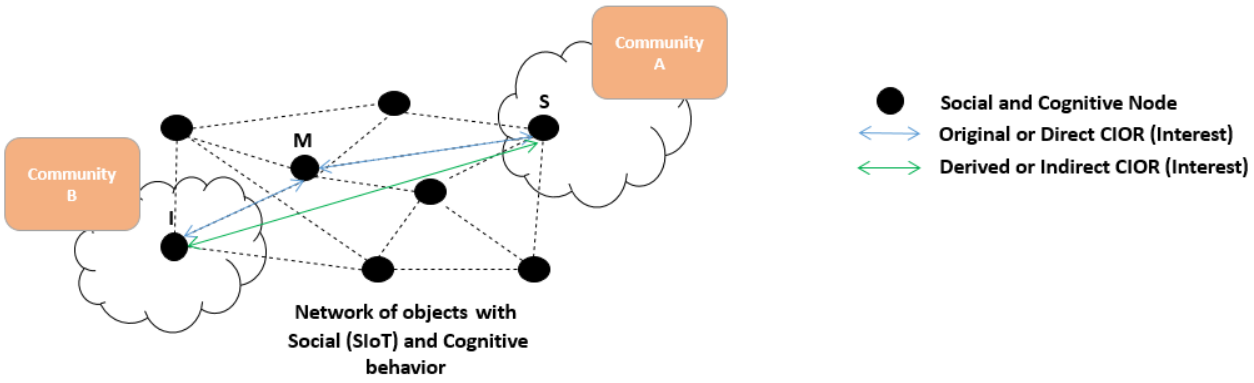


Fig. 3.20 Scenario 1.

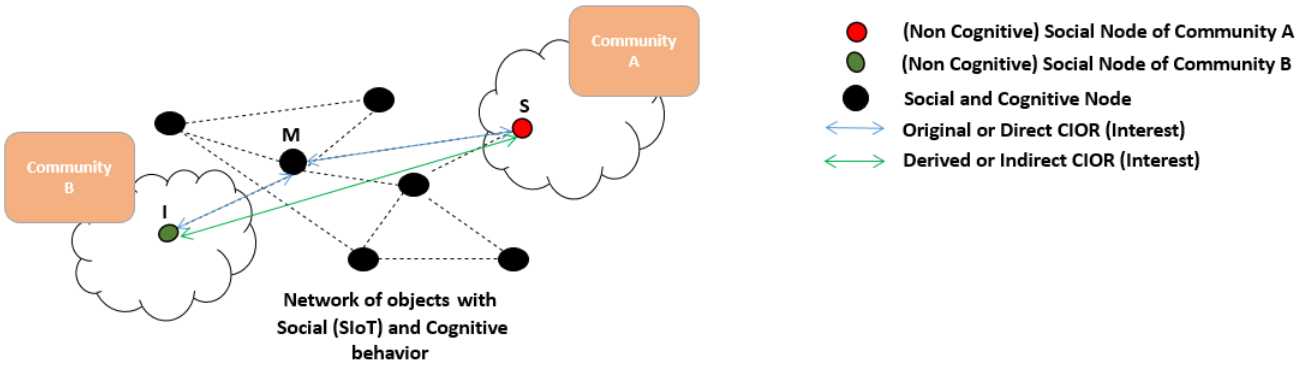


Fig. 3.21 Scenario 2.

In Figs. 3.22 and 3.23 we show Scenarios where the establishment for the Indirect C-IOR is "mediated".

Schematizing the establishment mechanism for the Indirect C-IOR of Scenario 4 in Fig. 3.23 (this also applies to Scenario 3 in Fig. 3.22), we will have that in Phase 1 (for greater clarity represented in Fig. 3.24):

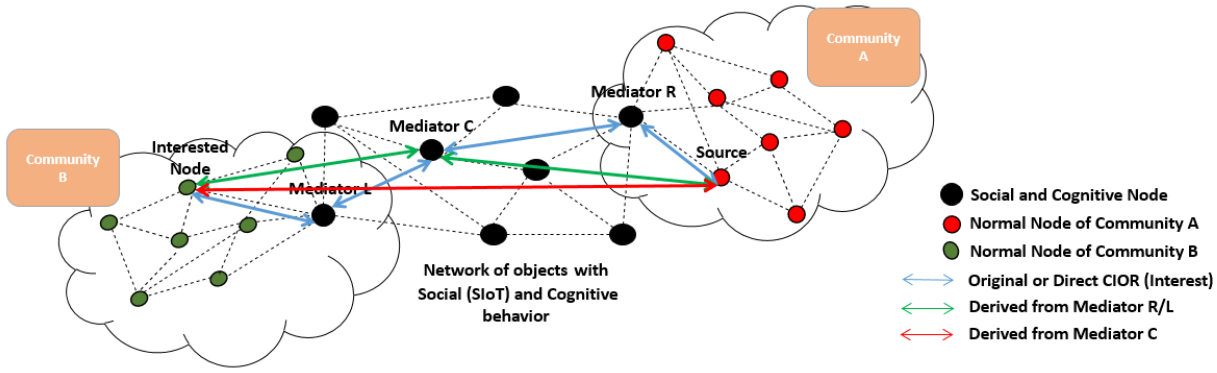


Fig. 3.22 Scenario 3 (mediated establishment).

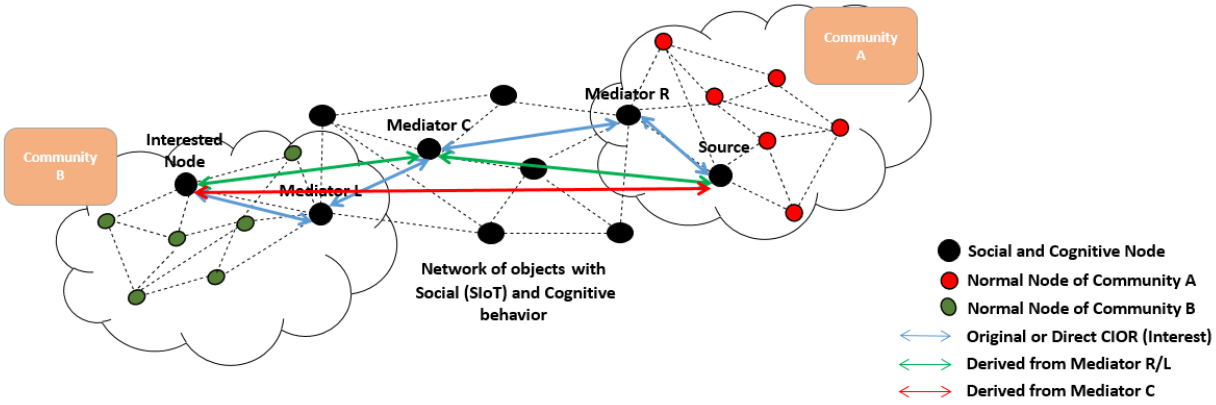


Fig. 3.23 Scenario 4 (mediated establishment).

- a Direct C-IOR exists between Source and Mediator *R*; another Direct C-IOR exists between Mediator *R* and Mediator *C*;
- Mediator *R* infers from its social object relationships table that Source and Mediator *C* share a common interest;
- Mediator *R* proposes to Source the establishment of a new C-IOR with Mediator *C*;
- Mediator *R* proposes to Mediator *C* the establishment of a new C-IOR with Source;
- if both Source and Mediator *C* accept the Mediator *R* proposal, the Derived (Indirect) C-IOR will be established between the two devices;
- a Direct C-IOR exists between Interested Node and Mediator *L*; another Direct C-IOR exists between Mediator *L* and Mediator *C*;

- Mediator *L* infers from its social object relationships table that Interested Node and Mediator *C* share a common interest;
- Mediator *L* proposes to Interested Node the establishment of a new C-IOR with Mediator *C*;
- Mediator *L* proposes to Mediator *C* the establishment of a new C-IOR with Interested Node;
- if both Interested Node and Mediator *C* accept the Mediator *L* proposal, the Derived (Indirect) C-IOR will be established between the two devices.

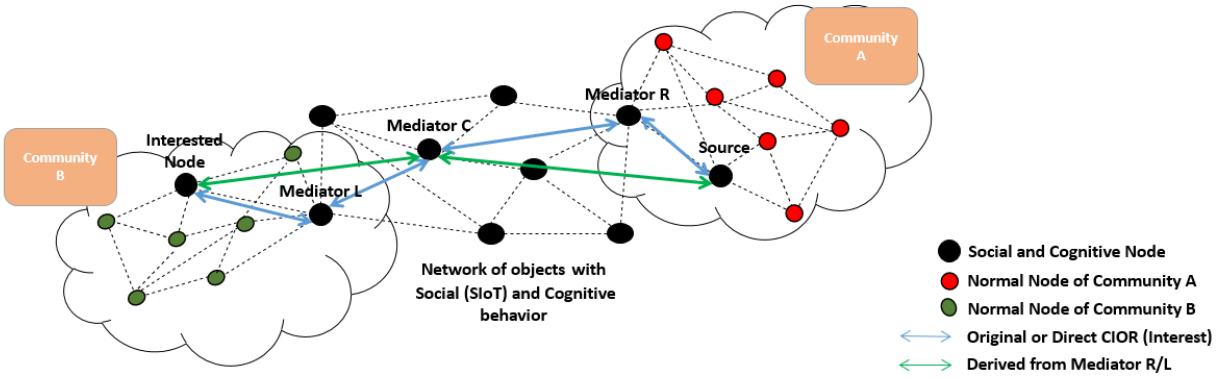


Fig. 3.24 Phase 1 of the establishment mechanism for the Indirect C-IOR in Scenario 4.

Schematizing the Phase 2, for greater clarity represented in Fig. 3.25, we will have that:

- Mediator *C* infers from its social object relationships table that both the Source and the Interested Node share a common interest;
- Mediator *C* proposes to Source the establishment of a new C-IOR with the Interested Node;
- Mediator *C* proposes to the Interested Node the establishment of a new C-IOR with the Source;
- if both Source and Interested Node accept the Mediator *C* proposal, the Derived (Indirect) C-IOR will be established between the two devices.

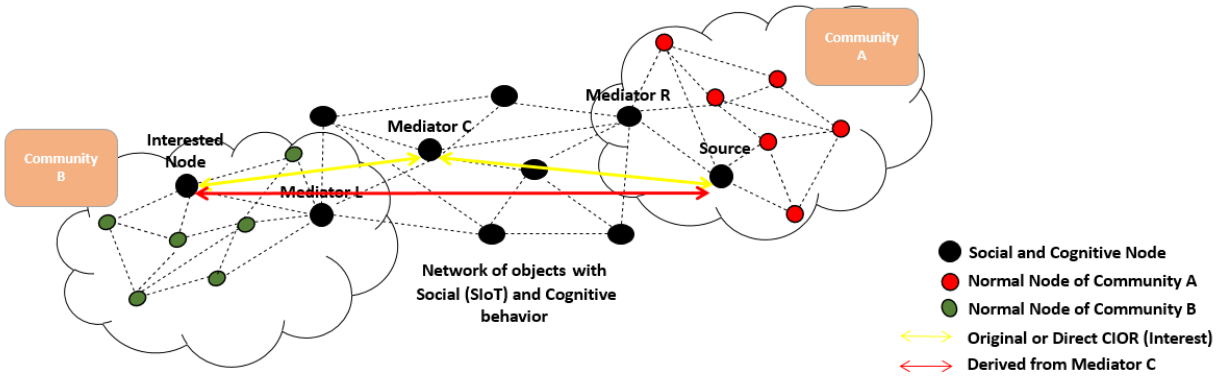


Fig. 3.25 Phase 2 of the establishment mechanism for the Indirect C-IOR in Scenario 4.

3.4 Enhanced Discovery and Enhanced Diffusion

The foundation of the Solid's "Contacts" is the "Contacts of Contacts" mechanism, in line with the more known "Friends of Friends" mechanism. Accordingly, each node performing Discovery (search) can scan, in addition to the Contacts (friends) contained in its User POD, also the Contacts of its own Contacts, which have granted it authorization to access its Users' PODs. This is possible thanks to the link-following SPARQL [56, 78].

We are assuming that in the User POD each user keeps her own list of contacts (*Contact List*). We call the contacts that a user keeps in her User POD Direct First Hand Reachable Contacts (DF-RC). In addition a user can scan the Contacts, contained in the User PODs of other users (her contacts), to which it is authorized to access. We call these contacts Direct Second Hand Reachable Contacts (DS-RC).

On the other hand, the list of nodes, which each device is linked through SIoT relationships (*SIoT Contact List*) is locally stored on the device itself, and specifies the type(s) of relationship(s) through which the nodes are mutually linked. More details on how to create a distributed SIoT network are illustrated in the literature [31]. We call Reachable SIoT Contacts (RSC) the SIoT contacts that a device has stored locally in the SIoT Contact List.

During the Discovery phase, a device will scan: (i) its User POD (i.e., its own Contact List), (ii) the User PODs of its Contacts that authorize it to access, (iii) and the User PODs of the Contacts reachable recursively (through the chain of authorizations) that authorize it to

access. In addition, the device will know its own RSCs contained in its own SIoT Contact List.

Specifically, the node will scan both the interested nodes and the nodes not interested in the content.

The result of the Discovery will return only the interested nodes among the scanned ones.

Consequently in Discovery a device will scan the set constituted by DF-RCs (its own Contact List) plus DS-RCs (the Contact Lists of other users (her contacts) that authorize it to access) plus RSCs (its own SIoT Contact List). For brevity we call them ED-RCs (Enhanced Direct Reachable Contacts). ED-RCs is the set DF-RCs + DS-RCs + RSCs.

The discovery result will be the set DF-IRCs (Direct First Hand Interested Reachable Contacts) plus DS-IRCs (Direct Second Hand Interested Reachable Contacts) plus IRSCs (Interested Reachable SIoT Contacts). For brevity we call them ED-IRCs (Enhanced Direct Interested Reachable Contacts). ED-IRCs is the set DF-IRCs + DS-IRCs + IRSCs.

We use the term "Enhanced" when we leverage also the SIoT Contacts, to distinguish it from the set consisting only of DF-IRCs plus DS-IRCs, excluding the IRSCs.

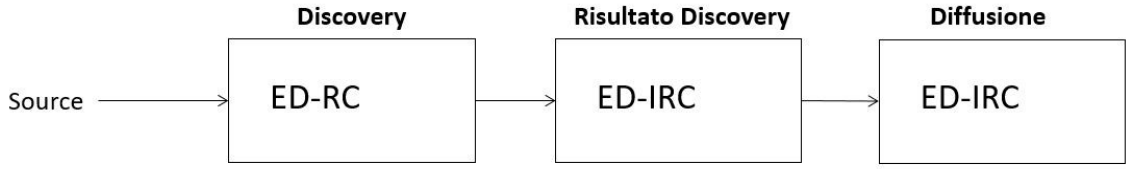


Fig. 3.26 Enhanced Discovery and Enhanced Diffusion Schematization.

In short, with *Direct* we identify nodes reachable in the Discovery phase, therefore directly reachable by the node that wants to disclose the content, while with *Indirect* we mean the nodes that cannot be reached in the Discovery phase, but are anyway reachable in the Diffusion phase through intermediary nodes, which do not authorize (in the Discovery phase) the Source node to access to their User PODs.

After each node has performed Discovery and knows its ED-IRCs, only Interested Nodes will be considered during the Diffusion phase. A node that wants to disclose content can disclose it directly to its ED-IRCs (DF-IRCs + DS-IRCs + IRSCs).

For greater clarity, we present in Tab. 3.1 all the acronyms used in this work.

Table 3.1 Table of acronyms.

Acronym	Description	Set
RCs	Reachable Contacts	
IRCs	Interested Reachable Contacts	
D-RCs	Direct Reachable Contacts	DF-RCs + DS-RCs
D-IRCs	Direct Interested Reachable Contacts	DF-IRCs + DS-IRCs
I-RCs	Indirect Reachable Contacts	
I-IRCs	Indirect Interested Reachable Contacts	
DF-RCs	Direct First Hand Reachable Contacts	
DF-IRCs	Direct First Hand Interested Reachable Contacts	
DS-RCs	Direct Second Hand Reachable Contacts	
DS-IRCs	Direct Second Hand Interested Reachable Contacts	
RSCs	Reachable SIoT Contacts	
IRSCs	Interested Reachable SIoT Contacts	
ED-RCs	Enhanced Direct Reachable Contacts	DF-RCs + DS-RCs + RSCs
ED-IRCs	Enhanced Direct Interested Reachable Contacts	DF-IRCs + DS-IRCs + IRSCs
UCs	Unreachable Contacts	

Chapter 4

Trustworthiness in SIoT

4.1 Background

First of all we see the bases on the Trustworthiness Theory. At the beginning of the Internet and the Web, there was not the problem of determining if someone online was reliable. Indeed all users online were motivated by common goals, and had strong mutual trust. Only after that the new technologies were opened to the public and for commercial uses, emerged malicious behaviors. As a result, in this scenario, grew the need to create methods to ensure Trustworthiness. The Trustworthiness is a relationship with a direction between two parties that can be called "trustor" and "trustee". The trustor is a "thinking entity", which has the ability to make assessments and make decisions based on information received and past experience. The trustee can be anything. For example, in the relation "Bob trusts Eric", Bob is the trustor, Eric is the trustee. A Trust relationship has a "scope", which means that it applies to a specific purpose or domain of action, e.g. "Bob trusts Eric to be a good car mechanic", "to be a good car mechanic" is the scope of trustworthiness in this case. Trustworthiness is mutual when both parties trust each other with the same "scope". The literature uses the term Trustworthiness with a variety of meanings. The two main interpretations consist in considering Trustworthiness as the perceived reliability of something or somebody, called "Reliability trust", and considering Trustworthiness as the decision to enter into a situation of dependence, called "Decision Trust". The complete definitions are as follows:

- Reliability Trust: “Trust is the subjective probability by which an individual, *A*, expects that another individual, *B*, performs a given action on which its welfare depends”.
- Decision Trust: “Trust is the extent to which a given party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible”.

From the first definition we understand that trustworthiness is subjective and includes an element of prediction or expectation on future behavior. From the second that trustworthiness implicitly includes environmental factors and related to the situation. At this point we define two fundamental concepts of the theory of Trustworthiness.

- The Functional Trust is defined as: “trust in the ability of a node to provide services”.
- The Referral Trust is defined as: “trust in the ability of a node to provide recommendations”.

Both the Functional and the Referral Trust can be Direct or Indirect.

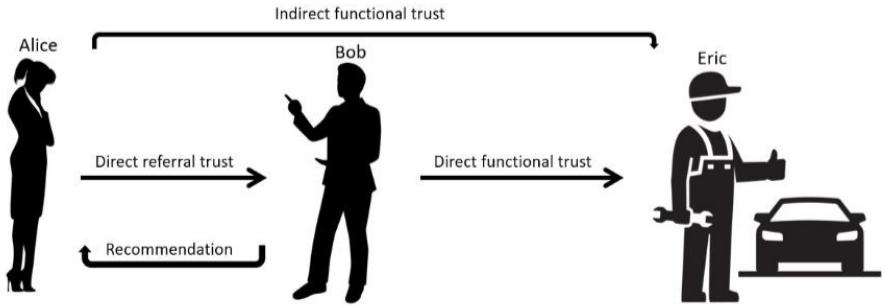


Fig. 4.1 Example of Functional Trust and Referral Trust.

In Fig. 4.1, assuming that, on several occasions Bob has tried to Alice to be informed about car maintenance issues, the Referral Trust of Alice in Bob for recommending a good car mechanic can be considered Direct. Assuming that Eric has on several occasions tried to Bob to be a good cat mechanic, Bob’s Functional Trust in Eric can also be considered Direct. Thanks to Bob’s recommendation, Alice will also trust Eric to be a good car mechanic. In this case the Functional Trust must be considered Indirect, because Alice did not directly observe or experience Eric’s skills in repairing the cars. The scope of the Trust is

nevertheless the same, “being a good car mechanic”. The concept of reputation is related to that of trustworthiness, but it is different. According to the Merriam-Webster dictionary the reputation is “The overall quality seen or judged by people in general” [89]. The trust is a personal and subjective concept based on different factors or evidence (received referrals, personal experience, etc.). Reputation is a collective measure of trust based on referrals or ratings from community members. In trustworthiness in order to avoid dependence and loops it is required that referrals be based on first hand experience only, and not on other referrals [41].

