DOCTORAL SCHOOL
MEDITERRANEA UNIVERSITY OF REGGIO CALABRIA

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

PHD IN
INFORMATION ENGINEERING

S.S.D. ING-INF/05
XXXIII CYCLE

TRUST, PRIVACY AND DATA MODELLING IN THE SMART CITY DOMAIN

CANDIDATE
LORENZO MUSARELLA

ADVISORS
[illegible] Prof. FRANCESCO BUCCAFURRI
Prof. GIANLUCA LAX
[illegible]

COORDINATOR
Prof. TOMMASO ISERNIA
[illegible]

REGGIO CALABRIA, JANUARY 2021

Finito di stampare nel mese di **Gennaio 2021**

Edizione Centro Stampa d'Ateneo

Quaderno N. 50

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

ISBN 978-88-99352-41-7

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

LORENZO MUSARELLA

TRUST, PRIVACY AND DATA MODELLING IN THE SMART CITY DOMAIN

Contents

1 Introduction . . . 1
  1.1 Scenario and motivation . . . 1
  1.2 The Blockchain technology . . . 3
  1.3 Exploit generated data . . . 5
    1.3.1 Smart Objects . . . 5
    1.3.2 Data Lakes . . . 6
  1.4 Smart Communities . . . 8
  1.5 Outline of the Thesis . . . 9

Part I Blockchains for smart and trusted interactions

2 Background . . . 15
  2.1 Blockchain and Distributed Ledger Technologies . . . 15
  2.2 Ethereum . . . 19
    2.2.1 Tokens . . . 21
  2.3 Smart City . . . 22
    2.3.1 Smart Grids . . . 23

3 A new architecture for energy trading in smart grids . . . 27
  3.1 Introduction . . . 27
    3.1.1 Scenario and motivation . . . 28
  3.2 The architecture of our proposal . . . 30
  3.3 Implementation . . . 33
  3.4 Security aspects . . . 37
    3.4.1 Discussion on security vulnerabilities of smart meters . . . 40
  3.5 Related work . . . 41
    3.5.1 Comparison . . . 45

4 A new solution for service delivery with accountability and privacy requirements ... 47
4.1 Introduction ... 47
4.2 The proposed solution ... 49
4.2.1 Case study ... 52
4.3 Implementation details ... 55
4.4 Security aspects ... 57
4.5 Related work ... 58

5 A new data access control mechanism based on smart contracts ... 61
5.1 Introduction ... 61
5.1.1 Attribute-Based Access Control ... 62
5.1.2 InterPlanetary File System ... 62
5.2 Scenario and motivation ... 63
5.3 The proposed access control mechanism ... 64
5.4 Security aspects ... 68
5.5 Implementation issues ... 69
5.6 Related work ... 72

6 A proposal to enable propagation in Web-of-Trust ... 75
6.1 Introduction ... 75
6.1.1 Web of Trust ... 76
6.2 Description of our proposal ... 79
6.3 Implementation issues ... 84
6.4 Future Works ... 87

7 An approach to enabling Ethereum transactions among secure identities 89
7.1 Introduction ... 89
7.1.1 Digital identity and IBE ... 90
7.2 Our proposal ... 92
7.3 Implementation ... 95
7.4 Related work ... 98

Part II Exploiting generated data

8 A new model to represent smart objects in multiple environments ... 105
8.1 Introduction ... 105
8.2 The MIoT paradigm ... 108
8.2.1 An example of an MIoT ... 113
8.2.2 Why using the MIoT paradigm? ... 116

8.3 CDS: a crawler tailored for MIoTs 119
8.3.1 Motivations underlying CDS 119
8.3.2 Description of CDS 120
8.3.3 Experimental campaign 124
8.4 Analytical discussion 131
8.5 Related work 133

9 A proposal to uniformly handle heterogeneous metadata sources 137
9.1 Introduction 137
9.1.1 Typologies of metadata 140
9.2 A network-based model for uniformly represent heterogeneous sources 141
9.3 Providing a structure to unstructured sources 143
9.3.1 Running example 147
9.3.2 Evaluation 150
9.4 Related work 151

10 A new approach to extracting inter-schema properties from heterogeneous sources 155
10.1 Introduction 155
10.2 Extracting inter-schema properties from heterogeneous sources 157
10.2.1 Semantic similarity degree computation 158
10.2.2 Semantic relationship detection 161
10.3 Experiments 167
10.3.1 Overall performances 167
10.3.2 Evaluation of the pros and the cons 169
10.3.3 Discussion on the scalability 172
10.4 Related Work 174

11 An approach to extracting thematic views from highly heterogeneous sources 177
11.1 Introduction 177
11.2 An approach to extracting thematic views 178
11.2.1 An example case 181
11.3 Related work 183

12 An approach to extracting complex knowledge patterns among concepts belonging to heterogeneous sources 187
12.1 Introduction 187
12.2 Extraction of complex knowledge patterns 189

12.2.1 General description of the approach 189
12.2.2 Technical Details 191
12.3 Some case studies 193
12.4 Discussion 199
12.4.1 Comparison 199
12.4.2 Performance of our overall approach 201
12.4.3 Efficiency for large data sets 202
12.5 Related work 205

Part III Social Networks, Participation and Analysis

13 An approach to contrast fake identities in Social Networks 213
13.1 Introduction 213
13.2 Overview of the approach 215
13.3 Security requirements 216
13.4 The Trust Graph 218
13.5 Decentralized Computation of Trust 221
13.5.1 Optimization 223
13.6 Datasets 223
13.7 Implementation issues 224
13.7.1 Experimental results 225
13.8 Related work 227
13.9 Conclusion and perspectives 229

14 A novel query language for data extraction from multiple social networks231
14.1 Introduction 231
14.2 The basic data model 232
14.3 Query language for data extraction 234
14.4 Using the query language 238
14.5 Related Work 238

Part IV Conclusions

References 245

List of Figures

2.1 Centralized vs decentralized vs distributed networks 16
2.2 Concatenations of blocks in blockchains 17
2.3 Classification of blockchains 19

3.1 The architecture of our proposal. 30

4.1 Scenario and steps of our solution. 53

5.1 Portion of the code of App.js 71

6.1 Signatures in Web of Trust 77
6.2 Attributes of the user 81
6.3 Store schema 86

7.1 Operations carried out in an IBE scheme. 92
7.2 Data flow in an authentication process. 94

8.1 Schematic representation of the proposed MIoT structure 109
8.2 Distribution of the number of connected components of the instances of our MIoT against distances 116
8.3 Graphical representation of our MIoT 116
8.4 Our case study 117
8.5 Trends of the number of seen nodes, visited nodes, IoT crossings and visited IoTs against the number of iterations performed by CDS (trends are separated in the first two graphs and put together in the last one) 125
8.6 Our Metric Dependency Graph 129

9.1 The three kinds of metadata proposed by our model. 141
9.2 Graphical representation of our approach to derive a “structure” for an unstructured source 148

10.1 Representation, in our network-based model, of the unstructured source of our interest 164
10.2 Structure of the JSON file associated with the semi-structured source of our interest 165
10.3 Representation, in our network-based model, of the semi-structured source of our interest 165
10.4 Distribution, in a semi-logarithmic scale, of the values of the the semantic similarity degrees of the objects belonging to the two sources of interest 166
10.5 Computation time of XIKE and our approach against the number of concepts to process 171
10.6 Computation time of DIKE, XIKE ($u = 5$ and $u = 2$) and our approach against the number of concepts to process 174

11.1 Network-based representations of the four sources into consideration. . 182
11.2 Ego networks corresponding to *V.Ocean*, *C.Sea*, *W.Place*, *C.Place*, *V.Region* and *E.Location*. 182
11.3 Ego networks corresponding to *Ocean* and *Area*. 183
11.4 The integrated thematic view. 184

12.1 The network corresponding to the source *Climate* 194
12.2 The network corresponding to the source *Energy* 195
12.3 The network corresponding to the source *Environment disasters* 195
12.4 Complex knowledge pattern from the node `Energy` to the node `Population` of the source *Energy* 197
12.5 Complex knowledge pattern from the node `Position` of the source *Environment disasters* to the node `Energy` of the source *Energy* 198
12.6 Complex knowledge pattern from the node `Fujita_scale` of the source *Environment disasters* to the node `Risk` of the source *Environment risks* 198
12.7 Complex knowledge pattern from the node `Risk_degree` of the source *Environment disasters* to the node `Risk` of the source *Environment risks* 199
12.8 Real and theoretical response time against data lake dimension and density 202
12.9 Real and theoretical response time against data lake dimension and density (zoom of Figure 12.8) 202
12.10 Real and theoretical response time against dimension and density for large data lakes (Scenario 1) 203

12.11 Real and theoretical response time against dimension and density for large data lakes (zoom of Figure 12.10) 204

12.12 Real and theoretical response time against dimension and density for large data lakes (Scenario 2) 204

12.13 Real and theoretical response time against data lake dimension and density for large data lakes (zoom of Figure 12.12). 205

14.1 Examples of usage of our query language: (a) content assist enabled; (b) wrong query (c) query for Facebook; (d) query for Instagram. 237

Listings

3.1 Application of the Solidity `modifier` in our smart contract 34
3.2 Creation of a new auction 35
3.3 Functions sendBlindBid() and sendBlind() 35
3.4 Ending of the auction and computation of the winner prosumer 35
4.1 The code of the smart contract 56
5.1 Code of the smart contract `SimpleStorage` used to publish on IPFS . . . 70
5.2 Code of the mapping between the document and its policy 70
5.3 Code of the smart contract query with parser XML 71
6.1 Function `giveTrust()` 84
6.2 Modifier `onlyNotRevoked` 84
6.3 Function `removeTrust()` 84
6.4 Declaration of the function `verify()` 85
6.5 Declaration of the function `trust_satisfied()` 85
7.1 The code of the smart contract. 97
9.1 Representation of the model via a JSON like template 143

List of Tables

3.1 Costs of the deployment of the Smart Contract and of the functions . . . 37
3.2 Comparison between existing solutions and ours. 45

4.1 Notation. 50

6.1 Costs of the deployment of the Smart Contract and its functions 87

8.1 Number of instances present in the IoTs of our MIoT 115
8.2 Betweenneess Centrality, Degree Centrality, Closeness Centrality and Eigenvector Centrality, and the corresponding ranks, for all the nodes of the case study of Figure 8.4 118
8.3 Number of seen nodes, number of visited nodes, number of IoT crossings and number of visited IoTs against the number of iterations performed by CDS 124
8.4 Number of seen nodes, visited nodes, IoT crossings and visited IoTs against the variation of inf and cnf 126
8.5 Values of the five metrics obtained by CDS, BFS, RW and MH 130
8.6 Values of OCQ obtained by CDS, BFS, RW and MH for the two weight configurations into examination 131

9.1 Precision, Recall, F-Measure and Overall of our approach when a cluster-based approach for structuring unstructured sources is applied 150

10.1 Keywords of the unstructured source of our interest 164
10.2 Derived Synonymies 164
10.3 Derived Type Conflicts 166
10.4 Derived Overlappings 166
10.5 Precision, Recall, F-Measure and Overall of our approach 169
10.6 Characteristics of the sources exploited for evaluating our approach . . 170
10.7 Precision, Recall, F-Measure and Overall of XIKE and our approach . . . 171

10.8 Precision, Recall, F-Measure and Overall of DIKE, XIKE ($u = 5$, $u = 2$) and our approach 173

12.1 Keywords of the source *Environment risks* 196
12.2 Keywords of the source *Air pollution* 196

13.1 Properties of the datasets used 224
13.2 Percentage of coverage in $D1$ as t_level and number of roots vary 226
13.3 Percentage of coverage in $D2$ as t_level and number of roots vary 227
13.4 Percentage of coverage in $D3$ as t_level and number of roots vary 227
13.5 Percentage of coverage in $D4$ as t_level and number of roots vary 227
13.6 Percentage of coverage in $D5$ as t_level and number of roots vary 228
13.7 Percentage of coverage in $D6$ as t_level and number of roots vary 228

1 Introduction

1.1 Scenario and motivation

In this era of great technological evolution and innovation, the daily life of people is quickly moving into the digital world and domain. Indeed, technology is involved in many daily operations that a person carries out. Just consider this last year of pandemic, in which many daily tasks were possible mainly thanks to ICT technologies that let people continuing working through the *smart working*. Online social communities and interactions represent a fact in modern days. However, the pandemic just accelerates the evolution from the *analog* world into the *digital* one. Indeed, we were living in this transition period well before the world-wide spreading of the virus.

Smart objects, IoTs, smartphones, smartwatches and laptops are only a subset of devices that we use every day to carry out many different activities: we can say that almost everything is now connected and brought into the digital domain.

This situation, in which data travels via internet through servers and services, opens new opportunities and challenges both for researchers and scientists but also for malicious people. Indeed, exposing information in the internet requires solutions, protocols and infrastructures to be secure, reliable and trusted, other than fast and user-friendly. We can not accepts trade-off in these situations: security, privacy and trust are properties on which every solution (both in the digital and the analog domain) should be built.

E-participation is now imposing as the new way to communicate, interact and sharing information among people. Moreover, services provided by governmental entities and companies are moving rapidly into the digital domain. On one hand we meet people and citizens that ask for new services and new levels of efficiency and reliability in the interactions between them and the city and, on the other hand, we have cities that are facing big problems in terms of scalability and security because of using legacy systems and also because of the global urbanization phenomenon: many citizens are moving from their native places to settle into big cities [282] saturating resources available. These two main situations are making cities literally moving from a classical interpretation to a new paradigm called *Smart City*.

In recent years energy consumption increased exponentially; the environment is hardly challenged; healthcare and education systems are demanding new approaches; public safety is challenged as well; and the potential for future cyberattacks against cities is dramatically high. Without innovative solutions, this situation can lead to further environmental degradation and poverty. It is necessary to rethink the models of access to resources, transport, waste management, and energy management [209]. Hence, smart, cost-effective, scalable, secure and innovative solutions that can address the problems of urbanization are needed [210].

Despite there is some kind of consensus that the label smart city as innovation in city management, its services and infrastructures, a common definition of the term has not yet been stated. There is a wide variety of definitions of what a smart city is [266].

This paradigm involves almost every area of research, from engineering to architecture, from social sciences to economics, etc. [63]. Smart cities aim at evolving everything inside a city into something smarter, more eco-friendly and sustainable so that reducing pollution, costs for citizens and improving services. Clearly, this goal is very ambitious but researchers are working hard in recent years to propose solutions able to reach it. Among others, cybersecurity is one of the area more affected by this new paradigm [210].

The concept of the smart city emerged during the last decades as a fusion of ideas about how information and communications technologies might improve the functioning of cities, enhancing their efficiency, improving their competitiveness and services, and providing new ways in which problems of poverty, social deprivation, and poor environment might be addressed. Intelligent cities, virtual cities, digital cities, information cities are all perspectives on the idea that ICT is central to the operation of the future city [69].

A big component in the smart city strategy is represented by *Smart Grids*. We can define the smart grid as the successor of the classical electric grids, having the main purpose to design new solutions that are more reliable, faster and more secure.

Smart grid technology is changing the way traditional power grids operate by reducing energy demands, global warming and consequently, utility costs. Consumers are required to share information about their energy consumption with their utility providers, over communication channels using smart meters [63]. The main functionalities of smart grids start with modernizing power systems by means of real-time monitoring, automation and self-controlling issues [149]. Smart meters play a crucial role in this sense. They are smart devices that connect the smart home of the citizen to the smart grid infrastructure by taking into account different parameters (amount of energy required, amount of energy consumed, etc.).

Moreover, smart meters are able to guarantee a real-time communication network connecting the grid with electricity providers and consumers. Smart grids use this communication network to collect demands from consumers and to reply to them by optimizing resources available on the grid [329, 177, 383]. Moreover, smart grids have some internal features, such as self-healing and the ability for fault detection, which allow them to continue providing the energy flow to the grid. Another challenge that the new smart grid paradigm wants to address regards the energy market mechanism, both in terms of production of new renewable energy and the energy trading aspect. Indeed, smart grids want to make closer all steps associated with the energy. For this scope, the *prosumer* actor is now involved in the scenario. Briefly, we can define a prosumer as a participant that both produce and consume energy and electricity in the network.

In this thesis, we propose new solutions and models that refer to the digital aspect of the smart city paradigm and that implement the concept of *security-by-design*. Indeed, security has been considered too much in past just as a patch implemented after the realization of the system. New solutions in the new paradigm of smart city must, instead, provide security properties already in the design phase. It would be unacceptable and very risky to expose our daily life information and actions into not secure systems.

1.2 The Blockchain technology

Blockchain technology is a hot topic for researchers of the very last years. It is part of the *Distributed Ledger Technologies* DLTs since it implements a distributed way of manage transactions between participants. There is no any central authority or central power and we can be see blockchain as a *P2P* network in which validity of transactions and some other fundamental properties are guaranteed by a particular mechanism in which every transaction is grouped in blocks that are linked each others through some security hashes.

Among all properties this technology provides, we remark *immutability, transparency, traceability* and *security*, that are achieved by implementing cryptographic protocols to add and link transactions in the network.

At the beginning, blockchain technology was quite misunderstood because people often confused or overlapped it with the *Bitcoin* [10], although this last was *only* an application based on the blockchain technology. In this sense, some people saw it as a new vehicle for malicious users since it provides anonymity in transactions while some other people saw the blockchain as a business source because of the mining process (that we will better explain in the thesis) and the opportunities to gain crypto-money by mining blocks and transactions.

However, researchers and scientists soon understood that this new technology could represent a big source for new applications, solutions and methods. Indeed, blockchain is not just a technology able to support cryptocurrencies (as happened in Bitcoin), but it is much more.

In particular, blockchain-based technologies are used in cases where there is need to reach an agreement between two parties and they do not want to use a trusted third party. The blockchain technology plays the role of an impartial referee, since it is required that the majority of the peers of the network agree to validate a new transaction. Moreover, this technology evolved to what we now call *Blockchain 2.0*.

The progenitor of this new family of blockchains is clearly *Ethereum* [20], a public permissionless platform created by Vitalik Buterin having as main goal to represent a platform able to manage and run decentralized applications (dApps) in a secure, reliable and fast way [19].

Ethereum presents some new very interesting features that makes this new technology more flexible than others. In particular, the introduction of two main concepts like *Smart Contracts* and *Tokens*, among others, let Ethereum have a worldwide success. A smart contract is a real actor and participant in the network that can be easily developed and published by users thanks to the high-level language called *Solidity*. Smart contracts are, indeed, pieces of code that follow the rule *if this, then that*. They run code and returns outputs and every peer of the network can execute them so that it is guaranteed the correctness of the execution. We can define Tokens as digital assets that have no value until they are contextualized in a certain domain. By the combination of these two features, this new platform let the design and implementation of many different applications still guaranteeing all the security properties of the blockchain technology [344].

In Ethereum every task and operation the user wants to carry out must be paid in terms of ether (that is the cryptocurrency, the fuel of the whole platform). For this reason, the smart contract should be as simple and fast as possible.

In the first part of this thesis we exploit Ethereum and the blockchain technology to design and proposes new architectures and solutions to address some of the challenges that smart cities open.

First, we address the problem of energy trading in a smart grid scenario. As discussed above, energy management is a key factor in the smart city paradigm because of urbanization and because of environmental problems. We propose a solution based on a Ethereum smart contract and tokens that can provide a new way for consumers, prosumers and retailers to exchange and trade electricity easier and faster via a blind auction where the smart contract is in charge to compute the best scenario for each situation.

Another scenario we investigate regards service delivery with accountability and privacy requirements. It is worth highlight that the smart city paradigm wants to enhance quality and efficiency of services provided to citizens. Many solutions that are implemented and used require the customer to register into the service supplier platform and give them its personal data to retrieve the asked service. However, there are several cases in which it is not necessary to disclose information about the customer's identity to have back the service but it would be just sufficient to demonstrate the possession of some requirements (attributes, licences, etc.). Furthermore, the customer can not be always sure about the trustworthiness of service providers because they are quite often third parties. For these reasons ee propose a new approach privacy-preserving that let service delivery without disclosing additional and not necessary information to the service provider. IAlso, we still manage and guarantee accountability of operations.

Smart cities are characterized by a huge amount of data. It is produced by every device and it represents, at the same time, a source of information but also a source for attacks. For this reason, smart cities provide two main different typologies of data: *open data* and *closed data* [114]. If for the former it is easy to understand that everybody can read and access it, for the latter there is necessary to spend some attention. Access control plays a fundamental role for closed data. The problem is not trivial, since we deal with a highly open and dynamic environment, and, at the same time, that a certain level of accountability should be guaranteed to contrast misbehaviour and solve possible legal controversies. We propose a new attribute-based access control mechanism based on a Ethereum smart contract in which consumers must fulfil a certain policy to gain access to the information. We also implement the solution using the InterPlanetary FileSystem as data storage.

1.3 Exploit generated data

1.3.1 Smart Objects

Internet of Things is another attractive and fashion topic of interest because it is currently considered the new frontier of the Internet, and a lot of research results about this topic can be found in literature. Among all, a very interesting proposal regards the application of the social network paradigm into IoTs, creating what is known as Social Internet of Things (SIoT) [57]. This paradigm introduces the concept of social relationships among smart objects so that things are empowered with social skills, making them more similar to people [58].

In literature, social network researchers have introduced new paradigms capable of capturing the growing complexity of this scenario. Social Internetworking System is one of these paradigm, which models a scenario comprising multiple related social networks. We investigate the possibility of applying this ideas to SIoT, and we propose a new paradigm that we call MIoT (Multiple Internets of Things), capable of modelling and handling the increasing complexity of this last context. MIoT can be seen as an evolution of SIoT. In SIoT smart objects can be linked by five kinds of relationship, namely: (i) parental object relationship; (ii) co-location object relationship; (iii) co-work object relation- ship; (iv) ownership object relationship; (v) social object relationship. If: (i) a node is associated with each thing, (ii) an edge is associated with each relationship between things, and, finally, (iii) all the nodes and the edges linked by the same relationship are seen as joined together, SIoT can be modeled as a set of five pre-defined networks. Here, some nodes belong to only one network (we call them inner-nodes), whereas other ones belong to more networks (we call them cross-nodes).

We model a MIoT as a set of things connected each other by relationships of any typology and, at the same time, as a set of related IoTs, one for each kind of relationship. Every smart object can join different IoTs because of the creation of its different instances. The nodes of each IoT represent the instances of the things participating to it. As a consequence, a thing can have several instances, one for each IoT to which it participates. This new way of thinking smart objects open new scenarios and challenges. For example, classical centrality measures or classical crawler are not anymore valid in this new paradigm.

1.3.2 Data Lakes

The widespread diffusion of new smart objects, smart devices and the technological progress Orof modern days let increase dramatically the amount and the variety of data produced, bringing to the concept of *big data* . Everything that surrounds us produce now data in each moment: smartphones, sensors, IoT devices, PCs, etc. The $DIKW$ pyramid (Data, Information, Knowledge, Wisdom) explains clearly how important is to collect and manage in a correct way data to support decision making. Briefly, in the bottom of the pyramid we find data, which becomes information if we contextualize it; from information we can retrieve knowledge if we are able to give it meaning. Finally, knowledge becomes wisdom if we understand it. Climbing in these levels we get more and more value from data.

In this scenario it is relevant also to have a comprehension of what data is now. In fact, if many years ago data was mostly structured and processed by the data-warehouse paradigm, now unstructured and semi-structured data are exploding in number. The data-warehouse solution does not perfectly handle semi-structured data and it has even more difficulty for unstructured one because of its rigid rules and complexity. New flexible and agile architectures are necessary to solve this situation. These requirements have been addressed by the new paradigm of *Data Lake*.

Data Lakes is a term introduced in 2010 by James Dixon (CTO at Pentaho, an American Business Intelligence company) [4] as a storage repository that implements a flat architecture so that the insertion and the removal of a source can be easily performed .

Soon big companies and researchers have been fascinated by the innovative idea proposed by Dixon to collect data in its native and raw format and not to transform and process the data (as happens during the process of data ingestion in data warehousing). This ambitious project could be implemented only with the sustain of adequate policies of data governance. Indeed, the main risk of data lakes is to transform this huge source of information and knowledge in a data swamp, that is something not useful and difficult to navigate for data scientists and that can not help and support the decision making process.

As suggested by many researchers and companies [280, 165], metadata plays a crucial role in the management of a data lake because queries and navigations over the data lake can benefit from metadata information: approaches and solutions will be more agile and flexible if the system processes metadata instead of data. For this reason, the main data lake companies are performing several efforts in the direction of proposing solutions to manage data lakes. For instance, Zaloni, one of the market leaders in the data lake field, proposed a metadata classification [280]). For this reason, the definition of new models and paradigms for metadata representation and management represents an open problem in the data lake research field.

Starting from the metadata classification proposed by Zaloni, we developed and implemented a network-based model that is able to handle uniformly many different and heterogeneous data lake sources. In particular, the most relevant challenge we face is to model and propose a way to *structure* unstructured and semi-structured sources. The model we propose is able to reach this goal and it helps us to address other important challenge of the data lake management and navigation.

Indeed, this model represents the basis on which we investigate and propose other ideas and studies. First, we address the problem of schema integration by proposing a model that is able to extract semantic relationships as synonymies, homonymies, overlappings and type conflicts from different data lake sources through interschema properties detections.

Another challenge we address regards the navigation of a data lake. As said, it is important to navigate the data lake because it is possible to gathering precious information and knowledge. For this reason we propose two solutions for the topic-driven extraction of thematic views and complex knowledge patterns from heterogeneous sources.

In both cases we exploit our previous proposals of metadata representation and schema integration, but they both can be considered stand alone and they can work with other models and solutions.

1.4 Smart Communities

Social networks represent the main vehicle for information sharing and interactions among users. Indeed, in daily life, communications are taking rapidly the direction of the digital and the virtual domain. The smart city paradigm also contribute to increasing the number of online interactions. E-participation plays clearly a key role in the everyday operations. We are almost dependent from social media and social networks; they are sources of information sharing, platforms where people can interact each other and where also trash news and data are collected and shared. For this reason, trust represents a fundamental property that users look for when they interact and use their social networks. In particular, we can face fake social network profiles and fake news. Usually, fake news are shared and forwarded mostly by fake profiles. This situation motive us to propose a new model that is able to detect if or not a given social profile can be considered trusted.

Social network profiles whose claimed identity does not match with the real user are certainly potential security threats in the Web [278]. This happens in two cases. The first case is that of fake profiles, in which the owner of a profile intentionally claims the real-life identity of another individual.

The second case is that of violated profiles, in which an intruder, permanently or temporarily, uses the profile of a victim in a fraudulent way.

In both cases, the risk of anomalous behaviour with potential damage of the victim reputation, espionage, or social engineering attacks towards people connected to the victim is very high. To give an example, according to security firm Symantec [7], a growing number of hackers are targeting professionals on LinkedIn. Through these connections, attackers can entice users to give up personal data, hijack them towards infected websites and, once their email addresses is known, launch spear-phishing campaigns.

The problem has thus a high practical relevance. Several studies have been proposed in the recent literature [170, 116] to contrast this problem. However, all the existing proposals require a strong effort of analysis done centrally by the social network provider, which takes into account all the behavioural and topological information of the profiles.

We think there is need to a decentralized approach where collaboration among peers of the network is the key to isolate fake profiles from the communication network.

We address this situation by proposing a decentralized model that is able to compute the trust level of each social profile so determining if it can be considered trusted from other users. This study is in collaboration with the French research group from ENSICAEN University, led by Prof. Rosenberger, that is in charge to enforce this model by using keystroke dynamics as biometric feature.

Another situation we investigate regards data extraction from Online Social Networks (OSNs). Many people create and manage more than one social profile in the different available OSNs. The combination and the data extraction contained in OSNs can produce a huge amount of additional information regarding both a single person and the overall society. Consequently, the data extraction from multiple social networks is a topic of growing interest. There are many techniques and technologies for data extraction from a single OSN, but there is a lack of simple query languages which can be used by programmers to retrieve data, correlate resources and integrate results from multiple OSNs. We propose a new language that can retrieve data from multiple Online Social Networks through a single query. Concerning existing languages, the proposal offers in addition the possibility for users to add keywords in the language which reflect the metadata used by social networks to address its data. This feature, which we called *awareness*, enables the possibility to add knowledge into the language.

1.5 Outline of the Thesis

This thesis describes some aspects related to trust, security and privacy in the smart city domain. The thesis is divided in four main parts.

Part I contains the proposals based on the blockchain technology that have the main objective to enhance the trust and privacy level for daily situations in the smart city and in the smart grid. First, in Chapter 2 we deepen some basic concepts that are involved in all this part of the thesis. In Chapter 3 we propose a solution for energy trading based on blind auctions managed by a suitable Ethereum smart contract. Chapter 4 contains new solution for service delivery problems with accountability and privacy requirements. In Chapter 5 we propose a new attribute based access control mechanism based on smart contract for *closed data* in the smart city. To reach this goal we also used as data storage IPFS. Chapter 6 present a model to enable the propagation of trust in the Web of Trust domain by exploiting Ethereum smart contracts. Finally, in Chapter 7 we propose an approach to enabling Ethereum transactions also among users that are not yet registered into the Ethereum system by means of the Identity Based Encryption (IBE).

Part II focuses mainly on data management. We first propose a new paradigm to represent smart objects networks. Then, we focus on how collect, store and navigate all the data that smart objects, sensors, smart devices and others produce in such a way it is easy to extract knowledge from it. For this scope, we model the new data lake paradigm with the help of network analysis and graph theory. In addition, we exploit the metadata role in such paradigm to propose models and approaches that are more flexible, lightweight and adapt to unstructured and semi-structured sources. In this sense, we propose a model able to uniformly handle heterogeneous sources that represents the basis for further proposals. In detail, in Chapter 8 we model with graph theory and network analysis properties a new paradigm that can represent smart objects in multiple environments and we propose also a ad hoc crawler for this new kind of network. Chapter 9 proposes a new approach to uniformly handle heterogeneous metadata sources in the data lake domain. In Chapter 10 we exploit the above model to present a new approach for semantic properties detection and schema matching and integration among different data lake sources. In Chapter 11 we propose a topic-driven model able to extract thematic views from data lakes. Finally, in Chapter 12 our contribution regards an approach to extract complex knowledge patterns among concepts belonging to heterogeneous sources.

Part III of the thesis is about social network, participation and analysis. In Chapter 13 we first propose a model in which it is possible to compute, in a decentralized way, the trust of a user in the social network domain to avoid fake profiles. This work is in progress and it is in collaboration to a French group of researchers from ENSICAEN led by Prof. Rosenberger that is in charge to enforce the model by applying biometric features. Chapter 14 proposes a new query language SQL-like that is able to query multiple online social networks at once to retrieve data, correlate resources and integrate results from multiple OSNs.

Finally, in Part IV we draw our conclusions and delineate some future future works and developments of our research.

Part I

Blockchains for smart and trusted interactions

In the last decade, we are spectators of the rapid and massive introduction of new technologies in everyday life. Indeed, almost everything that involves both working and daily operations is transposed into the digital domain through the internet infrastructure and all services that lean on it. The new concept of Smart City is growing as a relevant challenge for scientists of several areas. Probably, the main cores of the smart city paradigm are data management (data generation, data collection, data navigation, etc.) and interactions among citizens and between users and the city itself.

New technologies clearly help to reach these objectives, but we know also that the digitalization we are searching and realizing often exposes some relevant and personal data on internet, so there is need to implement new solution that are, at the same time, faster, easier, more reliable and secure.

Smart city is one of the hotter topics for scientists also because it involves different areas of research (e.g., engineering, computer science, phycology, sociology, etc.). As for engineering, it is quite evident that it is necessary to contribute by proposing new solutions that are in steps with the new technologies for both old and new situations.

At the same time, the Blockchain technology is spreading rapidly and it is gaining momentum due to its advantageous properties.

This part of the thesis is devoted to investigate new proposals that could solve some open problems in the new smart city scenario. In particular, Chapter 2 deals with the technical study of some background concepts, technologies and properties that are helpful such as the the blockchain technology, with a focus on Ethereum, and smart cities, focusing on the energy management and the service delivery in this context.

The smart grid paradigm is very challenging for scientists since it requires to innovate every operation related to the energy infrastructure. In this sense, energy trading is a crucial phase and there is space to new proposals and approaches that fit with the requirements of smart grids and smart cities. Chapter 3 addresses the problem of energy trading in smart cities and smart grids for which a solution based on smart contracts is presented. Furthermore, this chapter faces and discusses the most relevant security aspects that are involved and guaranteed in the proposal.

The smart city concept has also the purpose to provide better services for citizens. When a customer wants to obtain a certain service, it is necessary to register into the service supplier platform. During this phase, the customer sends some personal data and identity information. However, there are many cases where the service could be granted by demonstrating to fulfil some requirements, like a certain licence, the possession of one or more certificates or attributes. In Chapter 4 we propose a solution in which the service can be provided to the customer based on attribute requirements. In addition, our proposal guarantee also accountability. We instantiate our model in a real-life scenario and we discuss and analyse security aspects as well.

Chapter 5 addresses on how an user can access information available in the smart city and how this operation can be protected against malicious people. Indeed, we can distinguish two kinds of data: *open* and *closed*. The former are available for everybody, while the latter are reserved to a certain subset of people. In detail, we propose an attribute-based access control solution for closed data and we combine it with the properties of smart contracts and blockchains. Moreover, we use *InterPlanetary FileSystem* (IPFS) as data storage to be more compliant with the direction of distributing information and operations.

In Chapter 6 we propose a solution that enables the propagation of trust in the Web of Trust concept. Several years ago Zimmerman [394] implemented this model in the *PGP* scenario, but, even if in theory it provided propagation, in practice this opportunity has never been implemented. We model a solution that enables this feature and we implement it through an Ethereum smart contract.

One of the known limitation of the platforms blockchain-based is that they require the user to be registered into the system before participating in. In Chapter 7 we model a solution for safe interactions in the blockchain domain among users that are not necessary registered yet to the system by means of *Identity Based Encryption*.

2 Background

This chapter introduces some fundamental concepts that are commonly exploited and used by the proposal of this first part of the thesis. First we discuss about the blockchain technology and how it works. Then, the focus is on the Ethereum blockchain, deepening some crucial aspects as Smart Contracts, Tokens and the programming language that let us implementing them, which is called Solidity. The second part of this Chapter is about Smart Cities and focuses mainly on Smart Grids and their open challenges concerning those situations in which is necessary to preserve and guarantee some security properties.

2.1 Blockchain and Distributed Ledger Technologies

The interest towards Blockchain [269] is constantly increasing during the last years, due to its power to enable new business scenarios. Blockchain technologies attract the attention of both industries and researches, in various fields, besides computer science, mainly also economics and law. As a consequence, any aspect regarding those technological features that impact how applications can be designed and used is very relevant.

Blockchains are part of the *Distributed Ledger Technologies* (DLTs) that implements a different logic from the centralized and the decentralized ones. In Figure 2.1 there is the representation of such differences.

In detail, the centralized logic obviously has a *one-to-many* ratio among the nodes belonging to the network, in such a way one node is much more important and has much more decision power with respect to the other ones. This could bring some advantages in specific situations, but otherwise this kind of network is very susceptible of attacks and, more in general, problems. Indeed, the whole network would be destroyed if the central node is, for any reason, off and not available.

Decentralized logic extends the previous one by applying it not just once to the network, but several times *locally*. In this way, this logic creates some satellites orga-

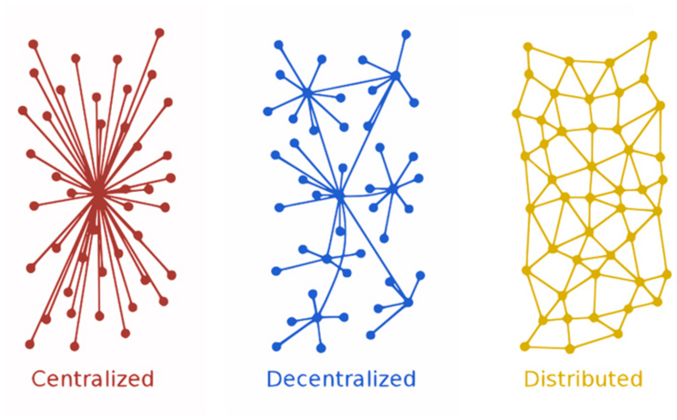


Fig. 2.1: Centralized vs decentralized vs distributed networks

nized, in turn, following the *one-to-many* distribution. This kind of logic is often used to represent governances establishing a coordination among locally central nodes.

The last logic, the distributed one, does not expect any kind of central nodes by creating the governance by means of the concept of *trust* among nodes. In this way of create and manage networks, decision making is achieved by an implementation of the *consensum mechanism*. Distributed Ledger Technologies (DLTs) are based on this last typology of network. In addition, they use a registry, named *Ledger* that is replicated, shared and synchronized over different and several peer of the network. Potentially, every single node could have a copy of this distributed registry.

As the term Blockchain suggests, it consists of a chain of blocks, each one connected to the previous one through the usage of cryptography. We can say that every Blockchain is a DLT but we can not say the vice-versa. Blockchain is a *crowd peer-to-peer* network, since community play a fundamental role to win against malicious attempts (the most famous example of these networks is *Wikipedia*).

The basic component of a blockchain network are the following:

- *nodes*, which consist of the participants of the network;
- *transactions*, which consist of the data and the value that the sender wants to forward to a recipient;
- *blocks*, which consist of a collection of transactions that must be verified and approved before adding it to the chain;
- *ledger*, which consists of the distributed registry where all information are persistently stored and available.

In particular, every block contains, among others, a direct link in its header (*prev-hash*) to the previous one, a timestamp of the creation of the block and all data of the

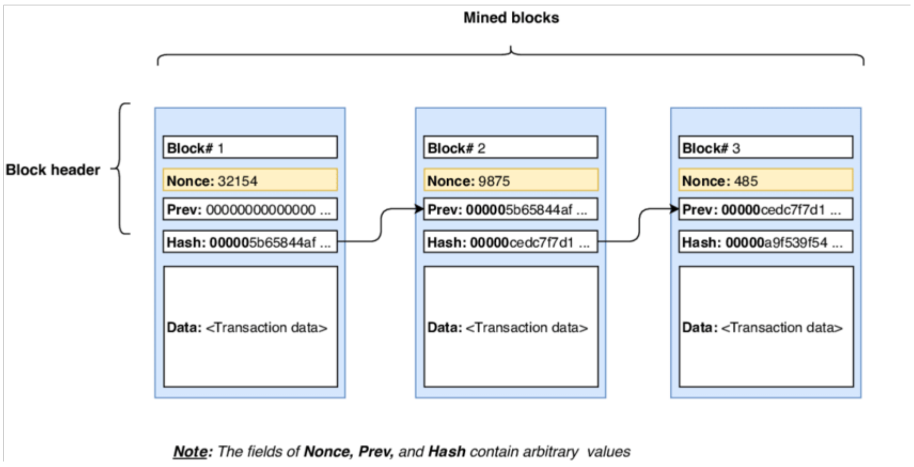


Fig. 2.2: Concatenations of blocks in blockchains

transaction included in the block. The hash code of a block is generated by the set of hash codes of every transaction grouped in the block and it represents the unique identifier of the block itself. This solution is summarized in Figure 2.2.

The most relevant properties and advantages of this technology are:

- *immutability*, once the transaction is added to the chain, it can not be altered;
- *transparency*, all operations are available and stored in the distributed registry, accessible from everyone;
- *traceability*, every block is linked to the previous one;
- *redundancy*, information is distributed;
- *security* based on cryptographic techniques.

Thus, the main challenge of blockchains is to find a distributed way to validate transactions among peers of the network through a valid consensus mechanism.

Indeed, one of the killer features that helped blockchains to impose themselves as a technology in the recent years can be seen in the *miner* figure and the *mining* process, which consist of the process necessary to ensure the security and verify the validity of all transactions inside a given block so that it can be added to the chain.

The mining process is correlated to the overall performances of the blockchain: the faster it is higher the performances of the network [372].

Definitely, the first platform that used the blockchain architecture is Bitcoin [269], a payment system that is served with a blockchain behind to register and validate transactions of cryptocurrencies (named Bitcoins) between people. In Bitcoin, users are identified by a string, so that sensitive and personal data are hidden to each other. This, combined with the properties of blockchain, let Bitcoin become soon very popular. Details and particular explanations of Bitcoin are not relevant for the purposes of this thesis, but it could be interesting to spend some words to sum up how the consensus mechanism works in this famous blockchain.

The mining process followed by Bitcoin is known as *Proof of Work*. In this solution miners have to solve a difficult mathematical challenge (known as *puzzle*) to verify the block and all the transactions inside it. In particular, the first miner who solved this enigma wins and can append the block at the end of the chain only if the majority of other miners accepts and validate it (here, the term consensus mechanism). The reward of this victory consists, in Bitcoin, of all fees (spent by the senders) included in the transactions plus an established amount of new cryptocurrency, defined as *coinbase*. It is evident that winning the mining process can bring some interesting benefits to miners in particular in financial terms. For this reason, during the years miners have bought a lot of performing hardware to have more computational power trying to solve these enigmas quicker than others, transforming, indirectly, the figure of the miner to real companies. On one hand, this helps to improve the overall performance of the blockchain but, on the other hand, this led to some, and not to underestimate, problems, like a huge consumption of electricity and the consequence of centralizing the power of the network in these (always less in number but bigger in computational power) mining pools, contrasting the original wills and principles of blockchain to distribute the decision power as well as information among all nodes.

In addition, since every mined block let the miner to earn new coin, and since it is fundamental to posses the majority of the computational power to be sure to mine a block in a malicious way (this would be very expensive for the attacker), blockchains and their protocols are considered very secure and the so-called *majority attack* is very difficult, if not almost impossible, to be carried out.

Finally, we can distinguish four category of blockchains that are shown in Figure 2.3 with respect to two different characteristic. A blockchain can be *public* or *private* [378] and *permissionless* or *permissioned*. Bitcoin is a *permissionless* and *public* blockchain because it does not implement any access control technique and everybody can read, join and participate to it. Instead, a blockchain is considered *private* and *permissioned* when data and information are available only to participants and not to external entities and when participants have to be accepted by a central, and more important, node. In particular, this last case is used often inside companies, where the management wants to follow internal transactions guaranteeing traceability and accountability.

More particular are the *hybrid* solutions. Indeed, we have a *public permissioned* blockchain when data is available and readable from every one (participants or not) but only accepted node can write on it. Instead, in *private permissionless* blockchains

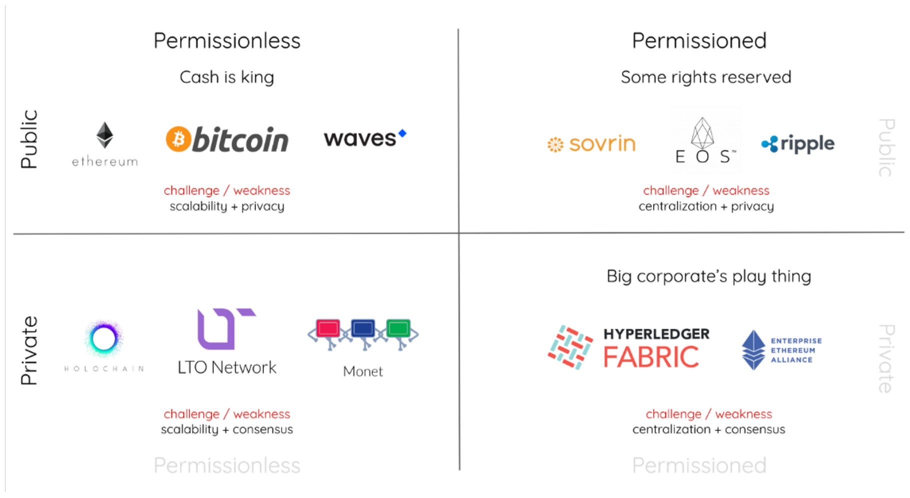


Fig. 2.3: Classification of blockchains

everyone can join the network but other nodes will not share any data. Indeed, every peer has its ad-hoc chain [124] so creating a private side-chain.

2.2 Ethereum

Ethereum is a public blockchain-based platform that allows the development of decentralized applications (known as Dapps) that let the interaction among nodes of the network be secure and fast [142, 375].

Ethereum is attracting the interest of both the industry and the research areas since from its publication mainly because of the new features of *smart contracts* and *tokens*.[1]

In last years, after the incredible and exponential success of Bitcoin [269] and others cryptocurrencies blockchains, it is growing the second era of blockchain (so called Blockchain 2.0) characterized by the new feature of smart contracts.

Ethereum [142] is one of the blockchain platforms attracting the interest of both research and industry, mainly due to the power of *smart contracts*. Indeed, when different parties with conflicting interests have to exchange value, a problem is how to prevent that one of the parties, in a certain moment, misbehaves to obtain an advantage, so that the agreement is not concluded fairly for everyone. Smart contracts solve this problem. The consensus mechanism implemented by Ethereum guarantees that all the contract steps are automatically executed in a transparent way, according to the agreed rules, without needing that parties have to trust each other.

Ethereum [20] is the progenitor and the most relevant technology of this new era. It is not just a blockchain for cryptocurrencies but a platform that can be defined as

[1] These characteristics will be explained in the following.

a programmable public and permissionless blockchain, having the main intent to provide an alternative protocol for building, in a fast, secure and interoperable way, decentralized applications (DApps).

Smart contracts are executed by the Ethereum decentralized and distributed Virtual Machine (EVM) [375]. They are written in the Turing-complete language *Solidity* [27] that implements the EVM bytecode. It is worth noting that the definition of a smart contract depicts it as a *self-executing contract that has inside the terms of the agreement between two parties without the need for a central authority*. The *ether* is the cryptocurrency generated by the Ethereum platform and used also to pay transaction fees. In Ethereum, differently from the Bitcoin blockchain, there are two different types of accounts:

- Externally Owned Accounts (EOAs);
- Contract Accounts (CAs)

The former are controlled by private keys, so by people who owns these keys, while the latter are controlled by the contract code. In detail, every account has a 20-byte address and has an ether-balance and ethers can be transferred among accounts. An EOA can send transactions that will be stored inside the blockchain to create a smart contract or invoke a function inside it, or again, simply to transfer ether to another account. Instead, the CA can be activated only by an EOA. These families of accounts open new horizons regarding transactions. Indeed, in Ethereum, there are the so-called *External Transactions (ETs)* and the *Internal Transactions (ITs)* (known also as *Contract Transactions*). The former are generated by EOAs and they are publicly and transparently recorded inside the blockchain [19] while the latter are sent from a contract to other contracts and these are not recorded on the blockchain and do not affect the states of other accounts [19]. An Ethereum environment [21] is used to create and publish smart contracts and DApps.

Ethereum, as a blockchain, requires that transactions and blocks have to be validated by miners. In addition, every computational step carried out requires also some extra-charges to be paid by users. This kind of fuel is called *gas*, which is a unit of measuring the computational work of running transactions or smart contracts in the Ethereum network. In particular, gas is expressed through *gwei*, that is a subunit of ether.

Furthermore, every user can specify, through the field `GAS_PRICE`, how much she/he is willing to pay for each computational step. Obviously, the higher this field is, the earlier the transaction will be chosen by miners. Another interesting field is `STARTGAS`, representing the maximum number of computational steps the transaction execution is allowed to take and that helps to avoid loops.

POW More in detail, this mechanism is established by mining based on the proof-of-work (PoW) scheme. The PoW assumes that the winner miner is that one who solves first some mathematical puzzles. The average time for mining a block of transactions is about $10-12$ seconds. As a consequence, new branches of the principal chain are generated very often and it is necessary to manage these forks to guarantee a certain level of security and decentralisation of the mining process. For this purpose, Ethereum implements a simplified and modified version of the protocol $GHOST$ (Greedy Heaviest Observed Subtree) in such a way also "uncles" nodes are partially considered in the computation of which block has the largest and heaviest total proof-of-work backing it.

2.2.1 Tokens

Tokens represent probably, together with smart contracts, the killer feature of the Ethereum environment and its success. Indeed, a token is a virtual asset that can be created by every peer of the blockchain [360] that has not a well-defined associated economic value. In fact, the token has not a value until it is contextualized in a certain domain. Generally speaking, we can see tokens as money: they initially are *just paper* and the value they have is given by the context in which they are used. So, we can define a token as a particular cryptocurrency that has no value until someone or something (e.g. the crypto-market) gives it to it. Usually a company or a single-person that decides to implement a new token via smart contract publishes the business idea in a white paper and offers the token during a Initial Coin Offering (ICO) period.

We can distinguish two main families of Ethereum tokens: *fungibles* and *non-fungibles*. The former are defined as tokens that are fully interchangeable (i.e., all tokens are alike) and they could be used, among others, as sub-cryptocurrency for payments, while the latter are tokens that have an identifier or a label, so that they are mostly used as virtual collectables [360] or in specific situations where it is required that every token is different from the others. The ethereum community worked on some standards helping the creation and the sharing of tokens by developing tokens as interfaces to be implemented in such a way users can use them. In particular, the standard *ERC-20* (Ethereum Request for Comments, number 20) is one of the most popular fungible tokens. It is composed of six mandatory functions to be implemented, plus three optional ones. However, since *ERC-20* has some limitations in terms of costs and functions, the new *ERC-223* standard has been recently proposed. It is fully backward compatible with *ERC-20* and it solves the above limitations.

On the other side, the most popular standard for non-fungible tokens is *ERC-721*, which has been recently improved by the new *ERC-1155*. These kinds of tokens are associated with some metadata, in which it is possible to save information that characterized and identify uniquely each token.

2.3 Smart City

E-participation is now imposing as the new way to communicate, interact and sharing information among people. Moreover, services provided by governmental entities and companies are moving rapidly into the digital domain. On one hand we meet people and citizens that ask for new services and new levels of efficiency and reliability in the interactions between them and the city and, on the other hand, we have cities that are facing big problems in terms of scalability and security because of using legacy systems and also because of the urbanization phenomenon: many citizens are moving from their native places to settle into big cities [282] saturating resources available. These two main situations are making cities literally moving from a classical interpretation to a new paradigm called *Smart City*.

In recent years energy consumption increased exponentially; the environment is hardly challenged; healthcare and education systems are demanding new approaches; public safety is challenged as well; and the potential for future cyberattacks against cities is dramatically high. Without innovative solutions, this situation can lead to further environmental degradation and poverty. It is necessary to rethink the models of access to resources, transport, waste management, and energy management [209]. Hence, smart, cost-effective, scalable, secure and innovative solutions that can address the problems of urbanization are needed [210].

Despite there is some kind of consensus that the label smart city as innovation in city management, its services and infrastructures, a common definition of the term has not yet been stated. There is a wide variety of definitions of what a smart city is [266].

This paradigm involves almost every area of research, from engineering to architecture, from social sciences to economics, etc. [63]. Smart cities aim at evolving everything inside a city into something smarter, more eco-friendly and sustainable so that reducing pollution, costs for citizens and improving services. Clearly, this goal is very ambitious but researchers are working hard in recent years to propose solutions able to reach it. Among others, cybersecurity is one of the area more affected by this new paradigm [210].

The concept of the smart city emerged during the last decades as a fusion of ideas about how information and communications technologies might improve the functioning of cities, enhancing their efficiency, improving their competitiveness and services, and providing new ways in which problems of poverty, social deprivation, and poor environment might be addressed. Intelligent cities, virtual cities, digital cities, information cities, smart cities are all perspectives on the idea that ICT is central to the operation of the future city [69].

Nowadays, cities face different problems and challenges to improve their citizens' quality of life [343]. Governments, communities, and businesses increasingly rely on technology to overcome the problems that daily arise [374]. Smart cities can make an intelligent response to different kinds of needs, including public safety and services, industrial and commercial activities, transportation, and healthcare [341]. In detail, a city becomes a smart city when it combines the usage of network infrastructure, software systems, server infrastructure, and client devices to better connect critical city infrastructure components and services. Smart cities are an effective integration of smart planning ideas, smart development approaches, and smart management methods. On the other hand, a city cannot be defined as smart if it adopts limited and sectorial improvements.

Indeed, a smart city must involve different elements such as smart governance, smart economy, smart mobility, etc. Smart cities make use of new types of information and communications technology to support common sharing which is one of their most important characteristics. It is well-known that the features of blockchain technology may contribute to the smart city development through sharing services.

As said smart cities can be seen and defined differently based on the area we are talking about. Architects will define them in a certain way, sociologists will see it in another perspective, etc. What is instead universally recognized is the relevance of data. Data represents in smart cities the most important feature to investigate and to study because of the huge advantages a correct data management could bring into this new paradigm.

In a high level perspective we can distinguish two different typologies of data:

- *open data*;
- *closed data*.

The former are data available for every access. Anybody can read them without any constraint. The latter are subjected to some policy and constrained.

2.3.1 Smart Grids

A big component in the smart city strategy is represented by *Smart Grids*. We can define the smart grid as the successor of the classical electric grids, having the main purpose to design new solutions that are more reliable, faster, more secure and more efficient.

Smart grid technology is changing the way traditional power grids operate by reducing energy demands, global warming and consequently, utility costs. Consumers are required to share information about their energy consumption with their utility providers, over communication channels using smart meters [63].

Smart grids have as main aim also to improve the overall reliability of the whole energy cycle and to guarantee a better ratio demand/response so that the financial field is interested as well by applying a new energy market pattern [329]. Moreover, by increasing the energy demand and the number of entities involved in the energy market, smart grids have to face the problem of guaranteeing a certain level of data and message availability in transmissions among peers of the network [96].

Smart grids (SGs) are designed to work most with renewable energies. The direction taken by SGs is migrating from a centralized to a distributed energy market model, in which customers have more decision-making power, according to their role of producers and suppliers of energy as well. Therefore, there is a new figure of the energy user in smart grids: the *prosumer*, who acts, in the smart grid environment, as both the consumer and the producer of energy. Indeed, the smart grid protocol is quite close to a P2P solution in which there is not anymore a hierarchical relationship among nodes.

The main functionalities of smart grids start with modernizing power systems by means of real-time monitoring, automation and self-controlling issues [149]. Smart meters play a crucial role in this sense. A smart meter is a hardware component that can run software that makes it capable to manage (also by sending) electricity generated by a prosumer and to respond to external requests [240].

Moreover, smart meters are able to guarantee a real-time communication network connecting the grid with electricity providers and consumers. Smart grids use this communication network to collect demands from consumers and to reply to them by optimizing resources available on the grid [329, 177, 383]. Moreover, smart grids have some internal features, such as self-healing and the ability for fault detection, which allow them to continue providing the energy flow to the grid. Another challenge that the new smart grid paradigm wants to address regards the energy market mechanism, both in terms of production of new renewable energy and the energy trading aspect. Indeed, smart grids want to make closer all steps associated with the energy. For this scope, the *prosumer* actor is now involved in the scenario. Briefly, we can define a prosumer as a participant that both produce and consume energy and electricity in the network.

Energy trading is one of the most important components in the smart grids' energy management as well as in the more classical energy market, in which it represents the last phase of the cycle. Indeed, a very high-level description of how the energy market works is the following:

1. the energy is produced by generators;
2. the energy is transmitted to the distribution network;
3. now, retails are in charge of connecting the distributors to consumers by buying and selling energy;
4. the consumer can obtain the energy needed by paying to retailers.

Smart grids aim to make these steps closer to each other, to improve the overall energy consumption and system efficiency, and to reduce cost and time.

If we just think of the new figure of the prosumer, it is clear that this cycle is inherently faster in a smart grid scenario with respect to a classic grid, since there are not only central generators but energy can be created and transmitted to the distribution network also via prosumers themselves. Anyway, step 4 is quite a bottleneck also in smart grids because it still needs to follow some traditional criteria that are not fully compliant with the smart grid proposal and it deserves new approaches to be investigated.

3

A new architecture for energy trading in smart grids

Smart Grids represent one of the most relevant challenge of smart cities since they aim at evolving from the classical electric grids to something more compliant with the technology progress of last decades. In details, Industry 4.0 and Smart Grids are pursuing the path of automation of operations to make closer and quicker all the steps of producing, collecting and distributing energy. We propose a new ethereum-based solution that provides trust and accountability to mutual interactions. However, since ethereum is public, we have also to ensure the solution from threats to privacy. The solution aims to be a concrete proposal to accomplish the needs of energy trading in smart grids, including the important feature that no information about the identity of the peers of the network is disclosed in advance.

3.1 Introduction

Due to the continued growth in energy demand, the issues of increasing its production, on the one hand, and to limit environmental pollution, on the other hand, are becoming global challenges.

Of course, it is necessary yet not sufficient to extend the usage of renewable energy. Only in 2018, renewable energy raised by 4%, accounting for almost one-quarter of global energy demand growth [22].

Energy systems in smart grids are taking the direction of decentralized architectures in which a device, known as *smart meter*, can manage requests and responses through the whole network. Since it would be not appropriate to implement centralized protocols over smart grids, it is fundamental to accommodate this decentralized and distributed direction by using technologies that are decentralized and distributed as well. This way, blockchain technology appears to be the best solution, because of its proven properties, such as immutability, transparency and decentralization. Indeed, thanks to the evolution of the blockchain paradigm originally born with Bitcoin blockchain [269] (mainly devoted to the cryptocurrency Bitcoin), blockchains supporting *smart contracts*, like Ethereum [20], can be viewed as platforms for secure, interoperable, and decentralized applications, in which conflicting parties may establish agreements and exchange value without trusting each other. Energy trading in smart grids perfectly fits with these features. Therefore, an interesting research direction is to investigate how to fully exploit the power of blockchain and smart contracts to envisage innovative applications and to increase the effectiveness of the notion of smart grid. Observe that the use of blockchain may introduce flexibility among operations carried out by stakeholders inside the energy trading market. In particular, if we overlap these features with the energy industry we can deduce that the sector that can benefit most from them is energy trading among applicants and bidders. Indeed, a blockchain-based solution for energy trading is able to improve accountability, reliability, fairness and to reduce time and costs.

The blockchain enables parties to transfer assets without the participation of a trusted third-party and all the transactions are stored and validated by the network, with no centralized unit control.

Indeed, in our proposal the power of smart contracts is also exploited to manage the offers in a blind fashion, so we can actually talk about an auction, in which identities are disclosed only when the agreement is established. Interestingly, the entire auction is managed with no intervention of any external referee.

3.1.1 Scenario and motivation

From the beginning, the electricity grid was conceived as a centralized system in which energy is produced in huge power plants. It is clear that this kind of system has limits in reliability, availability, and, as a consequence, also in business terms. Moreover, the growing world population generates a rising demand for energy and, due to the increasing level of pollution, the request for sustainable and renewable energy is necessary. Indeed, investing in renewable energy is becoming central in most of the world governments for environmental protection. Energy is a raw material and for this reason it can be exchanged; energy trading term means buying, selling, and moving energy from where it is produced and generated to where it is requested.Through the years, the energy systems have been developed into four different stages: decentralized or centralized energy systems, either distributed or smart and connected energy systems.

A decentralized approach can evaluate the energy exchange [388], considering the decreasing price of distributed energy resources in the ten past years, known energy consumers can become prosumers, that is they can both consume and generate energy. The consumers, instead, only purchase energy. There are several business initiatives whose aims are to improve the energy use and consumption all over a (smart) grid. Many of these initiatives are characterized by similar actors and operations. Indeed, the actors in an energy trading scenario could be represented by:

- *Consumer*, a physical person who needs to buy electricity.
- *Prosumer*, an entity that acts as an energy supplier (such as farmers with wind turbines or an individual who produces additional energy) and at the same time uses and buys electricity. In detail, we can consider a prosumer as a consumer with the ability to produce energy as well. So, every prosumer is a consumer while the contrary is not always true.
- *Retailer*, which buys and stores electricity from prosumers and sells it to customers (both prosumers and consumers). The retailer is also responsible for getting customers connected with the network and for customers' billing and service. A retailer can be seen as an intermediary agent between the energy market and the energy consumers.

The operations carried out by the actors could be divided into three phases, implemented through different approaches, as the study [369] suggests. The first step consists of making aware the network about the own energy supply and demand. This step requires the adequate controls to ensure the privacy and security of the actors. The second step regards the matching among consumers and prosumers. Specifically, the consumer chooses the most suitable prosumer able to fulfil the request. Many times, this phase is implemented through an auction process. The transaction settlement is the last phase, it consists of establishing the rules, among the parties, to guarantee the transfer of energy.

The main aim of this proposal is to provide a protocol that takes into account the security and privacy requirements in the new energy trading scenario.

Indeed, during the various processes in which the prosumers are involved, they should declare and disclose their identities because of the huge trading activities. When a consumer demands for energy, an auction starts, the winner prosumer stipulates a contract with the consumer. During these phases, being aware of the actors' identity could cause a possible impairment. Furthermore, dynamics is required because prosumers are not known first. For this reason, an important issue is to implement a privacy-preserving approach in the auction phase. Exploiting the blockchain technology in an energy trading scenario can include the well-known advantages, such as the elimination of a central governing institution, a distributed consensus, and the immutability and accountability of transactions. At the same time, designing a smart grid fully automated can be advantageous and helpful in the cost reduction of transactions and electricity. Although the other proposed solution consisting in the integration of blockchain for energy trading seems to solve the problem, there

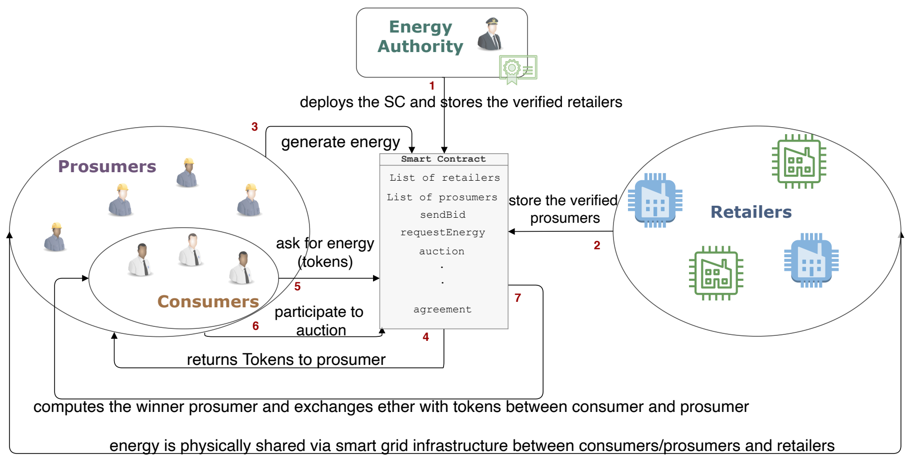

Energy Authority
1
deploys the SC and stores the verified retailers
3
generate energy
Prosumers
Smart Contract
List of retailers
List of prosumers
sendBid
requestEnergy
auction
agreement
store the verified prosumers
2
Retailers
ask for energy (tokens)
5
Consumers
participate to auction
6
7
4
returns Tokens to prosumer
computes the winner prosumer and exchanges ether with tokens between consumer and prosumer
energy is physically shared via smart grid infrastructure between consumers/prosumers and retailers

Fig. 3.1: The architecture of our proposal.

are still open issues such as the spreading of energy trading in a public blockchain, or the responsibility in the transactions derived from the anonymity ensured by blockchain. For these reasons, we propose an approach that includes the management of the actors' identity to make transactions accountable.

This way, the final agreement will be achieved among not anonymous entities.

3.2 The architecture of our proposal

In this section, we define the architecture and we describe the steps of our solution. First, we denote the involved entities and, then, we show the entire process of the proposal.

Figure 3.1 illustrates the overall architecture of our solution.

The actors involved in this scenario are *consumers*, *prosumers*, and *retailers*. We exploit the Ethereum blockchain to store the information in a distributed and immutable way and also to guarantee the security properties. Consequently, in our proposal, we include a new actor, the *Energy Authority*, which is the entity that deploys the smart contract needed to drive our solution.

In our solution, the following steps can be identified:

1. **Setup.**
   In the initialization phase, a suitable smart contract SC is deployed on Ethereum by the Energy Authority. It implements the functions that are described and used in the following. Moreover, both prosumers and retailers register an Ethereum address. For the sake of clarity and to better improve readability of the proposal, we will call from this point prosumers as producers that keep maintaining both their role of producing and consuming energy.

2. **System Registration.**

   In this phase, the entities join the system. First, the owner of the smart contract identifies each retailer and verifies its Ethereum address by a challenge-response scheme. In particular, the retailer must sign a challenge sent by the owner by using the private key of its Ethereum account. For each verified retailer, the smart contract owner (that is the Energy Authority) invokes a function of SC and gives the retailer's address A_R as an input parameter. This function adds A_R to the list of the *verified* retailers L_R managed by the smart contract. A verified retailer R can register one or more producers P in the system. This operation is done by calling another function of SC and giving the prodicer's address A_P as an input. Again, the retailer verifies the producers's address by a challenge-response procedure. The result of the function call is the inclusion of this address to the list of the *verified* producers L_P, which is also managed by the smart contract. These procedures are repeated every time the smart contract owner wants to add a new retailer or a retailer wants to add a new producer to the system. At the end of this phase, it is possible to verify whether an Ethereum address (that we call Main Ethereum Address MEA) is associated with a verified retailer or producer.

3. **Energy Production.**

   The actor involved in this step is the producer, which generates energy and trades it with the retailers. In particular, given the producer P_i, she/he can transfer a given amount of energy, say E, to the retailer R_j thanks to the smart grid infrastructure. Indeed, in the smart grid environment, there exists an IoT device, the smart meter, that is fundamental to link the consumer to the whole energy infrastructure. We propose an easy extension of such a smart meter that will include also the possibility of connecting to the Ethereum blockchain. This can be reached by adding a new feature on this device that will have associated an Ethereum address and, through the Internet, it will be able to interact with the blockchain network. In particular, this device acknowledges an input and output energy transfer in terms of tokens via SC.

   The smart contract sends a certain amount Tk of tokens to the producer. The value Tk is computed as $Tk = E \cdot c_{i,j}$, where $c_{i,j}$ is the exchange rate between the producer P_i and the retailer R_j. Moreover, SC generates, at this point, an event *Transfer* to log the operation carried out.

4. **Energy Request.**

   In this step, a consumer (or a prosumer acting as a consumer) asks for energy (i.e, tokens) by building a request containing the amount of energy needed. In particular, this task is carried out by calling the function `newAuction()` and giving as parameters the amount of requested energy, and two timestamps d_1 and d_2 used as deadlines of the auction. Observe that the consumer does not call this function by the Ethereum account generated during the System Registration step, but generates a new address for the function call in such a way that the energy applicant is still unknown and not identifiable. We call this address Temporary Ethereum Address TEA. At this point, the smart contract starts a blind auction with a fixed deadline d_1.

5. **Auction.**
   Any producer can participate in the auction by bidding a price p for this supply. In particular, the producer has to call the function `SendBlindBid()` of the smart contract by giving it the blind offer of the price $H(p\|r)$, where r is a random value and H stands for a cryptographic hash function. This way, the real bid is hidden to the other competitors.
   We remind that to prevent identity disclosure the producer uses a new TEA to participate in this auction.
6. **Awarding.**
   At the auction deadline d_1, each producer that participated in the auction calls the function `sendBid()` and passes as parameters the values p and r in plain-text to disclose its offer.
   After all participants reveal their offers or after the deadline d_2 established previously by the energy applicant, the auction is awarded to the best bidder. In fact, the energy applicant retrieves the best offer related to its auction by calling the function `endAuction()`, which computes the best offer and returns the winner bidder.
   Before establishing the winner, this function calculates $H(p\|r)$ and verifies that the result is equal to the value submitted in the previous step, thus validating the offer.
7. **Agreement.**
   Now, both the consumer and the awarded producer must disclose their identities. For this purpose, the producer has to link its Temporary Ethereum Address TEA used in the previous phase to its Main Ethereum Address MEA (which is publicly available) by generating a transaction from MEA to TEA and another from TEA to MEA.
   This way, the producer proves to be the owner of both the Ethereum addresses. It is now necessary to check that the MEA associated with the awarded producer has, at least, p token available in its wallet. This means that the producer can fulfill the consumer request. If this check fails, the smart contract discards the awarded producer and, by shifting the list of producer participants in the auc-

tion, it repeats the operations with the newly awarded producer. This cycle is repeated until all the requirements are fully satisfied.

At this point, the consumer has to prove to be the owner of the address A used during the auction. To do this, the prosumproducerer generates a random value r (challenge), which is sent to the consumer. The consumer generates a new transaction from A to MEA of the prosuproducermer having as payload r, thus proving to be able to win the challenge. Moreover, the consumer uses an identity-based authentication scheme to disclose her/his identity: for example, schemes such as OpenId-Connect and SAMLv2 [268] can be used. If these operations succeed, the consumer and producer complete the auction by exchanging tokens and ethers as resulting from the energy request and auction.

8. **Redeem Tokens for Energy.**

   This step can be carried out by everyone with tokens in their Ethereum wallets, so both producers and consumers, which want to redeem tokens for energy. During this process, the energy applicant has to send tokens towards the retailer by using a given function of the smart contract. This method will check that the sender has got the amount of token in the wallet and that the recipient of this amount is a registered retailer. If these controls succeed, then tokens are transferred from the applicant wallet to the retailer one. At this point, the retailer sends to the applicant electricity through the smart grid's infrastructure and generates, at the same time, an Ethereum transaction with the information regarding the amount of energy sent.

   However, since the retailer is not fully trusted (as it happens in real-life architectures as well), it is necessary to adopt some countermeasures to contrast a hypothetical malicious behaviour of the retailer. At this point, the applicant's smart meter plays a fundamental role.

   There are, potentially, four options: (*i*) the energy received is compliant with the agreement, (*ii*) the energy received is less than the agreement, (*iii*) the energy received is more than the agreed amount, (*iiii*) the energy is not received. Based on these situations, the smart meter will generate, as an answer, an Ethereum transaction by calling a function of *success* or *failure*. In this last case, a dispute arises between the energy applicant and the retailer. Here, the Energy Authority plays the key role as a *super party* actor to effectively mitigate and solve the problem.

3.3 Implementation

In this paragraph, we present the implementation of our proposal and describe the Ethereum smart contract that provides the needed functionalities. First of all, it is necessary to set-up the environment useful for the development of such a smart contract. In particular, we use *Remix* as Integrated Development Environment (IDE) and *MetaMask* that is a browser extension that allows us to run Decentralised Applications (dApps) directly on the browser without running a full Ethereum node. The programming language is Solidity [27] and the smart contract has been deployed on the Ropsten TestNet.

Let's move now into the real implementation of the proposal. First, we had to define and declare the $ERC20$ token interface, in such a way our smart contract can inherit it, by implementing its functions. We gave the token the name of *SET*, which stands for both *Smart Energy Token* and *Smart Energy Transfer*. Because of the aim of such a token, the Initial Coin Offering (*ICO*) period is not necessary so that the initial total supply was given totally to the developer of the smart contract by means of the constructor function. In fact, in Solidity, the constructor method is called and executed only once, that is when the smart contract is deployed. We use also this characteristic for storing the information about the real developer of the smart contract in the `owner` variable. We remind that, in our case, the developer of the smart contract is the Energy Authority.

