Noname manuscript No.
(will be inserted by the editor)

Enhancing Decentralized Service Discovery in Open Service-Oriented Multi-Agent Systems

E. del Val · M. Rebollo · V. Botti

Received: date / Accepted: date

Abstract Service-oriented multi-agent systems are dynamic systems that are populated by heterogeneous agents. These agents model their functionality as services in order to allow heterogeneous agents or other entities to interact with each other in a standardized way. Furthermore, due to the large-scale and adaptative needs of the system, traditional directory facilitators or middle-agents are not suitable for the management of agent services. This article proposes the introduction of homophily in service-oriented multi-agent systems to create efficient decentralized and self-organized structures where agents have a greater probability of establishing links with similar agents than with dissimilar ones. This similarity is based on two social dimensions: the set of services that an agent provides and the organizational roles that it plays. A second contribution is an algorithm for service discovery that it is carried out taking into account the local information that is related to the homophily between agents. The experiments compare our proposal with other proposals in distributed environments. The results show that the proposed structure and algorithm offer desirable features for service discovery in decentralized environments. Specifically, these features provide short paths and a high success rate in the service discovery process and resilience under deliberate failures.

E. del Val, M. Rebollo, V. Botti
Universitat Politècnica de València, Camí de Vera s/n, València, Spain
Tel.: (+34) 96 387 73 50
Fax: (+34) 96 387 73 59
E-mail: edelval@dsic.upv.es

M. Rebollo
Universitat Politècnica de València, Camí de Vera s/n, València, Spain
Tel.: (+34) 96 387 73 50
Fax: (+34) 96 387 73 59
E-mail: mrebollo@dsic.upv.es

V. Botti
Universitat Politècnica de València, Camí de Vera s/n, València, Spain
Tel.: (+34) 96 387 73 50
Fax: (+34) 96 387 73 59
E-mail: vbotti@dsic.upv.es

Keywords Service Discovery · Complex Networks · Homophily

1 Introduction

Service-Oriented Computing (SOC) and Service-Oriented Architectures (SOA) are gaining in importance in industry due to their suitability for quickly coping with new business models and requirements. In these areas, services are considered to be the basic building blocks of complex business applications. Services are platform-independent and can be described, discovered, and composed dynamically. These features make services suitable for giving support to the high rate of change in business demands. In the last few years, there has been a trend in SOC to provide higher levels of functionality in order to facilitate the emergence of new services in a flexible and dynamic way exploiting existing services and avoiding the implementation of redundant services [37]. This trend brings additional considerations to the services. In order to create more complex, flexible, and adaptative systems, services cannot simply be passive and reactive entities. They should be considered as heterogeneous entities that are reactive and proactive and that interact with other entities in a flexible way. Consequently, services are becoming more agent-like [20, 7]. Agents are autonomous, adaptative entities that are aware of what is happening in their environment and that decide to perform local actions (behaviors) based on their observations. Agents are able to learn from previous experiences and update and reason about their information in order to improve their decisions and achieve their goals. The result of these two technologies results has led to a new type of systems: service-oriented multi-agent systems. These systems are populated by agents that provide their functionality through services. These agents are social entities that are aware of the existence of other agents. This awareness facilitates cooperation and collaboration with each other to achieve individual or collective goals that cannot be achieved with individual services [19]. Thus, service management is a key issue in facilitating the cooperation and the goal fulfillment of agents.

Nevertheless, service management is a challenging task due to the inherent characteristics of current service-oriented multi-agent systems. These systems are characterized by their openness. Agents join or leave the network taking into account changes in the environment or in the service demand. Moreover, since there is no central control on how agents should be connected or disconnected and there is no maintenance of system structure. These features provide more flexibility and adaptability to the system. However, service discovery becomes more complicated. The structure of the system cannot provide information that guides the search for services, and agents usually have no global knowledge about the system structure —they only know information about their direct neighbors. For these reasons, agents need the collaboration of the rest of the agents in the system to succeed in the service discovery process. Thus, locating a service efficiently is considered to be one of the most important challenges in this area [5].

In these environments, centralized mechanisms such as registries or middle-agents are not efficient in dealing with this challenge. Weaknessess such as bottlenecks, lack of coordination, outdated data, or the need of huge amounts of memory to store infor-

mation about the agent's services make centralized approaches unsuitable for coping with dynamic system requirements. Moreover, one of the most important drawbacks is that these mechanisms rely on global knowledge and, this global knowledge is usually not present in open service-oriented multi-agent systems. Hence, decentralized service discovery mechanisms are required in these systems.

One of the areas of interest that have structures and search strategies for dealing with decentralized service discovery is the area of Complex Networks [6]. Complex Networks have new, less rigid structures that are inspired in social, biological, or technological networks and algorithms that facilitate the search in distributed environments. This area proposes models to create efficient structures in a self-organized way without the supervision of a central authority. Moreover, in some of these structures, a target can be found in just a few steps and considering only local information [49,23,48,1,22]. Some of these models take into account properties that are present in human societies as a criteria for establishing links. One of these properties is homophily [31].

Homophily is one of the most salient properties present in complex networks [31,9,4]. The term 'homophily' was introduced by Lazarsfeld and Merton [27] in 1954. The idea behind this concept is that individuals tend to interact and establish links with similar individuals. Therefore, homophily establishes the proportion in which two individuals are similar based on a set of social dimensions. These social dimensions are attributes such as religion, age, or education. Therefore, in a complex network model based on homophily, an individual has a higher probability of being connected to a more similar individual than to a dissimilar one. This criterion to establish links between individuals creates structures that facilitate the location task [48,41,22,12]. For this reason, homophily could be considered as a self-organizing principle to generate searchable structures.

In this article, we present a decentralized service management system for service-oriented multi-agent systems where homophily has been introduced as a self-organizing criterion to create the social structure of the system and as a criterion to guide the service discovery process. The structure is a preferential-attachment network where agents create links with other agents by considering their homophily based on two social dimensions: services and organizational roles. We also propose an algorithm that allows agents to locate services offered by other agents using only local information, without any centralized service repository or directory. This algorithm offers good performance not only in networks based on homophily but also in other network structures. The proposed approach is designed to be applied to the context of semantic web services and agents, nevertheless, it could be applied to other contexts where decentralized search is required and semantic information is available. This work is based on an initial proposal presented in [47]. The work presented here extends the previous version in several ways. We present a more extensive review of related proposals in the area of search in distributed environments. We describe in more detail the proposed model and how the different types of homophily are calculated. A more complete set of experiments based on a real test collection of semantic web services is presented for the evaluation of the proposal. We analyze the influence of functional and organizational information in the network structure and in the service discov-

ery process. We also compare our proposal with other traditional complex network models and search algorithms, and we test its performance under deliberate failures.

2 Related Work

Nowadays, there is a trend towards large-scale, open, and highly-dynamic systems. These systems are populated by entities that have to deal with complex tasks and need the services provided by other entities in order to fulfill their goals. Therefore, these systems should provide mechanisms to manage the information about the services available in the system and to determine which entities provide them. Traditionally, this task has been carried out in environments such as Peer-to-Peer (P2P), Service-Oriented Environments (SOE), or Multi-Agent Systems (MAS) through *centralized*, *distributed*, and *decentralized* approaches.

Centralized approaches such as super-peers [18], central registries [45], or middle-agents [24] are appropriate for systems with a low number of entities. In these approaches, the search process is fast and considers all the information that is available in the system. This global knowledge provides efficiency and accuracy in the search process. However, these approaches could be a bottleneck if they have a very limited capability, if the number of entities increases, or if the number of search requests and the information to take into consideration increase. Moreover, the existence of a single entity that is responsible for the management of the information about services seriously affects the robustness of the system. In order to avoid these drawbacks, distributed approaches have been proposed.

In *distributed* approaches, the responsibility of resource management relies on a set of specific entities to provide scalability and robustness. In P2P systems, structures based on *super-peers* [10] and *Distributed Hash Tables* (DHT) [43,40,39,30] have been proposed. *Super-peer* approaches have problems when several super-peers fail and other peers that are less qualified must replace them. *DHT* approaches are able to locate resources in O(log n). Nevertheless, the maintenance of the indexes when the peers join and leave the system affects the performance of the system. Updates imply the interchange of messages among peers; therefore, the system could be in an inconsistent state during a period of time due to outdated references. Furthermore, these mechanisms are not very effective in locating resources with partial information. The accuracy of the search is reduced since the search is based on numeric keys and does not consider semantic information, which allows more flexible and accurate search processes. In the area of SOE there are proposals that distribute the content of the service descriptions in several registries; however there is still a central entity that coordinates, supervises, and is responsible for the maintenance of the structure [42,38,8]. This implies that the search process relies on this central entity and could be a critical point of failure. Some approaches that makes use of coalitions or sets of adaptative matchmakers have been proposed in MAS to provide more scalability [34]. The main problem with these approaches is that the formation process of the optimal coalition requires coordination extra effort among the entities that participate in the coalition.

There are other works based on *decentralized* structures where all the entities are considered to be equal and there is an arbitrary topology. These structures provide more flexibility and adaptability. Entities only have a partial view of the system structure or service organization and need the collaboration of the rest of the system in order to succeed in the search process. The search approaches in decentralized systems use *blind* or *informed* algorithms for locating services or resources.

Blind algorithms do not consider any information about resource locations and use *flooding* or *random* strategies. In flooding strategies, if the entity that receives the query about a resource does not have it, the entity forwards the query to all its neighbors [10,51,35,28]. The efficiency of this strategy depends on the underlying network, the number of copies of one resource, and the Time To Live (TTL) that the query has. In general, flooding algorithms overload the system with the traffic generated during the search process. Random-walks have been presented as an alternative to flooding strategies [29,53]. A random walk strategy is based on the random selection of a subset of the neighbors of the entity to forward the message. Each message follows its own path and is called a walker. A walker can be successful or fail. If the search fails, the reason could either be that the TTL has been consumed or that the query has been satisfied. This algorithm reduces the number of messages considerably when compared to flooding algorithms [5].

In order to prevent the generation of traffic, *informed* algorithms that consider local information have been proposed. These algorithms consider the information that is stored about their direct neighbors or statistics of previous searches in local registries. An example of these algorithms is presented by Crespo et al. [11]. They present a proposal that is based on *Routing indices*. These indices allow nodes to forward queries to the neighbor that is most likely to have answers. Each node has a routing index (RI) with information about the number of documents along the path and the number of documents on each topic of interest. If a node cannot answer the query, it forwards the query to a subset of its neighbors based on its local RI rather than randomly selecting or flooding the network. The problem with this proposal it keeping the large amount of information updated. The number of messages required to propagate changes in the system could overload the system. If the update process is delayed, a node can have information about routes that are not valid. Moreover, the precision of the method depends on the number of categories that are considered in the search process. Similar to the work of Crespo et al. [11], Yang et al. [51] present the *Directed Breath First* search, which forwards the queries only to a subset of neighbors considering several heuristics that are based on information from previous searches (neighbors with the highest success in previous searches, or the neighbor that finds the shortest paths, etc.). *Adaptive Probabilistic Search* is a similar approach presented by Tsoumakos et al. [44]. It is an algorithm that is based on the combination of the *k*-random walk algorithm and probabilistic forwarding. Each peer has a local index that keeps one entry for each neighbor. The value of each entry is a tuple that contains the identifier of a neighbor and the probability that the neighbor be selected the next time based on its success in previous searches. Analogous work is presented by Kalogeraki et al. [21]. The authors propose an *Intelligent Search Mechanism* that allows peers to identify links that are likely to have relevant information. The drawback of these algorithms is that a period of time is needed to collect the

information that improves the search. Moreover, if the links between peers change frequently, the statistical information stored in the local indexes could become useless. Another drawback is that some of the heuristics that are used to guide the search process could overload some peers and leave other potential peers without traffic.

