

How to exploit the Social Internet of Things: Query Generation Model and Device Profiles' Dataset

Claudio Marche^{a,b}, Luigi Atzori^{a,b}, Virginia Pilloni^{a,b}, Michele Nitti^{a,b}

^a*DIEE, University of Cagliari, UdR CNIT of Cagliari, Italy*

^b*National Telecommunication Inter University Consortium (CNIT), Research Unit of Cagliari, Italy*

Abstract

The future Internet of Things (IoT) will be characterized by an increasing number of object-to-object interactions for the implementation of distributed applications running in smart environments. The Social IoT (SIoT) is one of the possible paradigms that is proposed to make the objects' interactions easier by facilitating the search of services and the management of objects' trustworthiness. In this scenario, we address the issue of modeling the queries that are generated by the objects when fulfilling applications' requests that could be provided by any of the peers in the SIoT. To this, the defined model takes into account the objects' major features in terms of typology and associated functionalities, and the characteristics of the applications. We have then generated a dataset, by extracting objects' information and positions from the city of Santander in Spain. We have classified all the available devices according to the FIWARE Data Models, so as to enable the portability of the dataset among different platforms. The dataset and the proposed query generation model are made available to the research community to study the navigability of the SIoT network, with an application also to other IoT networks. Experimental analyses have also been conducted, which give some key insights on the impact of the query model parameters on the average number of hops needed for each search.

Keywords:

Social Internet of Things; query generation model; dataset; test management algorithms

1. Introduction

2 The Internet of Things (IoT) has become a reality with billions of devices
3 able to send key information about the physical world and implementing simple
4 actions, which leads to the paradigm of the anytime and anyplace connectivity
5 for anything [1]. The massive amount of data flowing through the IoT has
6 pushed forward the development of new applications in several domains, such
7 as the management of industrial production plants, the logistics and transport
8 supply chain, the e-health, the smart building, just to cite a few.

9 However, IoT solutions have posed new challenges in the management of
10 the amount of information produced. Indeed, searching for (reliable) time-
11 and location- relevant information, services and resources for the deployment of
12 running applications exploiting the IoT infrastructure is a crucial challenge: in
13 addition to the size of the searching space, most of the data produced by the sensors
14 produce rapid changes, making the system highly dynamic, as it happens
15 for instance when tracking the position of moving objects. A further complication
16 derives from the shift we are witnessing in the interaction model. From
17 a paradigm where humans look for information provided by objects (human-
18 object interaction), the IoT will surely move towards a model where things look
19 for other things to provide composite services for the benefit of human beings
20 (object-object interaction). With such an interaction model, it will be essential
21 to understand how the information provided by each object can be processed
22 automatically by any other peer in the system. This cannot clearly disregard
23 the level of trustworthiness of the object providing information and services,
24 which should take into account the profile and history of it. If not, attacks and
25 malfunctions would outweigh any of the benefits of these technologies [2].

26 An approach with the potential to properly address the mentioned issues,
27 which is recently gaining increasing popularity, is based on the exploitation of
28 social networking notions into the IoT, as formalized by the Social IoT (SIoT)
29 concept [3]. It introduces the vision of social relationships among different devices,
30 independently from the fact that they belong to the same or different
31 platforms owned and managed by different individuals or organizations. According
32 to this vision, all the IoT objects are willing to collaborate with others
33 and create relationships among them as humans do. This is expected to make
34 the exchange of information and services among different devices easier and to
35 perform the identification of malicious nodes by creating a society-based view
36 about the trust level of each member of the community.

37 In the resulting social network, each application running in the devices (or in
38 the cloud) will be looking for information and services by crawling the social network
39 starting from a requesting node towards the potential service provider(s).
40 The performance of such a process of service/information retrieval is clearly dependent
41 on several aspects: i) the structure of the social network; ii) the types
42 of service/information requests that will mostly characterize the interaction in
43 the IoT/SIoT; iii) the rules that are used to navigate the network.

44 To analyze these aspects, there is a need for a proper model of the behavior
45 of objects that generate queries of services and information when interacting
46 with other peers in the SIoT. Such a model represents an alternative to the
47 analysis of large sets of real interactions among devices; indeed, this data is
48 not publicly available, so the model can represent a valid alternative for the
49 analysis of the networks. Understanding the interactions among peers can help
50 in discovering which devices are more likely to interact and then can assist in
51 the design of search engines, in the management of the trustworthiness or the
52 creation of clusters of nodes with frequent interactions. To this, the model
53 should take into account the objects' major features in terms of typology and
54 associated services, and the applications that may need to interact with the

55 different objects. Another essential element to test management algorithms is
56 represented by a suitable IoT dataset. It has to exhibit a realistic behavior based
57 on real IoT objects and show information regarding the position and the profile
58 of a large set of IoT devices, both public and private, both fixed and mobile,
59 arranged with their respective owner. All these objects need to be categorized
60 by their typology, brand and model, but also considering the set of services
61 they are able to offer and the possible applications they can request. The major
62 contributions of the paper are the following:

- 63 • Definition of the generic process of service search in the Social IoT and
64 modeling of objects behavior when interacting with other peers in the
65 network for the exchange of information and services.
- 66 • Definition of a query generation model, which is able to simulate the cor-
67 relation between objects and applications and represents a fundamental
68 tool to test the interaction among peers in the network. The proposed
69 model is then used to evaluate the benefits of the social approach in terms
70 of global navigability.
- 71 • Creation of a dataset, which not only include objects' information and
72 positions as done in [4], but also the services and applications they offer
73 and use. The collected data derives from the devices installed in the city
74 of Santander in Spain and on the data about people's mobility. This
75 is made available to the research community to test (S)IoT management
76 algorithms (e.g., relationship management, service search, trustworthiness
77 management), with particular attention to network navigability.

78 The paper is organized as follows. Section 2 introduces the relevant back-
79 ground regarding the social IoT and provides a short survey of the research
80 related to service search in the IoT. In Section 3 we provide an introduction to
81 the proposed solution by defining the scenario and introducing the used nota-
82 tions, whereas in Section 3.2 we propose a query generation model and test it
83 based on real traces. Section 5 presents a dataset based on real objects and uses
84 it to construct a social network of devices and Section 6 draws final remarks.

85 **2. Background**

86 *2.1. The Social IoT*

87 The SIoT represents the convergence of the technologies belonging to two
88 domains: IoT and Social Networking. The result is the creation of social net-
89 works in which things are nodes that establish social links as humans do [3].
90 This concept is fast gaining ground thanks to the key benefits deriving from the
91 potentials of the social networks within the IoT domain, such as: simplification
92 in the navigability of a dynamic network of billions of objects [3]; robustness in
93 the management of the trustworthiness of objects when providing information
94 and services [5]; efficiency in the dynamic discovery, selection and composition of

95 services (and of information segments) provided by distributed objects and net-
96 works [6]. According to the SIoT model, every node is an object that is capable
97 of establishing social relationships with other things autonomously, according
98 to rules set by the owner.

99 To this aim, as underlined in [7], there is a strong need to improve the degree
100 of connectivity between users and things, where things should be socialized
101 to allow humans to easily establish relationships with them. The resulting
102 paradigm of the Social Internet of Things (SIoT) [3] includes these notions, so
103 that people, through their IoT devices, can transparently (although according to
104 clear policies they have set for inter-device interactions) improve the experience
105 in the fruition of smart services and applications.

106 When it comes to the IoT paradigm, the idea is to exploit social awareness
107 as a means to turn communicating objects into autonomous decision-making en-
108 tities. The new social dimension shall, somehow, be able to mimic interactions
109 among users and to motivate a drift from an egoistic behavior to altruism or
110 reciprocity. The main principle is to enable objects to autonomously establish
111 social links with each other (by adhering to rules set by their owners) so that
112 “friend” objects exchange data in a distributed manner. Every network object
113 will be capable of: (a) establishing social relationships with other objects au-
114 tonomously with respect to the owner, but according to the preset rules for the
115 owner; (b) interact with its friends when in need for some assistance, such as
116 the provisioning of a piece of important information or a key service.

117 According to this model, a set of forms of socialization among objects is
118 foreseen. The parental object relationship (POR) is defined among similar ob-
119 jects, built in the same period by the same manufacturer (the role of the family
120 is played by the production batch). Moreover, objects can establish co-location
121 object relationship (CLOR) and co-work object relationship (CWOR), like hu-
122 mans do when they share personal (e.g., cohabitation) or public (e.g., work)
123 experiences. A further type of relationship is defined for objects owned by the
124 same user (mobile phones, game consoles, etc.) that is named ownership ob-
125 ject relationship (OOR). The last relationship is established when objects come
126 into contact, sporadically or continuously, for reasons purely related to relations
127 among their owners (e.g., devices/sensors belonging to friends); it is named so-
128 cial object relationship (SOR). These relationships are created and updated on
129 the basis of the objects features (such as type, computational power, mobility
130 capabilities, brand, etc.) and activities (frequency in meeting the other objects,
131 mainly).

132 However, to fully exploit the benefits of a SIoT network, realistic networks
133 and models of object-object interaction are still needed, which we investigate in
134 this paper.

135 *2.2. Query Generation models*

136 A common problem in the IoT is how to efficiently retrieve information
137 among the billions of devices composing it. Anytime there is the need to obtain
138 data from a given system, two essential elements are needed [8]:

- 139 • The objects, which are the entities that provide information through the
140 services they can offer.
- 141 • The query, which is the formal statement of the needed information (e.g.,
142 a search string).

143 The link between these two elements is represented by a search engine, which
144 has the goal to match the query with the data provided by the objects.

145 The IoT provides several approaches to searching services, i.e., to the de-
146 velopment of engines able to look for the required objects, such as [9] and [10].
147 These works proposed different mechanisms for both indexing and ranking of
148 the objects that can be used to search and select the objects and their offered
149 services. In the first work, the authors propose a context-aware sensor search,
150 based on a ranking model, to improve the selection of relevant sensors in large
151 sets; the context information related to each sensor can be used to search the
152 sensors in accordance with the user's requirements. The second paper focuses on
153 the requirements and challenges that need to be addressed to construct efficient
154 search engines.

155 Another approach is described in [11]: a framework composed of a con-
156 text module and a search engine is developed to interact with the IoT devices.
157 The context module is responsible to assign the semantic characteristics to the
158 devices while the search engine has to evaluate the users' queries, select and
159 interact with the objects, based on a proposed indexing technique.

160 Moreover, a recent work [12] analyses the most important IoT search engines
161 in the literature and introduces a classification. One of the major issues pointed
162 out by the authors is the lack of open datasets that contain IoT data and of
163 query models to test the proposed engines. These aspects are critical for the
164 community since they simplify carrying out the simulations and making them
165 repeatable.

166 Even if all these systems analyzed different features for the service search
167 systems, there are almost no works about the modeling of the query generation
168 process in the IoT, which, as said, represents a key component to test search
169 engines. Among these few works, in [13] the authors propose four basic query
170 models to search devices by their name, identifier, time and location of data.
171 Another example is provided in [14] which proposes spatial range queries with
172 location constraints to facilitate data indexing.