Another fundamental Solidity properties we exploited is the $modifier$, which is used to limit the access to functions. In detail, in Listing 3.1, we implemented a modifier that, if declared in a given function, limits the access only to the developer of the contract. For example, we used this modifier in function `_add_retailer()`, which can be called only by the Energy Authority to add the addresses of verified retailers to this particular list. An analogous pattern has been used also in the case of the insertion of verified prosumers into the list by declaring the function`_add_prosumer()` with the corresponding modifier `onlyRetailer` and. Generally speaking, this pattern is used every time a function needs this kind of restriction.

```
modifier onlyOwner() {
      if (msg.sender != owner) {
            revert();
      }
      _;
}
function _add_retailer(address _new_retailer)onlyOwner public{
      retailers_list[_new_retailer]=true;
}
```

Listing 3.1: Application of the Solidity `modifier` in our smart contract

In the whole demand-response cycle, the first operation that is carried out in the Ethereum environment is the *Energy Request*. In Listing 3.2, we implemented the function `newAuction()` that, if called by an user, generates and activates a new auction on the system. The energy applicant has to specify how many $kWhs$ are needed and the periods of time she/he wants to wait for the completion of the whole process. In detail, the applicant has to give two timeouts to the function. The first timeout denotes the period of time in which the auction is active while the second one denotes the period of time until the prosumer can send the plaintext bid.

When the new auction is created the smart contract adds it into the mapping `all_auctions` and the function emits also an event to log this operation.

```
function newAuction(uint kWh, uint timeout1, uint timeout2)public {
      uint id_auction = getID();
      all_auctions[id_auction].consumer = msg.sender;
      all_auctions[id_auction].active = true;
      all_auctions[id_auction].end_of_auction = now+timeout1;
      all_auctions[id_auction].end_of_disclosurement = now+timeout1+timeout2;
      emit newAuctionGenerated(msg.sender, id_auction, kWh, now+timeout1,
       now+timeout1+timeout2, now+timeout1+timeout2);
 }
```

Listing 3.2: Creation of a new auction

At this point, prosumers can send their blind bids to answer the token request by calling the function `sendBlindBid` and, before the second timeout expires, they call the function `sendBid()`, in which they reveal the real offer made 3.3.

```
function sendBlindBid(uint idAuction, bytes32 blind, bytes32 hashRandom) public returns (bool) {
      require(all_auctions[idAuction].active==true && now<all_auctions[idAuction].end_of_auction,
       "The auction is now closed");
      blindBid[msg.sender].idAuction=idAuction;
      blindBid[msg.sender].blind=blind;
      blindBid[msg.sender].hashRandom=hashRandom;
      blindBids[idAuction].push(blindBid[msg.sender]);
      return true;
}

function sendBid( uint idAuction, uint cost , uint _random ) public returns (bool){
      require(all_auctions[idAuction].active==true && now>all_auctions[idAuction].end_of_disclosurement ,
       "It's too late");
      if(blindBid[msg.sender].blind == keccak256(abi.encodePacked(toBytes(cost),toBytes(_random)))){
            bid[msg.sender].cost=cost;
            bid[msg.sender].idAuction=idAuction;
            bid[msg.sender].bidderAddress=msg.sender;
            bid[msg.sender].random=_random;
            bids[idAuction].push(bid[msg.sender]);}
      return true;
}
```

Listing 3.3: Functions sendBlindBid() and sendBlind()

The next step is to compute the winner prosumer after the end of the auction. So, the tokens applicant calls the function `endAuction()` that first checks whether the auction is closed and, if this operation successes, it computes the winner prosumer. The code of these steps is shown in Listing 3.4.

```
function getBestValue(uint idAuction)public returns(offer memory){
      require(all_auctions[idAuction].consumer==msg.sender && now >
       all_auctions[idAuction].end_of_disclosurement);
      offer memory _o = bids[idAuction][0];
      uint best_cost= bids[idAuction][0].cost;
      uint n=bids[idAuction].length;
      uint pos = 0;
      for(uint j=1;j<(n);j++) {
```

```
                if (bids[idAuction][j].cost<best_cost && bids[idAuction][j].unvalid == false ){
                        best_cost = bids[idAuction][j].cost;
                        _o = bids[idAuction][j];
                        pos = j;}}
        bids[idAuction][pos].unvalid=true;
        return _o;
}

function endAuction(uint idAuction) public returns ( address, uint) {
        require(all_auctions[idAuction].consumer==msg.sender && now>all_auctions[idAuction].end_of_auction,
         "The auction is still active");
        offer memory best_offer=getBestValue(idAuction);
        address winnerAddress= best_offer.bidderAddress;
        uint winnerBid = best_offer.cost;
        all_auctions[idAuction].winner=winnerAddress;
        emit eventEndAuction(winnerAddress, winnerBid);
        return (winnerAddress, winnerBid);
}
```

Listing 3.4: Ending of the auction and computation of the winner prosumer

Now, the winner prosumer and the energy applicant have to send, respectively, tokens and ethers to the smart contract, which will collect and exchange them with each other. Since the prosumer participated in the auction with a temporary Ethereum address TEA, now it has to use the main Ethereum address MEA to send tokens and receive ether. In particular, the function `putToken()` is called by the main Ethereum address of the prosumer, to demonstrate it is the real owner also of the address that won the auction. To achieve this goal, the prosumer has to carry on the following steps. First, it has to sign the hashed MEA with the private key corresponding to the temporary Ethereum address that has been used to participate in the auction. At this point, the prosumer uses its MEA to send this signed hashed information together with tokens in such a way to demonstrate it is the actual possessor of both the MEA and the TEA.

Finally, the energy applicant, which can be both a prosumer or a consumer, has to exchange its tokens with the retailer to obtain physically the energy needed. This operation is carried out by calling another function that is used to receive tokens and triggering the dispatch of the electricity thanks to the smart grid infrastructure.

It is also necessary to analyse the performances of our proposal. Indeed, some issues may arise with respect to the latency between the energy demand and the final energy response since we propose an ethereum-based solution. Let be t_e the time that it is necessary to verify and validate a transaction in Ethereum. In average, Ethereum add a block of transactions to the network each 10 seconds. We have also to sum every timeout (for the sake of presentation we indicate the sum of every timeout with to) the energy applicant chooses. indicatively, we obtain a time $T = (n * t_e) + to + \varepsilon$ to complete a whole cycle of demand-response through our solution. It is now necessary to add the physical energy transmission time as well that is over the contribution of this study but it depends on the smart grid infrastructure. Clearly,

Function	*Milliether*	*US Dollars*
`newAuction()`	0,149	0,035
`sendBlindBid()`	0,023	0,005
`sendBid()`	0,027	0,006
`endAuction()`	3	0,71
`Whole Smart Contract`	4,684	1,12

Table 3.1: Costs of the deployment of the Smart Contract and of the functions

as other approaches correlated to the transmission of electricity in smart grids, it is not possible to satisfy a real-time request for energy from consumers because of known physical limitations of the infrastructure. For this reason, many researchers are working and proposing new solutions to forecasting the energy loads by means of many different approaches. In particular, in these lasts years the new proposals involves often machine learning algorithms or the usage of historical smart meter data, that are able to learn from the past to better forecast needs of consumers [135, 298, 346].

Since every operation carried out on the Ethereum blockchain has a related cost, we summarise in Table 3.1 the costs associated with our implementation. In particular, we focus on the most common and used functions of the smart contract and also the entire (and unique) deployment of the smart contract, reporting both the values in milliether and in US dollars (in July 2020).

3.4 Security aspects

In this section, we discuss the security properties and the adversary model of the solution described above. We show that the following security properties are guaranteed:

Access Control refers to protecting information from unauthorized users. In our case, the real values of the bids should be hidden and protected from the other auction competitors until the auction deadline.

Data integrity refers to the completeness, consistency, and accuracy of data. Data used for energy trading should not be tampered with: in particular, once declared, the price of bids during the auction phase should not be modified by anyone.

Privacy requires that no identifying or sensitive information is disclosed if not necessary. In our case, both prosumers' and consumers' identity information should be preserved during an auction to assure fairness.

Authentication guarantees the verification of the identity of the entities accessing a protected system or a resource. We require that, after the auction, the involved actors are aware of their reciprocal identity.

Accountability assures that the operations carried out in a collaborative system occur in an open and accountable way. In our solution, we refer to the accountability of every transaction among actors.

Reliability is the probability that a system can perform a predetermined function under given conditions for a given time. In our scenario, reliability means that the actors can exploit system functionalities, such as the request for energy, the auction, or the agreement between prosumers and consumers ensuring the continuity of correct services.

After describing the security properties to guarantee, we define the adversary model. In our analysis, the energy authority is a trusted party and behaves responsibly and correctly in the system. In contrast, a retailer, a prosumer, or a consumer can be malicious and act as an adversary internal to the system. Clearly, the adversary can also be an external entity of the system. In our attack model, the adversary cannot compromise the behavior of the energy authority and cannot guess randomly generated values, secret information, blockchain private keys, passwords of the other entities. Furthermore, the adversary cannot execute transactions from the Ethereum accounts of the other entities. The adversary cannot break the cryptography primitives (e.g., it cannot revert cryptographic hash values or decrypt ciphered messages) and cannot perform physical attacks on the infrastructure (e.g., tampering with smart meters). The goal of the adversary is to violate at least one of the security properties listed above.

Let start by describing how these properties are guaranteed in our proposal.

Data Access Control is reached during the auction. Indeed, the prosumer does not send to the smart contract the price p of the supply in plain text but sends the value $H(p\|r)$, where r is a random value. To violate the confidentiality of the price p, the adversary should either (1) break the one-wayness property of H or (2) guess the random r and use a brute-force approach. Both of these possibilities are unfeasible.

Concerning data integrity, the price p of the supply offered in the auction as $h = H(p\|r)$ cannot be modified. Suppose that, in the awarding phase, the adversary sends the values p_1 and r_1, with $p_1 \neq p$, thus trying to change the offered price. As the smart contract calculates $h_1 = H(p_1\|r_1)$, if $h_1 \neq h$, this attack is detected. Having that $h_1 = h$ with $p_1 \neq p$ is impossible because this would violate the second pre-image resistance property of cryptographic hash function [310]. Moreover, the integrity of the values sent to the smart contract cannot be tampered with, thanks to the immutability of blockchain transactions: when transactions are mined by the network, data contained into the transactions are stored and not modifiable any more.

The privacy of the users is obtained because the identity of the auction winner and the consumer is disclosed only after the end of the auction, in the last phases of the energy request/supply. Indeed, the auction participants do not use their main Ethereum address MEA, which is linked to their identity, but a Temporary Ethereum Address TEA that is randomly generated and used only for this auction. In effect, the reuse of blockchain addresses is strongly discouraged since the initial adoption of the blockchain technology [68]: Ethereum addresses are pseudo-anonymous and their reuse can favor the break of pseudo-anonymity of the owners. It is worth noting that not reusing the main address at each auction also contrasts an attack based on behaviour. Indeed, an attacker could track and link the activities of prosumers and consumers to gather useful information for predictive analysis based on energy consumption or the price offered for supply.

The authentication is achieved by using a challenge-response protocol, a protocol widely used for authentication [255], which is robust provided that the random number used as a challenge is generated from a sufficiently large domain and is never reused. The awarded prosumer has to link its TEA to its MEA. To do this, the prosumer signs by the TEA private key the value MEA, thus declaring its MEA. This association is guaranteed by the secrecy of the TEA private key. Consumers have also to disclose their identity when a request of energy is supplied by the winner prosumer. The robustness of this authentication depends on the corresponding robustness of the digital identity chosen. Indeed, our solution is orthogonal to the identification scheme. We suggest the use of a digital identity compliant with the *eIDAS* Regulation [139], which is recognized to be robust and provides a normative basis for secure electronic interactions among citizens and companies all over Europe.

Accountability is reached because all the operations of energy production, request, provision and payment are logged and stored in a public blockchain. By looking at the Blockchain transactions, it is possible to verify the behaviour of any entity. The accountability of the operations carried out in the entire environment avoids the arising of disputes among the actors: no one can claim something different from what has been reported on the blockchain.

The reliability of the solution is based on the features of blockchain. The robust Ethereum network counts a large number of nodes that work for keeping alive the network, ensuring the reliability of the blockchain-based solutions. Observe that, each actor is advantaged by well-behaving: indeed, participating in the auction requires a fee to be paid by every participant. This fee is not refunded in case of protocol violations. For example, the prosumer winner is discouraged from not providing the offered token because, in this case, the participation fee is not refunded by the smart contract (thus, protecting against attacks aiming at the denial of service).

3.4.1 Discussion on security vulnerabilities of smart meters

The Advanced Metering Infrastructure (AMI) is the aggregation of smart meters, communications networks and data management systems that are tailored to meet the integration if renewable energy resources in an efficient way [211]. AMI is responsible for collecting, analyzing, storing and providing measurement data sent by smart meters to authorized parties. So, they can process the data for demand forecasting, outage management, billing. It helps consumers to optimize power utilization knowing the real-time price of electricity. Also, it helps to acquire precious information about consumption of the consumers to maintain the reliability of the power system [176]. Smart meters play a crucial role in this infrastructure because they are fundamental to link peers of the network and to let them communicate to execute the steps of the solution. Smart meter is a stand-alone embedded system that contains a microcontroller that has some hardware and software features (e.g., non-volatile and volatile memory, analog/digital ports, timers, clocks) that can be used also in energy-demand side management [44]. Generally speaking, smart grids incorporates two different types of communication: Home Area Network (HAN) and Wide Area Network (WAN). The former uses protocols like ZigBee, Bluetooth and wired or wireless Ethernet, while the latter can communicate using WiMAX, cellular networks (2G, 3G, 4G, 5G) or fiber optics.The smart meter is in charge to act as a gateway between the in-house devices and the external world [44]. Clearly, due both to their crucial role played in the AMI and to the protocols they run, smart meters can be seen also as weak points of the overall chain because of some relevant security issues that are related to these two aspects. Since our proposal gives an important role to smart meters, we can say that this proposal suffers also from the same weaknesses. At the same time, if it is important to take into account the security of smart meters' issues, inherited from the implementation of protocols like ZigBee and/or GPRS+A5 by these devices [208], we want to underline that these possible threats are common to the whole AMI and smart grids infrastructure, and our proposal is orthogonal to them. Researchers and scientists are working hard to try to address, or possibly solve, these issues by proposing new countermeasures, enhanced protocols and solutions for known attacks [211, 32, 334, 194, 176].

Another aspect to take into account about smart meters regards their physical productions and distribution in the big market because they could be potentially untrusted devices. In addition, every component (both hardware and software) that shapes the whole process must be flagged and checked as trusted, otherwise every proposal (and not only the one presented in this study) could be considered untrusted. To reach the complete trustworthiness of the production and distribution of these devices I can suggest to implement some severe protocols in which a governmental entities is involved in order to check and verify the trustworthiness in every step.

3.5 Related work

Smart grid is one of the topics on which the research interest has observed an outstanding increase. The authors of [200] propose a complete survey about smart grids. In particular, after a clear explanation about the infrastructure, they focused their attention on smart metering and communication technologies and methods and on what is, in the smart grid scenario, security. Indeed, from this survey, it emerged that the most important area on which smart grids researchers should carry on is the one that let reliability, privacy and data protection increase. The paper [147] is one of the most appreciated surveys about smart grids in literature since authors looks at SGs from a technical point of view. In particular, they focused their studies on three different areas, such as *(i) smart infrastructure system*, *(ii) smart management system* and *(iii) smart protection system*. Concerning the first element of the list, it includes the information, energy and communication infrastructure on which SGs are based. Instead, the smart management system includes the tools and methods sets necessary for the advanced management of functionality of SGs that are developed on the smart infrastructure system. Finally, the smart protection system provides advanced methods for data and failure protection, reliability analysis and for facing cybersecurity issues.

In [140], authors proposes a work about different quality indicators in smart grids. In particular, their work aims at catching information from smart grids when multiple aspects, like energy level and network structure are jointly assessed. For this reason, the authors proposes an *Indicator Model* for combining a priori information about grids to estimate the service quality of the network in such a way different network configurations can be comparable. The authors of [385] propose an approach cloud based for smart grid applications since some features required by SGs, such as flexibility, scalability and omnipresent access are guaranteed by cloud computing. Anyway, a solution based on cloud computing for SG applications needs still some additional studies and it has open research issues in terms of, among others, security, reliability and robustness. Also in some past work we focused on smart grids, discussing the trade-off among energy, performance and availability [162] [126].

The survey [49] provides an overview of solutions exploiting the blockchain technology in energy sector. The authors classify the proposals into different categories based on the field of activity (e.g., e-mobility, grid management, decentralised energy trading), the platform used, and the relative consensus algorithm. They introduce security and identity management as a possible outcome of the blockchain technology in energy applications. They conclude that smart contracts simplify and make faster the cooperation and competition among energy suppliers. According to this result, our solution aims at protecting consumers' and prosumers' privacy by creating temporary Ethereum addresses (TEA) exploited for the auction phase. This way no information related to the real identity is exchanged before the agreement phase.

The authors of [35] focus on the security and privacy challenges of energy trading in smart grids. The proposed system, *PriWatt*, relies on Bitcoin and Bitmessage: the former technology guarantees security and privacy without the need of a third party, and the latter assures anonymity through encrypted messages in messaging streams. A system limitation regards the message redundancy in the communication necessary to guarantee high levels of privacy and security.

The system proposed in [381] is based on an Ethereum private blockchain that allows the participation of only authorized users. No identity management mechanisms are implemented but the access control and authentication are guaranteed through the blockchain's smart contract feature of restrict modifiers. In our smart contract we assure that only authorized users can run the functions through the modifiers but we assume that this is not enough. Indeed, during the system registration (see Section 3.2) we propose a challenge-response protocol to verify the Main Ethereum Address, which has been used to confirm the agreement between the parties.

The authors of [137] present a Secure Private Blockchain-based platform assuring the privacy of producers and consumers. While the producer can exploit different energy accounts, the consumers' privacy is preserved by changeable public keys of their smart meters. Nevertheless, to reduce the computation, the negotiation between the producer and consumer is conducted off-the-chain. This choice limits the security properties of energy bids that are not evaluated by the smart contract as our solution contemplates. Indeed, one of our strengths results in the creation of a blind auction managed by the smart contract, in a trusted way and avoiding unfairness among prosumers.

In [158], the authors propose a solution to implement traceable energy governance in Smart Grid Networks. The schema provides a transparent and traceable tracking of energy usage and consumption via the blockchain transactions. This proposal uses permissioned blockchain and super-nodes in charge of validating users' identities and activities. In contrast, our approach uses a public blockchain (Ethereum), which allows us to implement an auction without referees. The authors of [70] deal with Energy Storage Units (ESU) in smart grids. In their proposal, they use certified pseudonyms and smart contracts with no centralized authority. Despite the similarity of the above choices with those of our proposal, the focus of their studt is quite different. Indeed, it does not deal with energy trading but only with the problem of charging coordination to avoid blackout. The authors of [157] solve the problem of privacy in an energy trading scenario with a consortium blockchain-oriented approach. During the energy trading phases, the authors introduce a privacy-preserving module named Black Box Module (BBM), whose main principle is to create a mapping accounts for energy sellers. Again, the focus of the work is different from the solution proposed in this thesis because their proposal concerns the protection of data stored in blocks against linking attacks and malicious data mining algorithms.

In [335], the authors face the problem of privacy in the blockchain-based solutions for energy trading in smart grids. Their proposal is based on the function-hiding inner product encryption to match every bid with its bidder. However, this solution requires a central trusted entity, the Distributed System Operator, that acts as a mediator between the user and the network. In our solution, no centralized entity is required.

A smart and scalable distributed ledger system for smart grids is proposed in [66]. The authors analysed the properties of this new protocol and instantiated it in an electrical vehicles scenario. *Ecash* is the energy cryptocurrency of the system, used as a digital asset for energy transactions. These transactions are added in form of a directed acyclic graph. The validation of transaction is done by checking the balance amount of *Ecash* spent or used in the transaction and through the proof-of-Time instead of the classical and more used Proof-of-Work. If the transaction is validated by more than half of the total SmartChain then the transaction is considered valid. In the proposal, both the seller and the buyer will have a different chain where the transactions of the respective actors are stored. This proposal is opposed to the current solution relying on the already existing Blockchain technologies, as our schema does. Indeed, the authors of [66] design a new system inspired by the blockchain paradigm and aimed at meeting the limited computational resources of electric vehicles. In [121], an implementation of a blockchain-enabled Internet of Things approach for microgrids is presented. The study underlines the need for a system that considers the security and privacy of microgrid operations. The authors demand these security requirements to the IoT network, made of different energy devices, and to the microgrid central control which securely collects the data and transfers it to the blockchain. Then, the blockchain enables the IoT to provide the requested power. Even if the solution aims at providing the network with security properties, it is not clear how the blockchain enables these operations. The IoT network needs to be resilient and reliable to assure data security and privacy. Although, as declared, the data is only accessible for the respective area micro-grids, the problem of privacy still exists, because the energy request and supply are shared inside the same area. In our solution, a smart contract certificates the validity of energy transactions and distributes them to the blockchain network. Furthermore, the privacy of stakeholders is guaranteed from the first phase of an energy request. Only when the prosumer wins the auction and the consumer is willing to buy the energy, they reveal their real identities.

The start-up Grid+ [1] improves the retailing energy process by using the public Ethereum Blockchain. Every user can become a prosumer with the help of an Intelligent Agent, which is a computer in charge of an Ethereum node. This agent manages the BOLT token, which is the currency required to use the Grid+ platform, and the GRID token used to convert the kWh of power. In a decentralized energy system based on the blockchain technology, parties could create trading energy transactions inside an immutable and transparent platform.

The authors of [304] underline the security concerns related to the current centralized energy trading systems and explore the corresponding security issues in the proposed solution.

In [217], the solar energy distribution system Helios controlled by an Ethereum smart contract is presented. Helios allows participants to make available a quantity of energy measured and controlled by dedicated meters. It is also used a private Blockchain as local energy markets to define a private and permissioned setup and a predefined set of agents that have access to the system. The authors of [258] simulate a scenario composed of one hundred residential households in a local energy market. The micro-grid proposed in [203] is similar to a private blockchain network where nodes are only authorized entities. Each node is a smart home able to produce renewable energy, the electricity exchange that is trading is governed by a smart contract.

The authors of [240] propose an infrastructure based on blockchain that supports reliable and cost-effective transactive energy by enabling autonomous and decentralized energy trading among nodes of the network. Although the scenario is quite close to ours, the solution does not address the privacy policy and does not aim at preventing any kind of fraudulent actions.

3.5.1 Comparison

We conclude this section with the comparison between our proposal and the solutions of the state of the art carried out considering four aspects:

1. If a solution contemplates a *Blind Auction*.
2. If *Access Control* of bids is preserved.
3. If *Identity-Management* mechanisms or schema are considered.
4. If a solution preserves users' privacy (*Privacy-Preserving*).
5. How much the solution is scalable (*Scalability*).

In our comparison, Blind Auction, Access Control, Identity-Management and Privacy-Preserving are boolean measures, while we use the values *low*, *medium*, and *high* for Scalability. Specifically, solutions based on private Blockchains are labeled as low scalability; solutions exploiting consortium Blockchains have medium scalability, whereas scalability is high when public Blockchains are used. Table 3.2 summarizes the results obtained from our comparison.

Techniques	[35]	[381]	[137, 158]	[157]	[335]	Our proposal
Blind Auction	Yes	No	No	No	Yes	Yes
Access Control	Yes	Yes	No	No	Yes	Yes
Identity Management	Yes	No	Yes	Yes	No	Yes
Privacy-Preserving	Yes	No	Yes	Yes	Yes	Yes
Scalability	Medium	Low	Low	Medium	Low	High

Table 3.2: Comparison between existing solutions and ours.

This comparison allows us to claim that our solution outperforms the state of the art. Among others, our solution includes innovative features, as it does not require a centralized unit control, it manages the energy trading in a blind fashion until the agreement, and it does not exploit any external referee.

4

A new solution for service delivery with accountability and privacy requirements

In smart cities, providing better services is one of the most relevant achievement. New technologies can contribute to improving old services and adding new ones for citizens, so that the overall life-quality of people can continue to grow. Among these new technology, blockchain is very interesting for its several properties we discuss earlier. In particular, the main benefit of ethereum smart contracts is that different parties with conflicting interests can exchange value without trusting each other. As a matter of fact, solutions in which service delivery is regulated by smart contracts are proliferating. However, services sometimes could be negotiated and delivered only on the basis of some attributes, without disclosing the identity of the customer to the service supplier. However, accountability is still required, so that, in case of need, the identity of the customer should be linked to the service delivered and communicated to the appropriate parties. We propose a practical solution to the above problem that integrates the features of Ethereum with a (Ciphertext-Policy) Attribute-Based Encryption scheme. To show the effectiveness of our proposal, we instantiate the general model to a significant use case.

4.1 Introduction

One of the most relevant goals of smart cities is to provide better services to citizens both in terms of quantity and quality with respect to what happens now and what in the past. Indeed, cities are mostly evaluated and classified for what they provide to people: the better the service are, the happier the citizens are. Furthermore, there are emerging new needs and wishes from people living the city.

For this scope, it is very important to adequate and to design new projects so that the overall quality of life can grow up. Clearly, technology and the evolution of infrastructures can support this ambitious scenario.

We are now in a particular period in which the technological progress is growing and bursting into the every day life of citizens, providing them many opportunities and many choices. At the same time, the citizen often does not know exactly who the service supplier is, and this is an aspect to take into consideration since the user almost every time has to register into the platform of the service provider to join and retrieve the asked service. In addition, this registration phase coincides with the disclosure of personal and sensitive data, that is mandatory. Obviously, this opens up some critical issues regarding privacy and data protection. Some questions could arise. *Can the company be considered trusted? How the company collect and manage data? Is it really necessary to disclose such a level of private and identity-related data?* Indeed, it could be desirable not to disclose to the (potentially untrustworthy) service supplier other identifying information to prevent data misuse even in the less severe hostile case of an honest-but-curious provider.

There are many situations in which the service can be delivered to a customer only on the basis of some requirements the customer has to satisfy, such as the age, the professional title, the possession of a certain licence, without disclosing any further information.

To reach this goal, one could think of standard techniques based on anonymous credential [100], but, to be realistic, a solution to the problem of anonymous payment should be provided, together with an appropriate level of guarantee that anonymity does not compromise obligations, non-repudiability and accountability of the agreement.

To the best of our knowledge, no solution has been proposed for this general problem so far. The idea of this proposal is to leverage the power of smart contracts to obtain all the above requirements. Pseudonymity of Ethereum can ensure a good level of privacy, but the problem of implementing attribute-based contracts is not solved, at the moment, by native features of Ethereum. The direction we follow is the integration of a public attribute certification process (possibly based on the ecosystem designed by the eIDAS EU Regulation [358]) into the smart-contract features at a cryptographic level, thus by exploiting a Ciphertext-Policy Attribute-Based Encryption Scheme (CP-ABE) [77].

We observe that an attempt of bypassing the cryptographic link between attribute possession and Ethereum transactions would not provide an adequate result in terms of trustworthiness. Indeed, we should require that an entity of the application (maybe a smart contract) should obtain by a Third Trusted Party (the Attribute Provider, in the eIDAS system) the proof of the possession of certain attributes for a given pseudonymous individual. This implicitly requires full trust in this node, concerning the assessment of attributes. In contrast, our solution requires that only the party certifying the attributes (assumption fully accepted in eIDAS) and the Private Keys Generator (PKG) of the CP-ABE are trusted parties, which are parties external to the application. Moreover, often attributes can be certified by Government bodies, which could also play the role of Private Keys Generator.

It is worth noting that, in our solution, the link between attributes and real-life identity is known only by the party certifying the attributes. Moreover, all the actions are immutably recorded over the blockchain and, in case of need, the link between the pseudonymous used in a certain transaction and the real-life identity of the customer can be disclosed by collecting information from the PKG and the parties certifying the attributes. This guarantees the accountability requirement of our solution.

In the smart city scenario, several proposal involved privacy-preserving authentication schemes. In particular, they are often proposed in the Vehicular Ad-hoc Networks (VANETs) and in intelligent transportation systems, where security and privacy are issued that must be addressed and taken into consideration. The biggest scope of these algorithms regards the preservation from the disclosure of identity data. From this perspective, they seem to be well suited for our aim. However, we can not take into consideration privacy-preserving authentication schemes because we need to implement an attribute-based access control mechanism where we ask the customer to satisfy a certain policy (that can be summed up as a set of attributes) for each of the services required.

4.2 The proposed solution

In this section, we present the architecture and the solution we propose to allow service delivery only to users having certain requirements, by preserving their identity. In Table 4.1, the entities involved in the scenario are depicted: we have a user U who needs a service s provided by one of the several available service suppliers (SS). Moreover, we have several Attribute Providers (AP), which are in charge of checking if a user fulfils or not some specific attributes. Finally, we have the Public Key Generator PKG, which is the Trusted Party issuing ABE private keys.

We introduce some preliminary background notions.

1. let SS be the service supplier providing the service s;
2. let $\mathcal{A}$ be the access structure [71] representing the policy associated with the service s;
3. let $P = \{a_1, \dots, a_n\}$ be the attributes of U that are compliant with the policy $\mathcal{A}$; [1];
4. we denote by $OW(a_i)$ the Attribute Provider that is in charge of checking if U fulfills or not the attribute a_i;

[1] For the sake of presentation, with an abuse of notation, we use $\mathcal{A}$ meaning the policy represented by $\mathcal{A}$.

U	User
s	Service
SS	Service Supplier
$\mathcal{A}$	Access Structure
a_i	i-th attribute
$OW(a_i)$	Attribute Provider of a_i
P	set of attributes
PKG	Public Key Generator
Eth_x	Ethereum Address of x
T	Transaction
pk	ABE public key
sk_P	ABE private key for P
$Encrypt(PK,M,\mathcal{A})$	ABE Encryption
$Decrypt(CT,SK)$	ABE Decryption
$KCK(M)$	Keccak digest of M

Table 4.1: Notation.

5. we denote by Eth_x an Ethereum address owned by x, where x can be U, SS, or a smart contract;
6. we model an Ethereum transaction T as a tuple $\langle id_T, Eth_{src}, Eth_{dest}, data\rangle$, where id is the identifier (usually, it is the digest of the transaction), src and $dest$ are the sender and receiver of the transaction, resp., and $data$ is the payload. Observe that, transactions include also an additional field `value`, which is not relevant for our scope, so that it is not considered here.
7. $KCK(M)$ denotes the Keccak digest of the message M.
8. Setup(k): This algorithm receives a security parameter k and returns a public parameter PK and a master secret key MSK.
9. KeyGen($MSK;P$): This algorithm takes as input a set of attributes P and the master secret key MSK. It outputs a private key SK associated with S.
10. $Encrypt(PK,M,\mathcal{A})$ denotes the encryption of the message M under the policy $\mathcal{A}$ (see item (2) above).
11. $Decrypt(CT,SK)$ denotes the decryption of the ciphered message CT with the ABE private key SK.

Now, we describe the steps carried out by the different actors of our scenario. These steps are also summarized in Figure 4.1.

Step 1: service request. First, the user U asks for the service s supplied by SS. For this purpose, U generates an Ethereum transaction $T_1 = \langle id_{T_1}, Eth_U, Eth_{SS}, data_1 \rangle$, where id_{T_1} and $data_1$ are the identifiers of the transaction and the service s, respectively.

Step 2: challenge start. When SS receives the service request T_1, the service supplier first checks whether Eth_U is not *revoked* (the detail about how this check is implemented is given in Step 5). If Eth_U is not revoked, SS acknowledges the request and generates a *challenge* needed to verify that the user is able to prove the possession of all the required attributes to satisfy $\mathcal{A}$. In particular, SS creates the new policy $\mathcal{A}'$ requiring, in addition to the conditions expected by $\mathcal{A}$, also the possession of the attribute a_{n+1}, where a_{n+1} represents the possession of the Ethereum address Eth_U. In words, the new policy enforces that the user has to satisfy the requirements of the policy $\mathcal{A}$ and, moreover, she/he owns the Ethereum address Eth_U. This is done in such a way to guarantee that only the user with address Eth_U can overcome the challenge.

The challenge starts with the generation of the transaction $T_2 = \langle id_{T_2}, Eth_{SS}, Eth_{SC}, data_2 \rangle$, where SC is a smart contract and $data_2 = \langle X_0 = id_{T_1}, X_1 = P', X_2 = Encrypt(PK, R, \mathcal{A}'), X_3 = KCK(R), X_4 = Eth_U \rangle$, where $P' = P \cup \{a_{n+1}\}$ and R is a secret value (the challenge solution) big enough to prevent from guessing k. In words, $data$ contains the reference to the first transaction done by U, the set of attributes, the challenge consisting of the encryption of a value R that can be deciphered only by users satisfying the policy $\mathcal{A}'$, and the digest of R computed by the Keccak function (the solution verification). The smart contract receives the transaction and *waits* for the user reply.

Step 3: challenge reply. The user U looks for the transactions made by SS with $X_0 = id_{T_1}$ (i.e., in the first input of `data` field) – thus, the transaction T_2. By using id_{T_1}, U can find the challenge and extract $X_1 = P'$.

In order to prove the possession of all the attributes in P', U needs the ABE private key associated with the attributes P'. The task carried out to obtain this key is the following.

1. U contacts PKG and asks for the ABE private key associated with the attributes P' (message 2 in Figure 4.1);
2. PKG computes the set $AP = \bigcup_{x=1}^{n+1} OW(a_x)$, which is composed of the attribute providers of the attributes involved in P';
3. now, each $AP_i \in AP$ performs a challenge-response-based authentication with the user U (messages 4 and 5);
4. In case of successful authentication, if the user owns the attributes requested, AP_i generates an *assertion* of attribute certification for PKG (message 6), which is signed by AP_i to guarantee integrity and authenticity. Moreover, AP_i stores the

mapping between the user identity and the assertion identifier, which can be used in case of revocation or accountability;

5. After collecting all the attributes certifications, PKG invokes KeyGen($MSK;P'$), which generates the ABE private key $sk_{P'}$ for U. PKG sends this key to U (message 7) and stores the mapping between the received assertion and the Ethereum address, again for accountability or revocation reasons.

Observe that this procedure is carried out only the first time U needs the ABE private key: indeed, this key will be used also for the next accesses to services with the same policy $\mathcal{A}'$.

Now, U extracts X_2 from T_2, calls Decrypt(X_2,$sk_{P'}$), and obtains R'. In order to demonstrate the knowledge of the ABE private key and, consequently, to prove the possession of the attributes P', U generates another transaction $T_3 = \langle id_{T_3}, ETH_U$ $, ETH_{SC}, R' \rangle$.

Step 4: agreement. The call to the smart contract starts the automatic check of the challenge reply. The smart contract acts as follows:

1. finds the *pending* challenge for the Ethereum address Eth_U, and retrieves $X_3 = KCK(R)$;
2. extracts R' from T_3;
3. computes $R^* = KCK(R')$;
4. if $R^* = X_3$, then U overcomes the challenge and the grant for providing the user U with the requested service is given.

Step 5: revocation. This step is performed whenever a user U loses an attribute a_i. In this case, the attribute provider $OW(a_i)$ searches for all the assertions previously sent to PKG mapped to U (these are stored into a map - see Step 3): this list of assertions is sent to PKG as *revoked* assertions.

PKG receives this list and searches for the Ethereum address mapped to each assertion. Then, an Ethereum transaction of revocation to each of these addresses is generated, reporting the revocation of the ABE key associated with this Ethereum address. This way, the list of revoked Ethereum addresses is stored on Ethereum and can be used in Step 2 to check if the Ethereum address of the user requiring the service has been revoked.

4.2.1 Case study

To better understand our proposal, we instantiate it into the real-life scenario in which a citizen wants to rent a car from a car sharing company.

Let suppose that John wants to rent a car (i.e., the *service*, in our scenario) s at the company Car4U (which plays the role of *service supplier* SS). As Attribute-Based

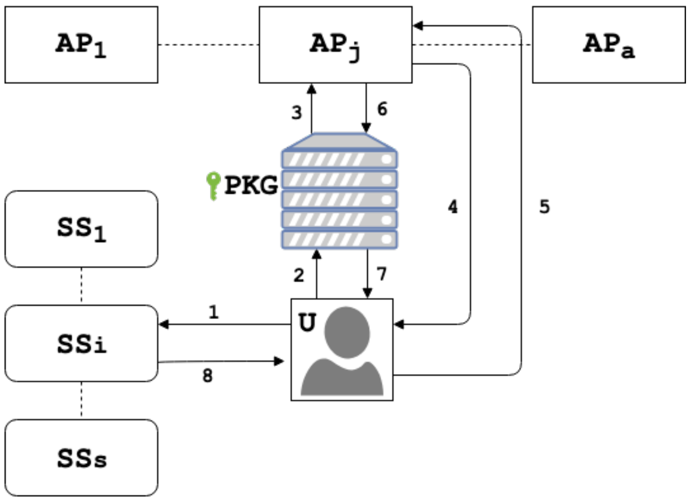


Fig. 4.1: Scenario and steps of our solution.

Encryption, we adopt a solution derived by the scheme [77]. Thus, we consider given a cyclic group $\mathbb{G}$ with order p, a generator g of $\mathbb{G}$, an hash function $H : \{0,1\}^* \to \mathbb{G}$ and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_{\mathbb{T}}$.

The Public Key PK and the Master Secret Key MSK are calculated as $PK = (\mathbb{G}, g, h = g^{\beta}, e(g,g)^{\alpha})$ and $MSK = (\beta, g^{\alpha})$, where α and β are two random elements $\in \mathbb{Z}_p$.

For the sake of presentation, we consider the case in which the service supplier policy consists of only one requirement. In particular, the access structure $\mathcal{A}$ of the policy for renting a Car4U car consists of the attribute a_1 *having the driving license*. Furthermore, the company *Car4U* has deployed its own smart contract described in Section 7.3.

Now, we detail the use case proposed above.

Step 1: service request. Once John has decided the car to rent, he generates an Ethereum transaction T_1 to Car4U. We assume that the Ethereum address of Car4U is public and extracted by the site, or a QR code, or in a similar way. The payload of the transaction (i.e., the data field) contains the license plate of the car to rent as service identifier.

Step 2: challenge start. Let suppose that John's address has not been revoked: *Car4U* acknowledges the request and sends a *challenge* needed to verify that John satisfies the policy $\mathcal{A}'$, which means a_1 *having the driving license* and a_2 *having the Ethereum address* Eth_{John}.

Now, Car4U picks up a random $R \in \mathbb{G}_{\mathbb{T}}$ and encrypts R under $\mathcal{A}'$. The encryption algorithm picks up two random $r, s \in \mathbb{Z}_p$. The ciphertext is: $X_2 = (\tilde{C} = Me(g,g)^{\alpha s}, C = h^s, C_1 = g^{r+s}, C_2 = g^{2r+s}, C'_1 = H(a_1)^{r+s}, C'_2 = H(a_2)^{2r+s})$.

Finally, Car4U generates a transaction to call the function cStart of the smart contract having in the payload $\langle id_{T_1}, \{a_1, a_2\}, X_2, KCK(R), Eth_{John}\rangle$.

Step 3: challenge reply. Now, John requests to the Public Key Generator PKG the ABE private key sk_{a_1,a_2} built from the attributes a_1, a_2. Observe that $OW(a_1)$ is the Motor Vehicle Office, which is in charge of checking whether a user has a driver license. In order to show to be the owner of Eth_{John}, John is required to sign a challenge by the Ethereum private key associated with Eth_{John}: thus, the role of $OW(a_2)$ is played by PKG. Now, John proves his identity to the Motor Vehicle Office: this is done by an eIDAS-compliant identification scheme. Once the identity of the person is verified, the Motor Vehicle Office can check whether John has a driving license. If this is the case, the Motor Vehicle Office sends to PKG a SAML assertion [239] and stores the mapping between the assertion identifier and the identity of the user.

If John proves also to be the owner of Eth_{John}, PKG calculates the ABE private key for John: PKG picks up the randoms $t, t_1, t_2 \in \mathbb{Z}_p$. The decryption key is: $sk_{a_1,a_2} = (D = g^{\frac{\alpha+t}{\beta}}, D_1 = g^t H(a_1)^{t_1}, D_2 = g^t H(a_2)^{t_2}, D_1^{'} = g^{t_1}, D_2^{'} = g^{t_2})$. This key is sent to John and PKG stores the mapping between the assertion identifier and Eth_{John}.

Now, in order to decipher X_2 (extracted from T_2), John computes: $F_1 = \frac{e(D_1,C_1)}{e(D_1^{'},C_1^{'})}$, $F_2 = \frac{e(D_2,C_2)}{e(D_2^{'},C_2^{'})}$, and $A = \frac{F_1^2}{F_2}$.

Then, the deciphered text is obtained as: $R' = \frac{\tilde{C}}{e(C,D)/A}$.

It is easy to check the decryption procedure. $F_1 = \frac{e(D_1,C_1)}{e(D_1^{'},C_1^{'})} = \frac{e(g^t H(a_1)^{t_1}, g^{r+s})}{e(g^{t_1}, H(a_1)^{r+s})}$. Since g is a generator, $H(a_1) = g^k$ for some k. Thus, $F_1 = \frac{e(g^t g^{kt_1}, g^{r+s})}{e(g^{t_1}, g^{k(r+s)})} = e(g,g)^{t(r+s)}$. Similarly, $F_2 = e(g,g)^{t(2r+s)}$. Therefore, $A = \frac{F_1^2}{F_2} = \frac{e(g,g)^{2t(r+s)}}{e(g,g)^{t(2r+s)}} = e(g,g)^{ts}$ and $\frac{\tilde{C}}{e(C,D)A} = \frac{Me(g,g)^{\alpha s}}{e(h^s, g^{\frac{\alpha+t}{\beta}})/e(g,g)^{ts}} = M$. Finally, John calls cResponse of the same smart contract and sends the deciphered value R'.

Step 4: agreement. According to the called function, the service is grant if $KCK(R')$ is equal to $KCK(R)$.

Step 5: revocation. Let suppose now that John's driving licence is revoked. Since John has asked for and obtained the ABE private key before the revocation, he is able to pass the challenge for rent a new car, and, thus, he could rent a car without a driver license.

To solve this problem, when the driver licence of John is revoked, the Motor Vehicle Office sends to PKG the identifiers of all the assertions sent to PKG in the past related to John. PKG, which knows the Ethereum addresses related to these assertions, generates an Ethereum transaction to John's Ethereum address having in the payload a field *type* sets to *revocation* and a field *address* sets to this Ethereum address. In such a way, if John tries to request a service to any service supplier, this request will be discarded because his Ethereum address is contained in a transaction of revocation stored on Ethereum (recall that this check is done by the service supplier in Step 2).

4.3 Implementation details

In this section, we describe the implementation of our solution. In order to test our proposal, we implemented the adopted CP-ABE scheme in JAVA language, leveraging the libraries [128]. This was necessary to include, in our system, some lightweight implementation of the scheme, tailored with our specific scenario, in which policies are single domain-dependent attribute (this results in concrete two-attribute policies since each policy must include also the Ethereum address as second attribute).

Furthermore, we designed also the smart contract used to ask for a service and decide whether to grant it. The smart contract is written in Solidity [27, 125], a high-level Turing-complete and object-oriented language.

The smart contract, whose code is reported in Listing 4.1, stores by `Owner` (Line 3) the Ethereum address of the service supplier, which is initialized with the address of the party that deployed the smart contract. The `struct data` (Lines 4-10) represents the skeleton of the challenge that must be won by the user to obtain the requested service. Mappings in Lines 11-12, are used, respectively, to save pending transactions (on which there is an open challenge) and to retrieve the challenge from a given transaction.

In Solidity, the modifier can be seen as an extension of a function and it is used to check a condition prior to executing the function. So, we implement the `modifier OnlyOwner` (Lines 17-20) in the `function cStart` (challenge start) (Lines 22-26) in such a way the condition that has to be checked is related to the sender of the transaction. In particular, `OnlyOwner` requires that the sender must be the owner of the smart contract, otherwise the function cannot be executed.

The function `cStart` can be called only by the service supplier and is used to reply to a *service request* of a user. In particular, the input of the function is the challenge, which is saved in an instance of `struct data` (Line 23); moreover, we set the mapping `txs_pending` for the given transaction `tx` to *true* so that, from now on, there is an open challenge for that specific request . Finally, we map the given transaction `tx` to this challenge.

The function `cResponse` (challenge response) (Lines 27-45) is called by the applicant of the service to reply and, possibly, win the challenge. This function receives the address of the challenge transaction and the challenge solution `r1`. First, if there is an open challenge for the given transaction (Line 28), we verify that the applicant who is calling the function is actually the target of that challenge (Line 30), and compare `keccak256(r1)` with `kck_R` of the challenge (Line 32). In case of success, the challenge is no more pending (Line 33) and the service can be grant (this part is application dependent so we omitted its implementation). In case any of the previous checks are not passed, the function ends (`revert()`).

We implemented this smart contract by *Remix - Solidity IDE* [25] and used *Ropsten* [26] as testnet, with the support of *Metamask* [23], which consists of a browser extension that allows us to run dApps (decentralized applications) directly on the browser without running a full Ethereum node. The deploy of the contract on the Ropsten Test Network costs 844 Micro(ETH) (in April 2019, this is about 0,13 $), the function `cStart` costs 644 Micro(ETH) (about 0,11 $) and the function `cResponse` costs 24 Micro(ETH) (about 0,0037 $). From this analysis, we can say that the implementation of our solution is feasible and cheap.

```
pragma experimental ABIEncoderV2;
contract Granting {
  address owner; //the service supplier
  struct data{ //the challenge
    bytes32 tx;
    string[] attributes;
    bytes32 encrypted;
    bytes32 kck_R;
    address user;
  }
  mapping(bytes32 => bool) public txs_pending;
  mapping(bytes32 => data) public fromTx_toData;

  constructor () public {
    owner = msg.sender;
  }
  modifier onlyOwner(){
    require (owner == msg.sender);
     _;
  }

  function cStart (bytes32 tx, string[] memory attributes, bytes32 encrypted, bytes32 kck_R, address user)
   public onlyOwner {
    data memory d1 = data(tx, attributes, encrypted, kck_R, user);
    txs_pending[tx]= true;
    fromTx_toData[tx] = d1;
  }
  function cResponse(bytes32 tx, uint256 r1) public {
    if(txs_pending[tx]==true){
      data memory d2= fromTx_toData[tx];
      if (msg.sender==d2.user){
        bytes32 rs= keccak256(abi.encodePacked(r1));
        if(rs==d2.kck_R) {
          txs_pending[tx]= false;
          //Grant the service
        }
        else { revert();}
      }
      else { revert();}
    }
  }
}
```

Listing 4.1: The code of the smart contract

4.4 Security aspects

In this section, we briefly discuss the security properties of our solution.

Let start by defining the *adversary model*. In our analysis, we assume that both PKG and Attribute Providers are trust parties and that the attacker can be a user, a service provider or external to the system. In addition to the standard security assumptions (unbreakability of cryptographic primitives and blockchain security properties), we assume that private keys and secret information are not disclosed by the owners and that users and service suppliers do not collude each other. The goal of the attacker is to break at least one of the following security properties: access-control, privacy, accountability, unlinkability, availability, non-repudiation.

Access control is reached. Indeed, to win the challenge, the user has to decipher a challenge by an ABE private key that PKG issues only to users who prove the possession of the attributes required by the policy.

The privacy requirement is that a service supplier is not aware of information identifying the user accessing the service. This goal is reached because the architecture of the solution allows the service supplier to know only the Ethereum address of the user, which is generated by the user and appears a random string. The level of protection of such information is that of pseudonymity given by blockchain, so it is not absolute, as de-anonymization is in general possible. However, for our specific context, unlike cryptocurrency transfers, if the user wants to protect her/his privacy by using always *one-shot* blockchain addresses, the above attacks on pseudonymity are not applicable.

Accountability is obtained by merging information coming from different parties. If we want to know the identity of the person who used the service s, we start from the transaction T_1 used to require the service and extract the Ethereum address used by this person. This can be done only by leveraging public information. Then, since PKG stores the mapping between Ethereum address and corresponding assertion, and any Attribute Provider stores the mapping between assertion identifier and digital identity, it is possible to disclose the identity of the person who used s.

Unlinkability is guaranteed provided that a user exploits a new Ethereum address for each service request (as discussed earlier, this measure makes ineffective also de-anonymization attacks).

Attacks on Availability are contrasted. The only attack that can be performed is a DoS, in which an attacker floods a service supplier with superfluous requests thus trying to overload it and prevent legitimate requests from being fulfiled. However, since any service request transaction has an even small price in Ether, an attack of this type would be very expensive.

Non-repudiation is obtained. Indeed, each action (service request, service grant, etc.) is logged into the Ethereum blockchain and it can be verified. Moreover, transactions are signed by an Ethereum private key, which is kept only by the owner of the address. Again, Ethereum transactions cannot be modified after they are validated. Moreover, the integrity of transactions is guaranteed by the properties of public blockchains. Thus, no party can repudiate an action.

4.5 Related work

The concept of ABE (Attribute-Based Encryption) was first proposed by Sahai and Waters in [314]. They presented their scheme as a new type of Identity-based Encryption in which the identity was formed by a set of attributes. More in detail, a user with an identity formed by the attributes w is able to decrypt a ciphertext encrypted with attributes w', if the *distance* between w and w', calculated according to some metric, is small. So, it is allowed a sort of error tolerance that makes it suitable for biometric applications. In [169], the first KP-ABE scheme was proposed, where a policy is associated with the decryption key and the attributes are associated with the ciphertext. In order to decrypt the ciphertext, the attributes have to match the policy. Similarly, in [77], the notion of CP-ABE was formalized, where the policy is associated with the ciphertext and the attributes with the key. A scheme that combines CP-ABE with KP-ABE was proposed in [51]. In this scheme, both policy and attributes are associated with the ciphertext and with the key. Regarding the key, the attributes are related to the user and the policy states which type of ciphertext the user can decrypt. On the other hand, regarding the ciphertext, the attributes are related to the latter and the policy states which type of user can decrypt the ciphertext. An evolution of ABE is ABPRE [232]. In [231], a CP-ABPRE scheme is presented that uses a semi-trusted proxy to transform a ciphertext encrypted under a certain policy to another ciphertext under a different policy.

Platforms which rise with the aim to share services among consumers have the need of a Trusted Third Party (TTP) [82], a Decentralised App (DAPP) based on Ethereum smart contracts can resolve the requirement of having a trusted intermediary. The authors propose a DAPP for the sharing of objects and all the processes of renting are ruled by the smart contract.

In service marketplaces scenarios, a blockchain that runs smart contracts can enable the concept of trustless intermediation. The need of considering a trusted figure, who plays the role of trusted intermediary, lays on the service markets nature: the authors of [214] present a distributed approach to the problem and propose a new concept of decentralized and trustless service marketplace. In [186], the authors consider the delivery of physical assets, the main requirement is establishing, through a smart contract, a series of agreements between the involved parties; this way, accountability of actions is preserved. In *Lelantos* [46], a blockchain based anonymity-preserving physical delivery system is proposed. This system can offer to consumers the fair exchange of services and the unlinkability of operations. Obviously the actors can operate in a anonymous way using a pseudonym revealed to the others parties, and all the processes are ruled also by a smart contract.

In [352], the author provides an overview of Ethereum and the principle of operation; moreover, open problems are listed and treated, differentiating the level of abstraction. The paper [42] collects different proposals on smart contracts, related to security, privacy issues, codifying and performance. Several platforms for smart contracts are compared and applications are examined. There are many technical and social implications in using smart contracts: [277] investigates the advantages of machine-readable smart contracts. In particular, this paper focuses on smart contracts and the management of their lifecycle. The author highlights existing gaps in industry solutions using smart contracts and proposes their solution. In [160], a SWOT analysis of blockchain is drafted to outline the advantages and disadvantages in using this technology in different areas. In particular, authors focus on the insurance field and highlight the scenarios in which it can be worthwhile to improve blockchain and smart-contract applications.

5

A new data access control mechanism based on smart contracts

In the smart-city paradigm, data sharing is one of the pillars needed for its full implementation. Among the other aspects, we refer to the opportunity for users (citizens, companies, organizations) of exploiting data sources managed both by institutional parties and third parties involved in the smart-city life. Open-data is an answer to above need, but, sometimes data cannot be disclosed publicly, coming to the concept of closed data. *In this case, access control takes a fundamental role. Since we deal with a highly open and dynamic environment, a certain level of accountability should be guaranteed to contrast misbehaviour and solve possible legal controversies. In this study we propose a solution based on the combination of Ethereum smart contracts, eIDAS-based attribute and identity management, and the distributed file system IPFS for the access control of the* closed data *in smart cities.*

5.1 Introduction

In these years we are spectators of a fast and incredible technology revolution that involves everything that surrounds us, from mobile devices to cars with autonomous driving, from the development of smart grids to new communication protocols. In this new world, there is a need of a new vision of what cities are and what cities should provide to populations. One of the features that smart cities should provide is the easy yet secure access to data which represent the substrate of the smart-city life. Although a lot of attention has been devoted to interoperability and open-data [114], larger space of investigation exists in the domain of *closed-data*, regarding different aspects, ranging from the way data sharing is implemented, to how the access is controlled, and how to assign responsibilities to the different involved parties, with the aim to make the access effective but accountable. We try to give a concrete solution leveraging the power of Ethereum smart contracts, the Interplanetary File System (IPFS) and the eIDAS European Regulation Ecosystem [358]. The idea is to implement on top of the above components, an Attribute-Based Access Control mechanism (ABAC). As a matter of fact, ABAC represents the emerging approach for large environments. Gartner predicts that by 2020, 70% of organizations worldwide will have moved to the ABAC model. However, one of the main issues to deal with for ABACs is how to assess attributes in a feasible way. In this study, we propose an approach in which institutional bodies are responsible for attribute certification, according to the eIDAS paradigm of *Attribute* and *Identity Providers*, and access control enforcement is done in a trust way, thanks to Ethereum smart contracts.

5.1.1 Attribute-Based Access Control

Access control is one of the hottest and trend topics of the last decades in the IT world. Indeed, to guarantee a right, secure and non-invasive mechanism that enable or not the access to the resource is always necessary. In particular, an access control mechanism should satisfy, among the others, the following properties: it does not have to allow non-authorized people to access the resource and it does not have to deny authorized people not to access the resource. The type of access must be subject of authorizations as well.

There exist many families of access control, such as *Mandatory Access Control* and *Discretional Access Control*, for what concerns the flexibility of authorization rules, *Role-Based Access Control*, *Context-Based Access Control*, *Attribute-Based Access Control*, etc., for what concerns the way to associate authorizations to subjects.

In particular, Attribute-Based Access Control (ABAC) is defined as an access control mechanism in which authorization is computed evaluating the fulfillment of one (or more) required attribute. The National Institute of Standards and Technology (NIST) defines ABAC as follow [190]: *"An access control method where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions."*

As a matter of facts, after some decades of supremacy Role-Based Access Control (RBAC), ABAC is emerging as definitely more suitable than RBAC to large environments, in which defined roles must be set by the management and associated with people, resulting in what is known as *role explosion*. In our context (i.e., smart city), ABAC is the elective choice.

5.1.2 InterPlanetary File System

InterPlanetary File System (IPFS) is a peer-to-peer distributed file system that connects all computing devices with the same system of files [370]. It provides a high-throughput content-addressed block storage model, with content-addressed hyperlinks. IPFS employs content-addressing to uniquely identify each file in a global namespace connecting all devices. Furthermore, it identifies, verifies and transfers files relying on the cryptographic hashes of their contents [337]. It also integrates technologies such as self-certifying namespaces, an incentivized block exchange, and distributed hash tables (DHT), etc. IPFS is built around a decentralized system of user-operators who own a portion of the total data, creating a resilient system of file storage and sharing. As a consequence of this decentralized approach, IPFS has not a unique point of failure, and nodes do not need to trust each other as well.

The IPFS user, when uploads a file to the system, will have back a unique cryptographic-hash string (IPFS-identifier of the document) through which she/he can retrieve the file every time and everywhere. Indeed, it is not required that the user stores the original file in her/his devices, but it is enough to know the IPFS-identifier of the document to obtain it. The hash string can be seen as a Uniform Resource Locator (URL) of the World-Wide-Web.

5.2 Scenario and motivation

In the widest interpretation of the concept of a smart city, one of the main challenges is to guarantee a secure, trusted and fast data sharing. This may have a significant impact both in open-government policies that are crucial in smart cities [386], and in the smart fruition of information to deliver complex services that need multiple data sources.

In this scenario, it is fundamental to implement an access control mechanism that is able to decide who can read what, by taking into account the fact that we operate in an open environment, in which interested subjects cannot be predetermined. It is worth noting that, although smart cities should provide *open data*, which are accessible with no limitation, also *closed data* are relevant for the full implementation of smart-knowledge-based communities. Therefore, access control becomes necessary.

To better explain, consider the following example. Suppose that some closed data are produced by a smart city entity like a hospital or a court. Since we are talking about sensitive data, it is reasonable to think of some policies for which only people belonging to the medical board (in the case of healthcare data) or lawyers (in the case of law data) can access.

As seen above, there are many access control models but, among all, due to the open nature of our scenario, Attribute-Based Access Control (ABAC) seems the most suitable one because neither identities nor roles are able to capture all the conditions which should be satisfied by subjects to access information. Moreover, ABAC allows us to implement anonymous-credential mechanisms to avoid that sensitive data of subjects are disclosed to possibly untrusted parties and, thus, to preserve privacy.

Obviously, if we think about a smart city and its data it is clear that it is necessary to ensure properties like accountability, privacy, trust and non-repudiability (among the others). In this sense, Ethereum blockchain perfectly fits with these requirements. Another motivation that led us to choose Ethereum is that the mechanism of verification of attributes must be trusted. Furthermore, the usage of an Ethereum smart contract enforces the attribute-based access control mechanism without any privacy leakage, since attribute-based authorizations are anonymous and prevent any disclosure of personal, sensitive, and not required information.

Anyway, Ethereum, as every blockchain, is not the most suitable platform for sharing and storing large files since the blockchain is replicated on many nodes and a lot of storage space is required without serving an immediate purpose [337]. Moreover, the blockchain becomes bloated with data that has to be propagated within the network and the price of operating blockchain nodes increases because more data needs to be processed, transferred and stored. File sharing platforms can be leveraged to solve these problems. Users can easily share large files and still benefit from the blockchain.

InterPlanetary File System (IPFS) is a particularly interesting protocol peer to peer file sharing platform that combines file sharing and hashes. Cryptographic hashes serve to securely identify a fileâĂŹs content. IPFS makes it possible to store and share large files more efficiently and it is based on cryptographic hashes that can easily be stored on a blockchain. Unlike existing cloud storage, IPFS has the advantage that data is distributed and stored in different parts of the world and not on a central server [370]. Finally, a solution exploiting IPFS guarantees data availability.

5.3 The proposed access control mechanism

In this section, we propose our solution regarding the scenario discussed above and at the same time we carry out a use-case in which applying the proposal.

First, we define all actors involved in it, then we present all the steps needed to reach our goal.

Our idea is the following. Suppose that the smart city produces, owns and provides data that it wants to share not to everybody but only to who fulfils some requirements. To be more concrete, but without loss of generality, we describe our solution in the healthcare setting, although it can be easily extended to every typical context of smart cities (e.g., transport, commercial and law data, etc.). In this scenario, we consider the user as a doctor that has to prove its role and the possession of such attribute in order to retrieve some medical information. In our proposal, we assume that documents stored on IPFS are already encrypted and, for this reason, objects of our access control authorizations are keys instead of final resources. Otherwise, the document could be easily read via a generic IPFS client.

In our solution, we define the following actors:

- User U, a citizen that asks for data;
- Identity Provider IP, whose task is the management and verification of user identities;
- Attribute Provider AP, whose task is the management and verification of user attributes;
- Access Service Provider ASP;
- Publish Smart Contract PSC, an Ethereum Smart Contract used for the publication of documents on IPFS;
- Access Smart Contract ASC, an Ethereum Smart Contract used for verifying the policy and, where appropriate, for granting the key for the decryption of the document;
- Oracle O, that is used by ASC for checking the validity of the certificate of U;
- Content Manager CM, who is in charge of encrypting documents, publishing them on IPFS, associating them to the right policy and addressing the key-request when U fulfils its requirements;
- IPFS, used for storing data in a distributed way;
- Ethereum, a public blockchain allowing the development of smart contracts.

Once defined all entities involved in our proposal, let's describe the steps of our solution:

1. **Policy Setup**
   In this first phase, the CM associates the document d_i with the policy $\hat{p}$. If $\hat{p}$ does not exists, CM will generate it compliant with the $XACML$ standard.
   In particular, every category has a set of attributes related to and, as a consequence, a different policy. In our scenario, we refer to the category of healthcare data, where, for the sake of presentation, the attribute to be fulfilled is *to be a doctor*.
2. **Encryption**
   CM encrypts d_i with a symmetric encryption function (such AES) by using the key related to the policy associated with healthcare data. Indeed, in accordance with the above, every category (and every policy) has a different encryption key. Let us denote by $\hat{k}$ the key associated with the policy $\hat{p}$. Now, CM obtains the encrypted document e_i.
3. **Publication**

The goal of this step is to publish on IPFS the encrypted document e_i and the related policy $\hat{p}$. This could be achieved through different ways. Indeed, CM could simply publish it directly with an IPFS client. Anyway, accordingly to the accountability features aimed by our proposal, we allow the publication only trough the Ethereum smart contract.

In detail, CM calls a function of the PSC through her/his Ethereum address ETH_{CM} with e_i as input obtaining as output the IPFS-hash h_i related to e_i. At the same time, CM calls another function of the same smart contract PSC to publish on IPFS the policy $\hat{p}$, if it still does not exist in the Ethereum environment, thus obtaining the IPFS-hash $h_{\hat{p}}$.

At this point, CM maps h_i with the corresponding policy $h_{\hat{p}}$ on the PSC. The result is a mapping between the policy $\hat{p}$ and the list of documents associated with. Furthermore, there is a mapping between the area of interest (in this case, healthcare) and $\hat{p}$.

4. **Attribute Verification**

In this phase, the user U requests ASC the policy that she/he has to satisfy related to healthcare, obtaining $h_{\hat{p}}$. Now, similarly to the *Publication* phase, U could obtain the document directly with an IPFS-client, but again, our protocol enforces U to get it only via ASC. At this point, U knows $\hat{p}$ and she/he can see that the attribute required is *to be a doctor*. We remind that $h_{\hat{p}}$ is the IPFS hash that stands for the identifier of the document, while $\hat{p}$ is the real data. When we ask for an IPFS hash we obtain back its content. For the assessment of the attributes owned by users, we apply an eIDAS-based approach [358], in which a SAML-2 authentication process is established to involve both an Identity Provider and one or more Attribute Providers which are institutional entities responsible for providing information (like title, licences, qualifications, age, etc.) about digital identities. To be compliant with real-life regulations, we cannot imagine that every user plays the role of the service provider in an eIDAS authentication loop. Therefore, we introduce an intermediate service, called Access Service Provider (ASP), needed to perform the authentication request and to obtain the valid corresponding assertion. In detail:

- U goes to ASP to request, by playing the role of Service Provider the assertion in which there is the information certifying the attribute *to be a doctor*;
- through a SAML2-compliant schema, ASP forwards the request to the Identity Provider IP;
- after that, IP contacts the Attribute Provider AP (in the use case, the medical board) asking for the certification of the attribute *being a doctor*;

- now, AP sends the reply to IP, which will contact ASP to communicate the information obtained through an assertion. Finally, ASP returns to U the assertion and the nonce related to. In particular, the nonce allows applications to correlate the identifier of the assertion with the initial authentication request and it is used also to avoid replay attacks (see Section 5.4).

ETH_U sends a transaction to ASC calling a function in which she/he puts the hashed identifier of the assertion and the nonce as input. The smart contract, now, has to verify the validity of the assertion and, as a consequence, the real possession of the attribute required by the policy $\hat{p}$. To do that, ASC invokes the Oracle O, that is in charge of checking the overall validity of the previous steps with IP. Indeed, an Ethereum smart contract is not able to interact with the external world but it is limited to what happens inside the blockchain. For this reason Ethereum enables this connection with the usage of oracles. An oracle is a service that receives requests from the blockchain, computes the answers through the external world and retrieve back the information to the network. Clearly it is possible that oracles collects garbage from the external world and they put it into the network. The "garbage-in garbage-out" paradigm is a possible weakness of this blockchain. Solving this situation is not the scope of this proposal, but it is still necessary to take into account countermeasures. For examples, many proposals deal with the possibility to query multiple oracles at once and to accept the answer that is retrieved by the majority of these oracles. This is clearly in line with the collaborative and majority approach on which blockchain is based. In this particular scenario, the oracle O is in charge to query th IP to verify the validity of previous steps.
If the check succeeds, ASC emits an event in which it confirms the satisfaction of the policy.

5. **Key Granting**
   In this step, CM sees the event of success on the blockchain (it can be done via a client application that is able to show, and possibly filter, events) and CM sends a transaction blockchain to ETH_U with the information about $\hat{k}$. Obviously, before sending the transaction, CM should encrypt the key to prevent from its disclosure to all blockchain. To reach this goal, CM encrypts $\hat{k}$ with U's public key, obtaining $E^U_{pk}(\hat{k})$. The result is an ethereum transaction from CM to ETH_U through the ASC having as `input_data` $E^U_{pk}(\hat{k})$. We remind that `input_data` is an optional field of an ethereum transaction that can be used to share other information.

Finally, U is the only one who can decrypt, with her/his private key, the chipered key $E^U_{pk}(\hat{k})$. After the derivation of $\hat{k}$, U is able to dechiper also e_i, thus obtaining d_i.

5.4 Security aspects

In this section, we briefly analyse the main security aspects that are involved in our proposal.

Let us begin with the definition of the *adversarial model*, that is particularly relevant for our proposal because by modeling the role of attackers, with their capabilities and goals, we could help to improve the cyber defense [134] and, since we are facing the case of closed data in the smart city scenario, it is necessary to ensure that some fundamental security evaluations are valid. In detail, in our proposal, we assumed that the Content Manager, the Identity Provider and the Attribute Provider are trusted parties and that the attacker can be either a user or the Access Service Provider.

In particular, the Access Service Provider could operate in a malicious way since it could give the wrong assertion to the wrong users. Anyway, this attack is contrasted by using the SAML-2 standard because the *Authentication Request* and the *Authentication Response* must coincide, so the ASP is not able to give the wrong response to the wrong applicant.

In sum, we do not require the ASP more trust then that one required by the eIDAS regulation. In addition to the standard security properties and assumptions, such as those ones related to the Ethereum blockchain and IPFS protocols, we assume that user's secret information and data are not disclosed to the public world and that users do not collude each other as well.

The goal of the attacker is to break, at least, one of the following secure properties: availability, non-repudiation, accountability, integrity and confidentiality.

In our proposal, data are stored on IPFS while their identifiers are stored on the Ethereum blockchain. This combination of these two technologies contrasts attacks on availability. Indeed, the usage of IPFS avoids the central and unique point of failure, since data are duplicated on multiple and random IPFS peers. Moreover, the DoS attack, in which the attacker floods the Ethereum network with a huge amount of requests, would be very expensive because every Ethereum transaction and every call to a function of an Ethereum smart contract has a cost in terms of gas.

Non-repudiation is obtained. In fact, every action is logged into Ethereum and it can be verified at any time and, in addition, Ethereum transactions are not editable after been mined. They are also signed by the Ethereum private key, that is known and kept only by the owner of the address. Furthermore, non-repudiation is ensured by our protocol also during the phases involving the publication of documents and policies on IPFS and the downloading of such files from IPFS. In particular, although these operations could be carried out by using a standard IPFS client application, we implemented an alternative approach, based on smart contracts, enforcing the non-repudiability of the overall protocol.

We can distinguish two different domains of interest: that one on the blockchain and the other one off-chain. Concerning the former, accountability is ensured similarly to the non-repudiation property. Instead, if we think to the off-chain side of our proposal, accountability is reached because only the Identity Provider and the Attribute Provider(s) know exactly the link between the identity of the user and its related Ethereum address. So, if for any reason it is necessary to reveal this mapping, it could be done by merging information from different parties.

Data integrity is, again, reached thanks to IPFS by the usage of the hash that is carried out for every document published on the InterPlanetary File System. Confidentiality is obtained as well, because sensitive and *closed* data are ciphered by the Content Manager and the decryption key is given only to those users that fulfil the policy associated with them. Furthermore, the Content Manager, before sending to the user the key, encrypts it with the ethereum public key of herself/himself, that can be easily derived from the Ethereum address. So, even if the Content Manager sends the key by using Ethereum, nobody can actually understand it excepts for the interested user.

Finally, our protocol reaches the goal of privacy requirement, since the Content Manager is not aware of personal and sensitive information about users except for their attributes and their Ethereum addresses. In this case, the level of protection of these data is that one related to the pseudonymity provided by the Ethereum blockchain itself, that is not full. However, if the user wants to preserve better her/his privacy, she/he can generate a new Ethereum wallet for every operation, making attacks on pseudonymity not possible.

5.5 Implementation issues

To implement our proposal, we used many different technologies and framework to integrate IPFS and XACML with ethereum smart contracts.

In particular, these are, among the others, the most relevant ones:

- *RemixIDE* [25], an online IDE for the development of ethereum smart contracts;
- *Truffle Suite* [28], a suite of tools useful for interfacing smart contracts (e.g., Ganache);

- *Metamask* [23], a browser extension that allows us to run decentralized applications (dAPPS) on the browser without running a full Ethereum node;
- *Web3.js* [29], a lightweight JavaScript library for integration with Ethereum clients;
- *Provable* [24], the most known oracle used for Ethereum.

For the sake of presentation, we show only the most interesting details we faced during the implementation of our solution and we miss some other details.

First, we developed a JavaScript web-page containing a form allowing the submission of files on IPFS that interfaces with the smart contract showed in Listing 5.1 in a transparent way.

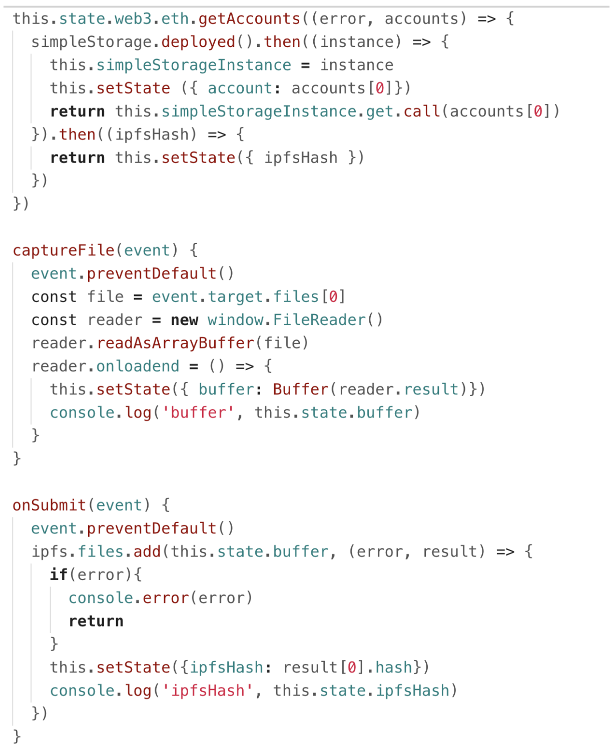
```
pragma solidity 0.5.8;

contract SimpleStorage {
   string ipfsHash;

   function set(string memory x) public{
      ipfsHash = x;
   }

   function get() public view returns (string memory){
      return ipfsHash;
   }
}
```

Listing 5.1: Code of the smart contract `SimpleStorage` used to publish on IPFS

Moreover, the web-page returns the IPFS-cryptographic-hash identifier of the document submitted and all this operation is permanently stored in Ethereum. This is done thanks to the JavaScript code shown in Figure 5.1.