Other approaches use biologically inspired techniques to locate and organize resources. For instance, *ant algorithms* are also suitable for unstructured networks because they do not rely on global knowledge about the network. The algorithm proposed by Michlmayr et al. [32] uses ants to guide the search. Each peer in the system maintains a repository of documents. Each document has the following information associated to it: a keyword, the neighbor that provides the document, and the quantity of pheromone. There are two types of ants in the system: *forward ants* and *backward ants*. The *forward ants* navigate the network until the document is found or the TTL finishes. In each step, the forward ant decides between two strategies: *exploiting* or *exploring*. The first strategy selects the best neighbor based on the quantity of pheromone. The second strategy encourages the forward ants to discover new paths. The *backward ant* is responsible for updating the path with the pheromone. The quantity of the pheromone depends on the goodness of the path. The algorithm also considers an evaporation rule to update the pheromone based on time. The main problem is that the pheromone is based on the keywords of the documents. Therefore, if a peer is looking for a document with a keyword that does not appear, even though similar documents exist, the peer will not find it in the network.

There are other approaches where the underlying structure of the system is *loosely structured* using certain criteria. This facilitates the search process. An example of this is presented by Zhang et al. [52]. The authors propose a completely decentralized MAS without mediators. Initially, agents are connected randomly. The authors propose a reorganization algorithm to group agents with similar services together. In order to avoid isolated clusters of agents, the algorithm establishes a percentage of similar and dissimilar agents that should be in the neighborhood of the agent. For distributed searches, the authors propose the use of two algorithms: K-Nearest Neighbors (KNN) and Gradient Search Scheme (GS). The idea of the first algorithm is to redirect the queries to the most similar k-agents. In this process, the algorithm also considers the degree of the agents. The second algorithm (GS) has a first stage where it tries to find a 'good starting agent'. An agent is considered to be a 'good starting agent' if its similarity with respect to the query is above a certain threshold. If the initial agent is a 'good starting agent', the algorithm performs like KNN. Otherwise, the agent selects the most similar neighbor to the target, and a message with the similarity information is sent to that neighbor. This process is repeated n times. The agent with the highest similarity value will be choosen to restart the search using the KNN algorithm. The main disadvantage of this approach is the high cost of communication required to organize the agents into communities.

Semantics has been included in the systems in order to guide the search process, improve the accuracy of the results, and as a criterion to establish links. Upadrashta et al. [46] present a routing protocol that uses semantics included in queries to improve the performance of Gnutella systems. The main idea is that each peer keeps a list of friends and learns about their interests to obtain more relevant sources faster and with less traffic. The list reflects similarity of interests (semantic categories) between

peers. Bianchini et al. [3] present a decentralized service system. Peers are connected through semantic and logical links. Semantic links are established between peers that offer similar services. Logical links are the links of the P2P system. During the search process, if a peer does not have a service that is similar to the target, it forwards the query taking into account its semantic links. If the peer does not find any semantic similar service, it queries its neighborhood. Specifically, a random subset of its neighbors in the logical layer is selected to redirect the query. This helps to prevent the formation of isolated clusters in the semantic layer. The drawback of this approach is that the peers are organized in clusters of similar services; therefore, a peer may not be able to find services that are semantically different to its services. In this situation, the required service cannot be found using the neighbors in the semantic level and the peer must choose a neighbor using random strategies. This reduces the system to a traditional P2P system without semantics.

Basters et al. [2] use a local training set that contains previous queries and their results and semantic information about services to determine which neighbor is the most promising to forward the query to. This selection is based on probability and uses the mixed conditional bayesian risk, which considers two parameters: the semantic gain and the communication loss (number of messages to find the required service). These two parameters are calculated taking the information of the training set into account. The main drawback of this approach is that it relies on a training set that each agent maintains individually. This training set allows agents to learn which neighbor will probably return relevant semantic web services. When the agent gets into the system, this training set is empty and the agent forwards the requests using a flooding algorithm until it has enough information. In highly dynamic environments, new agents frequently join and leave the system; therefore, they will initially use flooding algorithms that overload the system.

In this article, we present a service discovery system that attempts to improve previous approaches in several ways. First, as a decentralized system, all the agents are considered to be equal and they only consider local information in the service discovery process and to establish links. Therefore, the system provides robustness, scalability, and adaptability. Second, the system is self-organized based on homophily between agents and does not need an initial period to establish its structure. Progressively, each agent that joins the system establishes links with agents that share features such as the organizational role or the services offered. Third, each agent only maintains a local view of the services it offers and who are its neighbors, and it does not maintain information about routes that could change frequently in highly dynamic environments. Fourth, in our system, the algorithm for the service discovery process is not based on previous information or statistics that require a training period in order to be reliable. The algorithm is based on similarity between agents, and this similarity is calculated considering the semantic descriptions of the agents and not just keywords or pre-defined categories. In the following sections, we describe the formal model of the proposal, the creation process of the structure, and how the service discovery process is carried out by the agents. Finally, experiments that evaluate our approach and compare it with other existing proposals are presented.

3 Service Discovery Scenario

To illustrate the context where decentralized service discovery is applied, let us present a service discovery where the discovery process is described (see Figure 1). Consider a network of agents as a form of autonomic cloud computing system. This network contains different groups of specialized computing systems as part of an overlaying network where semantic web services are provided by software agents. Agents play an organizational role that defines the type of services they offer. Agents only have information about their direct neighbors with which they have a connection with and they do not know about the other agents that are part of the system, the number of agents or the system structure. Neither, there are the figure of an intermediator or central registry that has a global and complete vision of the hole system or part of it.

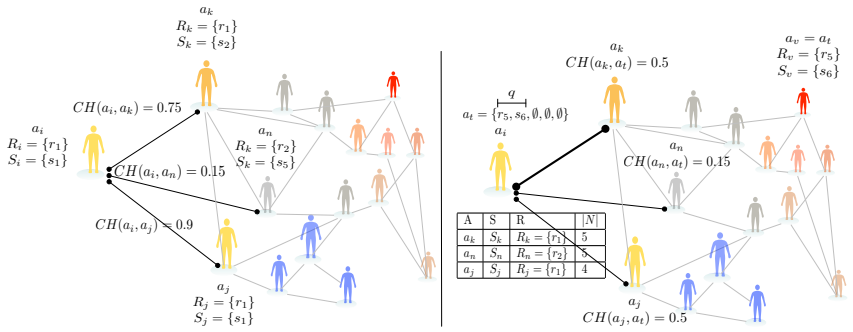
The structural relations between these agents have been established taking the homophily criterion into account [27] also known in complex networks as assortative mixing [33]. Homophily is present in many complex networks [48,41,22]. The idea behind the homophily concept is that individuals tend to interact and establish links with similar individuals through a set of social dimensions. In the context of service-oriented multi-agent systems, two agents are considered similar if they play similar roles and offer similar services. For each pair of agents, the higher the homophily value is, the more similar the agents are. Homophily is a probabilistic concept; therefore, agents have a higher probability of establishing connections with similar agents than with dissimilar ones.

In Figure 1, agent a_i has connections with agents a_k, and a_j, which play similar roles and offer similar services, and a_n, which plays a dissimilar role and offers a dissimilar service. Note that agents that play similar roles are represented in Figure 1 with similar colors.

Agents, in some situations, should interact with each other to achieve a task that they cannot afford to do individually since they are not specialized in that area or because the task is too complex to be carried out by a single agent.

Agent a_i offers the service s_1; however, in order to achieve one of its goals, it needs to locate an agent that offers a service similar to s_6 and plays a role similar to r_5. In that moment, agent a_i creates a query $q = \{s_6, r_5\}$ that consists of the required semantic service description and the organizational role that the target agent should play. The query has an associated Time To Live (TTL), which is the maximum number of times that it can be forwarded. If the query exceeds the TTL, it is considered to be a failure of the service discovery process. Otherwise, the query is forwarded to one of the neighbors. It is assumed that all the agents are collaborative and follow the same criterion to forward the queries.

In the scenario of Figure 1, agent a_i should choose one of its neighbors, a_n,a_j, or a_k, to forward the query q. In order to select the most promising neighbor, the agent a_i considers: (i) the homophily between the neighbors and an hypothetical unknown target agent $a_t = (s_6, r_5, \emptyset, \emptyset)$ that offers the service and plays the role specified in the query q; and (ii) the degree of connection of the neighbors. Assuming the values of choice homophily that appear in Figure 1, agent a_i sends the query to the most promising agent (i.e., agent a_k). This process is repeated until the similarity between a local services of an agent and the service in the query is over a certain threshold,



(a) (b)

Fig. 1: An example of decentralized service discovery system. (a) Agent a_i establishes a link with two similar agents a_k and a_j and with a dissimilar one a_n; (b) Agent a_i only knows its direct neighbors a_k, a_j, and a_n. If a_i needs to locate a service (i.e., s_6), it will forward the query to its most promising neighbor (i.e., a_k) based on the homophily between the neighbor and the target agent (i.e., a_t) that should provide the required service and the degree of the neighbor.

or the query exceeds the TTL. In the described scenario, the process ends when the query arrives to agent a_v that is similar to the hypothetical target agent a_t that a_i was looking for.

4 System Model

The system is made up of a set of autonomous agents that offer their functionality through a set of semantic services. These agents have a reduced view of the global community: just a limited number of direct neighbors are known and the rest of the network remains invisible to them.

DEFINITION **1** *(System). The system is a tuple (A, L), where $A = \{a_1, ..., a_n\}$ is a finite set of autonomous agents and, $L \subseteq A \times A$ is a set of links, where each link $(a_i, a_j) \in L$ indicates the existence of a direct relationship between agent a_i and a_j*

It is assumed that the knowledge relationship among agents is symmetric; therefore, the network is an undirected graph. In this context, an agent is considered to be a social entity that collaborates with other agents in the system. It controls its own information about (i) the semantic services it offers, (ii) the roles it plays in the organization, and (iii) local knowledge about its immediate neighbors. The agent is unaware of the rest of the agents in the system.

DEFINITION **2** *(Agent). In this context, an agent $a_i \in A$ is characterized by a tuple of four elements (R_i, N_i, st_i, π_i) where:*

- *$R_i \subseteq R$ is the set of roles an agent can hold, where R are the roles defined in an organizational ontology;*

- *N_i is the set of neighbors of the agent, $N_i \subseteq A - \{a_i\} : \forall a_j \in N_i, \exists (a_i, a_j) \in L$, and $|N_i| > 0$. It is assumed that $|N_i| \ll |A|$;*
- *st_i is the internal state of the agent. It includes the tuple $(q(t), \varepsilon)$, where:*
  - *$q(t)$ is the service query that the agent receives at a given time t,*
  - *ε is the threshold established by the agent to determine that a service is similar enough to a query.*
- *$\pi_i : A \rightarrow A$ is the neighbor selection function that returns the most promising neighbor to provide a service.*

Note, that all the agents that are part of the system have at least one neighbor. We assume that the network is strongly connected. The condition $|N_i| \ll |A|$ indicates that the network is a sparse network, where each agent is only connected to a small group of agents regarding the size of the system. The function π_i is responsible for forwarding queries and indicates which of the neighbors is nearest to the target. This function returns a unique agent, avoiding other flooding mechanisms that overload the network with messages.