173 However, the problem of generating requests of information has been deeply
174 investigated for the World Wide Web, where the first approaches can be dated
175 back to the late '90s. Indeed, there is a strong similarity between searching the
176 Web, which is composed of a widely accessible, large and distributed source of
177 text data, i.e. documents, and the service search in IoT, whereas documents
178 can be seen as IoT applications and the words describing a document as the
179 services needed by an application. Accordingly, query generation methods used
180 to search the Web might be adapted to be used in the IoT.

181 Table 1 describes all the examined query generation methods. Among them,
182 the simplest approach is based on uniform distribution, where all the terms from
183 a given vocabulary have the same probability to occur in a query [23].

Table 1: Query Generation Methods.

Method	Description
Uniform [15]	Select terms with uniform probability from current vocabulary
Term-frequency [15]	Select most frequent words from a cluster of documents
Probabilistic term-frequency [16]	Select terms with probability proportional to their term frequency
Odds-ratio [17]	Select terms in according to maximum odds-ratio score
Probabilistic odds-ratio [18]	Select terms with probability proportional to their odds-ratio score
Boley [19]	Intersection of the top ranking terms according to term frequency and document frequency
Markov chain [20]	Method that allow to chose consecutively query terms from a first term
Poisson distribution [21]	Generate query using frequency words from a series of independent Poisson processes
Query expansion [22]	Process of reformulating query given by a user

184 Another approach is the term-frequency method, which classifies terms based
185 on their frequency in a cluster of documents [15]; only the terms above a given
186 threshold are considered, according to a uniform distribution, to compose a
187 query. The authors in [16] propose a variant of this method called probabilistic
188 term-frequency, where all the terms in the cluster are considered for a query,
189 but with a probability proportional to their frequency.

190 The authors in [17] propose an approach based on odds ratio; this score
191 measures how strong the probability of a term appearing in relevant documents
192 is w.r.t. the probability of the same term appearing in non-relevant documents.
193 Only the term with a ratio higher than a threshold can be selected to generate a
194 query. Similarly to [16], the authors in [18] improve the odds ratio method with
195 a probabilistic approach by assigning a probability proportional to the odds
196 ratio score and by selecting the terms according to this probability.

197 Another approach is proposed by Boley in [19]. The query terms are selected
198 in accordance to an intersection of two sets: the Text Frequency (TF) word list,
199 which refers to words frequency in a selected text and the Document Frequency
200 (DC) word list, which considers the terms' frequency in all the documents of a
201 cluster.

202 Authors in [20] show additional developments in query generations using
203 Markov chains. To estimate the query, they calculate the probability that a
204 word "translates" to a query term, and then they chose a document containing
205 the selected word, based on the number of times the word appears in it. The
206 Markov chain alternates the choices between words and documents iteratively
207 until the query is generated.

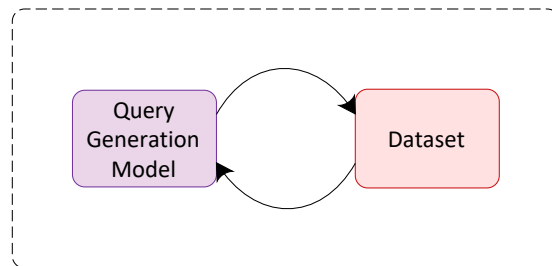


Figure 1: Proposed system to develop and test IoT management algorithms

208 In [21], the authors study a new family of query generation models based
 209 on the Poisson distribution. These models calculate the frequency of each term
 210 independently with a Poisson distribution. To rate a document, the authors
 211 estimate a multivariate Poisson model based on the document, and then give a
 212 score to it based on the likelihood of the query given by the estimated Poisson
 213 model.

214 In the last years, query generation techniques have developed methods to
 215 expand the query and change it to reformulate a given query with relevant in-
 216 formation so as to improve retrieval performance and to take into account the
 217 user profile [24]. A common technique consists of expanding the original query
 218 with synonyms: the new terms are chosen based on a probabilistic approach.
 219 In [22], the authors propose a relevance model, which assigns a relevance prob-
 220 ability to each word of the collection to measure how relevant a synonym could
 221 be for the terms in the original query.

222 All the cited techniques might not adapt directly to the IoT, but can be used
 223 as a starting point for the design of a query generation model for the IoT. In
 224 particular, in this paper we focus on the probabilistic methods and adapt them
 225 to the IoT scenario. Moreover, we validate the obtained model by making use
 226 of a real set of queries, obtained from the Lysis IoT platform [25].

227 3. Introduction to the proposed solutions

228 This paper aims to provide the two essential elements needed to develop and
 229 test management algorithms in an IoT ecosystem, namely a query generation
 230 system and a dataset of objects with realistic behavior. As it is depicted in
 231 Figure 1, even though these two elements can cooperate, i.e. the query gener-
 232 ation model can be tested by using the dataset, they exist and can be used
 233 independently. The details about their functionalities will be better explained
 234 in Section 4 (for the query generation model) and Section 5 (for the dataset).

235 In the following, we present the reference scenario and the needed notation
 236 to describe it.

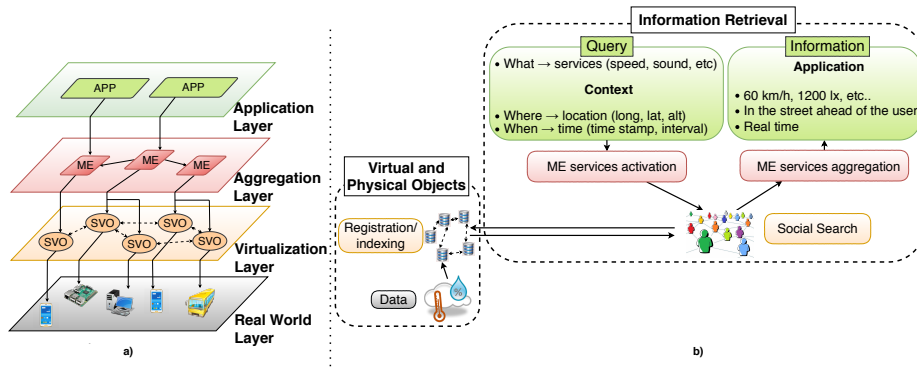


Figure 2: Reference SIoT architecture and query generation model.

3.1. Reference Scenario

The SIoT provides the objects with some capabilities typical of humans' behavior when looking for and providing information in their social communities. Accordingly, social relationships are created among objects, which are used when the peers are looking for help [3]. As in most of the IoT architectures, the owner has the control on which social interactions the objects are allowed to perform and which information and services can be shared with other peers. The applications installed by owners in their cloud space and that rely on their objects' capabilities often need to look for services provided by other objects. This results in queries that are managed by the SIoT by making use of objects' social connections through word of mouth.

The focus of the paper is the analysis and modeling of this query generation process. To this, we consider the reference SIoT architecture shown in Figure 2a), which is based on four levels [26]: Application, Aggregation, Virtualization and Real World. The lower layer is made up of the "things" of the real world, which have the role to sense the physical environment and provide data to the higher layers. The Virtualization layer is made of Social Virtual Objects (SVOs), which represent the digital counterparts of any entity of the real world enhanced with social capabilities, fully describing their characteristics and the services they are able to provide [27]. The Micro Engine (ME), which is the main entity of the Aggregation layer, is a mash-up of one or more SVOs and other MEs, and it is responsible for getting and processing information from SVOs into high-level services requested by applications at the higher level. Finally, the Application layer is installed in the Cloud and partially in the devices, so that applications can be deployed and executed exploiting one or more MEs.

Figure 2b) illustrates a generic service query in the SIoT, which highlights all the components involved in the process. The whole process starts when the application layer triggers some processing that requires to look for other services and then generates a relevant query. The query specifies *what* services are

266 required and it is enhanced with context parameters, which represent the appli-
267 cation requirements, such as a specific time (*when*) or a specific place (*where*).

268 The generated query is then handled at the Aggregation Layer, where the
269 needed MEs for data elaboration are activated. After this, the query is taken
270 over by the SVO of the device that triggered the process, which navigates its
271 social network in order to search for other SVOs that can offer the data re-
272 lated to the desired services by the application. Indeed, in the SIoT, each SVO
273 maintains information related to its friends and to the services that the corre-
274 sponding physical object can provide. In this sense, SVOs can be seen as atomic
275 registration/indexing servers. However, it is not the focus of the paper to design
276 an indexing mechanism of data.

277 In order to better explain the query process, an explanatory example is
278 presented here. Suppose that an object installed the application *RealTimeTraf-*
279 *ficEvaluationApp*, which evaluates the traffic of a specific street in real-time, i.e.
280 within a limited time interval with respect to the current time. Accordingly,
281 the object creates a query with the list of services needed to execute the ap-
282 plication and the requirements related, in this case, with the reference location
283 and time. The aggregation layer then activates the MEs associated with the
284 services and passes the query to the SVO, which looks for the objects, among its
285 social network, that can execute the services fulfilling the desired requirements.
286 Once they are found, the aggregation layer processes the result and provides the
287 requested information, i.e. the real-time traffic condition in the specified street,
288 to the user.

289 When all the services are retrieved, they are forwarded again to the Aggrega-
290 tion Layer which composes them through the activated MEs and finally provides
291 the result of the application back to the device that triggered the request.

292 The depicted scenario where objects collaborate by mashing their services
293 has great potentials as this allows for the deployment of powerful applications.
294 This is the case of objects (e.g. cars) that share information to decide on the
295 best route to get to a destination, objects that perform collaborative spectrum
296 sensing and objects that need to send alarms to all the people in a given area
297 to reach humans nearby, just to cite few examples. Reaching the right device(s)
298 with whom interact is a key task in this context, and the SIoT provides a po-
299 tentially effective approach to this by relying on the created social network.
300 However, to evaluate the relevant performance, there is the need to model the
301 generation of the query characterizing these scenarios, which should help in con-
302 ducting a proper system performance evaluation. Such a model should describe
303 which object (with relevant characteristics) would typically need to retrieve in-
304 formation from any other objects with other relevant features. Whereas the
305 query model that is proposed in the following is adopted to evaluate the perfor-
306 mance of the SIoT paradigm, it can be adopted for other IoT architectures as
307 well. Finally, since our goal is to model the objects behavior when requesting
308 services at the application layer, in this paper we do not consider how the query
309 is handled.

310 *3.2. Nodes and network modeling*

311 In our modeling, the set of nodes in the SIoT, i.e., the set of SVOs, is
 312 represented by $\mathcal{N} = \{n_1, \dots, n_i, \dots, n_I\}$ with cardinality I , where n_i represents a
 313 generic SVO. Its physical counterpart can be static or mobile with position $\mathbf{L}_i =$
 314 $[l_i^a, l_i^b]$, which can then be fixed or varying over time. In our problem setting,
 315 SVOs create social relations so let the resulting social network be described by
 316 an undirected graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$, where $\mathcal{E} \subseteq \{\mathcal{N} \times \mathcal{N}\}$ is the set of edges, each
 317 representing a social relation between a couple of nodes.