4.2 State of the Art: Models of Trust

Many models of Trustworthiness have been proposed in the literature. An interesting Method of Classification is presented in [83]. In this survey a techniques-based subdivision of the models is carried out. In particular they have classified the trustworthiness models into statistical and machine learning techniques, heuristics-based techniques, and behavior-based techniques. Moreover “Bayesian systems and belief models are the major examples of purely statistical techniques” [83]. Bayesian Systems are based on computing reputation scores. As is clearly explained in [44], Bayesian systems take as input the binary ratings (i.e. positive or negative, honest or dishonest) and calculate the reputation scores by statistical updating of beta probability density functions (PDF). The beta distribution is a family of continuous probability distributions defined on the interval [0,1] indexed by the two positive shape parameters α and β. In particular, combining the a priori (i.e. the previous) reputation score with the new rating [43, 65–67] we compute the a posteriori (i.e. the updated) reputation score. In general the reputation score can be represented in the form of the beta PDF parameter tuple (α, β), or in the form of the probability expectation value of the beta PDF. Here α and β represent respectively the quantity of positive and negative ratings. In Belief Models as the sum of probabilities over all possible outcomes not necessarily add up to 1, the remaining probability is interpreted as uncertainty. In [39, 40] the author proposes a new metric called opinion denoted by $\omega_x^A = (b,d,u,a)$ which expresses the relying party A's belief in the truth of statement x. Here b represents the belief, d the disbelief, u the uncertainty about a certain statement and a the relative atomicity, that represents the base rate probability in the

absence of evidence (in the absence of belief and disbelief). Here is presented also a mapping between the Bayesian approach and the Belief approach that allows to transfer an opinion from the evidence space to the opinion space. Moreover the author defines two operators for combining opinions (consensus) and weighting recommendations (discounting) opinions. These allows to evaluate trust based on recommendations. The consensus and the discounting operators form part of Subjective Logic [40, 42]. Some trustworthiness models are based on machine learning and typically use techniques such as Artificial Neural Networks (ANNs) and Hidden Markov Models (HMMs). In [84] an HMM is used for evaluating recommender trust and in [28] the author proposes a discrete HMM-based trust model. As explained in [83] since both statistical solutions and machine learning solutions are highly complex, researchers have moved towards heuristics-based solutions. These models have the goal to define practical, robust and easy trust management system. Two of the most famous heuristics-based solutions are explained in [46, 90]. Another category of trust models are the behavior-based. Here the trustworthiness is evaluated based on the communication behavior of members. In [2] the trust evaluation is based on conversation trust and propagation trust. The conversation trust calculates the duration and frequency of communications between a pair of users. Longer and more frequent communications translate into greater trust between the two users considered. The propagation trust considers the propagation of information. The greater the number of times a node propagates the information received from a certain source, the greater the trust that node has in that source. In Fuzzy Models the calculated trust values and reputation are represented as fuzzy concepts [26]. The membership functions describe the degree of trustworthiness with which an agent can be described. So the final interpretation of a fuzzy value like “very good” is left up to the user or agent.

The trust models, in addition to the type of technique on which they are based, can also be classified according to the type of technology in which they are applied. For example, in telecommunications networks, many models have been proposed in the context of P2P networks, within the social networks and in the IoT and SIoT field. In P2P distributed network EigenTrust [46] is one of the most famous reputation models. It assigns to each agent a unique global trust value, based on his history of uploads, that reflects the experiences of all peers in the network with that agent. EigenTrust computes the level of trust that a system places on a participant based on the normalized local trust vector of the participant and its eigenvector. In

this case the trust value itself has not a real meaning, indeed the semantics of ranking based trust values only states that a higher value is better. So Eigentrust allow to identify the entity with the highest trust value, but does not allow to state "how good" the best entity is. In this model is interesting the idea to ask for the opinions (what your friends think of another peer) of people who you trust and weight their opinions by your trust in them. Another famous model in P2P networks is PeerTrust [90]. Using parameters such as feedback, credibility, the number of transactions, the transaction context factor (importance of transaction) and the community context factor can calculate the trust value. It is a context-aware trust evaluation system able to find the dishonest feedbacks so that any participant, which was previously trusted but is currently giving malicious feedback, can also be identified. While transaction context factor addresses application dependent factors, community context factor is used to address some of the community-specific issues and vulnerabilities. The concept of credibility it's similar to our concept of Recommendation Trust that we will introduce in the following. In particular in PeerTrust a feedback from peers with higher credibility should be weighted more than those with lower credibility. A very interesting model for P2P distributed systems is proposed in [15]. Self-ORganizing Trust model (SORT) defines two context of trust, one for service and one for recommendation. The first measure the capabilities of peers in providing services (service trust) and the second measure the capabilities of peers in giving recommendations (recommendation trust). We find this separation also in PreferTrust [62] between direct trust vector and recommending trust vector. The parameters considered to evaluate interactions and recommendations are satisfaction, weight (importance of transaction), and fading effect (decaying factor). Recommenders trustworthiness and confidence about recommendation are considered when evaluating recommendations. In particular a recommendation contains the recommenders own experience, information from its acquaintances, and level of confidence in the recommendation. In SORT each peer develops its own local subjective view of trust about the peers with which interacted in the past. Here the local trust information is enough to make decisions and is not required the global trust information. This is coherent with the subjective nature of the concept of trustworthiness. So a peer can form a trust network in its proximity and can isolate malicious peers around itself as it develops trust relationships with good peers. No a priori information or a trusted peer is used to leverage trust establishment. Two very important concepts are the competence belief that represents

how well an acquaintance satisfied the needs of past interactions and the integrity belief that represents the level of confidence in predictability of future interactions. In service trust and recommendation trust is assigned an increasingly large weight to direct experience factor as the number of interactions/recommendations increases. In particular is used the following structure: $\alpha * DirectExperience + (1-\alpha) * IndirectExperience$. This is present in many works [10, 17, 26]. This weight α is also build in such a way to not consider all past service/recommendation history, but only the recent service/recommendation history. Finally, it is interesting how the reputation is used in the indirect experience factor. In addition to P2P networks and Social Networks, other scenarios in which researchers focused on creating new trust models are IoT (Internet of Things) and SIoT (Social Internet of Things) [10, 17–19, 71]. In [17] is presented a so-called honesty parameter $D_{ij}^{honesty}(t)$ which is obtained by comparing $D_{jq}^{honesty}(t-\Delta t)$ with $D_{iq}^{honesty}(t-\Delta t)$. Here the concept is that the only way for i to understand if j is a reliable recommender is to compare j's opinion about another node q with his opinion about q. We will resume the principle behind this parameter to build the recommendation satisfaction in the following. In [19] in addition to the fact that to calculate the trust value are used parameters such as direct feedback (satisfaction), the transaction weight factor (importance) and the decay factor (decaying factor), the interesting thing is the way in which the decay factor is calculated. In particular, it takes into consideration only the time elapsed from the interaction considered. Moreover another interesting thing is the presence of the energy status parameter. In [71] is provided a subjective trust model for Social Internet of Things Scenario. Here each node p_i computes the trustworthiness T_{ij}, i.e. the trustworthiness of node p_j seen by node p_i, on the basis of its own experience and on the opinion of the K_{ij} friends in common. In reality, even though, p_i and p_j are not friends (i.e. are not adjacent in social graph) the trustworthiness can be calculated by word of mouth through a chain of friendships. The trust value is determined through the following parameters: feedback system (satisfaction of transaction), total number of transactions (to detect if two nodes have an abnormally high number of transactions), credibility (where 1 represents full credibility for the node), transaction factor (importance of transaction), relationship factor (indicates the type of relation that connects two nodes), centrality (if a node has many relationships or is involved in many transactions, it is expected to assume a central role in the network) and computation capabilities (intelligence of device). The

trustworthiness of a node is computed, by the nodes in the network that interacted with it, through evaluation of its behaviour performed. Moreover the reputation reflects the degree of trust that other nodes in the social network have on the given node on the basis of their past direct (direct interactions) or indirect (through intermediate nodes) experiences. Also here like in SORT [15] the trust is calculated at local level. Interesting the concept of opinion long and recent that we will use to compute the integrity in our approach. Moreover here we find again the concept of credibility to weight the opinion of a recommender. A feedback from peers with higher credibility should be weighted more than those with lower credibility. We will retake this concept in service reputation using the recommendation trust instead of credibility.

4.3 Proposed Trust Model

In the field of Trustworthiness it's important to separate between trust in the ability to provide services (Functional Trust) and trust in the ability to provide recommendations (Referral Trust) [42]. The difference between Functional Trust and Referral Trust has already been explained in the past in some papers [15, 41]. In addition, Functional Trust and Referral Trust can in turn be divided into Direct and Indirect Functional Trust and Direct and Indirect Referral Trust. According to [15] in this paper we refer to the Functional Trust with the term Service Trust and to the Referral Trust with the term Recommendation Trust for a more immediate understanding of the concept that they imply. In order to model the Service Trust, we must first introduce some factors. To describe these terms we will use some of the concepts of Graph Theory. With i and j we represent two nodes of the graph and with l the interaction that occurs between them. sf_{ij}^{l} represents the Satisfaction of i's l^{th} interaction with j. This factor allows a node i to provide an evaluation of the service it has received by the provider j. $s\omega_{ij}^{l}$ is the Importance of i's l^{th} interaction with j and indicates the relevance of transaction l between node i and node j. It is used to discriminate important transactions from irrelevant ones. $s\delta_{ij}^{l}$ represents the Decaying Factor of i's l^{th} interaction with j. With sh_{ij} we mean the Size of i's Service (Interaction) History with j that is the total number of interactions occurred between i and j. F_{ij} is the Relationship Factor and indicates the type of relation that connects i with j. It represents a unique characteristic of the SIoT. We briefly

describe the types of existing social relationships. The Owner Object Relationships (OORs) are established between two objects that belong to the same owner. In this kind of relation it's very unlikely to find a malicious node. The Co-Location Object Relationships (C-LORs) connect domestic objects, the Co-Working Object Relationships (C-WORs) link objects of the same workplace. The Social Object Relationships (SORs) are relationships established between objects that are encountered occasionally. The Parental Object Relationships (PORs) are created between objects of the same model. In Tab. 4.1 we show the weight values associated with Social relationships.

Table 4.1 Values of the weights of social relationships.

Social Relationship	**Weight**
Ownership object relationship (OOR)	0.9
Co-location object relationship (C-LOR)	0.8
Co-work object relationship (C-WOR)	0.8
Social object relationship (SOR)	0.6
Parental object relationship (POR)	0.5
No relationship	0.1

$R_{ij} = \frac{|K_{ij}|}{|N_i|-1}$ is the Centrality of j in the "life" of i, where $|K_{ij}|$ represents the Common friends between i and j and $|N_i|$ is the Neighborhoods of node i [70, 71]. With I_j we mean the Intelligence of j that represents the computational capabilities of an object. Let's look at the Decaying Factor in more detail:

$$s\delta_{ij}^{l} = \begin{cases} \mu\dfrac{l}{sh_{ij}} + \nu, & \text{for } |t-t^{l}| \leq e \\ \mu\dfrac{l}{sh_{ij}} + \nu\dfrac{1}{ln(|t-t^{l}|)}, & \text{otherwise} \end{cases} \tag{4.1}$$

where t is the actual time and t^l is the occurrence time (generation time) of this interaction. The first contribute is taken from [15], the second from [19]. The idea is that the first contribution of the Decaying Factor takes into account the number of interactions occurred after the considered one (Current validity of the interaction) and the second contribution the time elapsed from the interaction considered (Recency of the interaction). To normalize the second contribution, if $|t-t^l| \leq e$ (Nepero's number) we assign $|t-t^l| = e$ so that $\frac{1}{ln(|t-t^l|)} = 1$.

Considering both contributions, we are able to evaluate the Decaying Factor more precisely and in all possible scenarios. A more detailed analysis on this parameter will be carried out later.

At this point we are going to introduce two contributions the Competence Belief and the Integrity Belief. In particular, we now present the Service Competence Belief and Service Integrity Belief, leaving the Recommendation Competence Belief and the Recommendation Integrity Belief for the second part of the discussion.

The Service Competence Belief measures how well an acquaintance satisfied the needs of past interactions.

$$scb_{ij} = \frac{\sum_{l=1}^{sh_{ij}} (sf_{ij}^{l} s\omega_{ij}^{l} s\delta_{ij}^{l})}{\sum_{l=1}^{sh_{ij}} (s\omega_{ij}^{l} s\delta_{ij}^{l})} \tag{4.2}$$

The previous formula is taken from [15], but it is also used in other paper like [71] with some differences.

The Service Integrity Belief is the level of confidence in predictability of future interactions.

$$sib_{ij} = \sqrt{\frac{1}{sh_{ij}} \sum_{1}^{sh_{ij}} (SO_{ij}^{rec} - SO_{ij}^{lon})^2} \tag{4.3}$$

Small values of Integrity translate into a more predictable behavior of j in future interactions. The idea is to consider not only the degree to which a node has satisfied past interactions, but also the deviation in the degree of satisfaction of the recent interactions with respect to the remote ones. The concept of predicting the degree of satisfaction of future interactions, based on past interactions, is found in literature in Bayesian Systems and Belief Theory models (including Subjective Logic). These models use the Probability Density Function (PDF) and the Expected Values of PDFs. A less complex way to try to calculate predictability of future interactions is by using the standard deviation and considering the Service Opinion Long as the mean value. This is showed in Equation 4.3.

The two terms we use to calculate Service Integrity Belief are Service Opinion Long and Service Opinion Recent. They can be expressed as:

$$SO_{ij}^{lon} = \frac{\sum_{l=min[L_{ij}^{lon}]}^{max[L_{ij}^{lon}]}(sf_{ij}^{l} s\omega_{ij}^{l} s\delta_{ij}^{l})}{\sum_{l=min[L_{ij}^{lon}]}^{max[L_{ij}^{lon}]}(s\omega_{ij}^{l} s\delta_{ij}^{l})} \tag{4.4}$$

$$SO_{ij}^{rec} = \frac{\sum_{l=min[L_{ij}^{rec}]}^{max[L_{ij}^{rec}]}(sf_{ij}^{l} s\omega_{ij}^{l} s\delta_{ij}^{l})}{\sum_{l=min[L_{ij}^{rec}]}^{max[L_{ij}^{rec}]}(s\omega_{ij}^{l} s\delta_{ij}^{l})} \tag{4.5}$$

The Equations 4.4,4.5 are also used in [71] for O_{ij}^{lon} and O_{ij}^{rec} without considering the Decaying Factor. For a more precise calculation we inserted $s\delta_{ij}^{l}$ in the formulas. They represent the Long-term and the Short-term Service Opinion of i about j and they are based on the satisfaction of i with respect to the services provided by j. L_{ij}^{lon} represents the long-term opinion temporal window for the pair i, j and L_{ij}^{rec} the short-term opinion temporal window for the pair i, j, with $Ms^{lon} > Ms^{rec}$. Ms^{lon} represents the fixed size of L_{ij}^{lon} set by the user. Ms^{rec} represents the fixed size of L_{ij}^{rec} set by the user.

The temporal windows for SO^{lon} and SO^{rec} are chosen in the following way:

$$L_{ij}^{lon} = \begin{cases} [(sh_{ij} - Ms^{lon} + 1), sh_{ij}], & \text{for } sh_{ij} \geq Ms^{lon} \\ [1, sh_{ij}], & \text{for } sh_{ij} < Ms^{lon} \end{cases} \tag{4.6}$$

$$L_{ij}^{rec} = \begin{cases} [(sh_{ij} - Ms^{rec} + 1), sh_{ij}], & \text{for } sh_{ij} \geq Ms^{rec} \\ [1, sh_{ij}], & \text{for } sh_{ij} < Ms^{rec} \end{cases} \tag{4.7}$$

Similarly, when we are going to deal with the part of the Recommendation instead of the Service Opinion Long and the Service Opinion Recent we will talk about the Recommendation Opinion Long and the Recommendation Opinion Recent.

Now we are ready to calculate the Service Trust:

$$\begin{aligned} st_{ij} = &\left(\frac{log(sh_{ij}+1)}{1+log(sh_{ij}+1)}\right)(\beta_1 scb_{ij} - \beta_2 sib_{ij}) + \\ &+\left(\frac{1}{1+log(sh_{ij}+1)}\right)(\gamma sr_{ij} + \varepsilon F_{ij} + \zeta R_{ij} + \theta(1-I_j)) \end{aligned} \tag{4.8}$$

In particular we have used the following formula:
$\alpha * DirectExperience + \beta * IndirectExperience$, where α grows and β decreases with the number of interactions. The values of α and β are taken from [71].

To compute the Service Trust, as the number of interactions increases, we will assign an increasing weight to the contribution of Direct Experience. This structure is used in many Trustworthiness models [10, 15, 17, 26, 71]. In the Direct Experience contribution, the first term (Service Competence Belief) corresponds to the Direct Functional Trust. In the contribution of Indirect Experience, the first term (sr_{ij} Service Reputation) corresponds to the Indirect Functional Trust. In the contribution of Indirect Experience we also consider parameters such as Relationship Factor, Centrality and Intelligence that allow us to calculate the Service Trust even in the absence of direct interactions.

The (Local) Service Reputation (seen by i) can be expressed as:

$$sr_{ij} = \frac{\sum_{k=1}^{|K_{ij}|^*} rt_{ik}(st_{kj})}{\sum_{k=1}^{|K_{ij}|^*} rt_{ik}} \tag{4.9}$$

where $|K_{ij}|^*$ is the set of nodes $\in K_{ij}$ (k at the same time neighbors both i and j) who requested a service for j and j supplied it. This formula is taken from [71], but instead of Credibility we use the recommendation trust rt to weight the opinion of k about j. In Service Reputation, therefore, let's weigh the Service Trust suggested to us by the neighbor with the Recommendation Trust that the neighbor has to our eyes. The idea of considering Recommendation Trust as a weight, instead of the Direct Trust (Service Trust), of Credibility or of Similarity, lead to greater precision in the calculation of Reputation. This allows us to consider the case in which a node, even if good in Providing Service, may not be as good in Providing Recommendation. On the contrary, the other parameters tend to mix the Service Trust with the Recommendation Trust.

A dual treatment must be done for the Recommendation Trust.

With rf_{ik}^z we will indicate the Satisfaction of i's z^{th} recommendation with k, $r\omega_{ik}^z$ the Importance of i's z^{th} recommendation with k and with $r\delta_{ik}^z$ the Decaying Factor of i's z^{th} recommendation with k that can be expressed as:

$$r\delta_{ik}^{z} = \begin{cases} \mu\frac{z}{rh_{ik}} + \nu, & \text{for } |t - t^{z}| \leq e \\ \mu\frac{z}{rh_{ik}} + \nu\frac{1}{ln(|t - t^{z}|)}, & \text{otherwise} \end{cases} \tag{4.10}$$

where t is the actual time and t^{z} is the occurrence time of this recommendation. rh_{ik} represents the Size of i's Recommendation History with k. Also here to normalize the second contribution, if $|t - t^{z}| \leq e$ (Nepero's number) we assign $|t - t^{z}| = e$ so that $\frac{1}{ln(|t-t^{z}|)} = 1$.