Agents in the system are characterized by the roles they play. The organizational role determines the type of services offered by the agent. The agent acquires a role that is defined inside an organization of the system if it satisfies a set of requirements [15].

DEFINITION **3** *(Role). A role $r_i \in R_i$ is defined by the tuple (ϕ_i, S_i) , where:*

- *ϕ_i is a semantic concept for the role defined in an organizational ontology θ;*
- *$S_i = \{s_1, \ldots, s_n\}$ is the set of services offered by the agent. Each service is defined by the tuple $s_n = (I_n, O_n, P_n, Eff_n)$, where the components are the set of inputs, outputs, preconditions, and effects of the services, respectively. All of them are semantic concepts that can be defined in different ontologies* [1].

Agents in this model are self-organized considering a social feature called *choice* homophily [31]. This type of homophily (CH) is the factor that allows the agents to establish links with similar agents based on a set of social dimensions. Choice homophily is subdivided into two types: (i) *value homophily*, which is based on the similarity of shared attributes (such as gender, age, geographical location, and so on); and (ii) the *status homophily*, which is related to the formal or informal status similarity of the individuals (social status, status within an organization, or professional degree). In the following sections, we describe how these social features are matched to agency-related concepts and how have been introduced to enhance the structure and service discovery process of the system.

5 Introducing Homophily in Service-Oriented Multi-Agent Systems

In this section, we focus on the formal definition of *choice homophily* in the context of service-oriented multi-agent systems and how it is included in the system as

[1] For the semantic description of the services we used the OWL-S language.

a criterion to self-organize the structure and guide the service discovery process. As stated above, choice homophily is divided into two types: *value* and *status*. If these two concepts are matched with the agency-related concepts, value homophily represents the individual characteristics of the agent (which are the services the agent offers), whereas status homophily can be identified with the semantic description of the role that an agent plays within an organization.

DEFINITION **4** *Choice homophily between two agents $a_i, a_j \in A$ in the system is defined as the linear combination of value and status homophily,*

$$CH(a_i, a_j) = (1 - \varphi) * H_v(S_i, S_j) + \varphi * H_s(R_i, R_j)$$

The φ parameter regulates the importance of the influence of services (value homophily) or roles (status homophily) in the total homophily of the agent with another agent.

The *value homophily* function $H_v(S_i, S_j)$ calculates the degree of matching between two sets of services, where S_i and S_j are the sets of services provided by the agents a_i and a_j, respectively. We consider each set of services S_i (or S_j) to be composed by a set of semantic concepts that can be classified in: Inputs (I_i), Outputs (O_i), Preconditions (P_i), and Effects (Eff_i).

To generalize, the level of matching between two sets of semantic concepts, C_i and C_j, is calculated through a *bipartite matching graph* (see Figure 2). Let $G = (C_i, C_j, E)$ be a complete, weighted bipartite graph that links each concept $c_i \in C_i$ to each concept $c_j \in C_j$, $e_{ij} = (c_i, c_j) \in E$, and let E represent the edges established in the graph $E = C_i \times C_j$. The term ω_{ij} represents the weight associated to the arc $e_i = (c_i, c_j) \in E$ between c_i and c_j as the semantic similarity between those concepts. Four degrees of matching can be identified: *exact, subsumes, plug-in*, and *fail* [36]. The match is considered to be *exact* if $c_1 \in C_i$ is equivalent to $c_2 \in C_j$ ($c_1 \equiv c_2$); it is *subsumes* if c_1 subsumes c_2 ($c_1 \sqsupset c_2$); it is *plug-in* if c_1 is subsumed by c_2 ($c_1 \sqsubset c_2$); and it is *fail*, otherwise. For simplicity, we have considered these four degrees of matching but other degrees could be considered [25]. A value in the interval $[0, 1]$ is assigned to each degree of matching, where 1 represents an exact matching among the terms. The best match among concepts is obtained by calculating the *maximum weighted bipartite matching*, $G' = (C_i, C_j, E')$, where $E' \subseteq E$ are the edges that have the maximal value. The graph G' is a relaxed bipartite graph because not all the concepts from C_j have to be connected to a concept in C_i; therefore, two concepts from C_i can share a concept from C_j.

Specifically, to calculate the *value homophily*, four bipartite graphs are defined, (one for each one of the components of services present in the sets S_i and S_j): Inputs (I_i, I_j), Outputs (O_i, O_j), Preconditions (P_i, P_j), and Effects (Eff_i, Eff_j). Let's explain the case of the inputs. The rest of the components are treated in the same way.

Let $\overline{I_i} = \bigcup_{s_i \in S_i} I_i$ be the set formed by all the inputs of all the services s_i of the agent a_i; Let $G_I = (\overline{I_i}, \overline{I_j}, E)$ be the weighted bipartite graph among the inputs of all the services S_i and S_j provided by agents a_i and a_j; and let $G'_I = (\overline{I_i}, \overline{I_j}, E')$ be the maximum weighted relaxed bipartite matching. Then $W_{G'_I}$ is defined as the normalized total weight of the maximum relaxed bipartite graph G'_I.

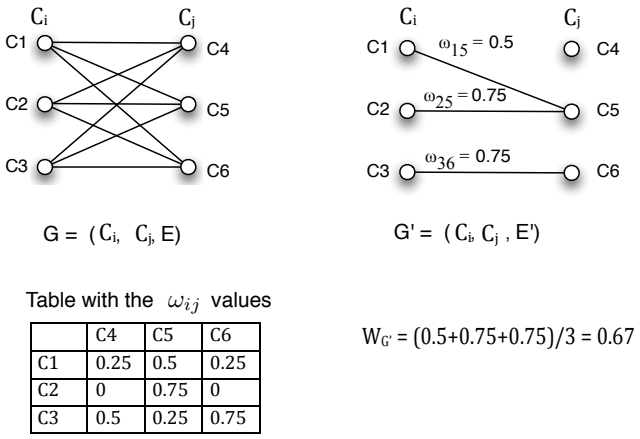


Fig. 2: (Left) Full connected weighted bipartite graph G, and (Right) resulting maximum weighted matching relaxed bipartite graph G'.

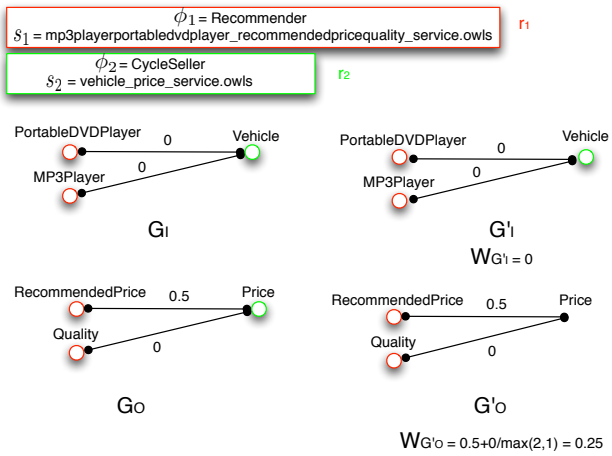


Fig. 3: Example of $W_{G'_I}$ (top) and $W_{G'_0}$ calculation (bottom) between two agents that offer one service each.

$$W_{G'_I} = \frac{\sum_{\omega_{ij} \in E'_I} \omega_{ij}}{\max\left(|\overline{I_i}|, |\overline{I_j}|\right)} \tag{1}$$

$W_{G'_O}$, $W_{G'_P}$, and $W_{G'_{Eff}}$ are similarly defined for outputs, preconditions, and effects, respectively.

DEFINITION **5** *The value homophily between two agents a_i and a_j is defined as*

$$H_v(S_i, S_j) = \alpha \left[\beta * W_{G'_I} + (1-\beta) W_{G'_O}\right] + (1-\alpha)\left[\beta * W_{G'_P} + (1-\beta) W_{G'_E}\right] =$$
$$= \alpha \left[\beta \frac{\sum_{w_{ij} \in E'_I} w_{ij}}{\max |\overline{I_i}|, |\overline{I_j}|} + (1-\beta) \frac{\sum_{w_{ij} \in E'_O} w_{ij}}{\max |\overline{O_i}|, |\overline{O_j}|}\right] +$$
$$+(1-\alpha) \left[\beta \frac{\sum_{w_{ij} \in E'_P} w_{ij}}{\max |\overline{P_i}|, |\overline{P_j}|} + (1-\beta) \frac{\sum_{w_{ij} \in E'_{Eff}} w_{ij}}{\max |\overline{Eff_i}|, |\overline{Eff_j}|}\right]$$

The parameters α and β assign different weights to the components of the formula. The adjustment of $\alpha, \beta \in [0, 1]$ allows varying how the parameters of the service are considered in the calculation of value homophily. The α parameter controls a data-driven homophily calculation (inputs and outputs) or a goal-driven homophily calculation (preconditions and effects). The β parameter determines the importance of the intakes (inputs and preconditions) or the consequences (outputs and effects) in the homophily calculation.

Let's see an example of how value homophily is calculated among two agents, a_1 and a_2 (see Figure 3). Agent a_1 has a set of services S_1 that contains a service s_1 (mp3playerportabledvdplayer_recommendedpricequality_service.owls) that has a set of inputs (I_1 = {PortableDVDPlayer, MP3Player}) and a set of outputs (O_1 = {RecommendedPrice, Quality}). Agent a_2 has a set of services S_2 that contains a service s_2 (vehicle_price_service.owls) that has a set of inputs (I_2 = {Vehicle}) and a set of outputs (O_2 = {Price}). In order to calculate the value homophily, a bipartite graph is created for the inputs G_I and for the outputs G_O. The similarity between the concepts from sets I_1 and I_2 (similarly for concepts from O_1 and O_2) labels the arcs of the graph. In this case, the concepts from I_1 only have one possible matching since there is only one input in the set I_2. Once the maximum bipartite graphs for inputs and outputs are built (G'_I and G'_O), their weights are calculated for inputs and outputs ($W_{G'_I}$,$W_{G'_O}$). To calculate the value homophily we instantiate the parameters α=1 (which means that the services only have inputs and outputs), and β = 0.5 (which means that inputs and outputs have the same importance). We replace the values of $W_{G'_I}$ and $W_{G'_O}$ in Definition 5, and we obtain 0.125 for the value homophily between agents a_1 and a_2.

$$H_v(S_1, S_2) = 1\,[0.5 * 0 + (1 - 0.5)0.25] = 0.125 \quad (2)$$

The *status homophily* $H_s(R_i, R_j)$ in the system calculates the best match between the set of roles R_i and R_j played by the agents a_i and a_j. The match between two individual roles $r_i \in R_i$ and $r_j \in R_j$ is based on the distance between the semantic concepts ϕ_i and ϕ_j. The function presented by [16] is used to calculate the distance.