In particular, after the connection with the library web3js, `captureFile` is the function to buffering the file once submitted on IPFS and `onSubmit` is used for the acquisition of the hash computed by IPFS. The operation carried out for the submission of the policy on IPFS via smart contract are the same. In Listing 5.2 there is the portion of the smart contract that is in charge of mapping the IPFS-hash of the document and its related IPFS-hash policy.

```
address content_manager;
mapping(string => File) public fileMap;
string[] public filePolicy;

constructor () public {
   content_manager = msg.sender;
  }

modifier onlyCM(){
   require (content_manager == msg.sender);
    _;
  }

function createMapping(string memory ipfsHash, string memory policyHash) public onlyCM{
   fileMap[ipfsHash].ipfsHash = policyHash;
   fileMap[ipfsHash].policyHash = policyHash;
```



```
this.state.web3.eth.getAccounts((error, accounts) => {
  simpleStorage.deployed().then((instance) => {
    this.simpleStorageInstance = instance
    this.setState ({ account: accounts[0]})
    return this.simpleStorageInstance.get.call(accounts[0])
  }).then((ipfsHash) => {
    return this.setState({ ipfsHash })
  })
})

captureFile(event) {
  event.preventDefault()
  const file = event.target.files[0]
  const reader = new window.FileReader()
  reader.readAsArrayBuffer(file)
  reader.onloadend = () => {
    this.setState({ buffer: Buffer(reader.result)})
    console.log('buffer', this.state.buffer)
  }
}

onSubmit(event) {
  event.preventDefault()
  ipfs.files.add(this.state.buffer, (error, result) => {
    if(error){
      console.error(error)
      return
    }
    this.setState({ipfsHash: result[0].hash})
    console.log('ipfsHash', this.state.ipfsHash)
  })
}
```

Fig. 5.1: Portion of the code of App.js

```
        filePolicy.push(ipfsHash);
    }
```

Listing 5.2: Code of the mapping between the document and its policy

It is worth noting that the function `createMapping` can be called only by the Content Manager (in this case, the deployer of the smart contract) thanks to the modifier `onlyCM`. Indeed, the modifier is a function of Solidity that, when it is added to the declaration of a function, limits the access to the function itself to those users who satisfy its requests.

Another interesting aspect about the implementation regards the request of the policy from the Ethereum smart contract to the IPFS network because the policy is written with XACML, an XML-like language and there is need to parse the result. This has been solved as shown in Listing 5.3.

```
    pragma solidity 0.5.8;

    import "./Oraclize.sol";

    contract Policy is usingOraclize{
        bytes32 public oraclizeID;
        string public results;
        event LogOraclizeResult(string result);

        function ipfs() payable{
            OAR = OraclizeAddrResolverI ( "address" );
        }

        function getAttribute() payable {
```

```
        oraclize_query("URL", "xml(https://ipfs.io/ipfs/"IPFS-hash identifier").AttributeValue");
    }

    function __callback(bytes32 myid, string result) {
        results = result;
        LogOraclizeResult(result);
    }
}
```

Listing 5.3: Code of the smart contract query with parser XML

Now that the smart contract has obtained the attribute (or attributes) required by the policy, it compares it to the attribute that is in the certificate presented by the user to the same smart contract. If the information completely overlaps, then the Content Manager generates a transaction to the user in which she/he specifies the key associated with the policy after encrypting it with the user's ethereum public key.

5.6 Related work

In this section, we investigate the state of the art regarding data access control on distributed information in a smart city environment.

For smart cities, cloud computing has become an important infrastructure as it can provide secure and reliable data storage and sharing. However, in the cloud storage system, the cloud server cannot be considered completely reliable. Therefore, several studies have focused on access control for smart city data using the cloud. In particular, the study [144] proposes a revocable access control scheme of cloud data for smart cities. They design a proxy-assisted access control framework for multi-authority cloud storage system and they construct a new multi-authority Chipertext-Policy Attribute-Based Encryption (CP-ABE) scheme with efficient decryption to realize data access control in the cloud storage system, and design an efficient user and attribute revocation method for it.

In [164] an advanced solution is proposed, which is based on Virtual EnviRonment (CLEVER) enabled for CLoud. The purpose of this proposal is to regulate user access to certain areas and to provide useful data for business intelligence oriented to multipurpose management. In particular, it aims to collect data on people's access and electricity consumption to provide information and services for public, private or governance use. The study [222] presents an Integrated Component for Cloud Services (ISCS) that enables secure and trusted access to data and related services in the cloud. The ISCS controls and handles access-related aspects such as authentication, authorization and registration. It is realized using OAuth and OpenID.

Always in the context of smart cities, several studies were carried out relating to access control and IoT. The IoT protocols must provide data security, in particular, they must guarantee data updating, integrity, confidentiality, authentication, and access control. The access control is perhaps the most important aspect of intelligent cities since the unauthorized access to critical infrastructures can endanger the inhabitants of smart cities (i.g. unauthorized access to traffic lights can cause accidents and traffic congestion and cause huge financial losses). Access control also faces several challenges such as limited resources of IoT devices, often with a limited power budget. Furthermore, access control must provide a high degree of scalability. The study [321] analyzes use cases of smart city and defines requirements of access control for the smart city IoT platform. Attribute-based access control is also analyzed to satisfy the requirements. The requirements of internal access control of smart city IoT platform are analyzed in-depth and an access control mechanism based on information flow history is proposed to control information flow between components of the platform. The most promising work in this area is on Delegated CoAP Authentication and Authorization Framework (DCAF) [163] and Capability Based Access Control (CBAC). CBAC as proposed by HernÃąndez-Ramos et al. solves the scalability issue of access control by decentralizing the validation of permissions, yet the AS remains a single point of failure and a possible bottleneck. The study [98] proposes an efficient format for capacity tokens that is used completely without status and decentralized. This allows deploying access control in scenarios of previous CBAC implementations and DCAF are impossible.

Unfortunately, the data collected and processed by IoT systems are vulnerable to threats of availability, integrity, and privacy. The work [248] takes advantage of the blockchain technology for the protection of privacy and the secure IoT data sharing in smart cities. The blockchain network is divided into various channels to preserve data privacy; each channel includes a finite number of authorized organizations and processes a specific type of data such as health, smart car, smart energy or financial details. Access to usersâĂŹ data is controlled by embedding access control rules into smart contracts and data within a channel is further isolated and protected using private data collection and encryption respectively. A reward system in the form of a digital token is also proposed for users who share their data with interested parties/third parties.

Many studies have focused on the use of blockchain technology as an access control manager for distributed systems. In [247] an approach based on blockchain technology is proposed to publish the policies that express the right to access a resource and to allow the distributed transfer of such right among users. Each user can know the policy associated with a resource and the subjects who currently have the rights to access the resource because the policies and the exchange of rights are publicly visible on the blockchain. This solution allows distributed auditability and a possible working implementation based on XACML policies is also shown. The authors of [338] use a modified version of the InterPlanetary Filesystem (IPFS) that exploits Ethereum's smart contracts to provide file-controlled access to files. IPFS interacts with the smart contract whenever a file is uploaded, downloaded or transferred.

Moreover, the authors of [88] proposed a solution based on Attribute-Based Encryption (ABE) and the Ethereum blockchain for facing the problem of service delivery with accountability and privacy requirements.

In the paper [122] the authors propose a blockchain-based framework, called Ancile, that allows safe and efficient access to medical records by patients, suppliers, and third parties, while preserving the privacy of patientsâĂŹ sensitive information. Ancile uses smart contracts in an Ethereum-based blockchain for greater access control and obfuscation data, and employs advanced cryptographic techniques for added security. The document shows how blockchain technology can be exploited in the health sector to achieve the delicate balance between privacy and accessibility of electronic health records. In [92], the authors integrate the Ethereum blockchain and the Identity-Based Solution (IBE) by using the Public Digital Identity to overcome the blockchain limitation regarding the fact that the recipient of transactions must be signed up to the blockchain before using it. Indeed, in this work, authors allow transaction between subject not yet registered to the system.

To the best of our knowledge, our proposal is the only one that tries to exploit the advantages of both smart contracts and ABAC into a smart city scenario with distributed information.

6

A proposal to enable propagation in Web-of-Trust

Web of Trust offers a way to bind identities with the corresponding public keys. It relies on a distributed architecture, where each user could play the role of certificate signer. With the widespread diffusion of social networks, the trust propagation is a matter of growing interest. This chapter proposes an approach enabling the propagation in Web of Trust by means of Ethereum. The usage of Ethereum eliminates the necessity of single-organization trusted services, which is, in general, not realistic. Although the information stored on Ethereum is public, the privacy of users is protected because trust chains involve only Ethereum addresses and strong measures are implemented to contrast their malicious de-anonymization. The approach relies on the usage of a smart contract for storing the status of certificate signatures and to manage revocations. When a user u wants to trust another user v, the smart contract checks the presence of trust chains originating from root nodes of u.

6.1 Introduction

The widespread diffusion of social and recommendation systems have experienced exponential growth in recent years. These systems offer very attractive means of social interactions and communications, but also threats for security concerns. Confidentiality, for example, is weakened by the lack of key management frameworks that are able to bind social identities with the corresponding public keys. Consequently, the risk of malicious events is very high. For this reason, we believe that more effective solutions and mechanisms are required when users, in open environments, rely on public key encryption to obtain security services. For example, a user should get answers to the following questions: "is the person I am talking to really the one she/he claims to be?", "who ensure the trust level of my recipient?", "is there somebody embodying the recipient?".

The design of a central authority, which is trusted by everyone, is often not applicable in these contexts. On the contrary, *Web of Trust* [159] ensures a higher level of flexibility since it adopts a distributed approach that better suits the nature of the context we are referring to. Indeed, Web of Trust offers a way to bind identities with the corresponding public keys in the form of certificates without relying on central authority and exploiting the direct trust between users. In Web of Trust, users have the capability to sign each other's certificates (i.e., the couple identity - public key), and this mechanism originates a directed trust graph in which arcs represent signatures. When a user needs to obtain information about a certificate issued by an unknown user, he/she has to check for the presence of one or more trusted parties in the list of signatures associated with that certificate.

Although Web of Trust allows in principle trust propagation, its direct implementation into the current architecture would require either the adoption of certificates with size exponentially growing with propagation or trusted servers to which users delegate trust chain verification. With no propagation, no trust is required for servers. Distributed ledgers offer a solution to avoid trusted central authorities and to guarantee the storage of shared information in an immutable and distributed way. In this chapter, we propose an approach that enables trust propagation in Web of Trust and exploits Ethereum to work as a public key infrastructure holding the list of signatures and to implements trust management. This approach matches the current state of Pretty Good Privacy (PGP) [394] public key server infrastructure. The result is a system where users can sign certificates of other users and can retrieve information about the trust level associated with a certificate by consulting the blockchain. Moreover, the proposed solution does not require the disclosure of social identities both for the signature and for the verification of trust phases, since it is based on users' pseudonyms given by the corresponding Ethereum addresses.

6.1.1 Web of Trust

Nowadays, we are spectators of an incredible growth of online social networks (OSNs). Unfortunately, at the same time, risks and attacks towards these systems are increasing [78, 116]. The risks associated with OSNs are of a different type. In our study we focus on the trust propagation, that is orthogonal and complementary to some other problems such as the identification of fake accounts to contrast social attacks. In fact, a lot of works can be found in this area ([91, 116, 379] to cite a few). Instead, in this work, we propose an approach that enables trust propagation in the Web of Trust for secure communications by exploiting Ethereum and the smart contract properties.

One of the most known methods that provide a mean to trust the association between identities and public keys is the "Web of Trust". It has been firstly proposed in the context of Pretty Good Privacy (PGP) in 1991 by Phil Zimmermann [394].

PGP offers authentication and privacy protection for data communication and Web of Trust which consists of a decentralized method for certifying a given PGP public-key certificate. Indeed, people can sign each other's public key so that progressively, and dynamically, they create a network of interconnected links and signatures [31]. When someone needs to trust the public key of an unknown user, she/he has to verify the set of signatures and of the signatures' identities associated with it. The result is that each person will have a different and subjective idea of the other one based on signatures of her/his certificate. This is due to the figure of "introducers", who are the trusted people whose signature represents a trustworthiness guarantee for a PGP certificate. The set of introducers is chosen by each person, so influencing the personal perception of the network identities.

In this context, it is fundamental to compute and manage, in a timely manner, the trustworthiness of both PGP public-key certificate and introducers. According to [31], the former can have three levels of trustworthiness (i.e., *undefined*, *marginal* and *complete*), while the latter can be *fully*, *marginal*, *untrustworthy* or *don't know* trusted. Moreover, since Web of Trust is generally subjective, every user is able to resolve her/his scepticism by tuning suitably two thresholds that are related to the minimum number of introducers' signatures that need to present a certificate to be considered as *complete* by a user.

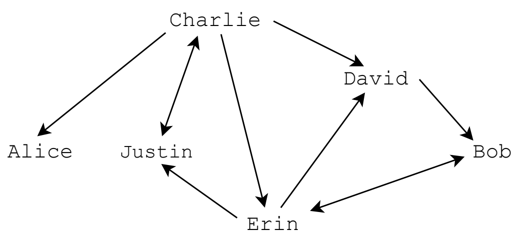


Fig. 6.1: Signatures in Web of Trust

To explain the basic idea behind the Web of Trust, let us consider the scenario depicted in Figure 6.1. A total of 6 users are interconnected by a set of edges representing signatures of public keys. The origin on an edge represents the signer, while the destination is the signed party. The edge is bidirectional when a mutual signature is present. For example, *Charlie* signs the certificates of *Alice*, *David*, *Erin* and *Justin*, while *Bob* is signed by *David* and *Erin*. The reader can deduce the other signatures by following the edges.

In this scenario the verification of a user's public key can be asked from each user after the definition of her/his own set of introducers. For sake of simplicity, we do not consider the levels of trustworthiness. It means that, for example, *Alice* could trust the *Bob*'s public key if *David* and/or *Erin* belong to the set of *Alice*'s introducers and she has settled for 1 signature. If she needs at least 2 signatures, both *David* and *Erin* needs to belong to her set of introducers; if she needs more than 2 signatures to trust a user's public key, she will never trust *Bob*'s public key.

Furthermore, what happens if *Alice* wants to verify *Bob*'s public key and her introducer is only *Charlie*? In our previous example, there is a path of signatures (of 2 steps) from *Charlie* to *Bob*. PGP is equipped with a parameter, named `CERT_DEPTH`, to establish the maximum length of the certification chain; unfortunately, this value is often not used in real applications since it can be quite difficult to use in an appropriate way [179, 31]. This led to decrease in the interest of researchers in working on new solutions for the propagation of trust. Anyway, since we are talking about Web of Trust, we think it is necessary to enable propagation to make the most of PGP, especially in OSN applications where the number of users is very high. In this sense, we can start from the assumption on "objectiveness" of trust, such that, if *Alice* fully trusts *Charlie* as her introducer, she is trusting, at the same time, his actions and his decisions. As a consequence, if *Charlie* considers *David* trustworthy, then *Alice* will consider *David* trusted as well.

In a standard PGP scenario, enabling propagation leads to keep an updated and trusted certification chain in each PGP member's certificate, and, moreover, it requires to store full trust chains on PGP servers. This is, clearly, totally in contrast with the Web of Trust principles. For this reason, the rest of the chapter will describe our solution that exploits Ethereum instead of PGP servers.

To the best of our knowledge, few works proposed ways to enable propagation in Web of Trust or use a blockchain-based solution to implement PGP and trust evaluation mechanisms. In [178], for example, the authors proposed a way to further expand the trusted neighbourhood. The basic idea relies on the possibility to sign a user's certificate with the couple of values $\{+1, -1\}$, where -1 represents a signer that believes the certificate is not authentic. Starting from these values, the authors provide the necessary formulas evaluating a user's feedback. Other existing works proposes the usage of blockchain to secure Trust Management system for authentication. For example, in [39] the authors formally model such systems as trust graphs and explore how the usage of a blockchain can mitigate attacks. In [301] the authors proposed a formula to calculate the trust degrees between two users in an e-commerce as a combination of direct and indirect trust degrees. Starting from this formula, the authors in [103] exploits blockchain as a mean to store the necessary information. The work in [382] proposes a framework supporting fast propagation of certificate revocation and elimination of man-in-the-middle risk by using blockchain. With respect to all these works, our proposals exploit blockchain to store certificate signatures and smart contract to verify the presence of trust chains enabling propagation in Web of Trust. The proposed solution also contrasts the malicious de-anonymization of social identities, in fact the trust propagation stores only information about Ethereum addresses. Moreover, the introduction of a smart contract does not ask clients to be full nodes (i.e., a node that stores the full blockchain and participates in block validation) and to perform expensive computations to navigate the trust graph. The smart contract also introduces economic disincentives to malicious behaviours (such as Denial of Service and Sybil attacks). In the next section, we describe the details of our Ethereum-based proposal for enabling propagation in Web of Trust.

6.2 Description of our proposal

Since we need a trusted and distributed mechanism to overcome these issues, we decide to implement a blockchain-based solution enabling the propagation in Web of Trust. The core of our proposal resides in the representation of a Web-of-Trust-based trust model, allowing the propagation of trust among a domain of public key associated with real-life identities. Even if we assume that real-life identities operate on an OSN context, the proposed solution is general enough to be applied in other contexts. The user needs to publish their public keys (e.g., on their own social network profiles). We remark that the focus of this proposal does not regard the problem of impersonation and fake profiles in social networks, for which a wide literature exists, and the existing approaches and techniques can be orthogonally applied together with our solution. We highlight that we use the blockchain technology as a mean to propagate trust in the network, while we need to involve also public keys because the aim is to trust identities.

According to the Web of Trust model, every user u elects a number of *introducers*, who are persons *objectively* trusted for u. From the point of view of trust propagation, public keys (actually, certificates) associated with the introducers play the role of *root certificates* whenever u wants to verify trust paths. More formally, we define a set of users U and a function $f_i : U \rightarrow 2^U$, which, for any user $u \in U$ returns the set $f_i(u) \in 2^U$ of the *introducers* of u.

As highlighted in Section 6.1, in order to avoid the necessity of single organization trusted services implementing trust propagation and certificate revocation, we design and implement an Ethereum-based solution. Thus, we refer to another domain of identities, which is composed of the set of Ethereum addresses. The idea is that the trust graph is built via an Ethereum smart contract SC, it is stored also into the state of SC and it is managed through the functions of the same smart contract.

Let denote by ETH_{SC} the Ethereum address of SC. The smart contract SC includes the following functions: $sign$, $verify$, and $revoke$. The exact definition of these functions will be explained throughout this section. The status of SC is composed of a directed graph G_s of Ethereum addresses, a list L_r of revoked Ethereum addresses, and a data structure storing the number of failing attempts of Ethereum addresses if not null (this point will be explained in the Trust Verification process below).

For instance, the user u_i has his Ethereum address ETH_{u_i}. The smart contract SC contains the information and the functions needed to carry out our proposal such as G_s, that represents, via the mapping data structure, the graph of friendships and trust among users.

We represent Ethereum transactions as tuples

$$\langle src\_address, recipient\_address, data \rangle$$

where $src\_address$ denotes the Ethereum address of the sender, $recipient\_address$ denotes the Ethereum address of the receiver, and $data$ is the field including additional information (allowed in Ethereum). In our representation, we do not make explicit the fact that any transaction is signed by the sender by means the Ethereum private key, and we omit some information related to specific features of Ethereum (e.g. GAS price). We highlight that we do not define a generic representation of smart-contract events because the structure of any event can be defined by the smart-contract designer in terms of both structure and content. In the following, we list the content of events into tuples.

Now, we describe how we map, in our model, the basic functions of Web of Trust, which are *certificate signature*, *trust verification*, and *certificate revocation*. We want to highlight that each of the following operations calling the smart contract can be performed only by those Ethereum addresses that have not been revoked yet since the smart contract will filter all the illegitimate requests received.

For the sake of clarity, when we say that a user u *trusts* another user v, we mean that u declares that the user v (actually, her/his certificate C_v) is the real owner of her/his public key. According to Web of Trust, there exists three different levels of trust, that are `COMPLETE` , `MARGINAL` , `UNDEFINED`. To be realistic, we assume that users trust only reciprocally because we consider that trust is required as a preliminary step of an interaction between two users that do not know each other. When the user u signs a certificate C_v, we say that u *gives trust* to v.

So, as shown in Figure 6.2, each user is characterized by the following attributes:

- a unique identifier for the social network;
- a pair of PGP keys (private and public);
- an Ethereum address;

- a list of Introducers.

The Ethereum address of the user can be trusted, via transactions, by other Ethereum addresses.

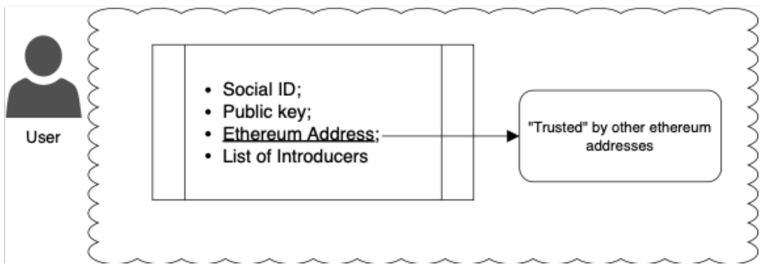


Fig. 6.2: Attributes of the user

Let suppose that Alice wants to know if Bob's public PGP key is trusted with respect to her list of introducers, and vice versa, so Alice is the first who asks for knowing the other's Ethereum address. First, they should demonstrate each other to own both an Ethereum addresses and the PGP key pair. Then they should check the trust of the specific Ethereum address by using the propagation of web-of-trust protocol. Anyway, since disclosing the Ethereum address is susceptible of some attacks (the attacker could reconstruct the trust graph), the protocol has to take some countermeasures.

Let us introduce here a parameter K_{ij} that represents the minimum number of signers that a generic user i requires to consider the other peer j as "trusted". Hence, in our example, K_{BA} is the minimum number of signers that Bob requires to consider Alice trusted while K_{AB} is the minimum number of signers that Alice requires to consider Bob trusted. For example, if $K_{BA} = 5$, it means that Alice must have at least 5 people that previously signed her certificate (by trusting her Ethereum address). In the case in which Alice has less than the minimum number of signers, the protocol terminates before Bob reveals his Ethereum address to Alice. It is very useful to prevent from failures or attacks. In particular, since it is more likely that the attacker is the one who asks first for the knowledge of the Ethereum address of the other one, our protocol obliged that this person (in our case Alice) must be the first also who discloses its Ethereum address.

We remark that K_{BA} is sent by Bob to Alice via social network and there is no motivations that led Bob to lie. Indeed, if the transaction is carried out with the correct random but with wrong K_{BA}, the whole protocol stops.

Certificate Signature.

This phase is carried out when a user u wants to sign the certificate C_v of another user v. She/he has to generates an Ethereum transaction $T_s = \langle ETH_u, ETH_{SC}, ETH_v \rangle$ calling the function *sign* of the smart contract SC, whose effect is to update the status SC by inserting the arc (ETH_u, ETH_v) in G_s, provided that ETH_v is not in L_r. Observe that, as said before, this operation can be carried out only by those Ethereum addresses who are not in the revocation list L_r.

Trust Verification.

The goal of this process is to obtain that the users u and v know the reciprocal Ethereum addresses and, at the same time, trust each other. Observe that the second result is reached only if, once the Ethereum addresses have been disclosed, the smart contract verifies that the trust paths starting from those addresses satisfy the policies required by the users.

We have to distinguish two main domains of interest: the social one and the ethereum one. Following the above example of Alice and Bob, here we summarized the steps needed:

We underline that if a user loose its ethereum address or its social profile the protocol continues to work. The only case an attack can success is when both the profile are lost by the owner.

Let now see how our protocol works.

First, let suppose that Alice wants to know weather Bob's PGP public key is trusted for her. PGP is based on the Web-of-Trust paradigm, which requires that at least someone who Bob trusts has previously signed Bob's PGP public key. We now add and introduce the feature of propagation of trust. In our proposal we admit not only directed signature from trusted people, but also indirected ones. In addition, we base our approach on the Ethereum blockchain, so now the ethereum addresses take part of the solution. Let denote by ETH_a Alice's ethereum address and by ETH_b Bob's one. We also know that everything that is stored or processed in blockchain is transparent and visible, it is important to try to find a way to not disclose the Ethereum address to the other peer until we are not sure about her/his trustworthiness, otherwise she/he would have all information about my relations. At the same time, it is no possible to check the trust without knowing the Ethereum address. This problem is actually difficult to solve because it could seem like the chicken and egg problem.

We propose an approach that try to overcome this issue. First, we assume that, in a conversation between unknown, it is more plausible that the possible attacker is that one who starts the dialogue. This assumption let us to find a solution for the previous problem.

Let suppose that is Bob that starts the conversation with Alice. He must be the first one to disclose his ETH_b to Alice. Although we are in the first step of the solution, it would be nice to trying to stop possible malicious attempts already in this phase. For this reason, we introduce the parameter $K_{B,A}$ that represents the minimum number of signers that Bob requires to consider Alice as trusted. Now, both Alice and Bob agree on a random R that is signed by the private key and encrypted with the other's public key. This let us to guarantee both data origin authentication and confidentiality. We obtain $R_B^{\prime A}$ signed by Bob with his private key and encrypted with Alice's public key and the vice versa $R_A^{\prime B}$.

At this point Bob discloses his ethereum address by calling a function of the smart contract giving $R_B^{\prime A}$ and $K_{A,B}$ as input parameters. The smart contract verifies first that Alice has got at least $K_{A,B}$ signers (1) while Alice, off-chain, verifies $R_B^{\prime A}$ (2). If (1) successes, the smart contract emits an event of success, otherwise there will be emitted an event of failure, so interrupting the whole process.

At the same time, if (2) fails, Alice blocks the protocol. In the case of both success, Alice can verify the trust of ETH_b. Since the control is on the ethereum blockchain, Alice could decide to use a temporary ethereum address ETH_{a_temp} giving as input parameters her list of introducers and ETH_b. If the verification step successes, Alice can share her real ETH_a to Bob, that will be now verified by himself through the same procedure (reversed).

In case of success of both verification, Alice and Bob obtained the information that they can trust each other and their communication can be considered trusted as a consequence.

Certificate Revocation.

When a user u wants to revoke her/his certificate, she/he has to generate a transaction to SC from her/his ethereum address ETH_u by calling the function *revoke*, which changes the status of the smart contract by adding ETH_u to the list of revoked ethereum addresses L_r. From this moment on, every trust path that passes through ETH_u will be considered as invalid and, moreover, if another user w will have ETH_u in her/his $f_i(w)$, the smart contract will filter this list of introducer by removing ETH_u (and possibly others revoked).

Furthermore, as we said before, there is the case in which Certificate Revocation is carried out automatically by the smart contract to prevent DoS, replay attacks, and so on. In particular, this Certificate Revocation happens when a user fails the *Trust Verification* phase more than n times.

6.3 Implementation issues

After seeing the description of our proposal, let's move on to some implementation details. First, in Section 7.2, we introduced the policy P_u of the user u that must be satisfied in order to proceed with the event emission from the function `verify` of the smart contract SC. In particular, with P_u, we intend a function based on the two well-known parameters of Web of Trust *(i)* `COMPLETES_NEEDED` and `MARGINALS_NEEDED` [31]. These two parameters work as thresholds, in the sense that they define the number of full trusted introducers or marginal trusted introducers needed to reach the desirable trustworthiness of the certificate. More in detail, Listing 6.4 depicts the declaration of the function `verify` which corresponds to the $Data_u$ field described in Section 7.2.

In Listing 6.1 we show the code of the function of the smart contract that is in charge to carry out the operation of give trust to another peer of the network. In particular, this function uses the modifier `onlyNotRevoked` that is shown in Listing 6.2 to ensure that only users that have not been revoked yet can actually call this function. In Listing 6.3 we show the code of the function `removeTrust()`, which is called when a user wants to remove the trust he previously gave to a certain node.

```
 function giveTrust(address to) public onlyNotRevoked{
       require(msg.sender!=to && revocationCounter[to]<maxRevocation);

       if(mapToStruct[msg.sender][to].validity==false){
              uint x = map [msg.sender].push(to);
              mapToStruct[msg.sender][to].position=--x;
              mapToStruct[msg.sender][to].validity=true;
       }
       else{
              revert();
       }
}
```

Listing 6.1: Function `giveTrust()`

```
modifier onlyNotRevoked(){
       require (revocationCounter[msg.sender]<maxRevocation);
       _;
}
```

Listing 6.2: Modifier `onlyNotRevoked`

```
function removeTrust(address to) public onlyNotRevoked{
     require(msg.sender!=to && revocationCounter[to]<maxRevocation);
     if(mapToStruct[msg.sender][to].validity==true){
            delete map [msg.sender][mapToStruct[msg.sender][to].position];
            delete mapToStruct[msg.sender][to];
     }
     else{
            revert();
     }
```

```
        }
```

Listing 6.3: Function `removeTrust()`

```
pragma solidity 0.5.7;

contract SC {
    ...
    function verify(address _to, address[] _introducers, bytes32 ciphered_random,
     uint256 random, uint256 completes_needed, uint256 marginals_needed){
    ...
    }
}
```

Listing 6.4: Declaration of the function `verify()`

Furthermore, it is important to give more details about how we effectively propagate trust. As said before, we want to remark that trust should be considered more objective than it is, because if I give trust to another person, then I am giving trust to her/his decisions too.

Anyway, it is also realistic to think that, during propagation, the trust value should decrease after a certain number of hops. Moreover, even if we implement a smart way to store and manage the trust graph in the smart contract, since every operation carried out by it is onerous, we apply the theory of the *small world* and *six degrees of separations* [261, 171] and of the so-called *horizon of observability*, which consists of a value, deriving from network theory which is, in turn, related to the FOAF (Friends of a Friend) concept, that oscillates between two and three [155] in the following way:

- if the hop counter needed to reach my introducers is less than the horizon of observability (that is equal to 3), then the trust level propagates without any decreasing;
- instead, if the hop counter needed to reach my introducer is greater than 3, the trust value decrease (from *full* to *marginal* and from *marginal* to *don't know*);
- in particular, when the hop counter reaches a value equal to six (like the degrees of separations), the algorithm stops in order to avoid to spend too much gas.

After explaining how we propagate trust, let's move on the emission of the event. As shown in Listing 6.5, we called the event `trust_satisfied` and it has all those parameters necessary to communicate that the phase verification of the trust has been successful. Observe that, the keyword `indexed` allows filtering queries on all events logged by the smart contract with respect to those parameters preceded by this keyword.

```
        pragma solidity 0.5.7;

        contract SC {
```

```
        ...
        event trust_satisfied(address indexed _from, address indexed _to, bytes32 ciphered_random,
        uint256 indexed random, string t_value);
        ...
    }
```

Listing 6.5: Declaration of the function `trust_satisfied()`

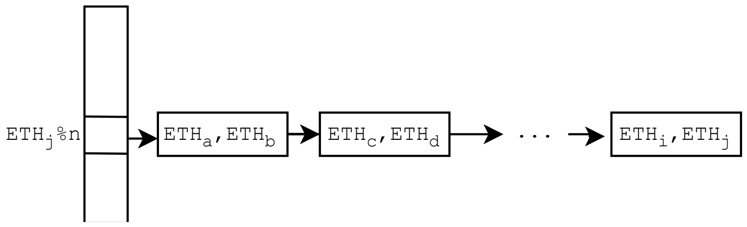


Fig. 6.3: Store schema

To store the information about the trust graph, the smart contract internally has a high storage capacity but it has to organize data in a static array. Figure 6.3 depicts the data structure used by the smart contract for the trust graph. A static array with a length n is used (left of the figure). We associate a list of blocks, each one containing a couple of Ethereum addresses, to each element of this array. When the smart contract has to store the information about a trust from the Ethereum address ETH_i to ETH_j, the smart contract enters the list of blocks addressed by the mod (modulo) operation on the address ETH_j and appends the new block made of the couple $< ETH_i, ETH_j >$ to the list. Intuitively, a high value for the dimension n reduces the number of collisions among Ethereum addresses, increasing the needed data space. This data structure represents also an efficient system in which to search for the addresses signing an Ethereum address ETH_i. The smart contract has to scan the list associated with the $i \bmod n$ location and search for ETH_i in the second element of the blocks.

The same data structure is also used inside the smart contract to store both the revocation list L_r and the number of failing attempts of Ethereum addresses. For both these cases, the key to access the array is represented by the Ethereum address to store $\bmod n$, while the blocks are structured as a couple of Ethereum address and a boolean flag for the revocation list, Ethereum address and a counter for the number of failing attempts.

Since every operation carried out on the Ethereum blockchain has a related cost, we summarise in Table 6.1 the costs associated with our implementation. In particular, we focus on the most common and used functions of the smart contract and also the entire (and unique) deployment of the smart contract, reporting both the values in Ether and in US dollars.

Function	*Ether*	*US Dollars*
`GiveTrust()`	0.000176	0,20
`Verify()`	0,023	0,005
`RemoveTrust()`	0,0011	1,25
`Whole Smart Contract`	0.001728	2,01

Table 6.1: Costs of the deployment of the Smart Contract and its functions

6.4 Future Works

We proposed a solution to enable trust propagation in the Web of Trust scenario. As further investigations, it would be interesting to exploit the three different levels of trust (complete, marginal, undefined) to improve the verification of the trustworthiness of another peer of the network. In detail, the user could distinguish his list of introducers by their degree of trustworthiness and the function verify of the smart contract could use this information to find a trust path that satisfy the user's requirements. For instance, the user could specify that he needs n_c number of complete trusted introducers or n_m number of marginal trusted introducer to consider the other peer as trusted.

7

An approach to enabling Ethereum transactions among secure identities

One of the limitations of the current Blockchains is that recipients of transactions (originated from both users and smart contracts) must preliminarily sign up the platform. Every solution based on such technology obliges peers to register into the platform before participate actively in the system. In contrast, the nature of Blockchain would allow the implementation of services with a high degree of flexibility and interoperability, once the subjects can be securely identified someway. We propose a solution that overcome this limitation by integrating Public Digital Identity with Ethereum via Identity-Based-Encryption (IBE). An important feature of the proposal is that it does not require additional trust w.r.t. that necessary for IBE and Public Digital Identity systems.

7.1 Introduction

Blockchain-based applications widely spread in recent years because of all the properties and advantages of the blockchain technology. These new solutions involve several domains, as computer science, economics, law, medicine, etc. As a consequence, any aspect regarding those technological features that impact how applications can be designed and used is very relevant.

The proposals shown above in this thesis are all based on this technology, thus implying that every participant in the above scenarios must be registered into the platform before taking part of the whole process.

This aspect could be seen as a limitation in some contexts, since it makes Blockchain platforms little appropriate in particular to those situations in which services may involve dynamically unregistered subjects. Consider, for example, a set S of subjects who are identifiable in a certain (secure) way. For the moment, it does not matter how. Suppose now that in this set there are users who may be involved in a service based on Blockchain (for example, Ethereum), in different moments, depending on dynamic conditions. Thus, for example, `Alice`, who is already registered to the service, has to transfer crypto-money (or a certain token) to `Bob`. But

Bob, despite being in S, is not in the service yet. In other words, we would like to enable some *suspended* actions, to avoid to compromise the liveness of the system. Indeed, according to the features currently supported by Ethereum (and the other Blockchains), `Alice`'s action should be denied by the system until Bob signs up the system. We obtain a similar use case when the sender is a contract instead of a human user.

Our proposal, contextualized in the Ethereum environment, is aimed at overcoming the above limitation, by enabling over Ethereum transactions and contracts among (secure) digital identities, whose existence is independent of the specific application platform. This allows the design of flexible, dynamic and interoperable services, with considerable benefits in many cases, especially in crowd-based or multi-organization domains.

To do this, we faced a number of problems. The first one is which notion of digital identity we may adopt to have a realistic result. One could think of an identity built as a combination of (verified) social network profiles owned by the subject being identified. This could be an option, but we think that whether a Public Digital Identity System exists, like those that are compliant in EU with the eIDAS regulation [358], this is the best way to follow. Thus, in our paper we refer to SPID [16], which is the Italian System of Public Digital Identity introduced in accordance with the eIDAS initiative.

The second point is how to link in a secure way digital identities and Ethereum addresses. Our solution leverages Identity-Based-Encryption (IBE) [12], which gives a direct role to the notion of identity and then a direct link between Ethereum keys and identity of the user, once she/he is able to provide the PKG (i.e., the party issuing the IBE private key) with the proof of her/his SPID identity. From this point of view, this paper is an evolution of the work presented in [95], in which the idea of integrating IBE and Blockchain is presented for the first time in the simpler context of Bitcoin Blockchain, thus without the possibility of involving unregistered users. We highlight that the role of IBE is crucial in our proposal, because a direct integration of SPID with Blockchain (like in [50]) would require that a Blockchain-side entity (an application or a smart contract) should play as a Service Provider of the public digital identity system. This would require the full trust in this entity, concerning the assessment of identity.

7.1.1 Digital identity and IBE

A digital identity is defined as information on an entity used by computer systems to represent an external agent that may be a person, organization, application, or device [11]. Another similar definition given by ISO/IEC 24760-1 reports digital identity as a set of attributes related to an entity [5].

In this study, we refer to a specific notion of digital identity, the *public digital identity*, which is recognized by law at international level making the basis for non-repudiable accountable applications. A concrete instantiation of this notion in the European Union is based on the Regulation (EU) N. 910/2014 [358] on electronic identification and trust services for electronic transactions in the internal market (eIDAS Regulation), issued on 23 July 2014 and fully effective from 1 July 2016. It has the purpose of providing a normative basis at EU level for fiduciary services and providing the means of Member States' electronic identification: the eIDAS regulation aims to provide a common regulatory basis for secure electronic interactions between citizens, businesses and public administrations and to increase the security and effectiveness of e-business services and e-business and e-commerce transactions in the European Union. Thanks to the principle of mutual recognition and reciprocal acceptance of interoperable electronic identification schemes, eIDAS wants to simplify the use of electronic authentication against public administrations, both by companies and by citizens. Each Member State maintains it own electronic identification systems, which have to be accepted by all other member states. For example, Italy has notified the EU Commission the institution of *SPID*, the Italian public system for the management of the digital identity of citizens and businesses [16]. Thanks to the eIDAS regulation, it is possible for Italian citizens to access the online services of other EU countries (university services, banking, public administration services, other online services) using SPID credentials, and at the same time, European citizens in possession of recognized national digital identities within the eIDAS framework will have access to the services of Italian public administrations.

The second concept we present is Identity-based Encryption (IBE) [12]. It is known that in asymmetric cryptography, each user owns a public and a private key, and public keys are typically arranged by a Public Key Infrastructure, which binds public keys with the respective identities of entities through a process of registration and issuance of certificates by a certificate authority (CA). Identity-based Encryption is a solution in those cases in which pre-distribution of keys is inconvenient or infeasible due to technical restraints.

Identity-based Encryption allows any party to generate a public key from a known identity value (for example, an e-mail address). A trusted third party, called the Private Key Generator (PKG), generates the corresponding private key. To operate, the PKG first publishes a master public key and retains the corresponding master private key (referred to as master key). Given the master public key, any party can compute a public key corresponding to an identity by suitably combining

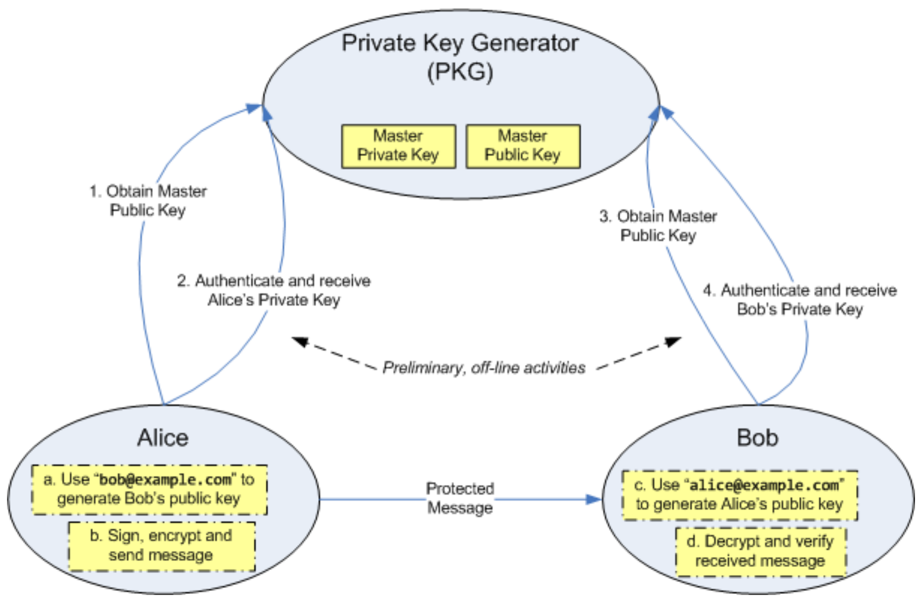

Private Key Generator (PKG)
Master Private Key
Master Public Key
1. Obtain Master Public Key
2. Authenticate and receive Alice's Private Key
3. Obtain Master Public Key
4. Authenticate and receive Bob's Private Key
Preliminary, off-line activities
Alice
a. Use "bob@example.com" to generate Bob's public key
b. Sign, encrypt and send message
Protected Message
Bob
c. Use "alice@example.com" to generate Alice's public key
d. Decrypt and verify received message

Fig. 7.1: Operations carried out in an IBE scheme.

the master public key with the identity value. To obtain a corresponding private key, the party authorized to use the identity ID contacts the PKG, which uses the master private key to generate the private key for the identity ID. The operations carried out in an IBE scheme are summarized in Figure 7.1.

As a result, parties may encrypt messages (or verify signatures) with no prior distribution of keys between individual participants, once their identity is known and well-defined. However, to decrypt or sign messages, the authorized user must obtain the appropriate private key from the PKG, by proving the possession of the proper identity. The most used IBE systems have been proposed by Boneh-Franklin [83] and by Sakai-Kasahara [317].

7.2 Our proposal

The goal of this study is to allow the association of a digital identity with a blockchain transaction. Among the possible mechanisms to handle digital identity, such as OAuth [13], OpenID [14] Windows CardSpace [18], we refer to the notion of public digital identity, which has been defined by the Regulation (EU) N. 910/2014 [358]. Our choice is motivated by the fact that we expected that, in the next years, public digital identity will involve the most of EU people: for example, on February 2017, Germany notified its national identity which has more than 40 million registered citizens [17].

In our solution, we have the following types of entity:

- an *user*, a physical or legal person using a digital identity for authentication. Each user can be associated with one or more public digital identities.
- a *public identity digital system* with identity provider IP, which creates and manages public digital identities. Without loss of generality, we assume it is unique.
- an *IBE system* with Private Key Generator PKG. It is managed by a public or private organization and provides the mapping between a digital identity and a pair of asymmetric encryption keys (called IBE keys).
- a *Distributed Ledger* allowing smart contracts (i.e., Blockchain 2.0).

In this scenario, we identify the following types of operation that users carry out.

1. *Digital Identity Registration*. To obtain a digital identity, a user must be registered to the public identity digital system. In this phase, the real identity of the user is verified before issuing the public digital identity and the security credentials.
   A public digital identity is identified by the pair $\langle username, IP \rangle$, where *IP* is the identifier of the identity provider that issued the public digital identity and *username* is a string. For example, the user X registered by the Identity Provider Y is identified by the X@Y. Moreover, any Public Digital Identity System compliant with eIDAS defines also a string *UID* (Universal ID), which is a single numeric identifier independent of the identity provider, in case of multiple identity providers.
2. *IBE private key gathering*. To obtain the IBE private key, a user contacts the Private Key Generator (PKG) of the IBE service to receive the master public key, if it is not already known. Then, the Private Key Generator, by acting as a service provider of the public digital identity system, authenticates the user by an eIDAS-compliant scheme, as illustrated in Figure 7.2.
   First, the user using a browser (`User Agent`) sends to PKG a request for gathering the IBE private key (Step 1). Then, PKG replies with an authentication request to be forwarded to *Identity Provider* (Step 2). If the received request is valid, *Identity Provider* performs a challenge-response authentication with the user (Steps 3 and 4). In case of successful user authentication, *Identity Provider* prepares the statement of user authentication, which is forwarded to PKG (Step 6). Finally, PKG provides the user with the IBE private key (Step 7).
   Observe that the IBE public key is always calculable from the digital identity of a user, provided that the IBE master public key is known.
3. *Blockchain Binding*. By this operation, a user associates his IBE public key IBE_p^K with his Blockchain address A. First, the user generates a pair of private and public blockchain keys, and, then, the blockchain address A of the user is computed as the cryptographic hash of the public key. Then, the user generates a transac-

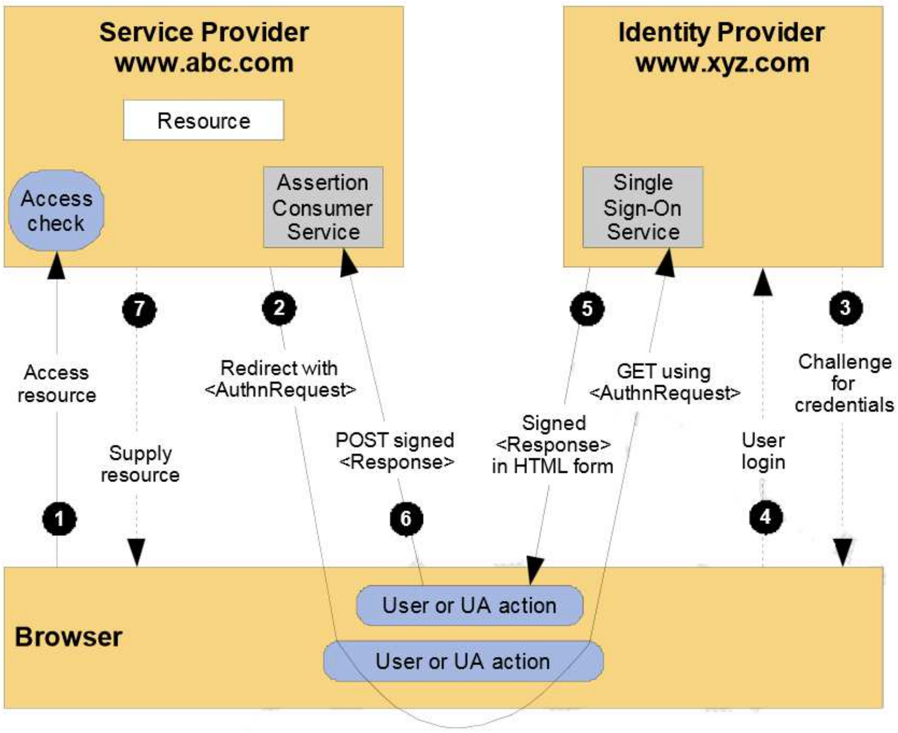


Fig. 7.2: Data flow in an authentication process.

tion from A to A on the blockchain, having in *data* field $\langle UID, E(A)\rangle$, where UID is the universal ID of the public digital identity of the user, and $E(A)$ is the encryption of the user's blockchain address by the user's IBE secret key. This transaction is called *binding transaction*. By this transaction, the user links her/his public digital identity to the blockchain address A: indeed, by computing $E(A)$, the user proves the knowledge of the IBE secret key associated with this UID.

4. *Transaction*. Suppose a user S (sender) wants to send to a user R (receiver) a transaction and let v be the value of the transaction (i.e., the amount of virtual money to transfer). In this case, the following operations are done. First, S obtains the universal ID of R, say UID_r and searches for the most recent binding transaction having UID_r in the payload: this search can be successful or not. If a transaction $T = \langle UID_r, E(A_r)\rangle$ of this type is found, then:
   a) S deciphers $E(A_r)$ by using the IBE public key calculated from UID_r, to verify that the authenticity of the signature (observe that $E(A_r)$ works as a signature to prove that the right user has generated the binding transaction). If this check fails, T is ignored and another search is carried out.
   b) After deciphering $E(A_r)$, the blockchain address of UID_r is obtained.
   c) S generates a blockchain transaction from his blockchain address A_s to A_r, with value v.

Consider now the case in which no transaction of this type is found, which is the most interesting case. This means that the user R exists but has not yet joined the blockchain. In this case, S generates a blockchain transaction from his blockchain address A_s to the blockchain address of a specific smart contract, say A_{sc}, specifying both UID_r and v. This smart contract stores the information that there is a *sleeping* transaction to UID_r, from the sender A_s and value v.

5. *Cashing*. Suppose that a user R, after registering to the blockchain, wants to receive the *sleeping* transactions sent to him before his registration (i.e., those transactions sent to the smart contract *sm* and intended for him). Then, he generates a blockchain transaction, named *cashing transaction*, from his blockchain address A_r to the blockchain address A_{sc} (i.e., the same smart contract referred above), having his UID (i.e., UID$_r$) in the payload. Now, the smart contract searches for the most recent binding transaction T sent from A_{sc} and computes the IBE public key IBE_r calculated from UID_r. Then, it extracts $E(A)$ from the payload of T and deciphers $E(A)$, verifying that UID_r is obtained. Finally, it extract from the stored sleeping transactions those sent to UID_r (if any): for each transaction found, a new transaction to A_r is generated, with the same value as the found transaction.

7.3 Implementation

In this paragraph, we instantiate the general approach presented in the previous one to the specific environment of Ethereum: in particular, we show all the operations carried out by two Ethereum users, say *Alice* and *Bob*.

1. *Digital Identity Registration*. Both Alice and Bob have a public digital identity: thus, they have been identified by an identity provider, say `example.com`, which gave each of them a public digital identity and a credential for authentication (typically, a password). Now, assume that the username of Alice is `alice` and the username of Bob is `bob`. Thus, the UIDs of Alice and Bob are `alice@example.com` and `bob@example.com`, respectively. Observe that, for the sake of presentation, we used the same identity provider (i.e., `example.com`) for both the users: however, no problem arises in case the public digital identities are issued by different identity providers, because the solution does not depend on the particular UID of the user.
2. *IBE private key gathering*. To obtain the IBE private key, a user connects to the site of the IBE system by the browser (i.e., the user agent) and sends a request for accessing the service. Observe that the IBE system acts as a service provider in this step, because it needs to authenticate the user before issuing the private key.

Then, the IBE system replies to the user agent with an authentication request to be forwarded to the identity provider. The identity provider is selected according to the user's UID.

If the received request is valid, the identity provider performs a challenge-response authentication with the user. In case of successful user authentication, the identity provider prepares the *assertion* containing the statement of the user authentication for the IBE service provider. The assertion contains the reference to the request message, the authenticated user, the identity provider, the personal information about the authenticated user, the temporal range of validity, and the description of the authentication's context. The assertion is signed by the identity provider to guarantee integrity and authenticity.

Now, the assertion returned to the user agent is forwarded via `http POST` Binding to the IBE service provider. The IBE system verifies the assertion and provides the user with her/his IBE private key. We denote by IBE_U^S the IBE private key of the user U.

Concerning the user's IBE public key, they are computed starting from the master public key and the user's UID. We denote by IBE_U^P the IBE public key of the user U.

3. *Blockchain Binding*. Each user needs to have a private and a public blockchain key. The private key is a randomly generated 256-bit string. The public key is generated by the private one by means of a cryptographic function named *elliptic curve point multiplication*. In particular, the used algorithm is Curve Digital Signature Algorithm (ECDSA) and the elliptic curve is `secp256k1` [254].

   The Ethereum address A of a user is computed from the public key K by apply Keccak-256 [15], and finally taking the last 20 bytes of that hash. We denoted by A_U the blockchain address of the user U. Finally, each user generates the binding transaction having as payload $\langle UID, E(A) \rangle$, where UID is the universal ID of the public digital identity of the user, and $E(A)$ is the encryption of the user's Ethereum address done by the user's IBE private key.

4. *Transaction*. Now, both Alice and Bob have their public digital identity associated with a blockchain address. Suppose that Alice has to send some Ether money to Bob, but Bob has not an Ethereum wallet (i.e., he has not an Ethereum address). Clearly, we can image that users run an application (on a PC or a smartphones) to manage transaction generations.

   First, Alice has to know the UID of Bob: the UID of Bob as well as the amount of money to transfer are inserted into the application, which generates a transaction to the smart contract. In Listing 7.1, we give an implementation of this contract written in Solidity, which is a JavaScript-like language. For the sake of presentation, we do not explain every line of the code: we assume the reader is familiar with Solidity and Oraclize, which is the leading oracle service for smart contracts and blockchain [6] In particular, it is called the function pay, using the UID of Bob as parameter (Lines 13-15): this function stores the amount that will be given to Bob when he will register (by payUID).

5. *Cashing*. After Bob creates his wallet, he can ask for receiving the amount from the *sleeping* transactions sent to him before his registration. To do this, he generates a cashing transaction to the smart contract illustrated in Listing 7.1, by calling the function cash. The smart contract first checks if there is some amount for Bob. If any, an oraclize function is used (Line 21), which returns the Ethereum address of Bob by the callback function. Finally, a money transfer to Bob is carried out by the smart contract and the amount to pay to bob is reset (Lines 33-34).

By this protocol, we enable on Ethereum the possibility to send money to users without the need to know their blockchain address. The suitable use of the secure digital identity guarantees that only the correct user receives money.

```
pragma solidity ^0.4.25;

import "github.com/oraclize/ethereum-api/oraclizeAPI_0.4.25.sol";
import "github.com/Arachnid/solidity-stringutils/strings.sol";

contract SleepingEther is usingOraclize {
   mapping(bytes32=>string) uidMapping; //mapping between queryID and bool
   mapping(string=>uint) payUid; //mapping between UID and eth value to send
   address public addr;
   using strings for *;
   string pi;

   function pay(string uid) public payable {
        payUid[uid] += msg.value; // add the ether addressed to uid
    }

    function cash (string uid) public payable{
        if(payUid[uid]>0)
            if (oraclize.getPrice("URL") <= address(this).balance) {
                pi = "URL".toSlice().concat(uid.toSlice());
                bytes32 queryId = oraclize_query("URL", pi);
                uidMapping[queryId]=uid;
           }
    }

   function __callback (bytes32 myid, string result, string uid) public {
        if (msg.sender != oraclize_cbAddress())
          revert ();
        bytes memory tempEmptyStringTest = bytes(result);
        if(tempEmptyStringTest.length != 0){
            addr = parseAddr(result);
            uint tot= payUid[uidMapping[myid]];
            addr.transfer(tot);
            payUid[uid]=0;
        }
    }
}
```

Listing 7.1: The code of the smart contract.

7.4 Related work

In this section, we survey the approaches present in the literature related to our topic.

Thanks to the assurance of authenticity and uniqueness of transactions, blockchain starts to become the technical core of cryptocurrency, access control systems, asset management, banking, e-voting, etc. [247, 297, 81, 123].

The authors of [296] provide an overview of the blockchain technology and its potential to disrupt the world of banking through facilitating global money remittance, smart contracts, automated banking ledgers and digital assets. In this regard, they provide a brief overview of the core aspects of this technology, as well as the second-generation contract-based developments. From there, their work enforces key issues that must be considered in developing such ledger based technologies in a banking context.

In [33], the authors review applications relying on blockchain by highlighting the potential benefit of such technology in the manufacturing supply chain. Furthermore, they propose a vision for the future blockchain ready manufacturing supply chain.

The paper [65] provides a high-level understanding of how blockchain technology will be a fundamental tool to improve supply chain operations. It illustrates theoretical and conceptual models for use of open blockchain in different supply chain applications with real-life practical use cases as is being developed and deployed in various industries and business functions.

In [216], the authors observe that digital supply chain integration is becoming increasingly dynamic and investigate the requirements and functionalities of supply chain integration, concluding that cloud integration can be expected to offer a cost-effective business model for interoperable digital supply chains. Furthermore, they explain how supply chain integration through the blockchain technology can achieve important transformation in digital supply chains and networks.

The authors of [371] propose an overview of what smart contracts are and what are their main challenges for the future. In particular, they state that smart contracts have three main characteristics: *(i)* autonomy, *(ii)* self-sufficiency and *(iii)* decentralization. Autonomy means that the contracts and the initiating agents do not need to be in further contact. Self-sufficient means that smart contracts are able to raise funds by providing services and spending them when needed. Furthermore, smart contracts are decentralized as they do not are valid on a single centralized server, but they are distributed and self-executed across network nodes. As they can be seen as a distributed application, they have to face almost the well-known challenges of them, such as the reentrancy vulnerability, the privacy issues, how to guarantee the reliability of external information, and so on.

Another topic related to our proposal regards digital identity, in which we can find a rich literature. Bitnation [10] is the world's first Decentralized Borderless Voluntary Nation (DBVN). Bitnation started in July 2014 and hosted the first blockchain for refugee emergency ID, marriage, birth certificate, World Citizenship and more. The website proof-of-concept, including the blockchain ID and Public Notary, is used by tens of thousands of Bitnation Citizens and Embassies around the world.

In [90], the authors focus on the Public Digital Identity System (SPID), the Italian government framework compliant with the eIDAS regulatory environment. They observe that a drawback limiting the real diffusion of this framework is that, despite the fact that identity and service providers might be competitor private companies, SPID authentication results in the information leakage about the customers of identity providers. To overcome this potential limitation, they propose a modification of SPID to allow user authentication by preserving the anonymity of the identity provider that grants the authentication credentials. This way, information leakage about the customers of identity providers is fully prevented.

In [195], the authors highlight that, since we are in the Internet era, it is necessary a right blockchain-based identity management, as we have faced identity management challenges since the sunrise of the Internet. In particular, they observe that blockchain technology should offer a way to circumvent this problem by delivering a secure solution without the need for a trusted, central authority. It can be used for creating an identity on the blockchain, making it easier to manage for individuals, giving them greater control over who has their personal information and how they access it. The proposed solution stores the encrypted identity of users, allowing them to share their data with companies and manage it on their own terms.

The paper [356] focuses on pseudonymization, a concept that was only recently formally introduced in the EU regulatory landscape. In particular, it attempts to derive the effects of the introduction of pseudonyms (or pseudonymous credentials) as part of the eIDAS Regulation on electronic identification and trust services and, ultimately, to compare them with the effects of pseudonymization within the meaning of the General Data Protection Regulation (the GDPR). The work examines how eIDAS conceives pseudonymization and explains how this interpretation would translate in practical uses in the context of a pan-European interoperability framework.

In [89], an advanced electronic signature protocol that relies on a public system for the management of the digital identity is proposed by the authors. The work aims at implementing an effective synergy to provide the citizen with a unique, uniform, portable, and effective tool applicable to both authentication and document signature.

In [36], the authors proposed SCPKI, an alternative PKI system based on a decentralized and transparent design using a web-of-trust model and a smart contract on the Ethereum blockchain, to make it easily possible for rogue certificates to be detected when they are published. The web-of-trust model is designed such that an entity or authority in the system can verify fine-grained attributes of another entity's identity, as an alternative to the centralized certificate authority identity verification model.

The paper [115] argues that existing laws, specifically the federal Electronic Signatures in Global and National Commerce Act ("ESIGN") and state laws modeled on the Uniform Electronic Transaction Act ("UETA"), render blockchain-based smart contracts enforceable and therefore immediately usable.

Part II

Exploiting generated data

Another key aspect of smart cities regards data in all its processes. Data generation, collection and management are topics addressed in this second part of the thesis. Indeed, smart cities are characterized by the relevance they give into data, representing their main core concept, around which the whole idea and architecture is developed. Furthermore, in the world we are living, there is an exponential production of such data because of sensors, smart objects, online social networks, blogs and so on. This massively generation led into what we call *Big Data*. Differently from the past, data are not now only structured into rigid and strictly ruled containers, but semi-structured and unstructured data are emerging rapidly. In particular, it is esteemed that, currently, more than 80% of the information available in the Internet is unstructured [117]. This phenomenon is changing the directions of the researchers and scientists since there is need of new solutions able to collect, store and manage all these (and different) data that goes beyond the *old* paradigm of the data warehouse, which does not perfectly fit with these new requirements.

In fact, if data are well collected and managed it could be easier for data scientists and analysts to extract from them information and knowledge in such a way to support better the decision making process. This could help both companies and users, in terms of business and of quality of services.

This part of the thesis could be divided into two main areas. In the first one, a new paradigm that models data sources (smart objects) exploiting of the network analysis and the graph theory is presented. In particular, in Chapter 8 we propose a new model that is able to manage different instances of the same object with respect to its usage, so creating a new way of representing communities of smart objects. Moreover, we propose a new crawler that fits with the new typology of network.

In the second area, we focus on data collection and management, by exploiting and investigating the new concept of *Data Lakes*.

Data Lakes is a term introduced in 2010 by James Dixon (CTO at Pentaho, an American Business Intelligence company) [4]. He coined this new term with the following definition:

If you think of a datamart as a store of bottled water - cleansed and packaged and structured for easy consumption - the data lake is a large body of water in a more natural state. The contents of the data lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples.

Soon big companies and researchers have been fascinated by this innovative idea. The interest through data lakes led to a rapid develop and implementation of such proposal and Microsoft Azure propose a new, and more complete definition, of data lakes:

A data lake is a storage repository that holds a large amount of data in its native, raw format. Data lake stores are optimized for scaling to terabytes and petabytes of data. The data typically comes from multiple heterogeneous sources, and may be structured, semi-structured, or unstructured. The idea with a data lake is to store everything in its original, untransformed state. This approach differs from a traditional data warehouse, which transforms and processes the data at the time of ingestion.

Briefly we can define data lake paradigm as the successors of data warehousing since they accept and store every data in its raw format. This is clearly a good way to collect different typologies of data (structured, semi-structured and unstructured) but, on the other hand, this requires a correct management of the whole data lake in order to not let it becoming a *Data Swamp*. This can be reached by using wisely metadata, that are information associated with the data. For this reason, in Chapter 9 we propose a new approach to uniformly represent different and heterogeneous metadata, in such a way it is possible to model every different typology of data in the same way. In particular, the main challenge addressed regards defining a new approach able to give a *structure* to unstructured sources.

Now that we have all metadata represented in the same way, we define and propose in Chapter 10 a new lightweight model to integrate different sources and extracting semantic relationships from them by means inter-schema properties.

In Chapter 11 we present an approach that exploits the above models to navigate the data lake by extracting thematic views in a topic-driven way.

Finally, in Chapter 12 we propose a new model to extract complex knowledge patterns among concepts belonging to different data sources.

8

A new model to represent smart objects in multiple environments

In smart cities, smart objects are main actors in in generating, producing and sharing information. We propose a new paradigm, called MIoT (Multiple Internets of Things), capable of modelling and handling the increasing complexity in this context. Researches and scientists highlight that it is possible to attribute social behaviours to smart objects in the Internet of Things, by means of different kinds of relationships that can be born. In this sense, we started from the very interesting and appreciated concept of Social Internet of Things on which we add the feature of separation of the physical role of the object with respect to its semantic one. In this proposal, we model a new paradigm where every smart object can participate into different networks via its instances that are individuated from different uses of the smart object itself.

8.1 Introduction

The Internet of Things can be considered as an evolution of the Internet, based on the pervasive computing concept [55]. In the past, several strategies to implement the IoT paradigm and to guarantee ubiquitous computing have been proposed [174, 389, 133]. One of the most effective of them is based on the use of the social networking paradigm [53, 56, 54]. In this case, IoT is represented as a social network and, thanks to this association, Social Network Analysis-based models can be used to empower IoT. One of the most advanced attempts in this direction is SIoT (Social Internet of Things). In SIoT, things are empowered with social skills, making them more similar to people [53, 56]. In particular, they can be linked by five kinds of relationship, namely: *(i)* parental object relationship; *(ii)* co-location object relationship; *(iii)* co-work object relationship; *(iv)* ownership object relationship; *(v)* social object relationship. If: *(i)* a node is associated with each thing, *(ii)* an edge is associated with each relationship between things, and, finally, *(iii)* all the nodes and the edges linked by the same relationship are seen as joined together, SIoT can be modeled as a set of five pre-defined networks. Here, some nodes belong to only one network (we call them inner-nodes), whereas other ones belong to more networks (we call them cross-nodes).

The idea underlying SIoT is extremely interesting and, as a matter of fact, has received, and is still receiving, a lot of attention in the literature. However, we think that, in the next future, the number of relationships that might connect things could be much higher than five, and relationships could be much more variegate than the ones currently considered by SIoT. As a consequence, we think that a new paradigm, taking into account this fact, is in order.

In [87, 276], authors introduced the concept of Social Internetworking System (SIS, for short) as a system comprising an undefined number of users, social networks and resources. The SIS paradigm was thought to extend the Single Social Network paradigm by taking into account that: *(i)* a user can join many social networks, *(ii)* these joins can often vary over time, and *(iii)* the presence of users joining more social networks can favor the cooperation of users, who do not join the same social networks. We think that the key concepts of SIS can also be applied to things (instead of to users) and to relationships between things and we propose the MIoT (Multiple Internets of Things) paradigm. The core of the SIS paradigm is modelling users and their relationships as a unique big network and, at the same time, as a set of related social networks connected to each other thanks to those users joining more than one social network. In this study, we propose to extend the ideas underlying the concept of SIS to IoT. The MIoT paradigm arises as a result of this objective.

Roughly speaking, an MIoT can be seen as a set of things connected to each other by relationships of any kind and, at the same time, as a set of related IoTs, one for each kind of relationship. Actually, a more precise definition of MIoT would require the introduction of the concept of instance of a thing in an IoT. According to this concept, the instance of a thing in an IoT represents a virtual view of that thing in the IoT. Having this in mind, an MIoT can be seen as a set of related IoTs, one for each kind of relationship into consideration. The nodes of each IoT represent the instances of the things participating to it. As a consequence, a thing can have several instances, one for each IoT to which it participates. As will be clear in the following, the existence of more instances for one thing plays a key role in the MIoT paradigm because it allows the definition of the cross relationships among the different IoTs of the MIoT.

Differently from SIoT, in the MIoT paradigm, the number of relationships is not defined a priori. In an MIoT, there is a node for each thing; furthermore, there is an edge between two nodes if the corresponding things are linked by a relationship. If more kinds of relationship exist between two things, then more edges exist between the corresponding nodes, one for each kind of relationship. All the nodes linked by a given kind of relationship, together with the corresponding edges, form an IoT of the MIoT.

Observe that, under this MIoT definition, SIoT can be seen as a specific case of MIoT in which the number of the possible kinds of relationship is limited to 5 and these kinds are pre-defined. IoTs are interconnected thanks to those nodes corresponding to things involved in more than one kind of relationship. We call *cross nodes* (*c-nodes*, for short) these nodes and *inner nodes* (*i-nodes*, for short) all the other ones. Then, a c-node connects at least two IoTs of the MIoT and plays a key role to favor the cooperation among i-nodes belonging to different IoTs. As a consequence, differently from SIoT, the nodes of an MIoT are not all equal: c-nodes will presumably play a more important role than i-nodes for supporting the activities in an MIoT.

Note that the MIoT paradigm can be seen as an attempt to address an open issue evidenced in [54] about some improvements that should be made on the SIoT paradigm. Among these improvements, other two very relevant ones evidenced in this proposal are the following:

- defining inter-objects relationships; this issue requires a correct representation of a smart object and the definition of both methods and tools to crawl and discover other (possibly heterogeneous) objects with which interactions can be established;
- modelling the new social networks thus obtained, characterizing them and defining new algorithms to perform their analysis.

The MIoT paradigm already mentioned, and the crawling strategy, which we present below, taken together, can represent an answer to these exigencies of improvement.

From a more applicative point of view, having some IoTs that can “communicate” through c-nodes can lead to some beneficial synergies. For instance, assume that an environment-related IoT can communicate with a home-related IoT through a cross node. Assume that the former IoT evidences an abnormal presence of dioxin in a place located some kilometers away from the home (for instance, owing to a fire of a plastic deposit). Assume, also, that this IoT is evidencing that the wind direction is pushing the dioxin towards the home. The home-related IoT could be “informed” through a cross node about this fact and could close all windows before the arrival of the dioxin.

Once an MIoT has been defined, it is possible to apply Social Network Analysis-based techniques on it to extract powerful knowledge concerning its things, their relationships, the IoTs formed by them, etc. However, in order to perform knowledge extraction, especially when the number of the things to investigate is huge, an important pre-requisite is having a good approach to crawl the underlying graph. Crawling is also extremely useful in a second family of applications, based on the exploration of the "neighborhood" (i.e., things and relationships) of a given thing (think, for instance, of the case in which a new thing is added to the Internet of Things and wants to create relationships with other things). There are also a lot of further possible applications of crawling, already known in the literature [279, 354], and that can be extended to the Internet of Things.

In the literature, several crawling strategies for single social networks have been proposed. Among them, the most representative ones are: *(i)* Breadth-First Search (BFS, for short) [384], which moves in breadth by exploring the neighborhood of each node; *(ii)* Random Walk (RW, for short) [242], which moves in random directions; *(iii)* Metropolis-Hastings Random Walk (MH, for short) [340, 219, 307], which moves in random directions, disfavoring high-degree nodes. These strategies were largely investigated for single networks, and their pros and cons have been highlighted in [166, 221].

However, we have seen that, in an MIoT, there exist two different kinds of node, and none of the previous strategies considers this fact, as they were developed for crawling a single network. We argue that a new strategy, capable of distinguishing c-nodes from i-nodes and of performing a right trade-off between breadth, depth and randomicity, is in order. Therefore, a second objective of this study is to address this issue. In fact, we propose a new crawling strategy, called *Cross Node Driven Search* (CDS, for short). CDS is centered on c-nodes; in fact, it allows users to privilege the visit of c-nodes over the one of i-nodes, if necessary, and to tune how much c-nodes should be privileged over i-nodes.

To prove the correctness of CDS, we tested it against the three main classic strategies mentioned above. In carrying out this task, we defined, and, then, used different metrics aimed to evaluate the quality of each crawler under consideration. The results of these experiments confirm our assumption about the inadequacy of the classic crawling strategies for an MIoT and, by contrast, the suitability of the new CDS strategy in this context.