Analogously RO_{ik}^{lon} is the Recommendation Opinion Long:

$$RO_{ik}^{lon} = \frac{\sum_{z=min[L_{ik}^{lon}]}^{max[L_{ik}^{lon}]} (rf_{ik}^{z} r\omega_{ik}^{z} r\delta_{ik}^{z})}{\sum_{z=min[L_{ik}^{lon}]}^{max[L_{ik}^{lon}]} (r\omega_{ik}^{z} r\delta_{ik}^{z})} \tag{4.11}$$

RO_{ik}^{rec} the Recommendation Opinion Recent that can be expressed as:

$$RO_{ik}^{rec} = \frac{\sum_{z=min[L_{ik}^{rec}]}^{max[L_{ik}^{rec}]} (rf_{ik}^{z} r\omega_{ik}^{z} r\delta_{ik}^{z})}{\sum_{z=min[L_{ik}^{rec}]}^{max[L_{ik}^{rec}]} (r\omega_{ik}^{z} r\delta_{ik}^{z})} \tag{4.12}$$

Mr^{lon} represents the fixed size of L_{ik}^{lon} set by the user. Mr^{rec} represents the fixed size of L_{ik}^{rec} set by the user.

The temporal windows for RO^{lon} and RO^{rec} are chosen in the following way:

$$L_{ik}^{lon} = \begin{cases} [(rh_{ik} - Mr^{lon} + 1), rh_{ik}], & \text{for } rh_{ik} \geq Mr^{lon} \\ [1, rh_{ik}], & \text{for } rh_{ik} < Mr^{lon} \end{cases} \tag{4.13}$$

$$L_{ik}^{rec} = \begin{cases} [(rh_{ik} - Mr^{rec} + 1), rh_{ik}], & \text{for } rh_{ik} \geq Mr^{rec} \\ [1, rh_{ik}], & \text{for } rh_{ik} < Mr^{rec} \end{cases} \tag{4.14}$$

At this point we define the Recommendation Competence Belief as how well an acquaintance satisfied the needs of past recommendations

$$rcb_{ik} = \frac{\sum_{z=1}^{rh_{ik}} (rf_{ik}^{z} r\omega_{ik}^{z} r\delta_{ik}^{z})}{\sum_{z=1}^{rh_{ik}} (r\omega_{ik}^{z} r\delta_{ik}^{z})} \tag{4.15}$$

and the Recommendation Integrity Belief as the level of confidence in predictability of future recommendations (deviation of recent-term recommendations from long-term)

$$rib_{ik} = \sqrt{\frac{1}{rh_{ik}} \sum_{1}^{rh_{ik}} (RO_{ik}^{rec} - RO_{ik}^{lon})^2} \tag{4.16}$$

Now we have all the elements to define the Recommendation Trust:

$$\begin{aligned} rt_{ik} = & \left(\frac{log(rh_{ik}+1)}{1+log(rh_{ik}+1)} \right) (\beta_1 rcb_{ik} - \beta_2 rib_{ik}) + \\ & + \left(\frac{1}{1+log(rh_{ik}+1)} \right) [\varepsilon F_{ik} + \zeta R_{ik} + \theta(1-I_k)] \end{aligned} \tag{4.17}$$

In this case we can not use the Recommendation Reputation. Indeed 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. As a consequence, an individual should only give subjective trust referral when it is based on first hand evidence or when second hand input has been removed from its derivation base [45].

The Recommendation Satisfaction is calculated as follows:

$$rf_{ik} = 1 - \frac{\sum_{j=1}^{N_j} |scb_{ij} - rcb_{ik} scb_{kj}|}{|N_j|} \tag{4.18}$$

where N_j is the set of j to which both i and k have requested a service and j has provided it.

The idea is that the only way to understand if a recommendation is reliable is to compare it with the Direct Experience that the node itself obtains a posteriori by carrying out a direct interaction. Basically the node goes to compare the judgment of another node with its own direct experience. The formula of rf_{ik} is designed based on honesty parameter in [17].

The Recommendation Importance is computed as follows:

$$r\omega_{ik} = \nu \frac{n_{kj}}{n_{kjtot}} + \rho \frac{\sum_{j=1}^{n_{kj}} \frac{sh_{kjlast}}{sh_{kjmax}}}{n_{kj}} \tag{4.19}$$

where n_{kj} is the Number of different Service Provider that provided a service to k and n_{kjtot} the Total number of different Service Provider that can provide a service to k. sh_{kjlast} is the Current size of each Recommender k's Service (Interaction) History Window with j,

which contains at most last $|sh_{kjmax}|$ interactions and then $0 \leq |sh_{kjlast}| \leq |sh_{kjmax}|$, sh_{kjmax} is the Max size of k's Service (Interaction) History Window with j.

The idea is that a recommendation will be as much important as higher the number of providers (n_{kj}) with which the recommender will be compared (higher the number of providers that will have provided a service, therefore, greater will be the ability to evaluate a service based on the acquired experience) and how much higher will be the number of interactions occurred (sh_{kj}) between the recommender (k) and each provider (j).

The first contribution considers the Number of Providers that have already provided at least one service to the recommender, compared to all the possible providers that can theoretically provide at least one service.

The second contribution considers the Average of the Current size of the Service (Interaction) History Window (sh_{kjlast}), compared to every j that provided at least one service to k.

We conclude by showing in Tab. 4.2 the weights of the parameters used to carry out the simulations.

Table 4.2 Values of the weights of parameters.

Parameter	Description	Weight
β_1	Weight of the Competence	1
β_2	Weight of the Integrity	0.5
γ	Weight of the Reputation	0.5
ε	Weight of the Relationship Factor	0.175
ζ	Weight of the Centrality	0.175
θ	Weight of the Intelligence	0.15

4.4 Performance Evaluation

4.4.1 Simulation Setup

The trustworthiness model presented in this paper was implemented in Matlab (MAtrix LABoratory). Thanks to this environment for numerical calculation, we will evaluate, through different simulations, the performance of our Trust model.

To conduct all our analyzes, a mobility model called Small World In Motion (SWIM) was employed. It has allowed us to generate realistic traces of users' movements, simulating their physical positions and movements. In particular, we assumed that each person, distinguished by an ID, possessed two devices. The nodes generated in SWIM can be fixed or mobile. The difference is that while the fixed nodes remain stationary in one position for the duration of the simulation, the mobile nodes move within the simulation area. This rectangular area has the dimensions of one kilometer in width by two in length. Inside there are six points of interest and an Home point for each node. The Home Point simulates the home of each user. At the end of the day, each user will return to his home. In the simulations we have assumed a number of mobile nodes equal to one hundred, and of fixed ones equal to twenty, which determines a set of devices altogether equal to two hundred and forty. We simulated the movement of this crowd with SWIM for a period of 72 hours. Within SWIM, when a physical contact occurs between a pair of nodes, a bidirectional arc is established between them. More in detail, this adjacency arc will be established if the distance between one node and another is less than 5 meters, and if the total duration of the contact will be less than or equal to five minutes. Transactions between two nodes are possible if and only if there is a physical contact between that pair of nodes. Each device is characterized by numerical parameters that represent its model and its intelligence. The social object relationships will be created according to the contacts that will take place between the owners of the devices.

In particular, referring to the rules set out in [6–8], we will have the following relationships established:

- OOR if the devices have the same owner;
- POR if the devices are of the same model;
- C-LOR if both devices meet in the respective home-points (the devices have the same home-point);
- C-WOR only if the devices meet in a certain set of locations (offices, factories, laboratories, etc.) and these meetings last longer than Ts, with Ts = 1 hour on 36 hours of simulation, 300 seconds on 4 hours of simulation, and if both devices meet in places other than their respective home-points, e.g. office, etc.;

- SOR if the owners of the devices meet at least Nc times, if successive meetings occur at intervals of longer duration than Ti and if each of the meetings lasts longer than Tc. $Nc = 2$, $Ti = 8$ hours on 36 hours of simulation, 3600 seconds on 4 hours of simulation, and $Tc = 30$ minutes on 36 hours of simulation, 150 seconds on 4 hours of simulation.

It is necessary to underline that a transaction between a pair of nodes is possible if and only if there is a social link between that same couple. On the contrary, the existence of a social bond does not necessarily imply that the two nodes must necessarily interact. The devices themselves will choose whether to have an interaction with an "acquaintance". For all the simulations we will assume that all physical contacts and the resulting social relationships have occurred in an instant before the first transaction. Furthermore, social relations will be maintained throughout the duration of the simulation.

4.4.2 Simulation Results

Comparison between our Model and the Subjective Model in Recommendation-Based Attacks

In this first phase of the Simulation Campaign the objective was to compare the performance of our model to those of the Subjective Model. In particular we have made a comparison with different percentages of attack of the single node and with different percentages of malicious nodes. Before seeing the results it is necessary to clarify the type of maliciousness considered in these simulations. In particular, a malicious node is a node that gives false recommendations for his personal benefit. Another type of maliciousness is that on the provision of the service, but we will consider it elsewhere. In reality, a node that provides bad recommendations does not necessarily be malicious, it could be simply a bad recommender. Likewise, a node that provides bad service will not necessarily be a mischievous service provider. In general, a malicious node is a node that behaves well and only on certain transactions reveals its true nature. On the contrary, a bad recommender always behaves badly due to incapacity. Later we will see how to try to counter the behavior of a malicious using the weight of Integrity or using the parameter of Importance. Let's assume we are in the Perfect Service Case, which is the case where the service satisfaction sf is always equal to 1.

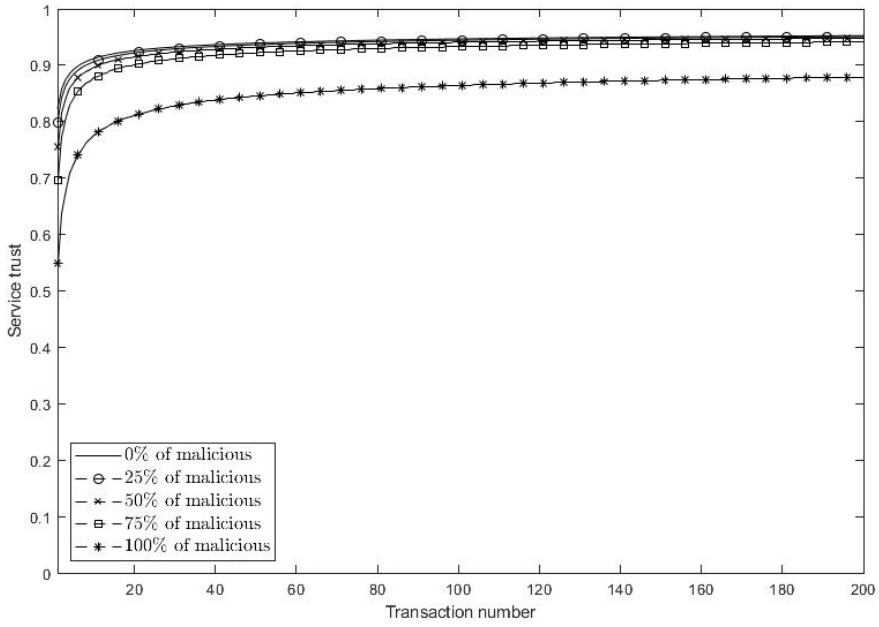


Fig. 4.2 Performance of our Model when it changes number of malicious.

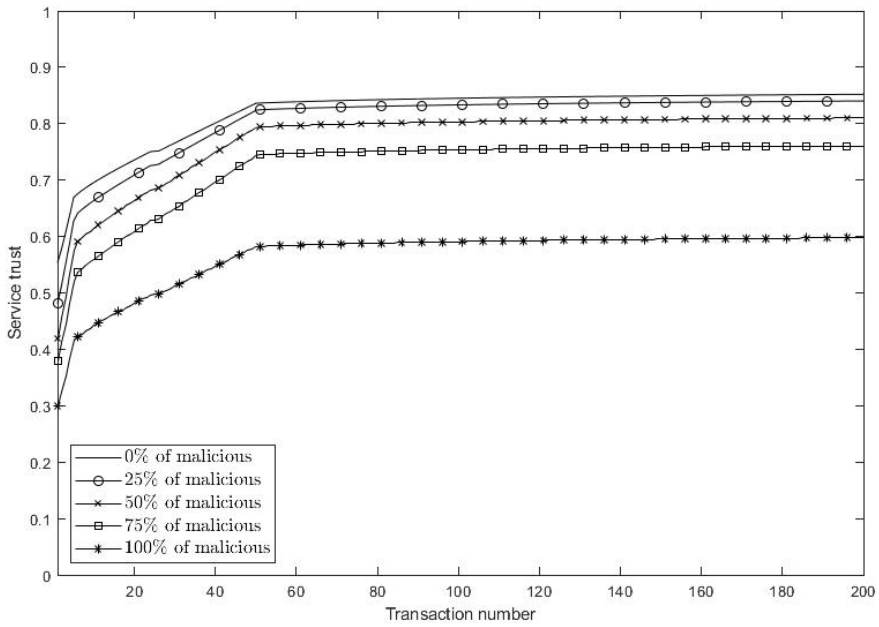


Fig. 4.3 Performance of Subjective Model when it changes number of malicious.

In Fig. 4.2 there is a comparison with different percentages of malicious (number of malicious), keeping the percentage of attack fixed at 100%.

In both models we note that the higher is the number of malicious the lower is the Service Trust. The performance in assigning the correct value of Service Trust, however, in our case are better than those of the Subjective Model, because it is closer to the value of Groundtruth ($sf = 1$). Even in the worst case (the one with 100% malicious), our model behaves better than the Subjective in the best case (the one with 0% malicious). This difference in performance is due to various reasons. In the Subjective Model there is no a clear separation between the Service Trust and the Recommendation Trust. We have found that considering separately the ability to provide the service and the ability to provide recommendations makes it possible to calculate a more correct Service Trust and that the worst performance of the Subjective is mainly due to the fact that it tends to mix them. Moreover, in our Model since Centrality, Intelligence and Relationship Factor are contained in the Indirect Contribution, after many interactions they become always less relevant. On the contrary, the Direct Experience assumes an increasing weight with the passing of transactions. In the case of the Subjective Model, Centrality and Intelligence are always present with the same weight in the calculation of the Trustworthiness formula. As a result, Service Trust's dependence on Centrality and Intelligence is much greater. Also for these reasons while in our Model the distance between the curves is uniform, in the Subjective it is logarithmic. Finally, our Model is much faster in converging to the final value of Service Trust than the Subjective Model.

In Fig. 4.4, there is a comparison to the variation of the attack percentage of the single malicious, considering a scenario with the 100% of malicious nodes. Also in this case we assume to be in the Perfect Service Case, that is the case in which the Service Satisfaction is always equal to 1.

Also in this case in both models we note that as the attack percentage increases the Service Trust decreases. The performance in assigning a Service Trust value that is as close as possible to that of Groundtruth ($sf = 1$) in our case are better than those of the Subjective. Even in the worst case (the one with 100% malicious), our model behaves better than the Subjective in the best case (the one with 0% malicious). We have already seen the reasons for this difference in performance previously. Also in this case our Model is much faster in converging to the final value of Service Trust compared to the Subjective Model. The

fluctuations are due in our Model to the Service Reputation trend, in the Subjective Model to the Opinion Indirect trend. By making the average of the curve with fluctuations in the case of attack percentage we obtain exactly the curve trend with the corresponding percentage in the case of malicious number. For example mediating the curve with 25% of attack (in the graph when the Attack Percentage varies) we obtain the trend of the curve with 25% of malicious (in the graph by varying the Percentage of the Malicious Number). The Service Trust fluctuations are related to the difficulty in understanding if the node is actually bad or a good provider. Obviously, if we have an attack percentage of 100% (bad provider) or of 0% (good provider) it is much easier to understand the true nature of the node, compared to the case with a different attack percentage. Consequently in these two cases there will be no fluctuations.

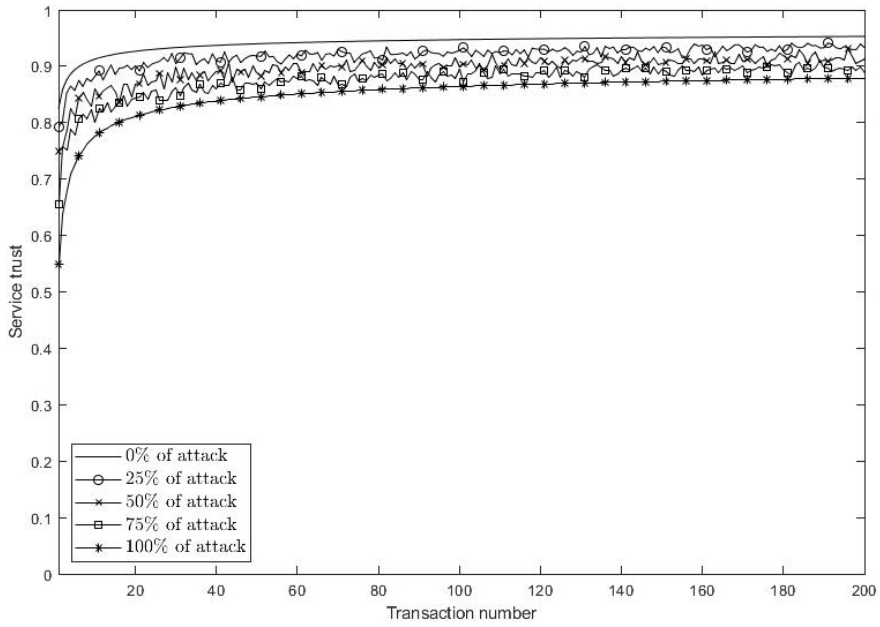


Fig. 4.4 Performance of our Model when it changes attack percentage of malicious.

Tuning of the constants in the Decaying Factor

Another very important phase was the Tuning phase. In particular we started from the Tuning of the constants in the Decaying Factor. The variables μ and ν allow us to weigh the two

contributions of the Decaying Factor and must be set according to the scenario in which we find.

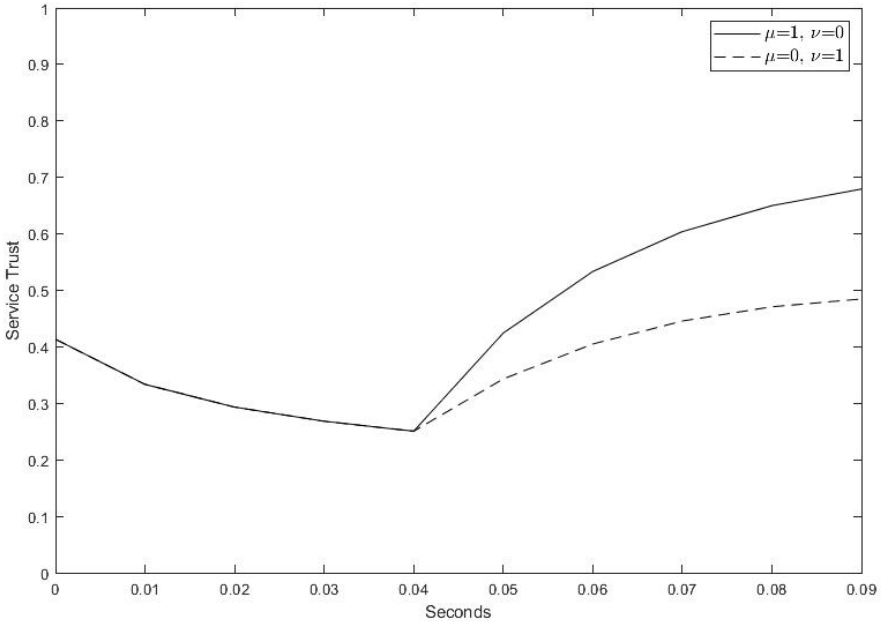


Fig. 4.5 Short Time Interval Case.