DEFINITION **6** *Status homophily between two agents a_i and a_j is defined as the maximum degree of match between the concepts ϕ_i and ϕ_j that describe the roles $r_i \in R_i$ and $r_j \in R_j$ for all possible pairs (r_i, r_j).*

$$H_s(R_i, R_j) = \max_{r_i \in R_i, r_j \in R_j} rmatch(\phi_i, \phi_j)$$

where

$$rmatch(\phi_i, \phi_j) = \begin{cases} 1 & \textit{if path length = 0} \\ \delta_{\phi_i\phi_j} \cdot e^{-\lambda(pl_{\phi_i\phi_j} + cp_{\phi_i\phi_j})} & \textit{if } \phi_i \textit{ and } \phi_j \textit{ are not siblings} \\ \delta_{\phi_i\phi_j} \cdot e^{-\lambda(pl_{\phi_i\phi_j} + cp_{\phi_i\phi_j}) - d_{\phi_i\phi_j}} & \textit{if } \phi_i \textit{ and } \phi_j \textit{ are siblings} \end{cases}$$

and

$$\delta_{\phi_i\phi_j} = \frac{e^{\gamma dp_{\phi_i\phi_j}} - e^{-\gamma dp_{\phi_i\phi_j}}}{e^{\gamma dp_{\phi_i\phi_j}} + e^{-\gamma dp_{\phi_i\phi_j}}}$$

The status homophily $H_s(R_i, R_j)$ takes into account the following:

- $pl_{\phi_i\phi_j}$ the shortest path length between ϕ_i and ϕ_j in an organizational ontology;
- $dp_{\phi_i\phi_j}$ the depth of the roles in the ontology;
- $d_{\phi_i\phi_j}$ the number of the sibling nodes of each role;
- $cp_{\phi_i\phi_j}$ the relationship 'parent-child' between roles;
- λ and γ are parameters that control the influence of path length and depth, respectively.

The value obtained in the calculation of $H_s(R_i, R_j)$ ranges in the interval [0,1], where 1 indicates that the roles are the same.

Let's see the calculation of status homophily between the agents a_1 and a_2 from the previous example (see Figure 3). The set of roles R_1 of a_1 is composed by the role r_1 (ϕ_1 = *CycleSeller*). The set of roles R_2 of a_2 is composed by the role r_2 (ϕ_2 = *Recommender*). The calculation of this type of homophily is based on the organization structure that is shown in Figure 4. The roles r_1 and r_2 are depicted with a thick line. The figure shows the organizational roles and the values of the parameters involved in the calculation of $rmatch(\phi_1, \phi_2)$. If we replace the value of the structural parameters and the parameters that control the influence of the path length $\lambda = 0.3$ and depth $\gamma = 1$ in the formula of $rmatch$, we obtain $rmatch(\phi_1, \phi_2) = 0.04$ and $H_s = 0.04$ (see Equation 3).

$$H_s(R_1, R_2) = \delta_{\phi_1\phi_2} \cdot e^{(-0.3(7+3))} = \delta_{\phi_1\phi_2} \cdot e^{-3} = 0.0497 \cdot 0.82 = 0.04 \quad (3)$$

$$\delta_{\phi_1\phi_2} = \frac{e^{(4/6+3/6)} - e^{(-4/6+3/6)}}{e^{(4/6+3/6)} + e^{(-4/6+3/6)}} = \frac{e^{1.16} - e^{0.16}}{e^{1.16} + e^{0.16}} = \frac{3.21 - 0.31}{3.21 + 0.31} = \frac{2.89}{3.52} = 0.82$$

Finally, once the value and status homophily are calculated (see Equations 2 and 3), the calculation of choice homophily between agents a_1 and a_2 is shown in Equation 4. In this equation, the value of φ is 0.75, which means that status homophily has more influence in the final homophily between agents.

$$CH(a_1, a_2) = (1 - 0.75) * 0.125 + 0.75 * 0.04 = 0.061 \quad (4)$$

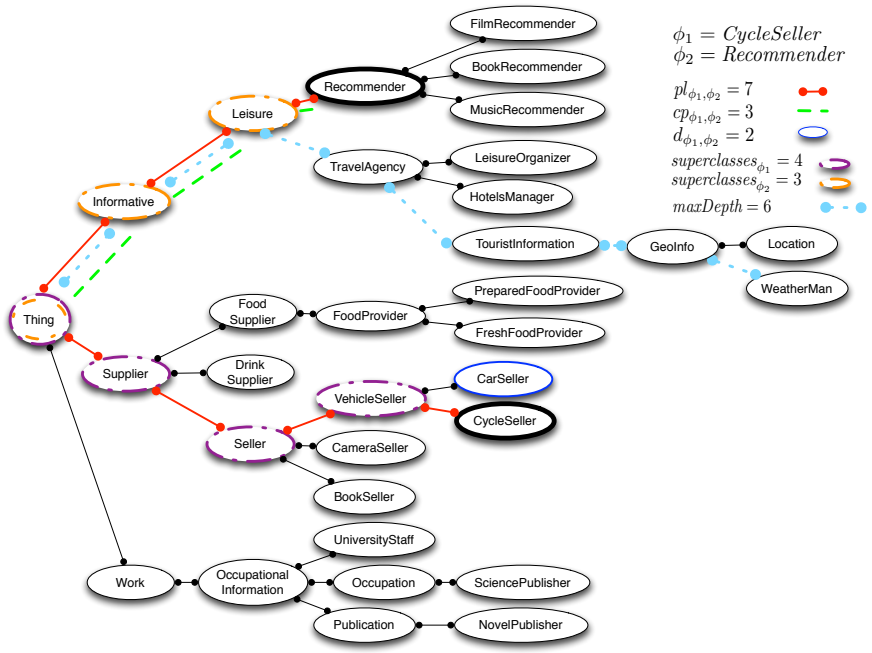


Fig. 4: Partial view of the organizational structure that contains the semantic concepts that define the roles present in the system.

5.1 Community Creation based on Homophily

Once we have defined choice homophily in the context of service-oriented multi-agent systems, we describe how it is included in the structure creation process and in the service discovery process. Choice homophily establishes a measure of semantic similarity between two agents. This similarity measure is taken into account by the agents in the system creation process.

Each agent that is part of the system is considered an entry point. If an external provider agent a_i wants to get into the system, it follows the protocol shown in Figure 5:

- Agent a_i should know at least one agent a_j already present in the system. Agent a_i sends a request to agent a_j to be part of the system.
- If a_j sends a_i a refuse message the interaction finishes. Otherwise, a_j allows a_i to get into the system and it sends a_i an agree message. This means that a_j, based on the choice homophily between them, is going to consider the establishment of a link with a_i. The probability P_l of establishing a connection between agent a_i and agent a_j is

$$P_l(\langle a_i, a_j \rangle) = \left(\frac{1 - CH(a_i, a_j)}{\rho} \right)^{-r} \tag{5}$$

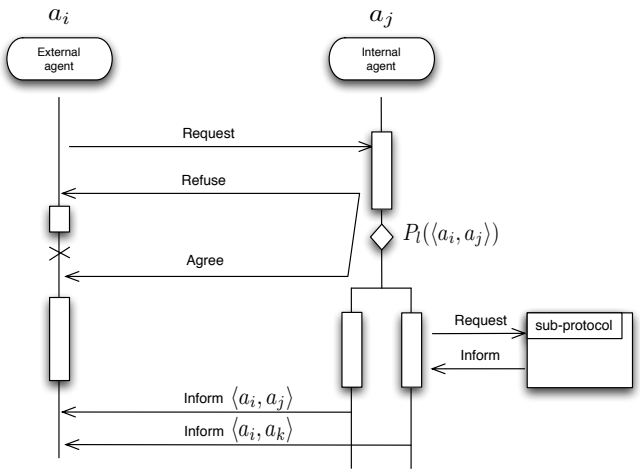


Fig. 5: Access protocol for new agents.

which considers the *choice homophily* between the agents. To obtain the probability distribution, the choice homophily between two agents should be divided by an appropriate constant ρ that indicates the degree of precision to consider two agents equal. The r parameter is a homophily regulator. When r is zero, the system shows no homophily (i.e, agents are not grouped by similar services). As r grows, links tend to connect agents with more similar services. Basically, r makes the system create communities with similar services [23].

– If agent a_j decides to establish a link with a_i, it sends a inform message to a_i with the link information ($\langle a_i, a_j \rangle$). Otherwise, a_j forwards the request to one of its neighbors a_k randomly selected. The process is repeated until agent a_i receives an inform message with its neighbor ($\langle a_i, a_k \rangle$) and establishes a connection with it. The number of connections that an agent establishes is predefined by the system. Note that the link establishment process uses a random walk strategy and a probability based on homophily to find neighbors. The reason to use this random strategy, instead of a strategy based only on homophily criterion, is to give new agents the chance of establishing links not only with similar agents, but also with dissimilar ones. Links between dissimilar agents allow agents to locate other agents communities in a few steps.

Agents have a greater probability of establishing connections with other agents if they provide similar services (value homophily) and play similar roles (status homophily) in the system. As a result of this behavior, communities of similar agents are created in a decentralized way. The resulting system structure is a network based on homophily, which grows according to a simple self-organized process. The construction process of a growing network ensures that the oldest nodes have a higher probability of receiving new links than the newest ones. Therefore, the total number

of neighbors an agent has will depend on its age. The average degree of connection of a network built following this process follows an exponential distribution [14].

Because the homophily condition is a probability function, it allows new agents not only to establish 'direct connections' between agents with similar attributes (services and roles), but also between agents that are not similar. These connections allow agents to interconnect communities and locate other agents efficiently in a few steps by using only local information [23].

An example of the resulting structures is shown in Figures 6 and 7. The networks of these figures represent the structure of a system with 1,000 agents. Each node of the network represents one agent that plays one role and offers one service. The color of the node represents the organizational role. Nodes with colors in the same range mean that their roles are close to each other in the organizational ontology (see Figure 4). Information about the structural properties of these networks such as average degree, path length, or clustering coefficient are described in Table 1. In this table, we have included the average choice homophily in the network $\overline{CH}$. This parameter measures the average homophily between an agent and its neighborhood. $\overline{CH}$ is calculated as follows:

$$\overline{CH} = \frac{\sum_{\forall a_i \in A} \sum_{\forall a_j \in N_i} \frac{CH(a_i, a_j)}{|N_i|}}{|A|} \tag{6}$$

The network shown at the top of Figure 6 is created considering only value homophily (φ = 0, see Definition 4). In this network, the agents are grouped based on similarity between services. The groups are tightly connected internally and there are few links that connect to other groups. Note that since organizational information has not been considered, in some communities, agents that offer similar semantic services but play different roles are connected. The communities obtained through a clustering algorithm are shown at the bottom of Figure 6. The clustering algorithm is based on the use of eigenvectors of the adjacency matrix. The x-axis shows the components of the first non-trivial eigenvector. The y-axis shows the components of the second non-trivial eigenvector. The number of clusters obtained is 10. The clusters are clearly defined and loosely connected with other clusters. The network shown at the top of Figure 7 is created taking into account only status homophily (φ = 1). In this case, the consideration of information from a higher level of abstraction, such as organizational roles, facilitates the interaction among groups of agents that offer different services. At the bottom of Figure 7, the clusters obtained are shown. Although the clusters in the figure cannot be easily distinguished, the algorithm detects 10 clusters again. The distances between different communities has been reduced considering organizational information, thereby facilitating the navigation among the communities. Nevertheless, when a query arrives at the appropriate community, it is more difficult to locate a service since no information related to services has been considered during the self-organization process. Section 6 discusses the effects of considering organizational information in the network creation process and in the service discovery process.