318 The modeling of our problem can not overlook the different typologies of
 319 objects in a network, since objects with different profiles can provide different
 320 services and are interested in different applications [28]. We then define the
 321 following sets: $\mathcal{T} = \{t_1, \dots, t_x, \dots, t_X\}$ as the set of possible typologies of objects,
 322 such as smartphones, cars, traffic lights, and others. For every typology t_x ,
 323 we define a set $\mathcal{B}_x = \mathcal{B}(t_x) = \{b_{x1}, \dots, b_{xy}, \dots, b_{xY}\}$ as the set of possible brands
 324 inside the typology t_x , while the set $\mathcal{M}_{xy} = \mathcal{M}(b_{xy}) = \{m_{xy1}, \dots, m_{xyz}, \dots, m_{xyZ}\}$
 325 represents the set of possible models for typology t_x and brand b_{xy} . All the
 326 possible models available in the network can then be described by the set $\bar{\mathcal{M}} =$
 327 $\{\cup_{\forall xy} \mathcal{M}_{xy}\}$, which allows us to define the following 2-tuple $\Gamma = \langle \mathcal{N}, \bar{\mathcal{M}} \rangle$, which
 328 associates to every node n_i the corresponding model of the device and thus
 329 enables also to infer the typology and the brand. This tuple will be useful to
 330 enable the creation of the parental relation (i.e. the POR defined in Section 2.1),
 331 which is based on these characteristics.

332 Then, we need to define the applications in the network, which are those that
 333 are requested during the querying process and the possible services provided by
 334 the nodes and that can satisfy the queries. Let $\mathcal{A} = \{a_1, \dots, a_w, \dots, a_W\}$ be the
 335 set of possible applications that can be installed by the devices in our scenario.
 336 However, applications do not run on all the devices but only on those they are
 337 meant to, so a single device will only have a subset of applications installed on
 338 it; we can then define the matrix $\mathbf{O} = [o_{iw}]$ where the generic element o_{iw} is
 339 equal to 1 if node n_i can potentially install application a_w and 0 otherwise.

340 Then, we define $\mathcal{S} = \{s_1, \dots, s_j, \dots, s_J\}$ as the set of services that can be
 341 performed by any node in the network and that can be used to compose the
 342 applications in \mathcal{A} . Thus, we can define the matrix $\mathbf{D} = [d_{ij}]$, where the generic
 343 element d_{ij} is equal to 1 if node n_i can provide service s_j and 0 otherwise.

344 It is true that both the matrices of installed applications and available ser-
 345 vices, namely \mathbf{O} and \mathbf{D} , should be related to the typology of the node, since
 346 it is the typology that determines the possible uses for an object. However,
 347 this approach is too simplistic since different nodes can offer different services
 348 and run different applications based on external characteristics related to their
 349 owner, such as privacy settings. Let us consider two users which own a smart-
 350 phone each: one of them is willing to share all the smartphone's services while
 351 the other one only one or two of them; similarly, even if the set of applications
 352 they can install on their smartphone is the same, they have decided to install
 353 different applications based on their interests.

354 To model how an application generates a query, let's recall that a query

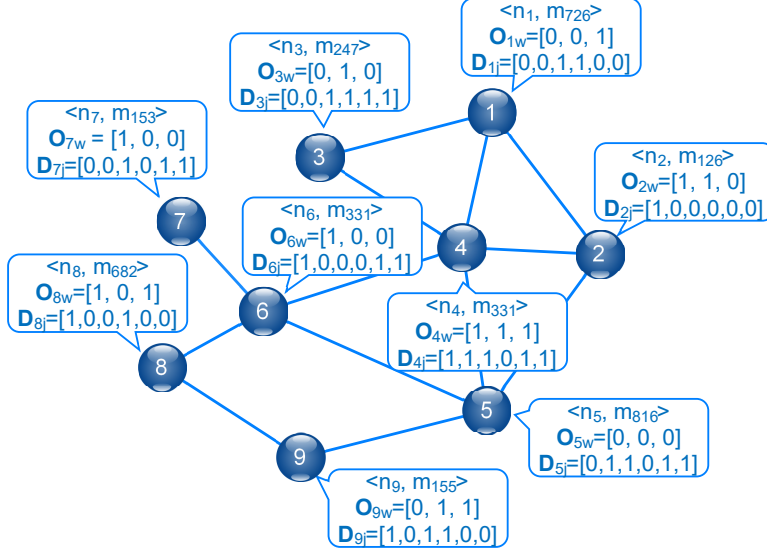


Figure 3: Representation of the network nodes.

355 only specifies the needed services and their requirements, and it is the aggregation layer that combines them to fulfill the request of the application. To
 356 this, we can model the query as the tuple $\Psi = \langle Q^{serv}, Q^{req} \rangle$, where $Q^{serv} =$
 357 $\{q_1^{serv}, \dots, q_h^{serv}, \dots, q_H^{serv}\}$ is the set of atomic queries representing the individual
 358 services needed to fulfill the application requests using a node's social network,
 359 while $Q^{req} = \{q_1^{req}, \dots, q_k^{req}, \dots, q_K^{req}\}$ is the set of requirements. The goal of a
 360 query generation model is then to calculate the probability to generate a specific
 361 query Ψ . In our modeling, we make the assumption that the number of
 362 atomic queries matches the number of services to be found; nonetheless, based
 363 on the particular search mechanism implemented in the (S)IoT, the number of
 364 queries can be lower w.r.t. the number of services, since a query can be used
 365 to find two or more services at the same time. However, the modeling of the
 366 search engine is not considered in this paper.
 367

368 Figure 3 provides a simple example of a generic SIoT graph \mathcal{G} , where $I = 9$
 369 and each node is characterized by a tuple $\Gamma_i = \langle n_i, m_{xyz} \rangle$, which defines for
 370 node n_i its model m_{xyz} , from which we can infer the typology t_x and the brand
 371 b_{xy} . In our example, we can notice how nodes can share the same typology, as
 372 it is the case of nodes n_2, n_7, n_9 , since they have the same first digit of the \mathcal{M}
 373 set, and even the same brand, as n_7 and n_9 are described by m_{153} and m_{155}
 374 respectively. In particular, if nodes belong to the same typology, brand and
 375 model, such as the case of nodes n_6 and n_4 , they are then able to create a POR.

376 In this example, each SVO can have up to 3 applications installed, as indicated
 377 by the number of columns of matrix \mathbf{O} , and it is capable of providing
 378 up to six services, as shown by the column dimension of \mathbf{D} . Suppose that a

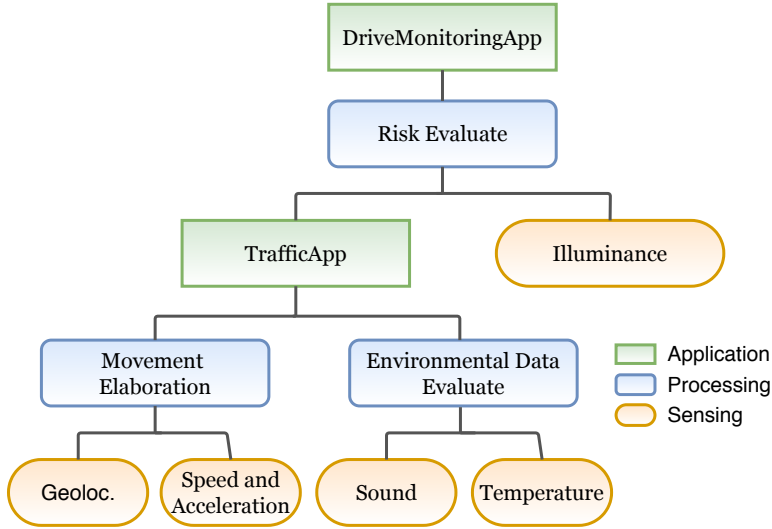


Figure 4: Decomposition of the *DriveMonitoringApp* application into services.

379 user, which owns node n_1 , is interested in the *DriveMonitoringApp* application
380 that monitors and evaluates his/her driving behavior and the related risks and
381 then installs it in n_1 . To provide the requested application, which is indicated
382 in our example as o_{13} , to the user, the related SVO will have to search for the
383 needed services, which are shown by the orange balloons in Figure 4 and that
384 are indicated as the services s_1 , s_2 , s_3 , s_5 and s_6 in our example scenario. Node
385 n_1 will then generate a query \mathcal{Q} with $\mathcal{Q}^{serv} = \{q_1^{serv}, q_2^{serv}, q_3^{serv}, q_4^{serv}, q_5^{serv}\}$
386 and $\mathcal{Q}^{req} = \emptyset$ to look for the five services among its friends. When all the ser-
387 vices are retrieved, they are sent to the aggregation layer, which provides the
388 necessary processing capabilities (blue balloons in Figure 4). Please note that
389 in some cases the node could be able to provide some of the services by itself as
390 the case of node n_1 , which can provide service s_3 ($d_{13} = 1$).

391 As we will see in the next section, the query generation model is more com-
392 plicated than this example, since it has to take into account space and time
393 requirements.

394 4. Query Generation Model and Simulation

395 In the next subsection, we illustrate our query generation model whereas in
396 the second subsection we evaluate the model performance in a real IoT scenario.

397 4.1. Query Generation Model

398 In the IoT, the number of possible applications is huge, but not all the types
399 of things can install the same set of applications and even the same application
400 installed in the same object can generate queries with different requirements.

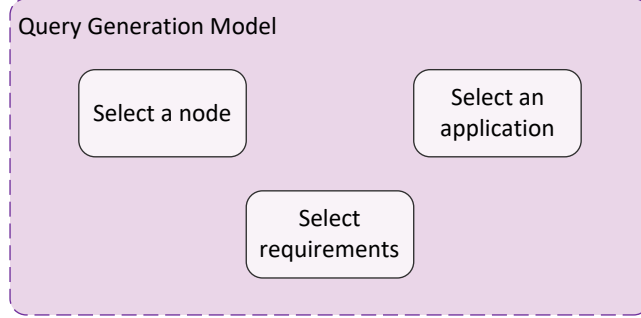


Figure 5: Query generation model functionalities

401 When studying the IoT, and in particular the Social IoT, it is difficult to
 402 evaluate the performance of service search mechanisms, i.e. how (S)IoT systems
 403 can fulfill application requests. This is due to the lack of query generation
 404 models, that are needed to understand which application can generate a query
 405 and with which requirements.

406 As described in Section 3.1, the goal of a query generation model is to
 407 compute the probability that a query \mathcal{Q} will be generated; the composition of the
 408 atomic queries in \mathcal{Q}^{serv} represents the set of services needed by the application.
 409 The choice of the application that will generate a query, and that will then
 410 determine the services to search, depends on the particular object in which the
 411 application is installed. Figure 5 shows the main functionalities that characterize
 412 the query generation model. According to this picture, based on the chosen
 413 application and on which node it is installed, the model has to generate the set
 414 of query requirements \mathcal{Q}^{req} , which are applied to the set of atomic queries.