8.2 The MIoT paradigm

We define a MIoT $\mathcal{M}$ as a set of m Internets of Things (see Figure 8.1 for a schematic representation of it). These trends suggest that, with the explosion of the number of available things, it is not realistic to talk about a unique Internet of Things. By contrast, it is more appropriate to consider several IoTs, each consisting of a (social) network of things. Formally speaking:

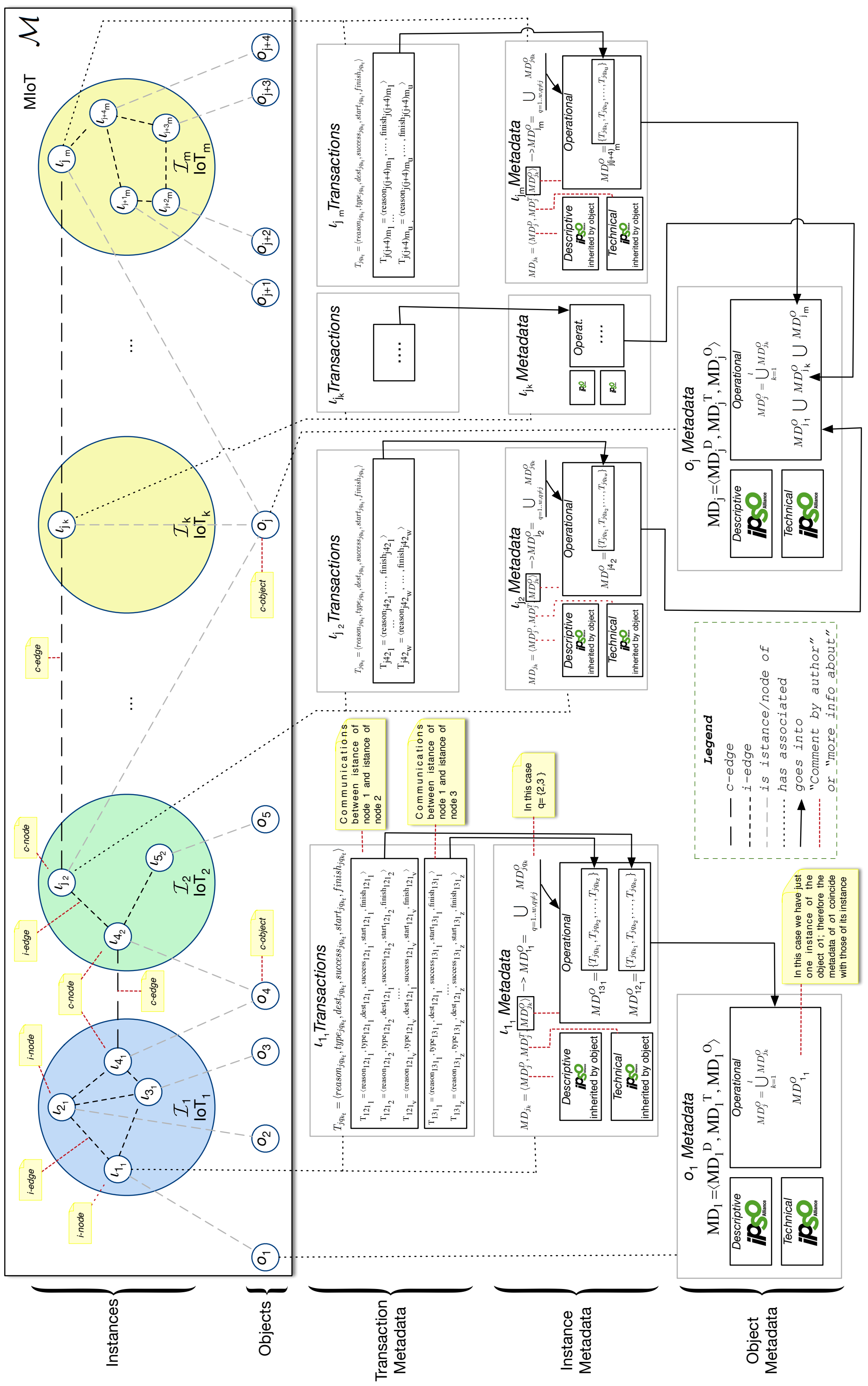


Fig. 8.1: Schematic representation of the proposed MIoT structure

$$\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \cdots, \mathcal{I}_m\}$$

where $\mathcal{I}_k$ is an IoT.

Let o_j be an object of $\mathcal{M}$. We assume that, if o_j belongs to $\mathcal{I}_k$, it has an instance ι_{j_k}, representing it in $\mathcal{I}_k$. The instance ι_{j_k} indicates a virtual view (or, better, a virtual agent) representing o_j in $\mathcal{I}_k$. For instance, it provides all the other instances of $\mathcal{I}_k$, as well as the users interacting with $\mathcal{I}_k$, with all necessary information about o_j. Interestingly, this information is represented according to the format and the conventions adopted in $\mathcal{I}_k$.

In $\mathcal{M}$, *a set MD_j of metadata* are associated with an object o_j. We define a rich set of metadata of an object, because these play a key role in favoring the interoperability of IoTs and of their objects, which is the main objective of an MIoT. As a consequence, MD_j consists of three different subsets:

$$MD_j = \langle MD_j^D, MD_j^T, MD_j^O \rangle$$

Here:

- MD_j^D represents the set of *descriptive metadata*. It denotes the type of o_j. For representing and handling descriptive metadata, a proper taxonomy, such as the one defined by the IPSO Alliance [8], can be adopted.
- MD_j^T represents the set of *technical metadata*. It must be compliant with the object type. In other words, there is a different set of metadata for each object type of the taxonomy. Also in this case, the IPSO Alliance provides a well defined set of technical metadata for each object type. It is worth pointing out that, in principle, we could have allowed much richer descriptive and technical metadata. However, we did not make this choice because we preferred to relate our definition of metadata to an international IoT standard, such as the one defined by the IPSO Alliance. Furthermore, as will be clear in the following, our approach needs mainly operational metadata. As a consequence, making descriptive and technical metadata more complex would have added a useless level of complexity to our model.
- MD_j^O represents the set of *operational metadata*. It regards the behavior of o_j. The operational metadata of an object o_j is defined as the union of the sets of the operational metadata of its instances. Specifically, let $\iota_{j_1}, \iota_{j_2}, \ldots, \iota_{j_l}$, $l \leq m$, be the instances of o_j belonging to the IoTs of $\mathcal{M}$. Then:

$$MD_j^O = \bigcup_{k=1}^{l} MD_{j_k}^O$$

$MD_{j_k}^O$ is the set of the operational metadata of the instance ι_{j_k}. In order to understand the structure of $MD_{j_k}^O$, we first have to analyze the structure of $MD_{jq_k}^O$, i.e. the set of operational metadata between two instances ι_{j_k} and ι_{q_k}, of the objects o_j and o_q, in the IoT $\mathcal{I}_k$.

Specifically, $MD^O_{jq_k}$ is given by the set of metadata associated with the transactions between ι_{j_k} and ι_{q_k}. In particular:

$$MD^O_{jq_k} = \{T_{jq_{k_1}}, T_{jq_{k_2}}, \ldots, T_{jq_{k_v}}\}$$

where $T_{jq_{k_t}}$, $1 \leq t \leq v$, represents the metadata of the t-th transaction between ι_{j_k} and ι_{q_k}, assuming that v is the current number of transactions between the two instances.

$T_{jq_{k_t}}$ can be represented as follows:

$$T_{jq_{k_t}} = \langle reason_{jq_{k_t}}, type_{jq_{k_t}}, inst1_{jq_{k_t}}, inst2_{jq_{k_t}}, success_{jq_{k_t}}, start_{jq_{k_t}}, finish_{jq_{k_t}} \rangle$$

where:

- $reason_{jq_{k_t}}$ denotes the reason causing the transaction, chosen among a set of default values.
- $type_{jq_{k_t}}$ indicates the transaction type (e.g., unicast, multicast, and so forth).
- $inst1_{jq_{k_t}}$ and $inst2_{jq_{k_t}}$ denote the two instances involved in $T_{jq_{k_t}}$. Observe that a transaction between ι_{j_k} and ι_{q_k} could be part of a longer path whose source and/or target nodes could be different from ι_{j_k} and ι_{q_k}. In principle, the source and/or the target nodes of a transaction could belong to an IoT different from $\mathcal{I}_k$. In this last case, it is necessary to reach $\mathcal{I}_k$ from the source, and/or to reach the target from $\mathcal{I}_k$, through one or more cross nodes, if possible.
- $success_{jq_{k_t}}$ denotes if the transaction succeeded.
- $start_{jq_{k_t}}$ is the timestamp associated with the beginning of the transaction.
- $finish_{jq_{k_t}}$ is the timestamp associated with the end of the transaction (its value is NULL if $T_{jq_{k_t}}$ failed).

In our model, the direction of a transaction is not considered. Furthermore, the parameter v, i.e., the number of transactions for each pair of instances, varies when moving from a pair of instances to another.

Observe that we have made our model powerful enough to represent and handle all the transactions between two instances of each IoT. Having all these detailed historical data at disposal could help the analysis of the real "social" behavior of each object. Furthermore, these data could be exploited in many applications; think, for instance, of the computation of the trust and reputation of each object, the investigation of objects with similar or complementary behaviors, and so forth. On the other hand, maintaining a full history of transactions may be very expensive and useless in many real life applications; in some cases, suitable data summarizations could be enough. As a consequence, when passing from the abstract model definition to real life applications, the transaction representation could be removed, extended or restricted on the basis of a tradeoff between costs and benefits for the current application.

We are now able to define the set of the operational metadata MD^O_{jk} of an instance ι_{j_k} of $\mathcal{I}_k$. Specifically, let $\iota_{1_k}, \iota_{2_k}, \ldots, \iota_{w_k}$ be all the instances belonging to $\mathcal{I}_k$. Then:

$$MD^O_{jk} = \bigcup_{q=1..w, q \neq j} MD^O_{jq_k}$$

In other words, the set of the operational metadata of an instance ι_{j_k} is given by the union of the sets of the operational metadata of the transactions between ι_{j_k} and all the other instances of $\mathcal{I}_k$.

Given an instance ι_{j_k}, relative to an object o_j and an IoT $\mathcal{I}_k$, we define the metadata MD_{j_k} of ι_{j_k} as:

$$MD_{j_k} = \langle MD^D_j, MD^T_j, MD^O_{j_k} \rangle$$

In other words, the descriptive and the technical metadata of an instance ι_{j_k} coincide with the ones of the corresponding object o_j. Instead, the operational metadata of ι_{j_k} is a subset of the operational metadata of o_j that comprise only those ones regarding the transactions, which ι_{j_k} is involved in.

It is possible to associate a graph:

$$G_k = \langle N_k, A_k \rangle$$

with $\mathcal{I}_k$. Here, N_k indicates the set of the nodes of $\mathcal{I}_k$. There is a node n_{j_k} for each instance ι_{j_k} of an object o_j in $\mathcal{I}_k$. A_k denotes the set of the edges of $\mathcal{I}_k$. There is an edge $a_{jq_k} = (n_{j_k}, n_{q_k})$ if there exists a link between the instances ι_{j_k} and ι_{q_k} of the objects o_j and o_q in the IoT $\mathcal{I}_k$.

Also the overall MIoT $\mathcal{M}$ can be represented as a graph:

$$\mathcal{M} = \langle N, A \rangle$$

Here:

- $N = \bigcup_{k=1}^m N_k$;
- $A = A_I \cup A_C$, where:
  - $A_I = \bigcup_{k=1}^m A_k$;
  - $A_C = \{(n_{j_k}, n_{j_q}) | n_{j_k} \in N_k, n_{j_q} \in N_q, k \neq q\}$; observe that n_{j_k} and n_{j_q} are the nodes corresponding to the instances ι_{j_k} and ι_{j_q} of the object o_j in $\mathcal{I}_k$ and $\mathcal{I}_q$.

In other words, an MIoT $\mathcal{M}$ can be represented as a graph whose set of nodes is the union of the sets of nodes of the corresponding IoTs. The set A of the arcs of $\mathcal{M}$

consists of two subsets, A_I and A_C. A_I is the set of the inner arcs of $\mathcal{M}$ and is the union of the sets of the arcs of the corresponding IoTs. A_C is the set of the cross arcs of $\mathcal{M}$; there is a cross arc for each pair of instances of the same object in different IoTs. We call:

- *i-edge* an edge of $\mathcal{M}$ belonging to A_I;
- *c-edge* an edge of $\mathcal{M}$ belonging to A_C;
- *c-node* a node of $\mathcal{M}$ involved in at least one c-edge;
- *i-node* a node of $\mathcal{M}$ not involved in any c-edge;
- *c-object* an object having at least one pair of instances whose corresponding nodes are linked by a c-edge; clearly, any object with at least two different instances is a c-object.

It is worth pointing out that, as mentioned in the Introduction, there is a strict correlation between the MIoT paradigm and the concept of Social Internetworking System (hereafter, SIS) already presented in the literature [87]. In particular: *(i)* the concept of c-edges shares several features with the one of "me"-edge in a SIS; *(ii)* the concept of c-node is similar to the one of bridge in a SIS; *(iii)* a c-object corresponds to a user joining more social networks.

8.2.1 An example of an MIoT

Since the MIoT paradigm is new, in the Internet there is no known case study or real example about it yet. As a consequence, to provide the reader with an example, and, at the same time, to have a testbed for our experiments, we constructed an MIoT starting from some open data about things available on the Internet. In particular, we derived our data from *Thingful* [9]. This is a search engine for the Internet of Things, which allows us to search among a huge number of existing things, distributed all over the world. Thingful also provides some suitable APIs allowing the extraction of all the data we are looking for.

In order to construct our MIoT, we decided to work with 250 things whose data was derived from Thingful. Given the huge number of things available in Thingful, it could appear that the number of things composing our testbed is excessively limited. However, we observe that:

- This was the first attempt to construct a real MIoT and, then, it was extremely important for us to have a full control of it in order to verify if we were proceeding well. A full human control with a much higher number of nodes was not possible.
- We wanted to fully analyze the behavior, the strengths and the weaknesses of our crawler and to understand, step by step, its way of operating vs the ones of

other crawlers. Again, a full human verification of these aspects was not possible with a larger testbed.

- As it will be clear in the following, our approach to obtaining the testbed is fully scalable. As a consequence, an interested researcher can apply it to construct a much larger testbed, if necessary.

We considered three dimensions of interest for our MIoT, namely:

a. *Category*: It specifies the application field which a given thing operates in. The categories we have chosen were five, namely *home*, *health*, *energy*, *transport*, and *environment*. Each category originated an IoT. Each thing was assigned to exactly one category.
b. *Coastal distance*: It specifies the coastal distance (i.e., the distance from any sea, lake or river) of each thing. The distance values we have set were:
    - *near*, for things distant less than 20 kilometres from the coast, for the categories *environment* and *energy*, and less than 5 kilometres, for the other three categories;
    - *mid*, for things whose minimum distance from the coast was between 20 and 105 kilometres, for the categories *environment* and *energy*, and between 5 and 25 kilometres, for the other three categories;
    - *far*, for things whose minimum distance from the coast was higher than 105 kilometres, for the categories *environment* and *energy*, and higher than 25 kilometres, for the other three categories.

    An IoT was created for each distance value. The different coastal distance values for *environment* and *energy*, on the one hand, and for the other three categories, on the other hand, have been determined after having analyzed the distribution of the involved categories of things against the coastal distance, in such a way as to produce a uniform distribution of each category of things in the three IoTs related to the coastal distance dimension.
c. *Altitude*: it specifies the altitude of the place where the thing is located. The altitude values we have defined were: *plain* (corresponding to an altitude less than 500 meters), *hill* (corresponding to an altitude between 500 and 1000 meters), and *mountain* (corresponding to an altitude higher than 1000 meters). An IoT was created for each altitude value.

As a consequence, our MIoT consists of 11 IoTs. We associated an object with each thing; therefore, we had 250 objects. In principle, for each object, we could have associated an instance for each dimension. However, in order to make our testbed closer to a generic MIoT, representing a real scenario, where it is not said that all the objects have exactly the same number of instances, we decided not to associate three

IoT	*Number of instances*
a.home	22
a.health	22
a.energy	22
a.transport	22
a.environment	22
b.near	14
b.mid	38
b.far	53
c.plain	44
c.hill	50
c.mountain	6

Table 8.1: Number of instances present in the IoTs of our MIoT

instances with each object. Instead, we associated only one instance (distributed uniformly at random among the three dimensions, and based on the features of the things of the IoTs of a given dimension) to 200 of the 250 objects. Analogously, we associated two instances (distributed by following the same guidelines mentioned above) to 35 of the 250 objects. Finally, we associated three instances, one for each possible dimension, to 15 of the 250 objects. At the end of this phase, we had 315 instances, distributed among the 11 IoTs of our MIoT as shown in Table 8.1.

To complete our MIoT and its network representation, we had to define a policy to create *i-edges*. In fact, it was clear that our MIoT should have had a node for each instance and a c-edge for each pair of instances referring to the same object. Therefore, the last decision regarded how to define i-edges. Given our scenario, it appeared reasonable to consider distances among things as the leading parameter for the creation of i-edges. To carry out this last task, we have preliminarily computed the distribution of the number of connected components possibly created from our instances against the maximum possible distance. Obtained results are reported in Figure 8.2. Based on this figure, in order to obtain a balanced number of connected components, we decided to connect two instances of the same IoT if the distance of the corresponding things was lesser than 1000 kilometres.

After this last choice, our MIoT was fully defined. In order to help the reader to mentally portray it, in Figure 8.3, we provide a graphical representation.

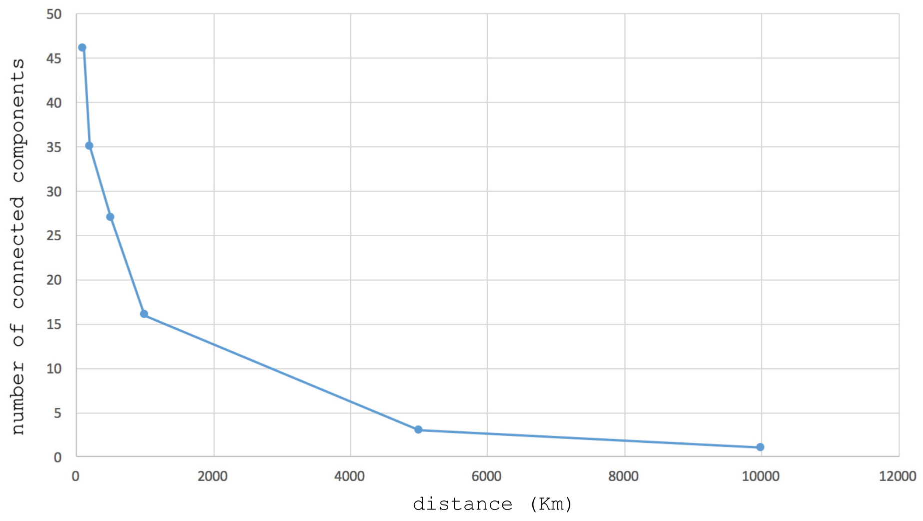


Fig. 8.2: Distribution of the number of connected components of the instances of our MIoT against distances

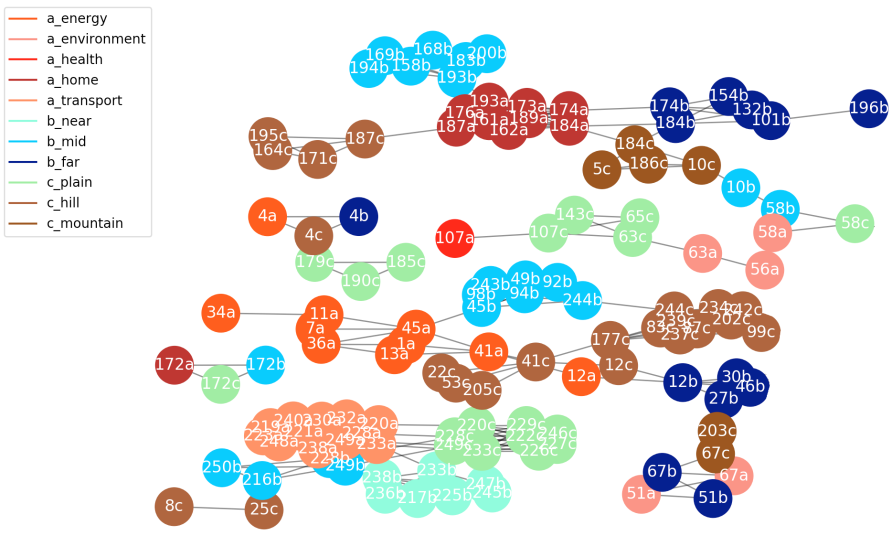


Fig. 8.3: Graphical representation of our MIoT

8.2.2 Why using the MIoT paradigm?

In the Introduction, we have specified that the MIoT paradigm goes in the direction suggested by some authors, who observe that it is no longer possible to think of a single global Internet of Things [54].

In this section, we present a case study aiming at comparing the classical vision of a unique global Internet of Things with the new MIoT-based vision of multiple networks connected to each other through cross nodes and cross edges.

First, we must clarify that a slavish comparison between the previous vision of IoT and the MIoT-based vision is not possible, because this last paradigm associates

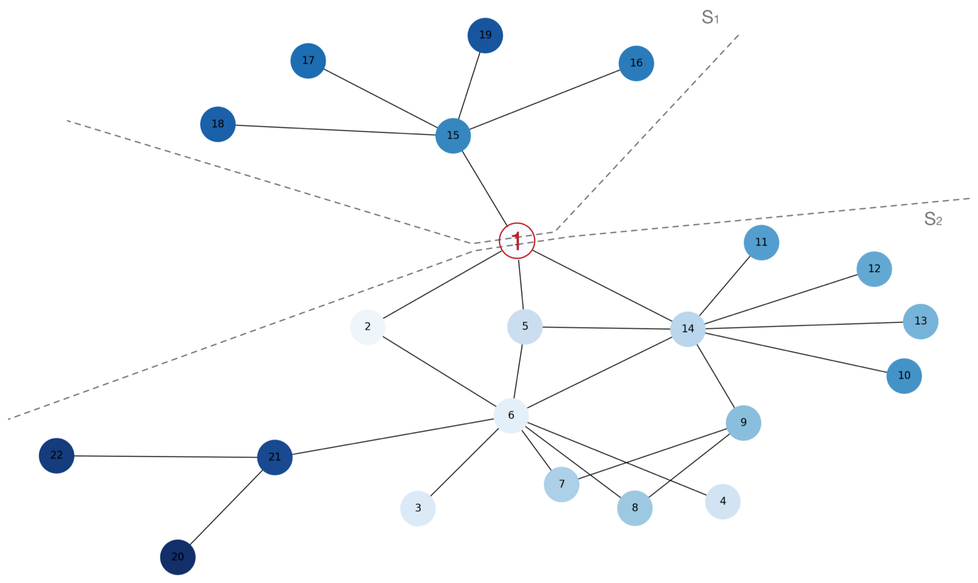


Fig. 8.4: Our case study

more instances with the same object, one for each network joined by it. By contrast, the classical global IoT-based vision considers only objects and does not allow the existence of more instances of the same object. In other words, the global IoT-based vision returns a coarser model of the involved things and their relationships, incapable of verifying if the same object shows different features or behaviors in different subnetworks of the global network. Vice versa, this verification is not only possible, but also natural, in the MIoT paradigm. Indeed, it is sufficient to investigate the different features and behaviors of the various instances of the same object in the IoTs they belong to.

After having made this important premise, which already represents a justification of the usefulness of the MIoT paradigm, we start by presenting our case study by which we aim at showing that the global IoT-based vision can provide imprecise information about the features and the roles of the corresponding things.

Since the global IoT-based vision does not consider object instances, in this case study we assume that all the instances of a cross object have been merged in a unique c-node.

With this considerations in mind, let us consider Figure 8.4. Here, we report a set of nodes each associated with an object. If we consider the global IoT-based vision, all these nodes form a unique IoT where it is possible to distinguish two quite separated subnetworks, called S_1 and S_2 in the figure, connected only thanks to the object represented by Node 1. If we consider the MIoT-based vision, we have two IoTs connected, by means of the object represented by Node 1, to form an MIoT.

Nodes	*Betweenness Centrality*	*Degree Centrality*	*Closeness Centrality*	*Eigenvector Centrality*
1	0.39 (3)	0.19 (4)	0.44 (4)	0.30 (4)
2	0.07 (6)	0.09 (8)	0.41 (5)	0.20 (6)
3	0.00 (11)	0.05 (11)	0.33 ()	0.13 (14)
4	0.00 (12)	0.05 (12)	0.33 ()	0.13 (15)
5	0.07 (7)	0.14 (6)	0.47 (3)	0.34 (3)
6	0.52 (1)	0.38 (1)	0.48 (2)	0.34 (2)
7	0.01 (9)	0.09 (9)	0.34 ()	0.19 (7)
8	0.01 (10)	0.09 (10)	0.34 ()	0.19 (8)
9	0.04 (8)	0.14 (7)	0.37 (6)	0.23 (5)
10	0.0 (13)	0.04 (13)	0.35 (9)	0.13 (10)
11	0.0 (14)	0.04 (14)	0.35 (10)	0.13 (11)
12	0.0 (15)	0.04 (15)	0.35 (11)	0.13 (12)
13	0.0 (16)	0.04 (16)	0.35 (12)	0.13 (13)
14	0.48 (2)	0.38 (2)	0.52 (1)	0.49 (1)
15	0.35 (4)	0.23 (3)	0.35 (7)	0.11 (16)
16	0.0 (17)	0.05 (17)	0.26 (17)	0.03 (19)
17	0.0 (18)	0.05 (18)	0.26 (18)	0.03 (20)
18	0.0 (19)	0.05 (19)	0.26 (19)	0.03 (21)
19	0.0 (20)	0.05 (20)	0.26 (20)	0.03 (22)
20	0.0 (21)	0.05 (21)	0.26 (21)	0.04 (17)
21	0.18 (5)	0.14 (5)	0.35 (8)	0.15 (9)
22	0.0 (22)	0.05 (22)	0.26 (22)	0.04 (18)

Table 8.2: Betweenneess Centrality, Degree Centrality, Closeness Centrality and Eigenvector Centrality, and the corresponding ranks, for all the nodes of the case study of Figure 8.4

Let us focus our attention on this node. Clearly, it is the most important node of this scenario because it is the only one allowing the communication and the cooperation between the nodes of the subnetwork S_1 and the ones of the subnetwork S_2.

However, if we compute the classical centrality measures for the nodes of this network, we have that the rank of Node 1 is not very high in any centrality measure (see Table 8.2). In other words, if we adopt the global IoT-based vision, no centrality measure is capable of capturing the importance of this node. By contrast, the MIoT paradigm is capable alone of intrinsically evidencing the key role played by Node 1, without the need of computing any centrality measure.

With regard to this last observation, we are also aware that, in a real scenario, where the IoTs composing an MIoT are many and the number of c-objects is high, it could be extremely challenging to define a new MIoT-oriented centrality measure. This should be capable of determining the most relevant nodes in an MIoT taking also (but not exclusively) into account if they are c-nodes or not. In the future, we plan to investigate the possibility to define such a measure.

8.3 CDS: a crawler tailored for MIoTs

8.3.1 Motivations underlying CDS

As pointed out in the Introduction, in real cases, when the number of involved things is huge, in order to investigate the main features of an MIoT and to extract useful knowledge from its data, a crawling strategy is mandatory. This strategy must be able to consider not only the instances and their connections in a single IoT (i.e., i-nodes and i-edges), but also the instances of the same objects (along with the corresponding connections) in different IoTs (i.e., c-nodes and c-edges). Furthermore, it must take into consideration that c-nodes and i-nodes have different nature and that c-nodes are more important than i-nodes in an MIoT, which implies that it must be possible to privilege c-nodes over i-nodes, if necessary. Finally, it must allow users to specify how much c-nodes must be privileged over i-nodes. Observe that this problem has a correspondence with the one of finding a crawler specifically tailored for a Social Internetworking Scenario and, therefore, a crawler privileging "me"-edges over intra-network edges and bridges over intra-network nodes.

In the past, several crawling strategies operating in a *single network* (and, therefore, in a *single IoT*) have been proposed. Among them, three very popular ones are Breadth First Search (BFS, for short), Random Walk (RW, for short) and Metropolis-Hastings Random Walk (MH, for short). BFS implements the classical Breadth First Search visit. RW selects the next node to visit uniformly at random among the neighbors of the current node. Both BFS and RW tend to favor power nodes (i.e., nodes having high outdegrees). As a consequence, both of them present bias in some network parameters [221]. MH is a more recent crawling strategy, conceived to unfavor power nodes in such a way as to remove the bias, in BFS and RW, caused by their tendency to favor this kind of node. It was shown that MH performs very well in a single network [166], especially for the estimation of the average degree of nodes. At each iteration, MH randomly selects a node n_j from the neighbors of the current node n_i. Then, it randomly generates a number p, belonging to the real interval $[0,1]$. If $p \leq \frac{outdeg(n_i)}{outdeg(n_j)}$, where $outdeg(n_i)$ and $outdeg(n_j)$ are the outdegrees of n_i and n_j, it selects n_j as the new current node. Otherwise, it maintains n_i as the current node.

The higher the outdegree of a node, the higher the probability that MH discards it. The way of proceeding of MH has been specifically conceived to reach the goal of disfavoring high-degree nodes in such a way as to remove the bias caused by them, as explained above.

In the past, BFS, RW and MH were deeply studied for single networks and it was found that none of them is always better than the other ones. However, no investigaton about the application of these strategies in a set of related IoTs (of which, SIoTs and MIoTs are specific cases) has been carried out. Thus, there is no evidence that they are still valid in this new context. Rather, it is easy to foresee that they will show some weaknesses, since they do not take into account the main actors of related IoTs, i.e., the instances of the same things in different IoTs and their connections (which represent c-nodes and c-edges in the MIoT paradigm).

We expect that these instances and their connections play a crucial role in crawling a set of related IoTs, since they allow different IoTs to be crossed, thus evidencing the main actors of related IoTs, i.e. c-nodes and c-edges, allowing their interconnections. These nodes and edges are not "standard" ones, due to their role. As shown in Section 8.2.2, we cannot see a set of related IoTs just as a unique huge IoT. By contrast, its nature, specificities and behavior must be strongly considered by a crawling strategy that aims to be effective and efficient for a set of related IoTs.

As it will be described in the next section, this original intuition has been fully confirmed by our experimental campaign, which clearly highlights the drawbacks of BFS, RW an MH when passing from a single IoT to a set of related IoTs.

8.3.2 Description of CDS

In the design of CDS, we start by analyzing some aspects limiting BFS, RW and MH in a set of related IoTs (and, therefore, also in an MIoT), in such a way as to overcome them.

BFS performs a Breadth First Search of a local neighborhood of the current node. Now, the average distance between two nodes of a single IoT is generally less than the one between two nodes of different IoTs. In fact, to pass from an IoT to another, it is necessary to cross a c-node and, since, in real cases, c-nodes are (much) less numerous than i-nodes, it could be necessary to generate a long path before reaching one of them. As a consequence, the local neighborhood considered by BFS includes one or a small number of IoTs.

To overcome this problem, a Depth First Search, instead of a BFS, could be performed. For this purpose, the way of proceeding of RW and MH should be included in our crawling strategy. However, since, generally, there is a limited number of c-nodes in an IoT, the simple choice to go in-depth blindly does not favor the crossing from an IoT to another. A solution that addresses the above issues could consist in the implementation of a "non-blind" Depth-First Search that favors c-nodes in the choice of the next node to visit. This is exactly the strategy we have chosen, and the name we give to it, i.e., Cross Node Driven Search (CDS, for short), clearly reflects its way of proceeding.

Observe that this problem has a correspondence with the one of finding a crawler specifically tailored for a Social Internetworking Scenario and, therefore, a crawler privileging "me"-edges over intra-network edges and bridges over intra-network nodes.

However, following exactly the strategy mentioned previously would make it impossible to explore (at least partially) the neighborhood of the current node because the visit would proceed in-depth very quickly and, as soon as a c-node is encountered, there is a cross to another IoT. The overall result of this strategy would be an extremely fragmented crawled sample. To avoid this problem, given the current node, our crawling strategy explores a fraction of its neighbors before performing an in-depth search of the next node to visit.

To formalize our crawling strategy, we need to introduce the following parameters:

- *inf* (i-node neighbors fraction). It represents the fraction of the i-node neighbors of the current node that should be visited. It ranges in the real interval $(0,1]$. When *inf* tends to 1, CDS behaves as BFS. By contrast, when *inf* tends to 0, CDS behaves as MH and RW[1]. In all these cases, CDS inherits all the strengths and the weaknesses of the corresponding strategies. Intermediate values of *inf*, suitably determined (see Section 8.3.3), allow CDS to maximize the pros and to minimize the cons of BFS, RW and MH.
- *cnf* (c-node neighbors fraction). It represents the fraction of the c-node neighbors of the current node that should be visited. It ranges in the real interval $(0,1]$. It allows the tuning of the number of IoT crossings performed by CDS. The higher its value, the higher this number. Clearly, an excessive number of crossings could return a sample involving many IoTs of the MIoT but with a very little number of connections between each pair of IoTs. This could cause, in the Multiple-Network context, the same problem caused by RW in the Single-Network scenario. As a consequence, also for this parameter, a tradeoff is necessary.

[1] To be extremely accurate and precise, this is true if the parameter cnf (that we introduce below) is fixed to 1, in case we want to visit the whole MIoT, or to 0, in case we want to restrict our visit to just one IoT of the MIoT.

For instance, in a configuration where $inf = 0.15$ and $cnf = 0.30$, CDS visits 15% of the i-node neighbors of the current node and 30% of the c-node neighbors of the current node.

We are now able to formalize our crawling strategy. We report its pseudocode in Algorithm 1.

CDS receives: *(i)* an MIoT $\mathcal{M}$, consisting of m IoTs; *(ii)* a non-negative integer n_{it}, denoting the number of iterations that must be still performed; *(iii)* cnf and inf; *(iv)* three sets of nodes, called $SeenNodes$, $VisitedNodes$ and $VisitedCNodes$, whose semantics will be clear in the following. It returns $SeenNodes$ and $VisitedNodes$ after having updated them.

It exploits: *(i)* a function $I(n)$ returning the number of i-node neighbors of the node n; *(ii)* a function $C(n)$ returning the number of c-node neighbors of the node n; *(iii)* two support nodes v and w; *(iv)* a support real number p in the real interval $[0,1]$; *(v)* a support counter c; *(vii)* a support queue $NodeQueue$ of nodes.

First CDS selects a seed node s (not already present in the list $VisitedNodes$ of the nodes already visited) from $\mathcal{M}$ uniformly at random, and inserts it in $NodeQueue$. Then, it starts a cycle that ends when the number n_{it} of iterations to be still performed is 0.

During each iteration, CDS extracts a node v from $NodeQueue$ and inserts it in $VisitedNodes$. At the same time, it inserts all the node neighbors of v in the list $SeenNodes$.

At this point, it computes $C(v)$ to verify if there exist c-node neighbors of v. In the affirmative case, it clears $NodeQueue$[2] and starts to examine these nodes until to either the number of examined c-nodes reaches the maximum value established through cnf or there are no available iterations.

During each of these internal iterations, CDS selects a node w, among the c-node neighbors of v not already present in the set $VisitedCNodes$ of the already visited c-nodes; the selection of w is performed uniformly at random. Then, it generates a real number p in the range $[0,1]$ uniformly at random. If $p \leq \frac{C(v)+I(v)}{C(w)+I(w)}$, then w is inserted in both $NodeQueue$ and $VisitedCNodes$, c is increased of 1 and n_{it} is decreased of 1. Note that the last condition implements the strategy of MH into CDS, in such a way as to let CDS to inherit the pros of MH.

After having processed the c-node neighbors of v, CDS starts to process the i-node neighbors of v in an analogous way. In particular, it selects a node w among the i-node neighbors of v uniformly at random. Then, it generates a number p in the

[2] Observe that this task is performed to privilege c-nodes over i-nodes and to favor crossings from one IoT to another. Indeed, if $NodeQueue$ would have not been cleared, there was the risk to remain in the same IoT or, in any case, to visit a very small number of IoTs.

Algorithm 1 CDS

Notation We denote by $I(n)$ a function returning the number of i-node neighbors of the node n and by $C(n)$ a function returning the number of c-node neighbors of n.

Input $\mathcal{M}$: an MIoT composed of m IoTs; n_{it}: a non-negative integer; cnf, inf: a real number in the range [0,1]; *SeenNodes*, *VisitedNodes*, *VisitedCNodes*: a set of nodes

Output *SeenNodes*; *VisitedNodes*;

Variable v, w: a node

Variable p: a real number in the range [0,1]

Variable c: an integer number

Variable *NodeQueue*: a queue of nodes

```
 1: $NodeQueue := \emptyset$
 2: select a seed node $s$ (not already present in $VisitedNodes$) from $\mathcal{M}$ uniformly at random
 3: insert $s$ in $NodeQueue$
 4: while $n_{it} > 0$ do
 5:     extract a node $v$ from $NodeQueue$
 6:     insert $v$ in $VisitedNodes$
 7:     insert all the nodes adjacent to $v$ in $SeenNodes$
 8:     if $(C(v) \geq 1)$ then
 9:         clear $NodeQueue$
10:         $c := 0$
11:         while $((c < \lceil cnf \cdot C(v) \rceil)$ and $(n_{it} > 0))$ do
12:             let $w$ be one c-node neighbor of $v$ not in $VisitedCNodes$ selected uniformly at random
13:             generate a number $p$ in the real interval $[0,1]$ uniformly at random
14:             if $\left(p \leq \frac{C(v)+I(v)}{C(w)+I(w)}\right)$ then
15:                 insert $w$ in $NodeQueue$ and in $VisitedCNodes$
16:                 $c := c + 1$
17:                 $n_{it} := n_{it} - 1$
18:             end if
19:         end while
20:     end if
21:     if $(I(v) \geq 1)$ then
22:         $c := 0$
23:         while $((c < \lceil inf \cdot I(v) \rceil)$ and $(n_{it} > 0))$ do
24:             let $w$ be one of the i-node neighbors of $v$ selected uniformly at random
25:             generate a number $p$ in the real interval $[0,1]$ uniformly at random
26:             if $\left(p \leq \frac{I(v)}{I(w)}\right)$ then
27:                 insert $w$ in $NodeQueue$
28:                 $c := c + 1$
29:                 $n_{it} := n_{it} - 1$
30:             end if
31:         end while
32:     end if
33:     if $((n_{it} > 0)$ and $(NodeQueue = \emptyset))$ then
34:         goto 37
35:     end if
36: end while
37: if $(n_{it} = 0)$ then
38:     return $SeenNodes, VisitedNodes$
39: else
40:     return $CDS(\mathcal{M}, n_{it}, cnf, inf, SeenNodes, VisitedNodes, VisitedCNodes)$
41: end if
    =0
```

real interval $[0,1]$ uniformly at random and, if $p \leq \frac{I(v)}{I(w)}$, it inserts w into $NodeQueue$, increases c of 1 and decreases n_{i_t} of 1.

Iterations	10	20	30	40	50	60	70	80	90	100
Seen nodes	50	78	107	150	165	163	183	187	181	198
Visited nodes	11	21	34	48	59	68	94	102	105	125
IoT Crossings	4	9	14	17	24	24	33	40	30	43
Visited IoTs	5	6	7	9	9	9	10	10	10	10

Table 8.3: Number of seen nodes, number of visited nodes, number of IoT crossings and number of visited IoTs against the number of iterations performed by CDS

CDS terminates the external cycle started at row 4 when $n_{it} = 0$ or when there are no nodes that can be visited starting from the current seed. In the former case, it returns $SeenNodes$ and $VisitedNodes$. In the latter case, it recursively calls another instance of itself in such a way as to re-start all the previous tasks from another seed node not already visited in the past.

8.3.3 Experimental campaign

We carried out our experiments on the testbed presented in Section 8.2.1. In particular, we performed two kinds of experiment, namely:

- *setting of CDS*; in this case, we aimed to choose the most suitable values of the input parameters of CDS;
- *evaluation of CDS*; in this case, we compared CDS with BFS, RW and MH to quantitatively determine its strengths and weaknesses.

In the next subsections, we present each of these experiments.

Setting of CDS

As pointed out in Section 8.3.2, CDS needs three input parameters that can be used to make it more responsive to our needs. These parameters are: *(i)* inf, i.e. the i-node neighbors fraction that should be visited; *(ii)* cnf, i.e. the c-node neighbors fraction that should be visited; *(iii)* n_{it}, i.e. the maximum number of iterations.

We recall that our testbed consists of 315 nodes; 200 of them are i-nodes, whereas 115 of them are c-nodes.

First, we computed the variation of the number of seen and visited nodes, IoT crossings and visited IoTs against the variation of the number of performed iterations. Obtained results are reported in Table 8.3.

From the analysis of this table, we can see that:

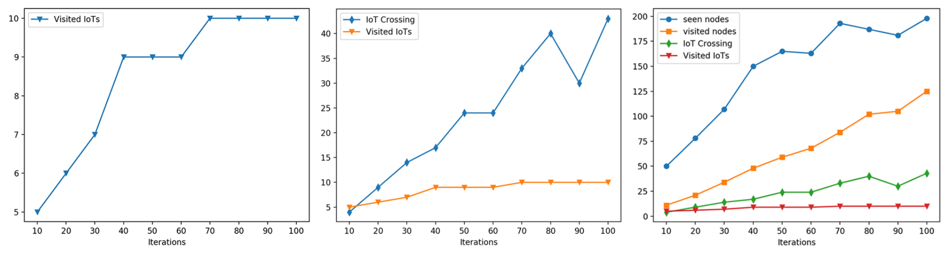


Fig. 8.5: Trends of the number of seen nodes, visited nodes, IoT crossings and visited IoTs against the number of iterations performed by CDS (trends are separated in the first two graphs and put together in the last one)

- after 20 iterations, 24.76% of all nodes are seen, 6.67% of all nodes are visited and 54.55% of IoTs are visited;
- after 50 iterations, 52.38% of all nodes are seen, 18.73% of all nodes are visited and 81.81% of IoTs are visited;
- after 70 iterations, 58.10% of all nodes are seen, 29.84% of all nodes are visited and 90.91% of IoTs are visited;
- after 100 iterations, 62.85% of all nodes are seen, 39.68% of all nodes are visited and 90.91% of IoTs are visited.

Taking into account these observations, as well as the trends of the corresponding measures reported in Figure 8.5, we observe that setting the number of iterations to 70 (or, more formally, setting $n_{it} = 0.22 \cdot |N|$) is a good trade-off between the capability of sampling the highest possible number of the MIoT nodes and the effort required to perform this task.

After having set $n_{it} = 70$, we computed the variation of the number of seen and visited nodes, IoT crossings and, finally, visited IoTs against the variation of the values of inf and cnf. In particular, we considered five possible values of inf (i.e., $inf = 0$, $inf = 0.25$, $inf = 0.50$, $inf = 0.75$, and $inf = 1$) and five possible values of cnf (i.e., $cnf = 0$, $cnf = 0.25$, $cnf = 0.50$, $cnf = 0.75$, and $cnf = 1$). Obtained results are reported in Table 8.4.

From the analysis of this table, we can see that the best values for the four parameters are found when inf is low and cnf is high. This is totally in line with the semantics of these two coefficients, as well as with the role that they play in CDS. In particular, we observe that, if we consider the four parameters overall, the best pair of values is $inf = 0.25$ and $cnf = 0.75$.

Evaluation of CDS

In this experiment, we compared CDS with BFS, RW and MH. In this activity, the first preliminary task was to find reasonable metrics for evaluating the performances

Seen nodes					
	$inf = 0$	$inf = 0.25$	$inf = 0.50$	$inf = 0.75$	$inf = 1$
$cnf = 0$	152	144	159	132	161
$cnf = 0.25$	200	178	201	212	171
$cnf = 0.50$	189	183	206	196	170
$cnf = 0.75$	199	212	204	172	204
$cnf = 1$	208	174	181	181	194

Visited nodes					
	$inf = 0$	$inf = 0.25$	$inf = 0.50$	$inf = 0.75$	$inf = 1$
$cnf = 0$	55	55	56	54	56
$cnf = 0.25$	64	61	65	65	62
$cnf = 0.50$	65	64	70	67	63
$cnf = 0.75$	71	70	69	62	70
$cnf = 1$	70	63	66	65	68

IoT crossing					
	$inf = 0$	$inf = 0.25$	$inf = 0.50$	$inf = 0.75$	$inf = 1$
$cnf = 0$	23	20	22	19	24
$cnf = 0.25$	29	26	31	31	26
$cnf = 0.50$	30	28	36	32	37
$cnf = 0.75$	36	37	34	26	35
$cnf = 1$	35	25	29	29	33

Visited IoTs					
	$inf = 0$	$inf = 0.25$	$inf = 0.50$	$inf = 0.75$	$inf = 1$
$cnf = 0$	9	8	8	8	10
$cnf = 0.25$	10	9	10	10	9
$cnf = 0.50$	9	9	10	9	9
$cnf = 0.75$	9	10	10	9	10
$cnf = 1$	10	9	9	9	9

Table 8.4: Number of seen nodes, visited nodes, IoT crossings and visited IoTs against the variation of inf and cnf

of crawlers that operate on a set of related IoTs. For this purpose, first we extended to the Multiple-Network context the metrics designed for evaluating the performances of crawlers that operate on a Single-Network context. Then, we introduced some other metrics specific for a set of related IoTs.

This section illustrates all our efforts in this direction and the results we have obtained. Specifically, it is organized in three subsections. The first presents our basic evaluation measures. The second describes a combined evaluation measure introduced by us. Finally, the last presents the results of the test that we have performed by means of these measures.

Basic evaluation measures

The basic evaluation measures that we designed for our experimental campaign are the following:

- *Cross Node Ratio (CNR)*: This is a real number, in the interval $[0,1]$, defined as the ratio of the number of crawled c-nodes to the number of all the c-nodes of the MIoT.
- *IoT Crossings (IC)*: This is a non-negative integer and denotes how many times the crawler switches from one IoT to another.
- *Visited IoTs (VI)*: This is a positive integer and measures how many different IoTs are visited by the crawler.
- *Unbalancing (UB)*: This is a non-negative real number defined as the standard deviation of the fraction of nodes discovered for each IoT w.r.t. the overall number of nodes discovered in the sample. UB ranges from 0, corresponding to the case in which each IoT is sampled with the same number of nodes, to a maximum value, corresponding to the case in which all sampled nodes belong to the same IoT.
- *Degree Bias (DB)*: This is a real number defined as the root mean squared error, for each IoT of the MIoT, of the average node degree estimated by the crawler and the one estimated by MH, which is considered the best crawling strategy for the estimation of the degree of a network node in the literature [221, 166]. If the crawled sample does not cover one or more IoTs, then these are not considered in the computation of DB.

If we consider the parallelism between MIoTs and Social Internetworking, we have that, in a Social Internetworking System: *(i)* CNR would return the ratio of the number of bridges discovered to the number of all the nodes in the sample; *(ii)* IC would measure how many times the crawler switches from one social network to another; *(iii)* VI would return how many different social networks are visited by the crawler; *(iv)* UB would represent the standard deviation of the percentages of nodes discovered for each social network w.r.t. the overall number of nodes discovered in the sample; *(v)* DB would denote the root mean squared error, for each social network of the SIS, of the average node degree estimated by the crawler and the one estimated by MH.

As for CNR, IC and VI, the higher their value, the higher the performance of the crawling strategy. By contrast, as far as UB and DB are concerned, the lower their values and the higher the performance of the crawling strategy. Observe that VI allows the evaluation of the crawler's capability of covering many IoTs of the MIoT. With regard to this measure, a further consideration is in order. Indeed, one could think that a fair crawling strategy should sample different IoTs proportionally to their respective overall size. Actually, this crawler behavior could result in incomplete samples in case of a high variance of these sizes. In fact, it could happen that some small IoTs would be not represented, or would be insufficiently represented, in the sample. CNR and IC are related to the coupling degree of the IoTs of the MIoT, whereas DB is related to the average degree.

A combined evaluation measure

Besides some separated metrics, each capturing an important aspect of the crawling strategy, it is certainly important to define a synthetic measure capable of capturing a sort of "overall" crawler behavior. Furthermore, this overall measure should allow users to tune the importance of the five metrics in it, which could be different in different application cases. A reasonable way to do this consists in defining the overall metric as a linear combination of the five ones introduced above, where the coefficients reflect the importance that users want to associate with them. We call *Overall Crawling Quality* (OCQ, for short) this measure and define it as:

$$OCQ = w_{CNR} \cdot \frac{CNR}{CNR_{max}} + w_{IC} \cdot \frac{IC}{IC_{max}} + w_{VI} \cdot \frac{VI}{VI_{max}} + w_{UB} \cdot (1 - \frac{UB}{UB_{max}}) + w_{DB} \cdot (1 - \frac{DB}{DB_{max}})$$

Here, CNR_{max}, IC_{max}, VI_{max}, UB_{max} and DB_{max} are the upper bounds of CNR, IC, VI, UB and DB, which, in a comparative experiment, can be set to the maximum value obtained by the crawlers into consideration. Furthermore, w_{CNR}, w_{IC}, w_{VI}, w_{UB} and w_{DB} are real numbers belonging to the interval $[0,1]$ such that their overall sum is 1.

Before reasoning about the possible values of the five weights of OCQ, we point that the defined metrics are not completely independent of each other. In fact, if $CNR = 0$, then IC and VI are also 0. Furthermore, the value of CNR influences the values of both VI and UB. As a consequence, it is reasonable to assign different weights to the five metrics by associating the highest weights with the most influential ones. To perform this task, we defined an algorithm that is based on the Kahn's approach for topological sorting of graphs [202]. This algorithm uses a data structure called Metric Dependency Graph. This graph has a node n_i for each metric M_i; there exists an edge from n_i to n_j if the metric M_i influences the metric M_j. Each node has associated a weight. Initially all the node weights are set to 0.20 (see Figure 8.6). Our algorithm starts from a node with no outgoing edges and splits the corresponding weight (in equal parts) between itself and the nodes it depends on. Clearly, if a node has no incoming edge, it maintains its weight. After the split of the weight, our algorithm removes all the incoming edges from the corresponding nodes and repeats the previous tasks until all the nodes of the graph have been processed.

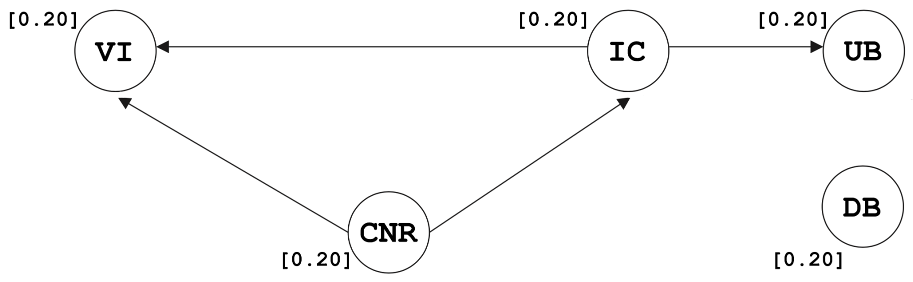


Fig. 8.6: Our Metric Dependency Graph

It is worth pointing out that the node processing order could be not unique, if there exists more than one node with no outgoing edges. However, it is possible to prove that the final metric weights returned by our algorithm do not depend on the adopted node processing order.

It is possible to formalize the previous algorithm in a closed formula allowing us to compute the weight w_i associated with each node n_i of the Metric Dependency Graph. In particular, we have:

$$w_i = \frac{\frac{1}{1+indeg(n_i)} \cdot \left(w + \sum_{n_j \in OSet(n_j)} w_j\right)}{\sum_{k=1}^{5} w_k}$$

Here, $indeg(n_i)$ is the indegree of n_i, w is a number representing the initial weight of n_i (that, in our case, is 0.20 for all the five nodes) and $OSet(n_j)$ is the set of the nodes reachable from n_j through its outgoing edges. This formula indicates that w_i consists of two components; the former is the initial weight w; the latter represents the weight gained by n_i thanks to the fact that other nodes depend on it. In turn, n_i splits its weight among the nodes it depends on and itself; this is handled by the term $1 + indeg(n_i)$. The denominator of the formula is used to normalize w_i in the interval $[0, 1]$.

By applying the previous formula to our five metrics we obtained the following weight values: $w_{CNR} = 0.45$, $w_{IC} = 0.18$, $w_{VI} = 0.07$, $w_{UB} = 0.10$ and $w_{DB} = 0.20$.

Test results

We are now ready to analyze the performances of CDS, BFS, RW and MH when applied on an MIoT. For this activity, we used the testbed described in Section 8.2.1. We applied BFS, RW and MH to each MIoT by regarding it as a unique graph. Furthermore, in order to make the MIoT graph totally compliant with the inputs classically received by BFS, RW and MH, we considered a "condensed version" of the MIoT graph by putting just one node for each c-object. We run CDS with $inf = 0.25$ and $cnf = 0.75$, which, as pointed out in Section 8.3.3, are the parameter settings that

	CDS	BFS	RW	MH
CNR	0.211	0.064	0.057	0.053
IC	9.133	6.400	2.333	2.333
VI	27.000	10.933	7.467	7,467
DB	3.476	0.844	0.026	0
UB	0.142	0.269	0.236	0.199

Table 8.5: Values of the five metrics obtained by CDS, BFS, RW and MH

guarantee the maximum number of IoT crossings. We report the obtained results in Table 8.5.

We recall that the higher the values of CNR, IC and VI and the lower the values of DB and UB, the better the performances of the strategies into examination.

From the analysis of Table 8.5, we can observe that, as far as CNR, IC, VI and UB are concerned, CDS outperforms BFS, RW and MH. For instance, the value of CNR obtained by CDS is about 230% (resp., 273%, 296%) better than the one of BFS (resp., RW, MH).

The only metric for which CDS shows a worse performance than the other strategies is DB. In fact, as for this metric, the value obtained by MH is 0. This was expected because DB is measured having the value of MH as the reference one since, in the literature, it is well known that MH guarantees the best Degree Bias among all crawling strategies [221, 166]. BFS and RW obtain values of DB near to the ones of MH, whereas CDS shows the worst performance, even if it is still acceptable. The results obtained by CDS for DB were also expected because the purpose of this crawler is to privilege c-nodes over i-nodes. As a consequence, when a c-node is encountered, the node queue is cleared (see Line 4 in Algorithm 1) in such a way as to stimulate the IoT crossings and, ultimately, the visit of c-nodes, which is the main objective of our crawler. Clearing the node queue produces a distortion because several nodes directly connected to the current one will not be put in the set of visited nodes. In turn, this produces an effect in the degree bias and, ultimately, the worst performance of CDS, as far as the value of DB is concerned. However, observe that these results are obtained with the default configuration of CDS (i.e., $inf = 0.25$ and $cnf = 0.75$). Actually, if necessary, it is possible to configure CDS in such a way that it behaves as RW and MH, which present the best values of DB. In fact, as seen in Section 8.3.2, this behavior can be obtained by making inf tend to 0.

Since there is one parameter for which CDS shows the worst results w.r.t. the other three crawlers, it is particularly important the computation of the values of OCQ, because this parameter summarizes the overall performance of the crawlers

	CDS	BFS	RW	MH
Configuration A	0.695	0.433	0.383	0.409
Configuration B	0.747	0.410	0.399	0.407

Table 8.6: Values of OCQ obtained by CDS, BFS, RW and MH for the two weight configurations into examination

into examination. We computed the values of OCQ for both the configuration that sets all the metric weights to 0.20 (we call it "Configuration A" in the following) and the one that takes the parameter dependencies into account ($w_{CNR} = 0.45$, $w_{IC} = 0.18$, $w_{VI} = 0.07$, $w_{UB} = 0.10$ and $w_{DB} = 0.20$ - we call it "Configuration B" in the following). In Table 8.6, we report the obtained results (we recall that the higher the value of OCQ and the better the performance of the corresponding crawler).

From the analysis of this table we can observe that, in both cases, CDS outperforms BFS, RW and MH. Interestingly, in the configuration taking the Metric Dependency Graph into account, CDS obtains even better results than in the other one.

In our opinion, these results clearly evidence that, in an MIoT scenario:

- The crawling strategies defined for single networks do not perform well because they do not consider the important differences existing between c-nodes and i-nodes and between c-edges and i-edges.
- A cross node centered crawler, like CDS, shows very satisfying results and, certainly, indicates a way to go for further crawler strategies specifically designed to operate on a set of related IoTs.

8.4 Analytical discussion

In this section, we propose an analytical discussion aiming at comparing our model and approach with other, more or less conventional, ones. We start by observing that, in the last years, the interest and the attention towards IoTs and sensor networks are enormously increased. This has led, and is currently leading, to a large variety of models and approaches. Some, very common and particularly interesting, families of approaches that can be recognized are the ones based on:

- fuzzy logic;
- neural networks;
- hierarchical models.

In the following, we present a comparison between our approach and each of these families.

Fuzzy logic based approaches allow the possibility that a thing belongs to more sets simultaneously [220, 290, 320, 43]. Also in our model, an object can belong to more IoTs, thanks to its instances. However, differently from fuzzy logic based approaches, in our case, when there is the instance of an object in an IoT, this means that the object surely belongs to that IoT. Instead, in fuzzy logic based approaches, an object belongs to a given IoT with a certain plausibility.

Neural network based approaches can exploit the potentialities of a highly dynamic structure, such as neural network [110, 308]. The dynamism of the support data structure certainly represents an analogy with our approach, which is based on an equally dynamic structure, i.e. social network. However, even if these two support data structures are graph based, they have totally different objectives. Indeed, neural networks are well suited for performing classifications and for handling non-linear scenarios. Social Networks are centered on node cooperation, node centralities and information diffusion. Furthermore, in an MIoT, there is no need to handle non-linearity.

Hierarchical approaches are certainly a bit more different from the MIoT paradigm than the other two families considered above [236, 357]. In fact, they mainly aim at detecting (more or less) hidden relationships among objects at different abstraction levels. Even if such a family of approaches is quite far from the current MIoT paradigm, it could represent a good starting point for an evolution of our model. Indeed, the current MIoT architecture consists of only two levels of control. Increasing the hierarchy length and, therefore, the granularity level, would allow the definition of more instances of one object in the same IoT, which could provide our model with a higher refinement capability.

Finally, to the best of our knowledge, the approach most similar to ours is the one described in [107]. In fact, analogously to what happens in an MIoT, in this approach an object is described by means of an ennuple. This choice allows an ordered representation of an object, its activities and its instances. However, very differently from our approach, the one of [107] models data coming from an IoT as a big data stream. This forces a kind of sampling allowing only the registration of the probability that a given object is in a given condition or in a given place. Interestingly, the approach of [107] provides the user with a strong support for data cleaning and integration. Instead, the MIoT paradigm does not address this issue because it assumes that cleaning and integration tasks have been performed before the construction of the MIoT graph.

8.5 Related work

Several years have passed since the IoT paradigm was introduced [52, 55, 265, 295]. During this period, the term "Internet of Things – IoT" has been associated with a huge variety of concepts, technologies and solutions. For instance, in the last few years, new technologies, such as Big Data [99] and Social Networking, have been applied to IoT and have changed, and are currently changing, the very definition of this term. What IoT will become in the future depends on the evolution of these technologies [355].

The current research on IoT focuses on the capability of connecting every object to the Internet. This way of thinking IoT led to the Web of Things (hereafter, WoT) paradigm [175, 174, 193] and to the application of Social Networking to the IoT domain [54]. In the next future, these technologies will be combined with other ones, such as Information Centric Networks [339, 389, 390, 47, 302, 48, 293] and Cloud [133, 349, 207]. As a matter of fact, the strengths of these last ones are exactly the features necessary to overcome the weaknesses of the current IoT concept [380]. Some examples of this combination can be already found in the literature [150, 172, 364, 363].

Significant efforts have been made to apply the Social Networking ideas to the IoT domain. Actually, the implementation of reliable IoTs [53] passes through the definition of a complex architecture capable of managing services. In this research direction, the authors of [294] propose CASCOM, a model devoted to simplify the interaction between consumers and data in an IoT context. It is also necessary that this complex architecture enables a complete connectivity among things [220], guarantees quick reactions to frequent state variations and, finally, ensures a good scalability.

Furthermore, as IoT is based on the Internet, it must address the same security issues characterizing this network [199]. Therefore, the development of new architectures capable of fulfilling security and privacy requirements is in order [392].

The first attempts to apply Social Networking to the IoT domain can be found in [173, 275, 218, 189]. In these papers, the authors propose to use human social network relationships to share services provided by a set of things.

An important step forward is performed in [53], where the SIoT paradigm is introduced. Here, the authors propose an approach to creating relationships among things, without requiring the owner intervention. Thanks to this idea, things can autonomously crawl the network to find services and resources of their interest provided by other things. In [56], the same authors clearly highlight what are the main strengths of SIoT. Specifically: *(i)* the SIoT structure can be dynamically modified to ensure network navigability and to find new things; *(ii)* scalability is guaranteed, like in human social networks; *(iii)* a level of trustworthiness between things can be established; *(iv)* the past social network approaches can be redefined to solve problems typical of the IoT context [281].

Today, the connection level of humans and things is continuously increasing, so that it appears reasonable to start to investigate the "network of networks" scenario, thus passing from Social Networking to Social Internetworking. One of the most interesting attempts in this direction is Social Internetworking System (hereafter, SIS); it regards the connection of several human networks to form a network of human networks [87, 276]. The strength of SIS resides in the fact that this structure is capable of interconnecting users joining different social networks. In this new scenario, concepts and tools of Social Network Analysis can be adapted to evaluate the main features concerning the interactions between users belonging to the same network or to different networks. This new paradigm aims at guaranteing a tradeoff between the autonomy of each network of the SIS and the possibility of increasing power, efficiency and effectiveness, obtained through the interaction of the networks of the SIS. To the best of our knowledge, no architecture similar to SIS has been proposed for networks of things yet.

In [54], the authors point out that there are still several open issues that must be investigated in the SIoT paradigm. In particular, making things capable of establishing heterogeneous social relationships requires specific investigations and new approaches. Among them, the most relevant ones for our context are: *(i) Defining inter-objects relationships*. This task requires a correct digital representation of a smart object and the definition of a methodological and technological solution capable of crawling and discovering other (possibly heterogeneous) objects, with which interactions can be established. *(ii) Modeling the new social graphs thus obtained*, in such a way as to characterize them and to define new algorithms for performing their analysis.

Crawling represents a key issue for the implementation of the IoT paradigm. The necessity of addressing this issue is mentioned in many papers (e.g., [54, 243, 326, 161, 136], to cite a few). In spite of this high demand, just few papers addressing this problem can be found in the past literature on IoTs. Most of the approaches proposed in these papers focus on the creation of search engines conceived to operate on IoT [243, 246] or, more often, on the Web of Things [354, 113]. In [354], an accurate survey on this last research area is presented.

In [153], the authors propose a geo-based crawler for IoT aiming at minimizing inter-site communication costs. Every site uses its own crawler that is provided with some predefined rules for fetching and parsing the Web. In [136], a framework to automatize the search, and the next classification, of services belonging to a digital health ecosystem, is proposed. This framework exploits both a focused web crawler, which explores the network, and a social classification system. In [230], the authors propose an approach aimed at improving the existing web crawlers, when they operate on IoT, and to catch up the fingerprints of the IoT nodes. This approach is based on an incremental crawler, which periodically classifies nodes in such a way as to ensure the highest classification accuracy for the most important ones.

In [227], the crawling problem is approached from a different perspective. Indeed, one of the main problems in a network of things is battery consumption. To avoid it, in most cases, sensors perform a working-sleeping duty cycle. The authors of [227] model the crawling problem as a scheduling one and define a sleep-aware schedule method called EasiCrow. This method is well suited to crawl sensors with an asynchronous sleeping cycle. In [325], the authors, starting from the assumption that things are becoming the major producers and consumers of data, propose a system to extract data from different sources. Once data has been acquired, this system provides suitable interfaces allowing both humans and machines to share and dynamically search the services of their interest.

9

A proposal to uniformly handle heterogeneous metadata sources

Metadata have always played a key role in favouring the cooperation and integration of different and heterogeneous data sources. With the advent of data lakes the relevance of such information is growing up because a correct metadata management could represent one of the key features that can led to success the data lake paradigm. In fact, data lakes accept and store every data in their raw formats and not to guarantee an efficient metadata management could rapidly drag the data lake into what scientists call Data Swamp, which can be seen as a disorganized and messy set of information. For this reason, the necessity to define new models and paradigms for metadata representation and management appears crucial in the data lake scenario. We propose a new network model to represent data lakes on which we base different applications and proposals. In detail, in this Chapter we describe a new approach that leverages it to give a structure to unstructured sources.

9.1 Introduction

Metadata have always played a key role in favouring the cooperation of heterogeneous data sources [129, 74, 303]. This role was already relevant in the past architectures (e.g., Cooperative Information Systems and Data Warehouses) but has become much more crucial with the advent of data lakes [146]. Indeed, in this new architecture, metadata represent the only possibility to guarantee an effective and efficient management of data source interoperability. As a proof of this, the main data lake companies are performing several efforts in this direction (see, for instance, the metadata organization proposed by Zaloni, one of the market leaders in the data lake field [280]). For this reason, the definition of new models and paradigms for metadata representation and management represents an open problem in the data lake research field.

We propose a new metadata model well suited for data lakes. Our model starts from the considerations and the ideas proposed by data lake companies (in particular, it starts from the general metadata classification also used by Zaloni [280]).

However, it complements them with new ideas and, in particular, with the power guaranteed by a network-based and semantics-driven representation of metadata. Thanks to this choice, our model can benefit from all the results already found in network theory and semantics-driven approaches. As a consequence, it can allow a large variety of sophisticated tasks that the metadata models currently adopted do not guarantee. For instance, it allows the definition of a structure for unstructured data, which currently represent more than 80% of available data sources. Furthermore, it allows the extraction of thematic views from data sources [59], i.e., the construction of views concerning one or more topics of interest for the user, obtained by extracting and merging data coming from different sources. This problem has been largely investigated in the past for structured and semi-structured data sources stored in a data warehouse [336], and this witnesses its extreme relevance.

Metadata are not always provided with data. For this reason, metadata generation received much attention in the literature. According to [37], metadata relative to a data source are currently generated by crawlers, by professional metadata creators, or, finally, by source creators. Generating metadata by means of automatic crawlers has great advantages, such as low cost and high efficiency; however, in some cases, the quality of generated metadata could be poor. In this context, it could be extremely useful the support of several mechanisms for controlling the quality of metadata, as well as the aid of metadata professionals, such as catalogers and indexers; these are people who have had a formal training and are efficient in using metadata. Generally, they produce high-quality metadata. However, it has been observed that, in some cases, even metadata generated by professionals or by source authors may have poor quality and might hamper, rather than aid, the usage of the corresponding sources. This happens because most authors have little previous knowledge on metadata creation [37].

As pointed out in [288], the widespread adoption of several mechanisms for controlling the quality of metadata witnesses a strong awareness of the importance of having high-quality metadata at disposal. However, despite the relevance and impact of metadata quality are universally recognized in the literature, there is no agreement yet on what metadata quality actually means. Obviously, this implies, among the other things, the impossibility of introducing systematic approaches to its automatic measurement and enhancement [348]. Metadata quality assurance should be verified simultaneously to metadata creation [287]. Indeed, poor quality of metadata negatively affects the performance of systems using them, and affects the overall user satisfaction. Quality assurance procedures are generally complemented by manual quality review and, if necessary, by the assistance of the technical staff during the process of metadata creation. Other mechanisms, such as metadata creation guidelines (sometimes embedded into the metadata creation system) and metadata generation tools, are on the rise.

The great relevance given to the improvement of the metadata quality, is observed in the study presented in [204]. Here, the authors introduce a quality measure and analyze the metadata quality in the Europeana context over the years. They observe that the metadata quality improves not only in new collections but also in the same collection over the years.

As pointed out in [288], in the metadata generation process, accuracy and consistency are prioritized over completeness, whereas the semantics of metadata elements is perceived to be less important. This, in principle, might be an issue for our approach, since it strongly relies on semantics; however, the authors of [288] also point out that semantic overlaps and ambiguities are by far the two most critical factors. Actually, as our approach exploits thesauruses, string, and semantic similarities to relate keywords, these negative factors are significantly mitigated.

As for the capability of handling unstructured sources, our approach is provided with a preliminary step capable of "structuring" unstructured sources, i.e., of (at least partially) deriving a structure for them. This is possible because our approach assumes that each unstructured source (e.g., a video, an audio, an image, a text) has associated a list of keywords describing it[1]. The "structuring" process is based exactly on these keywords. This is another main contribution of this study, which, generally speaking, allows the unstructured sources to be uniformly handled as the structured and the semi-structured ones.

With regard to this aspect, some clarifications of what we intend with the terms "structured" and "semi-structured" sources are in order. In particular, we use these terms as they are generally adopted in databases and information systems research field. Here, a structured source consists of some concepts, each having a precise set of attributes and relationships with other concepts of the source. A semi-structured source has similar characteristics, but the set of attributes and relationships characterized a given concept is handled in a more flexible fashion. Indeed, given a property p or a relationship r of a concept c, some instances of c might have exactly one instance of r and/or one instance of p; other instances of c might have more instances of r and/or more instances of p; finally, other instances of c might have no instances of r and/or no instances of c. A classical example of structured sources is a relational databases (that can be conceptually represented by means of an E/R diagram). A classical example of an unstructured source is an XML document (that can be conceptually represented by means of a DOM).

[1] Here, we are assuming that the list is ordered and the order the order is the one in which the queries appear in the list

Unstructured sources are videos, audios, images or texts. They do not generally have a conceptual representation showing their concepts, along with the corresponding properties and relationships. However, they are generally provided with a set of keywords, denoting the main concepts they are representing. The purpose of our approach for "structuring" unstructured sources is exactly the derivation of the relationships existing among the concepts represented by the keywords associated with the unstructured sources. If we are capable of performing this task, unstructured sources can be handled similarly to structured and semi-structured ones. Furthermore, their analysis and management could benefit from the wide amount of results found in the past for structured and semi-structured sources. Finally, the integration, the cooperation and the simultaneously querying of structured, semi-structured and unstructured sources are possible.

As for the lightweightness of our approach, we observe that, in a big data scenario like the one currently characterizing the information system field, a new proposed approach must take scalability into a primary consideration [233, 229]. As a matter of fact, the sources interacting in every task are always very numerous and large (think, for instance, of a data lake constructed to support data analytics in an organization) and the time allowed for each transaction is very limited (think, for instance, of streaming applications). As a consequence, even approaches considered very scalable in the past (such as DIKE [286], MOMIS [73], and Cupid [245]) are not adequate anymore. In our opinion, the tests performed to evaluate our approach and described in Section 10.3, confirm that our approach is really capable of satisfying the lightweightness requirement without sacrificing, if not in a very small extent) the result accuracy.

9.1.1 Typologies of metadata

Following what it is said in [280], metadata can be divided into three categories, namely: *(i) Business metadata*, which include business rules (e.g., the upper and lower limit of a particular field, integrity constraints, etc.); *(ii) Operational metadata*, which include information generated automatically during data processing (e.g., data quality, data provenance, executed jobs); *(iii) Technical metadata*, which include information about data format and schema. Based on this reasoning, $\mathcal{M}_k$ can be represented as the union of three sets $\mathcal{M}_k^B \cup \mathcal{M}_k^O \cup \mathcal{M}_k^T$.

As an advancement of the model of [280], we observe that these three subsets are intersected with each other (as shown in Figure 9.1). For instance, since business metadata contain all business rules and information allowing to better understand data fields, and since the data schema is included in the technical metadata, we can conclude that data fields represent the perfect intersection between these two subsets. Analogously, technical metadata contain the data type and length, the possibility that a field can be NULL or auto-incrementing, the number of records, the data format and some dump information. These last three things are in common with operational metadata, which contain information like sources and target location and the file size as well. Finally, the intersection between operational and business metadata represents information about the dataset license, the hosting server and so forth (e.g. see the DCMI Metadata Terms).