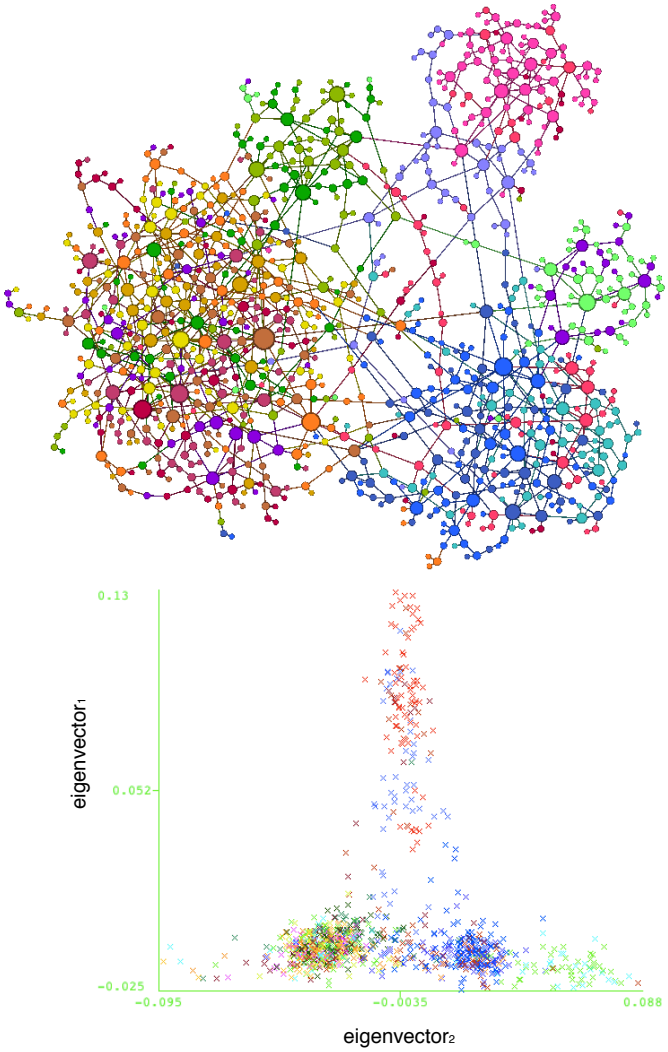


Fig. 6: System with 1000 agents, 16 roles, and one semantic service per agent. Each node color reflects one role. Similar colors reflect that the roles are close to each other in the organizational ontology (see Figure 4). (Top) The network structure reflects the effects of introducing choice homophily to the system with the parameter $\varphi = 0$. (Bottom) Communities obtained through a clustering algorithm based on eigenvectors of the adjacency matrix [13].

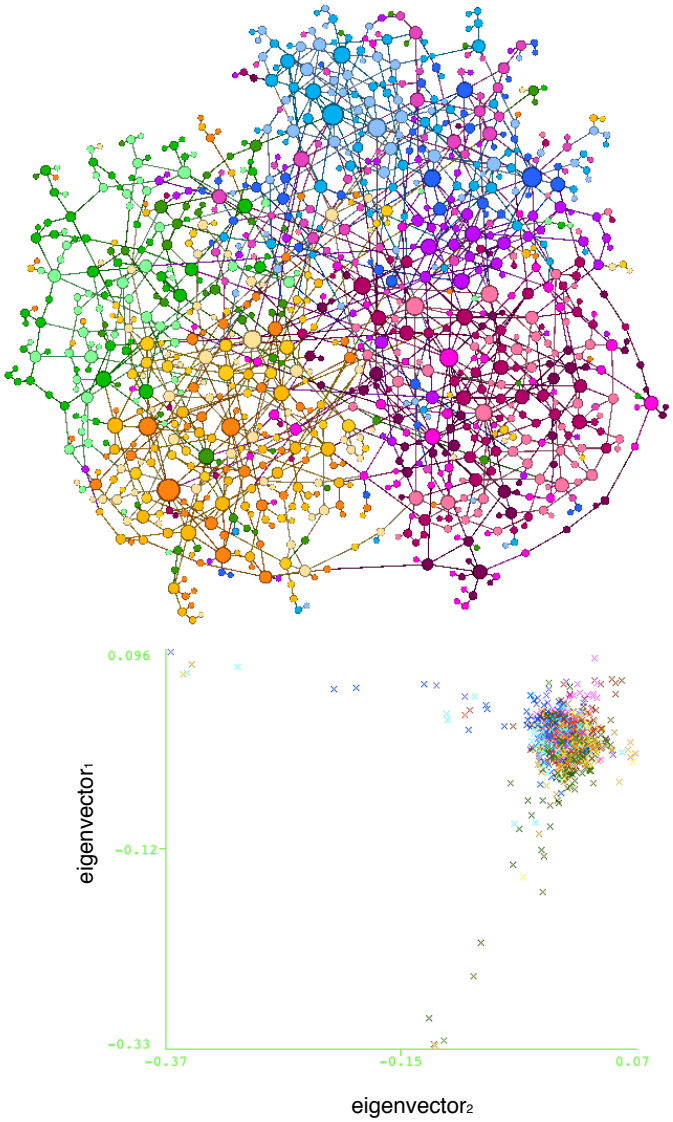


Fig. 7: System with 1,000 agents, 16 roles, and one semantic service per agent. The network structure reflects the effects of integrating choice homophily in the system. Each node color reflects one role. Similar colors reflect that the roles are close to each other in the organizational ontology (see Figure 4). (Top) The network structure reflects the effects of introducing choice homophily into the system with the parameter $\varphi = 1$. (Bottom) Communities obtained through a clustering algorithm based on eigenvectors of the adjacency matrix [13].

5.2 Decentralized Service Discovery Using Homophily

In open, large, and dynamic systems, agents should rely on local information during the service discovery process for several reasons. One reason is to prevent dependence on a single point of failure. Another reason is to avoid the effects of changes in the system structure. A third reason is that global information may not be available in open and dynamic systems. In this situation, it is important to provide agents of mechanisms that are based on local information. In this section, we describe a service discovery process that relies on local information about the direct neighbors of the agents. Agents are able to locate the required service with only this information.

The selected algorithm for service discovery in the system is an extension of the Expected-Value Navigation (EVN) algorithm [41], which is a greedy, mixed algorithm that considers local information related to the similarity and the degree of connection. It has been modified to use choice homophily as the similarity measure that integrates organizational information with the service description. The proposed algorithm is called Choice Homophily Navigation (CHN). This algorithm performs as follows. When an agent a_i looks for an unknown target agent a_t (which provides a required service s_t and plays a certain role r_t), it sends a query to the most promising agent in its neighborhood. Likewise, when an agent a_i receives a query about a service that it cannot provide, it forwards the query to the most promising agent in its neighborhood (see Algorithm 1). The most promising neighbor, $a_j \in N_i$, is the most similar neighbor to the target agent a_t and that has the highest number of connections. This process is repeated until an agent that offers a service that is 'similar enough' is found or when the TTL (Time To Live) of the query ends. The criterion of 'similar enough' is established by the agent that generates the query as a semantic similarity threshold ε.

The selection function that calculates the most promising neighbor a_j of an agent a_i to reach the agent a_t is:

$$\pi_i(a_t) = \underset{a_j \in N_i}{\operatorname{argmax}} P_s(\langle a_j, a_t \rangle) \tag{7}$$

For each neighbor a_j, $P_s(a_j, a_t)$ determines the probability that the neighbor a_j redirects the search to the nearest network community where there are more probabilities of finding the agent a_t.

$$P_s(\langle a_j, a_t \rangle) = 1 - \left(1 - \left(\frac{CH(a_j, a_t)}{\sum_{a_j \in N_i} CH(a_j, a_t)} \right) \right)^{|N_j|} \tag{8}$$

This probability uses homophily-based factors (choice homophily CH) and degree-based factors (number of neighbors $|N_j|$) to explore the network.

An example of this process is described considering the scenario presented in Section 3 (see Figure 1). Agent a_i should choose the most promising neighbor from its neighborhood (a_n,a_j, or a_k), to forward the query q. To do that, the agent a_i applies the function that appears in Equation 7. This function considers: (i) the choice

homophily between each neighbors of agent a_i and the profile of an unknown target agent $a_t = (s_6, r_5, \emptyset, \emptyset)$ that offers the service and plays the role specified in the query q; and (ii) the connection degree of the neighbors. Assuming the values of choice homophily that appear in Figure 1 ($CH(a_k, a_t) = CH(a_j, a_t) = 0.5$, and $CH(a_n, a_t) = 0.15$):

$$\begin{aligned}\pi_i(a_t) &= \underset{a_k,a_j,a_n}{\operatorname{argmax}} \left[1 - \left(1 - \frac{0.5}{1.15}\right)^5, 1 - \left(1 - \frac{0.5}{1.15}\right)^4, 1 - \left(1 - \frac{0.15}{1.15}\right)^5\right] \\ &= \underset{a_k,a_j,a_n}{\operatorname{argmax}}[0.942, 0.897, 0.5] = a_k\end{aligned}$$

Therefore, agent a_i sends the query to the most promising agent, i.e agent a_k. This process is repeated until the similarity between a local services of an agent and the service in the query is over a threshold, or the query exceeds the TTL. In the described scenario, the process ends when the query arrives to agent a_v that is similar to the target agent a_t that a_i was looking for (see Figure 1b).

Algorithm 1 Function that analyzes the query and solves it or selects the most promising neighbor to forward the query to.

```
function ServiceDiscoveryQuery(a_t, steps)
p_max ← 0
if steps ≤ TTL then
    if CH(a_i, a_t) > ε then
        return a_i, steps
    else
        for a_j ∈ N_i do
            p ← P_s(CH(a_j, a_t), N_j))
            if p > p_max then
                p_max ← p
                a ← a_j
            end if
        end for
        steps ← steps + 1
    end if
end if
return a, steps
end function
```

6 Structure and Service Discovery Evaluation

In this section, we evaluate the proposed system structure and the service discovery strategy based on choice homophily. For the evaluation, we performed to compare structural features and the success rate of the service discovery of our proposal with other network structures, common in the complex networks area. The network structures that were considered in the experiments are:

– *Random networks* (R), where links between agents are established randomly.

- *Scale-Free networks* (SF), where links between agents are established based on the degree of connection. Agents with a high degree of connection have a greater probability of receiving a new link than agents that have a low degree of connection.
- *Networks based on value homophily among agents* (Networks based on H_v). Agents in these networks are based on the value homophily and the degree of connection.
- *Networks based on choice homophily among agents* (Networks based on CH). Links in these networks are established based on the choice homophily between agents and the degree of connection.

We also compare our search strategy for decentralized service discovery with typical search strategies that are used in complex networks. The difference among them is how the most promising neighbor is selected in each step. These strategies are:

- *Random*: a search process that uses random walks [5,17];
- *Degree*: a search process that uses only degree of connection information [50];
- *Similarity*: a search process that uses only service similarity information [52,46, 3];
- *VHN*: a mixed search process that uses a combination of degree of connection and service similarity [41];
- *CHN*: a mixed search process that is based on the degree of connection and choice homophily $CH(a_i, a_j)$.

6.1 Experimental Design

Each network of this experiment was undirected and had 1,000 agents. Each agent played one role and offered one semantic web service. The agents were distributed uniformly over the roles. The role played by an agent was defined in a common organizational ontology (see Figure 4). All the experiments were performed with real semantic services. The set of semantic services used for the experiments were from the test collection OWL-S TC4 [2].