415 Applications and nodes are highly intertwined: choosing a node determines
 416 which applications can be installed on that node, and selecting an application
 417 fixes the possible nodes in which the application can be installed. In order
 418 to obtain the probability to generate a specific set of atomic queries \mathcal{Q}^{serv} ,
 419 that corresponds to application $\mathcal{A} = a_w$, we have to compute the joint density
 420 function of nodes \mathcal{N} and application \mathcal{A} as follows:

$$p_{\mathcal{A}, \mathcal{N}}(a_w, n_i) = p(\mathcal{A} = a_w \cap \mathcal{N} = n_i) = \begin{cases} 0 & \text{if } o_{iw} = 0 \\ p_i(\mathcal{Q}^{serv}) & \text{if } o_{iw} = 1 \end{cases} \quad (1)$$

421 where $p_i(\mathcal{Q}^{serv})$ is the probability that node n_i , which potentially installed
 422 application a_w , generates the set of atomic queries \mathcal{Q}^{serv} . For $o_{iw} = 1$, it can
 423 also be written in terms of conditional distributions:

$$\begin{aligned} p_i(\mathcal{Q}^{serv}) &= p(\mathcal{N} = n_i | \mathcal{A} = a_w) * p(\mathcal{A} = a_w) = \\ &= p(\mathcal{A} = a_w | \mathcal{N} = n_i) * p(\mathcal{N} = n_i) \end{aligned} \quad (2)$$

424 Eq. 2 shows the double nature of the query generation process, which can

425 begin both by selecting an application or a node.

426 The probability that the set of atomic queries \mathcal{Q}^{serv} is generated by any
 427 node in \mathcal{N} is then defined as

$$P(\mathcal{Q}^{serv}) = \sum_i p_i(\mathcal{Q}^{serv}) \quad (3)$$

428 The application selection greatly influences the difficulty of the search op-
 429 erations, since applications can have different levels of intricacy, ranging from
 430 simple ones, which only need one or two services, to complex ones, with nested
 431 applications and multiple services. Moreover, not all applications require infor-
 432 mation with the same frequency. To this, in Section 5.3, we will test several
 433 different distributions for the applications' frequency, namely $p(\mathcal{A} = a_w)$, to
 434 evaluate how the SIoT network reacts in terms of navigability.

435 The choice of the node affects both its geographical and social position. The
 436 first one is important since it influences the requirements of the query, while
 437 the position of the node in the social network impacts on the number of friends
 438 selectable and thus on the number of friends a node can rely upon when looking
 439 for services. Since there is no particular constraint in the choice of a node, i.e.,
 440 every node has the same probability to trigger an application request, $p(\mathcal{N} = n_i)$
 441 follows a uniform distribution.

442 Once the query for services has been generated, it is important to know
 443 which requirements are needed for the specific application, namely to gener-
 444 ate the set of requirements for the query. Indeed, different nodes requesting
 445 the same application can also specify different attributes or characteristics for
 446 it. The set of possible requirements can be quite large, ranging from the ac-
 447 curacy of the sensed data to their precision; however, not all the requirements
 448 are always needed: the *only* ones that need to be declared, either explicitly or
 449 implicitly, are space and time. For example, an application that needs temper-
 450 ature measurements as inputs could be requested in different areas, such as in
 451 a room or a park (space requirement) and for different time intervals, as it is
 452 the case for historical or real-time data (time requirement). The minimum set
 453 of requirements can then be expressed as $\mathcal{Q}^{req} = \{q_{s1}^{req}, q_{s2}^{req}, q_t^{req}\}$, where q_{s1}^{req}
 454 and q_{s2}^{req} indicate the space requirements, namely for the x and y-coordinates,
 455 while q_t^{req} expresses the time requirement.

456 As suggested in [29], to describe the concept of interest in a specific point
 457 in space, the best distribution should be normal: to this, we describe the space
 458 requirements as a 2-dimensional normal distribution, where the probability den-
 459 sity function can be expressed as follows:

$$f_{wi}(q_{s1}^{req}, q_{s2}^{req}) = \frac{1}{2\pi\sigma_{q_{s1}^{req}}\sigma_{q_{s2}^{req}}} * exp \left(-\frac{1}{2} \left[\frac{\left(q_{s1}^{req} - l_i^a - \mu_{q_{s1}^{req}} \right)^2}{\sigma_{q_{s1}^{req}}^2} + \frac{\left(q_{s2}^{req} - l_i^b - \mu_{q_{s2}^{req}} \right)^2}{\sigma_{q_{s2}^{req}}^2} \right] \right) \quad (4)$$

460 where $\mu_{q_{s1}^{req}}$, $\sigma_{q_{s1}^{req}}^2$ and $\mu_{q_{s2}^{req}}$, $\sigma_{q_{s2}^{req}}^2$ are the mean and variance values for the
 461 x and y-coordinates respectively.

462 All these values are application dependent, i.e. they depend on the particular
 463 application a_w at hand, but we have decided not to show such dependence in
 464 the above formula to keep it clean. In particular, when the mean values, $\mu_{q_{s1}^{req}}$
 465 and $\mu_{q_{s2}^{req}}$, are both equal to 0, then the distribution is centered on the current
 466 position of the node n_i , namely l_i^a and l_i^b , i.e a node is looking for information
 467 around itself.

468 Also, the time requirement can be modeled using the time interest of applica-
 469 tions, as suggested in [30], since objects require information mostly in real-time
 470 and less as we move farther in time, i.e. historical data. We modeled such
 471 behavior as an exponential distribution as follows:

$$f(q_t^{req}) = \begin{cases} 0 & \text{if } q_t^{req} > 0 \\ \lambda_a * exp(\lambda_a q_t^{req}) & \text{if } q_t^{req} \leq 0 \end{cases} \quad (5)$$

472 where λ_a is a constant, depending on the particular application at hand.
 473 The requirement for $q_t^{req} = 0$ means that the application is needed in real-
 474 time, while the values of $q_t^{req} < 0$ indicate that historical data are requested.
 475 Whenever a SVO receives a request with a temporal requirement, it will check
 476 if its stored data can satisfy the requirements, otherwise, it has to contact the
 477 physical objects to retrieve the data; however, in some cases, the SVO would
 478 not contact its physical counterpart, in order to avoid consuming resources.

479 Once the query has been generated, the goal of the SIoT system will be to
 480 find all the services in \mathcal{Q} starting from the SVO of the node with the selected
 481 application, making use of its social relations to crawl the network.

482 As an example of query generation, let us consider the following flow: the sys-
 483 tem chooses randomly an object among the available ones, e.g. a car. This car
 484 can be interested in several applications, so the model has to pick one of them,
 485 based on how frequently they require information, e.g. the *DriveMonitoringApp*
 486 showed in Figure 4: the resulting set of atomic queries is then $\mathcal{Q}^{serv} = \{ \text{Geoloc.,}$
 487 $\text{Speed and Acceleration, Sound, Temperature, Street Lights} \}$. The final step is
 488 to set the requirements for the application, that will be inherited by every ser-

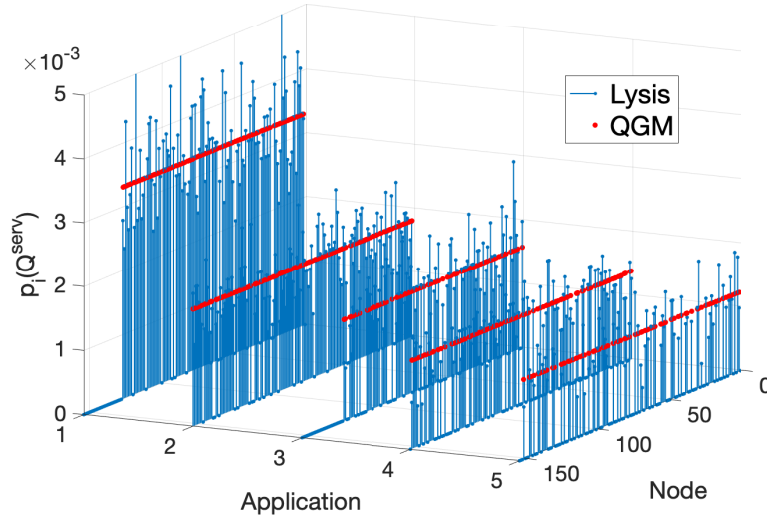


Figure 6: Query probability distribution for the Lysis data and for the Query Generation Model (QGM).

489 vice in Q^{serv} : as spatial requirement, the car chooses an area of [100 m x2 km]
 490 around itself (i.e. in the road ahead), while as time requirement, the car se-
 491 lects $q_t^{req} = 0$ thus asking for the information to be obtained continuously in
 492 real-time.

493 The goal of the SIoT system will then be to find all the services in Q^{serv}
 494 starting from the SVO of the selected car, making use of its social relations to
 495 crawl the network.

496 4.2. Model Simulation

497 In order to simulate and validate the query generation model proposed, a set
 498 of real IoT queries is required. These data are obtained by the Lysis platform
 499 [25]: a collection of more than 11000 queries from 154 devices over a period of 7
 500 months, from April 2017 to October 2017 (a complete description of the data is
 501 available here¹). The network is composed of two types of nodes: smartphones
 502 and Raspberry boards; based on the typology, the devices can require up to five
 503 different applications.

504 Figure 6 illustrates how the proposed query generation model, displayed by
 505 red dots and labeled QGM, matches the probability for each node to generate
 506 a specific set of atomic queries Q^{serv} obtained from the Lysis dataset, repre-
 507 sented with blue lines. Moreover, since not all the devices can install the same
 508 applications, then it will happen that some nodes will *never* require a given ser-
 509 vice and then *never* generate the corresponding query and thus $p_i(Q^{serv}) = 0$.

¹<http://www.social-iot.org/index.php?p=downloads>

Table 2: Requirements for the Lysis Applications.

App	$\mu_{q_{s1}^{req}}$	$\mu_{q_{s2}^{req}}$	$\sigma_{q_{s1}^{req}}$	$\sigma_{q_{s2}^{req}}$	H_1	λ_a	H_2
1	0	0	0	0	0	0.50	0.099
2	0	0	0.1	0.1	0.019	2.00	0.116
3	0	0	0.3	0.3	0.013	0.50	0.108
4	0.45	-0.45	0.1	0.1	0.012	10.0	0.118
5	0.70	-0.70	0.1	0.1	0.018	20.0	0.159

510 In our model, the nodes follow a uniform distribution, while the applications’
511 frequency is proportional to the number of services needed by each application,
512 i.e. that the first app requires more services than the last app. To evaluate the
513 performance of our model, we made use of an f-divergence measure, namely the
514 Hellinger distance [31], to quantify the similarity of the two probability distribu-
515 tions. Unlike other f-divergence measures, the Hellinger distance is a bounded
516 metric: given two probability distribution P and Q , the maximum distance 1
517 is achieved if P and Q are completely divergent, while a distance $H(P, Q) = 0$
518 means that the two probability functions are completely overlapping and hence
519 identical. In our case, the value of the Hellinger distance is equal to 0.0047, so we
520 can conclude that our model is able to generate an almost identical distribution
521 w.r.t. the real data.