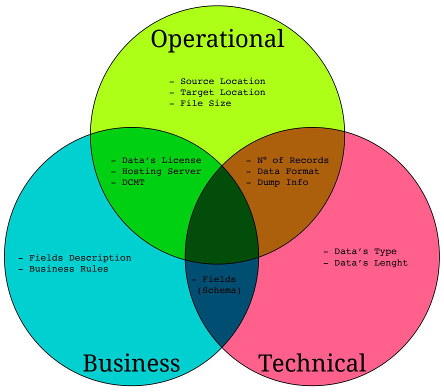


Fig. 9.1: The three kinds of metadata proposed by our model.

In this study, we focus on business metadata and on the intersection between them and the technical ones. This intersection contains the data fields, both domain description and technical details. For instance, in a structured database, this intersection contains the attributes of the tables. Instead, in a semi-structured one, it consists of the names of the (complex or simple) elements and attributes of the schema. Finally, in an unstructured source, it could consist of a set of keywords generally adopted to give an idea of the source content.

9.2 A network-based model for uniformly represent heterogeneous sources

In this section, we present a network-based model for uniformly representing data sources of different formats, structured, semi-structured and unstructured. This model will be used as root on which the other proposals are based. In order to understand the peculiarities of our model, we assume to have a set DS of m data sources of interest possibly characterized by different data formats.

$$DS = \{D_1, D_2, \cdots, D_m\}$$

Each data source D_k has associated a rich set $\mathcal{M}_k$ of metadata. We indicate with $\mathcal{M}_{DS}$ the repository of the metadata of all the data sources of DS:

$$\mathcal{M}_{DS} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_m\}$$

Given the source D_k, in order to represent the information content stored in $\mathcal{M}_k$, our model starts from a notation typical of XML, JSON and many other semi-structured data models. According to this notation, Obj_k denotes the set of all the objects stored in $\mathcal{M}_k$. Obj_k consists of the union of three subsets:

$$Obj_k = Att_k \cup Smp_k \cup Cmp_k$$

where:

- Att_k denotes the set of the attributes of $\mathcal{M}_k$;
- Smp_k indicates the set of the simple elements of $\mathcal{M}_k$;
- Cmp_k represents the set of the complex elements of $\mathcal{M}_k$.

Here, the meaning of the terms "attribute", "simple element" and "complex element" is the one typical of semi-structured data models.

$\mathcal{M}_k$ can be also represented as a graph:

$$\mathcal{M}_k = \langle N_k, A_k \rangle$$

N_k is the set of the nodes of $\mathcal{M}_k$. There is a node n_{k_j} in N_k for each object o_{k_j} in Obj_k. According to the structure of Obj_k, N_k consists of the union of three subsets:

$$N_k = N_k^{Att} \cup N_k^{Smp} \cup N_k^{Cmp}$$

where N_k^{Att} (resp., N_k^{Smp}, N_k^{Cmp}) denotes the set of the nodes corresponding to Att_k (resp., Smp_k, Cmp_k). There is a bi-univocal correspondence between a node of N_k and an object of Obj_k. Therefore, in the following, we will use the two terms interchangeably. Each node has associated a name that identifies it in the schema which the corresponding element or attribute belongs to.

Let x be a complex element of $\mathcal{M}_k$. We denote by Obj_{k_x} the set of the objects directly contained in x and by N_x^{Obj} the set of the corresponding nodes. Finally, let x be a simple element of $\mathcal{M}_k$. We indicate by Att_{k_x} the set of the attributes directly contained in x and by N_x^{Att} the set of the corresponding nodes.

A_k denotes the set of the arcs of $\mathcal{M}_k$. It consists of three subsets:

$$A_k = A'_k \cup A''_k \cup A'''_k$$

where:

- $A'_k = \{(n_x, n_y, L_k) | n_x \in N_k^{Cmp}, n_y \in N_{n_x}^{Obj}\}$; in other words, there is an arc in A'_k from a complex element of $\mathcal{M}_k$ to each object directly contained in it. L_k represents the label of A'_k.

- $A''_k = \{(n_x, n_y, L_k) | n_x \in N_k^{Smp}, n_y \in N_{n_x}^{Att}\}$; in other words, there is an arc in A''_k from a simple element of $\mathcal{M}_k$ to each attribute directly contained in it. L_k represents the label of A''_k.
- $A'''_k = \{(n_x, n_y, L_k) | n_x \in N_k, n_y \in N_k$, D_k is unstructured and $\sigma(n_x, n_y)$=`true`$\}$. Here, $\sigma(n_x, n_y)$ is a function that receives two nodes and returns `true` if there exists a similarity between n_x and n_y. For instance, $\sigma(n_x, n_y)$ could return the semantic similarity of the concepts represented by n_x and n_y or the semantic and string similarity of the names identifying n_x and n_y in the corresponding schema.

As for the label L_k associated with each arc, in the current version of this model, it is `NULL` for the arcs of A'_k and A''_k. However, we do not exclude that, in the future, enrichment of our model might lead us to use this field for storing some knowledge. Instead, L_k has an important meaning for the arcs of A'''_k. In fact, as will be clear in Section 10.2, it is used to denote the strength of the correlation between n_x and n_y.

In Listing 9.1 we summarize our model by representing it through a JSON like template.

```
        {
        "Metadata": {
                "Object": {
                        "Business": {
                                "Complex": {
                                        "Simple": {
                                                "Attribute": []
                                        }
                                }
                        },
                        "Technical": [],
                        "Operational": []
                }
        }
}
```

Listing 9.1: Representation of the model via a JSON like template

9.3 Providing a structure to unstructured sources

Our network-based model for uniformly representing and handling data sources with disparate formats is perfectly fitted for semi-structured sources. Indeed, it is sufficient:

- to derive the metadata of the source (if not yet provided) by applying one of the several techniques and tools defined for this purpose w.r.t. the various kinds of formats;
- to define a complex element to represent the source as a whole;
- to introduce a complex element, a simple element and an attribute for each complex element, simple element and attribute present in the metachema of the source;

- to define an arc of A'_k from the source to the root of the document;
- to introduce an arc of A'_k or A''_k for each relationship existing among the objects composing the source metadata.

Clearly, our model is sufficiently powerful to represent structured data. Indeed, it is sufficient:

- to derive the E/R schema of the source (if not yet provided) by performing a classical database reverse engineering activity;
- to define a complex element to represent the source as a whole;
- to introduce a complex element for each entity in the E/R schema and an attribute for each attribute of the schema;
- to define an arc of A'_k from the complex element corresponding to the source to each complex element associated with an entity of the E/R schema;
- to introduce an arc of A''_k from an entity to each of its attributes;
- to define an arc of A'_k for each one-to-many relationship of the E/R schema; this arc is from the entity participating to the relationship with a maximum cardinality equal to 1 to the entity participating with a maximum cardinality equal to N;
- to model a many-to-many relationship without attributes as a pair of one-to-many relationships and to model them accordingly;
- to model a many-to-many relationship R with attributes connecting two entities E_1 and E_2 as an entity having the same attributes as R and linked to E_1 and E_2 by means of two one-to-many relationships; the new entity and the new relationships are then suitably modelled by applying the rules defined in the previous cases.

Paradoxically, the highest difficulty regards unstructured data because it is worth avoiding a flat representation consisting of a simple element for each keyword provided to denote the source content. As a matter of fact, this flat representation would make the reconciliation, and the next integration, of an unstructured source with the other semi-structured and structured sources of DS very difficult. This is a very challenging issue to address. Since it is very important for the current scenario where more than 80% of available sources are unstructured, we will discuss it in detail in the next section.

In the following, we propose our approach to "structuring" unstructured sources. It is in itself a major issue in the current information system scenario and, at the same time, plays a key role to provide a model on which base further investigations and proposals.

Our approach assumes that each unstructured source into consideration (e.g., a video, an audio, an image, a text) is provided with a list of keywords describing it[2]. They will play a key role, as will be clear in the following. We observe that this assumption is not particularly strong or out of place. As a matter of fact, in the reality, most video, image or audio providers associate a list of keywords (sometimes, in the form of tags) with the contents they deliver. As for text, representing keywords can be also easily derived through suitable techniques, like TF-IDF [251].

Our approach consists of four phases, namely: *(1)* creation of nodes; *(2)* management of lexical similarities; *(3)* management of string similarities; *(4)* management of (temporary) duplicated arcs. We describe these phases below.

- **Phase 1.** During this phase, our approach creates a complex node representing the source as a whole and a simple node for each keyword[3]. Furthermore, it adds an arc of A'_k from the node associated with the source to any node corresponding to a keyword[4]. Initially, there is no arc between two keywords. To determine the arcs to add, the next phases are necessary.
- **Phase 2.** During this phase, our approach handles lexical similarities. For this purpose, it leverages a suitable thesaurus. Taking the current trends into account, this thesaurus should be a multimedia one; for this purpose, in our experiments, we have adopted BabelNet [271]. In particular, our approach adds an arc of A'''_k from the node n_{k_1}, corresponding to the keyword k_1, to the node n_{k_2}, corresponding to the keyword k_2, and vice versa, if k_1 and k_2 have at least one common lemma[5] in the thesaurus. Furthermore, it transforms the nodes n_{k_1} and n_{k_2} from simple to complex. The new arcs have a label corresponding to the number of common lemmas for k_1 and k_2 in the thesaurus.
- **Phase 3.** During this phase, our approach derives string similarities and states that there exists a similarity between two keywords k_1 and k_2 if the string similarity degree $kd(k_1,k_2)$, computed by applying a suitable string similarity metric on k_1 and k_2, is "sufficiently high" (see below). In this case, it adds an arc of A'''_k

[2] Here, we are assuming that the list is ordered and the order is the one in which the queries appear in the list.

[3] Here and in the following, to make the presentation smoother, we use the term "complex node" to indicate a node belonging to N_k^{Cmp} and the term "simple node" to denote a node of N_k^{Smp}.

[4] Here and in the following, to make the presentation smoother, we use the term "source" (resp., "keyword") to denote both the source (resp., a keyword) and the corresponding node associated with it.

[5] We use the term "lemma" according to the meaning it has in BabelNet [271]. Here, given a term, its lemmas are other objects (terms, emoticons, etc.) contributing to specify its meaning.

from n_{k_1} to n_{k_2} and an arc of A_k''' from n_{k_2} to n_{k_1}. Both of them have $kd(k_1,k_2)$ as their label. We have chosen N-Grams [215] as string similarity metric because we have experimentally seen that it provides the best results in our context. Again, if n_{k_1} and n_{k_2} are simple, our approach transforms them into complex. In particular, we have chosen bi-grams as the best trade-off between accuracy and costs. In fact, mono-grams would require a lower cost but they would also return a lower accuracy than bi-grams. By contrast, tri-grams would guarantee a very high accuracy but at the expense of the computational cost which would be excessive.

Now, we illustrate in detail what "sufficiently high" means and how our approach operates. Let $KeySim$ be the set of the string similarities for each pair of keywords of the source into consideration. Each record in $KeySim$ has the form $\langle k_i,k_j,kd(k_i,k_j)\rangle$. Our approach first computes the maximum keyword similarity degree kd_{max} present in $KeySim$. Then, it examines each keyword similarity registered therein. Let $\langle k_1,k_2,kd(k_1,k_2)\rangle$ be one of these similarities. If $((kd(k_1,k_2) \geq th_k \cdot kd_{max})$ and $(kd(k_1,k_2) \geq th_{kmin}))$, which implies that the keyword similarity degree between k_1 and k_2 is among the highest ones in $KeySim$ and that, in any case, it is higher than or equal to a minimum threshold, then it concludes that there exists a similarity between n_{k_1} and n_{k_2}. We have experimentally set $th_k = 0.70$ and $th_{kmin} = 0.50$.

Observe that the choice to consider string similarities, in particular the one to adopt N-Grams as the technique for detecting string similarities, makes our approach robust against mispelling errors possibly present in the keywords. In fact, as shown in [185], N-Grams is well suited to handle also this kind of error.

- **Phase 4.** This phase is devoted to handle possible simultaneous presences of both lexical and string similarities for the same pair of keywords. Indeed, it may occur that, for a pair of nodes n_{k_1} and n_{k_2}, there are two arcs from n_{k_1} to n_{k_2} belonging to A_k''' and generated by both lexical and string similarities, and two arcs from n_{k_2} to n_{k_1}. In this case, the two arcs from n_{k_1} to n_{k_2} corresponding to these two forms of similarities must be merged in only one arc, which has associated a label denoting both the number of common lemmas between k_1 and k_2 in BabelNet and the value of $kd(k_1,k_2)$. The same happens for the two arcs from n_{k_2} to n_{k_1}.

From this description, it emerges that, at the end of our approach, given two nodes n_{k_1} and n_{k_2}, four cases may exist, namely:

1. There is no arc from n_{k_1} to n_{k_2}.
2. A pair of arcs derived from a lexical similarity links them. In this case, the two arcs actually coincide (also in their labels); therefore, one of them can be re-

moved. Note that the choice of the arc to be removed has deep implications in the definition of the topology of the corresponding network. Indeed, one of the two nodes involved (i.e., the source node of the maintained arc) will be certainly a complex node, whereas the other one may be a simple node (if no other arc starts from it) or a complex node (if at least another arc, different from the removed on, starts from it). In turn, the topology of the network has deep implications in the nature and the quality of the interschema properties that can be extracted, as will be clear in Section 10.2. Therefore, it is appropriate that the choice of the arc to be removed is not random and that a clear rule guiding it is defined. The rule defined by our approach is the following: given a pair of arcs between two nodes n_{k_1}, corresponding to the keyword k_1, and n_{k_2}, corresponding to the keyword k_2, with k_1 preceding k_2 in the list of keywords associated with the source, the arc from n_{k_1} to n_{k_2} is maintained and the one from n_{k_2} to n_{k_1} is removed.

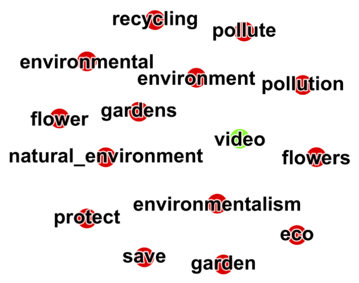
3. A pair of arcs derived from a string similarity links them. As in the previous case, the two arcs coincide and one of them can be removed. The policy adopted to determine the arc of the pair to be removed is the same as the one followed in the previous case.
4. A pair of arcs derived from Phase 4 links them. As in the previous case, the two arcs coincide and one of them can be removed.

Actually, arc labels introduced above are not necessary in our approach for the extraction of semantic relationships described in Section 10.2. However, we have decided to maintain them in our model because we aim at providing an approach to "structuring" unstructured sources that is general and may be adopted in several future applications, some of which could benefit from this information. Moreover, we point out that, in the prototype implementing our approach, in order to increase its efficiency, we directly added only one arc, namely (n_{k_1}, n_{k_2}), during Phases 2, 3 and 4.

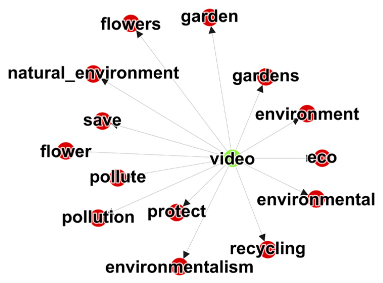
9.3.1 Running example

In this section, we propose an example of how our approach to construct a "structured" representation of an unstructured data source operates. In particular, the unstructured data source into consideration is a video, which talks about environment and pollution. As we said before, for each unstructured source, our approach begins from a list of keywords representing its content. In order to keep our description simple and clear, in this example, we assume that our video has a limited number of keywords, namely the ones shown in Figure 9.2.

(a)



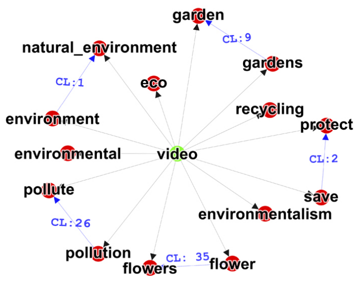
(b)



(c)

	Lemmas
Save	blown_save, bring_through, carry_through, conditional_salvation, deliver, economise, economize, file, hold_open, individual_salvation, kaligtasan, **keep**, keep_open, lay_aside, make_unnecessary, noar, noarus, **preserve**, proprietary_salvation, pull_through, redeem, relieve,rescued, salvage, salvation, sava, save, save_situation, save_up, saved, saving, savior, saviour, savus, sin_and_salvation, spare, special_salvation, svop, szva, tough_save, write
Protect	defend, guard, **keep**, **preserve**, protect, protect_(organisation), protect_(organization), protect_(political_organization), The_national_association_to_protect_children

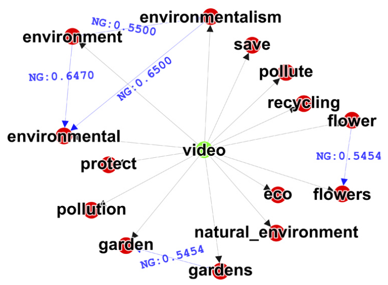
(d)



(e)

k_1	k_2	$kd(k_1, k_2)$
environmentalism	environmental	0.6500
environmental	environment	0.6470
environment	environmentalism	0.5500
flower	flowers	0.5454
gardens	garden	0.5454

(f)



(g)

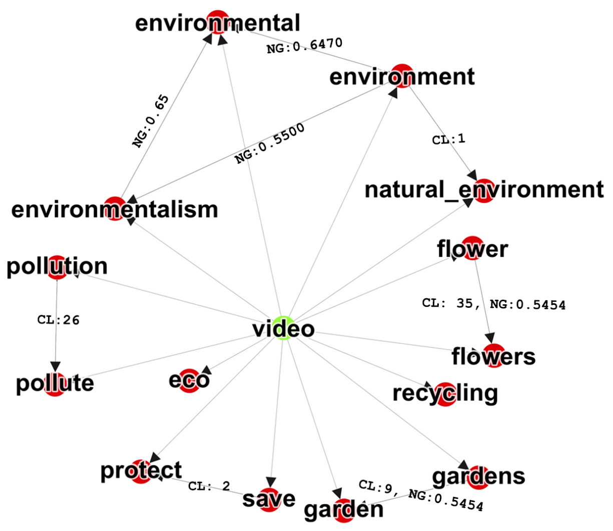


Fig. 9.2: Graphical representation of our approach to derive a “structure” for an unstructured source

Our approach starts with Phase 1. As we can see in Figure 9.2(a), during this phase, it constructs a graph having a node for each keyword. A further node is added to represent the video as a whole; nodes representing keywords are colored in red, whereas the other one is colored in green. Following our strategy, in Figure 9.2(b), we added an arc from the node representing the whole video to each node associated with a keyword.

Now, Phase 2 starts. During this phase, our approach uses a thesaurus. In our example we leveraged BabelNet. In particular, let k_1 and k_2 be two keywords of Figure 9.2(a) having at least one common lemma in BabelNet. An arc is added from the node n_{k_1}, associated with k_1, to the node n_{k_2}, associated with k_2, and vice versa. In Figure 9.2(c), we show two keywords ("Save" and "Protect") and the corresponding lemmas in BabelNet. Common lemmas (i.e., "keep" and "preserve") are in bold. Since "Save" and "Protect" have at least one common lemma, two arcs are added between the corresponding nodes in Figure 9.2(d). These arcs are highlighted in blue in this figure and, due to layout reasons, we report only one arc with two arrows, instead of two arcs with one arrow. Each arc has a label representing the number of common lemmas between the corresponding keywords in BabelNet.

After having examined lexical similarities, Phase 2 terminates and our approach proceeds with Phase 3, which leverages string similarities. In particular, let k_1 and k_2 be two keywords of Figure 9.2(a) having a string similarity degree higher than or equal to $th_k \cdot kd_{max}$ and, at the same time, higher than or equal to th_{kmin}. An arc is added from the node n_{k_1}, corresponding to k_1, to the node n_{k_2}, corresponding to k_2, and vice versa. In Figure 9.2(e), we report the pairs of keywords that satisfy this feature. In Figure 9.2(f), we added a pair of arcs for each pair of keywords of Figure 9.2(e). Here, to better evidence them, we have omitted the arcs constructed during Phase 2. Again, these arcs are highlighted in blue and, due to layout reasons, we report only one arc with two arrows, instead of two arcs with one arrow. Each arc has a label representing the string similarity degree (computed by means of N-Grams) between the corresponding keywords.

Finally, in Figure 9.2(g), Phase 4 of our approach combines the arcs derived in Phases 2 and 3. In particular, it may happen that, for a pair of keywords (see, for instance, the keywords "garden" and "gardens"), two pairs of arcs have been generated, one in Figure 9.2(d) and one in Figure 9.2(f). In this case, in Figure 9.2(g), the two pairs of arcs are substituted by only one pair, representing both of them. The label of this pair reports the label of both the original arcs.

Property	*Precision*	*Recall*	*F-Measure*	*Overall*
Synonymies	0.76	0.82	0.79	0.56
Overlappings	0.69	0.65	0.67	0.36
Type Conflicts	0.72	0.64	0.68	0.39
Homonymies	0.91	0.88	0.89	0.79

Table 9.1: Precision, Recall, F-Measure and Overall of our approach when a cluster-based approach for structuring unstructured sources is applied

9.3.2 Evaluation

In the Introduction, we have seen that an important theoretical property of our approach (that distinguishes it from several possible alternative approaches, like the ones based on ontologies) is its applicability in all possible scenarios because it does not require a support knowledge, except for a (possibly generic) thesauruses like BabelNet. In this section, we want to test its accuracy by comparing it with an alternative approach. For this purpose, we extended to unstructured data the clustering-based family of approaches defined for structured and semi-structured sources (see, for instance [41, 299]).

We performed the extension as follows: we considered the keywords associated with an unstructured sources and used WordNet to derive a semantic distance coefficient for each pair of keywords. Then, we applied a clustering algorithm (specifically, Expectation Maximization [183]) to group keywords into homogeneous clusters. In this way, we obtained a possible structure for unstructured sources. This structure is in line with what has been done in the past for the clustering-based family of approaches, when they were applied on structured and semi-structured sources. This way of proceeding gave us the possibility to still apply the interschema property extraction approach defined in Section 10.2. In this case, we assumed that, given a keyword, the corresponding neighborhood constisted of the other keywords of its clusters.

We performed the same experiment described in Section 10.3.1 on the same sources. The only difference was the substitution of our approach for structuring unstructured sources with the clustering-based family of approaches outlined above. The obtained results are shown in Table 9.1. Clearly, the differences between the performance reported in Tables 10.5 and 9.1 were due exclusively to the merits or demerits of our approach for structuring unstructured sources.

From the analysis of this table we can observe that our approach for structuring unstructured sources presents a better performance than the corresponding cluster-based one described above. The differences are evident even if not extremely marked. For instance, we can observe a gain in Precision (resp., Recall, F-Measure, Overall) ranging from 4% (resp., 4%, 4%, 9%) to 10% (resp., 12%, 10%, 25%).

The results of this experiment, coupled with the theoretical analysis performed in the Introduction and mentioned above, allow us to conclude that our approach for structuring unstructured data is really capable of satisfying the requirements for which it was defined.

9.4 Related work

The representation mechanisms of unstructured sources (basically texts) are mainly based on two strategies, namely analysis of contents and analysis of references [345]. The former infers a representation of a document from the corresponding content, whereas the latter focuses on relationships among documents. Clearly, our interest is mainly on the former strategy, because its objective is similar to the one of our approach.

The most basic approach to representing texts leverages Bags of Words (BOW) [62, 322]. In this case, machine learning approaches are used to identify the set of words that mostly characterize a text. Some more sophisticated strategies are based on the extraction of sentences [152]. In this case, a text is mapped onto semantic spaces, such as WordNet or Wikipedia. Another strategy is Explicit Semantic Analysis (ESA) [156], which mixes BOW and document references. In ESA, the relatedness between documents is computed by extracting similarities between the concepts identified within them, thanks to the cross-references expressed therein.

An important model in the BOW context is word2vec [259, 260]. This model is based on neural networks. It constructs a vector space and associates each word of the text into examination with a vector in this space in such a way that words sharing common contexts have close corresponding vectors in the vector space. The word2vec model has been extended to the doc2vec one [224], which exploits similarities and contextual information of each word to reduce the dimensionality of the vector space. Other approaches reach the same objective (i.e., dimensionality reduction) by means of Latent Semantic Analysis [212], which exploits matrix decomposition methods.

Word-based methods are currently flanked by concept-based ones. As an example, [316, 315] introduce the idea of Bag of Concepts, in place of Bag of Words. In this approach, concepts are generated by disregarding semantic similarities between words. Semantic similarities have been considered only recently [213].

Another relevant set of approaches use ontologies or, in general, external sources of semantics, to generate conceptual representations of documents by matching document terms with ontology concepts (see, for instance, [79, 198, 365, 38]). The performance of these approaches is strongly related to the quality of the adopted external sources. As a consequence, in these approaches, very specific domains can strongly benefit from the availability of dedicated ontologies.

An attempt to define a "structure" for an unstructured source can be found in [250]. This approach generates a rowset with n attributes, i.e. a tabular representation from unstructured data. A single rowset is a set of tuples and is equivalent to a relation in relational databases; logical associations may exist between rowsets, but these are not explicitly defined. The schema of a rowset may be defined on read. Transformation functions, possibly based on fuzzy logic, are used to properly read the complex unstructured data and map them on the rowset schema. These functions are also exploited to address the data variety issue, by means of an interface for the dataset extraction, which is unified and valid for all the sources. Different transformation functions can be used to map different unstructured data onto the same schema. The content of a rowset depends on the membership function associated with a fuzzy logic and on the possible constraints regarding it. However, data extraction is only one of the steps defined in [250], which develops a general data processing system based on an Extract, Process, and Store (EPS) paradigm. From the above description, it appears evident that the approach of [250] shares several features with ours; in particular, the purpose of structuring unstructured data is common to both of them. However, the two approaches also present several differences. Indeed, for the structuring task, the approach of [250] strongly depends on user defined transformation functions and on rowset schemas (which are not automatically inferred by the sources). Now, the definition of both the functions and the schema may be difficult on complex sources. Furthermore, mapping more sources on the same schema requires a manual integration step, which, again, may be a difficult task when the number of involved sources is high. On the other hand, querying obtained data sources is particularly effective with the use of fuzzy techniques and declarative U-SQL query language characterizing the approach of [250]. On the contrary, in our proposal, in order to perform the structuring of unstructured sources, we leverage network analysis, as well as lexical and string similarities, in order to automatically derive a general and uniform schema for different unstructured sources. In fact, as we will see in the following, unstructured sources are "structured" by first representing them as a network, starting from a set of keywords associated with them; then, this structure is enriched thanks to the addition of arcs that link nodes presenting lexical or string similarities even if they belong to different sources. As a consequence, it is possible to state that the approach presented in [250] is more effective and flexible in querying data lakes contents, but it requires a more complex design phase with a heavy human intervention, difficult to sustain in presence of numerous data sources. On the contrary, our approach simplifies the structuring phase, because it does not need a preliminary structure to use as model and it does not require a human intervention. On the other side, its querying capabilities are limited to the summarization of unstructured sources provided by the keywords representing them. Therefore, in a certain sense, our approach and the one of [250] can be considered orthogonal.

10

A new approach to extracting inter-schema properties from heterogeneous sources

The knowledge of interschema properties (e.g., synonymies, homonymies, hyponymies, subschema similarities) plays a key role for allowing decision making in situations where information sources are heterogeneous. This is the case of smart cities, where sensors, smart objects, social communications and internet of things contribute to generate and collect different data. In the past, a large amount and variety of approaches to deriving interschema properties from structured and unstructured data have been proposed. However, currently, it is esteemed that more than 80% of data sources are unstructured. Furthermore, nowadays the number of sources generally involved in an interaction is much higher than in the past. As a consequence, the necessity arises of new approaches to addressing the interschema property derivation issue in this new scenario. In this study, we aim at providing a contribution in this setting by proposing an approach capable of uniformly extracting interschema properties from a huge number of structured, semi-structured and unstructured sources.

10.1 Introduction

In the last few years, we are assisting to a real revolution in the information system scenario. In fact, the number and the size of available data sources have dramatically increased. Furthermore, most of them (i.e., more than 80%) are unstructured [117, 105]. These facts are rapidly changing the research and technological "coordinates" of the information system research field [75]. As a consequence of this phenomenon, even issues successfully addressed in the past must be re-considered and re-investigated. One of these issues is certainly the derivation of inter-schema properties (i.e., *intensional* relationships between concepts represented in different data sources [284], like synonymies, homonymies, hyponymies, overlappings, subsechema similarities). This issue has been largely studied in the past [303, 76]; however, the proposed approaches generally considered structured or, at most, semi-structured sources. Furthermore, the number of involved sources, for which most of past approaches were targeted, was very small if compared with a typical current source interaction and cooperation scenario.

In this study, we propose a novel approach to uniformly performing the extraction of inter-schema properties from structured, semi-structured and unstructured sources. Our approach has been specifically conceived having in mind two peculiarities that should have characterized it, namely: *(i)* the capability of handling unstructured sources; *(ii)* the lightweightness, making it capable of managing a huge number of data sources. We have a more detailed look to these two specificities.

As for the capability of handling unstructured sources, our approach is based on the proposal presented above. In addition, we exploit the network model we gave to data lakes, on which this whole idea is based.

Our proposal also differs from the other ones presented in related research fields in the past and that could be extended to solve this problem. Think, for instance, of ontologies. We could link each available keyword to an ontology and use this last as the "infrastructure" through which establishing the relationships among the keywords once they have been linked to it. This approach is certainly valid. However, it needs a support ontology. As a consequence, it can be employed only in those application fields for which an ontology exists and only if all the involved information sources can be mapped in a unique ontology. If only some of the involved unstructured sources can be referred to an ontology and/or some of them can be mapped into another ontology and/or, finally, some of them cannot be referred to any ontology, this way of proceeding cannot be adopted. From this point of view our approach is more general because it can be applied in all cases, independently of the presence of none, one or more ontologies, which the unstructured sources can be referred to. It only needs a thesaurus. If there exists one specific for the scenario which the unstructured sources into examination belong to, it uses this theasurus. Otherwise, it can still work with a general-purpose thesaurus, like BabelNet [271]. Clearly, if the unstructured sources are specific of a certain field, the availability of a specific thesaurus can help to obtain a better accuracy. However, if this kind of thesaurus is not available, a general-purpose one is sufficient to proceed even if, in this case, accuracy will be lower.

As for the lightweightness of our approach, we observe that, in a big data scenario like the one currently characterizing the information system field, a new proposed approach must take scalability into a primary consideration [233, 229]. As a matter of fact, the sources interacting in every task are always very numerous and large (think, for instance, of a data lake constructed to support data analytics in an organization) and the time allowed for each transaction is very limited (think, for instance, of streaming applications). As a consequence, even approaches considered very scalable in the past (such as DIKE [286], MOMIS [73], and Cupid [245]) are not adequate anymore. In our opinion, the tests performed to evaluate our approach and described in Section 10.3, confirm that our approach is really capable of satisfying the lightweightness requirement without sacrificing, if not in a very small extent) the result accuracy.

Summarizing, the main contribution of this proposal is about providing an overall procedure capable of extracting inter-schema properties from structured, semi-structured and unstructured sources. Our procedure is lightweight because it has been specifically conceived to operate with big data in the smart city domain. We determine its computational complexity and compare it with the one of similar approaches conceived to work on smaller (only) structured and semi-structured data sources. In spite of its lightweightness, the accuracy of our procedure is very satisfying, as witnessed by the quantitative evaluations presented. An important component of our approach, which could also be used separately from the overall procedure in other contexts, is the technique for "structuring" unstructured sources whose peculiarities, which distinguish it from some related approaches proposed in the past, have been described above.

10.2 Extracting inter-schema properties from heterogeneous sources

We are now ready to illustrate our strategy for uniformly extracting inter-schema properties from structured, semi-structured and unstructured sources. Here, we assume that the content of the sources of interest is represented by means of the model described in Section 9.2, and that our approach to "structuring" unstructured sources, has been already applied on all unstructured sources.

Before delving into a detailed description of our approach, a discussion about the role played by source metadata and about the consequences of this role is in order. Indeed, as previously pointed out, our approach assumes that some metadata are available for each structured, semi-structured and unstructured source. This assumption is important both our approach for structuring unstructured sources and the one for extracting inter-schema properties use these metadata. It is, then, of outmost importance to analyze the possible issues (and the corresponding solutions) in obtaining good quality metadata, when they are not directly provided with the sources, and the impact that they have on the results returned by our approach.

We recall that, in the current big data scenario, any inter-schema property extraction strategy must be lightweight. For this reason, in our effort to define a new approach for this task, we avoided highly complex choices, such as the fixpoint computation, characterizing DIKE [286, 285] and XIKE [129], or the clustering-based computation, characterizing MOMIS [74], or, again, the wide range of parameter computation, characterizing Cupid [245]. These choices, as well as most of the other ones present in the past approaches proposed for reconciling and integrating structured and semi-structured data sources (e.g., to construct a data warehouse) [303, 76], would certainly return very accurate results. However, their speed is incompatible with the one required in many current applications, which must allow the derivation of semantic relationships "on-the-fly" from a very high number of data sources, most of which are unstructured, i.e., in a format not considered by classic approaches. So, our strategy must necessarily privilege quickness over accuracy even if, clearly, accuracy must be high. In Section 10.3, we will see if, and how, this issue has been addressed.

Our strategy consists of two phases; the former computes the semantic similarity degree of each pair of objects stored in the metadata of the involved sources. The latter derives semantic relationships between the same objects starting from the results returned by the former.

10.2.1 Semantic similarity degree computation

Our approach to semantic similarity degree computation consists of three steps, namely:

1. *basic similarity computation*;
2. *standard similarity computation*;
3. *refined similarity computation*;

In the next subsections, we illustrate these three steps in detail.

Basic similarity computation

Basic similarities consider only lexicon (determined with the support of suitable thesauruses, such as BabelNet [271] and WordNet [262], and string similarity metrics, such as N-Grams [215]), and object types.

Let D_1 and D_2 be two sources, let $\mathcal{M}_1$ and $\mathcal{M}_2$ be the corresponding metadata, let $x_1 \in Obj_1$ and $x_2 \in Obj_2$ be two objects belonging to $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The basic similarity degree $bs(x_1, x_2)$ between x_1 and x_2 can be computed as:

$$bs(x_1, x_2) = \omega \cdot \sigma_L(x_1, x_2) + (1 - \omega) \cdot \sigma_T(x_1, x_2)$$

In other words, the basic similarity degree between x_1 and x_2 can be computed as a weighted mean of two components. The former, σ_L, returns their lexical similarity, whereas the latter, σ_T, specifies the similarity of their types. ω is a weight belonging to the real interval $[0,1]$ and used to tune the importance of σ_L w.r.t. σ_T. We have experimentally set ω to 0.90.

σ_L can be directly detected from a thesaurus. In our experiments, we used WordNet in the first beat, because it provides the similarity degree between the two objects, and BabelNet, when WordNet did not provide any result. Since this last thesaurus does not return the similarity degree of two objects that it considers similar, we coupled BabelNet with a suitable string similarity metric (in particular, N-Grams). This last is applied to the objects and the corresponding lemmas returned by BabelNet; obtained results are then combined to compute the lacking similarity degree. Furthermore, in very specific application contexts, specialized thesauruses could be used.

σ_T is defined as follows:

$$\sigma_T = \begin{cases} 1 & \textit{if } (x_1 \in Cmp_1 \textit{ and } x_2 \in Cmp_2) \textit{ or } (x_1 \in Smp_1 \textit{ and } x_2 \in Smp_2) \textit{ or} \\ & \quad (x_1 \in Att_1 \textit{ and } x_2 \in Att_2) \\ 0.5 & \textit{if } (x_1 \in Cmp_1 \textit{ and } x_2 \in Smp_2) \textit{ or } (x_1 \in Smp_1 \textit{ and } x_2 \in Cmp_2) \textit{ or} \\ & \quad (x_1 \in Smp_1 \textit{ and } x_2 \in Att_2) \textit{ or } (x_1 \in Att_1 \textit{ and } x_2 \in Smp_2) \\ 0 & \textit{otherwise} \end{cases}$$

Standard similarity computation

Standard similarities take both basic similarities and the neighbors of the involved objects into account.

Let D_k be a source of the set DS of the sources of interest, let $\mathcal{M}_k = \langle N_k, A_k \rangle$ be the corresponding set of metadata, let Obj_k be the set of the objects of $\mathcal{M}_k$. The set $nbh(x)$ of the neighbors of an object $x \in Obj_k$ is defined as:

$$nbh(x) = \left\{ y | y \in Obj_k, (n_x, n_y) \in A_k \right\}$$

Let D_1 and D_2 be two sources, let $\mathcal{M}_1$ and $\mathcal{M}_2$ be the corresponding sets of metadata, let $x_1 \in Obj_1$ and $x_2 \in Obj_2$ be two objects belonging to $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The standard similarity degree $ss(x_1, x_2)$ between x_1 and x_2 can be computed as follows:

- If both $nbh(x_1) = \emptyset$ and $nbh(x_2) = \emptyset$[1], then $ss(x_1, x_2) = bs(x_1, x_2)$.
- If either $nbh(x_1) = \emptyset$ and $nbh(x_2) \neq \emptyset$ or $nbh(x_2) = \emptyset$ and $nbh(x_1) \neq \emptyset$, then $ss(x_1, x_2) = f_p \cdot bs(x_1, x_2)$. Here, f_p is a factor, whose possible values belong to the real interval $[0,1]$, which "penalizes" the value obtained for basic similarities. Indeed, these are the only similarities that we can compute and, therefore,

[1] For instance, this happens when both x_1 and x_2 are attributes; indeed, the nodes corresponding to attributes do not have outgoing arcs.

we must base our standard similarity computation on them. However, we must consider that, differently from the previous case, the sets of neighbors of x_1 and x_2 have different features (because one of them is empty and the other one is not empty), and this fact must be taken into account. We have experimentally set $f_p = 0.85$.

- In all the other cases, i.e., if $x_1 \in (Smp_1 \cup Cmp_1)$ and $x_2 \in (Smp_2 \cup Cmp_2)$, then $ss(x_1, x_2)$ can be computed as follows:
  1. $nbh(x_1)$ and $nbh(x_2)$ are computed.
  2. A bipartite graph, whose nodes are the ones of $nbh(x_1)$ and $nbh(x_2)$, is constructed.
  3. For each pair (p,q), such that $p \in nbh(x_1)$ and $q \in nbh(x_2)$, an arc is added in the bipartite graph; the weight of this arc is set to $bs(p,q)$.
  4. The maximum weight matching is computed on this bipartite graph. Let A_M be the set of the returned arcs. Then:

$$ss(x_1,x_2) = \begin{cases} \frac{2 \cdot \sum_{(p,q) \in A_M} bs(p,q)}{|nbh(x_1)|+|nbh(x_2)|} & \textit{if neither } D_1 \textit{ nor } D_2 \textit{ are unstructured} \\ \\ \frac{2 \cdot \sum_{(p,q) \in A_M} bs(p,q)}{2 \cdot min(|nbh(x_1)|,|nbh(x_2)|)} & \textit{otherwise} \end{cases}$$

  In this formula, if neither D_1 nor D_2 are unstructured, $ss(x_1, x_2)$ returns the value of an objective function that takes into account how many nodes of $nbh(x_1)$ and $nbh(x_2)$ are linked by basic similarity relationships and how strong these relationships are. Furthermore, the objective function penalizes the presence of dangling nodes, i.e., nodes of $nbh(x_1)$ or $nbh(x_2)$ that do not participate to the maximum weight matching.

  If D_1 and/or D_2 are unstructured, then it is necessary to consider that, even if our approach performed a “structuring” task, its final result is limited if compared with the rich structure characterizing the other kinds of source. As a consequence, the sets of neighbours of the nodes belonging to unstructured sources are generally much smaller than the ones characterizing the other kinds of source. Therefore, in this case, using the same objective function adopted when neither D_1 nor D_2 are unstructured would not take this important feature into account, and the overall result would be biased. To address this issue, if D_1 and/or D_2 are unstructured, in the denominator of $ss(x_1, x_2)$ we consider the minimum size between $|nbh(x_1)|$ and $|nbh(x_2)|$, clearly multiplied by 2 to indicate the maximum number of nodes that could be linked by a similarity relationship in this situation.

Refined similarity computation

Refined similarities are based on standard similarities (for simple and complex objects), basic similarities (for attributes) and object neighbors.

Let D_1 and D_2 be two sources, let $\mathcal{M}_1$ and $\mathcal{M}_2$ be the corresponding sets of metadata, let $x_1 \in Obj_1$ and $x_2 \in Obj_2$ be two objects belonging to $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The refined similarity degree $rs(x_1, x_2)$ between x_1 and x_2 can be computed as follows:

- If $nbh(x_1) = \emptyset$ and/or $nbh(x_2) = \emptyset$, then $rs(x_1, x_2) = ss(x_1, x_2)$.
- Otherwise, if $x_1 \in (Smp_1 \cup Cmp_1)$ and $x_2 \in (Smp_2 \cup Cmp_2)$, then $rs(x_1, x_2)$ is obtained by applying the same four steps described for $ss(x_1, x_2)$ with the only difference that, in Step 3, the weight of the arc (p, q), such that $p \in nbh(x_1)$ and $q \in nbh(x_2)$, is set to $ss(p, q)$, and no more to $bs(p, q)$. In other words, while standard similarity computation leverages basic similarities, refined similarity computation is based on standard similarities.

Clearly, from a theoretical point of view, it would be possible to perform other refinement steps. In this case, at the i^{th} refinement step, similarities would be computed starting from the ones obtained at the $(i-1)^{th}$ step, by setting these last ones as the weights of the arcs of the bipartite graph. However, the advantages in accuracy that these further refinement steps could produce do not justify the computational costs introduced by them (see Section 10.3), especially in an agile and lightweight context, such as the one characterizing the big data scenario.

10.2.2 Semantic relationship detection

The derivation of semantic relationships among the objects of the sources of DS represents the second phase of our strategy. It takes the refined semantic similarities among the objects of DS as input. The semantic relationships that it can return are the following:

- *Synonymies*: A synonymy between two objects $x_1 \in Obj_1$ and $x_2 \in Obj_2$ exists if they have a high similarity degree, the same type (i.e., both of them are complex objects or simple objects or attributes) and (possibly) different names.
- *Type Conflicts*: A type conflict between two objects $x_1 \in Obj_1$ and $x_2 \in Obj_2$ exists if they have a high similarity degree but different types.
- *Overlappings*: An overlapping exists between two objects $x_1 \in Obj_1$ and $x_2 \in Obj_2$ if they have (possibly) different names, the same type and an intermediate similarity degree, in such a way that they can be considered neither synonymous nor distinct.

- *Homonymies*: A homonymy between two objects $x_1 \in Obj_1$ and $x_2 \in Obj_2$ exists if they have the same name and the same type but a low similarity degree.

Let D_1 and D_2 be two sources, let $\mathcal{M}_1$ and $\mathcal{M}_2$ be the corresponding sets of metadata, let $x_1 \in Obj_1$ and $x_2 \in Obj_2$ be two objects belonging to $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. Finally, let $RefSim_{12}$ be the set of refined similarities involving the objects of Obj_1 and Obj_2.

First, our approach computes the maximum refined similarity degree rs_{max} present in $RefSim_{12}$. Then, it examines each similarity $\langle x_1, x_2, rs(x_1, x_2)\rangle$ registered in $RefSim_{12}$ and verifies if a semantic relationship exists between the corresponding objects as follows:

- If $\left(rs(x_1, x_2) \geq th_{Syn} \cdot rs_{max}\right)$ and $(rs(x_1, x_2) \geq th_{min})$, which implies that the refined similarity degree between x_1 and x_2 is among the highest ones in $RefSim_{12}$ and, in any case, higher than or equal to a minimum threshold, then:
  - if x_1 and x_2 have the same type, it is possible to conclude that a synonymy exists between them;
  - if x_1 and x_2 have different types, it is possible to conclude that a type conflict exists between them.
- If $\left(rs(x_1, x_2) < th_{Syn} \cdot rs_{max}\right)$ and $(rs(x_1, x_2) \geq th_{Ov} \cdot rs_{max})$ and $(rs(x_1, x_2) \geq th_{min})$, which implies that the refined similarity degree between x_1 and x_2 is higher than or equal to a minimum threshold, it is not among the highest ones in $RefSim_{12}$, but it is significant, then:
  - if x_1 and x_2 have the same type, it is possible to conclude that an overlapping exists between them.
- If $(rs(x_1, x_2) < th_{Hom} \cdot rs_{max})$ and $(rs(x_1, x_2) < th_{max})$, which implies that the refined similarity degree between x_1 and x_2 is among the lowest ones in $RefSim_{12}$ and, in any case, lower than a maximum threshold, then:
  - if x_1 and x_2 have the same name and the same type, it is possible to conclude that a homonymy exists between them.

Here, th_{Syn}, th_{min}, th_{Ov}, th_{Hom} and th_{max} have been experimentally set to 0.85, 0.50, 0.65, 0.25 and 0.15, respectively.

As pointed out in the Introduction, the knowledge of inter-schema properties is very relevant for several applications, for instance source integration, source querying, data warehouse and/or data lake construction, data analytics, and so forth. For instance, as far as integration is concerned:

- If a synonymy exists between $x_1 \in Obj_1$ and $x_2 \in Obj_2$, then x_1 and x_2 must be merged in a unique object, when the integrated schema is constructed.

- If a homonymy exists between x_1 and x_2, then it is necessary to change the name of x_1 and/or x_2, when the integrated schema is constructed.
- If an overlapping exists between x_1 an x_2, then it is necessary to restructure the corresponding portion of network. Specifically, a node x_{12}, representing the "common part" of x_1 and x_2, is added to the network. Furthermore, each pair of arcs (x_1, x_T) and (x_2, x_T), starting from x_1 and x_2 and having the same target x_T, is substituted by a unique arc (x_{12}, x_T). Finally, an arc from x_1 to x_{12} and another arc from x_2 to x_{12} are added to the network.
- If a type conflict exists between x_1 and x_2, then it is necessary to change the type of x_1 and/or x_2 in such a way as to transform the type conflict into a synonymy. Then, it is necessary to handle this last relationship by applying the corresponding integration rule seen above.

The way of proceeding described above can be extended to the detection of hyponymies. In particular, the extension already proposed in [283] for structured and semi-structured data can be probably adopted to this scenario. We plan to investigate this issue in the future. Finally, an analogous way of proceeding can be performed when querying or other activities must be carried out on a set of sources of interest.

An example case

In this section, we provide an example of the behaviour of our approach to the extraction of semantic relationships. To fully illustrate its potentialities, we derive these relationships between objects belonging to an unstructured source and a semi-structured one.

In particular, the unstructured source taken into consideration is a video from $YouTube$. The corresponding keywords are reported in Table 10.1. Its "structured" representation, in our network-based model, obtained after the application of the approach described in Chapter 9, is reported in Figure 10.1. The semi-structured source is a JSON file whose structure is shown in Figure 10.2. Its representation in our network-based model is reported in Figure 10.3.

By applying the first phase of our approach we obtained the refined semantic similarity degrees between all the possible pairs of nodes (n_U, n_S), such that n_U belongs to the unstructured source and n_S belongs to the semi-structured one. To give an idea of these similarity degrees, in Figure 10.4, we report their distribution in a semi-logarithmic scale. From the analysis of this figure, we can observe that a very few number of pairs have a significant similarity degree, which could make them eligible to be selected for synonymies, type conflicts and overlappings. At a first glance,

Keywords
video, reuse, flower, easy, tips, plastic, simple, environment, pollution, garbage, wave, o3, reduce, pollute, help, natural_environment, educational, green, environment_awareness, bike, life, environmentalism, planet, earth, climate, clime, save, nature, environmental, gardens, power, recycling, garden, protect, flowers, eco, fine_particle, recycle, atmospheric_condition, ocean, metropolis, weather, spot, waving, aurora

Table 10.1: Keywords of the unstructured source of our interest

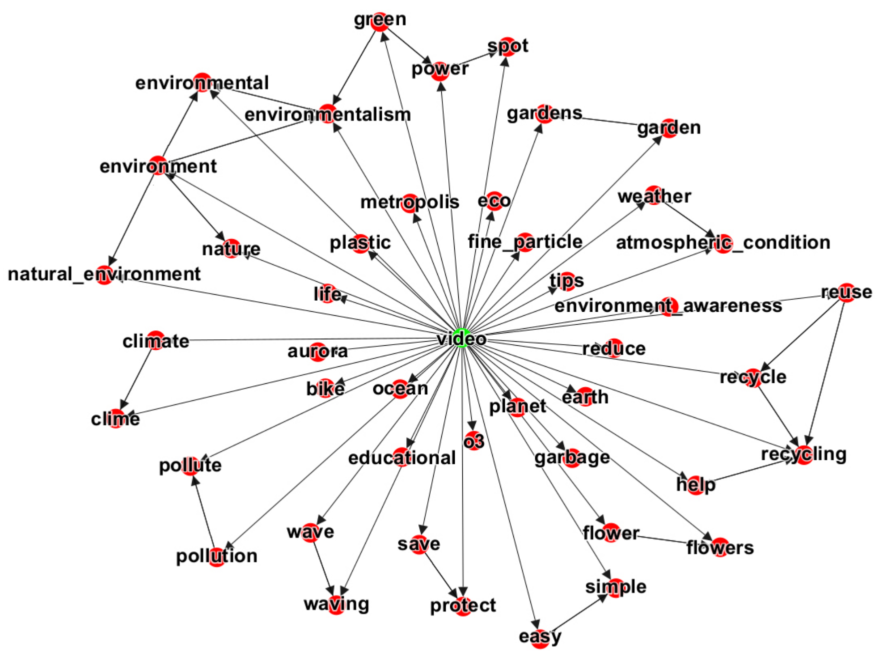


Fig. 10.1: Representation, in our network-based model, of the unstructured source of our interest

this trend appears correct and intuitive, even if this conclusion must be confirmed or rejected by a much deeper analysis (see below).

At this point, by applying the second phase of our approach, we obtained the synonymies, the type conflicts and the overlappings reported in Tables 10.2 - 10.4. Instead, as for this pair of sources, we found no homonymies.

Semi-Structured Source Node	*Unstructured Source Node*
climate	*climate*
climate	*clime*

Table 10.2: Derived Synonymies

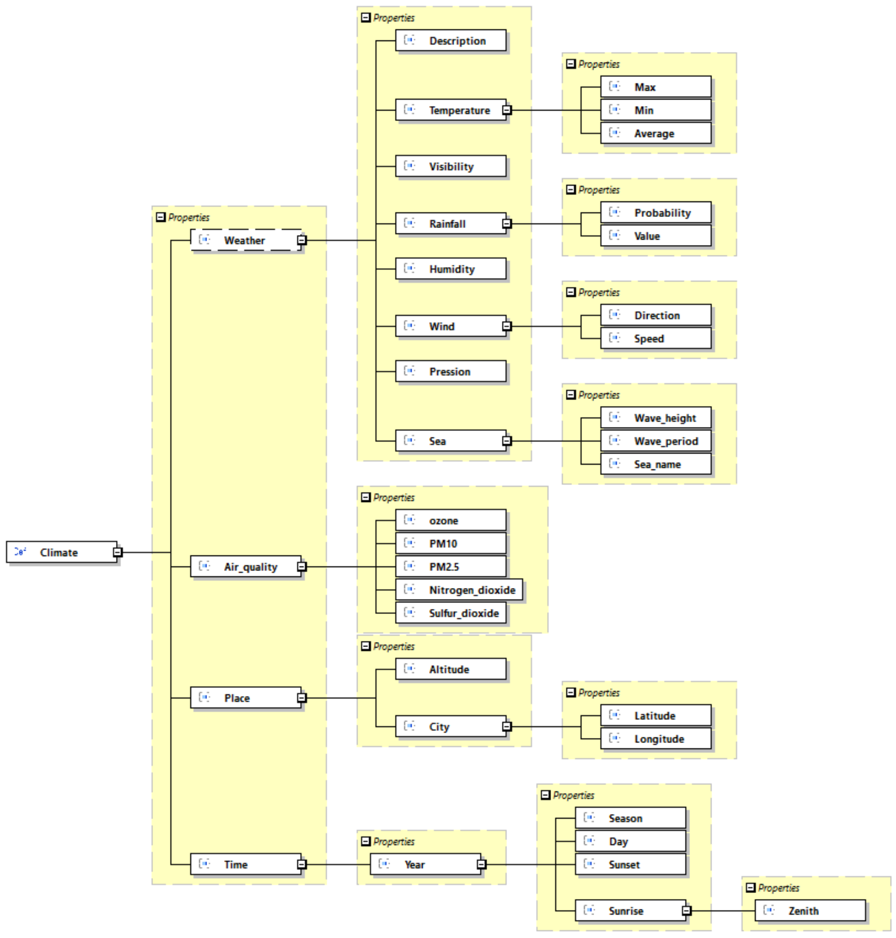


Fig. 10.2: Structure of the JSON file associated with the semi-structured source of our interest

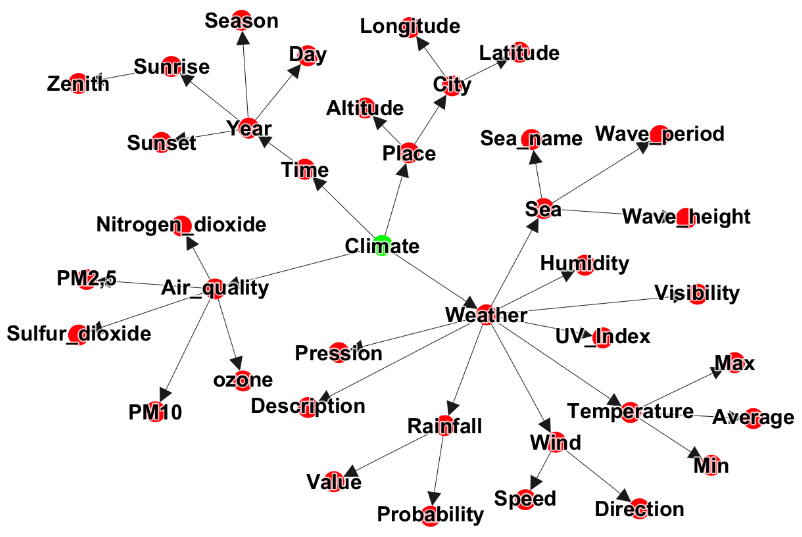


Fig. 10.3: Representation, in our network-based model, of the semi-structured source of our interest

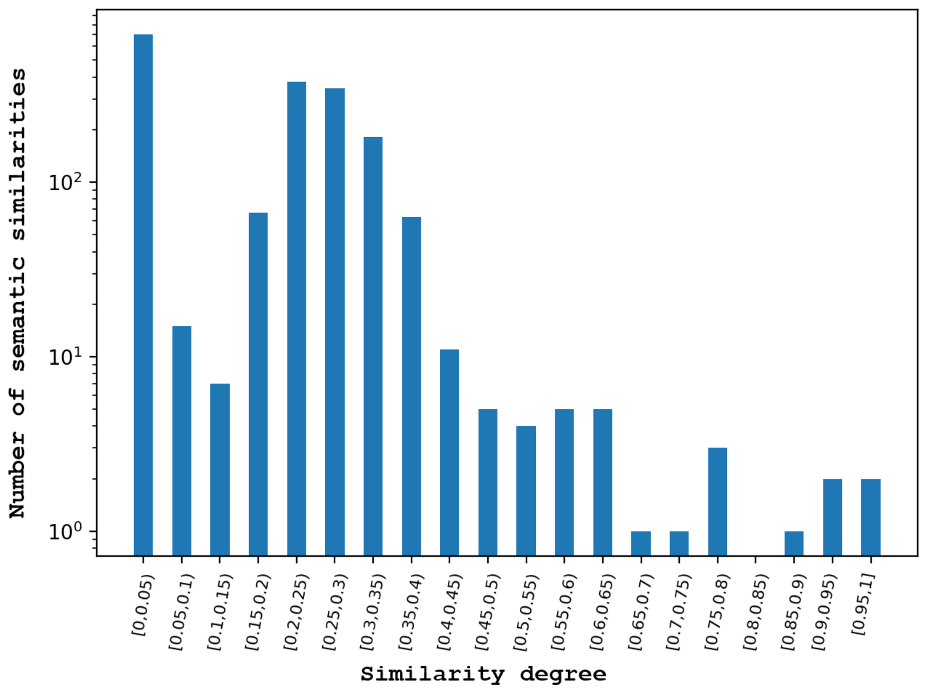


Fig. 10.4: Distribution, in a semi-logarithmic scale, of the values of the the semantic similarity degrees of the objects belonging to the two sources of interest

Semi-Structured Source Node	*Unstructured Source Node*
*pm*10	*fine_particle*
ozone	*o*3

Table 10.3: Derived Type Conflicts

Semi-Structured Source Node	*Unstructured Source Node*
sea	*ocean*
city	*metropolis*
sunrise	*aurora*
place	*spot*
wind	*tips*
sulfur_dioxide	*garbage*
weather	*clime*

Table 10.4: Derived Overlappings

We asked a human expert to validate these results. At the end of this task, he reported the following considerations:

- The synonymies provided by our approach are correct. No further synonymy can be manually found in the two considered sources.
- The type conflicts provided by our approach are correct. No further type conflict can be manually found in the two sources.
- The overlappings provided by our approach are correct, except for the one linking "wind" and "tips", which actually represent two different concepts. A very interesting overlapping found by our approach is the one between "sulfur_dioxide" and "garbage", in that, even if they represent two seemingly different concepts, both of them denote harmful substances. Some further overlappings could be manually found in the two sources into consideration (for instance, the one between "climate" and "environment"), even if they are semantically weak, and considering them as overlappings or as distinct concepts is subjective.

10.3 Experiments

Our test campaign had two main purposes, namely: *(i)* evaluating the performance of our inter-schema property derivation approach when applied to the scenario for which it was thought, and *(ii)* evaluating the pros and the cons of this approach w.r.t. approaches thought for structured and semi-structured sources of Data Warehouses. We describe these two experiments in the next subsections.

10.3.1 Overall performances

To perform our experiments, we constructed a set DS of data sources consisting of 2 structured sources, 4 semi-structured sources (2 of which were XML sources and 2 were JSON ones), and 4 unstructured sources (2 of which were books and 2 were videos). All these sources stored data about environment and pollution. To describe unstructured sources, we considered a list of keywords for each of them. These keywords were derived from Google Books, for books, and from YouTube, for videos.

It could appear that taking only 10 sources is somehow limited. However, we carried out this choice because we wanted to fully analyse the behaviour and the performance of our approach and, as it will be clear below, this requires the human intervention for verifying obtained results. This intervention would have become much more difficult with a higher number of sources to examine. At the same time, our test set is fully scalable.

For our experiments, we used a server equipped with an Intel I7 Dual Core 5500U processor and 16 GB of RAM with the Ubuntu 16.04.3 operating system. Clearly, the capabilities of this server were limited. However, they were adequate for the (small) data set DS we have chosen to use in our tests.

As the first task of our experiment, we represented the metadata of all the sources by means of the data model described in Section 9.2. Then, we applied the approach described in Chapter 9 to (at least partially) "structure" the unstructured sources of our test data set. Finally, we extracted semantic relationships existing between all the possible pairs of our test sources. After this, we asked the human expert to examine all the possible pairs of our test sources and to indicate us the semantic relationships that, in his opinion, existed therein.

At this point, we were able to evaluate the correctness and the completeness of our approach by using the classical measures adopted in the literature for this purpose, i.e., Precision, Recall, F-Measure and Overall [359].

Precision is a measure of correctness. It is defined as:

$$Precision = \frac{|TP|}{|TP|+|FP|}$$

where TP are the true positives (i.e., semantic relationships detected by our approach and confirmed by the human expert), whereas FP are the false positives (i.e., semantic relationships proposed by our approach but not confirmed by our expert).

Recall is a measure of completeness. It is defined as:

$$Recall = \frac{|TP|}{|TP|+|FN|}$$

where FN are the false negatives (i.e., semantic relationships detected by the human expert that our system was unable to find).

F-Measure is the harmonic mean of Precision and Recall. It is defined as:

$$F\text{-}Measure = 2 \cdot \frac{Pecision \cdot Recall}{Precision + Recall}$$

Overall measures the post-match effort needed for adding false negatives and removing false positives from the set of matchings returned by the system to evaluate. It is defined as:

$$Overall = Recall \cdot (2 - \frac{1}{Precision})$$

Precision, Recall and F-Measure fall within the interval $[0,1]$, whereas Overall ranges between $-\infty$ and 1; the higher Precision, Recall, F-Measure and Overall, the better the performance of the evaluated approach.

In Table 10.5, we report obtained results. From the analysis of this table, we can observe that, although our approach has been designed with the intent of privileging

Property	*Precision*	*Recall*	*F-Measure*	*Overall*
Synonymies	0.82	0.87	0.84	0.68
Overlappings	0.77	0.69	0.73	0.48
Type Conflicts	0.78	0.73	0.75	0.52
Homonymies	0.95	0.92	0.93	0.87

Table 10.5: Precision, Recall, F-Measure and Overall of our approach

quickness and lightweightness over accuracy, for the reasons explained in the Introduction, its performance, in terms of correctness and completeness, is extremely satisfying.

We also point out that the values reported in Table 10.5 are those obtained by applying the threshold values reported in Section 10.2. These are the ones guaranteeing the best tradeoff between Precision and Recall and, consequently, the best values of F-Measure and Overall.

Interestingly, if, in a given application context, a user must privilege correctness (resp., completeness) over completeness (resp., correctness), it is sufficient to increase (resp., decrease) the values of th_{min} and to decrease (resp., increase) the values of th_{Ov} and th_{max}.

10.3.2 Evaluation of the pros and the cons

In order to provide a quantitative evaluation of the pros and the cons of our approach to extracting inter-schema properties w.r.t. the past ones thought for structured and semi-structured sources[2] [303, 76], we compared our approach with XIKE [129]. Indeed, in [129], XIKE was already compared with several other systems having the same purposes (namely, Autoplex, COMA, Cupid, LSD, GLUE, SemInt, Similarity Flooding) and it was shown that it obtained comparable or better results.

First, we evaluated Precision, Recall, F-Measure and Overall of our approach and XIKE. Clearly, since this last system (as well as all the other ones mentioned above) did not handle unstructured data sources, we had to limit ourselves to consider only structured or semi-structured sources. Furthermore, as performed in [129], we limited our attention to synonymies and homonymies.

In a first experiment, we considered the same sources exploited in [129] for evaluating the performance of XIKE. In particular, we considered sources relative to Biomedical Data, Project Management, Property Register, Industrial Compa-

[2] Actually, to the best of our knowledge, no approach to uniformly extracting inter-schema properties from structured, semi-structured and unstructured data sources has been proposed in the past.

Application context	*Number of Schemas*	*Max depth*	*Average Number of nodes*	*Average Number of complex elements*
Biomedical Data	11	8	26	8
Project Management	3	4	40	7
Property Register	2	4	70	14
Industrial Companies	5	4	28	8
Universities	5	5	17	4
Airlines	2	4	13	4
Biological Data	5	8	327	60
Scientific Publications	2	6	18	9

Table 10.6: Characteristics of the sources exploited for evaluating our approach

nies, Universities, Airlines, Biological Data and Scientific Publications. Biomedical Schemas have been derived from various sites; among them we cite `http://www.biomsediator.org`. Project Management, Property Register and Industrial Companies Schemas have been derived from Italian Central Governmental Office (ICGO) sources and are shown at the address `http://www.mat.unical.it/terracina/tests.html`. Universities Schemas have been downloaded from the address `http://anhai.cs.uiuc.edu/archive/domains/courses.html`. Airlines Schemas have been found in [289]; Biological Schemas have been downloaded from the addresses `http://smi-web.stanford.edu/projects/helix/pubs/ismb02/schemas/`, `http://www.cs.toronto.edu/db/clio/data/GeneX_RDB-s.xsd` and `http://www.genome.ad.jp/kegg/soap/v3.0/KEGG.wsdl`. Finally, Scientific Publications Schemas have been supplied by the authors of [225].

We considered 35 sources whose characteristics are reported in Table 10.6. The minimum, the maximum and the average number of concepts of our sources were 12, 829 and 79, respectively.

The number of synonymies (resp., homonymies) really present in these sources was 498 (resp, 66). The number of synonymies (resp., homonymies) returned by XIKE was 541 (resp, 76). Finally, the number of synonymies (resp., homonymies) returned by our system was 593 (resp., 84). By comparing real synonymies and homonymies with the ones returned by XIKE and our approach we computed Precision, Recall, F-Measure and Overall for these two systems. They are reported in Table 10.7.

From the analysis of this table we can observe that Precision, Recall, F-Measure and Overall are better in XIKE even if those of our approach are satisfying. This was expected because it has been designed to be lightweight and, therefore, it introduces some approximations. For instance, while XIKE considers the neighbors of all levels in the computation of the similarity degree of two objects, our approach considers only the neighbors of levels 1 and 2.

System	*Precision*	*Recall*	*F-Measure*	*Overall*
XIKE (Synonymies)	0.92	0.90	0.91	0.82
XIKE (Homonymies)	0.87	0.95	0.91	0.81
Our approach (Synonymies)	0.84	0.87	0.85	0.70
Our approach (Homonymies)	0.79	0.92	0.85	0.68

Table 10.7: Precision, Recall, F-Measure and Overall of XIKE and our approach

Until now, our experimental campaign highlighted the cons of our approach. To evidence and quantify the pros, we measured its response time and the one of XIKE when the number of involved concepts represented in the corresponding metadata to examine increases. Obtained results are reported in Figure 10.5.

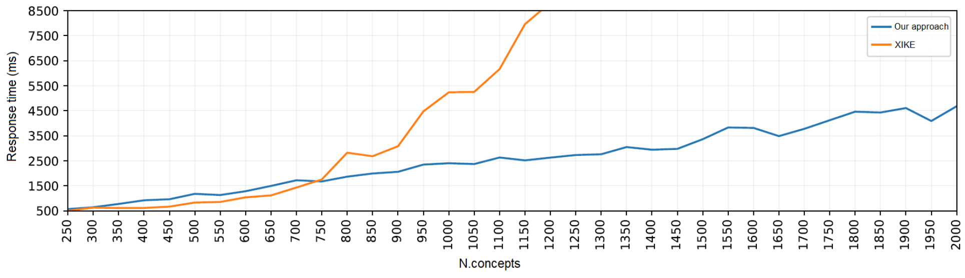


Fig. 10.5: Computation time of XIKE and our approach against the number of concepts to process

From the analysis of this figure, it clearly emerges that, as for this aspect, our approach is much better than XIKE. Indeed, the difference in the computation time between it and XIKE is of various orders of magnitude and is such to make XIKE, and the other systems mentioned above, unsuitable to handle the number and the size of the data sources characterizing the current big data scenario.

With reference to this claim, we observe that in this experiment the response time is measured against the number of concepts in the source metaschema. As such, already a set of sources with 1500 concepts can be considered "large". Indeed, it would correspond, for instance, to a set of E/R schemas consisting of about 1500 entities or a set of XML Schemas defining about 1500 different element types.

Furthermore, in this analysis, we must not forget that XIKE and the approaches mentioned above are not capable of handling unstructured data, which represents the second (and, for many verses, most important) peculiarity of our approach.

10.3.3 Discussion on the scalability

The previous experiment represents a first confirmation of the quickness and the scalability of our approach. In this section, we aim at finding a further confirmation of this trend by considering a much more numerous and articulated set of sources and by comparing the accuracy and the response time of our approach, of XIKE [129] and DIKE [284]. This last is an approach operating very well in structured sources, as shown by the comparison tests described in [303].

Clearly, if we want to compare these three approaches, the only way of proceeding is to consider structured sources because they are the only ones handled by DIKE. In particular, we considered the database schemas of Italian Central Government Offices (hereafter, ICGO). They include about 300 databases that are heterogeneous both in the data model and languages (e.g., hierarchical, network, relational), as well as in their structure and complexity, ranging from simple databases with schemas including few objects, to very complex databases [286].

Observe that our approach, XIKE and DIKE are all based on graphs and on the computation of similarities of the neighbors of the involved objects. However, DIKE has been thought for relatively small structured databases. As a consequence, when it computes the similarity of two objects belonging to different sources, it considers the similarity of their direct neighbors, the one of the neighbors of the direct neighbors, and so forth, until it terminates a fixpoint computation. In the worst case, the number of iterations of the fixpoint computation could be equal to the number of concepts of one of the involved sources. Clearly, performing such a high number of iterations allows DIKE to return very accurate results, but the required computation time is generally very high not only from the worst case computational complexity viewpoint but also from the real computation time point of view. In XIKE, the possible number and dimension of data sources is higher than DIKE and they can be both structured and unstructured. As a consequence, there is the need to limit the number of iterations of the fixpoint computation. For this reason, the concept of severity level has been introduced. In particular, a user can choose a severity level u between 1 and n and the fixpoint computation is not completed but terminates after u iterations. The higher u the more accurate and slower XIKE. The approach here presented has been designed having in mind a big data scenario and structured, semi-structured and unstructured data sources. As a consequence, it must be necessarily lightweight. For this reason, differently from DIKE and XIKE, we removed the fixpoint computation or, better, we limited it to 2 iterations. This could cause a reduction of accuracy but it is the only way to extend the approach of DIKE and XIKE also to a big data scenario.

System	*Precision*	*Recall*	*F-Measure*	*Overall*
DIKE (Synonymies)	0.98	0.93	0.95	0.91
DIKE (Homonymies)	0.96	0.95	0.95	0.91
XIKE $u = 5$ (Synonymies)	0.96	0.91	0.93	0.87
XIKE $u = 5$ (Homonymies)	0.93	0.93	0.93	0.86
XIKE $u = 2$ (Synonymies)	0.84	0.86	0.85	0.70
XIKE $u = 2$ (Homonymies)	0.85	0.86	0.85	0.71
Our approach (Synonymies)	0.83	0.81	0.82	0.64
Our approach (Homonymies)	0.81	0.83	0.82	0.64

Table 10.8: Precision, Recall, F-Measure and Overall of DIKE, XIKE ($u = 5$, $u = 2$) and our approach

Analogously to what happened in the previous section, in order to verify the theoretical conjectures explained above, we applied our approach, DIKE and XIKE (with $u = 5$ and, then, with $u = 2$) to ICGO databases. The obtained results are reported in Table 10.8.

The results of this table confirm our conjectures. DIKE provides a higher Precision, Recall, F-Measure and Overall than XIKE which, in turn, provides better results than our approach. Finally, XIKE, with a severity level equal to 5 provides better results than XIKE with a severity level equal to 2. The former tend to be comparable with the ones of DIKE; the latter tend to be comparable with the ones of our approach. This is in line with the fact that when u tends to 5 the fixpoint computation tends to be complete; instead, when u tends to 2, it tends to be removed and substituted with only one iteration.

In any case, we would like to remark that the results obtained by our approach are still acceptable.

After having verified our conjectures about accuracy, we analyzed the ones regarding computation time. In particular, the average computation time of DIKE, XIKE (with $u = 5$ and $u = 2$) and our approach is reported in Figure

From the analysis of this figure, it is easy to observe that the lower performance in terms of accuracy of our approach is largely balanced by an increased performance in terms of computation time. In a big data context, this aspect is mandatory. As a matter of fact, Figure 10.3.3 shows that DIKE and XIKE (especially when the severity level is high), even if very accurate, could not be applied in a big data scenario associated with smart cities.