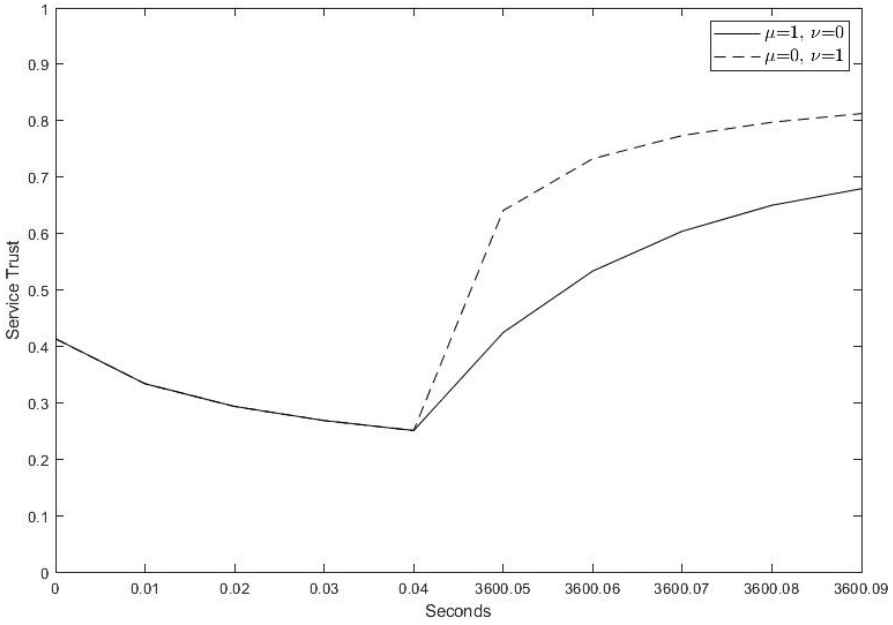


Fig. 4.6 Long Time Interval Case.

The First Contribution considers the number of interactions occurred after the one taken into consideration and we can define it as the Current Validity of the interaction, the Second Contribution considers the time spent from the interaction considered and we can define it as the Recency of interaction. In particular, the First Contribution is important in scenarios in which the interactions occur in a short time interval. In fact, even if an interaction took place recently, since many interactions occurred after it, it may not be consistent with the current situation. On the other hand, the Second Contribution is important in scenarios in which interactions occur in a long time interval. In fact, even if an interaction is the last occurred between that specific pair of nodes, it may not be more reliable in case it happened a long time ago. Below we see the Short Time Interval and the Long Time Interval Case.

In both cases, we simulated a scenario in which the first five transactions had sf equal to 0 and the last five transactions had sf equal to 1. The goal was to understand which contribution would allow us to better evaluate the latest interactions in the case of a short time interval and the most recent interactions in the case of a wide time interval. In the Figure 4.5 we see how the First Contribution allows us to assign greater importance to the latest interactions and in the Figure 4.6 as the Second Contribution allows us to assign greater importance to the most recent interactions. The Decaying Factor in [15] consists only of the First Contribution. This means that if an interaction is the last one between that specific pair of nodes and it happened a long time ago, the model is not able to understand that it could not be more reliable. This is because it considers only the Current Validity of the interaction and does not consider the Recency of the interaction.

Comparison between the Service Integrity formulas

We focus on the calculation of the service integrity belief sib_{ij} measured by node i by referring to the service received by provider j (see Eq. 4.3). Similarly to [15], we follow an approach based on the standard deviation since it is simpler than the Bayesian and Belief models and allows greater control. In Fig. 4.7, we compare our approach in case service integrity belief is calculated according to Eq. 4.3 or with the SORT integrity formula proposed in [15]. We analyze the ideal case in which node j always provides an excellent service (i.e., sf is always equal to 1 and corresponds to the Ground truth). In such a condition, since the deviation in the degree of satisfaction of recent interactions with respect to the remote ones is null, node

i expects that *j* exhibits the same behavior shown in the past. By looking at Fig. 4.7, we can appreciate that, by using SORT, we tend to overestimate the service integrity belief and underestimate the service trust. Differently, the adoption of our model allows to obtain the expected service integrity belief (the curve is always equal to 0) and a service trust very close to the Ground truth. Similar conclusions can be drawn also in non-ideal scenarios (i.e., when node *j* does not always provide satisfactory services).

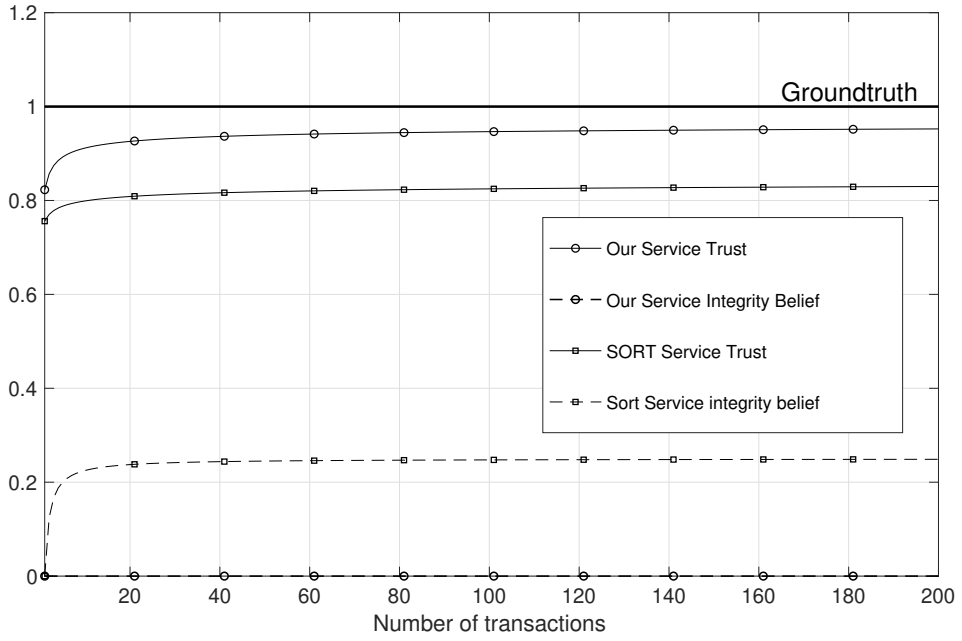


Fig. 4.7 Performance comparison obtained by our model with different service integrity belief formulas, in a scenario with sf (service satisfaction) always equal to 1.

Tuning of the constants in the Direct Contribution and Comparison between our Model and the Subjective Model in Service-Based Attacks

Let's move on to the Tuning of the constants in the Direct Contribution, i.e. the Competence and Integrity weights. Since Integrity is the "Error in Predicting the Future Behavior of the Provider", the choice of how much weigh the constant of the Integrity must be made in function of how much it wants to punish the Provider. In most simulations we set the values of the variables according to the empirical evidence presented in [15]. We resume the concept of maliciousness on the provision of the service and suppose that a provider behaves

well for 45 times and badly for 5 times in Fig. 4.8. In this case we want to make sure that the Trust Model is as reactive as possible to malicious transactions. We want to low as more as possible the Service Trust value in correspondence to the bad service received. Consider, for example, the case of important transactions such as banking.

A first way to punish the malicious, by obtaining a great decrease in the Service Trust, is to assign the values of Importance in a prudent manner. For example, we can assign to transactions with sf equal to 1 an Importance of 0.1 and to transactions with sf equal to 0 an Importance of 1. In this way the attitude of the malicious node is strongly punished, despite the good behavior in the first 45 transactions.

Suppose we found in a Time Critical application, where the delay Δt between the request time and the execution time of a task must be a fixed value. The higher is the difference from the prefixed delay value, the lower the satisfaction of the transaction. In this case we will assign to transactions with Δt equal to the prefixed value ($sf = 1$) an importance of 0.1 and to transactions with Δt different from the prefixed value ($sf = 0$) an Importance of 1. In this way we will punish a lot transactions that do not respect the pre-established constraint, despite the correct behavior shown up to that moment.

Also by correctly calibrating the Competence and Integrity weights we obtain the desired decrease of Service Trust. We see that also assigning a great weight to the constant β_2 which weighs the Integrity, we can punish very much the Service Trust of the node that provided a bad service. Obviously if we exceed with the weight of the Integrity we will get that the first bad transaction the Service Trust will drop so much that it remains null for the rest of the simulation. This obviously depends on the application and how much we want to punish a possible malicious. In this way we reach a closer value to Groundtruth, compared to the previous cases. We have a greater decrease than before in correspondence with the series of 5 $sf = 0$.

Taking up the example of Time Critical application, as long as the transactions respect the fixed value of Δt we will keep a low β_2 value, as soon as a transaction deviates from this constraint we will assign a very high value to β_2, in this way we will obtain a great lowering in the value of st.

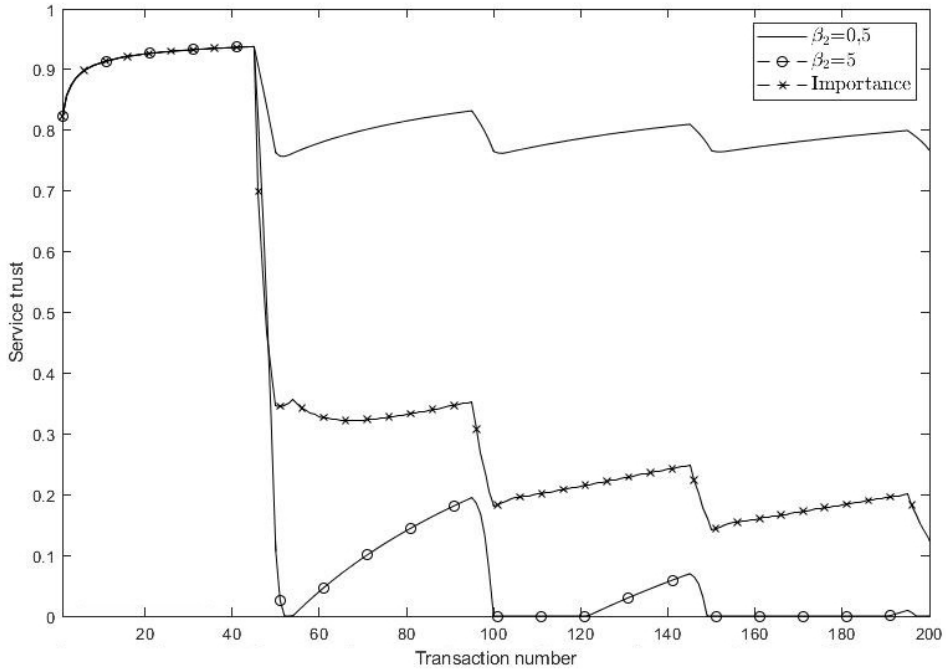


Fig. 4.8 Performance of our Model in Service-Based Attack and enhanced version using Importance and β_2 in case with 45 sf=1 and 5 sf=0.

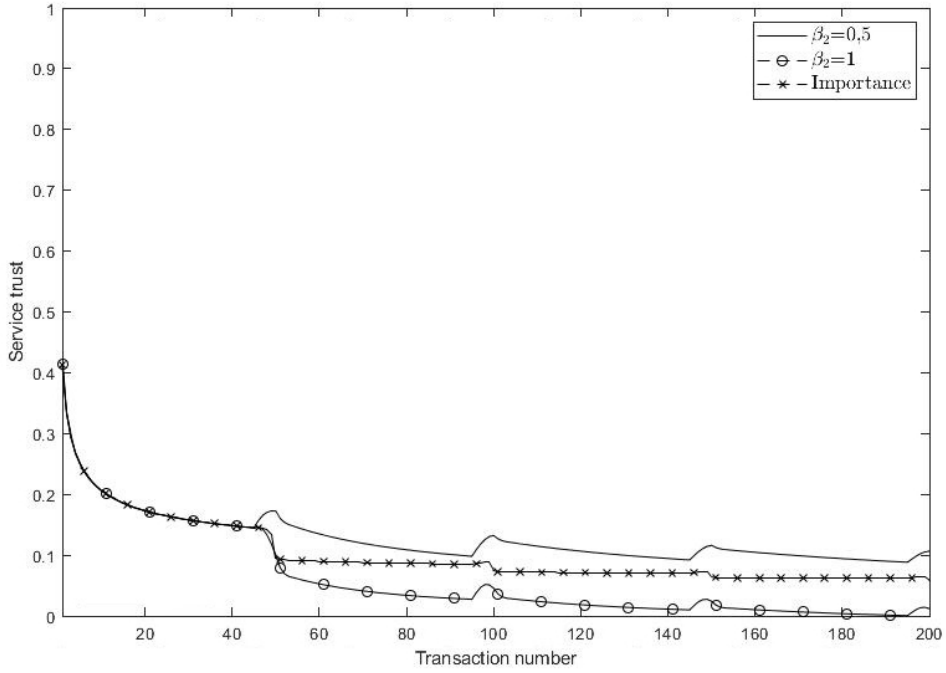


Fig. 4.9 Performance of our Model in Service-Based Attack and enhanced version using Importance and β_2 in case with 45 sf=0 and 5 sf=1.

Even in the opposite case where the provider behaves badly for 45 times and well for 5 times by correctly calibrating the Competence and Integrity weights we obtain the desired decrease of Service Trust in Fig. 4.9.

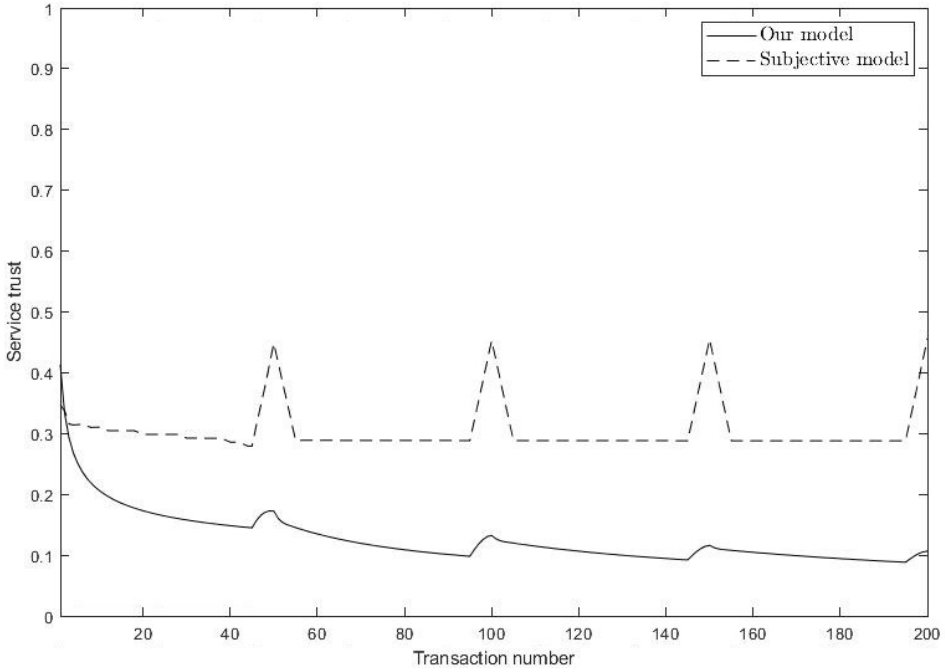


Fig. 4.10 Comparison between our Model and the Subjective Model in the case with 45 sf=0 and 5 sf=1.

Comparing our Model and the Subjective in all cases, we see that ours is closer to the Groundtruth value. In Fig. 4.10 we see for example the case with 45 sf=0 and 5 sf=1. Ours takes into consideration the past history progressively lowering despite the fact that the behavior is always alternated in the same way (so we will see that in correspondence with the series of 5 sf=1 the value of st assigned will always be lower). The Subjective Model does not take into consideration all the past history, but only the last L_{long}, L_{rec} transactions. Furthermore, ours converges much faster to the value of Groundtruth.

Cold Start Problem

The Cold Start Problem consists of the fact that especially in the early stages we do not have information about the reliability of the service provider. In fact, since the requester

has never interacted directly with the provider, he is not able to judge it. Furthermore, many interactions will be needed before the requester can understand whether the provider is actually a benign node or a malicious node. Not having directly interacted directly the Direct Contribution will be null and will remain null until the moment in which there will be a first direct interaction between requester and provider. A way to assess the provider's maliciousness even in the absence of previous transactions and, consequently, to reduce the Cold Start Problem is to include in the calculation of the Service Trust not only the Direct Contribution, but also the Indirect Contribution, which considers structural and social properties, parameters independent of transactions between nodes. For more details about this part, we refer the reader to the work done in [71].

Chapter 5

A real dataset for SIoT

5.1 Introduction

Today, the availability of IoT datasets is one of the challenges that researchers face when testing their models and algorithms. Although large companies have already developed their own IoT platforms, such as Amazon or Intel, they do not want to share their datasets in the form of open data [58]. Our objective is to obtain realistic datasets containing the following information: co-locations (meetings) between devices, device models, user Home-Points, user Interests Profiles, friendships between users, social objects relationships between devices. To the best of our knowledge it is the first work that allows to obtain all this information based only on real data (not simulated).

5.2 Existing IoT and SIoT datasets

Among the IoT datasets available online we mention the following projects in Smart City scenarios. Sacramento [77], which solved traffic congestion problems with new solutions for parking-seeking cars. Louisville [55] that recently launched an open data portal as a smart approach to local governance. Amsterdam [23] that provides data on several areas of interest, such as mobility and the environment. Unfortunately, all of these projects only deal with a specific sector and do not consider the heterogeneity of an IoT system. A very interesting procedure is described in [58]. Also in this case it is considered a Smart City Scenario.

In fact from the SmartSantader [79] project, information about objects and locations that have been used for public devices was extracted. The devices can be both static and mobile and have been classified according to the data models proposed in FIWARE Data Models [29]. Through Small World In Motion (SWIM) [50] they simulated the presence of a sample of 4000 users that have been used for private devices. According to the ownership report of the Global Web Index in the 2017 [37] calculated on 50,000 users, each user has been assigned a certain number of devices. Devices can be either mobile or static. In the first case it is assumed that they are carried by users during their travels, in the second case they are considered to be left at the users' home. Obviously, although the SWIM mobility model allows to obtain an accurate correspondence between the output of the model itself and the most popular mobility traces available in CRAWDAD [51] (generating data with the same statistical properties), the mobility data thus obtained remain synthetic, simulated. The resulting network is constituted by a total of 16216 devices, of which 14600 for private users and 1616 for public services. The entire simulation lasts 10 days. As for the profile, it defines the set of possible services offered and applications distributed by each device category. These profiles can be useful in testing search and discovery algorithms in IoT/SIoT. The datasets obtained in [58] includes information of the object (category, owner, etc.), the traces on the locations and timestamps of the devices, the list of the applications expected in a Smart City scenario, the objects profiles (expressed as a set of available services and possible application requests) and social object relationships. Let's compare our work to that described in [58]. First of all we must highlight that in our case we are not interested in obtaining a specific dataset for Smart City Scenarios, but more general and more aimed at Social Networks on which the SIoT is based (SIoT = IoT + Social Networking). Second, in our work we do not use any simulator to generate the mobility traces of private devices, on the contrary we use real data. Another difference is that in our case the Interest Profile are not based on the category of the device as in [58], but on the places frequented by the user/device. In this way we are able to capture even interests that can's be known based only on the device category. Let's think about difficulty in understanding what are the interests associated with the category of device truly owned by the specific user. Another advantage is that in our case we have friendships, which we can't have using SWIM. Obviously in the case of Smart City Scenarios they may not be necessary, but in Social Network Scenarios

they are often fundamental. Furthermore, while trying to derive them in some way, with the SWIM procedure we will never have the certainty about the friendships between users. Among the procedures to derive the a posteriori friendships we mention the work in [72]. Unfortunately, in this type of work the low percentage of friendships obtained a posteriori should be noted. The drawbacks in our case are that the Home Points and the co-locations are not provided directly by the SWIM simulator and must be inferred. The accuracy of the method that we will use to infer the Home-Points it is higher than 85%. Regarding the co-locations, the advantage of SWIM is that it allows me to fully know the mobility of users (time continuous traces). On the contrary, through datasets we will know the mobility of users only in specific instants of time corresponding to the check-ins (time discrete traces), we will know only the positions of the check-ins. Finally, both in their work and in ours the traces are user-based and not device-based and the devices are obtained from the users. The procedure for assigning models to devices is identical. Due to the absence of real traces of mobility of objects, many works had to simulate them to conduct their own analyzes. The most used simulator to generate synthetic mobility traces is definitely SWIM [50]. Here are some of the works in which have been used [58, 8, 71, 57, 70]. Another method used in literature and in particular in [68, 69] is to analyze a real human social network (specifically the Brightkite social network obtained from the Stanford Large Network Dataset Collection [Leskovec]) and extract from this is the necessary information to build the social network of objects. At this point the characteristics of this network are extracted and used to run a model that generates synthetic networks with similar properties. In reality, to better analyze the properties of the network, they say that of the initial 58k nodes and 200k arcs they consider only 12k nodes and 40k arcs, in particular those between Atlanta and Boston. From this network they assume that each user carries at least one smart object so when they come in contact with their friends, their objects also come in contact and have the possibility to create a SOR. Similarly they simulate the creation of C-WORs and C-LORs. At this point from the resulting SIoT network with about 14.5k nodes and 67k arcs, to generate and analyze similar networks, they rely on the Barabási - Albert [85] model. Comparing this last method briefly with our work we note that the initial approach and in particular the idea of deriving the social network of objects starting from the social network between humans is very similar. The Brightkite dataset used is the same even if, unlike them, we consider it in full. Furthermore,

unlike them, we use the thresholds like in [72] to infer the co-locations and we resume the procedure for determining the Home-points from [20]. Finally, obviously in [68, 69] the part for obtaining the Profiles of Interests and Points of Interests is not dealt with.