522 Table 2 shows the parameters used to describe the space and time require-
523 ments for each of the five applications. The two columns labeled as H_1 and H_2
524 indicates the Hellinger distance between the real data from the Lysis dataset
525 and our requirement distributions for space and time, respectively. The maxi-
526 mum value of the Hellinger distance is under 0.16 thus indicating a very good
527 approximation of our model.

528 The values of the model’s parameters, namely $\mu_{q_{s1}^{req}}$, $\mu_{q_{s2}^{req}}$, $\sigma_{q_{s1}^{req}}$, $\sigma_{q_{s2}^{req}}$ and
529 λ_a , are computed by applying linear regression to a small set of interactions for
530 each application (around 1% of the total number of requests).

531 5. Data Analysis and Simulations

532 This Section presents a dataset of profiles for the objects in a Smart City
533 environment, based on the FIWARE Data Models [32]. The dataset is then
534 used to construct a social network of objects, which is analyzed in Subsection
535 5.2. Finally, the last Subsection assesses the performance of the network when
536 tested with the query generation model in terms of navigability.

537 5.1. Dataset

538 The main functionalities required to create a dataset are illustrated in Figure
539 7. As it will be better explained in the rest of the subsection, these functionalities
540 are in charge of creating: objects’ information (e.g. owner, typology, brand,
541 model), traces of the positions and timestamps of the devices, the list of all the
542 applications that can be installed by the objects, objects’ profiles (expressed

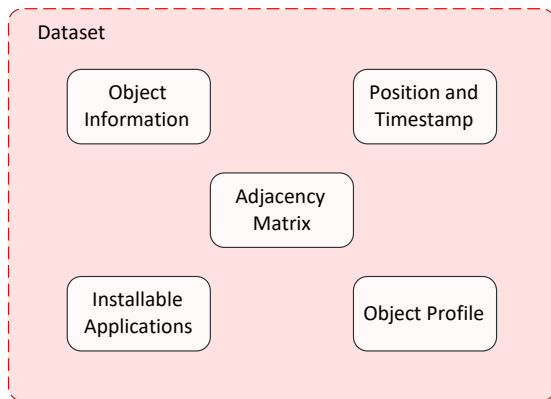


Figure 7: Dataset functionalities

543 as the set of the available services) and an adjacency matrix with the social
 544 relationships for each object.

545 The first step to construct a dataset is to obtain the profile of the objects. To
 546 this, we extracted objects' information and positions from the SmartSantander
 547 project [33], which is experimental research in support of typical applications
 548 and services for a smart city. We have classified all the available devices accord-
 549 ing to the data models proposed in the FIWARE Data Models. This enables
 550 the portability of the dataset among different platforms. These models consider
 551 both static and mobile objects and are mostly located in the city center of the
 552 city.

553 Each of the three public mobile categories of objects, namely buses, taxis
 554 and garbage trucks, moves in an independent way: buses' movement is created
 555 according to the list of bus stops, which are available from the Servicio Municipal
 556 de Transportes Urbanos de Santander (TUS) [34]; taxis can start from 1 out of
 557 3 taxi stations around the city; garbage trucks start from the landfill and cover
 558 all the city.

559 However, a complete Smart City scenario must also consider devices from
 560 private users. To this, we introduce 4000 users in the city, so that each user
 561 owns a certain number of devices. The devices' distribution is based on the
 562 ownership report of the Global Web Index in 2017 [35] calculated over 50000
 563 users aged among 16 and 64 years old and it is shown in Table 3; some of these
 564 devices are considered mobile, i.e. they are carried by the users during their
 565 movements, while others are static and are then left at the user's home.

566 To simulate the mobility of these 4000 users, we rely on the well-known mo-
 567 bility model *Small World In Motion* (SWIM) [36]. SWIM can generate synthetic
 568 data, which can create mobility traces able to mimic human social behaviors. In
 569 fact, it has been proven that the SWIM mobility model allows obtaining accu-
 570 rate matching between the output of the model and the most popular mobility
 571 traces available in CRAWDAD [37], generating data with the same statistical
 572 properties, such as in terms of inter-contact time between people. The simu-

Table 3: Distribution Ownership Devices over 50.000 Users Aged 16-64.

Mobile Devices	Ownership (%)
Smartphone	91
Car	55
Tablet	40
Smart Fitness	22
Smartwatch	5
Static Devices	Ownership (%)
Pc	84
Printer	53
Home sensors	15

573 lation area needs to match the city center of Santander, so since SWIM only
 574 considers areas of interest of unitary square, we had to scale down the city center
 575 (which roughly has an area of 4 km x 4 km) and then modify the model to
 576 avoid users to move towards uninhabited areas, such as the sea.

577 The simulator requires some additional parameters. The user perception
 578 radius, set to 0.015, indicates the distance within which a user, or in our case
 579 a device, can *see* all other users/devices; this parameter is set according to the
 580 communication range of a Wi-Fi connection [38] specifically scaled considering
 581 that the simulation area of SWIM is a unitary square. The parameter α , which
 582 can have values in the range $[0; 1]$, is used to determine whether the users prefer
 583 to visit popular sites (smaller values) rather than nearby ones (bigger values).
 584 It has been set to 0.9. The entire simulation covers a time-lapse of ten days.

585 Following the network modeling proposed in Section 3.2, the dataset as a
 586 total number of devices equals $I = 16216$, 14600 of which are private and 1616
 587 are public. The resulting network comprehends a total of $X = 16$ typologies
 588 of objects and to each of the typologies owned by private users a brand and
 589 a model selected randomly among $Y = 12$ brands and $Z = 24$ models have
 590 been assigned. We suppose that the municipality bought all the objects inside
 591 an object's typology with the same brand and model, so only the category is
 592 needed to classify public objects.

593 The devices of the smart cities, compared to the dataset in [4], are able to
 594 provide $J = 18$ services, which can be arranged to provide $W = 28$ different
 595 applications for the users.

596 A complete description of the data obtained in this paper is available for tests
 597 here² and includes objects' information (such as owner or typology), traces on
 598 the positions and timestamps of the devices, the list of all the applications we
 599 envision in a Smart City scenario, objects' profiles (expressed as set of available
 600 services and possible applications requests) and an adjacency matrix with the
 601 social relationships for each object.

²<http://www.social-iot.org/index.php?p=downloads>

602 *5.2. Network Analysis*

603 Based on object movements and profiles, each device can create its own set
604 of relations with other devices. All the relations depend on the rules set in the
605 system: as explained in [39], these rules have a direct impact on the overall
606 navigability of the network: for the overall network to be navigable, i.e. to
607 enable a node to easily reach any other node in the network, all, or the most
608 of, the nodes must be connected, i.e., a giant component must exist in the
609 network, and the effective diameter must be low. Moreover, the distribution
610 of the number of connections each node has with its peers, namely the degree
611 distribution, should be close to a power-law distribution. This results in a scale-
612 free network and indicates the presence of hubs, i.e. nodes with a large number
613 of connections w.r.t. the average, in the network. With this goal in mind, in
614 the following, we discuss the characteristics of the obtained resulting network.
615 The only relation we did not consider in these experiments is the C-WOR since
616 it has been demonstrated from [39] that its contribution to the navigability of
617 the network can be negligible.

618 All relationships, except for the OOR, are created using as a starting point
619 [4]. An overview of the relations and their differences is illustrated below:

- 620 • The *Ownership Object Relationship* (OOR) is created between devices that
621 belong to the same owner. To avoid too many relations, objects will create
622 a relation only if they are in the communication range of each other. We
623 assume that private devices use one out of three possible technologies:
624 LoRa, Wi-Fi and Bluetooth.
- 625 • The *Parental Object Relationship* (POR) is created among two objects in
626 the same category, brand and model. Since the reasoning behind the POR
627 is to create long-distance links, two devices owned by private users, with
628 the same typology, brand and model, will establish a relationship only if
629 their distance is greater than a threshold, which is set to 3.8 km in order
630 to reduce the number of relationships. For the public devices, a node is
631 elected as a hub and all the other nodes with the same model will create
632 a POR with the hub.
- 633 • Devices located in the same place can create a *Co-Location Object Rela-*
634 *tionship* (C-LOR). These relationships are created between a static device
635 and a mobile one and do not take into account the contact duration but
636 only the number of meetings between the two objects. A number of meet-
637 ings equal to 10 has given an appropriate number of relations.
- 638 • The *Social Object Relationship* is a relation type that can be created
639 among mobile devices and it is based on three parameters, namely the
640 number of meetings (N), the meeting duration (T_M) and the interval be-
641 tween two consecutive meetings (T_I). These parameters are set to $N = 3$,
642 $T_M = 15$ minutes and $T_I = 3$ hours, respectively.
- 643 • Mobile public object have hardly any chance to create SORs, so in order
644 to include them in the SIoT network, we introduce another specific type

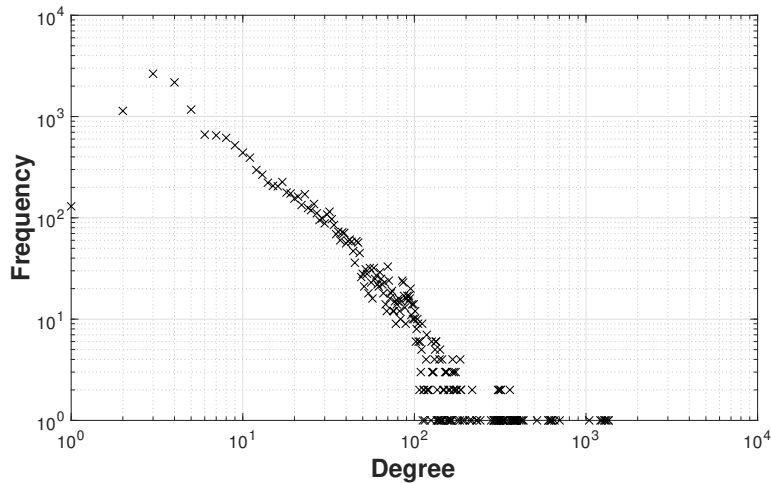


Figure 8: SIoT Degree Distribution.

Table 4: Relationships' parameters.

Parameters	OOR	POR	C-LOR	SOR	SOR ₂	SIoT
Number of relationships	58173	21245	27440	21245	20910	146117
Giant component (%)	8.99	4.17	51.28	24.06	15.23	100
Average degree	50.01	2.00	6.59	10.89	16.93	18.02
Average path length	2.15	1.99	27.31	4.34	3.01	4.22
Diameter	5	2	69	8	7	8

645 of SOR. This SOR, which we called **SOR₂**, uses the same parameter of
 646 the SOR but with less stringent constraints; in particular, we set them to
 647 $N = 2$, $T_M = 2$ minutes and $T_I = 1$ hour.