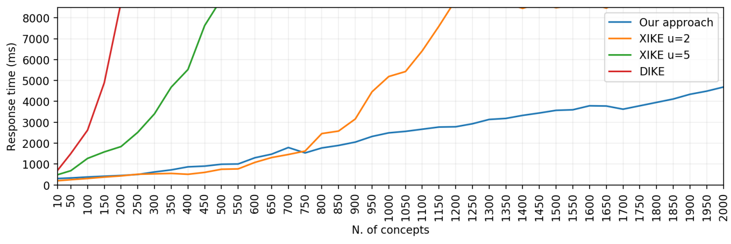


Fig. 10.6: Computation time of DIKE, XIKE ($u = 5$ and $u = 2$) and our approach against the number of concepts to process

10.3.3

10.4 Related Work

Schema matching is one of the most investigated topics in the past database research. At the beginning, all schema matching approaches proposed by researchers were manual and operated only on structured databases. With the passage of time, researchers proposed semi-automatic or automatic schema matching approaches capable of handling both structured and semi-structured data sources. With the advent of big data, unstructured data are becoming more and more frequent and important.

Schema matching approaches were thought to consider several kinds of heterogeneities; the most relevant of them are lexicographic, structural and semantic ones. The first deals with names and terms; the second considers type formats, data representation models and structural relationships among concepts; the third regards the meaning of involved data.

Let us see, now, in more detail, an overview of several approaches to perform schema matching from the beginning to the present day.

In [85], an approach to transform structured documents by leveraging schema graph matching is proposed. In particular, an XML schema to map each structured document is defined; for this purpose, some XSLT scripts are automatically generated. In [245], Cupid, a system for deriving inter-schema properties among heterogeneous sources, is proposed. Cupid leverages two different matchings, namely the *structure* and the *linguistic* ones. In [73], MOMIS, a system supporting querying and information source integration in a semi-automatic fashion, is presented. MOMIS implements a clustering procedure for the extraction of inter-schema properties. DIKE and XIKE [286, 129, 285], as well as the approaches described in [102, 138] also belong to this generation. An overview of this generation of schema matching approaches can be found in [303, 76].

More recent approaches, which significantly differ from the classical ones, are based on probabilistic methods, applied to networks of schemas [192]. They allow the definition of network-level integrity constraints for matching, as well as the analysis of query/click logs [141, 270], specifying the class of desired user-based schema matching.

In [40], an XML-based schema matching approach conceived to operate on large-scale schemas is presented. This approach leverages Prufer sequences. It performs a two-step activity; during the first step it parses XML schemas in schema trees; during the second one, it exploits Label Prufer Sequences (LPS) to capture schema tree semantic information.

In [274], SMART, a Schema Matching Analyzer and Reconciliation Tool, designed for the detection and the next reconciliation of matching inconsistencies, is proposed. SMART is semi-automatic because it requires the intervention of an expert for the validation of results.

In [253], the authors propose an approach to determine the semantic similarity of terms using the knowledge present in the search history logs from Google. For this purpose, they exploit four techniques that evaluate: *(i)* frequent co-occurrence of terms in search patterns; *(ii)* computation of the relationship between search patterns; *(iii)* outlier coincidence on search patterns; *(iv)* forecasting comparisons.

In [45], a framework for the management of a data lake through the corresponding metadata, is proposed. This framework leverages schema matching techniques to identify similarities between the attributes of different datasets. These techniques consider both schemas (specifically, attribute types and dependencies) and instances (specifically, attribute values) [76]. It integrates different schema matching approaches proposed in the last years, like graph matching, usage-based matching, document content similarity detection and document link similarity detection. [256] proposes an instance-based approach to find 1-1 schema matches. It combines the semantics provided by Google and regular expressions. It does not work well in a scenario where sources are very heterogeneous and data are stored in their row way. Another instance-based approach is presented in [196]. It faces the heterogeneity of the different schemas by leveraging an ad-hoc mapping language.

Most schema matching approaches based on similarities often filter out unnecessary matching and information [291] in such a way as to operate easier and faster.

As we have seen in this overview, schema matching has been largely investigated in the past for very heterogeneous scenarios, and very different approaches have been adopted to reach disparate goals. In this "mare magnum" of approaches, ours is characterized by the following features: *(i)* it has been specifically conceived to handle also unstructured sources; *(ii)* it has been designed to be scalable and, therefore, lightweight; *(iii)* it is automatic; *(iv)* in spite of these two last features, it presents a good accuracy, as we saw in Section 10.3.

11

An approach to extracting thematic views from highly heterogeneous sources

In the last years, data lakes are emerging as an effective and efficient support for information and knowledge extraction from a huge amount of highly heterogeneous and quickly changing data sources. Data lake management requires the definition of new solutions, very different from the ones adopted for data warehouses in the past. In addition, new techniques of navigation of such data lakes are required by the scenario. Indeed, one of the main issues to address consists in the extraction of thematic views topic-driven from the (very heterogeneous and generally unstructured) data sources of a data lake. So, we propose a two-step technique to extract thematic views from the sources of a data lake, based on similarity and other semantic relations among the metadata of data sources.

11.1 Introduction

In the last years, data lakes are emerging as an effective and efficient answer to the problem of extracting information and knowledge from a huge amount of highly heterogeneous and quickly changing data sources [146]. Data lake management requires the definition of new techniques, very different from the ones adopted for data warehouses in the past. These techniques may exploit the large set of metadata always supplied with data lakes, which represent their core and the main tool allowing them to be a very competitive framework in the big data era.

After proposing a new model to handle heterogeneous sources of information (Chapter 9) and a new approach to extract semantic properties from these different sources we now investigate and present a new supervised approach to extracting thematic views from highly heterogeneous sources of a data lake. Our solution can cooperate and it can be fully integrated with the above proposals, representing all the data lake sources by means of a suitable network. Indeed, networks are very flexible structures that allow the modelling of almost all phenomena that researchers aim at investigating [84]. Thanks to this uniform representation of the data lake sources, the extraction of thematic views from them can be performed by exploiting graph-based tools. We define "supervised" our approach because it requires the user to specify the set of topics $T = \{T_1, T_2, \ldots, T_n\}$ that should be present in the thematic view(s) to extract. However, this technique could work also without combining it with our previous proposals.

With the term *thematic views* we mean the situation in which the data manager, the analyst or anyone who has to navigate the huge data lake searches for some information filtered by one or more topics of interests.

In the new scenario of data lakes, where data are stored in its row format and where information and knowledge deriving by it are always more relevant in the decision making process, it is crucial to propose new techniques able to satisfy such requirements as faster as possible. This is the main reason that led us to investigate this scenario.

In detail, our approach consists of two steps. The former is mainly based on the structure of involved sources. It exploits several notions typical of (social) network analysis, such as the notion of ego network, which actually represents the core of the proposed approach. The latter exploits a knowledge repository, which is used to discover new relationships, other than synonymies, among metadata, with the purpose to refine the integration of different thematic views obtained after the first step. In this step, our approach relies on DBpedia[1].

11.2 An approach to extracting thematic views

Let define with DL the data lake that we want navigate in order to extract thematic views, whose different sources are represented by means of the model described in Chapter 9. Our approach is defined as "supervised" just because it requires the user to specify the set of topics $T = \{T_1, T_2, \cdots, T_l\}$, which should be present in the thematic view(s) to extract. It consists of two steps, the former mainly based on the structure of the sources at hand, the latter mainly focusing on the corresponding semantics.

Step 1

The first step of our approach receives a data lake DL, a set of topics

$$T = \{T_1, T_2, \cdots, T_l\}$$

representing the themes of interest for the user, and a dictionary Syn of synonymies involving the objects stored in the sources of DL. This dictionary could be obtained as output of the proposal we presented in Chapter 10, but at the same time we could

[1] DBpedia: `http://dbpedia.org`

a generic thesaurus, such as BabelNet [271], a domain-specific thesaurus, or a dictionary obtained by taking into account the structure and the semantics of the sources, which the corresponding objects refer to (such as the dictionaries produced by XIKE [129], MOMIS [74] or Cupid [245]).

In this step, the concept of *ego network* from graph theory and network analysis plays a key role. We recall that an ego network consists of a focal node (the ego) and the nodes it is directly connected to (the "alters"), plus the ties, if any, between the alters.

Let T_i be a topic of T. Let $Obj_i = \{o_{i_1}, o_{i_2}, \cdots, o_{i_q}\}$ be the set of the objects synonymous of T_i in DL. Let $N_i = \{n_{i_1}, n_{i_2}, \cdots, n_{i_q}\}$ be the corresponding nodes.

First, Step 1 constructs the ego networks

$$E_{i_1}, E_{i_2}, \cdots, E_{i_q}$$

having

$$n_{i_1}, n_{i_2}, \cdots, n_{i_q}$$

as the corresponding egos. Then, it merges all the egos into a unique node n_i. In this way, it obtains a unique ego network E_i from $E_{i_1}, E_{i_2}, \cdots, E_{i_q}$.

If a synonymy exists between two alters belonging to different ego networks, then these are merged into a unique node and the corresponding arcs linking them to the ego n_i are merged into a unique arc. At the end of this task, we have a unique ego network E_i corresponding to T_i.

After having performed the previous task for each topic of T, we have a set $E = \{E_1, E_2, \cdots, E_l\}$ of l ego networks. At this point, Step 1 finds all the synonymies of Syn involving objects of the ego networks of E and merges the corresponding nodes. After all the possible synonymies involving objects of the ego network of E have been considered and the corresponding nodes have been merged, a set $V = \{V_1, \cdots, V_g\}$, $1 \leq g \leq l$, of networks representing potential views is obtained.

If $g = 1$, then it is possible to conclude that Step 1 has been capable of extracting a unique thematic view comprising all the topics required by the user. Otherwise, there exist more views each comprising some (but not all) of the topics of interest for the user. If $g = 1$, Step 2 is performed to make more precise and complete the unique view representing all the topics of T. If $g > 1$, Step 2 aims at finding other relationships, different from synonymies, among the objects of the views of V in such a way as to try to obtain a unique view embracing all the topics of interest for the user.

Step 2

This step starts by enriching each view $V_i \in V$. For this purpose, it connects each of its elements to all the semantically related concepts taken from a reference knowledge repository.

In this work, we rely on DBpedia, one of the largest knowledge graphs in the Linked Data context, including more than 4.58 million entities in RDF. To this aim, first each element of V_i (including its synonyms) is mapped to the corresponding entry in DBpedia. In many cases, such a mapping is already provided by BabelNet[2]. Then, for each DBpedia entry, all the related concepts are retrieved. In DBpedia, knowledge is structured according to the Linked Data principles, i.e. as an RDF graph built by triples. Each triple $\langle s(ubject), p(roperty), o(bject)\rangle$ states that a subject s has a property p, whose value is an object o. Both subjects and properties are resources (i.e., nodes in DBpedia's knowledge graph), whereas objects may be either resources or literals (i.e., values of some primitive data types, such as strings or numbers). Each triple represents the minimal component of the knowledge graph. This last is built by merging triples together. Therefore, retrieving the related concepts for a given element x implies finding all the triples where x is either the subject or the object.

For each view $V_i \in V$, the procedure to extend it consists of the following three sub-steps:

1. *Mapping*: for each node $n \in V_i$, its corresponding DBpedia entry d is found.
2. *Triple extraction*: all the related triples $\langle d, p, o\rangle$ and $\langle s, p, d\rangle$, i.e., all the triples in which d is either the subject or the object, are retrieved.
3. *View extension*: for each retrieved triple $\langle d, p, o\rangle$ (resp., $\langle s, p, d\rangle$), V_i is extended by defining a node for the object o (resp., s), if not already existing, linked to n through an arc labeled as p.

These three tasks are repeated for all the views of V. As previously pointed out, this enrichment procedure is particularly important if $|V| > 1$ because the new derived relationships could help to merge the thematic views that was not possible to merge during Step 1. In particular, let $V_i \in V$ and $V_j \in V$ be two views of V, and let V_i' and V_j' be the extended views corresponding to them. If there exist two nodes $n_{i_h} \in V_i'$ ad $n_{j_k} \in V_j'$ such that $n_{i_h} = n_{j_k}$[3], then they can be merged in one node; if this happens, V_i' and V_j' become connected. After all equal nodes of the views of V have

[2] Whenever this does not happen, the mapping can be automatically provided by the DBpedia Lookup Service (`http://wiki.dbpedia.org/projects/dbpedia-lookup`).

[3] Here, two nodes are equal if the corresponding name coincide.

been merged, all the views of V could be either merged in one view or not. In the former case, the process terminates with success. Otherwise, it is possible to conclude that no thematic views comprising all the topics specified by the user can be found. In this last case, our approach still returns the enriched views of V and leaves the user the choice to accept of reject them.

11.2.1 An example case

In this section, we present an example case aiming at illustrating the various tasks of our approach. Here, we consider: *(i)* a structured source, called *Weather Conditions* (W, in short), whose corresponding E/R schema is not reported for space limitations; *(ii)* two semi-structured sources, called *Climate* (C, in short) and *Environment* (E, in short), whose corresponding XML Schemas are not reported for space limitations; *(iii)* an unstructured source, called *Environment Video* (V, in short), consisting of a YouTube video and whose corresponding keywords are: $garden$, $flower$, $rain$, $save$, $earth$, $tips$, $recycle$, $aurora$, $planet$, $garbage$, $pollution$, $region$, $life$, $plastic$, $metropolis$, $environment$, $nature$, $wave$, eco, $weather$, $simple$, $fineparticle$, $climate$, $ocean$, $environmentawareness$, $educational$, $reduce$, $power$, $bike$.

By applying the approaches mentioned in Chapter 9, we obtain the corresponding representations in our network-based model, shown in Figure 11.1.

Assume, now, that a user specifies the following set T of topics of her interest: $T = \{Ocean, Area\}$. First, our approach determines the terms (and, then, the objects) in the five sources that are synonyms of $Ocean$ and $Area$. As for $Ocean$, the only synonym present in the sources is Sea; as a consequence, Obj_1 comprises the node $Ocean$ of the source V ($V.Ocean$[4]) and the node Sea of the source C ($C.Sea$). An analogous activity is performed for $Area$. At the end of this task we have that $Obj_1 = \{V.Ocean, C.Sea\}$ and $Obj_2 = \{W.Place, C.Place, V.Region, E.Location\}$.

Step 1 of our approach proceeds by constructing the ego networks corresponding to the objects of Obj_1 and Obj_2. They are reported in Figure 11.2.

Now, let's consider the ego networks corresponding to $V.Ocean$ and $C.Sea$. Our approach merges the two egos into a unique node. Then, it verifies whether further synonyms exist between the alters. Since none of these synonyms exists, it returns the ego network shown in Figure 11.3(a). The same task is performed to the ego networks corresponding to $W.Place$, $C.Place$, $V.Region$ and $E.Location$. In particular, first the four egos are merged. Then, synonyms between the alters $W.City$ and $C.City$ and the alters $W.Altitude$ and $C.Altitude$ are retrieved. Based on this, $W.City$ and $C.City$ are merged in one node, $W.Altitude$ and $C.Altitude$ in another

[4] Here and in the following, we use the notation $S.o$ to indicate the object o of the source S.

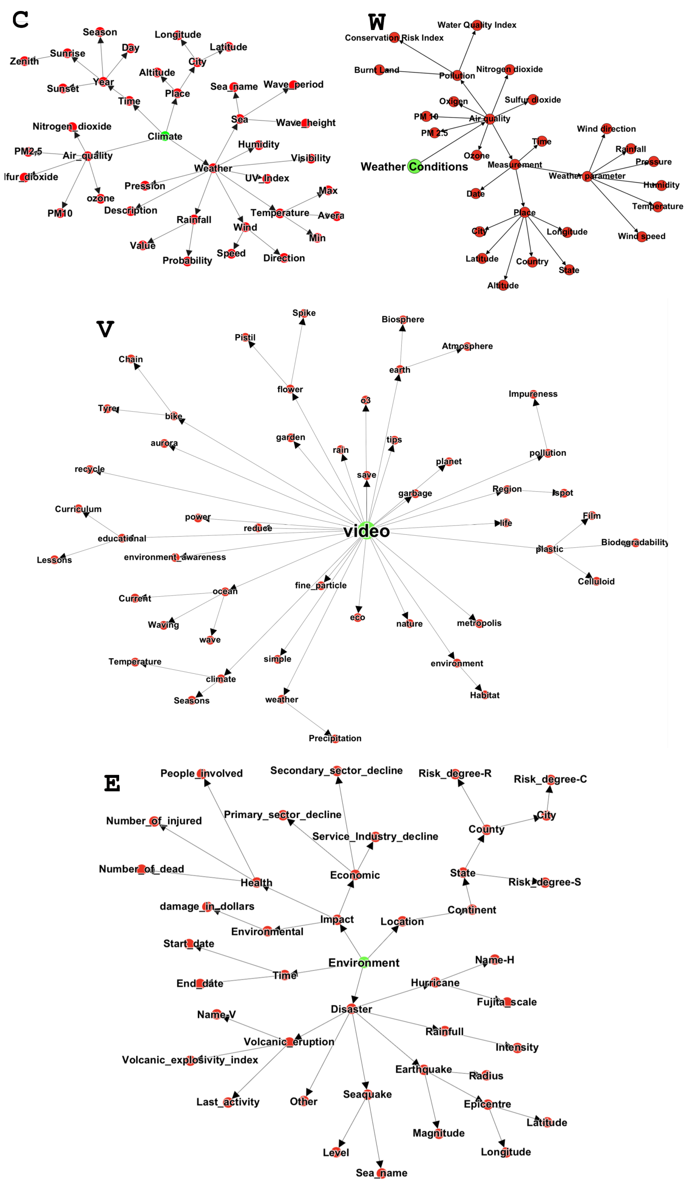


Fig. 11.1: Network-based representations of the four sources into consideration.

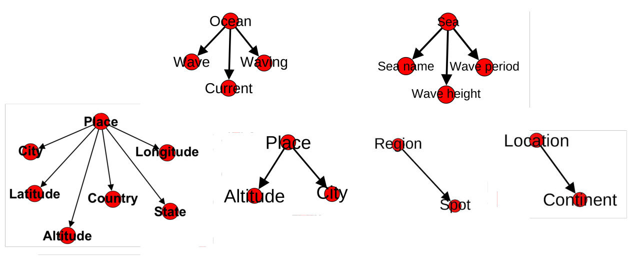


Fig. 11.2: Ego networks corresponding to *V.Ocean*, *C.Sea*, *W.Place*, *C.Place*, *V.Region* and *E.Location*.

node, the arcs linking the ego to *W.City* and *C.City* are merged in one arc and the ones linking the ego to *W.Altitude* and *C.Altitude* in another arc. In this way, the ego network shown in Figure 11.3(b) is returned. At this point, there are two ego networks, E_{Ocean} and E_{Area}, each corresponding to one of the terms specified by the user.

Now, Step 1 verifies if there are any synonyms between a node of E_{Ocean} and a node of E_{Area}. Since this does not happen, it terminates and returns the set $V = \{V_{Ocean}, V_{Area}\}$, where V_{Ocean} (resp., V_{Area}) coincides with E_{Ocean} (resp., E_{Area}).

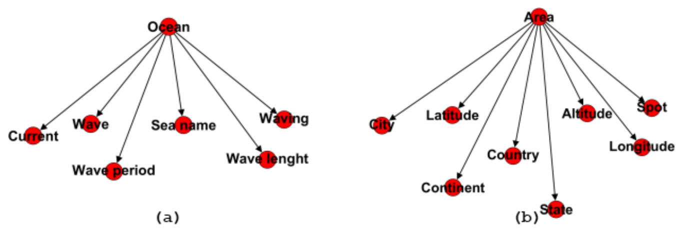


Fig. 11.3: Ego networks corresponding to *Ocean* and *Area*.

At this point, Step 2 is executed. As shown in Figure 11.4, first each term (synonyms included) is semantically aligned to the corresponding DBpedia entry (e.g., *Ocean* is linked to *dbo:Sea*, *Area* is linked to *dbo:Location* and *dbo:Place*, while *Country* to *dbo:Country*[5], respectively). After a single iteration, the triples ⟨*dbo:sea rdfs:range dbo:Sea*⟩ and ⟨*dbo:sea rdfs:domain dbo:Place*⟩ are retrieved. They correspond to the DBpedia property *sea*, which relates a country to the sea from which it is lapped. Other connections can be found by moving to specific instances of the mentioned resources. In this way, the following triples are retrieved: ⟨*instance rdf:type dbo:Sea*⟩, ⟨*instance rdf:type dbo:Location*⟩, ⟨*instance rdf:type dbo:Place*⟩, meaning that there are resource instances having types *Sea*, *Location* and *Place* simultaneously (e.g., *dbr:Mediterranean_Sea*). Furthermore, a triple ⟨*instance dbo:country dbo:Country*⟩ can be retrieved, meaning that those instances being a *Sea*, a *Location* or a *Place* specify in which *dbo:Country* they are located through the *dbo:country* property.

In this example case, Step 2 succeeded in merging the two views that Step 1 had maintained separated.

11.3 Related work

The new data lake scenario is characterized by several peculiarities that make it very different from the data warehouse paradigm. Hence, it is necessary to adapt (if possible) old algorithms conceived for data warehouses or to define new approaches

[5] Prefixes *dbo* and *dbr* stand for `http://dbpedia.org/ontology/` and `http://dbpedia.org/resource/`

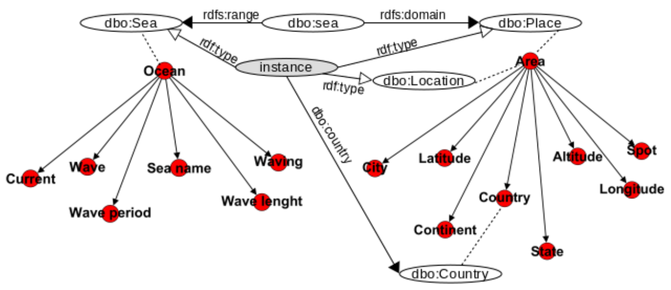


Fig. 11.4: The integrated thematic view.

capable of handling and taking advantage of the specificities of this new paradigm. However, most approaches proposed in the literature for data integration, query answering and view extraction do not completely fit the data lake paradigm. For instance, [101] proposes some techniques for building views on semi-structured data sources based on some expected queries. Other researchers focus on materialized views and, specifically, on throughput and execution time; therefore, they a-priori define a set of well-known views and, then, materialize them. Two surveys on this issue can be found in [182, 34]. The authors of [377] investigate the same problem but they focus on XML sources. The approach of [368] addresses the same issue by means of query rewriting; specifically, it transforms a query Q into a set of new queries, evaluates them, and, then, merges the corresponding answers to construct the materialized answer to Q. [60] proposes an approach to constructing materialized views for heterogeneous databases; it requires the presence of a static context and the pre-computation of some queries.

Another family of approaches exploits materialized views to perform tree pattern querying [367] and graph pattern queries [145]. Unfortunately, all these approaches are well-suited for structured and semi-structured data, whereas they are not scalable and lightweight enough to be used in a dynamic context or with unstructured data. An interesting advance in this area can be found in [333]. Here, the authors propose an incremental approach to address the graph pattern query problem on both static and dynamic real-life data graphs. Other kinds of views are investigated in [80] and [59]. In particular, this last paper uses virtual views to access heterogeneous data sources without knowing many details of them. For this purpose, it creates virtual views of the data sources.

Finally, semantic-based approaches have long been used to drive data integration in databases and data warehouses. More recently, in the context of big data, formal semantics has been specifically exploited to address issues concerning data variety/heterogeneity, data inconsistency and data quality in such a way as to increase understandability [188]. In the data lake scenario, semantic techniques have been successfully applied to more efficiently integrate and handle both structured and unstructured data sources by aligning data silos and better managing evolving data models. For instance, in [180], the authors discuss a data lake system with a semantic metadata matching component for ontology modeling, attribute annotation, record linkage, and semantic enrichment. Furthermore, [148] presents a system to discover and enforce expressive integrity constraints from data lakes. Similarly to what happens in our approach, knowledge graphs in RDF[6] are used to drive integration. To reach their objectives, these techniques usually rely on information extraction tools (e.g., Open Calais[7]) that may assist in linking metadata to uniform vocabularies (e.g., ontologies or knowledge repositories, such as DBpedia).

[6] RDF Concepts and abstract Syntax: `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`

[7] `http://www.opencalais.com`

12

An approach to extracting complex knowledge patterns among concepts belonging to heterogeneous sources

In the last few years, the "big data phenomenon" has disrupted both the research and the technological coordinates in the information systems cultural area. As a matter of fact, new models for representing big data, new architectures for handling them and new approaches to extracting information from them appear in order. Data represents the basis on which data scientists and decision makers extract information, knowledge and wisdom. For this reason it is important to provide the new paradigm of data lakes with a model that allows the user to navigate the whole data lake in a agile way. We propose a solution for this problem by defining a new approach to extracting complex knowledge patterns from the data stored in a data lake.

12.1 Introduction

In the last few years, the "big data phenomenon", with the corresponding $5V$ paradigm (Volume, Velocity, Variety, Veracity and Value), is rapidly changing the research and technological "coordinates" of the information system area [104, 366]. For instance, it is well known that data warehouses, generally handling structured and semi-structured data offline, are too complex and rigid to manage the wide amount and variety of rapidly evolving data sources of interest for a given organization, and the usage of more agile and flexible structures appears compulsory [130]. Data lakes are one of the most promising answers to this exigency [244]. A data lake is a storage repository containing a wide amount of raw data in its native format. Differently from a data warehouse, a data lake uses a flat architecture (so that the insertion and the removal of a source can be easily performed). However, the agile and effective management of data stored therein is guaranteed by the presence of a rich set of extended metadata. These allow a very agile and easily configurable usage of the data stored in the data lake. For instance, if a given application requires the querying of some data sources, one could process available metadata to determine the portion of the involved data lake to examine.

One of the most radical changes caused by the big data phenomenon is the presence of a huge amount of unstructured data. As a matter of fact, it is esteemed that, currently, more than 80% of the information available on the Internet is unstructured [117]. In presence of unstructured data, all the approaches developed in the past for structured and semi-structured data must be "renewed", and the new approaches will be presumably much more complex than the old ones [206, 342]. Think, for instance, of schema integration: unstructured sources do not have a representing schema and, often, only a set of keywords are given (or can be extracted) to represent the corresponding content [131].

We propose an approach to the extraction of complex knowledge patterns among concepts belonging to structured, semi-structured and unstructured sources in a data lake. Here, we use the term "complex knowledge pattern" to indicate an intensional relationship involving more concepts possibly belonging to different (and, presumably, heterogeneous) sources of a data lake. Formally speaking, a complex knowledge pattern consists of a logic succession $\{x_1, x_2, \dots, x_w\}$ of w objects such that there is a semantic relationship (specifically, a synonymy or a part-of relationship) linking the k^{th} and the $(k+1)^{th}$ objects $(1 \leq k \leq w-1)$ of the succession.

Our approach is network-based in that it represents all the data lake sources by means of suitable networks. As a matter of facts, networks are very flexible structures, which allow the modelling of almost all phenomena that researchers aim at investigating [84]. For instance, they have been used in the past to uniformly represent data sources characterized by heterogeneous, both structured and semi-structured, formats [129]. We also use networks to represent unstructured sources, which, as said before, do not have a representing schema. Furthermore, we propose a technique to construct a "structured representation" of the flat keywords generally used to represent unstructured data sources. This is a fundamental task because it highly facilitates the uniform management, through our network-based model, of structured, semi-structured and unstructured data sources.

Thanks to this uniform, network-based representation of the data lake sources, the extraction of complex knowledge patterns can be performed by exploiting graph-based tools. In particular, taking into consideration our definition of complex knowledge patterns, a possible approach for their derivation could consist in the construction of suitable paths going from the first node (i.e., x_1) to the last node (i.e., x_w) of the succession expressing the patterns. In this case, a user specifies the "seed objects" of the pattern (i.e., x_1 and x_w) and our approach finds a suitable path (if it exists) linking x_1 to x_w.

Since x_1 and x_w could belong to different sources, our approach should consider the possible presence of synonymies between concepts belonging to different sources, it should model these synonymies by means of a suitable form of arcs (cross arcs, or c-arcs), and should include both intra-source arcs (inner arcs, or i-arcs) and c-arcs in the path connecting x_1 to x_w and representing the complex knowledge pattern of interest.

Among all the possible paths connecting x_1 to x_w, our approach takes the shortest one (i.e., the one with the minimum number of c-arcs and, at the same number of c-arcs, the one with the minimum number of i-arcs). This choice is motivated by observing that a knowledge pattern should be as semantically homogeneous as possible. With this in mind, it is appropriate to reduce as much as possible the number of synonymies to be considered in the knowledge pattern from x_1 to x_w. This because a synonymy is weaker than an identity and, furthermore, it involves objects belonging to different sources which, inevitably, causes an "impairment" in the path going from x_1 to x_w. Moreover, there is a further, more technological reason leading to the choice of the shortest path. Indeed, it is presumable that, after a complex knowledge pattern has been defined and validated at the intensional level, one would like to recover the corresponding data at the extensional level. In this case, in a big data scenario, reducing the number of the sources to query would generally reduce the volume and the variety of the data to process and, hence, would increase the velocity at which the data of interest are retrieved and processed.

As it will be clear in the following, there are cases in which synonymies (and, hence, c-arcs) are not sufficient to find a complex knowledge pattern from x_1 to x_w. In these cases, our approach performs two further attempts in which it tries to involve string similarities first, and, if even these properties are not sufficient, part-whole relationships. If neither synonymies nor string similarities nor part-whole relationships allow the construction of a path from x_1 to x_w, our approach concludes that, in the data lake into consideration, a complex knowledge pattern from x_1 to x_w does not exist.

12.2 Extraction of complex knowledge patterns

12.2.1 General description of the approach

Our approach to the extraction of complex knowledge patterns operates on a data lake *DL* whose data sources are represented by means of the formalism described in Chapter 9 and we base our approach on the network model of data lakes there we presented.

It receives a dictionary *Syn* of synonymies involving the objects stored in the sources of *DL*. This dictionary could be a generic thesaurus, such as BabelNet [271], a domain-specific thesaurus, or a dictionary obtained by taking into account the structure and the semantics of the sources, which the corresponding objects refer to (such as the dictionaries produced by XIKE [129], MOMIS [74] or Cupid [245]). Furthermore, it receives two objects x_{i_h}, belonging to a source D_h of DL, and x_{j_q}, belonging to a source D_q of DL. x_{i_h} and x_{j_q} represent the base on which the complex knowledge pattern to extract should be constructed. As a matter of fact, we recall that a complex knowledge pattern consists of a logic succession $\{x_1, x_2, \ldots, x_w\}$ of w objects such that: *(i)* x_1 coincides with x_{i_h}; *(ii)* x_w coincides with x_{j_q}; *(iii)* there is a semantic relationship (specifically, a synonymy or a part-of relationship) linking the k^{th} and the $(k+1)^{th}$ objects $(1 \leq k \leq w-1)$ of the succession.

Before illustrating our approach in detail, it is necessary to introduce some preliminary concepts, namely the ones of i-arcs, c-arcs and pw-arcs. In Chapter 9, we have seen that, given a source D_k of DL, $\mathcal{M}_B^k = \langle N_k, A_k \rangle$ and $A_k = A'_k \cup A''_k \cup A'''_k$. All the arcs of A_k are internal to D_k; we call them *i-arcs* (i.e., internal arcs) in the following. Now, let x_{i_h} and x_{j_q} be two objects for which a synonymy exists in Syn. We call *c-arcs* (i.e., cross arcs) the arcs (n_{i_h}, n_{j_q}) and (n_{j_q}, n_{i_h}) corresponding to this synonymy. These arcs are extremely important because they allow the extraction, the processing and the management of information coming from different sources. Finally, given an arc $(n_{i_k}, n_{j_k}) \in A'_k \cup A''_k$, we call *pw-arc* (i.e., part-whole arc) the arc (n_{j_k}, n_{i_k}). The pw-arc (n_{j_k}, n_{i_k}) is the "reverse" arc of (n_{i_k}, n_{j_k}) because it starts from the part and ends to the whole[1]. The name of this arc clearly indicates its nature. As it is evident from the definition of A'_k and A''_k, the i-arc (n_{i_k}, n_{j_k}) specifies the existence of a part-of relationship, from the whole (n_{i_k}) to the part (n_{j_k}). The arc (n_{j_k}, n_{i_k}) is the reverse one going from the part to the whole.

Our approach operates as follows. Let n_{i_h} (resp., n_{j_q}) be the node corresponding to x_{i_h} (resp., x_{j_q}).

- If $h = q$, we have a trivial case and the complex knowledge pattern from n_{i_h} to n_{j_q} is obtained by selecting the set of the arcs belonging to the shortest path from n_{i_h} to n_{j_q}.
- If $h \neq q$, then c-arcs and pw-arcs must be considered. First, our approach determines the set of complex knowledge patterns each formed by a c-arc from n_{i_h} to a node n_{t_l} synonymous of n_{i_h}, followed by a complex knowledge pattern from n_{t_l} to n_{j_q}. Then, it determines the set of complex knowledge patterns each formed by an i-arc from n_{i_h} to a node n_{t_h}, being a part of n_{i_h}, followed by a complex knowledge pattern from n_{t_h} to n_{j_q}. If at least one of these two sets is not empty, it returns the complex knowledge pattern having the minimum number of c-arcs.

[1] In order to keep the layout simple, in the graphical representation of our model, we explicitly show only the arcs from the whole to the parts; the corresponding pw-arcs are considered implicit.

If both these sets are empty, then our approach performs a last attempt to find a complex knowledge pattern by considering pw-arcs having n_{i_h} as target, if they exist. In this case, it determines the set of complex knowledge patterns each formed by a pw-arc from n_{i_h} to a node n_{t_h} followed by a complex knowledge pattern from n_{t_h} to n_{j_q}. If this set is not empty, it returns the complex knowledge pattern having the minimum number of pw-arcs.

12.2.2 Technical Details

As previously pointed out, our approach operates on a data lake DL. It receives a dictionary Syn of synonymies regarding the objects of DL, along with two objects x_{i_h}, belonging to a source D_h of DL, and x_{j_q}, belonging to a source D_q of DL. Let n_{i_h} (resp., n_{j_q}) be the node corresponding to x_{i_h} (resp., x_{j_q}), then the computation of $CKP(n_{i_h}, n_{j_q})$, i.e. of the complex knowledge pattern from n_{i_h} to n_{j_q}, is recursively performed as follows:

- If $h = q$, x_{i_h} and x_{j_q} belong to the same source and, therefore, n_{i_h} and n_{j_q} belong to the same network. In this case, the complex knowledge pattern $CKP(n_{i_h}, n_{j_q})$ from n_{i_h} to n_{j_q} is obtained by selecting the set of the arcs belonging to the shortest path from n_{i_h} to n_{j_q}. Any algorithm previously proposed in the literature for computing the shortest path between two nodes (see [118] for a detailed description of them) can be adopted.
- If $h \neq q$, then n_{i_h} and n_{j_q} belong to different networks.
  First, the set $NSynSet_{i_h}$ of the nodes corresponding to the objects synonymous of x_{i_h} in Syn is determined as:

  $$NSynSet_{i_h} = \{n_{t_l} \mid (n_{i_h}, n_{t_l}) \in Syn\}$$

  Then, the set $CKPSynSet(n_{i_h}, n_{j_q})$ of the complex knowledge patterns from n_{i_h} to n_{j_q} and passing through a node of $NSynSet_{i_h}$ is computed. Formally:

  $$CKPSynSet(n_{i_h}, n_{j_q}) = \{SynCKP(n_{i_h}, n_{j_q}, n_{t_l}) | n_{t_l} \in NSynSet_{i_h}\}$$

  where:

  $$SynCKP(n_{i_h}, n_{j_q}, n_{t_l}) = \{(n_{i_h}, n_{t_l}) \cup CKP(n_{t_l}, n_{j_q})\}$$

  After this, the set $NPartSet_{i_h}$ of the nodes representing a part of n_{i_h} (which, in this case, is the whole) is determined as:

  $$NPartSet_{i_h} = \{n_{t_h} | (n_{i_h}, n_{t_h}) \in A'_h \cup A''_h\}$$

  Then, the set $CKPPartSet(n_{i_h}, n_{j_q})$ of the complex knowledge patterns from n_{i_h} to n_{j_q} and passing through a node of $NPartSet_{i_h}$ is computed. Formally:

$$CKPPartSet(n_{i_h}, n_{j_q}) = \{PartCKP(n_{i_h}, n_{j_q}, n_{t_h}) | n_{t_h} \in NPartSet_{i_h}\}$$

where:

$$PartCKP(n_{i_h}, n_{j_q}, n_{t_h}) = \{(n_{i_h}, n_{t_h}) \cup CKP(n_{t_h}, n_{j_q})\}$$

If $CKPSynSet(n_{i_h}, n_{j_q}) \neq \emptyset$ and/or $CKPPartSet(n_{i_h}, n_{j_q}) \neq \emptyset$, our approach selects as $CKP(n_{i_h}, n_{j_q})$ the complex knowledge pattern having the minimum number of c-arcs. If more than one pattern exists with the same minimum number of c-arcs, it returns the one with the minimum number of i-arcs. If more than one pattern exists with these characteristics, it randomly selects one of them.
If $CKPSynSet(n_{i_h}, n_{j_q}) = \emptyset$ and $CKPPartSet(n_{i_h}, n_{j_q}) = \emptyset$, then c-arcs are not sufficient to find a complex knowledge pattern from n_{i_h} to n_{j_q}. However, a last attempt to find such a pattern can be performed by exploiting a pw-arc having n_{i_h} as target, if it exists.
In particular, let $NWholeSet_{i_h}$ be the set of the nodes of which n_{i_h} is a part. It is determined as:

$$NWholeSet_{i_h} = \{n_{t_h} | (n_{t_h}, n_{i_h}) \in A'_h \cup A''_h\}$$

Then, if $NWholeSet_{i_h} = \emptyset$, there is no possibility to find a complex knowledge pattern from n_{i_h} to n_{j_q}. Otherwise, the set $CKPWholeSet(n_{i_h}, n_{j_q})$ of the complex knowledge patterns between n_{i_h} and n_{j_q} and passing through a node of $NWholeSet_{i_h}$ is computed. Formally:

$$CKPWholeSet(n_{i_h}, n_{j_q}) = \{WholeCKP(n_{i_h}, n_{j_q}, n_{t_h}) | n_{t_h} \in NWholeSet_{i_h}\}$$

where:

$$WholeCKP(n_{i_h}, n_{j_q}, n_{t_h}) = \{(n_{i_h}, n_{t_h}) \cup CKP(n_{t_h}, n_{j_q})\}$$

Once $WholeCKP(n_{i_h}, n_{j_q}, n_{t_h})$ has been constructed, if it is not empty, our approach selects as $CKP(n_{i_h}, n_{j_q})$ the complex knowledge pattern having the minimum number of pw-arcs. If more than one pattern exists with the same minimum number of pw-arcs, it returns the one with the minimum number of c-arcs. If more than one pattern exists with these characteristics, it selects the one with the minimum number of i-arcs. Finally, if more than one pattern exists with the same minimum number of i-arcs, it randomly selects one of them.

Computational complexity

As for the computational complexity of this approach, we can observe that:

- If $h = q$, any algorithm previously proposed in the literature for computing the shortest path between two nodes can be adopted. For instance, if the Dijkstra algorithm using a binary heap is implemented, the computational complexity of this step is $O(|A| \cdot log|N|)$, where $|A|$ is the total number of arcs of the data lake and $|N|$ is the total number of its nodes.
- If $h \neq q$, in the worst case, it is necessary to determine the sets $NSynSet_{i_h}$, $NPartSet_{i_h}$ and $NWholeSet_{i_h}$ and, then, for each node of these sets, to compute the shortest path from n_i to n_j bounded to pass through it.
  Now, $|NSynSet_{i_h}|$ is $O(|DL|)$ because there could be at most one synonymous of a node for each source. $|NPartSet_{i_h}|$ is $O(|N_{max}|)$, where $|N_{max}|$ is the number of nodes of the largest source of the data lake. For the same reason, $|NWholeSet_{i_h}|$ is $O(|N_{max}|)$.
  The complexity of the computation of the shortest path from n_i to n_j bounded to pass through a node is $O(|A| \cdot log(|N|)$, if the Dijkstra algorithm with the support of the binary heap is applied.
  Therefore, in this case, the computational complexity of the algorithm is:
  $$O(|A| \cdot log|N|) \cdot O(max(|N_{max}|, |DL|))$$
  Now, since generally $|N_{max}| \gg |DL|$, we have that the computational complexity of this step is:
  $$O(|A| \cdot log|N| \cdot |N_{max}|)$$

Since the computational complexity of the case $h \neq q$ is higher than the one of the case $h = q$, we can conclude that the overall computational complexity of our approach is $O(|A| \cdot log|N| \cdot |N_{max}|)$.

This computational complexity can be judged very good if we consider the problem to solve. Actually, one could say that it is high for real data lakes consisting of many sources. However, in these cases, we have that, in the reality, the corresponding graphs are very sparse and, therefore, $|A|$ is small. To better quantify this fact, in Section 12.4.2, we compare the theoretical and the real computation time of our approach against the number of nodes of the data lake.

12.3 Some case studies

In this section, we present some case studies devoted to illustrate the behaviour of our approach in the various possible cases. To perform our test cases, we constructed a data lake consisting of 2 structured sources, 4 semi-structured sources (i.e., 2 XML sources and 2 JSON ones) and 4 unstructured sources (i.e., 2 books and 2 videos). All

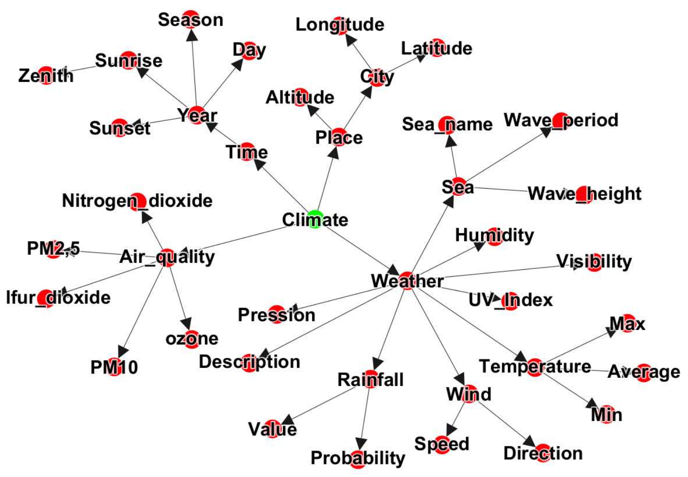


Fig. 12.1: The network corresponding to the source *Climate*

these sources store data about environment and pollution. To describe unstructured sources, we initially considered a set of keywords derived from Google Books, for books, and from YouTube, for videos.

The case studies we present in this section involve five sources of our data lake. These sources are the following:

- *Climate*. This is a JSON source storing data about weather and climatic conditions of several places. In this dataset, space and time are expressed at several granularity levels. In particular, time is expressed in years, seasons and days, whereas space is expressed in countries and cities; these last ones have associated the corresponding latitude and longitude. The network representing *Climate* is reported in Figure 12.1.
- *Energy*. This is a JSON source storing data about renewable and non-renewable energy sources used in the countries worldwide. Energy also stores data about the investments on the various kinds of energy source. The network representing *Energy* is reported in Figure 12.2.
- *Environment disasters*. This is an XML source storing data about environment disasters (e.g., earthquakes, seaquakes, volcanic eruptions, etc.), the places where they happened, the damages caused by them, and so forth. The network representing *Environment disasters* is reported in Figure 12.3.
- *Environment risks*. This is a book discussing about environment risks, their probabilities, their damages, etc. This is an unstructured source and, as such, it is represented by a set of keywords, which is reported in Table 12.1.
- *Air pollution*. This is a book focusing on air pollution, its causes, its consequences and the possible control strategies that can mitigate their impact on the environment. This is an unstructured source. Its keywords are reported in Table 12.2.

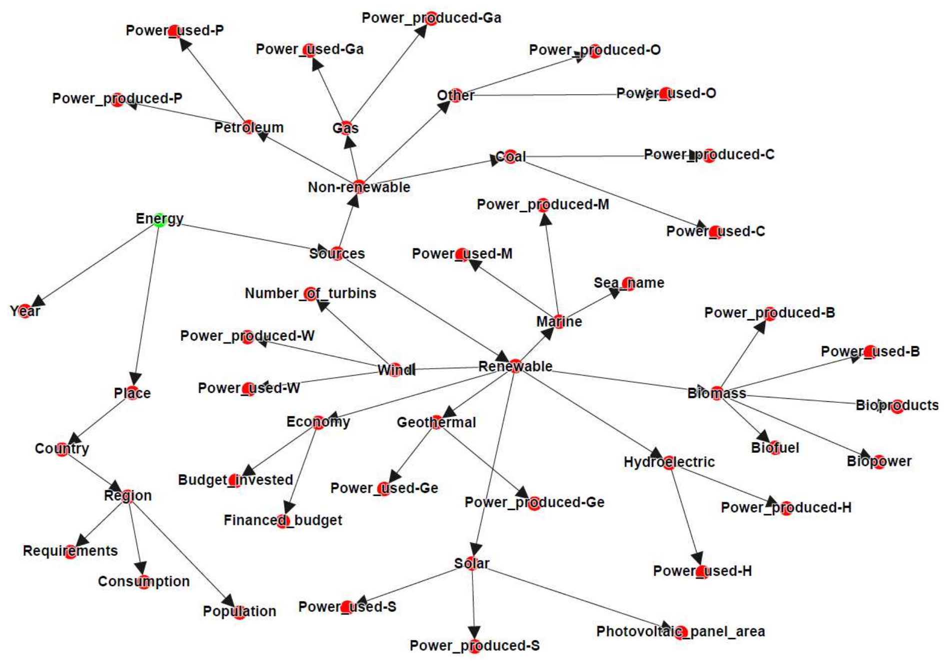


Fig. 12.2: The network corresponding to the source *Energy*

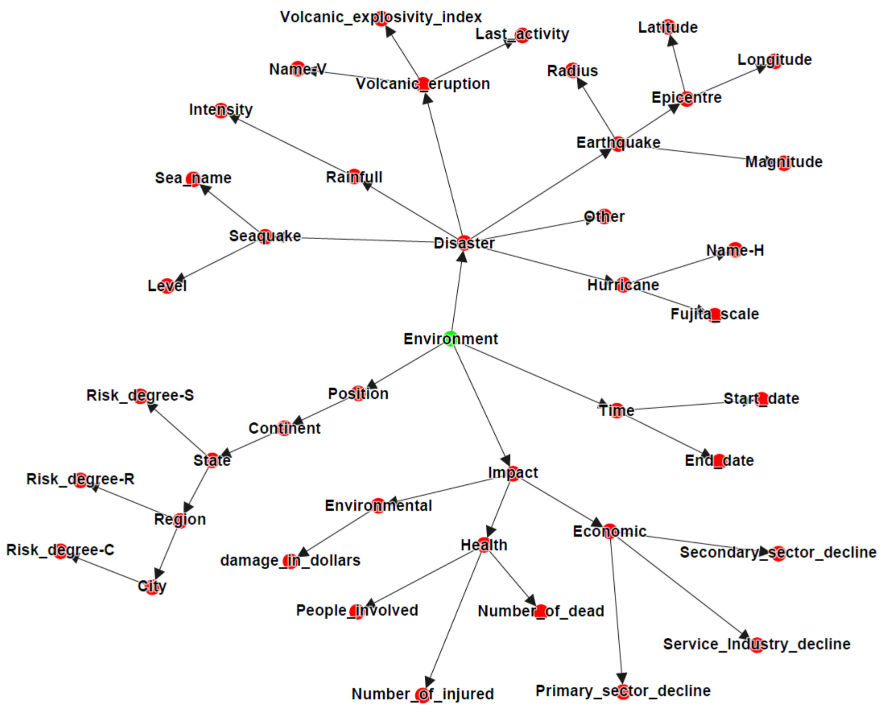


Fig. 12.3: The network corresponding to the source *Environment disasters*

The first case study we are considering is very simple because the two "seed objects" of the complex knowledge pattern are `Energy` and `Population`, both belonging to the source *Energy*. Since both the source and the target node of the knowledge pattern belong to the same network, the pattern is obtained simply by computing the

Keywords
absorptivecapacity, action, adjustments, adopted, agencies, agricultural, air, appraisal, areas, Bangladesh, Burton, capita, catastrophe, choice, coast, alcomprehensive, coping, cost, crops, damage, deaths, developingcountries, relief, drought, earthquake, economic, effects, effort, emergency, environment, environmental, estimated, evacuation, experience, extremeeventsfarmers, federa, Figure, flood, floodplain, forecasting, frequency, global, globalwarming, groups, hazardevent, hazardresearch, human, hurricane, impact, income, increase, individual, industrial, Kates, LabourBrigade, land, LiuLing, loss, magnitude, maize, major, measures, ment, million, naturaldisasters, naturalevents, naturalhazards, hazard, Nicaragua, occur, organization, pattern, people'scommunepercent, percent, population, possible, potential, prevent, protection, range, reduce, regions, risk, River, scale, scientific, social, society, SriLanka, storm, studies, threshold, tion, tornado, TristandaCunha, tropical, cyclone, TropicalStormAgnes, tsunami, UnitedKingdom, urban, vulnerable, warning, systems, zone, Managua, air, plant, disaster, airpollution, natural, Tanzania, TropicalStormAgnes.

Table 12.1: Keywords of the source *Environment risks*

Keywords
acid, activatedsludge, activity, aerosol, airpollution, airquality, air, anaerobicdigestion, approach, aquatic, areas, Assesment, atmosphere, biofuels, carbon, catalyst, cause, chemical, chlorine, climatechange, combustion, concentrations, contaminated, cycle, CycleAssessment, deposition, diesel, dose, drinkingwater, ecosystem, effects, effluent, emissions, energy, EnvironmentAgency, European, EuropeanCommission, EuropeanUnion, exposure, Figure, fuel, gases, global, human, hydrocarbons, impacts, important, industrial, landfill, legislation, levels, London, major, materials, measures, models, monitoring, nanoparticles, nitrate, nitrogen, nitrogendioxide, nuisance, operation, organic, oxidation, oxygen, ozone, particles, PBDEs, PCBs, pesticides, plant, potential, radiation, radiativeforcing, radioactive, range, reaction, recycling, reduce, regulation, regulatory, release, response, result, risk, sewage, significant, sludge, soil, sources, species, standards, stratosphere, studies, substances, sulfurdioxide, surface, temperature, toxic, transport, treatment, typically, urban, vehicles, wastemanagement, ambient, biological, compounds, Directive, engine, example, increase, metals, petrol, reactor, eutrophication.

Table 12.2: Keywords of the source *Air pollution*

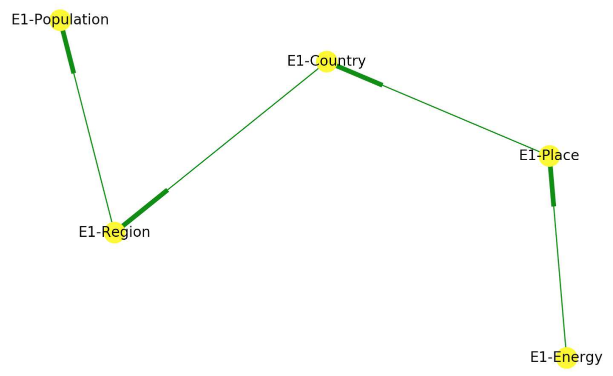


Fig. 12.4: Complex knowledge pattern from the node Energy to the node Population of the source *Energy*

shortest path from Energy to Population in the network of Figure 12.2. Actually, in this case, we have only one possible path, shown in Figure 12.4. This path consists of 4 i-arcs, no c-arcs and no pw-arcs.

The second case study we are considering involves as "seed objects" Position, belonging to *Environment disasters*, and Energy, belonging to *Energy*. In this case, it is evident the necessity of passing through at least one c-arc because the two objects belong to different sources. One of the synonyms of the object Position is the object Place, belonging to the source *Energy*. As a consequence, it is possible to define at least one path, starting from Position, passing through Place and reaching Energy. This path is shown in Figure 12.5 and consists of 1 i-arc, 1 c-arc and no pw-arc. An alternative path would involve the nodes Position and Continent of *Environment disasters* and the nodes Country, Place and Energy of *Energy*. However, this path would consist of 3 i-arcs, 1 c-arc and no pw-arc and, clearly, it is not the shortest path. As a consequence, in this case, our approach returns the path shown in Figure 12.5 as the complex knowledge pattern from Position to Energy.

The third case study we are considering involves, as "seed objects", Fujita_scale of *Environment disasters* and Risk of *Environment risks*. In this case, synonymies are not sufficient because there is no synonymy involving Fujita_scale. However, the "whole" node of Fujita_scale is Hurricane and there is a synonymy between Hurricane and Tornado. As a consequence, it is possible to define at least one path starting from Fujita_scale, passing through Hurricane and Tornado and ending to Risk. This path is shown in Figure 12.6. It consists of 1 i-arc, 1 c-arc and 1 pw-arc. This is also the shortest path from Fujita_scale to Risk and, therefore, the complex knowledge pattern between these two nodes.

The fourth and last case study is the most complex one because it involves more alternative synonymies that could be selected, with the consequent need to determine the best one. The "seed objects" are Risk_degree of *Environment disasters* and

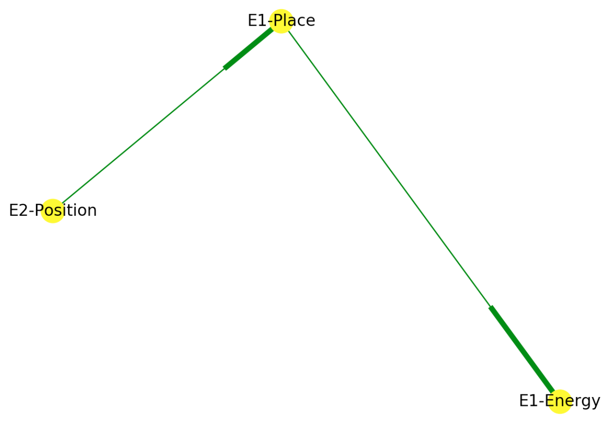


Fig. 12.5: Complex knowledge pattern from the node `Position` of the source *Environment disasters* to the node `Energy` of the source *Energy*

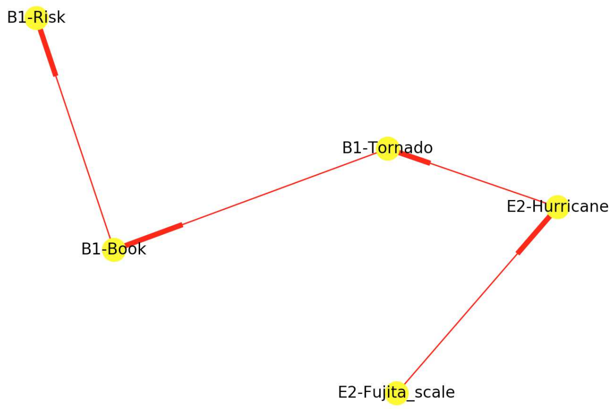


Fig. 12.6: Complex knowledge pattern from the node `Fujita_scale` of the source *Environment disasters* to the node `Risk` of the source *Environment risks*

`Emergency` of *Environment risks*. `Risk_degree` presents two synonymies in the dictionary; the former involves the object `Risk` of *Environment risks*; the latter regards the object `Risk` of *Air pollution*. As a consequence, at least two extremely different paths could exits. However, the path involving the node `Risk` of *Environment risks* can reach the target source through only 1 c-arc. The other one would need at least another c-arc to reach the target source. In particular, it should use the synonymy between `Risk` of *Air pollution* and `Hazard` of *Environment risks*. In Figure 12.7, we report both these paths. The former involves the nodes `Risk_degree`, `Risk`, Book and `Emergency` and consists of 2 i-arcs and 1 c-arc. The latter involves the nodes `Risk_degree`, `Risk`, `Hazard`, Book and `Emergency` and consists of 2 i-arcs and 2 c-arcs. Clearly, the shortest path is the former, which is selected as the complex knowledge pattern between the two seed nodes.

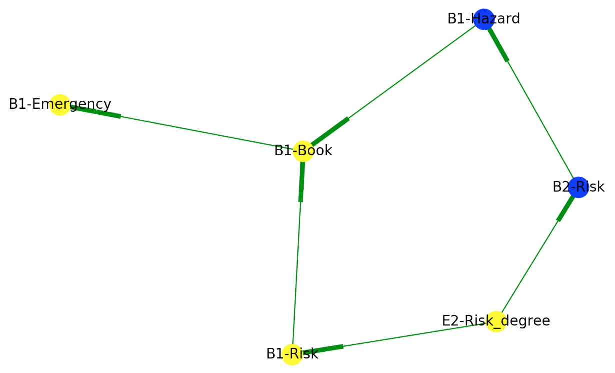


Fig. 12.7: Complex knowledge pattern from the node `Risk_degree` of the source *Environment disasters* to the node `Risk` of the source *Environment risks*

12.4 Discussion

This section is devoted to present a critical discussion of several aspects concerning our approach. It consists of three subsections. In the first, we present a comparison between our approach and the related ones. In the second, we evaluate the performance of our overall approach. Finally, in the fourth, we measure its efficiency for large datasets. To carry out the experiments described in this section, we used a server equipped with an Intel I7 Dual Core 5500U processor and 16 GB of RAM with the Ubuntu 16.04.3 operating system. Clearly, especially for the last experiments, the capabilities of this server were limited. However, as we will see below, we were mostly interested to compare theoretical worst case response times with the real ones. Clearly, in real contexts, whenever necessary, much more powerful servers could be used to reduce the response time.

12.4.1 Comparison

In Section 12.5, we have seen that we can distinguish four main families of approaches that are most related to ours. Specifically, these approaches aim at extracting: *(i)* keyword patterns; *(ii)* knowledge from structured sources; *(iii)* knowledge from heterogeneous sources; *(iv)* knowledge patterns through network analysis-based techniques.

As for the first family, the corresponding approaches share with ours the objective, i.e. the extraction of some form of knowledge. However, the knowledge derived by them consists simply in keyword patterns. Furthermore, the techniques they leverage to carry out this task are different from ours, especially if we consider the sub-family operating on RDF data. A higher similarity can be found with the other sub-family, i.e., the one operating on graph databases [127, 201].

As for the second family, the corresponding approaches present some analogies, but also some differences, with ours. In particular, both of them exploit metadata to perform the knowledge extraction task. However, the approaches of this second family operate only on structured sources, whereas our approach manages sources with disparate formats.

The approaches of the third family extract knowledge from heterogeneous (both structured and semi-structured) sources. For instance, the approach of [235] aims at querying heterogeneous data in fuzzy XML documents by using a tree-pattern based algorithm. This approach has several differences with respect to ours. In fact, it focuses mainly on querying, whereas our approach considers the extraction of knowledge patterns. Furthermore, it operates on XML documents, whereas our approach operates on sources with different formats. Interestingly, the approach of [235] leverages a tree pattern-based algorithm, whereas ours exploits a graph pattern-based one. Another approach of this family is the one described in [324], which is based on Similarity Join. This approach and ours are similar in that both of them have a step in which a similarity detection task is performed. However, the approach of [324] needs a support knowledge hierarchy, whereas our approach exploits one or more thesauruses. Furthermore, the data sources considered by the approach of [324] are just collections of objects (e.g., documents) and not a variegate data lake, which is the reference data structure for our approach.

The fourth family comprises network-based models and algorithms that exploit network analysis to extract knowledge patterns. One of these approaches is described in [318]. It operates in the context of Music Information Retrieval, which is actually quite far from the scenarios of interest to our approach. However, both this approach and ours share the usage of network to represent available data and of network analysis to extract the desired knowledge. The approach of [318] focuses on non-traditional data sources, and, in this fact, is similar to ours. However, the source typology considered by it has a very specific nature, whereas the ones handled by our approach are numerous and are the most common ones encountered in the reality. Another approach belonging to the last family is the one described in [238]. This approach and ours present some analogies in that both of them use network analysis to extract knowledge of interest. However, the approach of [238] operates on only one kind of databases (e.g., relational ones) and focuses on a very specific topic, i.e., patent and applicant analysis. By contrast, our approach considers heterogeneous data formats and can operate on sources concerning different topics.

Other two approaches of this family that we have mentioned in Section 12.5 are the ones presented in [249] and [205]. [249] proposes a network-based formalism for representing available knowledge. In this formalism, nodes indicate concepts and arcs denote relationships between concepts. This representation coincides with the one adopted by our approach. The main difference between them consists in the fact that the approach of [249] operates only on information organized in structured databases. This fact contributes to perform data investigation and formalization very easily but, on the other hand, it prevents from managing semi-structured and unstructured data. The approach of [205] aims at performing keyword search in a graph to facilitate the identification of sub-graphs. This approach and ours are similar in that both of them are network-based. However, they also present several differences. Indeed, the algorithm underlying the approach of [205] is centered on cliques, whereas the one underlying our approach is based on paths. Furthermore, the approach of [205] focuses on keyword search, and the consequent subgraph identification, whereas ours aims at detecting knowledge patterns.

12.4.2 Performance of our overall approach

In Section 12.2.2, we have seen that the computational complexity of the extraction of complex knowledge patterns is $O(|A| \cdot log|N| \cdot |N_{max}|)$. We have also seen that this complexity can be judged very satisfactory, if we consider the problem to solve.

However, in real data lakes, the number of involved sources is high and so, in principle, $|N|$ (and $|A|$, which is $O(|N|^2)$ could be very high. Nevertheless, in real situations, the number of relationships among attributes and elements is very small and, consequently, the corresponding networks are very sparse. As a consequence, $|A|$ should be very low, if compared with $|N|^2$, and, therefore, we were confident that, in real cases, the performance of our approach should be very good.

To verify this hypothesis we measured the response time of our approach when the number of involved nodes to examine increases; in particular, we measured the response time obtained by considering the theoretical computational complexity and the real response time. Obtained results are reported in Figure 12.8, whereas in Figure 12.9 we propose a "zoom" for those cases that in Figure 12.8 appeared superimposed on the axis of abscissas. In these graphs, in the computation of the theoretical response time, we considered several values of graph density.

From the analysis of these figures, it clearly emerges that, in real cases, the response time of our approach is much smaller than the one determined by the worst case time complexity, even when the network density is low or very low. This fact leads our approach to work very well also in presence of large data lakes, provided that the corresponding networks are sparse or very sparse, which is the general condition that is found in practice. As a consequence, we can conclude that our hypothesis was true and, therefore, that our approach shows a good performance in real scenarios.

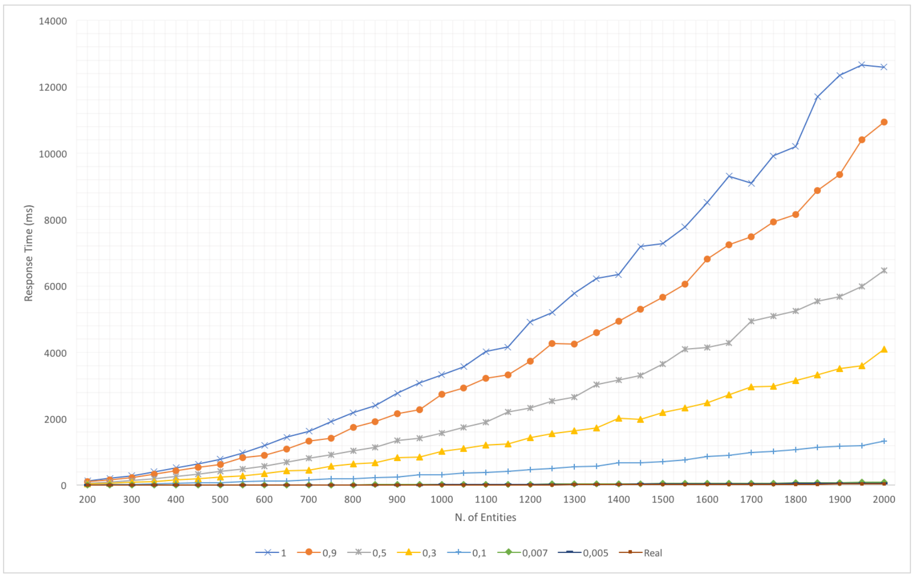


Fig. 12.8: Real and theoretical response time against data lake dimension and density

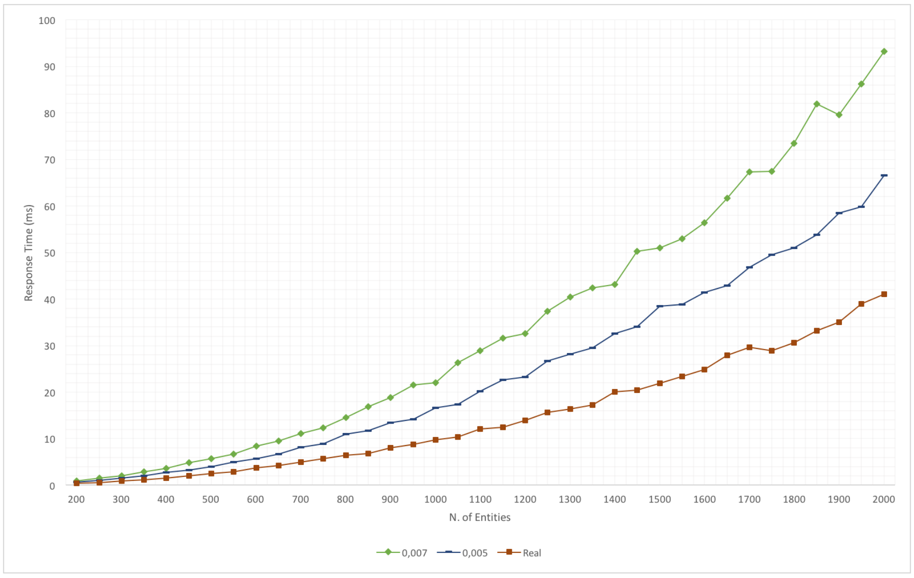


Fig. 12.9: Real and theoretical response time against data lake dimension and density (zoom of Figure 12.8)

12.4.3 Efficiency for large data sets

In Section 12.2.2, we have seen that, from a theoretical point of view, in order to determine the computational complexity of our approach, we must consider two main scenarios, namely:

1. the detected path involves nodes of only one source, in which case the theoretical computational complexity is $O(|A| \cdot log(|N|)$;
2. the detected path involves nodes of more sources, in which case the theoretical computational complexity is $O(|A| \cdot log|N|) \cdot O(max(|N_{max}|, |DL|))$.

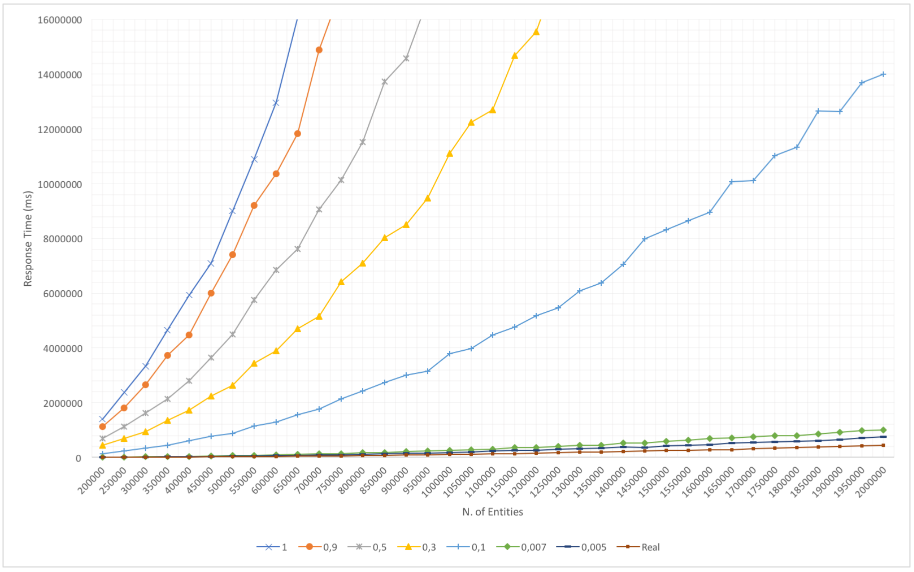


Fig. 12.10: Real and theoretical response time against dimension and density for large data lakes (Scenario 1)

Now, in presence of large data lakes, both $|N_{max}|$ and $|DL|$ are much smaller than $|N|$; as a consequence, from a theoretical point of view, the two cases could be referred to a single one. However, since we aim at measuring the efficiency of our approach in the reality (and not only from a theoretical viewpoint), we prefer to keep the two cases separate and to verify if this hypothesis is also confirmed in practice.

To carry out this experiment, we decided to repeat the tasks already performed in the previous one, but with a data lake having a number of nodes that is three orders of magnitude higher than the maximum one considered in the previous experiment. This number of nodes is clearly much higher than the ones we can currently meet in real situations. However, we preferred to put our approach under stress to see if, even in these extreme cases, it shows an acceptable behaviour. Also in this case, we computed the response time of our approach against the number of nodes of the data lake and compared the response time obtained by considering the theoretical computational complexity against the real response time.

Obtained results are reported in Figure 12.10, for the Scenario 1 mentioned above, and in Figure 12.12, for the Scenario 2 considered previously. A "zoom" of these figures, limited to those cases that appeared superimposed on the axis of abscissas, are reported in Figures 12.11 and 12.13, respectively. From the analysis of these figures we can observe that, in presence of very large data sets, the theoretical response time of our approach would make it not applicable for high values of density. Instead, our approach shows an acceptable response time for low values of density.