5.3 Procedure for generating SIoT traces

1. To obtain the contacts between people (co-locations, meetings), we follow the procedure used in [72]. Starting with the check-ins dataset (Brightkite with approximately 4,700,000 check-ins) we will have co-location if and only if two check-ins of two different users occurred within 250 meters (space distance threshold $\Delta \leq 250$ meters) and within 1,800 seconds (30 minutes) (time distance threshold $\tau \leq 1800$ seconds). Assuming that each person is carrying a mobile device and leaving a fixed device at home, we get the *Contacts* (co-locations, meetings) that happened between the devices. To make the data more interesting, as in [72], we will only consider users with at least 10 check-ins and at least 10 different check-ins locations. This allows us to exclude inactive users.

   co-location (contacts, meetings) can be of three types:

   (a) Mobile-mobile case: when the checks-ins of two different users have occurred within a certain time interval below the time threshold (τ) and within a space distance below the space threshold (Δ).

   (b) Mobile-fixed case: when a mobile user's check-in and the Home-Point location (of another user or of the same user) are less than a certain space threshold (Δ) and the time distance between the mobile user's check-in and his next check-in is less than a certain time threshold (τ).

   (c) Fixed-fixed case: when the distance between two Home Points is less than a certain space threshold, there is no time threshold discourse as fixed positions.

2. [58] provides full distribution of commonly-owned device models. Based on this, we're doing the *Assignment of Models to Devices*. This is based on the ownership report of the Global Web Index in 2017 [37] calculated on 50000 users. This way, by

replacing each user with his devices, we get from meetings that happened between users, which happened between devices.

3. Now we need to understand how we got the user's *Home-Points* positions. These are not explicitly provided in datasets. Home-Point is crucial in determining where each user will leave their fixed devices. [80] offers a first way to infer Home-Points. In particular, by discretizing the world in cells with a size of 25km x 25km and defining Home-Point as the average position of the check-ins in the cell containing the largest number of check-ins. This method is also used in [21]. In [21], authors say that manual inspection shows that this method infers Home-Points with 85% accuracy. [20] has an even more accurate method, albeit a little heavier from a computational point of view. This method is based on recursive grid search. First, we group the check-ins in squares of 1 degree of latitude for 1 degree of longitude (covering approximately 4000 square miles). Then we select the square containing as many checkins as the center, and we select the eight neighboring squares to form a latex. We divide the latex into squares of 0.1 x 0.1 degree and repeat the selection procedures for the center and the neighbors. This process repeats until it reaches squares with a size of 0.001 x 0.001 degrees (covering approximately 0.004 square miles). Finally, we select the center of the square with the highest number of checks-ins, and the user's Home-Point. This is the method that we use to compute Home-points of the users. This method as explained in [20] does not suffer from the "Splitting-the-difference" problem that occurs when you choose the user's Home-Point equal to the center of all checks-ins. In fact, in this case, the Home Point of a user that lives in Houston and sometimes travels to Dallas, will be located between the two cities.

4. In general, there are places we will call for convenience *Point of Interest*, to which the nodes will most likely move. We can see them as places of particular importance, places where specific activities are carried out, where people cultivate certain interests. Around these places we will have a greater concentration of meetings (contacts). Each contact will take place more or less close to a place of interest. To each contact, we will assign a specific place of interest. This means that if a contact has taken place close to a certain Point of Interest, that user (device) is likely to attend that place of interest and

therefore have the interest of that specific place. In particular, to assign an Interest to a co-location, we used a Foursquare dataset [91–93] that associates each PoI (in terms of latitude and longitude) with an Interest. In practice, by putting into relationship the meeting position and the PoI position in the Foursquare dataset, it is possible to assign a relevant Interest to each co-location. As for the Interests considered in our experiments, we started with the Foursquare Interests but we had to group them into Macro-categories because they were defined in a very specific way and this had the effect that a large enough number of Communities with common Interest were not created to guarantee us a good statistical confidence in the results of our analyzes. Each Foursquare Interest is described by a single keyword, while a Macro-category (Interest) is made up of a set of keywords. The interest associated with a user (or device) and also with a content will be a Macro-category (Interest). All Foursquare's Interests fall into 52 Macro-categories that we call Interests. As an example, the Macro-categories we used in the performance evaluation studies illustrated in the remainder of the thesis are: Sweet Food (Interest 3) including the Foursquare Interests: {'Pastelaria', 'Ice Cream', 'Yogurt', 'Donut', 'Dessert'}; Italian Food (Interest 4) including the Foursquare Interests: {'Meatball', 'Wine', 'Pizza', 'Ice Cream'}; Café Bar (Interest 6) including the Foursquare Interests: {'Bistro', 'Breakfast', 'Cafe', 'Tea Room', 'Donut', 'Dive Bar', 'Cupcake', 'Coffee', 'Bar'}.

5. Obviously, a meeting near a certain PoI could happen casually. To understand if the user assiduously attends that PoI, a given number of meetings must take place near that place, or better, near places of that type. Therefore, we set a threshold on the minimum number of meetings near a PoI (set to 10 in the shown performance campaign) required to assign that Interest to the user. In this way, from the co-locations (meetings) it is possible to obtain the PoI frequented by people, from which we can obtain their interests.

6. Now we can create the data, which are the source-owned contents. We also *assign the content to nodes (devices)*: a data can be assigned to a node if and only if the interest associated with the data is also an interest owned by that node.

7. Now let's see how to establish the *Community*. We've seen that, given a Source that wants to diffuse its content related to a specific interest, the Source Community is made up of Source itself and all of the interested (to the content) reachable nodes to which the Source can spread its content directly or indirectly. In the same way also the other communities will be formed. The Community of an Interested Node consist of the Interested Node itself and all of the interested (to the content) reachable nodes to which the Interested Node could disseminate the content directly or indirectly. In addition, we have already seen that two nodes belonging to the same Community can reach each other directly (DF-RCs, DS-RCs) or Indirectly (I-RCs) through other nodes. Two nodes belonging to different communities cannot reach each other either directly or indirectly, they are called Unreachable Contacts (UCs).

   (a) From the Brightkite dataset [21] we get the *friendships* among people. Users are friends because they know each other, so their devices also are friends. Each device will have a number of friends stored in its own PODs, they are the DF-RCs.

   (b) We must now establish the *Authorizations to Access Contacts in Solid PODs*. Each user can decide which of their contacts to make visible and to which users to make visible. As the social distance increases, less likely a node allow another to access its own PODs. The Source DS-RCs, are the contacts contained in the PODs of other user (her contacts) to which it is authorized to access.

   (c) At this point, we need to compute all the interested nodes that can be reached through intermediaries, otherwise not reachable from the Source; in brief all the I-RCs.

   (d) The Discovery result will return the only interested nodes, the ED-IRCs. At this point we can find the communities.

   (e) Assuming that there's a Source that wants to spread a content of interest, we can infer the *Community*. The Community of a node is constituted by its DF-IRCs plus DS-IRCs plus I-IRCs.

8. From Contacts, we build the *Social Object Relationships* according to the SIoT rules already in the literature [8]. The first social neighbors of a node are the nodes with

which it has directly established at least one Social Object Relationship. In particular, we consider that between two devices, an OOR is set up if they have the same owner. A POR is installed if the devices are of the same model. A C-LOR if both devices meet in their Home-Points. In the case of C-LOR, we consider that the distance between the two Home-Points of the two nodes must be less than or equal to 10 meters and the distance between each of the two nodes and its own Home-Point must be less than or equal to 10 meters. Finally, a SOR is set up if the device owners meet at least 3 times, if the considered meetings take place at intervals longer than 6 hours (Ti) and if each of the encounters lasts more than 10 minutes (Tc).

9. Each node will send to its first social neighbors its Interest Descriptor to compute VUIP-VUIP similarity. In this way from social object relationships we establish the *C-IORs* according to the basic establishment mechanism. As constraints, we set the time to live to 6 hops for spreading each node's Interest Descriptor and the similarity threshold to 0.5.

Some datasets generated through this procedure are available to other researchers upon request.

Chapter 6

Simulation campaign

6.1 Use cases

In the studies presented in the remainder of the thesis, the objective is to compare the mean IRN percentage (i.e., the percentage of interested nodes reachable) obtained in the cases *Enhanced SIoT*, in which we use the basic mechanism for the establishment of C-IOR and *Friendships*, in which only Brightkite friendships are leveraged.

In general in the simulations since we are dealing with percentages (percentage of nodes that spread the VUIP of the Source to each hop and of nodes that authorize access to their PODs at each hop), based on the randomly selected nodes we will have different results in terms of IRN Percentage (of interested nodes reachable). Consequently, for greater precision we have carried out more simulations and for each point of each of the curves we have obtained more values. Each point of each curve is the average of the values obtained at that point in the different simulations. This applies to all simulations.

More specifically, in the Enhanced SIoT case, each node will be able to diffuse the Source content to all its DF-IRCs and DS-IRCs as well as its IRSCs; in other words to all the interested nodes contained in its own Contacts List, in the Contact Lists of other users (her contacts) to which it is authorized to access and in its own SIoT Contact List.

In the Friendships case, each node will be able to diffuse the Source content only to all its DF-IRCs and DS-IRCs; in other words to all the interested nodes contained in its own

Contacts List and in the Contact Lists of other users (her contacts) to which it is authorized to access.

6.2 Assumptions

The following assumptions hold:

- All SIoT relationships are considered except the C-WORs, which as demonstrated in the article [57] has a negligible contribution in terms of navigability [58].
- A threshold of 10 check-ins is set in a specific type of PoI for the assignment of the relevant interests of a user.
- Each person brings a mobile device with her and leaves a fixed device at her home.
- In the Enhanced SIoT case (with C-IOR basic mechanism) node *A* spreads the Source data to node *B* of another community if and only if:
  - the two nodes are connected via a SIoT relationship or via a SIoT relationships path (connection in the social graph of devices);
  - the VUIPs of the two nodes have a similarity higher than a certain threshold (Cosine Similarity ≥ 0.5 [97]). The first two conditions imply the establishment of a C-IOR between the two nodes;
  - the *B* node has the specific interest of the data (which the *A* node wants to spread) in its own VUIP. The third condition implies the presence of a C-IOR between the two nodes associated to such specific interest.
- Each node that has SIoT relationships with nodes belonging to other communities (communities other than its own) acts as a potential Mediator.
- Scenarios as realistic as possible are considered. A limit is set on the number of hops for the diffusion of the Source's VUIP (TTL) in the Discovery phase, as it is more realistic to assume that not all nodes are willing to spread the VUIP on behalf of another node. The percentage of nodes that spreads the Source VUIP to the different

hops is varied during the simulations. In addition, since it is objectively less likely that a node makes its contacts available when increasing the social distance, then the percentage of nodes that provide authorization to access their PODs is assumed lower as the number of hops increases.

- It is assumed that every Source that spreads its own content will spread it to all possible interested nodes. In particular, in the Friendships case it will spread it to its DF-IRCs and DS-IRCs; in other words to all the interested nodes contained in its own Contacts List and in the Contact Lists of other users (her contacts) to which it is authorized to access. In the Enhanced SIoT case it will diffuse it to its DF-IRCs, DS-IRCs, and to its IRSCs; in other words to all the interested nodes contained in its own Contacts List, in the Contact Lists of other users (her contacts) to which it is authorized to access and in its own SIoT Contact List.
- Without losing generality, we assume that unless otherwise indicated we consider Interest 3 ("Sweet Food") and the related Communities.
- It is assumed that not only interested nodes, but also not interested ones can authorize access to their PODs. This is important in calculating DS-IRCs.
- Unless otherwise indicated, all nodes, including isolated nodes, are considered.

6.3 Performance by varying the number of nodes that spread the Source's VUIP

The aim of the first performance evaluation is to investigate how the mean IRN percentage varies when varying the percentage of the nodes that diffuse the VUIP of the Source at each hop, by keeping fixed the percentage of nodes that authorize access to their PODs. The nodes that spread the Source's VUIP are the nodes that act as intermediaries, allowing the Source to reach Contacts otherwise unreachable. The reported results consider a percentage of the nodes that spreads the Source's VUIP at each hop equal to 100%, 90%, 60%, 30%, and 10%, and a number of hops for the VUIP diffusion equal to 4. All simulations were carried out in order to obtain a high statistical confidence (95%).

In Fig. 6.1 the solid curves represent the trends obtained when exploiting all the social object relationships in the Enhanced SIoT case. The dotted curves in figure represent the trends obtained if only Brightkite friendships are used (Friendships case). It is assumed that the percentage of nodes that authorize access to their PODs is 100% at the first hop.

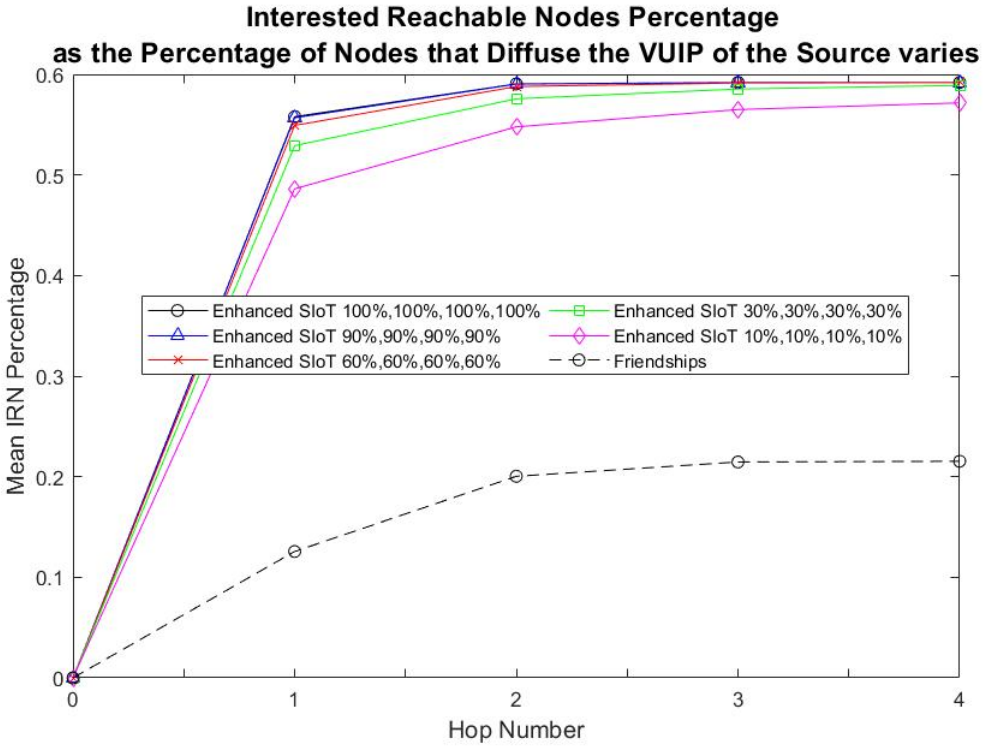


Fig. 6.1 Mean IRN percentage as the percentage of nodes that diffuses the Source's VUIP at the different hops varies (Enhanced SIoT case vs. Friendships case).

By observing Fig. 6.1 we can appreciate the higher values in terms of mean IRN percentage obtained in the Enhanced SIoT case compared to the Friendships case. This means that through the Enhanced SIoT it is possible to reach a greater number of interested nodes. This is due to the presence of SIoT relationships and of all the additional proposed features and mechanism previously described, from the Mediator object to the basic establishment mechanism for the C-IOR.

The first two hops are those that have a greater increase in terms of mean IRN percentage (greater slope). We can note also the faster convergence in the Enhanced SIoT case compared to the Friendships case. This not only means that with the Enhanced SIoT a greater number

of interested nodes can be reached, but also that they can be reached in a lower number of hops.

By observing Fig. 6.1 also clearly emerges, as we expected, that the obtained values in terms of mean IRN percentage increase with the increase in the percentage of nodes that diffuses the Source's VUIP and with the increase in the number of hops. We can note that also in the worst Enhanced SIoT case (in which only the 10% of the nodes diffuse the Source VUIP at each hop), we obtain higher performance levels with respect to the Friendships case.

The low values obtained in general depend on the high number of interested isolated nodes present in the network for the specific scenario chosen. As the number of hops increases, the increase in terms of mean IRN percentage becomes smaller, because most of the interested nodes that can be reached have already been reached.

Table 6.1 Performance by varying the number of nodes that spread the Source's VUIP.

Mean IRN Percentage **PercAcces2POD=[1]** **Hop 1 in Diffusion**	**Friendships**	**Enhanced SIoT**
PercNodeDiff=[1]	0.1256	0.5579
PercNodeDiff=[0.9]	0.1255	0.5565
PercNodeDiff=[0.6]	0.121	0.5502
PercNodeDiff=[0.3]	0.1024	0.5313
PercNodeDiff=[0.1]	0.0729	0.4871
Hop 2		
PercNodeDiff=[1,1]	0.2004	0.5905
PercNodeDiff=[0.9,0.9]	0.2002	0.5903
PercNodeDiff=[0.6,0.6]	0.1947	0.5885
PercNodeDiff=[0.3,0.3]	0.1706	0.578
PercNodeDiff=[0.1,0.1]	0.1133	0.5457
Hop 3		
PercNodeDiff=[1,1,1]	0.2146	0.5919

Continued on next page

Table 6.1 – continued from previous page

	Friendships	**Enhanced SIoT**
PercNodeDiff=[0.9,0.9,0.9]	0.2146	0.5919
PercNodeDiff=[0.6,0.6,0.6]	0.2136	0.5914
PercNodeDiff=[0.3,0.3,0.3]	0.2034	0.5864
PercNodeDiff=[0.1,0.1,0.1]	0.1475	0.5655
Hop 4		
PercNodeDiff=[1,1,1,1]	0.2153	0.5919
PercNodeDiff=[0.9,0.9,0.9,0.9]	0.2153	0.5919
PercNodeDiff=[0.6,0.6,0.6,0.6]	0.2152	0.5918
PercNodeDiff=[0.3,0.3,0.3,0.3]	0.212	0.5897
PercNodeDiff=[0.1,0.1,0.1,0.1]	0.1732	0.5729

6.4 Performance as the percentage of nodes that authorize access to their PODs changes

The second study aims to investigate how the mean IRN percentage varies with the percentage of nodes that authorize access to their PODs at different hops, by keeping the percentage of nodes that spread the Source's VUIP fixed. Let's consider the limit of 4 hops in which this time there will be nodes authorizing the access to their PODs. The label of Fig. 6.2 report the percentages of nodes that authorize the Source to access their PODs in each of the 4 hops. Again, in Fig. 6.2 the solid curves represent the trends obtained when all the social object relationships are considered in the Enhanced SIoT case. The dotted curves in figure represent the trends obtained if only Brightkite friendships are used (Friendships case). We assume that the percentage of nodes diffusing the Source VUIP is 100% at the first hop, i.e. all the nodes spread the Source's VUIP.