648 The resulting distribution for the network is shown in Figure 8, while Table 4
 649 shows the main network parameters for each relation and the whole network. We
 650 can notice that the SIoT degree distribution can be approximated to a power-
 651 law distribution, thus indicating its navigability. This is due to the presence
 652 of C-LOR and SOR, while the OOR and POR, which originate from other
 653 parameters, i.e., nodes characteristics and number of devices owned by a user,
 654 deviate from such a distribution: however, these relationships are still important
 655 since they connect groups of nodes so that the majority of nodes have more than
 656 one connection.

657 The average degree indicates the average number of edges connected to each
 658 node: OOR is the relation that creates the greatest number of friendships,
 659 however, it only creates small clusters of highly interconnected objects and thus

660 the dimension of the giant component, the highest percentage of nodes belonging
661 to the largest finite fraction of the entire graph’s nodes, is low. Similar reasoning
662 also applies to the POR: since the goal of the POR is to create long-distance
663 links, the relation is created only if the distance between two devices is greater
664 than a threshold so that the resulting number of relations is lower w.r.t. OOR.
665 Finally, OOR and POR are able to create a highly connected cluster has can
666 be inferred by the low values of the average path length, which is the average
667 number of steps along the shortest paths for all possible pairs of network nodes.
668 In order to connect the public mobile devices (buses, taxies and garbage trucks)
669 we had to add another type of relation, that we called SOR_2 : this relation
670 makes use of the same parameter of the SOR but considering less stringent
671 requirements. The contribution of the SOR_2 to the navigability of the network
672 is the same as C-LOR and SOR: since all these relations create short distance
673 links among devices regardless of their characteristics, they are able to connect
674 the cluster of objects created by OORs and PORs. The resulting SIoT network
675 then comprehends a giant component with all the devices where the longest
676 shortest path between any two nodes, i.e. the diameter of the network, is still
677 low.

678 This result is important since it ensures that every query generated by any
679 object in the network can be fulfilled.

680 *5.3. SIoT network navigability*

681 The navigability in a network indicates how a node can reach any other
682 peer and thus represents a fundamental parameter both for the generation of
683 a network and to understand the average distance between cooperating nodes.
684 To test the navigability of our dataset and query generation model, we have
685 chosen the object typology with the highest number of requested applications,
686 the smartphones, and analyze 1000 processes of the query generation model. All
687 the results are shown with a 95% confidence interval around the mean value,
688 i.e. that 95% of the values from the distribution lie within ± 1.96 standard
689 deviations.

690 The network’s response is calculated in terms of average distance, i.e. the
691 average number of hops needed to find all the required services, computed on
692 the number of services of the application. This is done to avoid disparity among
693 applications that require a different number of services: if, for example, the
694 application depicted in Figure 4 is satisfied in three hops, that means that the
695 five services composing it are found by the search engine in 15 hops, and then
696 with an average of three hops each. This is also justified by the fact that the
697 services can be found in parallel; however, in this paper we do not implement
698 any specific searching mechanism, i.e. we are not using any mechanism for a
699 node to navigate the network on its own with local information. On the other
700 hand, we compute the distance among two nodes in terms of global network
701 navigability, i.e., routing is performed by assuming that each object has a view
702 about the global social network topology.

703 In order to compare the performance of the query generation model for
704 the SIoT, we also created two other networks with similar characteristics: a

Table 5: Characteristics of the Random, Barabási-Albert and SIoT networks.

Parameters	Random	BA	SIoT
Number of relationships	145852	146449	146117
Average degree	17.99	18.06	18.02
Average path length	3.68	3.17	4.22
Diameter	5	5	8

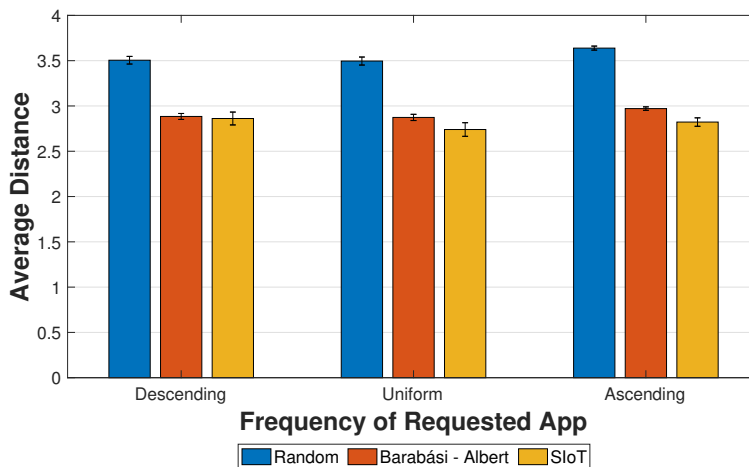


Figure 9: Average number of hops needed to solve a query. The applications' frequency changes based on the number of services requested by the application itself.

705 Random network and a Barabási-Albert network, which is able to generate
 706 scale-free networks based on preferential attachments [40]. The characteristics
 707 of the three networks are shown in Table 5: we can see how, at a global level,
 708 the SIoT has a higher average path length and diameter w.r.t the other two
 709 networks.

710 The first set of simulations focuses on the impact of the applications' fre-
 711 quency. Queries are then generated with a frequency related to the number of
 712 services needed by the application and without any kind of requirements, i.e.
 713 the network has to find all the nodes that can provide the required service. The
 714 results are shown in Figure 9.

715 We can notice that the SIoT network is able to outperform the other two
 716 networks, independently of the frequency. This is due to the fact that the
 717 SIoT relations are created to connect nodes with similar interests, so as to
 718 facilitate the discovery of information. Moreover, the impact of the applications'
 719 frequency is negligible. This result can be explained considering that the final
 720 goal of a search engine is to find the services needed by an application and
 721 that the same services can be arranged in several ways thus providing different

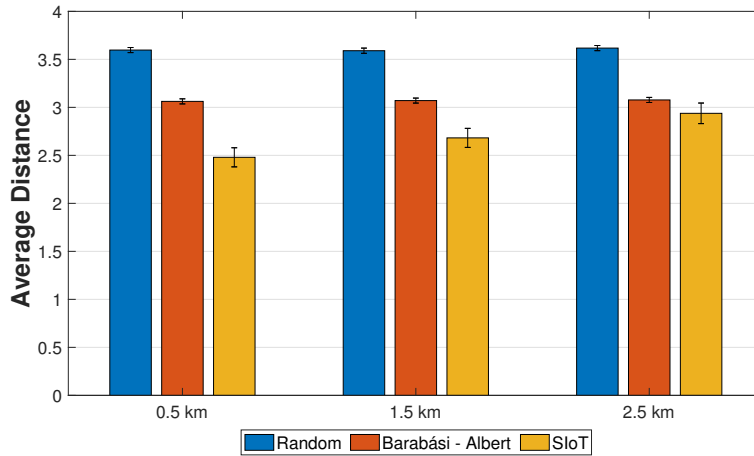


Figure 10: Average distance for different mean values of the space requirement.

722 applications: this means that even by changing the frequency, the services that
 723 need to be found are mostly the same. In the following, we will consider that
 724 all applications generate queries with the same frequency.

725 The second set of experiments consists of the analysis related to the space
 726 requirement: we first evaluate the impact of the mean values for the x and
 727 y-coordinates and then we investigate the effects of the variance.

728 Figure 10 shows the hop distance when nodes request applications that are
 729 located, on average, 0.5, 1.5 and 2.5 km away from them. This value is calculated
 730 as the Euclidean distance between the requester and the possible providers. We
 731 can note that there is a difference of almost half a hop between the two extreme
 732 cases, namely 0.5 and 2.5 km; even if the SIoT envisages the creation of long-
 733 distance links, such as the PORs, the greatest number of relations are created
 734 with nearby devices, so the best results can be obtained when a node looks for
 735 services in its vicinity. This is justified by Figure 11, which shows the average
 736 number of friends created within 1, 2 and 3 km from a node. We can note that,
 737 w.r.t. the other two networks, in the SIoT the greatest number of relations are
 738 created with nearby devices, so the best results can be obtained when a node
 739 looks for services in its vicinity.

740 To test the impact of the variance, shown in Figure 12, we consider different
 741 values that can cover respectively 50, 100 and 500 meters. As expected, the big-
 742 ger the variance, the bigger the number of nodes that can provide the requested
 743 services and thus is simpler for the search engine to quickly find them.

744 The third set of simulations focuses on the time requirements, i.e. how fresh
 745 the information a node is requesting must be. As explained before, the search
 746 mechanism is performed at the virtual level, where the virtual counterparts store
 747 the information provided by the physical objects: however, this information can

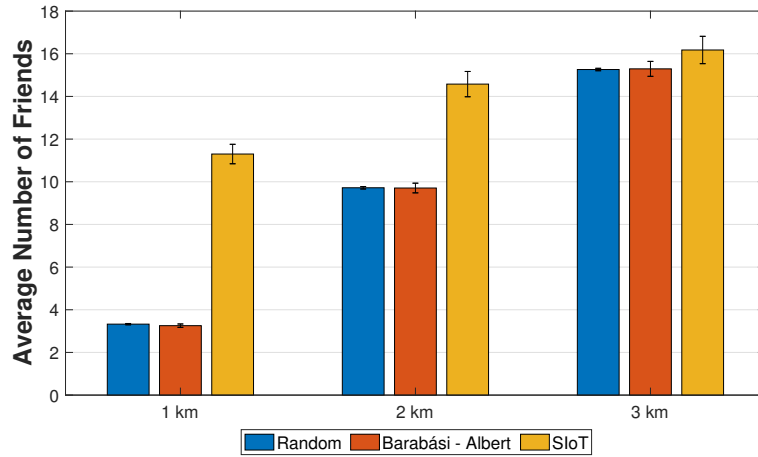


Figure 11: Average number of friends for objects within an area of 1, 2 and 3 km radius.

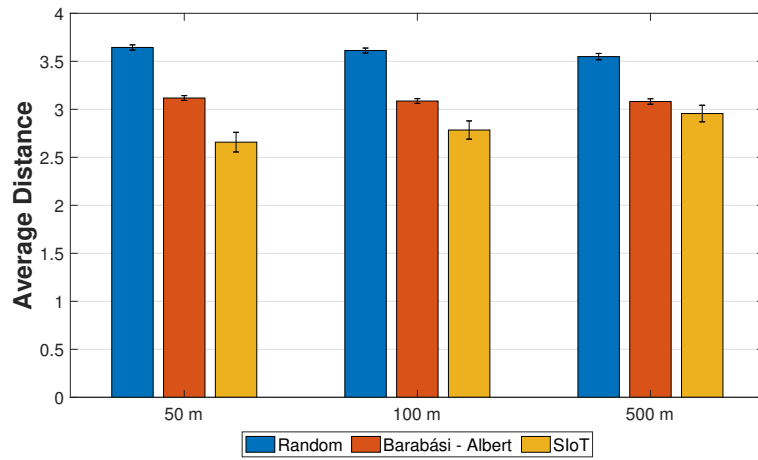


Figure 12: Average distance for different variance values of the space requirement.