To decide the average degree of connection we evaluated its influence on the average path length. As the average degree of connection increases, the paths get shorter and it is easier to locate the target agent since agents have available more possibilities to guide the search (see Figure 8). In this case, all the algorithms perform similarly. Furthermore, the resulting networks are more robust. On the other hand, if agents join one neighbor, the resulting structure is weak and could be broken in isolated components under same failure conditions. We consider that agents when arrive to the system can establish a number of connections up to 2. This criterion creates networks with an average degree of 2.5 approximately. This degree of connection allows to analyze the worst and most interesting scenario.

Queries are uniformly generated among all the agents. This means that all the agents in the system had the same probability of generating service queries. A query

[2] http://www.semwebcentral.org/projects/owls-tc/

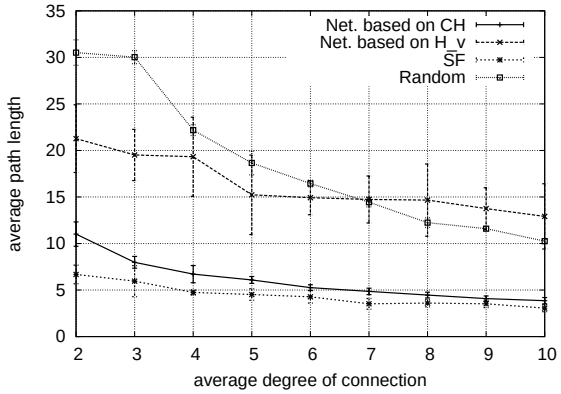


Fig. 8: Influence of the average degree of connection on the average path length on different networks: random, scale-free, networks based on H_v, and networks based on CH. The mean-shortest path length dismisses when agents have more connections.

consisted of two features that characterize the required provider agent: the semantic concepts that identified the organizational role and the semantic service description. Each query was forwarded by the agents following a criterion determined by the search strategy until the query was successfully solved or the query reached its TTL. A query was successfully solved when an agent that offers a similar enough semantic service to the target service and played a similar role to the target role was found. A similarity threshold ($\varepsilon = 0.75$) is defined. This means that the semantic similarity between the services and organizational roles was over a threshold ($\varepsilon = 0.75$).

Each experiment has been done over 10 networks for each of the structures and 5,000 queries have been generated in each network.

6.2 Influence of Value and Status Homophily

In this section the influence of φ parameter in the network structure based on CH and in the search process is also analyzed. The φ parameter balances the weight of value homophily and status homophily to determine the similarity measure between two agents (i.e., the importance of the role or service description in the overall similarity is regulated by φ)(see Equation 4). Details about the structural properties of these networks as φ changes are shown in Table 1. These properties are the following: $\overline{k}$ is the average degree of connection of the agents in the network; $\overline{c}$ is the average clustering coefficient [26] (indicates how nodes are embedded in their neighborhood); $\overline{d}$ is the average diameter of the network (the diameter is the longest graph distance between any two nodes in the network); $\overline{p}$ is the average distance between all pairs of nodes; $\overline{CH}$ is the average choice homophily in the network.

Properties / φ	$\overline{k}$	$\overline{c}$	$\overline{d}$	$\overline{l}$	$\overline{CH}$
0	2.5	0.0099	18.10	8.02	0.48
0.25	2.65	0.0062	16.20	7.05	0.42
0.5	2.70	0.0045	15.19	6.82	0.48
0.75	2.65	0.0052	15.2	6.84	0.58
1	2.67	0.0052	15.60	6.92	0.71

Table 1: Structural properties of Networks based on CH with different values of φ. $\overline{k}$ is the average degree of connection of the agents; $\overline{c}$ is the average clustering coefficient; $\overline{d}$ is the average diameter of the network; $\overline{p}$ is the average distance between all pairs of nodes; $\overline{CH}$ is the average choice homophily in the network

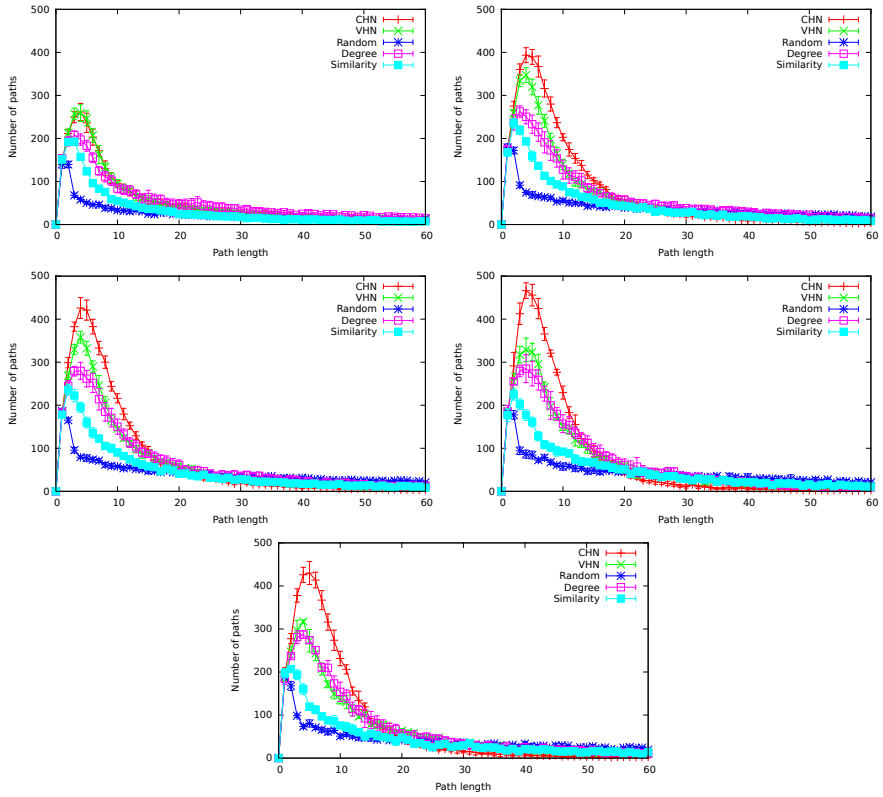


Fig. 9: Search Performance in Networks based on CH with different values for φ (Top row: φ [0, 0.25], Middle row: φ [0.5, 0.75], and Bottom row: φ=1).

Figure 9 compares the results obtained with the different algorithms in networks based on CH varying the value of φ parameter. In each graph, the φ parameter takes different values that range from 0 to 1 giving more importance to functional information (services) or to organizational information (roles). The x-axis shows the average

number of steps required in the service discovery process. The y-axis shows the number of queries that were solved in a certain number of steps before the TTL. When $\varphi = 0$, the networks have been built using only value homophily information (see Figure 6). As φ increases, status homophily appears and organizational information has more influence in the network creation process (see Figure 7).

In general, it can be observed that, independently of the value of φ, the CHN algorithm obtains the best results providing with a greater number of short paths than other traditional algorithms used in distributed environments. The consideration of status homophily (i.e., role information $\varphi > 0$) improves the results obtained by all the search strategies but specially the results obtained by the CHN algorithm.

In networks that are built based only on semantic service information ($\varphi = 0$), several small agent communities that are specialized in certain types of services emerge and there are only a few connections between communities. In these communities, agents that play different roles but offer semantically similar services could be connected directly. These features make the navigation from one community to another more complicated. Consequently, the path lengths obtained by the search strategies are longer.

In the networks that are built based on a combination of organizational and service information ($\varphi = [0.5, 0.75]$), networks are not divided in loosely connected communities. Agents self-organize taking roles and services into account. Agents are connected to agents that play similar roles that are situated close to each other in the organizational ontology. Specifically, the best parameter configuration is when $\varphi = 0.75$. Furthermore, with low probability, agents also establish connection with agents that play completely dissimilar roles. This makes the navigation between communities easier. Consequently, the path lengths obtained in the search process are shorter. Organizational information is useful in guiding the search. Nonetheless, in networks that have been built with only this information, the search becomes complicated ($\varphi = 1$). The reason is that once an agent arrives to a community that has a high probability of containing the required service, it does not have any criteria for determining which agent is better for reaching the required service since all of them play similar roles.

In general, it can be observed that the consideration of both status and value homophily in the network structure improves the service discovery process. Also, the CHN algorithm significantly reduces the length of the paths to the target (see Figure 9). The success rate obtained by CHN increases when organizational information is included in the decision process, $\varphi > 0$ (see Table 3). The algorithm based on the degree of connection provides similar success rate; however, the mean path length is almost double the mean path length obtained with CHN algorithm.

6.3 Comparison with other Complex Networks Models

The aim of the second test set is to compare our proposed structure with other complex network models. In each model, we evaluate the performance of different search strategies. Specifically, we focus on two metrics: the average number of steps required by a search strategy to find the target agent and the percentage of success-

φ \ Algorithm	Random	Degree	Similarity	VHN	CHN
0	39.22 ±3.63	73.47 ±9.96	45.25 ±7.52	71.84 ±13.31	71.96 ±13.70
0.25	64.64 ±2.18	89.19 ±2.39	57.03 ±3.08	85.51 ±3.57	94.14 ±0.93
0.5	65.85 ±1.79	88.63 ±1.96	63.15 ±9.48	88.71 ±3.27	95.55 ±1.78
0.75	65.48 ±2.74	88.85 ±4.00	60.23 ±5.94	87.03 ±3.11	96.91 ±0.85
1	64.64 ±2.18	89.19 ±2.39	95.14 ±0.93	57.03 ±3.07	85.51 ±3.57

Table 2: Success rate (%) obtained by different algorithms in networks with φ values from 0 to 1.

φ \ Algorithm	Random	Degree	Similarity	VHN	CHN
0	31.72 ±1.70	25.02 ±5.84	19.36 ±3.38	20.57 ±2.88	20.52 ±3.06
0.25	33.55 ±1.04	21.63 ±6.19	19.88 ±1.87	16.99 ±2.92	12.36 ±0.98
0.5	33.44 ±1.22	19.02 ±4.77	19.48 ±2.35	15.79 ±1.86	11.38 ±1.24
0.75	33.77 ±1.07	18.78 ±5.02	20.10 ±1.63	16.83 ±3.00	10.68 ±1.76
1	33.66 ±1.06	18.21 ±2.96	21.83 ±2.60	17.72 ±2.32	12.08 ±1.16

Table 3: Mean path length obtained by different search algorithms in networks with φ values from 0 to 1.

ful searches obtained (searches that end before the TTL) in each network structure. We performed the experiment in 10 networks for each type: networks based on H_v ($\varphi = 0$), networks based on CH ($CHwith\varphi = 0.75$), Scale-Free (SF), and Random (R). Details about the structural properties of these networks such as average degree of connection, path length, or clustering coefficient are described in Table 4. In this table, we have included the average choice homophily in the network $\overline{CH}$. This parameter measures the homophily between an agent and its neighborhood. The search strategies were those previously mentioned above (random, degree, similarity, VHN, and CHN).

With regard to the structural properties, SF and R networks are characterized by a low clustering. Moreover, the $\overline{CH}$ takes low values since choice homophily it is not considered in the network generation process. SF networks have a small diameter and short paths. Networks based on H_v and CH have low clustering values and high values of $\overline{CH}$. Networks based on CH have a higher value due to the consideration of organizational information. Note that the $\overline{CH}$ values are around 0.5. This is because the homophily criterion for establishing links do not limit agents to establishing links with only similar agents, but they can also establish links with dissimilar agents.