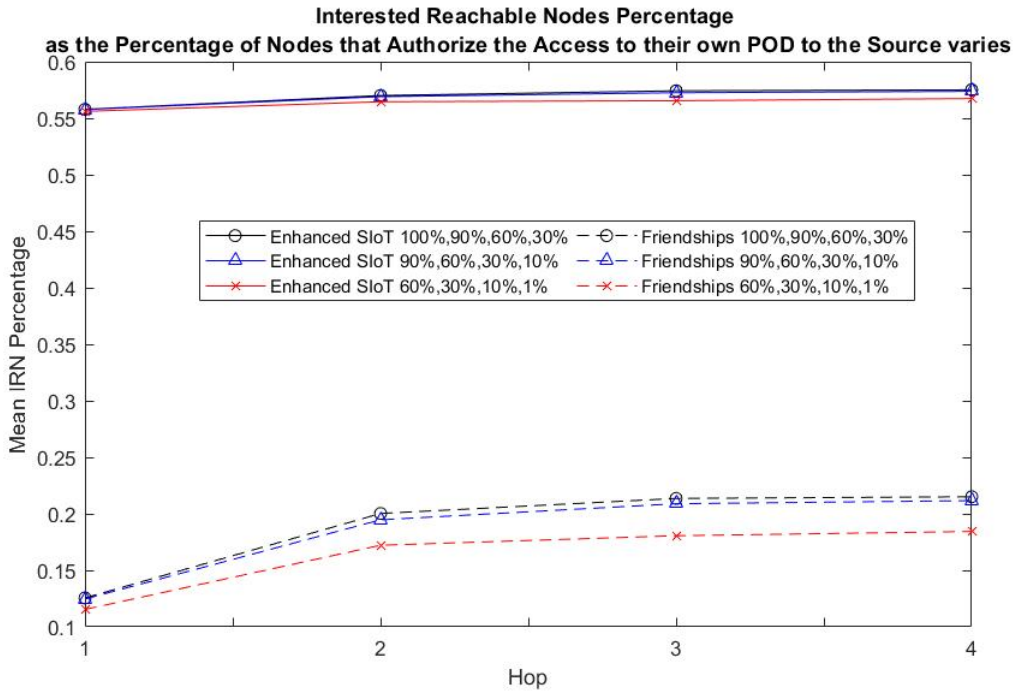


Fig. 6.2 Mean IRN percentage when varying the percentage of nodes authorizing access to their PODs at the different hops (Enhanced SIoT case vs. Friendships case).

From Fig. 6.2 we can note the higher values in terms of mean IRN percentage obtained in the Enhanced SIoT case compared to the Friendships case. Also here through the Enhanced SIoT we are able to reach a greater number of interested nodes. By observing Fig. 6.2 it also clearly emerges, as we expected, that the obtained values in terms of mean IRN percentage increase with the increase in the percentage of nodes that authorizes the access to their PODs and with the increase in the number of hop (in which there are node that provide authorization to access their PODs to the Source). We can note that also in the worst Enhanced SIoT case, we obtain higher performance levels with respect to the Friendships case. Here, again the low values in general depend on the high number of interested isolated nodes present in the network. The reader notes that the gain obtained with a higher percentage of nodes that authorize to see their contacts is more accentuated in the Friendships case than in the Enhanced SIoT case. Also, the first two hops are those that show a greater increase in terms of mean IRN percentage (greater slope of the curves). This is due both to the fact that with the increase in the number of hops, most of the nodes that can be reached have already been reached, and to the fact that in the first hops we set higher percentages of nodes authorizing

access to their PODs. This latter assumption has not to surprise because it is correct to assume that friends are more willing to authorize access to their PODs, than friends of friends and so on. The more socially distant one node is, the less likely this node will authorize access to its PODs.

Table 6.2 Performance as the percentage of nodes that authorize access to their PODs changes.

Mean IRN Percentage **PercNodeDiff=[1]** **Hop 1 in Access**	**Friendships**	**Enhanced SIoT**
PercAcces2POD=[1]	0.1256	0.5579
PercAcces2POD=[0.9]	0.1251	0.5578
PercAcces2POD=[0.6]	0.1166	0.5555
Hop 2		
PercAcces2POD=[1,0.9]	0.2003	0.5701
PercAcces2POD=[0.9,0.6]	0.1951	0.5685
PercAcces2POD=[0.6,0.3]	0.1654	0.5635
Hop 3		
PercAcces2POD=[1,0.9,0.6]	0.2138	0.5743
PercAcces2POD=[0.9,0.6,0.3]	0.2093	0.5728
PercAcces2POD=[0.6,0.3,0.1]	0.18	0.5662
Hop 4		
PercAcces2POD=[1,0.9,0.6,0.3]	0.215	0.5751
PercAcces2POD=[0.9,0.6,0.3,0.1]	0.2105	0.5731
PercAcces2POD=[0.6,0.3,0.1,0.01]	0.1836	0.5677

6.5 Performance by varying the kind of SIoT relationships between devices

A further objective of our study is to observe how the mean IRN percentage changes when the combination of SIoT relationships vary. For this purpose, simulations have been conducted in which six different combinations of SIoT relationships are considered. Fig. 6.3 shows the variation of the mean IRN percentage, assuming that the 100%, 90%, 60%, and 30% of nodes respectively spreads the VUIP of the Source (act as intermediaries), in the Enhanced SIoT case.

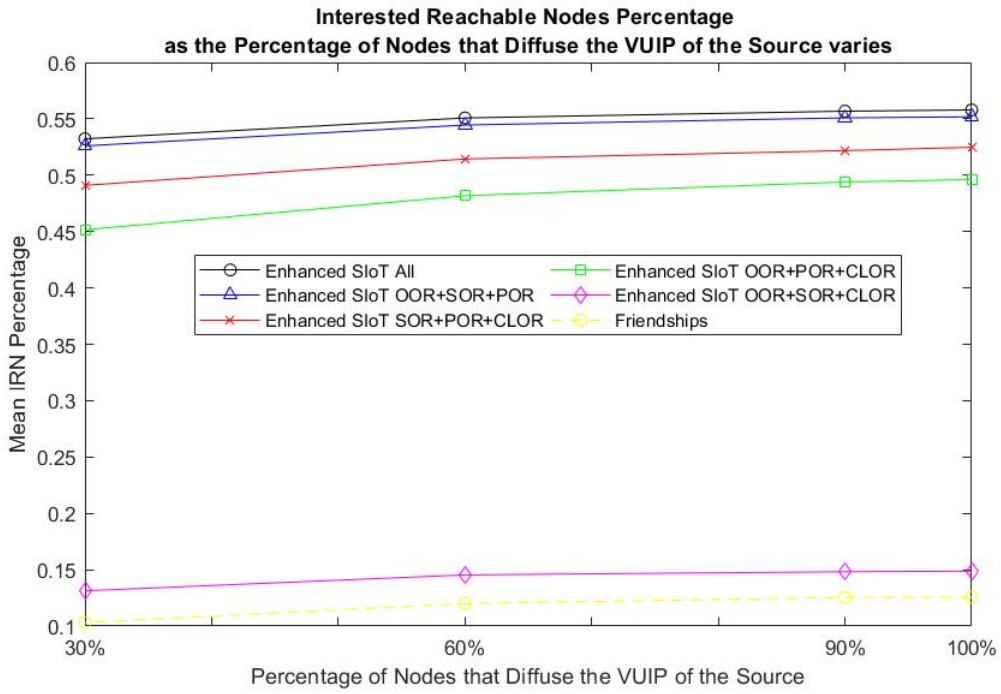


Fig. 6.3 Mean IRN percentage for different combination of SIoT relationships, as the percentage of nodes that diffuses the Source's VUIP changes (Enhanced SIoT vs. Friendship).

A first evident result is that POR is clearly the social object relationship that weighs most on the obtainable mean IRN percentage values, followed by the SOR, the OOR, and the C-LOR. POR friendships in fact depend only on the model of the device and are often relationships that connect devices that are very distant from each other and belong to different communities. Given their characteristic of being "long-range" relationships, the relevant role, confirmed by the curves, in connecting users belonging to different communities otherwise

separated was expected. The advantage in terms of the considered metric that the Enhanced SIoT case offer compared to the Friendships case, for any combination of SIoT relationships, is evident from the curves shown in Fig. 6.3; the values in terms of mean IRN percentage obviously increases with the increase in the percentage of nodes that spread the Source's VUIP.

Table 6.3 Mean IRN percentage for each combination of SIoT relationships, as the Percentage of the nodes diffuses the Source's VUIP, in the Enhanced SIoT case.

Mean IRN Percentage PercAcces2POD=[1]	**100%NodeDiff**	**90%NodeDiff**	**60%NodeDiff**	**30%NodeDiff**
SIoT ALL	0.5579	0.5568	0.5508	0.5323
OOR+SOR+POR	0.5518	0.5509	0.5445	0.526
SOR+POR+C-LOR	0.5248	0.5218	0.5144	0.491
OOR+POR+C-LOR	0.4961	0.494	0.4818	0.4516
OOR+SOR+C-LOR	0.1487	0.1482	0.1454	0.1313
Only Friendships	0.1256	0.1254	0.12	0.1031

6.6 Performance by varying the type of Interest

Up till now, in our performance evaluation study we have always considered Interest 3. Obviously, the performance figures may depend on the scenario considered. Fig. 6.4 shows the output of a study aimed at comparing, in terms of obtainable mean IRN percentage, what happens in the Enhanced SIoT compared to the Friendships case in scenarios characterized by different Interests. The goal is to understand if and how much the advantages of leveraging the SIoT relationships, compared to the case where they are not exploited, depend on the type of Interest considered.

Without losing generality, we consider six hops for the diffusion of the Source VUIP and we establish that the percentage of nodes that spread the VUIP of the Source and that authorize access to their PODs to the Source are both 100% at each hop.

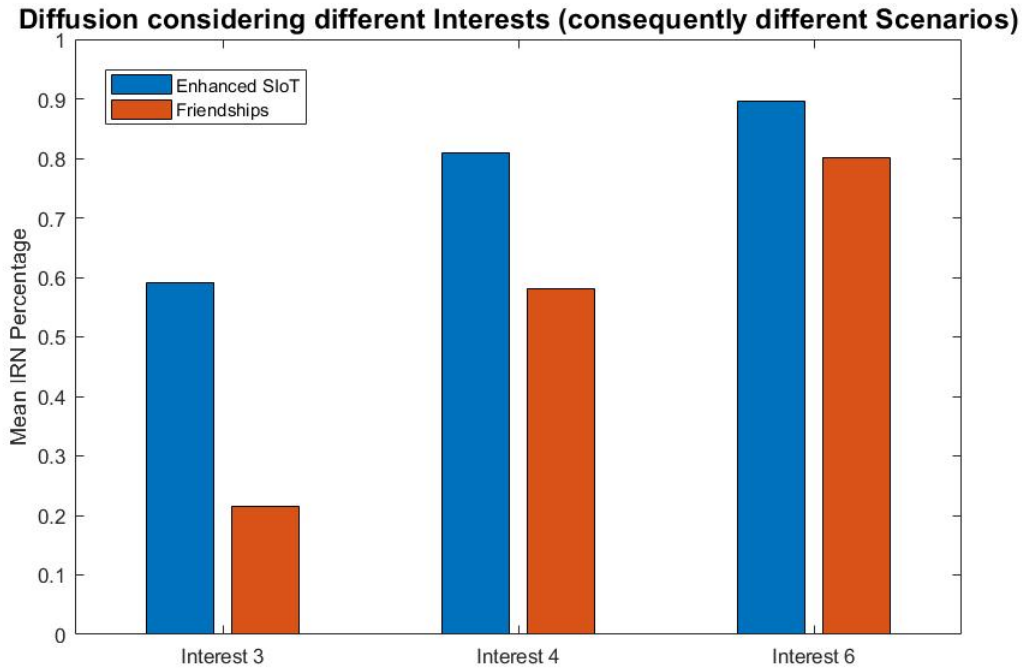


Fig. 6.4 Mean IRN percentage when varying the considered Interest.

Table 6.4 Mean IRN percentage considering different interests.

Mean IRN Percentage PercAcces2POD=[1] PercNodeDiff=[1,1,1,1,1,1]	**Interesse 3 (Sweet Food)**	**Interesse 4 (Italian Food)**	**Interesse 6 (Bar Caffè)**
Enhanced SIoT	0.5919	0.8099	0.8967
Friendships	0.2153	0.5817	0.8017

In Fig. 6.4 we see that the percentage of nodes belonging to the giant component increases when moving from Interest 3 to Interest 6.

In cases where almost all nodes belong to the giant component, by considering just the Friendships case, the Source manage to reach almost all the interested nodes. Consequently, the advantage in terms of mean IRN percentage obtained by using the Enhanced SIoT compared to the Friendships case is reduced. Like before, if we do not consider the interested but isolated nodes, the behaviours remain the same while the reachable performance levels are higher, as shown in Fig. 6.5

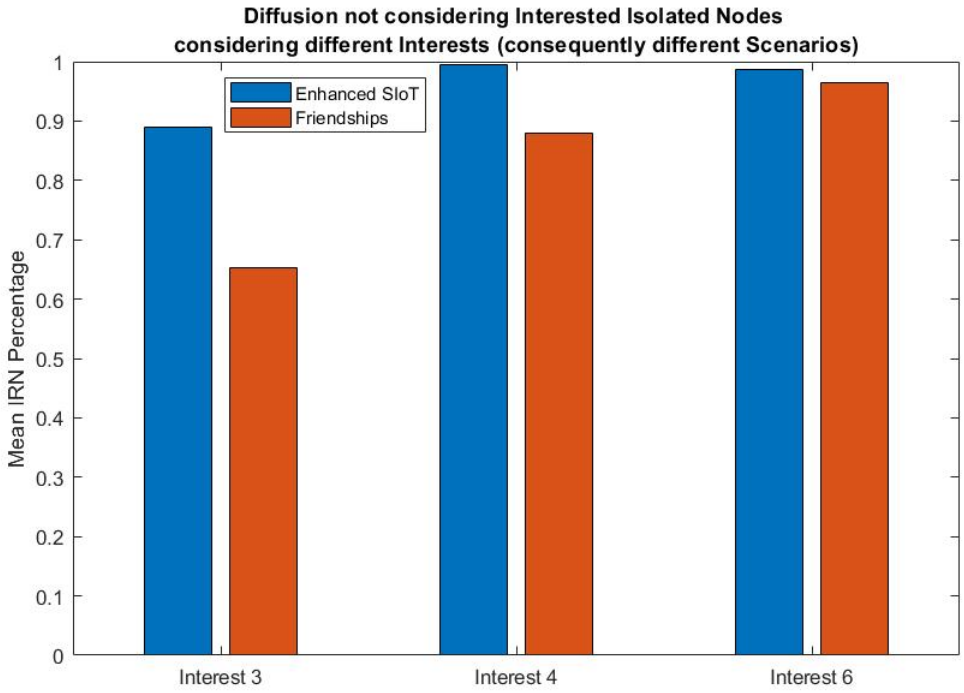


Fig. 6.5 Mean IRN percentage when considering different Interests (isolated nodes NOT considered).

Table 6.5 Mean IRN percentage considering different interests (isolated nodes NOT considered).

Mean IRN Percentage PercAcces2POD=[1] PercNodeDiff=[1,1,1,1,1,1]	**Interesse 3 (Sweet Food)**	**Interesse 4 (Italian Food)**	**Interesse 6 (Bar Caffè)**
Enhanced SIoT	0.889	0.9948	0.9869
Friendships	0.6526	0.88	0.9645

6.7 Comparison between Discovery that considers both interested and not interested nodes vs Discovery that considers only the interested ones

This Section aims to demonstrate and quantify the advantage in terms of mean IRN percentage that is obtained also considering the nodes not interested in the Discovery phase. We want to demonstrate that the role of intermediaries played by not interested nodes that allows the Source to indirectly reach nodes otherwise unreachable is fundamental. Then we will show how, also considering the nodes not interested in Discovery, the Source will be able to reach a greater number of interested nodes by varying the percentage of nodes that spread the VUIP of the Source, by varying the percentage of nodes that authorize access to their own PODs and varying the interest considered.

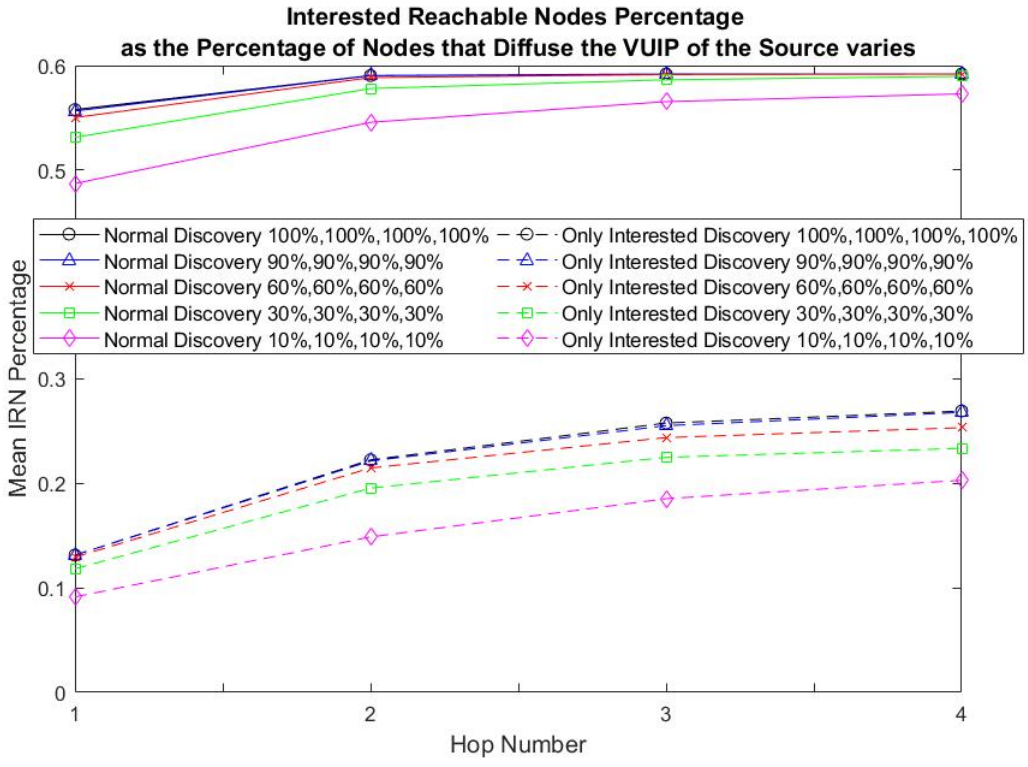


Fig. 6.6 Mean IRN percentage as the percentage of nodes that diffuses the Source VUIP at the different hops varies (Normal Discovery case vs. Only Interested Discovery case).

First of all we investigate how the mean IRN percentage varies in both the Discovery cases when varying the percentage of the nodes that diffuse the VUIP of the Source at each hop, by keeping fixed the percentage of nodes that authorize access to their PODs. The reported results consider a percentage of the nodes that spreads the Source VUIP at each hop equal to 100%, 90%, 60%, 30%, and 10%, and a number of hops for the VUIP diffusion equal to 4. In Fig. 6.6 the solid curves represent the trends obtained when exploiting both the interested and not interested nodes in Discovery phase. The dotted curves in figure represent the trends obtained if only the interested ones are used. We have assumed that the percentage of nodes that authorize access to their PODs is 100% at the first hop. In Fig. 6.6 we can appreciate the higher values in terms of mean IRN percentage obtained considering both the interested and not interested nodes compared to case in which we consider only the interested ones. This means that in the first case we are able to reach a greater number of interested nodes. This is due to the presence of not interested intermediaries.

Table 6.6 Performance by varying the number of nodes that spread the Source VUIP.