Table 6: Synchronization time for all types of objects

Typologies	Sync. Time
Car, Indicator, Smart Fitness, Street lighting and Waste Management	7 minutes
Alarms, Home sensors, Parking, Smartphone, Smartwatch, Tablet and Transportation	17 minutes
Environment, Pc and Weather	23 minutes
Point of Interest and Printer	29 minutes

748 not be always synchronized with the ones sensed by the objects due to energy
749 and bandwidth constraints. Based on the characteristics of the objects, every
750 typology has a different synchronization time (see Table 6), so it may happen
751 that the information found by the search engine is not fresh enough. In this case,
752 the SVO interacts with the physical object, and then consumes its resources,
753 to ask for additional reading in order to satisfy the query. Synchronization
754 times are chosen as prime numbers to avoid that a large number of objects
755 upload information to their corresponding SVOs at the same time. At the
756 start of our simulation (time 0), all the devices synchronize their data with the
757 corresponding SVO, and then they follow the synchronizations depicted in Table
758 6. So at any point in time, when a request with a temporal constraint arrives,
759 we are able to compute if the SVO can satisfy it with its information or it has
760 to contact the physical device.

761 The following results are shown only for the IIoT network; indeed, the network
762 is created considering only space parameters, so there are no further differences
763 among the networks. Figure 13 shows the average distance to satisfy
764 a query looking for information generated within 1, 5 or 10 minutes. We run
765 100 query processes and each process is repeated 10 times. Relaxing the time
766 constraint, as it happens with a 10 minutes requirement, leads to results similar
767 to those obtained without any requirement for the query; on the other hand,
768 the number of hops increases when the application is requested within a short
769 time. As we approach the real-time requirement, $q_t^{req} = 0$, we can note that
770 some points start to be missing from the curves: in particular, when requesting
771 applications with a 1-minute requirement, the corresponding curve has no data
772 for processes 12, 74 and 77. This means that the search engine has not been
773 able to find any SVO satisfying the query in any of the 10 runs.

774 We then decided to analyze the number of times an SVO has to contact its
775 physical counterpart during the 10 runs to satisfy the query. Figure 14 shows
776 the corresponding results: as expected, with 1-minute requirement and during
777 processes 12, 74 and 77, the SVO had to contact the physical object for all 10
778 runs and 6.43 times on average over the 100 processes, while with the 10-minutes
779 requirement it is always possible to find an SVO with the required service.
780 Finally, with the 5-minutes requirement, the physical objects are contacted on
781 average less than once for each process (0.77 times).

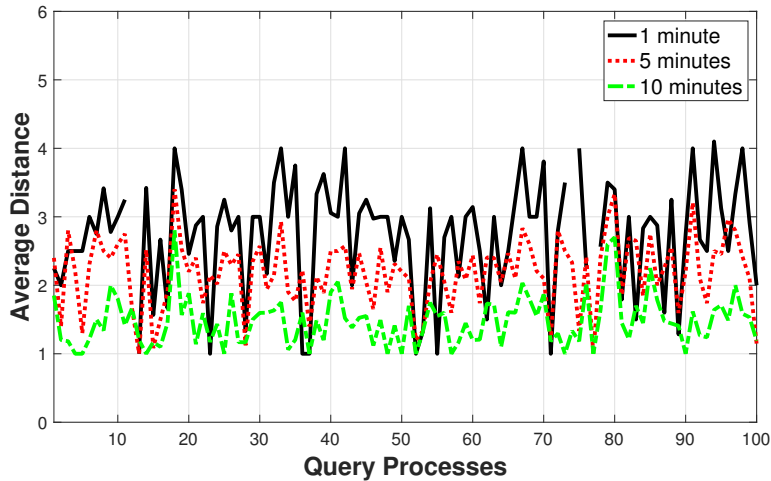


Figure 13: Average distance for different values of the time requirement.

782 The last set of simulations concerns the performance of the network to satisfy
 783 a complete query with both space and time requirements. To this, we decide
 784 to create a generic query requesting an application 1.5 kilometer away from the
 785 requester (mean value) with a range of 200 meters (variance value) and with
 786 information related to the last 5 minutes (time value). Figure 15 on the left axis
 787 shows the distance to solve the query, which is 3.16 hops over the 100 processes.

788 Finally, on the right axis of Figure 15, we try to analyze if hubs are involved
 789 in the search process by plotting the average degree of the intermediate nodes
 790 between requester and provider, i.e. the degree of the nodes forwarding the
 791 query. Given that the global network navigability returns the best possible path,
 792 it is also able to find the best intermediate nodes, i.e. the hubs in the network.
 793 By analyzing the degree of such nodes, we provide hints to the development
 794 of local routing algorithms, that should make use of these nodes. The average
 795 degree for the requesters is 17.43 friends, which is in line with the average
 796 degree of the network, while the average degree for the providers is 50.25 friends.
 797 However, when studying the degree of the intermediate nodes we find that its
 798 value is 149.08 connections thus confirming that the hubs are a crucial part of
 799 the search mechanisms in the SIoT.

800 6. Conclusions

801 In this paper, we have proposed a query generation model that can be used
 802 to analyze the performance of search and discovery mechanisms in the SIoT.
 803 To define the model, we have generated a dataset, which is based on real IoT
 804 objects, available in the city of Santander, and makes use of people mobility

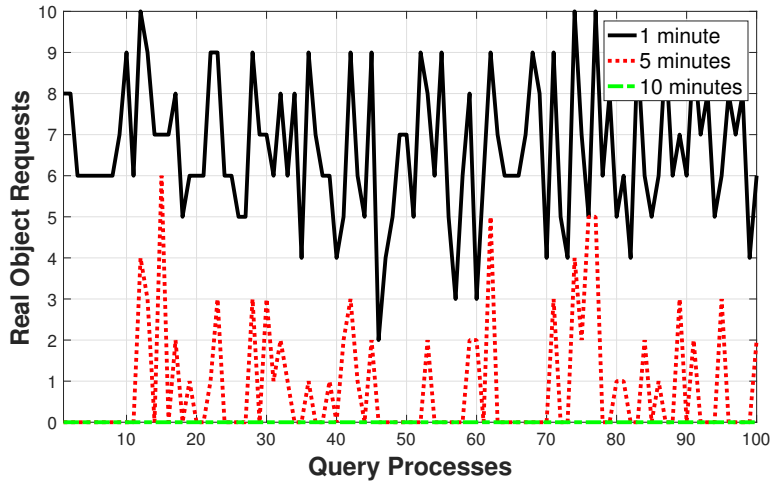


Figure 14: Number of request to real world objects due to the time requirement.

805 models. The dataset and the resulting social net- work are made available to
 806 the research community in order to test several (S)IoT management algorithms.

807 Moreover, we introduce a query generation model, which is able to generate
 808 applications requests from any given node. Our simulations have proven that, if
 809 opportunely tuned, our model is able to generate a query probability distribution
 810 almost identical to the one obtained from real data.

811 Moreover, through experimental analysis we were able to com- pare the SIoT
 812 networks obtained from our dataset with two other network, namely a Random
 813 network and a Barabási-Albert network. Even if, at the global level, the SIoT
 814 network shows worse parameters w.r.t. the other two networks, when tested with
 815 our query model, the SIoT network is able to outperform them: in particular,
 816 we tested the average distance in terms of the number of hops between requester
 817 and provider to respond both to simple queries with no requirements and even
 818 to more complex queries with space and time requirements.

819 Acknowledgements

820 This work was partially supported by Italian Ministry of University and
 821 Research (MIUR), within the Smart Cities framework (Project Netergit, ID:
 822 PON04a200490), within the PON R&I 2014-2020 framework (Project AIM (At-
 823 trazione e Mobilit Internazionale), ID: AIM1873058-1) and by the Italian Min-
 824 istry of Economic Development (MiSE, Project INSIEME, HORIZON 2020,
 825 PON 2014/2020 POS. 395).

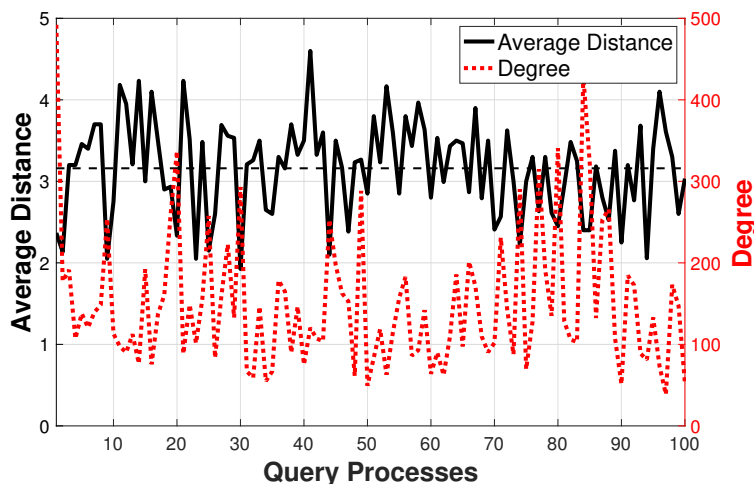


Figure 15: Average Distance and average of intra-nodes degree for queries with both requirements, *Space* and *Time*.

826 References

- 827 [1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, In-
 828 ternet of things: A survey on enabling technologies, protocols, and applica-
 829 tions, *IEEE Communications Surveys & Tutorials* 17 (4) (2015) 2347–2376.
- 830 [2] K. Li, L. Tian, W. Li, G. Luo, Z. Cai, Incorporating social interaction into
 831 three-party game towards privacy protection in iot, *Computer Networks*
 832 150 (2019) 90–101.
- 833 [3] L. Atzori, A. Iera, G. Morabito, M. Nitti, The social internet of things
 834 (siot)—when social networks meet the internet of things: Concept, archi-
 835 tecture and network characterization, *Computer Networks* 56 (16) (2012)
 836 3594–3608.
- 837 [4] C. Marche, L. Atzori, M. Nitti, A dataset for performance analysis of the
 838 social internet of things, in: *2018 IEEE 29th Annual International Sym-
 839 posium on Personal, Indoor and Mobile Radio Communications (PIMRC),
 840 IEEE, 2018, pp. 1–5.*
- 841 [5] M. Nitti, R. Girau, L. Atzori, Trustworthiness management in the social
 842 internet of things, *IEEE Transactions on knowledge and data engineering*
 843 26 (5) (2014) 1253–1266.
- 844 [6] H. Xia, C.-q. Hu, F. Xiao, X.-g. Cheng, Z.-k. Pan, An efficient social-
 845 like semantic-aware service discovery mechanism for large-scale internet of
 846 things, *Computer Networks* 152 (2019) 210–220.