The results of this experiment are shown in Figures 10 and 11. The graph on the left contains four histograms (one for each network structure). In each histogram, the x-axis shows the different search strategies and the y-axis shows the average number of steps required to reach the target agent. In general, in each type of structure, the shortest paths are obtained by the search strategies that are based on the criteria used to build the network. For example, random walks perform better in random networks

Properties / Network Structure	$\overline{k}$	$\overline{c}$	$\overline{d}$	$\overline{l}$	$\overline{CH}$
Random	2.53	0.0014	19.89	8.71	0.15
Scale-Free	2.48	0.0028	9.0	4.83	0.15
Net. based on H_v	2.53	0.0093	18.10	8.02	0.48
Net. based on CH	2.66	0.0058	15.80	6.91	0.57

Table 4: Structural properties of networks.

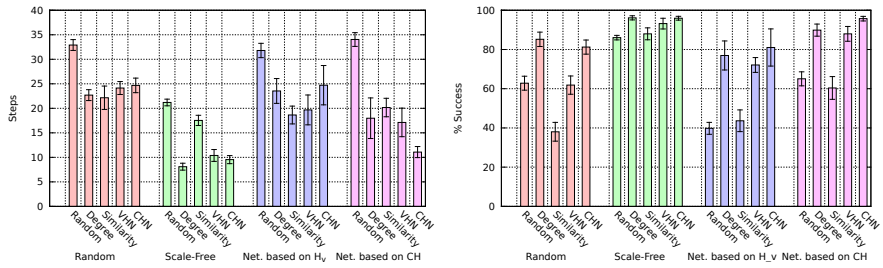


Fig. 10: Search results using different network structures with an average connection degree of 2.5.(Left) Average number of steps in the discovery process. (Right) Success rate (%) of queries. The error interval is depicted with I at the top of each bar.

or a degree-based method in SF networks. This highlights the close relationship between the search strategies and the network structures.

With regard to the average path length, the best structures that provide the shortest path are the networks based on CH and the SF networks. In the networks based on CH, the best strategy is the CHN. Moreover, strategies based on similarities, degree of connection, and the combination of both also provide short paths. In the SF networks, the best strategy is the strategy based on the degree of connection. Nevertheless, in structures of this type, strategies that combine degree of connection and similarity also obtain good results, particularly the CHN strategy. The conclusions to be drawn from this data are that Scale-Free networks and networks based on CH are structures where short paths can be found through different search strategies. Moreover, the CHN search strategy obtains good results in both structures: SF networks and networks based on CH.

In the graph on the right, the x-axis shows the different search strategies and the y-axis shows the percentage of successful searches (i.e, the percentage of queries that were solved before the TTL). For SF networks and networks based on CH the success in the service discovery process provide the best results with Degree, VHN, and CHN strategies. In SF networks, the success rate is over 80% with all the search strategies. In networks based on CH, the success rate is over the 80% in three search strategies. We can conclude that the search strategy that provides the highest success rate independently of the network structure is the CHN.

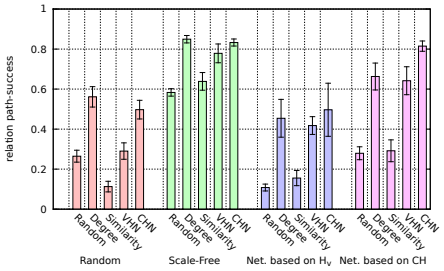


Fig. 11: Relation between the path length and the success rate (PS) in different network structures with an average connection degree of 2.5.

The path length and the success rate are closely related and can influence each other. For instance, even though short path lengths could be obtained in a network structure, this does not always mean that the structure offers good performance. The success rate could be too low, and only searches that are solved in the surroundings of the source agent (path with a low number of steps) are considered. For this reason, we have analyzed the relation between the success and the path length (PS). This relation is measured using the following equation, which takes into consideration the number of queries that were successfully solved as well as the average mean path,

$$PS = \frac{\sum_{a_i \in A} \#sq(t)}{\#Q(t)} \cdot \frac{TTL - \bar{p}}{TTL} \tag{9}$$

In the equation, $\#sq(t)$ is the number of queries generated by an agent that were solved before the TTL at a given time t. The term $\#Q(t)$ is the total number of queries generated in the system at a given time t. TTL reflects the maximum path length allowed in the system, and $\bar{p}$ is the average path length (number of steps) of a service discovery process in the system. The first term evaluates the success rate and the second term evaluates the significance of the path length. The values of PS range in the interval [0,1], where 0 means that none of the queries generated in the system where solved, and 1 means that all the queries were successfully solved and by direct neighbors ($\bar{p} = 0$). This situation is possible in regular networks where each node is connected to the rest of the nodes of the network.

The results of PS obtained with the different network structures are shown in Figure 11. In this figure, the x-axis shows the different search strategies, and the y-axis shows the PS. In general, it can be observed that the success rate is the information that has the most influence on the PS. The PS values obtained confirm that SF networks and networks based on CH obtain the best results. The CHN is the strategy that offers better performance in different network structures than the other strategies that are commonly used in complex networks.

The results of this experiment allow us to conclude that degree-based algorithm and CHN perform well independently of the underlying network structure. Moreover, networks based on CH and Scale-Free networks have desirable characteristics for

providing an underlying structure to a discovery system. They have high percentage of success in the search process and short paths. Therefore, the traffic generated by the service discovery process is reduced and its efficiency improved.

6.4 Tolerance to Failures

Networks could be sensitive to failure or deliberate attacks. The most critical situation for networks where the distribution of the degree of connection follows an exponential or power-law distribution is when deliberate attacks on highly connected nodes are produced. In the context of service discovery, the failure of an agent implies the removal of all its links. We evaluate the performance of the service discovery process in different network structures as the number of failed nodes increases. We have studied the failure tolerance of networks based on *CH* and SF networks in different situations, but in this article, we only include a subset of these experiments. Specifically, we focus on 'sabotage' situations in two particular network structures, networks based on *CH* and SF networks. We have only included these two types of networks since both of them obtained the best results in the experiments described above. We only show the results obtained in scenarios where deliberate failure ('sabotage') is produced. We consider this scenario to be a more interesting scenario than random failures since the network structures and the service discovery strategies are evaluated in the worst case.

Figure 12 shows the behavior of networks based on *CH* and Scale-Free networks under 'sabotage'. The results for the Scale-Free networks are shown in the left column. The results for the networks based on *CH* are shown in the right column. The top row shows the average number of steps required to reach the target agent as the number of failures of highly connected agents increases (50,100,150, and 200). Taking into account the average path length of the successful searches, the results indicate that the SF networks obtain shorter paths than networks based on *CH* in the presence of failures. When the number of failures is 50, this difference is not as significant. This difference is greater when the number of agents that have failed is between 50 and 100. The main reason of this fact is that SF networks are more sensitive to failures; therefore, as the number of highly connected nodes that fail increases, the network is divided in a higher number of isolated parts where the only successful searches are those which can be solved by a nearby agent.

The middle row shows the success rate in the discovery process as the number of agents that fail increases. The graph on the right shows that the rate of success in networks based on *CH* is over 40% until 150 agents are removed (using the CHN search strategy). In the case of SF networks (left), the success rate is seriously reduced when more than 50 agents fail (5% success when the number of deleted agents is 150). Note that, in the case of SF networks, the tests range from 50 to 150 failure agents. This is because in the experiments with 200 failure agents, the network is disconnected in so many isolated parts that it is not possible to find the required services. The PS relation is shown in the bottom row. The network structure based on *CH* offers the best results where path length and success rate under intentional failures is concerned. Moreover, the strategy based on *CH* also obtains the best results

in networks based on homophily and almost the same as the strategy based on the degree of connection in SF networks.

In general, it can be observed from the results that since SF networks have a distribution of the degree of connection that follows a power-law, SF networks are more vulnerable to intentional failures. This is due to the existence of hubs that concentrate network connections. If an attack is addressed to these hubs, the network can be broken into isolated groups. However, a growing network based on choice homophily generates structures with an exponential degree distribution. In networks based on choice homophily, the size of the hubs is limited. For example, whereas a SF network with 1,000 agents and an average degree of connection of 2.5 can contain nodes connected to more than 75 nodes, a network with an exponential distribution degree of connection barely arrives to hubs with 20 connections with other nodes. The absence of highly connected hubs makes the network more robust under a deliberate attack.

Based on these results, we consider that the most suitable structure for the self-organization of services is the network structure based on choice homophily between agents. When agents join other similar agents, the network has short paths to locate the desired services. Moreover, a hill climbing mechanism can be implemented with a high success rate in the service discovery process. Therefore, despite the network structure has not small-world network properties, its performance is in the same order of magnitude.

Finally, the proposed CHN algorithm, which uses node degree and the homophily information, performs as the best method in SF networks and CH based networks. Networks based on CH are more robust than SF networks under targeted attacks. Therefore, homophily-based networks seem a good structure for self-organized systems and the CHN algorithm offers a good performance even in other type of networks that are not self-organized considering homophily criterion.

7 Conclusions

The aim of this work is to study how the integration of different areas such as Service-Oriented systems, MAS, Semantics, and Complex Networks provide the necessary tools to build a decentralized service management system. We have proposed a Service-Oriented MAS where agents offer their functionality through services. In this system, agents establish relations with other agents by taking into account a social feature that is present in complex networks and that acts as a self-organizing criterion. This feature is called homophily. Specifically, we have considered a type of homophily called *choice homophily* that is composed of the combination of *value* and *status homophily*. In the context of agents, choice homophily is based on agent attributes. *Value homophily* is based on the services provided by the agent. *Status homophily* is based on the organizational role played by the agent. Choice homophily is used to create a network based on homophily without the supervision of a central authority where agents have a greater probability of establishing links with other agents that share attributes with them (such as services and roles) than with dissimilar agents. Therefore, the system does not need an initial training period to establish its structure. The resultant structure is a growing network based on homophily. The degree of

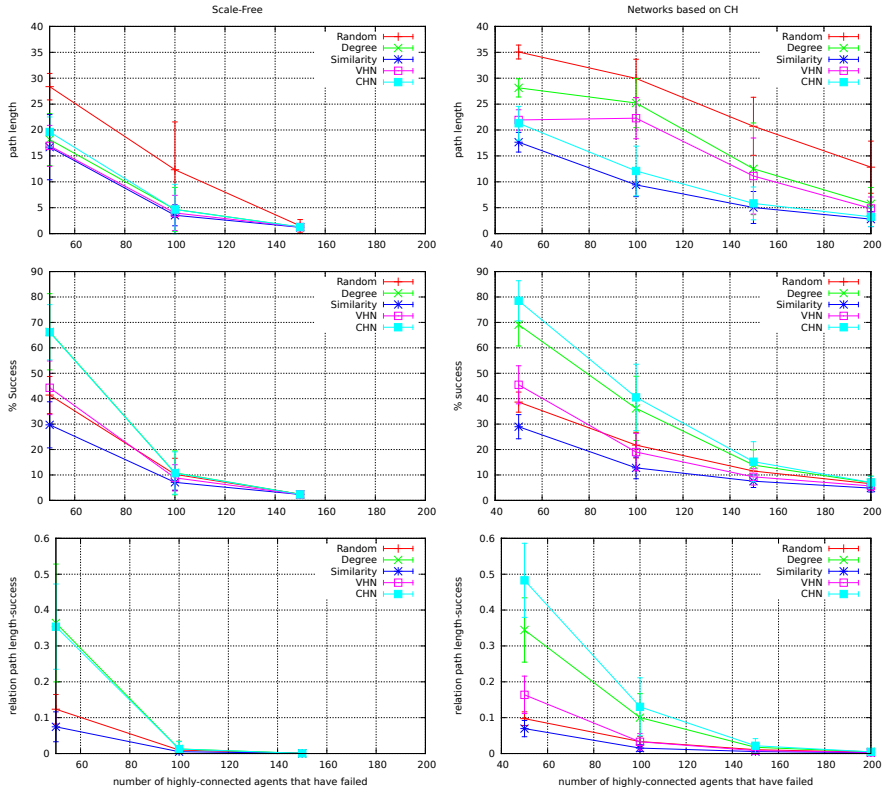


Fig. 12: Sabotage in Scale-Free networks (Left column) and in Networks based on Choice homophily (Right column). (Top row) The average number of steps required to reach the target using different search strategies as the number of agents that have failed increases. (Middle row) The success rate (%) in the service discovery process using different search strategies as the number of agents that have failed increases. (Bottom row) The relation between path length and success (PS) using different search strategies as the number of agents that have failed increases.

connection of this type of networks follows an exponential distribution. Moreover, in the presented model, agents only have to maintain their local view and do not have to store information about routes that could frequently change. The proposed algorithm for decentralized service discovery is based on semantic information and considers local information about choice homophily between agents in its decision process.