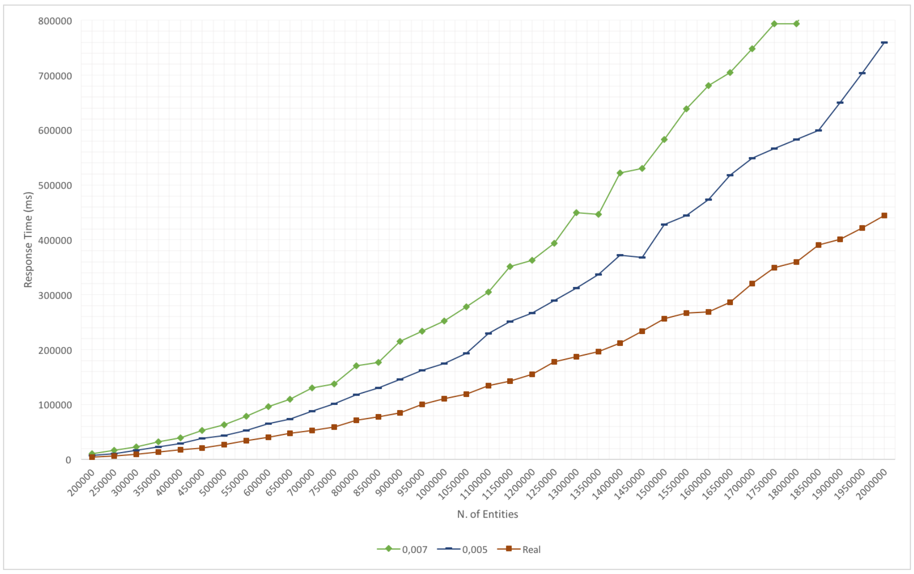


Fig. 12.11: Real and theoretical response time against dimension and density for large data lakes (zoom of Figure 12.10)

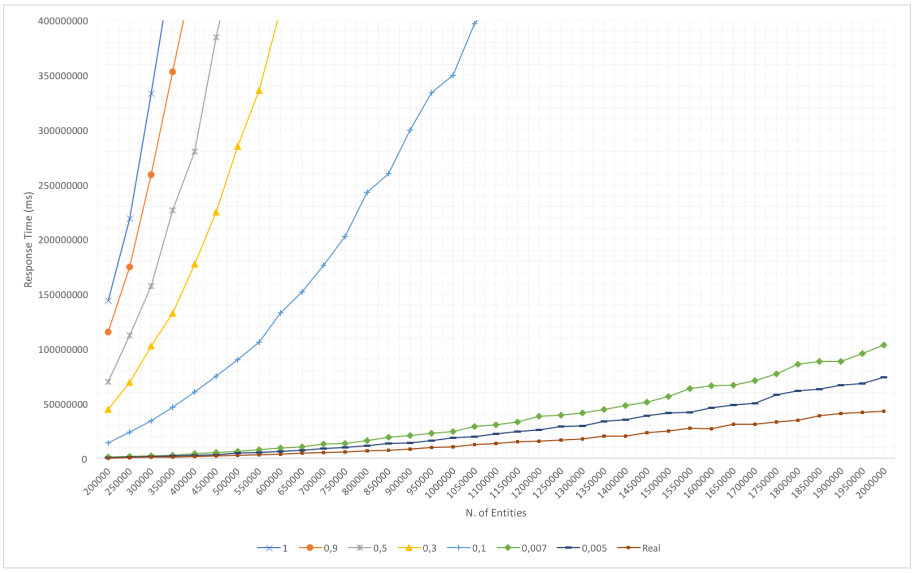


Fig. 12.12: Real and theoretical response time against dimension and density for large data lakes (Scenario 2)

Actually, we have already seen that, in real cases, data lake density is very low. This is also witnessed by the trend of the real response time shown in Figures 12.10 - 12.13, which is even better than the response time derived from the theoretical computational complexity obtained with a small density (i.e., 0.005). Interestingly, the trends of the real response time for Scenarios 1 and 2 are actually the same. The only difference regards the corresponding values that, in case of Scenario 2, are about two orders of magnitude higher than the ones shown in Scenario 1.

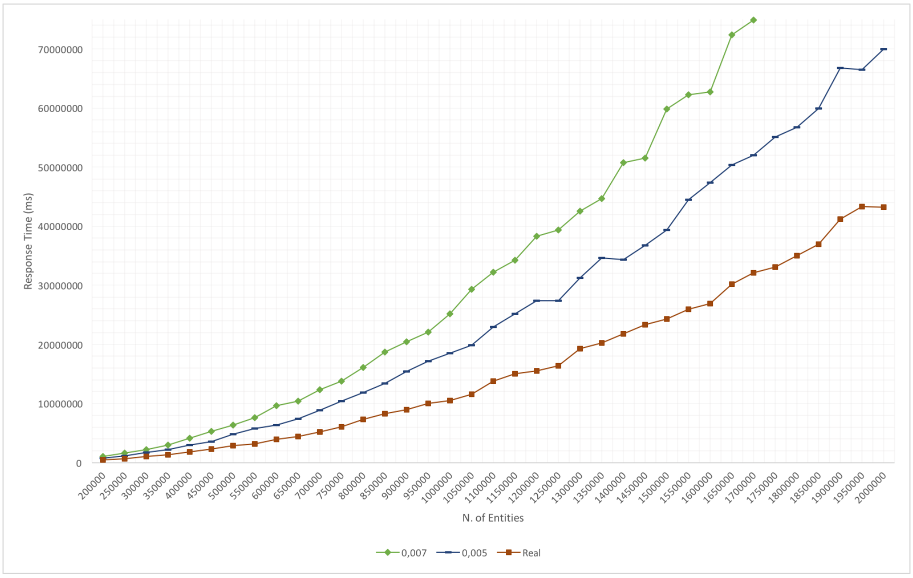


Fig. 12.13: Real and theoretical response time against data lake dimension and density for large data lakes (zoom of Figure 12.12)

All the results described in this section, coupled with the fact that we stressed our approach in extreme cases generally not found in the current reality, lead us to conclude that our approach presents a very good efficiency, which makes it well suited also for large datasets.

12.5 Related work

Today, big data and smart data are considered as two core kinds of data [323]. The former is perceived as unstructured, implicit, messy, heterogeneous and with a huge volume. In the opposite, the latter is presented as structured or semi-structured, explicit, enriched with markups, metadata and annotations. Furthermore, it is characterized by a small volume because creating the enrichments that characterize it needs human intervention. With the passing of time, this apparently strong distinction has gradually narrowed. Indeed, smart data are seen as processed big data [361] or, better, they are considered as available, helpful and processed information. According to this vision, smart data can be just seen as filtered big data on which data analytics techniques can be applied.

Actually, in order to support decision making activities, the capability of extracting information from every kind of data we can store becomes crucial. In this sense, a new way of storing, managing and processing structured, semi-structured and unstructured data is growing. It is resumed in the data lake paradigm [244].

In the literature there is a strong agreement in the definition of data lake. For instance, [181] defines data lakes as "big data repositories which store raw data and provide functionality for on-demand integration with the help of metadata descriptions". [350] resumes a data lake definition provided by [111] and claims that "a data lake is a set of centralized repositories containing vast amounts of raw data (either structured or unstructured), described by metadata, organized into identifiable data sets, and available on demand". Analogously, [264] resumes a definition of [223] and says that "a data lake refers to a massively scalable storage repository that holds a vast amount of raw data in its native format (≪as is≫) until it is needed plus processing systems (engine) that can ingest data without compromising the data structure". Finally, [306] says that "a data lake uses a flat architecture to store data in their raw format. Each data entity in the lake is associated with a unique identifier and a set of extended metadata, and consumers can use purpose-built schemas to query relevant data, which will result in a smaller set of data that can be analyzed to help answer a consumerâĂŹs question". A step forward, but in the same direction, can be found in [250], where the authors introduce the concept of Big Data Lake as "a central location in which users can store all their data in its native form, regardless of its source or format. Big data lake can be used as an environment for the development of in-depth analytics oriented toward fast decision making on the basis of raw data". Clearly, this strong agreement on the data lake definitions does not prevent the possibility to have very different architectures, management approaches and querying techniques in the data lake context, as we will see in the following.

The data lake paradigm requires each raw data to have associated a set of metadata. These represent a key component in the data lake architecture because they let data to be searchable and processed whenever this is necessary [362]. In [148], metadata are also used for bringing quality to a data lake. Here, the authors present CLAMS, a system for discovering integrity constraints from raw data and metadata. To validate obtained results, CLAMS needs human intervention.

In [151], the authors propose a data lake management approach that aims at extracting metadata from the Hive database. To reach its objective, it applies Social Network Analysis based techniques. Instead, in [332], iFuse, a data fusion platform that uses a Bayesian graphical model for both managing and querying a data lake, is proposed.

In the literature, there are many approaches to querying and managing both structured and semi-structured data (see [245, 74, 129, 286], to cite a few of them). However, they are generally incapable of managing unstructured data and are not lightweight and flexible enough to be used in the new data lake context. Furthermore, most of the approaches used for representing unstructured data are limited to texts [212, 316]. In order to address this issue, the authors of [393, 106] propose a generalized data model to represent unstructured data, a method to process it (called RAISE) and an associated SQL-like query language. The authors of [154] propose the usage of machine learning for managing and extracting information from unstructured data. They motivate this proposal by observing that, currently, unstructured data represent about the 80% of stored information and, therefore, they must be necessarily processed with a limited human intervention.

The extraction of Complex Knowledge Patterns (CKPs) is a topic widely investigated in the literature. This is due to the fact that CKPs can refer to a lot of research fields and, therefore, their extraction is a challenging issue in several research areas. Research concerning CKPs goes from keyword search and rank (see [127, 184, 168, 226], just to cite a few approaches) to visual knowledge extraction [313, 108]. In the literature, a huge variety of approaches to extracting CKPs has been proposed. Some of them are based on Network Analysis [387, 257], others are centered on "questions and answers" mechanisms [197], further ones exploit Similarity Join [324], and so forth. Each family of approaches has its pros and cons, as well as its corresponding tools [330].

As for the approaches most related to ours, there are four main families that we need to investigate, namely: *(i)* extraction of keyword patterns; *(ii)* extraction of knowledge from structured sources; *(iii)* extraction of knowledge from heterogeneous sources; *(iv)* extraction of knowledge patterns through network analysis-based approaches.

As far as the first family is concerned, it is necessary to further differentiate the corresponding approaches. A first sub-family focuses on RDF analysis. In this context, several proposals can be found in the literature. For instance, in [127], approaches to keyword search inside RDF data are proposed. These approaches exploit user feedback to relax the search constraints and to identify a higher number of matches. The authors of [391, 184] build a bipartite graph from RDF data and aim at solving a graph assembly problem. Since this problem is NP-hard, they propose two heuristics for facing it. In [267], models letting knowledge patterns to be represented by means of RDF are investigated. The second sub-family, instead, aims at extracting keyword patterns in a graph database. For instance, in [201], the authors represent different relational, HTML and XML sources as graphs and propose a new bidirectional search algorithm for effectively extracting the best answer tree. In [187], the authors propose BLINKS, a system consisting of an algorithm for bi-level indexing and a query processor useful for searching the top-k keywords in a graph. In [168], an engine for enumerating keywords and evaluating their search in a data graph is proposed. In the same way, in [226], EASE, a framework allowing indexing and keyword querying, is described.

As for the second family, most of the corresponding approaches are summarized in [237]. Here, the authors claim that, thanks to metadata, it is possible to think of a new, completely automated, approach.

As for the third family, in [109], the authors provide an overview of techniques used in the literature to support keyword search in structured and semi-structured data. In [235, 201], the authors operate on semi-structured sources and try to make the extraction process as automated as possible. More recent approaches try to extract knowledge from heterogeneous sources. In fact, as evidenced in [234], the big data phenomenon led to the creation of a lot of heterogeneous sources that include unstructured data. These need to be integrated exactly as it was made before for structured data. Starting from this consideration, the authors analyze the most important challenges introduced by this new reality and present a unique query format taking this issue into account. In [313], the authors assert that, in order to have a user-friendly graph query engine, it is necessary to support different kinds of task, like synonymy detection and ontology usage. Based on this assertion, they propose a framework allowing these operations on data without schema or structure. In [324], the authors argue that Similarity Join is a fundamental operation for clearing data and integrating different sources. It involves two big challenges, namely quantifying knowledge aware similarities and identifying similarity pairs efficiently. To address these issues, they propose a new framework. Likewise, in [347], a system to integrate different sources through keyword search, and an evaluation system based on user feedback, are proposed.

The last family of approaches is based on network analysis. In [318], network communities and the apriori algorithm are used to identify rhythmic knowledge patterns of musical work. In [238], the authors represent patent data as a network and, then, propose a new approach that analyzes this network for extracting CKPs about patent applicants. In [249], the authors propose a new formalism to represent a knowledge base through a network whose edges denote the semantic proximity between two or more concepts. This representation allows the discovery of association models among different concepts. In [205], the authors propose an algorithm that uses the cliques in a graph for searching the keywords linked to a given input. According to what the authors claim, keyword search is necessary because it facilitates the identification of sub-graphs in a network.

Part III

Social Networks, Participation and Analysis

Social Networks are universally recognized as a key aspects of everyday life: we can say that they represent the most relevant vehicle of information and sharing of knowledge and ideas, as well as a fundamental means to connect people quicker and easier from different places. Indeed, just think that, by now, even national and international political choices are shared directly by the main actors through online social networks. Obviously, if on one hand social networks are gaining momentum and power in whatever daily situation, we have to be really careful in exploiting and using such technology. In fact, social networks are becoming (if they are not already), a very dangerous place for users because of the exponential growing of fake news, fake profiles and fake information.

In addition, since we are going to migrate into the new context of smart cities, where e-participation and social communities will continue to grow in everyday life, it is necessary to prevent some hostile situations that will continue to arise.

This part of the thesis is divided into two chapters. In Chapter 13 we propose a new decentralized model able to discover and compute the trust level of a participant in the social network so that it is possible to consider it *trusted* or not. This study is in collaboration with the French research group from ENSICAEN University, led by Prof. Rosenberger, that is in charge to enforce this model by using keystroke dynamics as biometric feature. In Chapter 14 we describe and propose a new language that is able to query multiple online social networks with a single interrogation so that it is possible extract precious data from social networks. With respect to existing languages, the proposal also offers the possibility for users to add keywords in the language which reflect the metadata used by social networks to address its data. This feature, which we called *awareness*, enables the possibility to add knowledge into the language

13

An approach to contrast fake identities in Social Networks

Fake identity in social networks is a phenomenon that is strongly increasing, which is used for discovering personal information, identity theft, influencing people, spreading fake news, and so on. We address this problem by introducing the concept of certified social profile and by propagating this property through a collaborative and decentralized approach that exploits keystroke-dynamic-recognition techniques to identify illegal access to certified profiles. We propose a decentralized approach to compute the trust level of a social profile so determining if it can be considered trusted from other users. This work is in collaboration with the French research group from ENSICAEN University, led by Prof. Rosenberger, that is in charge to enforce this model by using biometric features, in detail keystroke dynamics. At the time of writing the thesis this study is in progress.

13.1 Introduction

In daily life, communications are taking rapidly the direction of the digital and the virtual domain. The smart city paradigm also contribute to increasing the number of online interactions. E-participation plays clearly a key role in the everyday operations. We are almost dependent from social media and social networks; they are sources of information sharing, platforms where people can interact each other and where also trash news and data are collected and shared. For this reason, trust represents a fundamental property that users look for when they interact and use their social networks. In particular, we can face fake social network profiles and fake news. Usually, fake news are shared and forwarded mostly by fake profiles. This situation motive us to propose a new model that is able to detect if or not a given social profile can be considered trusted.

Social network profiles whose claimed identity does not match with the real user are certainly potential security threats in the Web [278]. This happens in two cases. The first case is that of fake profiles, in which the owner of a profile intentionally claims the real-life identity of another individual.

The second case is that of violated profiles, in which an intruder, permanently or temporarily, uses the profile of a victim in a fraudulent way.

In both cases, the risk of anomalous behaviour with potential damage of the victim reputation, espionage, or social engineering attacks towards people connected to the victim is very high. To give an example, according to security firm Symantec [7], a growing number of hackers are targeting professionals on LinkedIn. Through these connections, attackers can entice users to give up personal data, hijack them towards infected websites and, once their email addresses is known, launch spear-phishing campaigns.

The problem has thus a high practical relevance. A number of studies have been proposed in the recent literature [170, 116] to contrast this problem. However, all the existing proposals require a strong effort of analysis done centrally by the social network provider, which takes into account all the behavioural and topological information of the profiles.

We offer a different approach based on a collaborative trust mechanism that may operate in principle in a truly distributed fashion, combined with behavioural biometric methods to contrast profile compromising. The originality of our proposal is that it only leverages user-to-user interactions, and no information that only the social network provider can have. Moreover, we adopt a conservative approach, because our goal is to provide assurance that a profile is genuine instead of detecting fake profiles. The underlying idea exploits the social structure of our domain: indeed, the trust model is based on a robust implementation of a *worth of mouth* approach and robustness is obtained by redundancy. In words, we follow the principle that if a sufficient number of people trust the identity of a social network profile, we can trust it too. This way, we obtain a graph of trust, because we propagate trust under the basic assumption that a fake user (and then fake behaviour) is transitively excluded. We base our assumption on the consideration that, when the real-life identity is known, sanctions are facilitated in case of misbehaviour (e.g., victims might sue users who certified the offender), thus misbehaviour is prevented.

Trust is obtained through redundant trust chains in which any node plays a role similar to an intermediate certifier in a certificate chain, until a certified profile is reached. In our model, indeed, the presence of some profiles certified by a Trusted Third Party is also required. In order to identify possible intrusions in a legitimate profile, the trust model takes also into account the behavioural biometric traits of users that they record and verify in a peer-to-peer fashion (i.e., no storing of biometric data is required to the social network provider). In other words, the word of mouth mechanism propagates the information that the current behaviour of a given node is not compliant with that of the initial safe state, thus reducing the trust of the community towards that node. However, the biometric feature is not covered by this thesis, since it is carried out by researchers of the University of Caen, France, led by Prof. Rosenberger. For this reason, in this Chapter we will not give deep details about this component of the solution, but we limit the presentation to the trust graph and the model. Finally, we present some early experiments results that encourage to carry on the investigation.

13.2 Overview of the approach

The reference scenario is that of a social network. The approach works by considering trust chains among users. Each chain starts from a certain root profile, which is a profile previously certified by a Trusted Third Party (TTP). To build a certified profile (root profile), a user has to register to the social network via TTP (also by exchanging identification documents or using a public digital identity system). In this phase, TTP gathers the biometric (behavioural) parameters of the user to create a model that will be exploited, in the future interaction with the user, to verify whether the account is still under this user control. In the negative case, the profile will be no longer classified as certified.

We exploit network analysis and graph theory concepts to represent a social network. Indeed, it is modelled as a directed graph $G = \langle N, E \rangle$, where N is the set of profiles, and E regards friendship among social network profiles. To be general, we use the notion of directed graphs, so that the case of symmetric friendship (as Facebook) is simply handled by including two edges in both directions.

N is thus partitioned into two subsets: (1) the set of certified nodes (denoted by N_c), and (2) the set of non-certified nodes. In this setup phase (1) coincides with the set of root nodes. Any node of the social network (both certified and non-certified) may directly recognize some of its direct contacts. The underlying idea is that a node recognizes only those adjacent nodes for which past real-life interactions occurred, allowing to conclude, also by using external knowledge and information, that the claimed identity is not fake (this typically happens for a significant portion of social network contacts). When a safe interaction occurs (for example, at the first message exchange allowing to recognizes the interlocutor) the profile playing the role of recognizer builds a biometric model of the recognizing node, in order to detect, in the future, the possible presence of an intruder. Importantly, only a node already recognized can play the role of recognizer. The underlying rationale is that the misbehaviour of a user is directly connected to her anonymity in the social network. In other words, making the recognizing process fully accounted and traced (and related to a real-life identity), we can increase the trust about recognized identities, provided that transitively, the process leads to root nodes. As we cannot give an absolute value to the principle above, we have to increase the level of trust by requiring redundancy in the recognizing process, thus making more improbable the conjunct misbehaviour of identified recognizers. The level of redundancy sets the level of trust. The biometric model built by any participant, allows us to detect possible profile compromising, thus including in the trust also the expectation that an initially identified profile is still under the exclusive control of the legitimate owner. It is worth noting that, in principle, the biometric model could be learned by means of multiple channels (social network interactions, chats, shared editing, and so on) by associating the model to the asserted identity.

Before giving into detail, we remark that the proposed approach is not aimed to define a digital identity system, since, as already observed, only a level of trust is obtained and, further, it regards not all identifying data. Indeed, when a user a recognizes a user b, she/he is stating just that b is not claiming an identity not belonging to him, not the veracity of all published information. The number and quality of information needed to a to reach this conclusion depend on the social context. Besides name and surname, they may regard the job, the age, the friends, etc. A detailed study of these aspects is out of the scope of this work whose aim is just to define the basic approach, leaving the detail (also for example the case of multiple profiles of the same user, that of social network profiles managed by more people, etc.) to future work.

13.3 Security requirements

Our proposed approach aims at reinforcing trust in social networks, by ensuring that a node is the one it claims to be. We thus aim to prevent *identity usurpations* (i.e. using an existing identity), *typo-squatting* (using a very similar identity), and *fictitious identities*.

By construction, our proposed approach requires the following security properties in order to be effective. Accountability prevents nodes from issuing certified profiles for nodes they do not recognize and trust (e.g. in exchange of money). Revocation enables a node to revoke a mistakenly issued certificates, and to revoke a certificate in case of a behaviour change from the certified node. Uniqueness prevents a same user from having multiple certifier nodes enabling it to issue several certificates for a same certified node.

- *Integrity:* one should not be able to forge or modify certificates in the name of another certifier.

- *Performance:* the computation of the T-confirmed nodes should be computed in a reasonable time and memory occupation.
- *Availability:* the computation of the T-confirmed nodes should be possible even if some nodes are unavailable or corrupted/malicious.
- *Accountability:* nodes should be held accountable of the certificates they issue, implying non-repudiation of the issued certificates.
- *Revocation:* nodes should be able to revoke the certificates they issued.
- *Uniqueness:* one should not have more of one certifier node.

Our proposed approach should be protected from (or at least mitigate) the following well-known reputation attacks:

- *Collusion:* system should be resilient to collusions.
- *Sybil attacks:* see uniqueness.
- *Slanders:* malicious nodes should not be able to remove a node confirmation.
- *Whitewashing:* a misbehaving nodes should not be able to create a new account to start with a new reputation.
- Identity usurpation, typo-squatting, and fictitious identities.
- *Hacking:* a hacked node should not be able to issues or revoke certificates, and should no longer be considered as T-confirmed. As possible, previously issued certificates should remain valid.

It is worth remarking that this study is in progress in the moment of writing the thesis. A crucial point of our proposal regards the possibility to revoke certificates. Indeed, we need to find a countermeasure against the possibility that a social profile that we previously trusted has been compromised or stolen by a malicious attackers. We are working at two different levels of solutions that have to be seen as complementary and not alternative to each other. The first step is to add the attribute *expiration date* to certificates and the second one is carried out when it is required to revoke the other's social user certificate immediately because of an attack or a compromise of the private key. Similarly to what happens in TLS/SSL, the TTP is in charge of storing these revoked certificates on a given list. We also require our approach to respect user privacy, thus to grant them their *anonymity*: one should not know who certify who.

The biometric system used in our approach requires the following security properties:

- *Confidentiality:* the biometric data should not be disclosed to others.
- *Non-reversibility:* the biometric template should not enable to retrieve the biometrics data.
- *Discriminant:* the biometrics should discriminate users.

- *Constant:* the biometrics should remain constant in order to be used.
- *Non-usurpation:* another user should not be able to forge/imitate the biometric of another.
- *Not-costly:* in terms of memory/time/ergonomics/money (no additional devices).

13.4 The Trust Graph

In this section, we describe how our trust model works. Throughout this section consider given a directed graph $G = \langle N, E \rangle$ representing a social network and a *redundancy* parameter t, i.e., a positive integer representing a level of trust. Let TTP be a Trusted Third Party. Let denote by N_c the set of *certified nodes*, that is the nodes whose identity is assured and monitored by TTP. Given a node $u \in N$ we denote by $\Gamma(u)$ the set of neighbours of u (i.e., adjacent nodes). In the social network domain taken into consideration, two nodes are adjacent if they are friends on the social network. Moreover, we denote by $R(u) \subseteq \Gamma(u)$ the set of nodes recognized by u.

Definition 13.1. *We say that a node $u \in N$ is* t-recognized *(*in $A \subseteq N$*) if either: (i) $u \in N_c$ (i.e., is a certified node), or (ii) there exist t other t-recognized nodes in A that recognize u.*

When the set A of the definition above is not specified, we intend that a node is t-recognized in N. From the above definition it immediately follows that nodes in N_c are t-recognized for any t and in any set A. We define now the notion of *t-closed* set.

Definition 13.2. *A set $A \subseteq N$ of t-recognized nodes in A is said* t-closed, *if there is no $u \in N \setminus A$ that is t-recognized in A too.*

From the above definition it immediately follows that all certified nodes must belong to any t-closed set.

Theorem 13.3. *For any t-closed set A, it holds that $N_c \subseteq A$.*

With the next theorem we state that the operator $\subseteq$ induces a partial order over the set of t-closed sets, which is a lower semi-lattice. First, we define this set.

Definition 13.4. *We denote by $N^t \subseteq 2^N$ the set of non-empty t-closed subsets of N.*

Now, we are ready to state the following theorem.

Theorem 13.5. *N^t is a lower semi-lattice.*

Let denote by N_b^t the bottom of the semi-lattice N^t. In our model, the role of N_b^t is central, because it includes exactly all nodes that are t-recognized, but, due to subset minimality, they do not form clusters whose recognizing is only mutual. In other words, N_b^t is the set of nodes for which trust paths start from certified nodes. For this reason, we use N_b^t to trust nodes.

Definition 13.6. *Given a node $u \in N$ we say that u is* t-trusted (in N) *if $u \in N_b^t$. N_b^t is also said the* set of t-trusted nodes (in N).

To formalize the relationship of t-trustworthiness of a node with the presence of certified nodes supporting the trust, we introduce the notion of *support* and *kernel* of a t-trusted node in N_b^t.

Definition 13.7. *A* support *for a node $u \in N_b^t \setminus N_c$ is any subset $S_u^t \subseteq N_c$ such that u is t-trusted also in the transformation of G obtained by restricting the set of certified nodes to S_u^t. A* kernel *K_u^t for u is any subset minimal support for u.*

The next theorem states in which terms we intend the level of trust represented by t-trustworthiness. Informally, being t-trusted for a node means that there are at least t trust chains starting from certified nodes.

Theorem 13.8. *Given a node $u \in N_b^t \setminus N_c$, any kernel K_u^t for u is such that $|K_u^t| \geq t$.*

The above definition of N_b^t and, consequently, of t-trustworthiness of a node, is declarative, so it does not give us any information about how to compute if a node is t-trusted or not. Thus, we provide an operational definition of N_b^t, based on the fixpoint of a monotone operator Λ_t, called *t-recognizing operator*. This definition also gives us a more intuitive support about the property stated earlier, for which the trust of nodes in N_b^t can be directly or indirectly linked to (at least) t certified nodes.

Definition 13.9. *We define the* t-recognizing operator $\Lambda_t : 2^N \to 2^N$ *as follows: (i) $\Lambda_t(\emptyset) = N_c$ (ii) $\Lambda_t(A) = \{u \in N \mid \exists B \subseteq A \text{ s. t. } |B| \geq t \wedge u \in \bigcap_{v \in B} R(v)\}$.*

Now, we define the following sequence of sets: $\Lambda_t^0 = \Lambda_t(\emptyset)$; $\Lambda_t^k = \Lambda_t(\Lambda_t^{k-1})$, for any $k > 0$.

By proving first that the operator is monotone, we can obtain the following results:

Theorem 13.10. *The operator Λ_t has a fixpoint, i.e., there exists $k > 0$ such that $\Lambda_t^k = \Lambda_t^{k-1}$. We denote this fixpoint as Λ_t^∞.*

The next theorem states the equivalence between the declarative definition above and the operational one.

Theorem 13.11. *The set of t-trusted nodes N_b^t coincides with the fixpoint of the t-consequence operator Λ_t^∞.*

The above notion of t-trustworthiness embeds a lossless propagation of trust, in which the level of assurance of identity based on recognition of users, does not degrade if the t redundancy property holds at every step of propagation. In other words, the t-redundancy property is considered as a threshold to propagate the trust. The t-redundancy parameter implicitly represents the assumption that the multiple identification of a node u done by nodes in turn identified with the same trust level, and so on, until t certified nodes are reached, can be considered sufficient to trust the identity of u. The approach applies the concept of trust chain used in the context of digital certification to the domain of identity management in social networks, with the aim of contrasting the problem of fake identities. It is worth remarking that the model cannot provide absolute guarantees, but only a trust level directly connected with the value t. The higher t, the higher the trust about identities.

So far, the trust model assumes that, once a user has recognized another user, no revision of this information must be done. This assumption would be valid only in absence of attacks able to give the attacker the access to the user profile (even temporarily). So we assume a sort of *safe state* with regards to fraudulent accesses. In other words, the trust model above prevents from the risk of fake profiles and fake identities but not from fraudulent access to legitimate profiles.

To contrast this further case, we introduce a biometric-based reinforcement to combine with the above trust-chain mechanism, in order to decrease the trust on a given subject if the biometric trait is not recognizable and thus managing also non-safe states. Indeed, the full trust in our mechanism is obtained by relying on the assumption that the disclosure of trusted real-life identities prevents from misbehaviour of users in the trust mechanism itself, under the t-redundancy assumption.

But, if the operating user is not the legitimate one, the above assumption fails, so the identity of those users whose trust is based on paths involving the potentially attacked profile should be not fully trusted. In other words, to take into account this aspect, we have to enable a gradual level of trust, from 0 to 1 (while before the trust was basically either 0 or 1), and use an ϵ-approximation approach to trust identities. The first step is to modify the notion of t-recognized. Obviously, we keep the redundancy parameter t in the new definition, but we introduce the possibility that a user is not fully identified in a given moment, due to the fact that the biometric support is giving a warning rate. We require that nodes in N_c (i.e., certified nodes) loose their state if the biometric support gives a warning rate. Thus, we can assume that certified nodes are not attacked. Given a node u, we define the set $R^\epsilon(u)$ (where $0 \leq \epsilon \leq 1$) as the set of pairs $\langle v, b_r(v) \rangle$ such that $v \in R(u)$ (i.e., v is a node recognized by u in the safe state) and $1 - \epsilon \leq b_r(v) \leq 1$ is the current biometric rate provided that it is higher than a given threshold $0 < 1 - \epsilon \leq 1$ under which v must be currently considered not recognized. Obviously, for $\epsilon = 0$ we fall in the safe state. We say that nodes in $R^{\epsilon}(u)$ are *ϵ-recognized* by u. At this point, we are ready to extend the notion of t-recognized to a non-safe state.

Definition 13.12. *We say that a node $u \in N$ is $\langle t, \epsilon, r \rangle$-*recognized (in N) *if either: (1) $u \in N_c$ (i.e., is a certified node), or (2) there exists a set B of $\langle t, \epsilon, r \rangle$-recognized nodes such that both (i) $u \notin B$, (ii) $|B| \geq t$, (iii) $u \in R^{\epsilon}(v)$, for each $v \in B$, (iv) $1 - \epsilon \leq r \leq 1$, and (iv) $\frac{\sum_{v \in R} b_r(v)}{|B|} \geq r$.*

It is easy to see that a node is t-recognized, according to Definition 13.1, if and only if it is $\langle t, 0, 1 \rangle$-recognized, according to the above definition. The intended meaning of Definition 13.12 is to take into account warnings triggered by the biometric support (through the parameter ϵ), and, at the same time, to require by means of the parameter r that a possible fault of trust introduced by ϵ can be partially recovered by fortifying redundancy in order to reduce approximation. In words, if we can trust less nodes because we are not sure they are not attacked we need a larger set of witnesses to reach a safe conclusion anyway. This means that r modulates the level of assurance of trust, so that the higher r, the higher the trust on the identity of $\langle t, \epsilon, r \rangle$-recognized nodes. Actually, to talk about trust we have to avoid mutual self-sustained cluster of $\langle t, \epsilon, r \rangle$-recognized nodes, so we have to proceed as in the safe state above by requiring the minimality condition. For brevity we do not give all detail, but it is rather clear that definitions of N_b^t and, consequently, of t-trustworthiness of a node, can be easily extended to the non-safe case, on the basis of Definition 13.12. We reach thus the definition of $N_b^{\langle t, \epsilon, r \rangle}$ as the bottom of the semi-lattice of subsets of $\langle t, \epsilon, r \rangle$-closed nodes of N. Therefore, a node is $\langle t, \epsilon, r \rangle$-trusted if belongs to the set $N_b^{\langle t, \epsilon, r \rangle}$. Also the definition of *recognizing operator* can be trivially extended so obtaining the operator $\Lambda_{\langle t, \epsilon, r \rangle}$ in such a way that $N_b^{\langle t, \epsilon, r \rangle}$ coincides with the fixpoint $\Lambda^{\infty}_{\langle t, \epsilon, r \rangle}$ of such operator.

13.5 Decentralized Computation of Trust

We present here a decentralized way to compute the trust level. Indeed, as for others approaches presented in this thesis, solutions in which a central node has more power have many issues regarding security, trustworthiness and reliability.

We assume the collaborative behaviour of participants in the network.

We thus propose a distributed algorithm for the evaluation of trust, in which each node send, upon query, a proof that they are t-confirmed. We choose the proof to be a certificate graph, a generalization of certificate chains, in which each node is certified t times, t being the trust level, instead of once.

We assume that each node posses its own pair of asymmetric cryptographic keys, and that the private key is kept secret. The public key is used also as an identifier of a user.

Now, we define the (recursive) structure of a certificate. We have two kinds of certificates:

- intermediate certificates: containing the certified node's public key and the certifier's public key;
- final certificates: containing explicitly the certified node's identity (e.g., URL) and public key, and the certifier's public key.

These certificates are the proof that the certifier node trust the certified node (only the certifier node is able to issue such certificates because they are signed). Intermediate certificates enable the certified node to certify other nodes without revealing its real identity in the certificate graph, and the final certificates enable the certified node to prove that it is t-confirmed by sending a certificate graph and its final certificates.

Specifically, a certificate of a user is composed of:

1. *issuer*: this field contains the public key of the generator of the certificate;
2. *target*: this is the public key of the user to be certified;
3. *profile*: this field is obtained as $url \oplus r$, where url is the URL of the social-network profile of the target, r is a randomly generated bit string, and $\oplus$ denotes the exclusive OR operation;
4. *key*, an optional field containing r (i.e., the value generated above). If this attribute is missing, then this certified is said *intermediate certificate*; otherwise, it is said *final certificate*;
5. *certifier*: this field may contain a set (even empty) of intermediate certificates of the users who previously have certified the issuer. If this field is empty, then the issuer should be a root node certifier and this target user is said *root node*.
6. *expiration date*: this field may contain the date until the certificate can be considered as valid.

Moreover, the fields 1, 2, 3, 5 and 6 are signed by the issuer's private key to guarantee information integrity.

Let assume that a user u is friend, in the social network domain, with a set of $u_1, ..., u_p$ t-confirmed profiles. When u wants to be t-confirmed, u asks to her/his social friends t-confirmed for a final certificate: thus, the i-th friend u_i of u signs and issues a certificate c_i having all fields filled-in (i.e., containing also the random r). At the end of this step, if u has obtained at least t final and valid certificates, then u becomes t-confirmed as well. Now, u can actively participate in the solution by propagating the trust to another friend f, by issuing and signing a new certificate of f, in which the field 5 (i.e., certifier) is composed of the certificates c_i^* with $1 \leq i \leq t$, where c_i^* is obtained from c_i by cancelling the field 4 (i.e., key).

Concerning the validation of certificates we have two cases. A intermediate certificate is valid, if and only if:

- it is signed by the issuer's private key (i.e., the signature is valid);
- if the field certifier is empty, then the issuer must be a root node certifier; otherwise, this field contains at least t (recursively) valid certificates;

A final certificate is valid, if and only if, in addition to the above conditions, it holds also that the value of the field profile $\oplus$ the value of the field key is equal to the URL of the social-network profile of the user to be certified.

13.5.1 Optimization

A second implementation of the distributed evaluation is to let each certificates to be independent, i.e. without containing the certificates that certify it, and to ensure that the certificate graph does not contain duplicate certificates.

When being certified, the intermediate certificate is added to inters, and the final certificate to finals. The other certificates included in the received certificate graph are added to a third internal structure (graph), a map in which each element keys' is the pair (certified public key ; certifier public key), thus ensuring the uniqueness of each certificates. When issuing a new certificate, the node send, with the certificate, graph, and either finals or inters, depending if issuing a final or an intermediate certificate. Thus sending a valid certificate graph.

To verify such certificate graph, the certificate graph's certificates are placed in a multi-map whose index is the certified public key, and containing t elements. For each index, the number of certificates, and the certificates are verified. Then, starting from the final certificates, the absence of loops is ensured by a deep-first search, if a visited node already belong to the current path, a loop is detected.

13.6 Datasets

Let first describe the datasets we used to carry our our experiments. To have a better proof of our algorithm, we used different datasets downloaded from the website `http://networkrepository.com/` [311]. In particular, we chose Facebook networks

	$\|N\|$	$\|E\|$	d_{max}	d_{avg}
$D1$	24k	1M	3k	96
$D2$	36k	2M	5k	87
$D3$	64k	1M	2K	39
$D4$	63k	817k	1k	25
$D5$	42k	1M	4k	65
$D6$	150K	10M	5k	100

Table 13.1: Properties of the datasets used

since they perfectly fit with our scenario. We generally prefer to use these datasets with respect to synthetic ones because they are real data so we can compare our algorithms over a real scenario.

In the Facebook section of the repository there are several datasets, each capturing a certain portion of the complete one, and each having some characteristics with respect to others. Indeed, we can find datasets with a huge amount of nodes and arcs but with a average degree quite low, or datasets where nodes are not huge in number but other properties are higher, and so on. For these reason we selected five different datasets where properties can cover heterogeneous sub-scenarios inside the big one. However, since these datasets reach no more than 2M edges I generated also a synthetic dataset, called $D6$, to test our approach over a dataset having 10^7 edges.

In Table 13.1 we give some details about these four datasets. In particular, we denote with $|N|$ the number of nodes involved in the dataset, $|E|$ is the number of edges, d_{max} is the maximum degree of a node in the given dataset while d_{avg} is the average degree of the nodes. Each dataset is represented in the edge-list format.

By analysing these numbers, we can say that $D1$ and $D2$ are the most connected datasets because they have high degrees and they have many edges. Moreover, $D3$ and $D4$ have more nodes while they have less edges and lower degrees, meaning that they are less connected and more scattered; instead, $D5$ is the trade-off. Further information about the characteristics and properties of the datasets can be found on the $NetworkRepository$ website.

13.7 Implementation issues

In this section we explain some issues related to the implementation process of our model.

As said before, investigations and experiments are still in progress at the writing of this thesis. For this reason we introduce, in this section, just preliminary information and results obtained by these early tests.

In particular, we first focus the attention trying to understand how our model and algorithm would work in an *ideal* world, even if the datasets used are real and not from synthetic-way of generation.

We implemented our simulator in Python 3.6, using a computer with 8GB of RAM and a $i5$ Intel processor, running $MacOS$ as operating system.

Datasets are in the format of edgelist, which is data structure used to represent a graph as a list of its edges. For example, if between nodes 1 and 2 there exists an arc, in the edgelist we will find a row containing the node pair 12.

So, first we read the edgelists downloaded from the internet so that we have all the information about nodes and relationships between peers of the network. After this operation we proceed by selecting root nodes. We could have implemented many different algorithms for this selection (for instance, we could have used different centrality measures to select best nodes); anyway we decided to select randomly roots among all the nodes since it would be fairer and more representative of a real situation. The only constraint we require is that a node, to be chosen as root, should have a number of relationships at least equal to the `t_level` indicated in the simulation.

If we think that we used `t_levels` that are in the order of magnitude of about ten, we can conclude that this condition is considerable as acceptable because we can declare that every real social profile easily reach ten friends.

Moreover, we used this condition in the root selection step because otherwise there would have been some root nodes that they would not have been a relevant power in the trust propagation in the network.

13.7.1 Experimental results

In this section we give and discuss results returned by early simulations.

The first experiment is devoted to the study of the performance of our approach when no attack is performed and where participants are always collaborative in trusting a friend node. The first parameter we wanted to control and check is the *coverage* that potentially could be reached by running our algorithm. As we explained above, we operate with five different datasets, so we expect to see some differences in results. Clearly, the more the network is dense and connected the higher we expect the coverage to be. However, this easy prediction does not invalidates our algorithm because it is plausible that different portions of the social network have different behaviour and characteristics.

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	95,9	90	83,9	72,3
roots = 1%	96,5	91,7	86,7	80,5
roots = 2%	96,9	92,6	89	85,1
roots = 5%	97,2	93,5	90,8	88,1
roots = 10%	97,5	94,5	92,1	89,9

Table 13.2: Percentage of coverage in $D1$ as t_level and number of roots vary

We carry out our experiments by varying the following parameters:

- t_level;
- number of roots.

We use $t\_level = \{3, 6, 8, 10\}$ and $numberofroots = \{0.1\%, 1\%, 2\%, 5\%, 10\%\}$ (with respect to the total number of nodes of the given network). Furthermore, since we select random roots, we run every combination of these two parameters fifty times and we averaged results so that avoiding outlier cases.

In Tables 13.2, 13.3, 13.4, 13.5, 13.6, 13.7 we report coverage results of these simulations for each dataset.

As expected, datasets more connected reached higher percentages of coverage. It is interesting underline that, in three out of five, the coverage reached is quite relevant with very just the 0.5% of roots selected ($D1, D2, D5$) and only in those two scenarios with few connections ($D3$ and $D4$) this little number of roots is not able to reach a relevant number of nodes If we briefly analyze those two datasets we can see that the average degree of a node is very low (resp. 39 and 25). This means that every node has, in average, thirty-nine and twenty-five friends. Another analysis that we can obtain by results in that

We remark that these results have been obtained by running, for each configuration fifty times, only one cycle of the trust operation. To be more clear, we can say that in the simulation every node asks for being certificated only once. In the real scenario, instead, we can easily think and admit that a participant can ask more than once to its friends to be recognized, so, in this sense, our experiment underestimate the real possible coverage.

However, these first results encourage us to continue to investigate and study other performances parameters, underling that the continuous validation of trust is obtained by the enforcement of biometric features. Simply speaking, if a trusted profile does not continue to maintain homogeneity of keystroke dynamics its level of trust is downgraded.

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	95,5	87,5	80,2	71,6
roots = 1%	96	90	84,7	77,3
roots = 2%	96,2	91,29	87,2	82,6
roots = 5%	96,5	93,5	89,19	85,5
roots = 10%	97	94,5	90,6	87,7

Table 13.3: Percentage of coverage in $D2$ as t_level and number of roots vary

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	69,9	44,4	6,6	4,6
roots = 1%	70,5	47	35,2	23,7
roots = 2%	71	49,4	38,9	29,6
roots = 5%	72	52	43,4	36,7
roots = 10%	73,5	55,1	47	41

Table 13.4: Percentage of coverage in $D3$ as t_level and number of roots vary

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	70,1	44,7	8,9	5,7
roots = 1%	71,2	47,1	34,4	20,2
roots = 2%	72	49,3	38,2	29,9
roots = 5%	73	52,4	43,5	36,4
roots = 10%	74,5	55,7	47,8	41,8

Table 13.5: Percentage of coverage in $D4$ as t_level and number of roots vary

13.8 Related work

In the Online Social Networks (OSNs), the detection of fake profiles is becoming every day more and more important because there are many threats, like scamming, trolling, phishing, sybil attacks, social bots and so on that need to be faced [353, 305, 373].

Clearly, in this scenario, it can be very helpful to create a model which includes the dynamic computation of a trust degree, named "social trust", for each user. Indeed, trust is becoming a fundamental element of a successful social network [328] and it derives from the "social capital", which is based on the density of interactions among people and which refers to a collective resource for the cooperation of them [86, 273].

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	93,1	82,8	75,1	64,4
roots = 1%	93,8	84,3	77,4	68,9
roots = 2%	94,3	86	79,7	72,7
roots = 5%	94,9	87,9	83,2	77,8
roots = 10%	95,4	89,5	85,6	81,4

Table 13.6: Percentage of coverage in $D5$ as t_level and number of roots vary

	$t = 3$	$t = 6$	$t = 8$	$t = 10$
roots = 0.5%	91,2	82,2	74,5	63,8
roots = 1%	92,1	84	76,7	67,9
roots = 2%	93,7	85,4	79,8	71,9
roots = 5%	94,4	87,1	83,3	77,1
roots = 10%	95,1	88,5	85,1	80,4

Table 13.7: Percentage of coverage in $D6$ as t_level and number of roots vary

In [328], the authors present a survey of trust in social networks. In particular, they distinguish three categories of trust models: *(i) graph-based*, which consider only how members are related to each other and does not consider the real interaction between them, *(ii) interaction-based*, which consider only interaction in the community and ignore the social network structure, and *(iii) hybrid*, which tries to consider both the aspects to compute the social trust.

The authors of [228] use two typical features of the OSNs, like the number of friends and the contact frequency, for deriving two factors: namely *Degree* and *Contact Interval*. The combination of these values produces the T-OSN, a trust evaluation model for social networks in which these two factors are known and available for the social network itself.

In [273], the authors propose *STrust*, a trust model for social networks based on users and interaction among themselves. In particular, they compute the social trust degree as a combination of two values, namely “Popularity Trust” and “Engagement Trust”. The former is based on the trustworthiness of a member in the community, while the latter is based on how much an actor trusts other actors.

One of the most crucial points of this research area is, surely, how trust propagates within the network. For this purpose, in [272] the authors enhanced *STrust* by proposing an association based trust propagation model which considers three different factors: *(i) the density of interactions*, *(ii)*, *the degree of separation* and *(iii) the decay of influence*.

Another idea for the propagation of social trust in OSNs is presented in [191], where the authors propose a PKI trust model using certificated chains. In particular, they start from the assumption that trust propagates in trust networks, which are defined as particular sub-graphs of the whole graph representing the social network, and they calculate the trust degree by applying two different kinds of aggregation: *(i) sequence aggregation*, which describes the aggregation of the trust degree along a trust path and *(ii) parallel aggregation*, which considers how aggregate trust degrees in multiple parallel trust paths. However, this assumption does not fit completely with our scenario because we do not know a priori the trust network and we do not want to disclose this kind of information for security reasons as well.

In [376], it is presented a model based on the uniform trust propagation called SN-GDM (Social Network based - Group Decision Making). In particular, the authors propose the two concepts of *Trust Score (TS)* and *Knowledge Degree (KD)*, which are combined in order to define a social trust value that does not lose any trust information during its propagation. Their model requires the intervention of Trusted Third Parts (TTPs) for the validation of results.

A decentralized and privacy-preserving OSN is presented in [120]. Here, authors propose *Safebook*, a system which provides registered users with data storage and data management functions relying on trust relationships that are part of social networks in real life. A similar system is presented in [61]. In this platform, users are associated with public keys they exchange out of band while creating OSN links, and data confidentiality and privacy are ensured through encryption. These systems do not protect from identity theft on the original OSN, but their aim is to provide a tool to anonymously communicate through hop-by-hop encryption among trusted users.

Our approach is also related to the concept of information diffusion in OSNs, in the sense that the roots influence the trust values of other nodes in the network following the rules of OSN information flow [64, 292]. Most of these works study how information flows in OSNs and propose strategies to maximize this diffusion by identifying strategical nodes for the information propagation. The aim of our paper is somehow orthogonal to these studies and may exploit these solutions to improve trust propagation through the OSN.

13.9 Conclusion and perspectives

We proposed a collaborative approach based on user-to-user interaction and keystroke dynamics to trust identities in a social network. The peculiarity of our method is that it only relies on the view a user has of the neighbourhood combined with real-life background information and trust propagation. We tested our method on a combination of real-life and synthetic data, by obtaining promising results (a good coverage with few certified nodes and a good resilience in case of attacks). The next step of our research will be to complete the theoretical characterization of the model, to deal with some detail about user recognition, to deepen the experimental analysis, and to deal with implementation issues. As for this last, we will investigate the performance of our model when attacks occur: specifically, we study the strength of trust relationships when some nodes are compromised and how our decentralized model reacts to these situations.

14

A novel query language for data extraction from multiple social networks

Online Social Networks (OSNs) represent an important source of information since they manage a huge amount of data that can be used in many different contexts. Moreover, many people create and manage more than one social profile in the different available OSNs. The combination and the extraction of the set of data from contained in OSNs can produce a huge amount of additional information regarding both a single person and the overall society. Consequently, the data extraction from multiple social networks is a topic of growing interest. There are many techniques and technologies for data extraction from a single OSN, but there is a lack of simple query languages which can be used by programmers to retrieve data, correlate resources and integrate results from multiple OSNs. This work describes a novel query language for data extraction from multiple OSNs and the related supporting tool to edit and validate queries. With respect to existing languages, the designed language is general enough to include the variety of resources managed by the different OSNs. Moreover, thanks to the support of the editing environment, the language syntax can be customised by programmers to express searching criteria that are specific for a social network.

14.1 Introduction

Over the past decade, Online Social Networks (OSNs) have become widely used by a large set of users in everyday life. Most people create and manage profile in one or more OSNs like Facebook, Twitter, LinkedIn, Instagram. Many people spend a lot of time creating many contents on every-day life, latest news, politics, economics, sport, and publish many information (posts, photos, videos, etc.) about their passions, travels, friends. As a consequence, OSNs represent a source of a huge amount of data, offering a great variety of information spanning from personal information to users interests. For this reason, OSNs are recognised as an important phenomenon from a social and economic point of view, and, thus, it is matter of study for the design and the development of innovative and modern web applications. In many cases, both personal information and social interactions coming from social network profiles can be part of innovative software solutions. Among these, social Web applications are the most significant example in which both peoples' identities and contents they produced are involved in the business process and data are mostly owned by users, strongly interlinked and inherently polymorphic [72].

To allow the data extraction, some of the social network providers offer different APIs and framework that reflect their internal metadata. In addition, some other tools and well-known techniques (e.g., web scraping) have been proposed in literature with the aim of extracting data from OSNs. Hence, when a developer needs to interact with different OSNs, she/he needs to manage different techinques and languages. In fact, there is a lack of standard languages and protocols to query both heterogeneous OSNs and multiple OSNs with the same language. In this direction, this study proposes a novel query language for the extraction of data from multiple OSNs. The proposed language is, according to Martin Glinz's insightful remarks [167], "as simple as possible and as rich as needed", in the sense that it provides simple keywords and powerful mechanisms to allow writing queries. Moreover, this language offers the possibility to be customized by its users in order to add keywords that better reflect the metadata used by the single OSN.

14.2 The basic data model

Each Online Social Network has been designed for a particular (main) purpose: for example, Facebook for keeping in touch with friends, Twitter for microblogging, LinkedIn is business-oriented, Instagram for sharing pictures. However, all social networks have some common properties on which they are built: some examples are (1) user's profile, which stores personal data of the user, (2) the concept of relationship, which is necessary to build the connections among users, (3) the presence of shared resources, which can be short texts, Web links, images, videos, and so on, and (4) actions that can be done from users, such as appreciating, commenting, sharing of resources. It is worth noting that social networks usually implement in a different way (or use a different name for) similar properties: for example, the friendship is symmetric in Facebook, whereas it is asymmetric in Twitter, it is built by the concept of *circle* in Google+, it is called *connection* in LinkedIn whereas in Twitter the terms *followers* and *following* are adopted. Again, appreciation is implemented by *like* in Facebook, *endorsement* in LinkedIn, *+1* in Google+ (recently closed).

From these examples, it is clear that, in spite of the intrinsic similarity of social networks, each one defines its own terminology, thus making social network depending on the reference to an entity when we want to write a high-level language for querying social network data: for example, to refer to the contacts of a social network user, we should ask for *connections* if the user is on LinkedIn and *followers/following* if the user is on Twitter. Consequently, the need to have an ontology describing and integrating the characteristics of social networks arises.

In this section, we solve this problem by presenting the ontology used to model social network concepts at an abstract level. The data model presented in this section is general enough and it is independent of the specific social network.

Among the several approaches proposed in the literature to represent social networks, we used an ontology based on the model presented in [93], which has been designed with the goal of integrating information coming from different social networks.

Social network data are modeled by a direct graph $G = \langle N, A \rangle$, in which the set of nodes is $N = P \cup R \cup B$, with $P \cap R \cap B = \{\}$, and the set of arcs is $A = F \cup M \cup Pu \cup S \cup T \cup Re \cup L \cup Co$ with $F \cap M \cap Pu \cap S \cap T \cap Re \cap L \cap Co = \{\}$. Now, let's define the sets introduced above.

Each element of the set P represents the *profile* of a user and consists in the tuple $\langle$`url, socialNetwork, screen-name, [personalInformation], [picture]`$\rangle$, where `url` is the Web address that identifies and localizes the profile, `socialNetwork` is the commercial name of the Online Social Network which the profile belongs to (the same value is shared by profiles in the same social network), `screen-name` is the name chosen by the user who registered the profile to appear in the home-page of the profile or when posting a resource, and, finally, `personalInformation` and `picture` are the information and the personal image which the user inserted as related to the profile.

The set R models *resources* of the Web or created by users. A resource is represented by a tuple $\langle$`url, type, [description], [date]`$\rangle$, where `url` is the Web address to access the resource, `type` indicates the type of the resource content, and finally, `description` and `date` represent the string inserted by who published the resource and the publishing date, respectively.

The set B models *bundles*, a set of resources handled simultaneously by a user, and represented by a tuple $\langle$`uri, [description], [date]`$\rangle$, where `uri` is the identifier of the bundle, `description` is the string chosen by the user to be shown with those resources and, finally, `date` represents the publishing date.

The *follow* arcs set $F \subseteq A = \{p_s, p_t \mid p_s, p_t \in P\}$ models the fact that in the (source) profile p_s, it has been declared a certain type of relationship towards the (target) profile p_t.

The *me* arcs set $M \subseteq A = \{p_s, p_t \mid p_s, p_t \in P\}$ denotes that the user with profile p_s has declared in this profile to have a second profile p_t.

The *publishing* arcs set $Pu \subseteq A = \{p_s, b_t \mid p_s \in P, b_t \in B\}$ indicates that the user with profile p_s has published in this profile a bundle b_t.

The *shared* arcs set $S \subseteq A = \{b_s, b_t \mid b_s, b_t \in B\}$ specifies that the bundle b_s (published by a user) is derived from an already published bundle b_t.

The *tagging* arcs set $T \subseteq A = \{p_s, br_t, w \mid p_s \in P, br_t \in B \cup R$ and w is a word$\}$, denotes that the user with profile p_s assigned the word w to describe a bundle or a resource br.

The *referencing* arcs set $Re \subseteq A = \{b_s, p_t \mid b_s \in B, p_t \in P\}$ models the fact that a bundle b_s includes a reference to the profile p_t.

The *like* arcs set $L \subseteq A = \{p_s, pbr_t \mid p_s \in P, pbr_t \in B \cup R \cup P\}$ describes the information that a user with the profile p_s expressed a preference/appreciation for a bundle, a resource or another user profile pbr_t.

The *containing* arcs set $Co \subseteq A = \{b_s, r_t \mid b_s \in B, r_t \in R\}$ indicates that a bundle b_s contains the resource r_t.

14.3 Query language for data extraction

Starting from the data model described in the previous section, we define the syntax of the query language which can be adopted by users to easily extract data from multiple social networks. We describe its syntax and its realization through a formal grammar. The structure of our query language is based on the well-known `select` statement of the SQL language, that retrieves zero or more tuples from one or more tables in a database. The basic idea behind this language is to conceptually substitute entities to tuples and social networks to tables in the standard meaning of the `select` statement. It means that, after the `select` keyword, the user has to specify the kind of entity he/she is looking for (i.e., profiles, resources or bundles), then he/she specifies the list of social networks to query after the `from` keyword and, at last, it is possible to add filtering criteria after the `where` keyword. Obviously, the language allows for the use of logical operators in the `where` clause in order to enable more complex predicates.

The final structure of our query language is reported in the following listing.

```
select E1, E2 ...
from S1, S2 ...
where P1 lop P2 ...
```

In the previous listing, `Ei` represents the kind of the requested entity. According to the data model described in the previous section, it is a keyword among `profile`, `resource` and `bundle`. Note that the query language allows selecting different kinds of entities in a single query. `Si` is a conventional name of the social network; the user specifies the set of social networks separating their names by commas. At last, `Pi` is a logical proposition which can be used to filter results; the logical proposition are combined together by logical operators `lop`. Following the data model of the previous section, in the *where* clause the user can specify the value of some elements of the tuple representative of the requesting entity (specified after the *select* keyword). For example, if the user is querying for resources, he/she can specify an url, a type, a description and/or a date. Similarly, if he/she is asking for profiles, it is possible to specify the url, the screen name, personal information and/or picture name as search criteria.

Among the different searching criteria that can be used in the *where* clause, some of them are strictly related to the specific social network. For example, the kind of a resource (e.g., film, book, song) strictly depends on the metadata used by the social network itself which stores that information. For this reason, our language implements a property we named *awareness*. It represents the capability of the language to be extended in keywords by users that are "aware" of the metadata used by every single social network to store its data. Consequently, we can distinguish between "known" and "unknown" social network. In the first category, the user inserts the social networks she previously analyzed, while in the second she puts the rest of social network she wants to use.

The list of social networks the user wants to use is stored in a configuration file. In this file, the user specifies the list of keywords identifying social networks he/she wants to add to the query language in the `from` clause. If a social network is "unknown" (i.e., the user does not know the metadata related to the social network), the user is automatically limited by the language syntax to use a set of general keywords given by the data model; for each "known" social network, instead, he/she has to define a specific configuration file in which he/she details the (sub-)kinds of profiles, resources and bundles can be requested. It means that, if a social network, for example, is able to manage books, films, and songs as kinds of resources, the user is allowed also to ask for a specific kind of them. In this case, the user specifies in the configuration file of the social network the keywords he/she want to use in its concrete language and the mapping with the metadata of the social network. After this one-time operation, he/she can open the language editor and use the keywords he/she specified.

The *awareness* property allows also for future easy integration of additional social networks. The operations needed are: (1) edit the configuration file of the social network and add the new social to the list, (2) possibly create a configuration file for the new social network with the mapping between keywords and metadata.

The described language has been completely realized by means of the Xtext framework [143]. Xtext is a complete framework for developing programming languages and domain-specific languages with textual grammars. Within Xtext it is possible to define your own language using a powerful grammar language. As a result, you get automatically the full infrastructure including parser, type checker and the entire editing environment for Eclipse.

To realize our query language in Xtext, it is necessary to code the syntax of the language. It has to be defined as a set of rules to which the text has to be compliant. The main rule, named *Query* checks for the overall structure of the query. The following listing reports this grammar rule.

```
Query:
  'select' selection+=Selection
  'from'   social+=Social(','social+=
         Social)*
  ('where' filter+=Filter)? ;
```

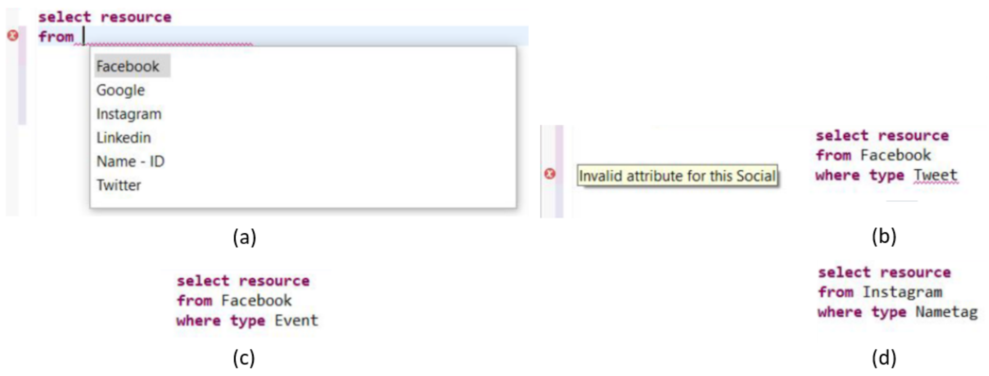
This rule specifies the sequence of keywords and textual portions that have to be satisfied by a query. The syntax of each clause has been designed with a specific rule. In the following listing, we report the remaining part of the grammar.

```
Selection:
  (('profile' profile+=Profile(',' profile+=Profile)*) |
  ('resource' resource+=Resource(',' resource+=Resource)*) |
  ('bundle' bundle+=Bundle(',' bundle+=Bundle)*))* ;

Social:
  name=ID ;

Filter:
  (condition+=Condition (('and'|'or')condition+=
   Condition)*) ;

Condition:
  ((profile+=Profile)? &
  (resource+=Resource)? &
  (bundle+=Bundle)?) ;

Profile:
  ('profileName' name=ID)?
  (('personalInformation' personalInformation= STRING)? &
  ('screenName' ScreenName=STRING)? &
  ('URLprofile' URL=STRING)?) ;

Resource:
  (('type' type+=Type)? &
  ('descriptionResource' description=STRING)? &
  ('creation_date' creation_date+=Date)? &
  ('URIresource' URI=STRING)?) ;

Type:
  name=ID ;

Bundle:
  (('descriptionBundle' description=STRING)? &
  ('creation_date' creation_date+=Date)? &
  ('URIbundle' URI=STRING)?) ;

Date:
  d=Day'-'m=Month'-'y=Year ;

Day : INT ;
Month : INT;
```

```
    Year : INT ;
```

The *Selection* rule mainly checks the usage of the three keywords (profile, resource, and entity), and properly calls three specific rules. The *Social* rule is very simple as, at grammar level, it could check only for the presence of a sequence of social network identifiers. As previously said, the keywords identifying social networks are defined by the user itself in a proper configuration file. At last, the *Filter* rule, together with an additional rule named *Condition*, checks the format of the search filter and allows to create complex combinations of conditions by using logical operators. The filtering mechanisms the user can specify for entities are defined in the three rules *Profile*, *Resource*, and *Bundle*. These rules check for the presence of keywords related to the attributes of these entities which are given by the data model of the previous section.

The implementation of the *awareness* property of the language, that is the verification of additional keywords in configuration files, is realized by specifying customized validation rules. Specifically, we defined the behavior of the *Validator* class that is used in the Xtext framework to customize the verification activity. Mainly, in this class we inserted the code necessary to check for the presence of the social network identifier in the configuration file and the verification of the existence of the configuration file related to the social network addressed in the query. In this case, the validation allows using additional keywords as given by the configuration file and previously described. The same logic has been implemented also in the *ProposalProvider* class to enable the content assist and suggest users the accepted keywords with respect to the structure of the query.



```
select resource
from 
```

Invalid attribute for this Social

```
select resource
from Facebook
where type Tweet
```

```
select resource
from Facebook
where type Event
```

```
select resource
from Instagram
where type Nametag
```

Fig. 14.1: Examples of usage of our query language: (a) content assist enabled; (b) wrong query (c) query for Facebook; (d) query for Instagram.

14.4 Using the query language

This section shows some examples of use in order to better clarify the objective of the defined query language and to demonstrate its effectiveness.

Fig. 14.1 depicts 4 different usage scenario. Fig. 14.1(a) shows the suggestions of the content assist in the *from* clause. In this scenario, the editing environment retrieved the list of social network from a configuration file and suggest them to a user. The user can select a social network from the shown list, avoiding syntactical errors.

Fig. 14.1(b) shows a validation error. In this scenario the user specifies that he/she is looking for resources from the Facebook social network and, in the *where* clause he/she filters result on the type *Tweet*. In this case, the environment check that this type (expressed as a keyword) is not specified in the configuration file related to the Facebook and raises an error.

Fig. 14.1(c) and Fig. 14.1(d) show two correct usage examples. In the former example the user asks for *nametags* in Instagram, while the latter shows a query requesting *events* from Facebook. These last examples show the concrete support the environment offers to users by enabling the writing of query which uses specific keywords related to the social network metadata.

14.5 Related Work

As stated previously, some OSNs offer some APIs as a part of their business [241]. In this way they allow third-party companies to use these APIs to develop complex applications. To the other hand, some works as [252, 94] analyzed and exploited web scraping for automated capture of data from OSNs. Web scraping (also known as web harvesting) is a well-known technique to extract online data. This technique is supported by different tools such as Selenium, cURL, Firebug, Node js [312].

As a consequence, the data extraction and the consequent realization of innovative web applications on the top of social networks is technically feasible. Nevertheless, its complexity is due to different and variable organization of data from the OSNs and the consequent variety of APIs and continuous updates they offer to programmers and/or the different organization of web pages. Indeed, despite the conceptual uniformity among the structures, mechanisms, features of existing social-networks, each platform adopts in practice its own terms, resources, actions. This is a strong handicap for the design and implementation of applications enabling internet-working functions among multiple social networks, and, then, for the achievement of the above goal. As a matter of fact, a few exist in terms of models and languages to support the interaction with multiple social networks.

Up to now a lot of effort has been spent in the definition of standards and models for the data exchange on web. For example, the Resource Description Framework RDF[3] is a W3C standard model for data exchange on web, in particular proposed for Semantic Web. Based on RDF, many researchers proposed different ontologies, because it guarantees interoperability. Indeed, the authors of [112] proposed SPIDER, a query processing system for RDF data which supports the SPARQL query language [2, 300]. In the literature, many works use RDF and/or SPARQL approaches for the manipulation and extraction of general data from web sources ([331, 132, 351, 67] to cite a few).

Some of the social network providers offer query languages and/or extraction techniques as well. For example, Facebook offers a simple query language [30] in order to process mainly your own and your friends' relational data. This language does not give any support for complex data extraction. From 2016, Facebook offers also a specification and a reference implementation of a framework, named GraphQL, which introduces a new type of web-based data access interface [97]. This framework is seen as an alternative to the standard REST-based interface [309]. The query language at the basis of GraphQL queries uses the JavaScript Object Notation (JSON) [119] and asks its users to define a schema in JSON. In particular, this schema defines the types of objects and the fields that the query has to populate. Consequently, this approach is very powerful but it is limited to complete knowledge of the possible types and it is also not extensible to multiple-social network scenario. The authors of [319] proposed SNQL, a data model and a query language for Social Network by representing data as graph database. Although the idea and the work are very interesting, these languages cannot be easily extendend to multiple social network scenarios given the different organization of the data.

In the context of multiple social networks, it is more appropriate to analyse the graph query languages. As an example, Neo4J [263] is one of the most complete query languages for Graph Databases, where data can be structured as edges and vertices. GOQL [327] is a different graph query language and it is based on an object-oriented data model. GOQL uses the traditional *select... from... where...* statement for querying and offers also temporal operators such as *next*, *until* and *connected*. The graph query languages can be better adapted to manage complex data structures, where nodes have properties and are organized by relationships, which can also have properties. However, these approaches are not tailored for extracting data from multiple social networks and they are not able to manage the different organization of data from the available sources, which represents, as said before, one of the the main issues in multiple social network.

With respect to the considered works, our approach is different because it aims at developing a complete framework for the creation, verification and execution of queries that are able to extract data from multiple OSNs. This framework has to offer the following functionalities:

1. enable users to define customized keywords and the mapping towards the specific metadata adopted by the OSNs;
2. support to users during the query editing and validate the results;
3. enable users to execute queries by choosing among the usage of APIs, of the web scraping techniques or of mixed approaches.

With respect to existing languages, the designed language aims at being general enough to include the variety of resources managed by the different social networks. Moreover, it allows users to extend its syntax, adding keywords to the language that map specific metadata offered by a single social network. The design of this language started from the formalization of a conceptual data model for the generalization of the structure of commonly known social networks. Then we defined the syntax and we realized the corresponding grammar file into the Xtext framework [143].

Part IV

Conclusions

In this thesis, we have presented several activities performed in the context of trust, security and privacy in the smart city domain. Specifically, the contribution provided by this thesis can be divided into three macro-areas, namely *(i)* blockchain for smart and trusted interactions, *(ii)* exploiting generated data and *(iii)* social networks, participation and analysis.

As for the first macro-area, we started by describing and deepening some basic concepts that are in common for every proposal included in this first part of the thesis. In particular, we presented the Blockchain technology, focusing on Ethereum and Smart Contracts, and the Smart City scenario. Then, we proposed a new architecture blockchain based for energy trading in smart grids through a blind auction managed by smart meters and the smart contract. After this proposal we presented a new approach for service delivery with accountability and privacy requirements that does not ask to the customer to register into any platform or to disclose any additional, and not necessary, personal information. An attribute-based access control mechanism based on smart contract is then discussed in which we address the problem of closed data in smart cities. We also proposed a new approach based on ethereum that enables the trust propagation in the Web of Trust scenario and, after that, we presented an approach that overcomes the blockchain limit related to the necessity of users to be already registered in the platform before participating in it. We address this situation by implementing a solution with the support of the Identity Based Encryption.

The second macro-area of this thesis opened with the proposal of a new paradigm able to represent multiple instances of smart objects into different networks. Then, we presented a model able to uniformly handle different and heterogeneous data lake sources through of graph representation. After, we proposed a solution for schema matching and integration for data lake sources using the detection of semantic relationships. We also applied these two lasts model to propose two approaches for the extraction of thematic views and complex knowledge patterns among concepts belonging to heterogeneous data lake sources.

As for the third macro-area we first proposed a decentralized solution that exploits collaboration among social networks users to compute the trust level of each social profile in order to avoid fake profiles and attacks. Finally, we proposed a new query language SQL-like that is able to question multiple online social networks with just one single query so that it is possible to retrieve data, correlate resources and integrate results from multiple OSNs.

References

1. Grid+ White Paper. https://gridplus.io/assets/Gridwhitepaper.pdf.
2. SPARQL Query Language for RDF . https://www.w3.org/tr/rdf-sparql-query/.
3. Resource Description Framework (RDF): Concepts and Abstract Syntax. Technical report, 2004.
4. Pentaho, Hadoop, and Data Lakes. https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/, 2010.
5. ISO/IEC 24760-1:2011 Information technology - Security techniques - A framework for identity management - Part 1: Terminology and concepts, 2011. http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html.
6. Oraclize, 2016. http://www.oraclize.it/.
7. BBC News [Online Version]. http://www.bbc.com/news/technology-34994858, 2017.
8. IPSO Alliance. *https://www.ipso-alliance.org/*, 2017.
9. Thingful: A Search Engine for the Internet of Things. *https://thingful.net/*, 2017.
10. Bitnation Pangea | Your Blockchain Jurisdiction, 2018. https://tse.bitnation.co/.
11. Digital identity, 2018. https://en.wikipedia.org/wiki/Digital_identity/.
12. ID-based Encryption, 2018. https://en.wikipedia.org/wiki/ID-based_encryption.
13. OAuth Community Site, 2018. https://oauth.net/.
14. OpenID âĂŞ The Internet Identity Layer, 2018. https://openid.net/.
15. SHA-3, 2018. https://en.wikipedia.org/wiki/SHA-3.
16. SPID Sistema Pubblico di Identità Digitale, 2018. https://www.spid.gov.it/.
17. The eIDAS Regulation in 2017 âĂŞ A pivotal year for digital services in the EU, 2018. https://www.gemalto.com/govt/identity/eidas-regulation-in-2017.
18. Windows CardSpace, 2018. https://en.wikipedia.org/wiki/Windows_CardSpace.
19. A Next-Generation Smart Contract and Decentralized Application Platform, 2019. https://github.com/ethereum/wiki/wiki/White-Paper.
20. Ethereum, 2019. https://www.ethereum.org/.
21. Ethereum Developer Resources, 2019. https://www.ethereum.org/developers/#getting-started.
22. Global Energy & CO2 Status Report 2019, (IEA), 2019. https://www.iea.org/reports/global-energy-co2-status-report-2019.
23. Metamask. https://metamask.io, 2019.