Mean IRN Percentage **PercAcces2POD=[1]** **Hop 1 in Diffusion**	**Normal Discovery**	**Only Interested Discovery**
PercNodeDiff=[1]	0.5579	0.1314
PercNodeDiff=[0.9]	0.5565	0.1313
PercNodeDiff=[0.6]	0.5502	0.1294
PercNodeDiff=[0.3]	0.5313	0.1183
PercNodeDiff=[0.1]	0.4871	0.0914
Hop 2		
PercNodeDiff=[1,1]	0.5905	0.2226
PercNodeDiff=[0.9,0.9]	0.5903	0.2215
PercNodeDiff=[0.6,0.6]	0.5885	0.2148
PercNodeDiff=[0.3,0.3]	0.578	0.1955
PercNodeDiff=[0.1,0.1]	0.5457	0.1488
Hop 3		

Continued on next page

Table 6.6 – continued from previous page

	Normal Discovery	**Only Interested Discovery**
PercNodeDiff=[1,1,1]	0.5919	0.2578
PercNodeDiff=[0.9,0.9,0.9]	0.5919	0.2553
PercNodeDiff=[0.6,0.6,0.6]	0.5914	0.2439
PercNodeDiff=[0.3,0.3,0.3]	0.5864	0.225
PercNodeDiff=[0.1,0.1,0.1]	0.5655	0.1854
Hop 4		
PercNodeDiff=[1,1,1,1]	0.5919	0.2693
PercNodeDiff=[0.9,0.9,0.9,0.9]	0.5919	0.268
PercNodeDiff=[0.6,0.6,0.6,0.6]	0.5918	0.2534
PercNodeDiff=[0.3,0.3,0.3,0.3]	0.5897	0.2336
PercNodeDiff=[0.1,0.1,0.1,0.1]	0.5729	0.203

Now we investigate how the mean IRN percentage varies in both the Discovery cases when varying the percentage of nodes that authorize access to their PODs at different hops, by keeping the percentage of nodes that spread the Source VUIP fixed. Let's again consider the limit of 4 hops in which this time there will be nodes authorizing the access to their PODs. The label of Fig. 6.7 report the percentages of nodes that authorize the Source to access their PODs in each of the 4 hops. Again, in Fig. 6.7 the solid curves represent the trends obtained when exploiting both the interested and not interested nodes in Discovery phase. The dotted curves in figure represent the trends obtained if only the interested ones are used. We assume that the percentage of nodes diffusing the Source VUIP is 100% at the first hop, i.e.all the nodes spread the Source VUIP. From Fig. 6.7 we can note the higher values in terms of mean IRN percentage obtained considering both the interested and not interested nodes compared to case in which we consider only the interested ones. Also here in the first case we are able to reach a greater number of interested nodes.

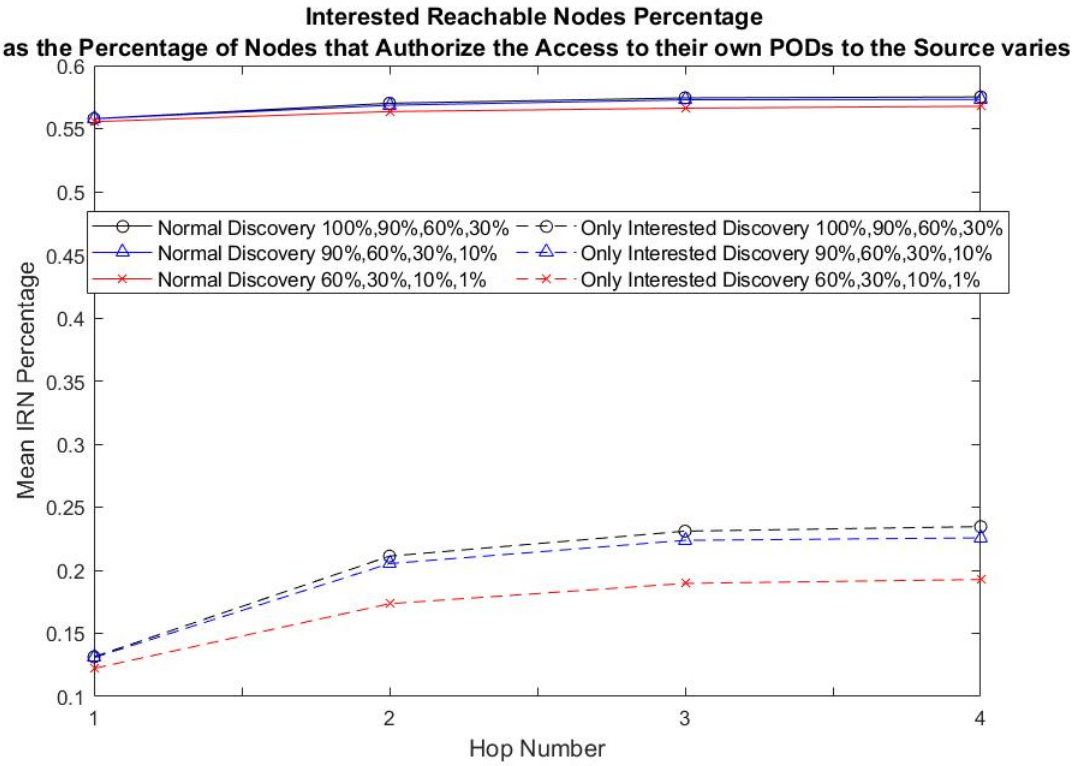


Fig. 6.7 Mean IRN percentage when varying the percentage of nodes authorizing access to their PODs at the different hops (Normal Discovery case vs. Only Interested Discovery case).

Table 6.7 Performance as the percentage of nodes that authorize access to their PODs changes.

Mean IRN Percentage **PercNodeDiff=[1]** **Hop 1 in Access**	**Normal Discovery**	**Only Interested Discovery**
PercAcces2POD=[1]	0.5579	0.1314
PercAcces2POD=[0.9]	0.5578	0.1308
PercAcces2POD=[0.6]	0.5555	0.1222
Hop 2		
PercAcces2POD=[1,0.9]	0.5701	0.2112
PercAcces2POD=[0.9,0.6]	0.5685	0.2053
PercAcces2POD=[0.6,0.3]	0.5635	0.1735
Hop 3		

Continued on next page

Table 6.7 – continued from previous page

	Normal Discovery	**Only Interested Discovery**
PercAcces2POD=[1,0.9,0.6]	0.5743	0.231
PercAcces2POD=[0.9,0.6,0.3]	0.5728	0.2237
PercAcces2POD=[0.6,0.3,0.1]	0.5662	0.1896
Hop 4		
PercAcces2POD=[1,0.9,0.6,0.3]	0.5751	0.2346
PercAcces2POD=[0.9,0.6,0.3,0.1]	0.5731	0.2257
PercAcces2POD=[0.6,0.3,0.1,0.01]	0.5677	0.1927

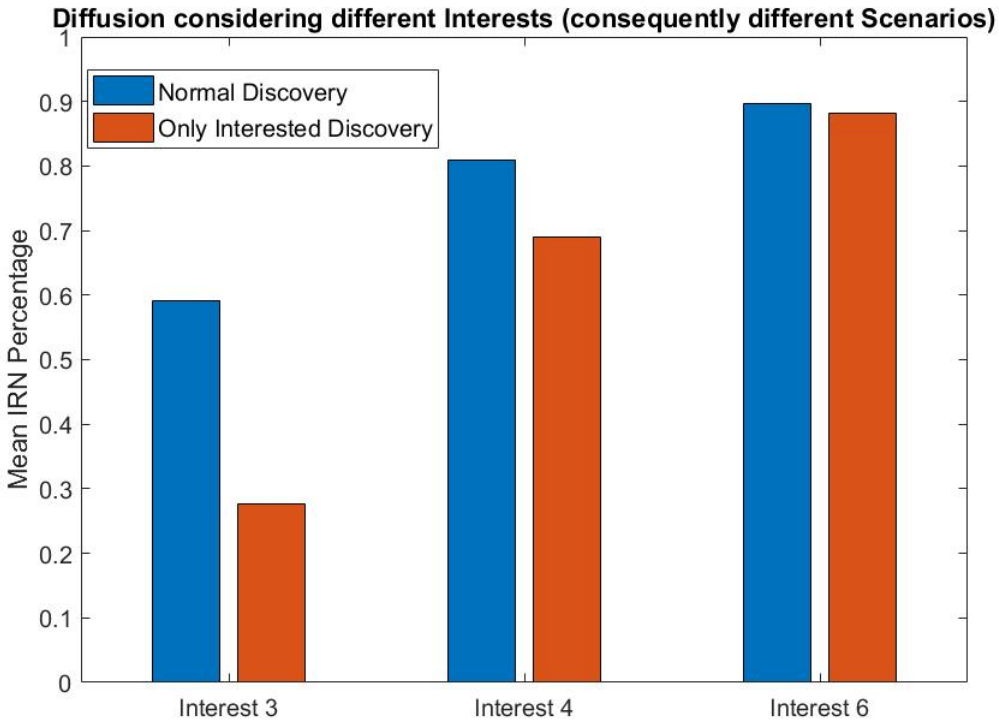


Fig. 6.8 Mean IRN percentage when varying the considered Interest.

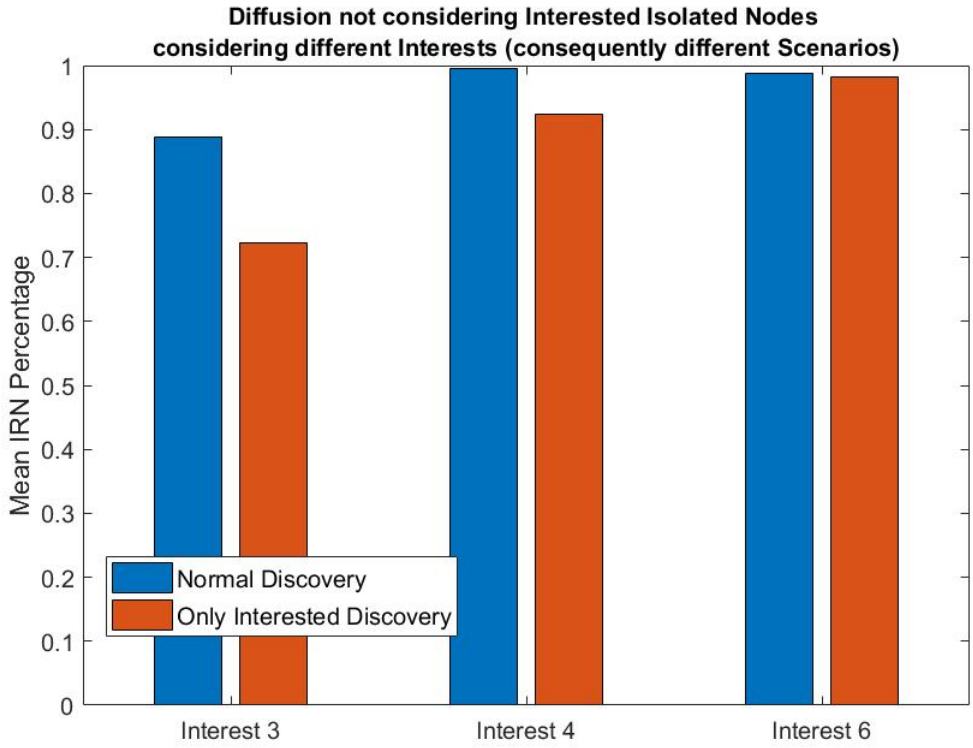


Fig. 6.9 Mean IRN percentage when considering different Interests (isolated nodes NOT considered).

Finally we compare in terms of obtainable mean IRN percentage, what happens in both the Discovery cases in scenarios characterized by different Interests. We consider six hops for the diffusion of the Source VUIP and we establish that the percentage of nodes that spread the VUIP of the Source and that authorize access to their PODs to the Source are both 100% at each hop. In Fig. 6.8 we can appreciate the higher values in terms of mean IRN percentage obtained considering both the interested and not interested nodes compared to case in which we consider only the interested ones. This means that in the first case we are able to reach a greater number of interested nodes. Like before, if we do not consider the isolated interested nodes, the behaviours remain the same while the reachable performance levels are higher, as shown in Fig. 6.9.

Table 6.8 Mean IRN percentage considering different interests.

Mean IRN Percentage PercAcces2POD=[1] PercNodeDiff=[1,1,1,1,1,1]	**Interesse 3 (Sweet Food)**	**Interesse 4 (Italian Food)**	**Interesse 6 (Bar Caffè)**
Normal Discovery	0.5919	0.8099	0.8967
Only Interested Discovery	0.2764	0.6904	0.8813

Table 6.9 Mean IRN percentage considering different interests (isolated nodes NOT considered).

Mean IRN Percentage PercAcces2POD=[1] PercNodeDiff=[1,1,1,1,1,1]	**Interesse 3 (Sweet Food)**	**Interesse 4 (Italian Food)**	**Interesse 6 (Bar Caffè)**
Normal Discovery	0.889	0.9948	0.9869
Only Interested Discovery	0.7234	0.923	0.9817

6.8 Comparison between the Mean Number of Hops to reach all the Interested Reachable Nodes considering and not considering the C-IORs

This Section aims to demonstrate and quantify the advantage in terms of number of hops that the Source employs to reach all the interested nodes reachable during the Discovery phase considering the C-IORs. In fact, the use of C-IORs allows a faster Discovery, thanks to the reduced number of hops that the Source will have to carry out to reach all the nodes interested. Also in this case we have considered the Interest 3. To calculate the shortest paths between the Source and each of its IRNs (interested reachable nodes) we have used the Dijkstra algorithm implemented in matlab. The Fig. 6.10 shows the results obtained for different Sources.

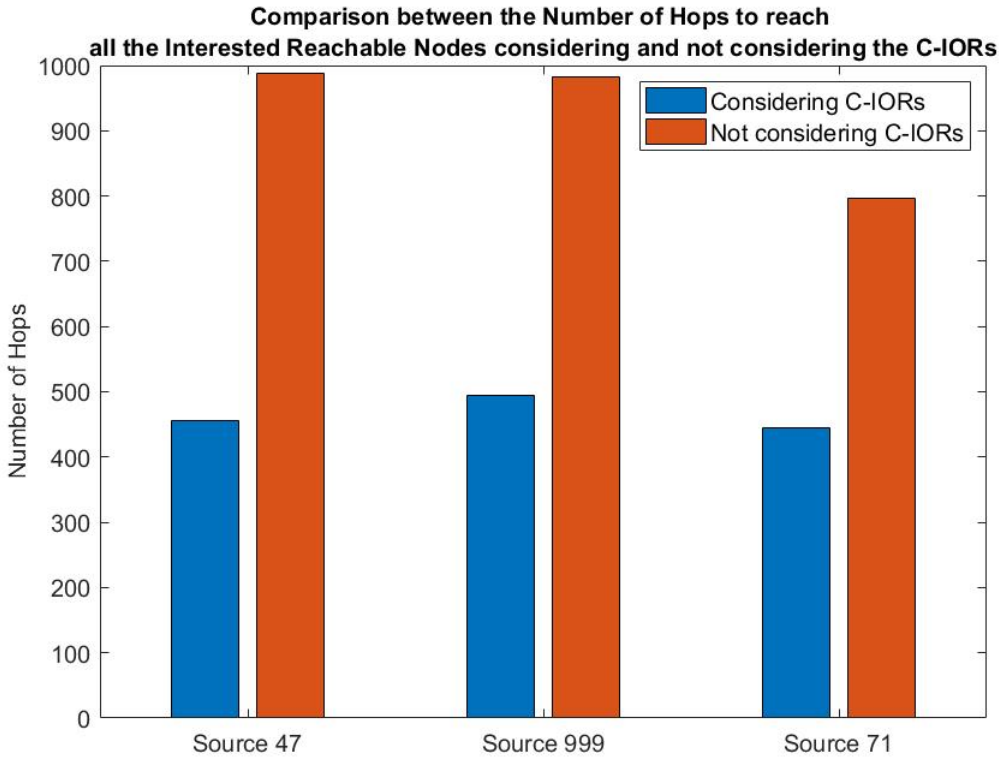


Fig. 6.10 Comparison between the Number of Hops that each Source employs to reach all her Interested Reachable Nodes considering and not considering the C-IORs.

Table 6.10 Comparison between the Number of Hops that each Source employs to reach all her Interested Reachable Nodes considering and not considering the C-IORs.

Number of Hops PercAcces2POD=[1] PercNodeDiff=[1,1,1,1,1,1]	**Source 47**	**Source 999**	**Source 71**
Considering C-IORs	455	494	445
Not considering C-IORs	988	983	797

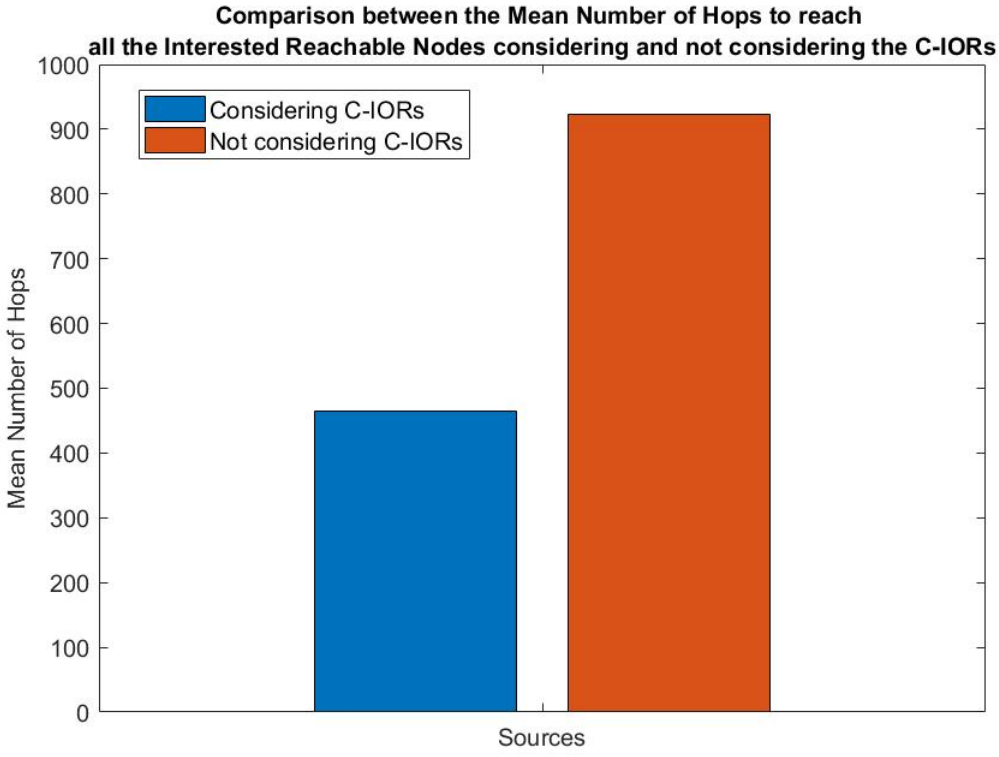


Fig. 6.11 Comparison between the Mean Number of Hops to reach all the Interested Reachable Nodes considering and not considering the C-IORs.

Table 6.11 Comparison between the Mean Number of Hops to reach all the Interested Reachable Nodes considering and not considering the C-IORs.