Several experiments have been performed to evaluate and compare our network model (network based on CH) and service discovery strategy (CHN) with other existing proposals in Complex Networks systems. We evaluated the influence of the inclusion of organizational information in the network structure and in the search strategy. The consideration of the organizational process in the network structure and

in the service discovery process considerably improves the performance of the system providing a high success rate and short paths. We also compared our proposal with other network structures and algorithms in complex networks. The results indicate that the service discovery strategy allows agents to locate the required service in just a few steps, not only in structures that are built following homophily criteria but also in other networks such as SF networks. The performance of the algorithm CHN in networks based on CH and SF networks is very similar considering the average path length of the searches in the discovery process and the success rate. Although networks based on CH do not have hubs nodes that facilitate the search process reducing the number of steps, agents considering the homophily information are also able to provide short paths similar to those obtained by SF structures. Furthermore, the proposed model to build networks based on homophily creates network structures with a degree of connection that follows an exponential distribution. This means that there are no hubs as in Scale-Free networks that follow a power-law distribution. Therefore, the system is more robust under critical situations such as sabotage. In these scenarios, their performance is better than other network structures such as SF networks.

References

1. Adamic, L.A., Adar, E.: How to search a social network. Social Networks **27**, 2005 (2005)
2. Basters, U., Klusch, M.: Rs2d: Fast adaptive search for semantic web services in unstructured p2p networks. In: International Semantic Web Conference, *Lecture Notes in Computer Science*, vol. 4273, pp. 87–100. Springer (2006)
3. Bianchini, D., Antonellis, V.D., Melchiori, M.: Service-based semantic search in p2p systems. European Conference on Web Services **0**, 7–16 (2009)
4. Bisgin, H., Agarwal, N., Xu, X.: Investigating homophily in online social networks. In: Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01, WI-IAT '10, pp. 533–536. IEEE Computer Society, Washington, DC, USA (2010)
5. Bisnik, N., Abouzeid, A.: Modeling and analysis of random walk search algorithms in p2p networks. In: Proceedings of the Second International Workshop on Hot Topics in Peer-to-Peer Systems, pp. 95–103. IEEE Computer Society (2005)
6. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.U.: Complex networks: Structure and dynamics. Physics Reports **424**(4-5), 175 – 308 (2006)
7. Brazier, F.M.T., Kephart, J.O., Parunak, H.V.D., Huhns, M.N.: Agents and service-oriented computing for autonomic computing: A research agenda. IEEE Internet Computing **13** (2009)
8. Cao, J., Yao, Y., Zheng, X., Liu, B.: Semantic-based self-organizing mechanism for service registry and discovery. In: 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 345 –350 (2010)
9. Centola, D., Gonzalez-Avella, J.C., Eguiluz, V.M., San Miguel, M.: Homophily, cultural drift, and the co-evolution of cultural groups. Journal of Conflict Resolution (2007)
10. Clip2: The gnutella protocol specification v0.4. `http://www.stanford.edu/class/cs244b/-gnutella_protocol_0.4.pdf` (2001)
11. Crespo, A., Garcia-Molina, H.: Routing Indices For Peer-to-Peer Systems. In: ICDCS '02: Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02), p. 23. IEEE Computer Society (2002)
12. Currarini, S., Vega-Redondo, F.: Search and homophily in social networks. World (24), 1–32 (2010)
13. Donetti, L., Munoz, M.A.: Detecting network communities: a new systematic and efficient algorithm. Journal of Statistical Mechanics: Theory and Experiment (10) (2004)
14. Dorogovtsev, S., Mendes, J.: Evolution of Networks (2003)

15. Ferber, J., Gutknecht, O., Michel, F.: From Agents to Organizations: An Organizational View of Multi-agent Systems. In: P. Giorgini, J.P. Müller, J. Odell (eds.) Agent-Oriented Software Engineering IV, *Lecture Notes in Computer Science*, vol. 2935, chap. 15, pp. 443–459. Springer Berlin / Heidelberg (2003)
16. Fu, P., Liu, S., Yang, H., Gu, L.: Matching algorithm of web services based on semantic distance. In: WISA (2009)
17. Gkantsidis, C., Mihail, M., Saberi, A.: Random walks in peer-to-peer networks: Algorithms and evaluation. Performance Evaluation **63**(3), 241–263 (2006)
18. Gummadi, P.K., Saroiu, S., Gribble, S.D.: A measurement study of napster and gnutella as examples of peer-to-peer file sharing systems. SIGCOMM Comput. Commun. Rev. **32**, 82–82 (2002)
19. Huhns, M.N.: Agents as web services. IEEE Internet Computing pp. 93–95 (2002)
20. Huhns, M.N., Singh, M.P., Burstein, M., Decker, K., Durfee, E., Finin, T., Gasser, L., Goradia, H., Jennings, N., Lakkaraju, K., Nakashima, H., Parunak, V., Rosenschein, J.S., Ruvinsky, A., Sukthankar, G., Swarup, S., Sycara, K., Tambe, M., Wagner, T., Zavala, L.: Research directions for service-oriented multiagent systems. IEEE Internet Computing **9**, 65–70 (2005). DOI http://doi.ieeecomputersociety.org/10.1109/MIC.2005.132
21. Kalogeraki, V., Gunopulos, D., Zeinalipour-Yazti, D.: A local search mechanism for peer-to-peer networks. In: CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management, pp. 300–307. ACM, New York, NY, USA (2002)
22. Kleinberg, J.: Complex networks and decentralized search algorithms. In: In Proceedings of the International Congress of Mathematicians (ICM) (2006)
23. Kleinberg, J.M.: Navigation in a small world. Nature **406**, 845 (2000)
24. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with owls-mx. In: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, AAMAS '06, pp. 915–922. ACM, New York, NY, USA (2006)
25. Klusch, M., Fries, B., Sycara, K.: Owls-mx: A hybrid semantic web service matchmaker for owl-s services. Web Semantics Science Services and Agents on the World Wide Web **7**(2), 121–133 (2009). URL http://linkinghub.elsevier.com/retrieve/pii/S1570826808000838
26. Latapy, M.: Main-memory triangle computations for very large (sparse (power-law)) graphs. Theor. Comput. Sci. **407**, 458–473 (2008)
27. Lazarsfeld, P.: Friendship as a social process: A substantive and methodological analysis. Freedom and Control in Modern Society (1954)
28. Lopes, A.L., Botelho, L.M.: Improving multi-agent based resource coordination in peer-to-peer networks. Journal of Networks **3**, 38–47 (2008)
29. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer networks. In: Proceedings of the 16th international conference on Supercomputing, ICS '02, pp. 84–95. ACM, New York, NY, USA (2002)
30. Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the xor metric. 2002. In Proceedings of the 1st International Workshop on Peer-to Peer Systems (IPTPS02) (2002)
31. McPherson, M., Smith-Lovin, L., Cook, J.: Birds of a feather: Homophily in social networks. Annual Review of Sociology (2001)
32. Michlmayr, E.: Ant algorithms for search in unstructured peer-to-peer networks. In: 22nd International Conference on Data Engineering (ICDE) (2006)
33. Newman, M.E.J.: Assortative mixing in networks. PHYS.REV.LETT. **89**, 208,701 (2002)
34. Ogston, E., Vassiliadis, S.: Matchmaking among minimal agents without a facilitator. In: Proceedings of the 5th International Conference on Autonomous Agents, pp. 608–615 (2001)
35. Ouksel, A., Babad, Y., Tesch, T.: Matchmaking software agents in b2b markets. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) (2004)
36. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities (2002)
37. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: State of the art and research challenges. Computer **40**, 38–45 (2007)
38. Perryea, C., Chung, S.: Community-based service discovery. In: International Conference on Web Services, pp. 903 –906 (2006)
39. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. In: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications. ACM, New York, NY, USA (2001)

40. Rowstron, A.I.T., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg, Middleware '01, pp. 329–350. Springer-Verlag (2001)
41. Simsek, Ö., Jensen, D.: Decentralized search in networks using homophily and degree disparity. In: IJCAI, pp. 304–310 (2005)
42. Sivashanmugam, K., Verma, K., Sheth, A.: Discovery of web services in a federated registry environment. Web Services, IEEE International Conference on **0**, 270 (2004)
43. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. Computer Communication Review **31**(4), 149–160 (2001)
44. Tsoumakos, D., Roussopoulos, N.: Adaptive probabilistic search for peer-to-peer networks. In: Peer-to-Peer Computing, pp. 102–109 (2003)
45. UDDI: Evolution of uddi, white paper available. `http://www.uddi.org/pubs/-the_evolution_of_uddi_20020719.pdf` (2002)
46. Upadrashta, Y., Vassileva, J., Grassmann, W.: Social networks in peer-to-peer systems. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences (2005)
47. Val, E.D., Rebollo, M., Botti, V.: Introducing homophily to improve semantic service search in a self-adaptive system. In: 10th Int. Conf. on Autonomous Agents and Multiagent Systems (2011)
48. Watts, D., Dodds, P., Newman, M.: Identity and search in social networks. Science **296**(5571), 1302 – 1305 (2002)
49. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**, 440–442 (1998)
50. Xiao, S., Xiao, G.: On degree-based decentralized search in complex networks. CoRR (2006)
51. Yang, B., Garcia-Molina, H.: Efficient search in peer-to-peer networks. In: Proceedings of the International Conference on Distributed Computing Systems (ICDCS) (2002)
52. Zhang, H., Croft, W.B., Levine, B., Lesser, V.: A multi-agent approach for peer-to-peer based information retrieval system. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1, AAMAS '04, pp. 456–463. IEEE Computer Society, Washington, DC, USA (2004)
53. Zhong, M.: Popularity-biased random walks for peer-to-peer search under the square-root principle. In: Proc. of the 5th International Workshop on Peer-to-Peer Systems (IPTPS) (2006)