- 847 [7] A. M. Ortiz, D. Hussein, S. Park, S. N. Han, N. Crespi, The cluster
848 between internet of things and social networks: Review and re-
849 search challenges, *IEEE Internet of Things Journal* 1 (3) (2014) 206–215.
850 doi:10.1109/JIOT.2014.2318835.
- 851 [8] S. Büttcher, C. L. Clarke, G. V. Cormack, *Information retrieval: Imple-*
852 *menting and evaluating search engines*, Mit Press, 2016.
- 853 [9] N. Ramachandran, V. Perumal, S. Gopinath, M. Jothi, Sensor search using
854 clustering technique in a massive iot environment, in: *Industry Interactive*
855 *Innovations in Science, Engineering and Technology*, Springer, 2018, pp.
856 271–281.
- 857 [10] P. Barnaghi, A. Sheth, On searching the internet of things: Requirements
858 and challenges, *IEEE Intelligent Systems* 31 (6) (2016) 71–75.
- 859 [11] W. T. Lunardi, E. de Matos, R. Tiburski, L. A. Amaral, S. Marczak,
860 F. Hessel, Context-based search engine for industrial iot: Discovery, search,
861 selection, and usage of devices, in: *2015 IEEE 20th Conference on Emerging*
862 *Technologies & Factory Automation (ETF A)*, IEEE, 2015, pp. 1–8.
- 863 [12] N. K. Tran, Q. Z. Sheng, M. A. Babar, L. Yao, W. E. Zhang, S. Dustdar,
864 Internet of things search engine: Concepts, classification, and open issues,
865 arXiv preprint arXiv:1812.02930.
- 866 [13] M. Liu, D. Li, Q. Chen, J. Zhou, K. Meng, S. Zhang, Sensor information
867 retrieval from internet of things: Representation and indexing, *IEEE Access*
868 6 (2018) 36509–36521.
- 869 [14] S. Wan, Y. Zhao, T. Wang, Z. Gu, Q. H. Abbasi, K.-K. R. Choo, Multi-
870 dimensional data indexing and range query processing via voronoi diagram
871 for internet of things, *Future Generation Computer Systems* 91 (2019) 382–
872 391.
- 873 [15] Y. Yang, J. O. Pedersen, A comparative study on feature selection in text
874 categorization, in: *Icml*, Vol. 97, 1997, pp. 412–420.
- 875 [16] R. Jones, R. Ghani, Automatically building a corpus for a minority lan-
876 guage from the web, in: *Annual Meeting-Association for Computational*
877 *Linguistics*, Vol. 38, UNKNOWN, 2000, pp. 29–36.
- 878 [17] D. Mladenic, M. Grobelnik, Feature selection for unbalanced class distri-
879 bution and naive bayes, in: *ICML*, Vol. 99, 1999, pp. 258–267.
- 880 [18] R. Ghani, R. Jones, D. Mladenic, Online learning for web query genera-
881 tion: Finding documents matching a minority concept on the web, in: *Web*
882 *Intelligence: Research and Development*, Springer, 2001, pp. 508–513.

- 883 [19] D. Boley, M. Gini, R. Gross, E.-H. S. Han, K. Hastings, G. Karypis, V. Ku-
884 mar, B. Mobasher, J. Moore, Document categorization and query gener-
885 ation on the world wide web using webase, *Artificial Intelligence Review*
886 13 (5-6) (1999) 365–391.
- 887 [20] J. Lafferty, C. Zhai, Document language models, query models, and risk
888 minimization for information retrieval, in: *Proceedings of the 24th annual*
889 *international ACM SIGIR conference on Research and development in in-*
890 *formation retrieval*, ACM, 2001, pp. 111–119.
- 891 [21] Q. Mei, H. Fang, C. Zhai, A study of poisson query generation model for
892 information retrieval, in: *Proceedings of the 30th annual international ACM*
893 *SIGIR conference on Research and development in information retrieval*,
894 ACM, 2007, pp. 319–326.
- 895 [22] V. Lavrenko, W. B. Croft, Relevance-based language models, in: *ACM*
896 *SIGIR Forum*, Vol. 51, ACM, 2017, pp. 260–267.
- 897 [23] G. L. Somlo, A. E. Howe, Using web helper agent profiles in query gen-
898 eration, in: *Proceedings of the second international joint conference on*
899 *Autonomous agents and multiagent systems*, ACM, 2003, pp. 812–818.
- 900 [24] H. K. Azad, A. Deepak, Query expansion techniques for information re-
901 trieval: A survey, *Information Processing & Management* 56 (5) (2019)
902 1698–1735.
- 903 [25] R. Girau, S. Martis, L. Atzori, Lysis: A platform for iot distributed appli-
904 cations over socially connected objects, *IEEE Internet of Things Journal*
905 4 (1) (2016) 40–51.
- 906 [26] R. Girau, S. Martis, L. Atzori, Lysis: A platform for iot distributed appli-
907 cations over socially connected objects, *IEEE Internet of Things Journal*
908 4 (1) (2017) 40–51.
- 909 [27] M. Nitti, V. Pilloni, G. Colistra, L. Atzori, The virtual object as a major
910 element of the internet of things: a survey, *IEEE Communications Surveys*
911 *Tutorials PP* (99) (2015) 1–1. doi:10.1109/COMST.2015.2498304.
- 912 [28] I. Grønbaek, Architecture for the internet of things (iot): Api and intercon-
913 nect, in: *Sensor Technologies and Applications, 2008. SENSORCOMM’08.*
914 *Second International Conference on*, IEEE, 2008, pp. 802–807.
- 915 [29] Q. Z. Sheng, E. Stroulia, S. Tata, S. Bhiri, Service-Oriented Computing:
916 14th International Conference, ICSOC 2016, Banff, AB, Canada, October
917 10-13, 2016, *Proceedings*, Vol. 9936, Springer, 2016.
- 918 [30] V. Issarny, G. Bouloukakis, N. Georgantas, B. Billet, Revisiting service-
919 oriented architecture for the iot: a middleware perspective, in: *Interna-*
920 *tional conference on service-oriented computing*, Springer, 2016, pp. 3–17.

- 921 [31] I. Sason, S. Verdu, f -divergence inequalities, *IEEE Transactions on Infor-*
922 *mation Theory* 62 (11) (2016) 5973–6006.
- 923 [32] Fiware data models (2018).
924 URL <https://www.fiware.org/developers/data-models/>
- 925 [33] L. Sanchez, et al., Smartsantander: Iot experimentation over a smart city
926 testbed, *Computer Networks* 61 (2014) 217–238.
- 927 [34] TUS-Santander, Tus santander (2018).
928 URL <http://www.tusantander.es>
- 929 [35] G. W. Index, GWI Social Q1 2017 (2017).
- 930 [36] S. Kosta, A. Mei, J. Stefa, Small world in motion (swim): Modeling commu-
931 nities in ad-hoc mobile networking, in: *Sensor Mesh and Ad Hoc Commu-*
932 *nications and Networks (SECON)*, 2010 7th Annual IEEE Communications
933 Society Conference on, IEEE, 2010, pp. 1–9.
- 934 [37] J. Leguay, A. Lindgren, J. Scott, T. Riedman, J. Crowcroft, P. Hui, Craw-
935 dad trace upmc/content/imote/cambridge (v. 2006–11–17) (2006).
- 936 [38] R. Girau, S. Martis, L. Atzori, Neighbor discovery algorithms for friendship
937 establishment in the social internet of things, in: *2016 IEEE 3rd World*
938 *Forum on Internet of Things (WF-IoT)*, IEEE, 2016, pp. 165–170.
- 939 [39] C. Marche, L. Atzori, A. Iera, L. Militano, M. Nitti, Navigability in social
940 networks of objects: The importance of friendship type and nodes’ distance,
941 in: *2017 IEEE Globecom Workshops (GC Wkshps)*, IEEE, 2017, pp. 1–6.
- 942 [40] L. Zhu, L. Wang, X. Zheng, Y. Xu, The barabasi and albert scale-free
943 network model, *Journal of Intelligent & Fuzzy Systems* 35 (1) (2018) 123–
944 132.

945



Claudio Marche received the M.Sc. degree in telecommu-
nication engineering with full marks in 2018 from the Uni-
versity of Cagliari. Since graduation, he has been working
as Researcher in the Department of Electrical and Electronic
Engineering at the University of Cagliari, in the MCLab re-
search group. He is currently a Ph.D. student in Electronic
and Computer Engineering at the University of Cagliari. His
current research interests include Internet of Things (IoT),
Social Internet of Things (SIoT) and Trustworthiness for IoT.



Luigi Atzori is currently a Professor with the Department of Electrical and Electronic Engineering, University of Cagliari, Italy, where he also leads the laboratory of Multimedia and Communications with around 15 affiliates. He is also the head of the CNIT Research Unit in Cagliari. His research interests are in multimedia communications and computer networking (wireless and wireline), with emphasis on multimedia QoE, multimedia streaming, NGN service management, service management in wireless sensor networks, and architecture and services in the Internet of Things. He served as a technical program chair for various international conferences and workshops. He has been an Area Editor and a Guest Editor for many journals, including ACM/Springer Wireless Networks Journal, the IEEE Communications Magazine, Springer Monet, Signal Processing: Image Communications Journals (Elsevier), the IEEE IOT JOURNAL, and Ad Hoc Networks (Elsevier). He served as a reviewer and panelist for many funding agencies, including H2020, FP7, Cost, and Italian MIUR.



Virginia Pilloni is an Assistant Professor at the University of Cagliari. In 2013 she was awarded with the PhD degree at the same university. From November 2011 to April 2012 she was a visiting PhD student at the Centre for Communication System Research at the University of Surrey. She has been involved in several national and international research projects, among which DEMANES (EU 7FP) and QoE-NET (EU H2020-MSCA-ITN-2014). She was in the organizing committee of 2018 QoMEX 2018 and chair of the 1st and 2nd Workshop on Smart Building and Internet of Things (2019 SBIoT at 2019 IEEE ICCCS and 2020 SBIoT to be held at 2020 IEEE GIoTS). She served as a reviewer for H2020 projects. Her main research interests are on the improvement of performance of Internet of Things nodes through task allocation, and on efficient scheduling of appliances in Smart Building scenarios.



Michele Nitti is an Assistant Professor at the University of Cagliari, Italy since 2015. In 2013, he has been a visited student at the Department of Management, Technology and Economics at ETH Zurich, Switzerland. He served as a technical program chair for various international conferences (IEEE BMSB 2017, IEEE IoT VT Summit 2020) and workshops (IEEE ICCCS 2019, IEEE GIoTS 2020) and he has been keynote invited speaker at the International Workshop of Trends and Challenges in Social Internet of Things (TC-SIoT2018) at PIMRC 2018 and at the Topical Area Track "Green Technologies - Environment, Sustainability the Circular Economy" held during the IEEE World Forum on the Internet of Things 2019. Currently, he is a member of the editorial board for the Computer Networks Journal and co-founder of an academic spin-off (GreenShare s.r.l.) which works in the mobility sector. His main research interests are in architecture and services for the Internet of Things (IoT), particularly in the creation of a network infrastructure to allow the objects to organize themselves according to a social structure (IoT), Social Internet of Things (SIoT) and Trustworthiness for IoT.