Mean Number of Hops **PercAcces2POD=[1]** **PercNodeDiff=[1,1,1,1,1,1]**	**Hops**
Considering C-IORs	465
Not considering C-IORs	923

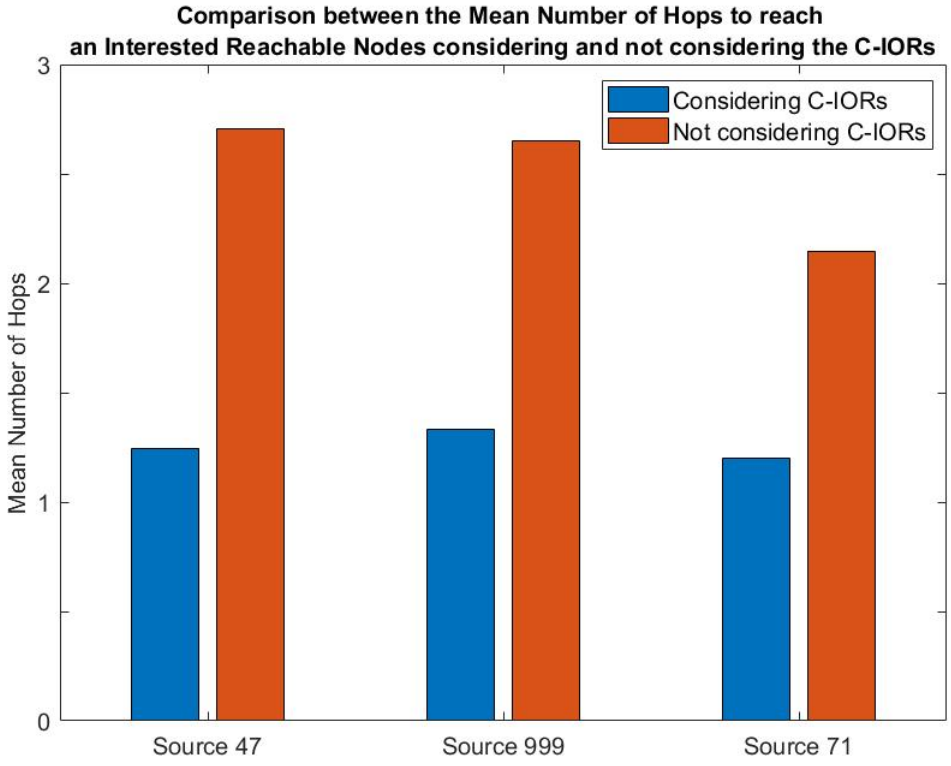


Fig. 6.12 Comparison between the Mean Number of Hops that each Source employs to reach an Interested Reachable Node considering and not considering the C-IORs.

By observing Fig. 6.11 we deduce that thanks to the presence of the C-IORs the mean number of hops is halved, consequently halving the times required in Discovery phase. This is due to the fact that the basic mechanism for establishing C-IORs allows the establishment

of direct Social Links (C-IORs) with interested nodes that would otherwise be connected but through a chain of SIoT relationships that could also involve non-interested nodes to the content. The advantage in terms of mean number of hops is very high, the disadvantage is the slight increase in computational complexity introduced by the basic mechanism for establishing C-IORs.

Table 6.12 Comparison between the Mean Number of Hops that each Source employs to reach an Interested Reachable Node considering and not considering the C-IORs.

Mean Number of Hops **PercAcces2POD=[1]** **PercNodeDiff=[1,1,1,1,1,1]**	**Source 47**	**Source 999**	**Source 71**
Considering C-IORs	1.2466	1.3315	1.1995
Not considering C-IORs	2.7068	2.6496	2.1482

We conclude this Section showing in Fig. 6.12 the comparison between the Mean Number of Hops that each Source employs to reach an Interested Reachable Node considering and not considering the C-IORs.

6.9 Final remarks

In conclusion, in all the conducted studies the advantage achieved in the Enhanced SIoT case compared to the Friendships case is evident, thanks to the possibility of using the SIoT relationships. This means that with the Enhanced SIoT we are able to reach a greater number of interested nodes.

Furthermore, the contribution given by the C-IOR relationship appears to be significant, making the Enhanced SIoT an advantageous solution in terms of mean number of hops compared to the case in which the C-IORs are not used. This means that the use of C-IORs allows a faster Discovery.

Chapter 7

Conclusions

In this thesis we have proposed a new platform model for Decentralized Online Social Networks (DOSNs) based on the joint use of the Solid platform and the new paradigm of Social Internet of Things (SIoT), emerging with increasing strength.

Evidence has been provided of the fact that by coupling these two concepts together it is possible to arrive at the design of a modern DOSN platform that permits users to maintain control over their personal information and, at the same time, effectively limits the intrinsic drawbacks that in the past made DOSNs unattractive compared to centralized solutions.

Through a simulation campaign aimed at comparing the ability to connect users with the same interest but belonging to separate communities within a DOSN platform, it was possible to prove that the road traced has the potential to make Distributed Social Networks more attractive and to facilitate their large-scale deployment. This can be achieved thanks to the synergies that can be obtained between human users and social devices.

References

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

[2] Adali, S., Escriva, R., Goldberg, M. K., Hayvanovych, M., Magdon-Ismail, M., Szymanski, B. K., Wallace, W. A., and Williams, G. (2010). Measuring behavioral trust in social networks. In *2010 IEEE International Conference on Intelligence and Security Informatics*, pages 150–152. IEEE.

[3] Aiello, L. M. and Ruffo, G. (2010). Secure and flexible framework for decentralized social network services. In *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 594–599. IEEE.

[4] Atzori, L., Campolo, C., Da, B., Girau, R., Iera, A., Morabito, G., and Quattropani, S. (2019). Smart devices in the social loops: Criteria and algorithms for the creation of the social links. *Future Generation Computer Systems*, 97:327–339.

[5] Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

[6] Atzori, L., Iera, A., and Morabito, G. (2011). Siot: Giving a social structure to the internet of things. *IEEE communications letters*, 15(11):1193–1195.

[7] Atzori, L., Iera, A., and Morabito, G. (2014). From" smart objects" to" social objects": The next evolutionary step of the internet of things. *IEEE Communications Magazine*, 52(1):97–105.

[8] Atzori, L., Iera, A., Morabito, G., and Nitti, M. (2012). The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization. *Computer networks*, 56(16):3594–3608.

[9] Babenko, A., Slesarev, A., Chigorin, A., and Lempitsky, V. (2014). Neural codes for image retrieval. In *European conference on computer vision*, pages 584–599. Springer.

[10] Bao, F., Chen, R., and Guo, J. (2013). Scalable, adaptive and survivable trust management for community of interest based internet of things systems. In *2013 IEEE eleventh international symposium on autonomous decentralized systems (ISADS)*, pages 1–7. IEEE.

[11] Birrane, E. and Soloff, J. (2020). *Designing Delay-Tolerant Applications for Store-and-Forward Networks*. Artech House.

[12] Boldrini, C., Conti, M., and Passarella, A. (2008). Contentplace: social-aware data dissemination in opportunistic networks. In *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 203–210.

[13] Buchegger, S. and Datta, A. (2009). A case for p2p infrastructure for social networks-opportunities & challenges. In *2009 Sixth International Conference on Wireless On-Demand Network Systems and Services*, pages 161–168. IEEE.

[14] Buchegger, S., Schiöberg, D., Vu, L.-H., and Datta, A. (2009). Peerson: P2p social networking: early experiences and insights. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pages 46–52.

[15] Can, A. B. and Bhargava, B. (2012). Sort: A self-organizing trust model for peer-to-peer systems. *IEEE transactions on dependable and secure computing*, 10(1):14–27.

[16] Cetina, K. K., Schatzki, T. R., and Von Savigny, E. (2005). *The practice turn in contemporary theory*. Routledge.

[17] Chen, R., Bao, F., and Guo, J. (2015). Trust-based service management for social internet of things systems. *IEEE transactions on dependable and secure computing*, 13(6):684–696.

[18] Chen, R., Guo, J., and Bao, F. (2014). Trust management for soa-based iot and its application to service composition. *IEEE Transactions on Services Computing*, 9(3):482–495.

[19] Chen, Z., Ling, R., Huang, C.-M., and Zhu, X. (2016). A scheme of access service recommendation for the social internet of things. *International Journal of Communication Systems*, 29(4):694–706.

[20] Cheng, Z., Caverlee, J., Lee, K., and Sui, D. Z. (2011). Exploring millions of footprints in location sharing services. In *Fifth International AAAI Conference on Weblogs and Social Media*.

[21] Cho, E., Myers, S. A., and Leskovec, J. (2011). Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090.

[22] Ciobanu, R.-I., Marin, R.-C., Dobre, C., Cristea, V., and Mavromoustakis, C. X. (2014). Onside: Socially-aware and interest-based dissemination in opportunistic networks. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–6. IEEE.

[23] City, A. S. (2012). Amsterdam smart city˜ projects.

[24] Conti, M., De Salve, A., Guidi, B., and Ricci, L. (2014). Epidemic diffusion of social updates in dunbar-based dosn. In *European Conference on Parallel Processing*, pages 311–322. Springer.

[25] Cutillo, L. A., Molva, R., and Strufe, T. (2009). Safebook: A privacy-preserving online social network leveraging on real-life trust. *IEEE Communications Magazine*, 47(12):94–101.

[26] Damiani, E., di Vimercati, S. D. C., Paraboschi, S., Pesenti, M., Samarati, P., and Zara, S. (2003). Fuzzy logic techniques for reputation management in anonymous peer-to-peer systems. In *EUSFLAT Conf.*, pages 43–48. Citeseer.

[27] Datta, A., Buchegger, S., Vu, L.-H., Strufe, T., and Rzadca, K. (2010). Decentralized online social networks. In *Handbook of social network technologies and applications*, pages 349–378. Springer.

[28] ElSalamouny, E., Sassone, V., and Nielsen, M. (2009). Hmm-based trust model. In *International Workshop on Formal Aspects in Security and Trust*, pages 21–35. Springer.

[29] Fiware (2018). Fiware data models.

[30] Gao, W. and Cao, G. (2011). User-centric data dissemination in disruption tolerant networks. In *2011 Proceedings IEEE INFOCOM*, pages 3119–3127. IEEE.

[31] Girau, R., Martis, S., and Atzori, L. (2016). Lysis: A platform for iot distributed applications over socially connected objects. *IEEE Internet of Things Journal*, 4(1):40–51.

[32] Giusto, D., Iera, A., Morabito, G., and Atzori, L. (2010). *The internet of things: 20th Tyrrhenian workshop on digital communications*. Springer Science & Business Media.

[33] Graffi, K. and Masinde, N. (2020). Libresocial: A peer-to-peer framework for online social networks. *arXiv preprint arXiv:2001.02962*.

[34] Guidi, B., Amft, T., De Salve, A., Graffi, K., and Ricci, L. (2016). Didusonet: A p2p architecture for distributed dunbar-based social networks. *Peer-to-Peer Networking and Applications*, 9(6):1177–1194.

[35] Guidi, B., Conti, M., Passarella, A., and Ricci, L. (2018). Managing social contents in decentralized online social networks: a survey. *Online Social Networks and Media*, 7:12–29.

[36] Guidi, B., Michienzi, A., and Rossetti, G. (2019). Towards the dynamic community discovery in decentralized online social networks. *Journal of Grid Computing*, 17(1):23–44.

[37] Index, G. W. (2017). Gwi device q1 2017. Available from https://cdn2.hubspot.net/hubfs/304927/Downloads/GWI-Device-Q1-2017-Summary.pdf.

[38] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678.

[39] Josang, A. (1999). Trust-based decision making for electronic transactions. In *Proceedings of the Fourth Nordic Workshop on Secure Computer Systems (NORDSEC'99)*, pages 496–502.

[40] Jøsang, A. (2001). A logic for uncertain probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(03):279–311.

[41] Jøsang, A. (2007). Trust and reputation systems. In *Foundations of security analysis and design IV*, pages 209–245. Springer.

[42] Josang, A., Hayward, R. F., and Pope, S. (2006). Trust network analysis with subjective logic.

[43] Josang, A. and Ismail, R. (2002). The beta reputation system. In *Proceedings of the 15th bled electronic commerce conference*, volume 5, pages 2502–2511.

[44] Jøsang, A., Ismail, R., and Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision support systems*, 43(2):618–644.

[45] Jøsang, A. and Pope, S. (2005). Semantic constraints for trust transitivity. In *Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling-Volume 43*, pages 59–68.

[46] Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. (2003). The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651.

[Keras] Keras. Keras.

[48] Kong, C. (2016). Content dissemination in mobile social networks.

[49] Kong, C. and Cao, X. (2014). Semi-controlled authorized information dissemination in content-based social networks. In *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6. IEEE.

[50] Kosta, S., Mei, A., and Stefa, J. (2010). Small world in motion (swim): Modeling communities in ad-hoc mobile networking. In *2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 1–9. IEEE.

[51] Leguay, J., Lindgren, A., Scott, J., Riedman, T., Crowcroft, J., and Hui, P. (2006). Crawdad trace upmc/content/imote/cambridge (v. 2006–11–17).

[Leskovec] Leskovec. Stanford large network dataset collection.

[53] Lew, M. S., Sebe, N., Djeraba, C., and Jain, R. (2006). Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2(1):1–19.

[54] Lin, Z. and Dong, L. (2017). Clarifying trust in social internet of things. *IEEE Transactions on Knowledge and Data Engineering*, 30(2):234–248.

[55] Louisville (2018). Louisville open data portal.

[56] Mansour, E., Sambra, A. V., Hawke, S., Zereba, M., Capadisli, S., Ghanem, A., Aboulnaga, A., and Berners-Lee, T. (2016). A demonstration of the solid platform for social web applications. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 223–226.

[57] Marche, C., Atzori, L., Iera, A., Militano, L., and Nitti, M. (2017). Navigability in social networks of objects: The importance of friendship type and nodes' distance. In *2017 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE.

[58] Marche, C., Atzori, L., and Nitti, M. (2018). A dataset for performance analysis of the social internet of things. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–5. IEEE.

[59] Mashhadi, A. J., Mokhtar, S. B., and Capra, L. (2009). Habit: Leveraging human mobility and social network for efficient content dissemination in delay tolerant networks. In *2009 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks & Workshops*, pages 1–6. IEEE.

[60] Mega, G., Montresor, A., and Picco, G. P. (2012). On churn and communication delays in social overlays. In *2012 IEEE 12th International Conference on Peer-to-Peer Computing (P2P)*, pages 214–224. IEEE.

[61] Mega, G., Montresor, A., and Picco, G. P. (2018). Social overlays meet the cloud: A hybrid architecture for profile dissemination in decentralized social networks. *IEEE Transactions on Network Science and Engineering*, 6(4):613–627.

[62] Meng, X., Li, T., and Deng, Y. (2016). Prefertrust: An ordered preferences-based trust model in peer-to-peer networks. *Journal of Systems and Software*, 113:309–323.

[ML] ML, C. Core ml.

[MNIST] MNIST. Mnist handwritten digit database.

[65] Mui, L., Mohtashemi, M., and Ang, C. (2001a). A probabilistic rating framework for pervasive computing environments. In *Proceedings of the MIT Student Oxygen Workshop (SOW'2001)*.

[66] Mui, L., Mohtashemi, M., Ang, C., Szolovits, P., and Halberstadt, A. (2001b). Ratings in distributed systems: A bayesian approach. In *Proceedings of the Workshop on Information Technologies and Systems (WITS)*, pages 1–7.

[67] Mui, L., Mohtashemi, M., and Halberstadt, A. (2002). A computational model of trust and reputation. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pages 2431–2439. IEEE.

[68] Nitti, M., Atzori, L., and Cvijikj, I. P. (2014a). Friendship selection in the social internet of things: challenges and possible strategies. *IEEE Internet of things journal*, 2(3):240–247.

[69] Nitti, M., Atzori, L., and Cvijikj, I. P. (2014b). Network navigability in the social internet of things. In *2014 IEEE world forum on internet of things (WF-IoT)*, pages 405–410. IEEE.

[70] Nitti, M., Girau, R., and Atzori, L. (2013). Trustworthiness management in the social internet of things. *IEEE Transactions on knowledge and data engineering*, 26(5):1253–1266.

[71] Nitti, M., Girau, R., Atzori, L., Iera, A., and Morabito, G. (2012). A subjective model for trustworthiness evaluation in the social internet of things. In *2012 IEEE 23rd international symposium on personal, indoor and mobile radio communications-(PIMRC)*, pages 18–23. IEEE.

[72] Njoo, G. S., Hsu, K.-W., and Peng, W.-C. (2018). Distinguishing friends from strangers in location-based social networks using co-location. *Pervasive and Mobile Computing*, 50:114–123.

[73] Nordström, E., Rohner, C., and Gunningberg, P. (2014). Haggle: Opportunistic mobile content sharing using search. *Computer Communications*, 48:121–132.

[74] Paul, T., Buchegger, S., and Strufe, T. (2011). Decentralized social networking services. In *Trustworthy Internet*, pages 187–199. Springer.

[PyTorch] PyTorch. Pytorch.

[76] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

[77] Sacramento (2018). Sacramento open data portal.

[78] Sambra, A. V., Mansour, E., Hawke, S., Zereba, M., Greco, N., Ghanem, A., Zagidulin, D., Aboulnaga, A., and Berners-Lee, T. (2016). Solid: a platform for decentralized social applications based on linked data. Technical report, Technical report, MIT CSAIL & Qatar Computing Research Institute.

[79] Sanchez, L., Muñoz, L., Galache, J. A., Sotres, P., Santana, J. R., Gutierrez, V., Ramdhany, R., Gluhak, A., Krco, S., Theodoridis, E., et al. (2014). Smartsantander: Iot experimentation over a smart city testbed. *Computer Networks*, 61:217–238.

[80] Scellato, S., Noulas, A., Lambiotte, R., and Mascolo, C. (2011). Socio-spatial properties of online location-based social networks. In *Fifth international AAAI conference on weblogs and social media*.

[81] Schurgot, M. R., Comaniciu, C., and Jaffres-Runser, K. (2012). Beyond traditional dtn routing: social networks for opportunistic communication. *IEEE Communications Magazine*, 50(7):155–162.

[82] Sehgal, A. and Kehtarnavaz, N. (2019). Guidelines and benchmarks for deployment of deep learning models on smartphones as real-time apps. *Machine Learning and Knowledge Extraction*, 1(1):450–465.

[83] Sherchan, W., Nepal, S., and Paris, C. (2013). A survey of trust in social networks. *ACM Computing Surveys (CSUR)*, 45(4):1–33.

[84] Song, W., Phoha, V. V., and Xu, X. (2004). The hmm-based model for evaluating recommender's reputation. In *IEEE International Conference on E-Commerce Technology for Dynamic E-Business*, pages 209–215. IEEE.

[85] Tie-li, S., Jing-wei, D., and Kai-ying, D. (2008). Scale-free network model with evolving local-world. In *2008 Fourth International Conference on Natural Computation*, volume 1, pages 237–240. IEEE.

[Toolkit] Toolkit, M. C. Microsoft cognitive toolkit.

[87] Wan, J., Wang, D., Hoi, S. C. H., Wu, P., Zhu, J., Zhang, Y., and Li, J. (2014). Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 157–166.

[88] Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., and Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393.

[89] Webster, M. (2007). Merriam-webster online. Available from http://www.m-w.com/, accessed June 2007.

[90] Xiong, L. and Liu, L. (2004). Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE transactions on Knowledge and Data Engineering*, 16(7):843–857.

[91] Yang, D., Qu, B., Yang, J., and Cudre-Mauroux, P. (2019). Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *The World Wide Web Conference*, pages 2147–2157.

[92] Yang, D., Zhang, D., Chen, L., and Qu, B. (2015). Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns. *Journal of Network and Computer Applications*, 55:170–180.

[93] Yang, D., Zhang, D., and Qu, B. (2016). Participatory cultural mapping based on collective behavior data in location-based social networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(3):1–23.

[94] Yeung, C.-m. A., Liccardi, I., Lu, K., Seneviratne, O., and Berners-Lee, T. (2009). Decentralization: The future of online social networking. In *W3C Workshop on the Future of Social Networking Position Papers*, volume 2, pages 2–7.

[95] Yuan, B., Liu, L., and Antonopoulos, N. (2018). Efficient service discovery in decentralized online social networks. *Future Generation Computer Systems*, 86:775–791.

[96] Zhitomirskiy, I., Grippi, D., Salzberg, M., and Sofaer, R. (2010). Diaspora. Available from https://diasporafoundation.org/.

[97] Zhou, J., Albatal, R., and Gurrin, C. (2016). Applying visual user interest profiles for recommendation and personalisation. In *International Conference on Multimedia Modeling*, pages 361–366. Springer.