24. Provable. `https://provable.xyz/`, 2019.
25. Remix - Solidity IDE. `https://remix.ethereum.org`, 2019.
26. Ropsten Testnet Explorer. `https://ropsten.etherscan.io`, 2019.
27. Solidity 0.5.7 documentation. `https://solidity.readthedocs.io/en/v0.5.7`, 2019.
28. Truffle Suite. `https://www.trufflesuite.com/`, 2019.
29. web3js. `https://github.com/ethereum/web3.js/`, 2019.
30. Facebook. `https://www.facebook.com`, 2021.
31. Alfarez Abdul-Rahman. The PGP Trust Model. In *EDI-Forum: the Journal of Electronic Commerce*, volume 10, pages 27–31, 1997.
32. Muhammad Daniel Hafiz Abdullah, Zurina Mohd Hanapi, Zuriati Ahmad Zukarnain, and Mohamad Afendee Mohamed. Attacks, vulnerabilities and security requirements in smart metering networks. *KSII Transactions on Internet & Information Systems*, 9(4), 2015.
33. Saveen A Abeyratne and Radmehr P Monfared. Blockchain ready manufacturing supply chain using distributed ledger. 2016.
34. S. Abiteboul and O.M. Duschka. Complexity of answering queries using materialized views. In *Proc. of the International Symposium on Principles of database systems (SIGMOD-/PODS'98)*, pages 254–263, Seattle, WA, USA, 1998. ACM.
35. Nurzhan Zhumabekuly Aitzhan and Davor Svetinovic. Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams. *IEEE Transactions on Dependable and Secure Computing*, 15(5):840–852, 2016.
36. Mustafa Al-Bassam. Scpki: A smart contract-based pki and identity system. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, pages 35–40. ACM, 2017.
37. B.M. Albassuny. Automatic metadata generation applications&#58; a survey study. *nternational Journal of Metadata, Semantics and Ontologies*, 3(4):260–282, 2008.
38. Z. Aleksovski, M.C.A. Klein, W.T. Kate, and F. van Harmelen. Matching Unstructured Vocabularies Using a Background Ontology. In *Proc. of the International Conference on Knowledge Engineering and Knowledge Management (EKAW'06)*, pages 182–197, Prague, Czech Republic, 2006. Lecture Notes in Computer Science. Springer.
39. Nikolaos Alexopoulos, Jörg Daubert, Max Mühlhäuser, and Sheikh Mahbub Habib. Beyond the hype: On using blockchains in trust management for authentication. In *2017 IEEE Trustcom/BigDataSE/ICESS*, pages 546–553. IEEE, 2017.
40. A. Algergawy, E. Schallehn, and G. Saake. Improving XML schema matching performance using Pr. *Data & Knowledge Engineering*, 68(8):728–747, 2009. Elsevier.
41. S. P. Algur and P. Bhat. Web Video Object Mining: Expectation Maximization and Density Based Clustering of Web Video Metadata Objects. *International Journal of Information Engineering and Electronic Business*, 8(1):69, 2016. Modern Education and Computer Science Press.
42. Maher Alharby and Aad van Moorsel. Blockchain-based smart contracts: A systematic mapping study. *arXiv preprint arXiv:1710.06372*, 2017.

43. F. Ali, S. Islam, D. Kwak, P. Khan, N. Ullah, S. Yoo, and K. Kwak. Type-2 fuzzy ontology–aided recommendation systems for IoT–based healthcare. *Computer Communications*, 2017. Elsevier.

44. Fadi Aloul, AR Al-Ali, Rami Al-Dalky, Mamoun Al-Mardini, and Wassim El-Hajj. Smart grid security: Threats, vulnerabilities and solutions. *International Journal of Smart Grid and Clean Energy*, 1(1):1–6, 2012.

45. A. Alserafi, A. Abello, O. Romero, and T. Calders. Towards information profiling: data lake content metadata management. In *Proc. of the International Conference on Data Mining Workshops (ICDMW'16)*, pages 178–185, Barcelona, Spain, 2016. IEEE.

46. Riham AlTawy, Muhammad ElSheikh, Amr M Youssef, and Guang Gong. Lelantos: A blockchain-based anonymous physical delivery system. In *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, pages 15–1509. IEEE, 2017.

47. M. Amadeo, C. Campolo, A. Iera, and A. Molinaro. Named data networking for IoT: An architectural perspective. In *Proc. of the European Conference on Networks and Communications (EuCNC'2014)*, pages 1–5, Bologna, Italy, 2014. IEEE.

48. M. Amadeo, C. Campolo, J. Quevedo, D. Corujo, A. Molinaro, A. Iera, R. Aguiar, and A. Vasilakos. Information-centric networking for the internet of things: challenges and opportunities. *IEEE Network*, 30(2):92–100, 2016. IEEE.

49. Merlinda Andoni, Valentin Robu, David Flynn, Simone Abram, Dale Geach, David Jenkins, Peter McCallum, and Andrew Peacock. Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable and Sustainable Energy Reviews*, 100:143–174, 2019.

50. Fabrizio Angiulli, Fabio Fassetti, Angelo Furfaro, Antonio Piccolo, and Domenico Saccà. Achieving Service Accountability Through Blockchain and Digital Identity. In *International Conference on Advanced Information Systems Engineering*, pages 16–23. Springer, 2018.

51. Nuttapong Attrapadung and Hideki Imai. Dual-policy attribute based encryption. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *Applied Cryptography and Network Security*, pages 168–185, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

52. L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A survey. *Computer networks*, 54(15):2787–2805, 2010. Elsevier.

53. L. Atzori, A. Iera, and G. Morabito. SIoT: Giving a social structure to the Internet of Things. *IEEE Communications Letters*, 15(11):1193–1195, 2011. IEEE.

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

55. L. Atzori, A. Iera, and G. Morabito. Understanding the Internet of Things: definition, potentials, and societal role of a fast evolving paradigm. *Ad Hoc Networks*, 56:122–140, 2017. Elsevier.

56. L. Atzori, A. Iera, G. Morabito, and M. Nitti. 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, 2012. Elsevier.
57. Luigi Atzori, Antonio Iera, and Giacomo Morabito. Siot: Giving a social structure to the internet of things. *IEEE communications letters*, 15(11):1193–1195, 2011.
58. Luigi Atzori, Antonio Iera, Giacomo Morabito, and Michele Nitti. 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, 2012.
59. L. Aversano, R. Intonti, C. Quattrocchi, and M. Tortorella. Building a virtual view of heterogeneous data source views. In *Proc. of the International Conference on Software and Data Technologies (ICSOFT'10)*, pages 266–275, Athens, Greece, 2010. INSTICC Press.
60. C. Bachtarzi and F. Bachtarzi. A model-driven approach for materialized views definition over heterogeneous databases. In *Proc. of the International Conference on New Technologies of Information and Communication (NTIC'15)*, pages 1–5, Mila, Algeria, 2015. IEEE.
61. Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. Persona: an online social network with user-defined privacy. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 135–146. ACM, 2009.
62. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. 1999. Addison Wesley Longman.
63. Zubair A Baig, Patryk Szewczyk, Craig Valli, Priya Rabadia, Peter Hannay, Maxim Chernyshev, Mike Johnstone, Paresh Kerai, Ahmed Ibrahim, Krishnun Sansurooah, et al. Future challenges for smart cities: Cyber-security and digital forensics. *Digital Investigation*, 22:3–13, 2017.
64. Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.
65. Arnab Banerjee. Blockchain Technology: Supply Chain Insights from ERP. *Advances in Computers*, 2018.
66. Gaurang Bansal, Amit Dua, Gagangeet Singh Aujla, Maninderpal Singh, and Neeraj Kumar. Smartchain: a smart and scalable blockchain consortium for smart grid systems. In *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2019.
67. Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, EMANUELE DELLA VALLE, and Michael Grossniklaus. C-sparql: a continuous query language for rdf data streams. *International Journal of Semantic Computing*, 4(01):3–25, 2010.
68. Jaume Barcelo. User privacy in the public bitcoin blockchain. *URL: http://www. dtic. upf. edu/jbarcelo/papers/20140704 User Privacy in the Public Bitcoin Blockc hain/paper. pdf (Accessed 09/05/2016)*, 2014.
69. Michael Batty, Kay W Axhausen, Fosca Giannotti, Alexei Pozdnoukhov, Armando Bazzani, Monica Wachowicz, Georgios Ouzounis, and Yuval Portugali. Smart cities of the future. *The European Physical Journal Special Topics*, 214(1):481–518, 2012.

70. Mohamed Baza, Mahmoud Nabil, Muhammad Ismail, Mohamed Mahmoud, Erchin Serpedin, and Mohammad Ashiqur Rahman. Blockchain-based charging coordination mechanism for smart grid energy storage units. In *2019 IEEE International Conference on Blockchain (Blockchain)*, pages 504–509. IEEE, 2019.
71. Amos Beimel. *Secure schemes for secret sharing and key distribution*.
72. Gavin Bell. *Building social Web applications: Establishing community at the heart of your site*. " O'Reilly Media, Inc.", 2009.
73. S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.
74. S. Bergamaschi, S. Castano, M. Vincini, and D. Beneventano. Semantic integration and query of heterogeneous information sources. *Data & Knowledge Engineering*, 36(3):215–249, 2001.
75. J. Bernabé-Moreno, A. Tejeda-Lorente, C. Porcel-Gallego, and E. Herrera-Viedma. Leveraging Localized Social Media Insights for Industry Early Warning Systems. *International Journal of Information Technology & Decision Making*, 17(01):357–385, 2018. World Scientific.
76. P.A. Bernstein, J. Madhavan, and E. Rahm. Generic Schema Matching, Ten Years Later. *Proceedings of the VLDB Endowment*, 4(11):695–701, 2011.
77. J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy (SP '07)*, pages 321–334, May 2007.
78. Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *Proceedings of the 18th international conference on World wide web*, pages 551–560. ACM, 2009.
79. L. Bing, S. Jiang, W. Lam, Y. Zhang, and S. Jameel. Adaptive Concept Resolution for document representation and its applications in text mining. *Knowledge Based Systems*, 74:1–13, 2015.
80. J. Biskup and D. Embley. Extracting information from heterogeneous information sources using ontologically specified target views. *Information Systems*, 28(3):169–212, 2003. Elsevier.
81. Stefano Bistarelli, Marco Mantilacci, Paolo Santancini, and Francesco Santini. An end-to-end voting-system based on bitcoin. In *Proceedings of the Symposium on Applied Computing*, pages 1836–1841. ACM, 2017.
82. Andreas Bogner, Mathieu Chanson, and Arne Meeuw. A decentralised sharing app running a smart contract on the ethereum blockchain. In *Proceedings of the 6th International Conference on the Internet of Things*, pages 177–178. ACM, 2016.
83. Dan Boneh and Matt Franklin. Identity-based encryption from the Weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.
84. M. R. Bouadjenek, H. Hacid, and M. Bouzeghoub. Social networks and information retrieval, how are they converging? A survey, a taxonomy and an analysis of social information retrieval approaches and platforms. *Information Systems*, 56:1–18, 2016.

85. A. Boukottaya and C. Vanoirbeek. Schema matching for transforming structured documents. In *Proc. of the ACM Symposium on Document Engineering (DocEng'05)*, pages 101–110, Bristol, United Kingdom, 2005. ACM.
86. Aurélie Brunie. Meaningful distinctions within a concept: Relational, collective, and generalized social capital. *Social science research*, 38(2):251–265, 2009.
87. F. Buccafurri, V.D. Foti, G. Lax, A. Nocera, and D. Ursino. Bridge Analysis in a Social Internetworking Scenario. *Information Sciences*, 224:1–18, 2013. Elsevier.
88. Francesco Buccafurri, Vincenzo De Angelis, Gianluca Lax, Lorenzo Musarella, and Antonia Russo. An attribute-based privacy-preserving ethereum solution for service delivery with accountability requirements. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, page 24. ACM, 2019.
89. Francesco Buccafurri, Lidia Fotia, and Gianluca Lax. Implementing Advanced Electronic Signature by Public Digital Identity System (SPID). In *International Conference on Electronic Government and the Information Systems Perspective*, pages 289–303. Springer, 2016.
90. Francesco Buccafurri, Lidia Fotia, Gianluca Lax, and Rocco Mammoliti. Enhancing Public Digital Identity System (SPID) to Prevent Information Leakage. In *International Conference on Electronic Government and the Information Systems Perspective*, pages 57–70. Springer, 2015.
91. Francesco Buccafurri, Gianluca Lax, Denis Migdal, Serena Nicolazzo, Antonino Nocera, and Christophe Rosenberger. Contrasting False Identities in Social Networks by Trust Chains and Biometric Reinforcement. In *2017 International Conference on Cyberworlds (CW)*, pages 17–24. IEEE, 2017.
92. Francesco Buccafurri, Gianluca Lax, Lorenzo Musarella, and Antonia Russo. Ethereum transactions and smart contracts among secure identities. In *DLT@ ITASEC*, pages 5–16, 2019.
93. Francesco Buccafurri, Gianluca Lax, Serena Nicolazzo, and Antonino Nocera. A model to support design and development of multiple-social-network applications. *Information Sciences*, 331:99–119, 2016.
94. Francesco Buccafurri, Gianluca Lax, Antonino Nocera, and Domenico Ursino. A system for extracting structural information from social network accounts. *Software: Practice and Experience*, 45(9):1251–1275, 2015.
95. Francesco Buccafurri, Gianluca Lax, Antonia Russo, and Guillaume Zunino. Integrating digital identity and blockchain. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 568–585. Springer, 2018.
96. Francesco Buccafurri, Lorenzo Musarella, and Roberto Nardone. A routing algorithm increasing the transmission availability in smart grids. In *Proceedings of the 2019 Summer Simulation Conference*, page 47. Society for Computer Simulation International, 2019.
97. Samer Buna. *Learning GraphQL and Relay*. Packt Publishing Ltd, 2016.
98. Marian Buschsieweke and Mesut Güneş. Securing critical infrastructure in smart cities: Providing scalable access control for constrained devices. In *2017 IEEE 28th Annual*

International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), pages 1–6. IEEE, 2017.

99. H. Cai, B. Xu, L. Jiang, and A. Vasilakos. IoT-based big data storage systems in cloud computing: perspectives and challenges. *IEEE Internet of Things Journal*, 4(1):75–87, 2017. IEEE.

100. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Annual International Cryptology Conference*, pages 56–72. Springer, 2004.

101. S. Castano and V. De Antonellis. Building views over semistructured data sources. In *Proc. of the International Conference on Conceptual Modeling (ER'99)*, pages 146–160, Paris, France, 1999. Springer.

102. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati. Global viewing of heterogeneous data sources. *IEEE Transactions on Data and Knowledge Engineering*, 13(2):277–297, 2001.

103. Kun-Tai Chan, Raylin Tso, Chien-Ming Chen, and Mu-En Wu. Reputation-Based Trust Evaluation Mechanism for Decentralized Environments and Its Applications Based on Smart Contracts. In *Advances in Computer Science and Ubiquitous Computing*, pages 310–314. Springer, 2017.

104. C.P. Chen and C.Y. Zhang. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Information Sciences*, 275:314–347, 2014.

105. J. Chen, N. Zhong, and J. Feng. Developing a Provenance Warehouse for the Systematic Brain Informatics Study. *International Journal of Information Technology & Decision Making*, 16(06):1581–1609, 2017. World Scientific.

106. L. Chen, J. Shao, Z. Yu, J. Sun, F. Wu, and Y. Zhuang. RAISE: A Whole Process Modeling Method for Unstructured Data Management. In *Proc. of the International Conference on Multimedia Big Data (BigMM'15)*, pages 9–12, China National Conference Center, China, 2015. IEEE.

107. L. Chen, M. Tseng, and X. Lian. Development of foundation models for Internet of Things. *Frontiers of Computer Science in China*, 4(3):376–385, 2010. Springer.

108. X. Chen, A. Shrivastava, and A. Gupta. Neil: Extracting visual knowledge from web data. In *Proc. of the International Conference on Computer Vision (ICCV'13)*, pages 1409–1416, Darling Harbour, Sydney, 2013. IEEE.

109. Y. Chen, W. Wang, and Z. Liu. Keyword-based search and exploration on databases. In *Proc. of the International Conference on Data Engineering (ICDE'11)*, pages 1380–1383, Hannover, Germany, 2011. IEEE.

110. Y. Chen, Z. Zhen, H. Yu, and J. Xu. Application of Fault Tree Analysis and Fuzzy Neural Networks to Fault Diagnosis in the Internet of Things (IoT) for Aquaculture. *Sensors*, 17(1):153, 2017. Multidisciplinary Digital Publishing Institute.

111. M. Chessell, F. Scheepers, N. Nguyen, R van Kessel, and R. van der Starre. Governing and managing big data for analytics and decision makers. *IBM Redguides for Business Leaders*, 2014.

112. Hyunsik Choi, Jihoon Son, YongHyun Cho, Min Kyoung Sung, and Yon Dohn Chung. Spider: a system for scalable, parallel/distributed evaluation of large-scale rdf data. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 2087–2088. ACM, 2009.

113. B. Christophe, V. Verdot, and V. Toubian. Searching the 'web of things'. In *Proc. of the International Conference on Semantic Computing (ICSC'2011)*, pages 308–315, Palo Alto, CA, USA, 2011. IEEE.

114. Amanda Clarke and Helen Margetts. Governments and citizens getting to know each other? open, closed, and big data in public management reform. *Policy & Internet*, 6(4):393–417, 2014.

115. Alan Cohn, Travis West, and Chelsea Parker. Smart after all: Blockchain, smart contracts, parametric insurance, and smart energy grids. *Georgetown Law Technology Review*, 1(2):273–304, 2017.

116. Mauro Conti, Radha Poovendran, and Marco Secchiero. Fakebook: Detecting fake profiles in on-line social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 1071–1078. IEEE, 2012.

117. Alejandro Corbellini, Cristian Mateos, Alejandro Zunino, Daniela Godoy, and Silvia Schiaffino. Persisting big-data: The nosql landscape. *Information Systems*, 63:1–23, 2017.

118. T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to algorithms*. 2009. MIT press.

119. Douglas Crockford. The application/json media type for javascript object notation (json). Technical report, 2006.

120. Leucio Antonio Cutillo, Refik Molva, and Thorsten Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. *IEEE Communications Magazine*, 47(12), 2009.

121. Morteza Dabbaghjamanesh, Boyu Wang, Shahab Mehraeen, Jie Zhang, and Abdollah Kavousi-Fard. Networked microgrid security and privacy enhancement by the blockchain-enabled internet of things approach. In *2019 IEEE Green Technologies Conference (GreenTech)*, pages 1–5. IEEE, 2019.

122. Gaby G Dagher, Jordan Mohler, Matea Milojkovic, and Praneeth Babu Marella. Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. *Sustainable Cities and Society*, 39:283–297, 2018.

123. Fangfang Dai, Yue Shi, Nan Meng, Liang Wei, and Zhiguo Ye. From bitcoin to cybersecurity: A comparative study of blockchain application and security issues. In *Systems and Informatics (ICSAI), 2017 4th International Conference on*, pages 975–979. IEEE, 2017.

124. Arnold Daniels. The rise of private permissionless blockchains , 2018. `https://medium.com/ltonetwork/the-rise-of-private-permissionless-blockchains-part-1-4c39bea2e2be/`.

125. Chris Dannen. *Introducing Ethereum and Solidity*. Springer, 2017.

126. Maurizio D'Arienzo, Mauro Iacono, Stefano Marrone, and Roberto Nardone. Petri net based evaluation of energy consumption in wireless sensor nodes. *Journal of High Speed Networks*, 19(4):339–358, 2013.

127. A. Dass, C. Aksoy, A. Dimitriou, and D. Theodoratos. Relaxation of keyword pattern graphs on RDF Data. *Journal of Web Engineering*, 16(5-6):363–398, 2017. Rinton Press, Incorporated.

128. Angelo De Caro and Vincenzo Iovino. jpbc: Java pairing based cryptography. In *2011 IEEE symposium on computers and communications (ISCC)*, pages 850–855. IEEE, 2011. `http://gas.dia.unisa.it/projects/jpbc/`.

129. P. De Meo, G. Quattrone, G. Terracina, and D. Ursino. Integration of XML Schemas at various "severity" levels. *Information Systems*, 31(6):397–434, 2006.

130. F. Di Tria, E. Lefons, and F. Tangorra. Cost-benefit analysis of data warehouse design methodologies. *Information Systems*, 63:47–62, 2017. Elsevier.

131. C. Diamantini, P. Lo Giudice, L. Musarella, D. Potena, E. Storti, and D. Ursino. An approach to extracting thematic views from highly heterogeneous sources of a data lake. In *Atti del Ventiseiesimo Convegno Nazionale su Sistemi Evoluti per Basi di Dati (SEBD'18)*, Castellaneta Marina (TA), Italy, 2018.

132. Claudia Diamantini, Paolo Lo Giudice, Lorenzo Musarella, Domenico Potena, Emanuele Storti, and Domenico Ursino. An approach to extracting thematic views from highly heterogeneous sources of a data lake. In *Proceedings of the 26th Italian Symposium on Advanced Database Systems (SEBD)*, 2018.

133. S. Distefano, G. Merlino, and A. Puliafito. Enabling the cloud of things. In *Proc. of the International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS'2012)*, pages 858–863, Taichung, Taiwan, 2012. IEEE.

134. Quang Do, Ben Martini, and Kim-Kwang Raymond Choo. The role of the adversary model in applied security research. *Computers & Security*, 81:156–181, 2019.

135. Chris Dong, Lingzhi Du, Feiran Ji, Zizhen Song, Yuedi Zheng, Alexander Howard, Paul Intrevado, and Diane Woodbridge. Forecasting smart meter energy usage using distributed systems and machine learning. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1293–1298. IEEE, 2018.

136. H. Dong, F. Hussain, and E. Chang. A framework for discovering and classifying ubiquitous services in digital health ecosystems. *Journal of Computer and System Sciences*, 77(4):687–704, 2011. Elsevier.

137. Ali Dorri, Fengji Luo, Salil S Kanhere, Raja Jurdak, and Zhao Yang Dong. Spb: A secure private blockchain-based solution for distributed energy trading. *IEEE Communications Magazine*, 57(7):120–126, 2019.

138. R. dos Santos Mello, S. Castano, and C.A. Heuser. A method for the unification of XML schemata. *Information & Software Technology*, 44(4):241–249, 2002. Elsevier.

139. Jos Dumortier. Regulation (eu) no 910/2014 on electronic identification and trust services for electronic transactions in the internal market (eidas regulation). In *EU Regulation of E-Commerce*. Edward Elgar Publishing, 2017.

140. Rolf Egert, Andrea Tundis, Stefan Roth, and Max Mühlhäuser. A Service Quality Indicator for Apriori Assessment and Comparison of Cellular Energy Grids. In *International Conference on Sustainability in Energy and Buildings*, pages 322–332. Springer, 2018.

141. H. Elmeleegy, M. Ouzzani, and A.K. Elmagarmid. Usage-Based Schema Matching. In *Proc. of the International Conference on Data Engineering (ICDE'08)*, pages 20–29, Cancún, México, 2008. IEEE.

142. ethereumWiki. Problems. `https://github.com/ethereum/wiki/wiki/Problems`, 2016.

143. Moritz Eysholdt and Heiko Behrens. Xtext: implement your language faster than the quick and dirty way. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, pages 307–309. ACM, 2010.

144. Kai Fan, Junxiong Wang, Xin Wang, and Yintang Yang. Proxy-assisted access control scheme of cloud data for smart cities. *Personal and Ubiquitous Computing*, 21(5):937–947, 2017.

145. W. Fan, X. Wang, and Y. Wu. Answering pattern queries using views. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):326–341, 2016. IEEE.

146. H. Fang. Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In *Proc. of the International Conference on Cyber Technology in Automation (CYBER'15)*, pages 820–824, Shenyang, China, 2015. IEEE.

147. Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart gridâĂŤthe new and improved power grid: A survey. *IEEE communications surveys & tutorials*, 14(4):944–980, 2012.

148. M. Farid, A. Roatis, I.F. Ilyas, H. Hoffmann, and X. Chu. CLAMS: bringing quality to Data Lakes. In *Proc. of the International Conference on Management of Data (SIGMOD-/PODS'16)*, pages 2089–2092, San Francisco, CA, USA, 2016. ACM.

149. Mina Farmanbar, Kiyan Parham, Øystein Arild, and Chunming Rong. A widespread review of smart grids towards smart cities. *Energies*, 12(23):4484, 2019.

150. I. Farris, R. Girau, L. Militano, M. Nitti, L. Atzori, A. Iera, and G. Morabito. Social virtual objects in the edge cloud. *IEEE Cloud Computing*, 2(6):20–28, 2015. IEEE.

151. A. Farrugia, R. Claxton, and S. Thompson. Towards social network analytics for understanding and managing enterprise data lakes. In *Proc. of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM'16)*, pages 1213–1220, San Francisco, CA, USA, 2016. IEEE.

152. F. Feng and W.B. Croft. Probabilistic techniques for phrase extraction. *Information Processing & Management*, 37(2):199–220, 2001.

153. C. Fetzer, P. Felber, E. Riviere, V. Schiavoni, and P. Sutra. Unicrawl: A practical geographically distributed web crawler. In *Proc. of the International Conference on Cloud Computing (CLOUD'2015)*, pages 389–396, New York, NY, USA, 2015. IEEE.

154. T. Foley, H. Hagen, and G. Nielson. Visualizing and modeling unstructured data. *The Visual Computer*, 9(8):439–449, 1993. Springer.

155. Noah E Friedkin. Horizons of observability and limits of informal control in organizations. *Social Forces*, 62(1):54–77, 1983.

156. E. Gabrilovich and S. Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 1606–1611, Hyderabad, India, 2007.

157. Keke Gai, Yulu Wu, Liehuang Zhu, Meikang Qiu, and Meng Shen. Privacy-preserving energy trading using consortium blockchain in smart grid. *IEEE Transactions on Industrial Informatics*, 15(6):3548–3558, 2019.

158. Keke Gai, Yulu Wu, Liehuang Zhu, Lei Xu, and Yan Zhang. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet of Things Journal*, 6(5):7992–8004, 2019.

159. Simson Garfinkel. *PGP: pretty good privacy.* " O'Reilly Media, Inc.", 1995.

160. Valentina Gatteschi, Fabrizio Lamberti, Claudio Demartini, Chiara Pranteda, and Víctor Santamaría. Blockchain and smart contracts for insurance: Is the technology mature enough? *Future Internet*, 10(2):20, 2018.

161. R. Gaur and D.K. Sharma. Review of ontology based focused crawling approaches. In *Proc. of the International Conference on Soft Computing Techniques for Engineering and Technology (ICSCTET'2014)*, pages 1–4, Nainital, India, 2014. IEEE.

162. Ugo Gentile, Stefano Marrone, Nicola Mazzocca, and Roberto Nardone. Cost-energy modelling and profiling of smart domestic grids. *International Journal of Grid and Utility Computing*, 7(4):257–271, 2016.

163. Stefanie Gerdes, Olaf Bergmann, and Carsten Bormann. Delegated coap authentication and authorization framework (dcaf). draft-gerdes-ace-dcafauthorize-02. *Work in progress*, 66, 2015.

164. M. Giacobbe, M. Coco, A. Puliafito, and M. Scarpa. A cloud-based access control solution for advanced multi-purpose management in smart city scenario. In *2014 International Conference on Smart Computing Workshops*, pages 35–40, Nov 2014.

165. Scott Gidley. Tips for managing metadata in a data lake.

166. M. Gjoka, M. Kurant, C.T. Butts, and A. Markopoulou. Walking in Facebook: A case study of unbiased sampling of OSNs. In *Proc. of the International Conference on Computer Communications (INFOCOM'10)*, pages 1–9, San Diego, CA, USA, 2010. IEEE.

167. Martin Glinz. Statecharts for requirements specification-as simple as possible, as rich as needed. In *Proceedings of the ICSE2002 workshop on scenarios and state machines: models, algorithms, and tools*, 2002.

168. K. Golenberg, B. Kimelfeld, and Y. Sagiv. Keyword proximity search in complex data graphs. In *Proc. of the International Conference on Management of data (SIGMOD-/PODS'08)*, pages 927–940, Vancouver, Canada, 2008. ACM.

169. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM*

Conference on Computer and Communications Security, CCS ’06, pages 89–98, New York, NY, USA, 2006. ACM.

170. John R Graham, Dana Watts, and Rodney E Timbrook. Detecting fake-good and fake-bad mmpi-2 profiles. *Journal of personality Assessment*, 57(2):264–277, 1991.
171. John Guare. *Six degrees of separation: A play*. Vintage, 1990.
172. J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013. Elsevier.
173. D. Guinard, M. Fischer, and V. Trifa. Sharing using social networks in a composable web of things. In *Proc. of the International Conference on Pervasive Computing and Communications (PERCOM 2010)*, pages 702–707, Mannheim, Germany, 2010. IEEE.
174. D. Guinard, V. Trifa, F. Mattern, and E. Wilde. From the internet of things to the web of things: Resource-oriented architecture and best practices. *Architecting the Internet of Things*, pages 97–129, 2011. Springer.
175. D. Guinard, V. Trifa, and E. Wilde. Architecting a mashable open world wide web of things. *Technical Report of the Institute for Pervasive Computing, ETH Zürich, Zürich, Switzerland*, 663, 2010.
176. Muhammed Zekeriya Gunduz and Resul Das. Cyber-security on smart grid: Threats and potential solutions. *Computer Networks*, 169:107094, 2020.
177. Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. Smart grid technologies: Communication technologies and standards. *IEEE transactions on Industrial informatics*, 7(4):529–539, 2011.
178. Guibing Guo, Jie Zhang, and Julita Vassileva. Improving PGP web of trust through the expansion of trusted neighborhood. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*, pages 489–494. IEEE Computer Society, 2011.
179. Rolf Haenni and Jacek Jonczy. A New Approach to PGPâĂŹs Web of Trust. In *EEMAâĂŹ07, European e-Identity Conference*, 2007.
180. R. Hai, S. Geisler, and C. Quix. Constance: An intelligent data lake system. In *Proc. of the International Conference on Management of Data (SIGMOD/PODS’16)*, pages 2097–2100, San Francisco, CA, USA, 2016. ACM.
181. R. Hai, S. Geisler, and C. Quix. Constance: An intelligent data lake system. In *Proc. of the International Conference on Management of Data (SIGMOD’16)*, pages 2097–2100, San Francisco, CA, USA, 2016. ACM.
182. A. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001. Springer.
183. J. Han and M. Kamber. *Data Mining: Concepts and Techniques - Second Edition*. Morgan Kaufmann notes, 2006.
184. S. Han, L. Zou, J.X. Yu, and D. Zhao. Keyword Search on RDF Graphs-A Query Graph Assembly Approach. In *Proc. of the International Conference on Information and Knowledge Management (CIKM’17)*, pages 227–236, Singapore, Singapore, 2017. ACM.

185. S.M. Harding, W.B. Croft, and C. Weir. Probabilistic retrieval of ocr degraded text using n-grams. In *Proc. of the International Conference on Theory and Practice of Digital Libraries (ECDL'97*.

186. Haya R Hasan and Khaled Salah. Blockchain-based solution for proof of delivery of physical assets. In *International Conference on Blockchain*, pages 139–152. Springer, 2018.

187. H. He, H. Wang, J. Yang, and P.S. Yu. BLINKS: ranked keyword searches on graphs. In *Proc. of the International Conference on Management of data (SIGMOD/PODS'07)*, pages 305–316, Beijing, China, 2007. ACM.

188. P. Hitzler and K. Janowicz. Linked Data, Big Data, and the 4th Paradigm. *Semantic Web*, 4(3):233–235, 2013.

189. L. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Proc. of the International Conference on Ubiquitous Computing (Ubicomp'2001)*, pages 116–122, Atlanta, GA, USA, 2001. Springer.

190. Vincent C Hu, David Ferraiolo, Rick Kuhn, Adam Schnitzer, Kenneth Sandlin, Robert Miller, and Scarfone. Guide to attribute based access control (abac) definition and considerations. *NIST special publication*, 800(162), 2014.

191. Jingwei Huang and David Nicol. A calculus of trust and its application to pki and identity management. In *Proceedings of the 8th Symposium on Identity and Trust on the Internet*, pages 23–37. ACM, 2009.

192. N.Q.V. Hung, N.T. Tam, V.T. Chau, T.K. Wijaya, Z. Miklós, K. Aberer, A. Gal, and M. Weidlich. SMART: A tool for analyzing and reconciling schema matching networks. In *Proc. of the International Conference on Data Engineering (ICDE'15)*, pages 1488–1491, Seoul, South Korea, 2015. IEEE.

193. I. Ishaq, D. Carels, G. Teklemariam, J. Hoebeke, F. Abeele, E. Poorter, I. Moerman, and P. Demeester. IETF standardization in the field of the Internet of Things (IoT): a survey. *Journal of Sensor and Actuator Networks*, 2(2):235–287, 2013. Multidisciplinary Digital Publishing Institute.

194. Shama Naz Islam, Zubair Baig, and Sherali Zeadally. Physical layer security for the smart grid: vulnerabilities, threats, and countermeasures. *IEEE Transactions on Industrial Informatics*, 15(12):6522–6530, 2019.

195. Ori Jacobovitz. Blockchain for identity management, 2016.

196. S. Jain and S. Tanwani. Schema matching technique for heterogeneous web database. In *Proc. of the International Conference on Reliability (ICRITO'15)*, pages 1–6, Noida, India, 2015. IEEE.

197. F. Jebbor and L. Benhlima. Overview of knowledge extraction techniques in five question-answering systems. In *Proc. of the International Conference on Intelligent Systems: Theories and Applications (SITA'14)*, pages 1–8, Rabat, Morocco, 2014. IEEE.

198. S. Jiang, L. Bing, B. Sun, Y. Zhang, and W. Lam. Ontology enhancement and concept granularity learning: keeping yourself current and adaptive. In *Proc. of the International Conference on Knowledge Discovery and Data Mining (KDD'11)*, pages 1244–1252, San Diego, CA, USA, 2011. ACM.

199. Q. Jing, A. Vasilakos, J. Wan, J. Lu, and D. Qiu. Security of the Internet of Things: perspectives and challenges. *Wireless Networks*, 20(8):2481–2501, 2014. Springer.

200. Yasin Kabalci. A survey on smart metering and smart grid communication. *Renewable and Sustainable Energy Reviews*, 57:302–318, 2016.

201. V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar. Bidirectional expansion for keyword search on graph databases. In *Proc. of the International Conference on Very Large DataBase (VLDB'05)*, pages 505–516, Trondheim, Norway, 2005. VLDB Endowment.

202. A.B. Kahn. Topological sorting of large networks. *Communications of the ACM*, 5(11):558–562, 1962.

203. Eung Seon Kang, Seung Jae Pee, Jae Geun Song, and Ju Wook Jang. A blockchain-based energy trading platform for smart homes in a microgrid. In *2018 3rd International Conference on Computer and Communication Systems (ICCCS)*, pages 472–476. IEEE, 2018.

204. S. Kapidakis. Rating quality in metadata harvesting. In *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '15, pages 65:1–65:8, New York, NY, USA, 2015. ACM.

205. M. Kargar and A. An. Keyword search in graphs: Finding r-cliques. *Proceedings of the VLDB Endowment*, 4(10):681–692, 2011. VLDB Endowment.

206. M.D. Karim, M. Cochez, O.D. Beyan, C.F. Ahmed, and S. Decker. Mining maximal frequent patterns in transactional databases and dynamic data streams: A spark-based approach. *Information Sciences*, 432:278–300, 2018.

207. S. Karnouskos. Smart houses in the smart grid and the search for value-added services in the cloud of things era. In *Proc. of the International Conference on Industrial Technology (ICIT'2013)*, pages 2016–2021, Cape Town, Western Cape, South Africa, 2013. IEEE.

208. Salam Khanji, Farkhund Iqbal, and Patrick Hung. Zigbee security vulnerabilities: Exploration and evaluating. In *2019 10th International Conference on Information and Communication Systems (ICICS)*, pages 52–57. IEEE, 2019.

209. Rida Khatoun and Sherali Zeadally. Smart cities: concepts, architectures, research opportunities. *Communications of the ACM*, 59(8):46–57, 2016.

210. Rida Khatoun and Sherali Zeadally. Cybersecurity and privacy solutions in smart cities. *IEEE Communications Magazine*, 55(3):51–59, 2017.

211. Asad Masood Khattak, Salam Ismail Khanji, and Wajahat Ali Khan. Smart meter security: Vulnerabilities, threat impacts, and countermeasures. In *International Conference on Ubiquitous Information Management and Communication*, pages 554–562. Springer, 2019.

212. H. Kim, P. Howland, and H. Park. Dimension Reduction in Text Classification with Support Vector Machines. *Journal of Machine Learning Research*, 6:37–53, 2005.

213. H.K. Kim, H. Kim, and S. Cho. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing*, 266:336–352, 2017.

214. Markus Klems, Jacob Eberhardt, Stefan Tai, Steffen Härtlein, Simon Buchholz, and Ahmed Tidjani. Trustless intermediation in blockchain-based decentralized service mar-

ketplaces. In *International Conference on Service-Oriented Computing*, pages 731–739. Springer, 2017.

215. G. Kondrak. N-gram similarity and distance. In *String processing and information retrieval*, pages 115–126, 2005. Springer.
216. Kari Korpela, Jukka Hallikas, and Tomi Dahlberg. Digital supply chain transformation toward blockchain integration. In *proceedings of the 50th Hawaii international conference on system sciences*, 2017.
217. Ioannis Kounelis, Gary Steri, Raimondo Giuliani, Dimitrios Geneiatakis, Ricardo Neisse, and Igor Nai-Fovino. Fostering consumers' energy market through smart contracts. In *2017 International Conference in Energy and Sustainability in Small Developing Economies (ES2DE)*, pages 1–6. IEEE, 2017.
218. M. Kranz, L. Roalter, and F. Michahelles. Things that Twitter: social networks and the Internet of Things. In *Proc. of the International Workshop on Pervasive Computing (Pervasive 2010)*, pages 1–10, Helsinki, Finland, 2010.
219. B. Krishnamurthy, P. Gill, and M. Arlitt. A few chirps about Twitter. In *Proc. of the First Workshop on Online Social Networks (WOSN'2008)*, pages 19–24, Seattle, WA, USA, 2008.
220. H. Kumarage, I. Khalil, Z. Tari, and A. Zomaya. Distributed anomaly detection for industrial wireless sensor networks based on fuzzy data modelling. *Journal of Parallel and Distributed Computing*, 73(6):790–806, 2013. Elsevier.
221. M. Kurant, A. Markopoulou, and P. Thiran. On the bias of BFS (Breadth First Search). In *Proc. of the International Teletraffic Congress (ITC'2010)*, pages 1–8, Amsterdam, The Netherlands, 2010. IEEE.
222. Philipp Lämmel, Nikolay Tcholtchev, and Ina Schieferdecker. Enhancing cloud based data platforms for smart cities with authentication and authorization features. In *Companion Proceedings of the10th International Conference on Utility and Cloud Computing*, pages 167–172. ACM, 2017.
223. N. Laskowski. Data lake governance: A big data do or die. *URL: http://searchcio. techtarget. com/feature/Data-lake-governance-A-big-data-do-or-die (access date 28/05/2016)*, 2016.
224. Q.V. Le and T. Mikolov. Distributed Representations of Sentences and Documents. In *Proc. of the International Conference on Machine Learning (ICML'14)*, pages 1188–1196, Beijing, China, 2014. JMLR.org.
225. M.L. Lee, L.H. Yang, W. Hsu, and X. Yang. XClust: clustering XML schemas for effective integration. In *Proc. of the ACM International Conference on Information and Knowledge Management (CIKM 2002)*, pages 292–299, McLean, Virginia, USA, 2002. ACM Press.
226. G. Li, B.C. Ooi, J. Feng, J. Wang, and L. Zhou. EASE: an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data. In *Proc. of the International Conference on Management of data (SIGMOD/PODS'08)*, pages 903–914, Vancouver, Canada, 2008. ACM.
227. M. Li, H. Chen, X. Huang, and L. Cui. EasiCrawl: A Sleep-Aware Schedule Method for Crawling IoT Sensors. In *Proc. of the International Conference on Parallel and Distributed Systems (ICPADS'2015)*, pages 148–155, Melbourne, Australia, 2015. IEEE.

228. Ming Li and Alessio Bonti. T-osn: a trust evaluation model in online social networks. In *2011 IFIP Ninth International Conference on Embedded and Ubiquitous Computing*, pages 469–473. IEEE, 2011.

229. X. Li, Y. Tian, F. Smarandache, and R. Alex. An extension collaborative innovation model in the context of big data. *International Journal of Information Technology & Decision Making*, 14(01):69–91, 2015. World Scientific.

230. X. Li, Y. Wang, F. Shi, and W. Jia. Crawler for Nodes in the Internet of Things. *ZTE Communications*, 3:009, 2015.

231. Kaitai Liang, Liming Fang, Willy Susilo, and Duncan S Wong. A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security. In *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, pages 552–559. IEEE, 2013.

232. Xiaohui Liang, Zhenfu Cao, Huang Lin, and Jun Shao. Attribute based proxy re-encryption with delegating capabilities. pages 276–286, 01 2009.

233. C. Lin, G. Li, Z. Shan, and Y. Shi. Thinking and Modeling for Big Data from the Perspective of the I Ching. *International Journal of Information Technology & Decision Making*, 16(06):1451–1463, 2017. World Scientific.

234. C. Lin, J. Wang, and C. Rong. Towards heterogeneous keyword search. In *Proc. of the ACM Turing 50th Celebration Conference-China (ACM TUR-C'17)*, page 46, Shanghai, China, 2017. ACM.

235. J. Liu, X. Zhang, and L. Zhang. Tree pattern matching in heterogeneous fuzzy XML databases. *Knowledge-Based Systems*, 122:119–130, 2017. Elsevier.

236. L. Liu, H. Ma, D. Tao, and D. Zhang. A hierarchical cooperation model for sensor networks supported cooperative work. In *Proc. of the International Conference on Computer Supported Cooperative Work in Design(CSCWD'06)*, pages 1–6, Nanjing, China, 2006. IEEE.

237. Z. Liu, P. Sun, and Y. Chen. Structured search result differentiation. *Proceedings of the VLDB Endowment*, 2(1):313–324, 2009. VLDB Endowment.

238. P. Lo Giudice, P. Russo, and D. Ursino. A new Social Network Analysis-based approach to extracting knowledge patterns about research activities and hubs in a set of countries. *International Journal of Business Innovation and Research*, 2017. Inderscience. Forthcoming.

239. Hal Lockhart and B Campbell. Security assertion markup language (saml) v2. 0 technical overview. *OASIS Committee Draft*, 2:94–106, 2008.

240. Federico Lombardi, Leonardo Aniello, Stefano De Angelis, Andrea Margheri, and Vladimiro Sassone. A blockchain-based infrastructure for reliable and cost-effective iot-aided smart grids. 2018.

241. Stine Lomborg and Anja Bechmann. Using apis for data collection on social media. *The Information Society*, 30(4):256–265, 2014.

242. L. Lovász. Random walks on graphs: A survey. In *Combinatorics, Paul Erdos is Eighty*, pages 1–46. 1993. Springer.

243. H. Ma and W. Liu. Progressive Search Paradigm for Internet of Things. *IEEE MultiMedia*, Forthcoming. IEEE.

244. C. Madera and A. Laurent. The next information architecture evolution: the data lake wave. In *Proc. of the International Conference on Management of Digital EcoSystems (MEDES'16)*, pages 174–180, Hendaye, France, 2016. ACM.

245. J. Madhavan, P.A. Bernstein, and E. Rahm. Generic schema matching with Cupid. In *Proc. of the International Conference on Very Large Data Bases (VLDB 2001)*, pages 49–58, Rome, Italy, 2001. Morgan Kaufmann.

246. T. Maekawa, Y. Yanagisawa, Y. Sakurai, Y. Kishino, K. Kamei, and T. Okadome. Context-aware web search in ubiquitous sensor environments. *ACM Transactions on Internet Technology (TOIT)*, 11(3):12, 2012. ACM.

247. Damiano Di Francesco Maesa, Paolo Mori, and Laura Ricci. Blockchain based access control. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 206–220. Springer, 2017.

248. Imran Makhdoom, Ian Zhou, Mehran Abolhasan, Justin Lipman, and Wei Ni. Privysharing: A blockchain-based framework for privacy-preserving and secure data sharing in smart cities. *Computers & Security*, 88:101653, 2020.

249. S. Mallat, E. Hkiri, M. Maraoui, and M. Zrigui. Semantic Network Formalism for Knowledge Representation: Towards Consideration of Contextual Information. *International Journal on Semantic Web and Information Systems (IJSWIS'15)*, 11(4):64–85, 2015. IGI Global.

250. B. Malysiak-Mrozek, M. Stabla, and D. Mrozek. Soft and Declarative Fishing of Information in Big Data Lake. *IEEE Transactions on Fuzzy Systems*, 2018. IEEE.

251. C.D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval.* 2008. Cambridge University Press Cambridge.

252. Noortje Marres and Esther Weltevrede. Scraping the social? issues in live social research. *Journal of cultural economy*, 6(3):313–335, 2013.

253. J. Martinez-Gil and J.F. Aldana-Montes. Semantic similarity measurement using historical google search patterns. *Information Systems Frontiers*, 15(3):399–410, 2013. Springer.

254. Hartwig Mayer. ECDSA security in bitcoin and ethereum: a research survey. *CoinFaabrik, June*, 28, 2016.

255. Catherine A Meadows. Analyzing the needham-schroeder public key protocol: A comparison of two approaches. In *European Symposium on Research in Computer Security*, pages 351–364. Springer, 1996.

256. O. Mehdi, H. Ibrahim, and L. Affendey. An approach for instance based schema matching with Google similarity and regular expression. *International Arab Journal of Information Technology*, 14(5), 2017.

257. M. Mejri and J. Akaichi. A survey of textual event extraction from social networks. In *Proc. of the International Conference on Language Processing and Knowledge Management (LPKM'17)*, page 1988, Kerkenah Sfax, Tunisia, 2017.

258. Esther Mengelkamp, Benedikt Notheisen, Carolin Beer, David Dauer, and Christof Weinhardt. A blockchain-based smart grid: towards sustainable local energy markets. *Computer Science-Research and Development*, 33(1-2):207–214, 2018.

259. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

260. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. of the International Conference on Advances in Neural Information Processing Systems (NIPS'13)*, pages 3111–3119, Lake Tahoe, NV, USA, 2013.

261. Stanley Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

262. A.G. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

263. Justin J Miller. Graph database applications and concepts with neo4j. In *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA*, volume 2324, 2013.

264. N. Miloslavskaya and A. Tolstoy. Big data, fast data and data lake concepts. *Procedia Computer Science*, 88:300–305, 2016. Elsevier.

265. D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7):1497–1516, 2012. Elsevier.

266. Andres Monzon. Smart cities concept and challenges: Bases for the assessment of smart city projects. In *2015 international conference on smart cities and green ICT systems (SMARTGREENS)*, pages 1–11. IEEE, 2015.

267. S. Murugesh and A. Jaya. Representing Natural Language Sentences in RDF Graphs to Derive Knowledge Patterns. In *Proc. of the International Conference on Data Engineering and Communication Technology (ICDECT'17)*, pages 701–707, Maharashtra, India, 2017. Springer.

268. Nitin Naik and Paul Jenkins. Securing digital identities in the cloud by selecting an apposite federated identity management from saml, oauth and openid connect. In *2017 11th International Conference on Research Challenges in Information Science (RCIS)*, pages 163–174. IEEE, 2017.

269. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

270. A. Nandi and P.A. Bernstein. HAMSTER: Using Search Clicklogs for Schema and Taxonomy Matching. *Proceedings of the VLDB Endowment*, 2(1):181–192, 2009.

271. R. Navigli and S.P. Ponzetto. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012. Elsevier.

272. Surya Nepal, Sanat Kumar Bista, and Cecile Paris. Behavior-based propagation of trust in social networks with restricted and anonymous participation. *Computational Intelligence*, 31(4):642–668, 2015.

273. Surya Nepal, Wanita Sherchan, and Cecile Paris. Strust: A trust model for social networks. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, pages 841–846. IEEE, 2011.

274. Q.V.H. Nguyen, T.T. Nguyen, V.T. Chau, T.K. Wijaya, Z. Miklos, K. Aberer, A. Gal, and M. Weidlich. SMART: A tool for analyzing and reconciling schema matching networks. In *Proc. of the International Conference on Data Engineering (ICDE'15)*, number EPFL-CONF-203576, Seoul, Korea, 2015.
275. H. Ning and Z. Wang. Future internet of things architecture: like mankind neural system or social organization framework? *IEEE Communications Letters*, 15(4):461–463, 2011. IEEE.
276. A. Nocera and D. Ursino. PHIS: a system for scouting potential hubs and for favoring their “growth” in a Social Internetworking Scenario. *Knowledge-Based Systems*, 36:288–299, 2012. Elsevier.
277. Alex Norta. Designing a smart-contract application layer for transacting decentralized autonomous organizations. In *International Conference on Advances in Computing and Data Sciences*, pages 595–604, 2016. `http://www.ctan.org/pkg/subcaption`.
278. Amanda Nosko, Eileen Wood, and Seija Molema. All about me: Disclosure in online social networking profiles: The case of facebook. *Computers in Human Behavior*, 26(3):406–418, 2010.
279. C. Olston and M. Najork. Web crawling. *Foundations and Trends*, 4(3):175–246, 2010. Now Publishers, Inc.
280. A. Oram. *Managing the Data Lake*. Sebastopol, CA, USA, 2015. O'Reilly.
281. A. Ortiz, D. Hussein, S. Park, S. Han, and N. Crespi. The cluster between internet of things and social networks: Review and research challenges. *IEEE Internet of Things Journal*, 1(3):206–215, 2014. IEEE.
282. Aleksi Paaso, Daniel Kushner, Shay Bahramirad, and Amin Khodaei. Grid modernization is paving the way for building smarter cities [technology leaders]. *IEEE Electrification Magazine*, 6(2):6–108, 2018.
283. L. Palopoli, D. Rosaci, G. Terracina, and D. Ursino. A graph-based approach for extracting terminological properties from information sources with heterogeneous formats. *Knowledge and Information Systems*, 8(4):462–497, 2005.
284. L. Palopoli, D. Saccà, G. Terracina, and D. Ursino. Uniform techniques for deriving similarities of objects and subschemes in heterogeneous databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(2):271–294, 2003.
285. L. Palopoli, G. Terracina, and D. Ursino. DIKE: a system supporting the semi-automatic construction of Cooperative Information Systems from heterogeneous databases. *Software Practice & Experience*, 33(9):847–884, 2003.
286. L. Palopoli, G. Terracina, and D. Ursino. Experiences using DIKE, a system for supporting cooperative information system and data warehouse design. *Information Systems*, 28(7):835–865, 2003.
287. J.-R. Park. Metadata quality in digital repositories: A survey of the current state of the art. *Cataloging & Classification Quarterly*, 47(3-4):213–228, 2009.
288. J.-R. Park and Y. Tosaka. Metadata quality control in digital repositories and collections: Criteria, semantics, and mechanisms. *Cataloging & Classification Quarterly*, 48(8):696–715, 2010.

289. K. Passi, L. Lane, S.K. Madria, B.C. Sakamuri, M.K. Mohania, and S.S. Bhowmick. A model for XML Schema integration. In *Proc. of the International Conference on E-Commerce and Web Technologies (EC-Web 2002)*, pages 193–202, Aix-en-Provence, France, 2002. Lecture Notes in Computer Science, Springer.

290. A. Patel and T.A. Champaneria. Fuzzy logic based algorithm for Context Awareness in IoT for Smart home environment. In *Proc. of the International Conference on Region 10 Conference (TENCON '16)*, pages 1057–1060, Singapore, 2016. IEEE.

291. M. Patella and P. Ciaccia. Approximate similarity search: A multi-faceted problem. *Journal of Discrete Algorithms*, 7(1):36–48, 2009. Elsevier.

292. Sancheng Peng, Aimin Yang, Lihong Cao, Shui Yu, and Dongqing Xie. Social influence modeling using information theory in mobile social networks. *Information Sciences*, 379:146–159, 2017.

293. C. Perera, Y. Qin, J. Estrella, S. Reiff-Marganiec, and A. Vasilakos. Fog computing for sustainable smart cities: A survey. *ACM Computing Surveys (CSUR)*, 50(3):32, 2017. ACM.

294. C. Perera and A. Vasilakos. A knowledge-based resource discovery for Internet of Things. *Knowledge-Based Systems*, 109:122–136, 2016. Elsevier.

295. C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the Internet of Things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1):414–454, 2014. IEEE.

296. Gareth W Peters and Efstathios Panayi. Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In *Banking Beyond Banks and Money*, pages 239–278. Springer, 2016.

297. Marc Pilkington. 11 blockchain technology: principles and applications. *Research handbook on digital transformations*, page 225, 2016.

298. Jelena Ponoćko and Jovica V Milanović. Forecasting demand flexibility of aggregated residential load using smart meter data. *IEEE Transactions on Power Systems*, 33(5):5446–5455, 2018.

299. D. V. Prasad, S. Madhusudanan, and S. Jaganathan. uclust - a new algorithm for clustering unstructured data. 10(5):2108–2117, 2015. ARPN Journal of Engineering and Applied Sciences.

300. Eric Prud, Andy Seaborne, et al. Sparql query language for rdf. 2006.

301. Xiu-Quan Qiao, Chun Yang, Xiao-Feng Li, and Jun-Liang Chen. A trust calculating algorithm based on social networking service users' context. *Jisuanji Xuebao(Chinese Journal of Computers)*, 34(12):2403–2413, 2011.

302. Y. Qin, Q. Sheng, N. Falkner, S. Dustdar, H. Wang, and A. Vasilakos. When things matter: A survey on data-centric internet of things. *Journal of Network and Computer Applications*, 64:137–153, 2016. Elsevier.

303. E. Rahm and P.A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

304. Sandi Rahmadika, Diena Rauda Ramdania, and Maisevli Harika. Security analysis on the decentralized energy trading system using blockchain technology. *Jurnal Online Informatika*, 3(1):44–47, 2018.

305. Devakunchari Ramalingam and Valliyammai Chinnaiah. Fake profile detection techniques in large-scale online social networks: A comprehensive review. *Computers & Electrical Engineering*, 65:165–177, 2018.

306. S. Rangarajan, H. Liu, H. Wang, and C.L. Wang. Scalable Architecture for Personalized Healthcare Service Recommendation Using Big Data Lake. In *Service Research and Innovation*, pages 65–79. 2015. Springer.

307. A.H. Rasti, M. Torkjazi, R. Rejaie, and D. Stutzbach. Evaluating Sampling Techniques for Large Dynamic Graphs. *Univ. Oregon, Tech. Rep. CIS-TR-08-01*, 2008.

308. C. Razafimandimby, V. Loscri, and A.M. Vegni. A neural network and iot based scheme for performance assessment in internet of robotic things. In *Proc. of the International Conference on Internet-of-Things Design and Implementation (IoTDI'16)*, pages 241–246, Orlando, USA, 2016. IEEE.

309. Leonard Richardson, Mike Amundsen, and Sam Ruby. *RESTful Web APIs: Services for a Changing World*. " O'Reilly Media, Inc.", 2013.

310. Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In *International workshop on fast software encryption*, pages 371–388. Springer, 2004.

311. Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

312. J TêŐň Rotruck, A LêŐň Pope, H Eu Ganther, AB Swanson, D Gr Hafeman, and WG1973 Hoekstra. Selenium: biochemical role as a component of glutathione peroxidase. *Science*, 179(4073):588–590, 1973.

313. F. Sadeghi, S.K. Kumar Divvala, and A. Farhadi. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPM'15)*, pages 1456–1464, Boston, MA, USA, 2015.

314. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 457–473. Springer, 2005.

315. M. Sahlgren and R. Cöster. Using Bag-of-Concepts to Improve the Performance of Support Vector Machines in Text Categorization. In *Proc. of the International Conference on Computational Linguistics (COLING'04)*, Geneva, Switzerland, 2004.

316. M. Sahlgren and J. Karlgren. Automatic bilingual lexicon acquisition using random indexing of parallel corpora. *Natural Language Engineering*, 11(3):327–341, 2005.

317. Ryuichi Sakai and Masao Kasahara. ID based Cryptosystems with Pairing on Elliptic Curve. *IACR Cryptology ePrint Archive*, page 54, 2003.

318. A. Salazar and L. Zhao. Rhythmic Pattern Extraction by Community Detection in Complex Networks. In *Proc. of the International Conference on Intelligent Systems (BRACIS'14)*, pages 396–401, Sao Paulo, Brazil, 2014. IEEE.

319. Mauro San Martın, Claudio Gutierrez, and Peter T Wood. Snql: A social networks query and transformation language. *cities*, 5:r5, 2011.

320. A.F. Santamaria, A. Serianni, P. Raimondo, F. De Rango, and M. Froio. Smart wearable device for health monitoring in the internet of things (IoT) domain. In *Proc. of the International Conference on Proceedings of the summer computer simulation conference(SCSC'16)*, page 36, Montreal, Canada, 2016. Society for Computer Simulation International.

321. Takayuki Sasaki, Yusuke Morita, and Astha Jada. Access control architecture for smart city iot platform. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 717–722. IEEE, 2019.

322. S.F. Sayeedunnissa, A.R. Hussain, and M.A. Hameed. Supervised Opinion Mining of Social Network Data Using a Bag-of-Words Approach on the Cloud. In *Proc. of the International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA'12)*, pages 299–309, Gwalior, India, 2012.

323. C. Schoch. Big? Smart? Clean? Messy? Data in the Humanities. *Journal of Digital Humanities*, 2(3):2–13, 2013.

324. Z. Shang, Y. Liu, G. Li, and J. Feng. K-join: Knowledge-aware similarity join. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3293–3308, 2016. IEEE.

325. A. Shemshadi, Q. Sheng, and Y. Qin. Thingseek: A crawler and search engine for the internet of things. In *Proc. of the International Conference on Research and Development in Information Retrieval (SIGIR'2016)*, pages 1149–1152, Pisa, Italy, 2016. ACM.

326. A. Shemshadi, L. Yao, Y. Qin, Q. Sheng, and Y. Zhang. Ecs: A framework for diversified and relevant search in the internet of things. In *Proc. of the International Conference on Web Information Systems Engineering (WISE'2015)*, pages 448–462, Miami, Florida, USA, 2015. Springer.

327. Lei Sheng, Z Meral Ozsoyoglu, and Gultekin Ozsoyoglu. A graph query language and its query processing. In *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*, pages 572–581. IEEE, 1999.

328. Wanita Sherchan, Surya Nepal, and Cecile Paris. A survey of trust in social networks. *ACM Computing Surveys (CSUR)*, 45(4):47, 2013.

329. Pierluigi Siano. Demand response and smart gridsâĂŤa survey. *Renewable and sustainable energy reviews*, 30:461–478, 2014.

330. A. Silva and C. Antunes. Constrained pattern mining in the new era. *Knowledge and Information Systems*, 47(3):489–516, 2016. Springer.

331. Rôney Reis C Silva, Bruno C Leal, Felipe T Brito, Vânia MP Vidal, and Javam C Machado. A differentially private approach for querying rdf data of social networks. In *Proceedings of the 21st International Database Engineering & Applications Symposium*, pages 74–81. ACM, 2017.

332. K. Singh, K. Paneri, A. Pandey, G. Gupta, G. Sharma, P. Agarwal, and G. Shroff. Visual bayesian fusion to navigate a data lake. In *Proc. of the International Conference on Information Fusion (FUSION'16)*, pages 987–994, 2016. IEEE.

333. K. Singh and V. Singh. Answering graph pattern query using incremental views. In *Proc. of the International Conference on Computing (ICCCA'16)*, pages 54–59, Greater Noida, India, 2016. IEEE.

334. Florian Skopik, Zhendong Ma, Thomas Bleier, and Helmut Grüneis. A survey on threats and vulnerabilities in smart metering infrastructures. *International Journal of Smart Grid and Clean Energy*, 1(1):22–28, 2012.

335. Ye-Byoul Son, Jong-Hyuk Im, Hee-Yong Kwon, Seong-Yun Jeon, and Mun-Kyu Lee. Privacy-preserving peer-to-peer energy trading in blockchain-enabled smart grids using functional encryption. *Energies*, 13(6):1321, 2020.

336. S. Spaccapietra and C. Parent. View integration: A step forward in solving structural conflicts. *IEEE Transactions on Knowledge and Data Engineering*, 6(2):258–274, 1994.

337. M. Steichen, B. Fiz, R. Norvill, W. Shbair, and R. State. Blockchain-based, decentralized access control for ipfs. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1499–1506, July 2018.

338. Mathis Steichen, Beltran Fiz, Robert Norvill, Wazen Shbair, and Radu State. Blockchain-based, decentralized access control for ipfs. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1499–1506. IEEE, 2018.

339. I. Stojmenovic and S. Olariu. Data-centric protocols for wireless sensor networks. *Handbook of sensor networks: algorithms and architectures*, pages 417–456, 2005. Wiley.

340. D. Stutzback, R. Rejaie, N. Duffield, S. Sen, and W. Willinger. On unbiased sampling for unstructured peer-to-peer networks. In *Proc. of the International Conference on Internet Measurements (IMC'2006)*, pages 27–40, Rio De Janeiro, Brasil, 2006. ACM.

341. K. Su, J. Li, and H. Fu. Smart city and the applications. In *2011 International Conference on Electronics, Communications and Control (ICECC)*, pages 1028–1031, Sep. 2011.

342. D. Sun, G. Zhang, S. Yang, W. Zheng, S.U. Khan, and K. Li. Re-Stream: Real-time and energy-efficient resource scheduling in big data stream computing environments. *Information Sciences*, 319:92–112, 2015.

343. Jianjun Sun, Jiaqi Yan, and Kem ZK Zhang. Blockchain-based sharing services: What blockchain technology can contribute to smart cities. *Financial Innovation*, 2(1):26, 2016.

344. Melanie Swan. *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

345. J. Szymanski. Comparative Analysis of Text Representation Methods Using Classification. *Cybernetics and Systems*, 45(2):180–199, 2014.

346. Souhaib Ben Taieb, James W Taylor, and Rob J Hyndman. Hierarchical probabilistic forecasting of electricity demand with smart meter data. *Journal of the American Statistical Association*, pages 1–17, 2020.

347. P.P. Talukdar, Z. Ives, and F. Pereira. Automatically incorporating new sources in keyword search-based data integration. In *Proc. of the International Conference on SIGMOD International Conference on Management of data (SIGMOD/PODS'10)*, pages 387–398, Indianapolis, IN, USA, 2010. ACM.

348. A. Tani, L. Candela, and D. Castelli. Dealing with metadata quality: The legacy of digital library efforts. *Information Processing Management*, 49(6):1194–1205, 2013.

349. K. Tei and L. Gurgen. ClouT: Cloud of things for empowering the citizen clout in smart cities. In *Proc. of the World Forum on Internet of Things (WF-IoT'2014)*, pages 369–370, Seoul, South Korea, 2014. IEEE.

350. I.G. Terrizzano, P.M. Schwarz, M. Roth, and J.E. Colino. Data Wrangling: The Challenging Journey from the Wild to the Lake. In *Proc. of the International Conference on Innovative Data Systems Research (CIDR'15)*, Asilomar, CA, USA, 2015.

351. AK Thushar and P Santhi Thilagam. An rdf approach for discovering the relevant semantic associations in a social network. In *Proceedings of the 16th International Conference on Advanced Computing and Communications (ADCOM)*, pages 214–220. IEEE, 2008.

352. Sergei Tikhomirov. Ethereum: state of knowledge and research perspectives. In *International Symposium on Foundations and Practice of Security*, pages 206–221. Springer, 2017.

353. Vijay Tiwari. Analysis and detection of fake profile over social network. In *Computing, Communication and Automation (ICCCA), 2017 International Conference on*, pages 175–179. IEEE, 2017.

354. N. Tran, Q. Sheng, M. Babar, and L. Yao. Searching the Web of Things: State of the Art, Challenges, and Solutions. *ACM Computing Surveys (CSUR)*, 50(4):55, 2017. ACM.

355. C. Tsai, C. Lai, and A. Vasilakos. Future Internet of Things: open issues and challenges. *Wireless Networks*, 20(8):2201–2217, 2014. Springer.

356. Niko Tsakalakis, Sophie Stalla-Bourdillon, and Kieron O'hara. What's in a name: the conflicting views of pseudonymisation under eidas and the general data protection regulation. 2016.

357. M. Tubaishat, J. Yin, B. Panja, and S. Madria. A secure hierarchical model for sensor network. *ACM Sigmod Record*, 33(1):7–13, 2004. ACM.

358. European Union. Regulation EU No 910/2014 of the European Parliament and of the Council, 23 July 2014. `http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX\%3A32014R0910&from=EN`.

359. C.J. Van Rijsbergen. *Information Retrieval*. 1979. Butterworth.

360. Friedhelm Victor and Bianca Katharina Lüders. Measuring ethereum-based erc20 token networks. In *International Conference on Financial Cryptography and Data Security*, pages 113–129. Springer, 2019.

361. W. Villegas-Ch, S. Luján-Mora, D. Buenaño-Fernandez, and X. Palacios-Pacheco. Big Data, the Next Step in the Evolution of Educational Data Analysis. In *Proc. of the International Conference on Information Theoretic Security (ICITS'18)*, pages 138–147, 2018. Springer.

362. C. Walker and H. Alrehamy. Personal data lake with data gravity pull. In *Proc. of the International Conference on Big Data and Cloud Computing (BDCloud'15)*, pages 160–167, Dalian, China, 2015. IEEE.

363. J. Wan, J. Liu, Z. Shao, A. Vasilakos, M. Imran, and K. Zhou. Mobile crowd sensing for traffic prediction in internet of vehicles. *Sensors*, 16(1):88, 2016. Multidisciplinary Digital Publishing Institute.

364. J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. Vasilakos. Software-defined industrial internet of things in the context of industry 4.0. *IEEE Sensors Journal*, 16(20):7373–7380, 2016. IEEE.

365. F. Wang, Z. Wang, Z. Li, and J.R. Wen. Concept-based Short Text Classification and Ranking. In *Proc. of the International Conference on Information and Knowledge Management (CIKM'14)*, pages 1069–1078, Shangai, China, 2014. ACM.

366. H. Wang, Z. Xu, H. Fujita, and S. Liu. Towards felicitous decision making: An overview on challenges and trends of Big Data. *Information Sciences*, 367:747–765, 2016.

367. J. Wang, J. Li, and J.X. Yu. Answering tree pattern queries using views: a revisit. In *Proc. of the International Conference on Extending Database Technology (EDBT/ICDT'11)*, pages 153–164, Uppsala, Sweden, 2011. ACM.

368. J. Wang and J.X. Yu. Revisiting answering tree pattern queries using views. *ACM Transactions on Database Systems*, 37(3):18, 2012. ACM.

369. Naiyu Wang, Xiao Zhou, Xin Lu, Zhitao Guan, Longfei Wu, Xiaojiang Du, and Mohsen Guizani. When energy trading meets blockchain in electrical power system: The state of the art. *Applied Sciences*, 9(8):1561, 2019.

370. Shangping Wang, Yinglong Zhang, and Yaling Zhang. A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access*, 6:38437–38450, 2018.

371. Shuai Wang, Yong Yuan, Xiao Wang, Juanjuan Li, Rui Qin, and Fei-Yue Wang. An overview of smart contract: architecture, applications, and future trends. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 108–113. IEEE, 2018.

372. Wenbo Wang, Dinh Thai Hoang, Zehui Xiong, Dusit Niyato, Ping Wang, Peizhao Hu, and Yonggang Wen. A survey on consensus mechanisms and mining management in blockchain networks. *arXiv preprint arXiv:1805.02707*, pages 1–33, 2018.

373. Mudasir Ahmad Wani and Suraiya Jabin. A sneak into the devil's colony-fake profiles in online social networks. *arXiv preprint arXiv:1705.09929*, 2017.

374. Doug Washburn, Usman Sindhu, Stephanie Balaouras, Rachel A Dines, N Hayes, and Lauren E Nelson. Helping cios understand âĂIJsmart cityâĂİ . *Growth*, 17(2):1–17, 2009.

375. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

376. Jian Wu, Ruoyun Xiong, and Francisco Chiclana. Uninorm trust propagation and aggregation methods for group decision making in social network with four tuple information. *Knowledge-Based Systems*, 96:29–39, 2016.

377. X. Wu, D. Theodoratos, and W.H. Wang. Answering XML queries using materialized views revisited. In *Proc. of the International Conference on Information and Knowledge Management (CIKM '09)*, pages 475–484, Hong Kong, China, 2009. ACM.

378. Karl Wüst and Arthur Gervais. Do you need a blockchain? In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 45–54. IEEE, 2018.

379. Cao Xiao, David Mandell Freeman, and Theodore Hwa. Detecting clusters of fake accounts in online social networks. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pages 91–101. ACM, 2015.

380. K. Xu, Y. Qu, and K. Yang. A tutorial on the internet of things: From a heterogeneous network integration perspective. *IEEE Network*, 30(2):102–108, 2016. IEEE.

381. Adamu Sani Yahaya, Nadeem Javaid, Fahad A Alzahrani, Amjad Rehman, Ibrar Ullah, Affaf Shahid, and Muhammad Shafiq. Blockchain based sustainable local energy trading considering home energy management and demurrage mechanism. *Sustainability*, 12(8):3385, 2020.

382. Alexander Yakubov, Wazen Shbair, and Radu State. BlockPGP: A Blockchain-based Framework for PGP Key Servers. In *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, pages 316–322. IEEE, 2018.

383. Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *IEEE communications surveys & tutorials*, 15(1):5–20, 2013.

384. S. Ye, J. Lang, and F. Wu. Crawling online social graphs. In *Proc. of the International Asia-Pacific Web Conference (APWeb'10)*, pages 236–242, Busan, Korea, 2010. IEEE.

385. Melike Yigit, V Cagri Gungor, and Selcuk Baktir. Cloud computing for smart grid applications. *Computer Networks*, 70:312–329, 2014.

386. Tan Yigitcanlar, Koray Velibeyoglu, and Cristina Martinez-Fernandez. Rising knowledge cities: the role of urban knowledge precincts. *Journal of knowledge management*, 12(5):8–20, 2008.

387. Y. Yuan, G. Wang, L. Chen, and B. Ning. Efficient pattern matching on big uncertain graphs. *Information Sciences*, 339:369–394, 2016. Elsevier.

388. Chenghua Zhang, Jianzhong Wu, Yue Zhou, Meng Cheng, and Chao Long. Peer-to-peer energy trading in a microgrid. *Applied Energy*, 220:1–12, 2018.

389. Y. Zhang, D. Raychadhuri, L. Grieco, E. Baccelli, J. Burke, R. Ravindran, G. Wang, A. Lindgren, B. Ahlgren, and O. Schelen. Requirements and Challenges for IoT over ICN. `https://tools.ietf.org/html/draft-zhang-icnrg-icniot-requirements-00`, 2015. IETF Internet-Draft.

390. Y. Zhang, D. Raychadhuri, R. Ravindran, and G. Wang. ICN based Architecture for IoT. `https://tools.ietf.org/html/draft-zhang-iot-icn-challenges-02`, 2013. IRTF contribution.

391. Z. Zheng, Y. Ding, Z. Wang, and Z. Wang. A Novel Method of Keyword Query for RDF Data Based on Bipartite Graph. In *Proc. of the International Conference on Parallel and Distributed Systems (ICPADS'16)*, pages 466–473, Wuhan, China, 2016. IEEE.

392. J. Zhou, Z. Cao, X. Dong, and A. Vasilakos. Security and privacy for cloud-based IoT: Challenges. *IEEE Communications Magazine*, 55(1):26–33, 2017. IEEE.

393. Y. Zhuang, Y. Wang, J. Shao, L. Chen, W. Lu, J. Sun, B. Wei, and J. Wu. D-Ocean: an unstructured data management system for data ocean environment. *Frontiers of Computer Science*, 10(2):353–369, 2016. Springer.

394. Philip R Zimmermann. *The official PGP user's guide*. MIT press, 1995.