An Overview of Search Strategies in Distributed Environments

E. DEL VAL, M. REBOLLO, V. BOTTI

Universitat Politènica de València, Camino de Vera sn. CP 46022 Valencia (Spain)
    E-mail: {edelval,mrebollo,vbotti}@dsic.upv.es

Abstract

Distributed systems are populated by a large number of heterogeneous entities that join and leave the systems dynamically. These entities act as clients and providers and interact with each other in order to get a resource or to achieve a goal. To facilitate the collaboration between entities the system should provide mechanisms to manage the information about which entities or resources are available in the system at a certain moment, as well as how to locate them in an efficient way. However, this is not an easy task in open and dynamic environments where there are changes in the available resources and global information is not always available. In this paper, we present a comprehensive vision of search in distributed environments. This review does not only considers the approaches of the Peer-to-Peer area, but also the approaches from three more areas: Service-Oriented Environments, Multi-Agent Systems, and Complex Networks. In these areas, the search for resources, services, or entities plays a key role for the proper performance of the systems built on them. The aim of this analysis is to compare approaches from these areas taking into account the underlying system structure and the algorithms or strategies that participate in the search process.

1   Introduction

Nowadays, there is a trend towards the design of open systems that are populated by a large number of entities that interact with each other in order to share their resources or achieve a complex goal. The entities that are part of these systems change in order to cope with environmental changes, such as new client requirements or the emergence of new business processes (Wei & Blake, 2010). Therefore, under these conditions, the management of the information about which entities or resources are available in the system at a certain moment, as well as how to locate them in an efficient way are considered to be challenges.

Throughout the last decade, the research done on search strategies in distributed environments has received important contributions from traditional research areas such as Peer-to-Peer (P2P) systems (Lua et al., 2005; Vanthournout et al., 2005; Risson & Moors, 2006; Meshkova et al., 2008). The work on this area has produced important influences in other areas that also deal with the issue of search in distributed environments, such as Service-Oriented Environments (SOE) (Bachlechner et al., 2006; Hughes et al., 2010), and Multi-Agent Systems (MAS) (Ben-Ami & Shehory, 2005; Val & Rebollo, 2007). These areas have adapted and extended architectures and algorithms that were initially proposed in P2P to deal with specific domain requirements. Moreover, in new distributed systems there is a growing interest in the area of Complex Networks (CN) (Watts, 2004). CN present new, less rigid structures that are inspired in social, biological, or technological networks, and algorithms that facilitate the search in distributed environments that consider local knowledge (Kleinberg, 2006).

In this article, we present an analysis of existing works that deal with search in distributed environments, such as P2P systems, Service-Oriented Environments, Multi-Agent Systems, and

Complex Networks. There are several review articles that have analyzed contributions to deal with search challenges in distributed environments (Lua et al., 2005; Risson & Moors, 2006; Meshkova et al., 2008; Vanthournout et al., 2005). However, all of them are focused on the area of P2P. For instance, in the review presented by Lua et al. (2005) they focus on the architectures and overlay schemes in the area of P2P. The work of Risson et al. (2006) makes an extensive review of P2P search methods. Meshkova et al. (2008) present a taxonomy of P2P systems and provide an overview of peer-to-peer overlays. A taxonomy for resource discovery systems based on eight design aspects that are used to compare a set of resource discovery systems is presented by Vanthournout et. al (2005). In this paper, we present a more comprehensive vision of search in distributed environments than previous works. This review not only considers the approaches of the P2P area, but also the approaches from three more areas: SOE, MAS, and CN. In these areas, the search for resources, services, or entities plays a key role for the proper performance of the systems built on them. The aim of this analysis is to compare approaches from these areas taking into account the underlying system structure and the algorithms or strategies that participate in the search process. In this analysis, we highlight the common features and differences that are present in the proposals as well as their weaknesses and strong points. Finally, taking into account the analyzed weaknesses, we describe a set of open issues and important aspects that should be taken into consideration in these type of systems.

The rest of the article is structured as follows: in section 2, we describe the scenarios where the search process plays an important role. In section 3, the approaches from different areas are analyzed considering structural and search dimensions. We have grouped the analyzed works by the underlying structure: centralized, distributed, or decentralized. Finally, in section 7, a set of open issues and conclusions are described.

2   Background

The challenge of search is present in many environments. In this article, we focus on a set of areas where this issue plays a critical role for the efficient performance of the system. Specifically, we analyze proposals from P2P, SOE, MAS, and CN. These four areas are similar since they face similar problems in different scenarios to locate different types of resources. These resources may vary within the context or the area of the approach. For instance, in P2P scenarios the majority of the proposals refer to the location of different types of files. In service-oriented scenarios the resources are services. In MAS, agents and services are the resources to be located. In CN scenarios the resources are entities or nodes situated in the network. In this section, we present each area, its context, how it is related to the other areas, and the entities that participate in the search process.

Peer-to-peer (P2P) systems appeared as an alternative to client-server systems. In P2P systems, the participants can play client or server roles depending on the interest at each moment. Peers are situated in an overlay network where they are connected to other peers. Peer-to-peer overlay networks are distributed systems where there is no a centralized control or a hierarchical organization (Lua et al., 2005). These networks overlay on the Internet Protocol (IP) networks. P2P systems have been used for sharing media content, services, communication, or processing capabilities. A peer looking for a resource sends a message that navigates the network following some criteria to find the peer with the required resource. In the community of P2P systems, there is a lot of work related to how the resources can be organized and how to locate them efficiently (Risson & Moors, 2006; Lua et al., 2005; Meshkova et al., 2008; Vanthournout et al., 2005). One of the aspects that determines the search strategy used by the peers is the structure of the overlay network where they are located (Lua et al., 2005). These structures range from centralized topologies to completely decentralized ones (Hughes et al., 2010). Considering the structure of the system, the search strategies vary from completely informed ones, where a set of entities has a global knowledge of the system to blind strategies where all the entities are equal and only have local knowledge.

P2P systems have a significant influence and inspire structures and search strategies in Service-Oriented Environments (SOE) and Service-Oriented Architectures (SOA) (Schmidt & Parashar, 2004; Hughes et al., 2010). We consider SOE as the systems based on SOC, i.e. systems that utilizes services as the basic constructs to support the development of rapid, low-cost and easy composition of distributed applications. The structures presented in P2P systems have been also used in Service-Oriented Architectures (SOA). SOA is a logical way of designing a software system based on services that provide published and discoverable interfaces Papazoglou et al. (2006). SOE provide a huge number of business services and applications. In these environments, the basic components are services, which are considered to be the basic building blocks to combine in order to get more complex services. Services are platform-independent and can be described, discovered, and composed dynamically. The aim of service-oriented approaches is the cooperation among services 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 (Papazoglou et al., 2007). One of the most challenging goals in Service-Oriented Environments is to facilitate service discovery (Papazoglou et al., 2007). This goal gains more importance as the number of services grows and the systems became more dynamic. Service discovery is a key process in order to select the set of suitable services and to facilitate service compositions that fulfill the user requirements (Rao & Su, 2004). Several proposals have been presented to deal with service discovery (Meshkova et al., 2008). Some of them are based on centralized paradigms such as registries. Federated registries, or systems based on decentralized P2P approaches, have been proposed to face typical drawbacks related to centralization. Moreover, the area of SOE, along with Semantics, makes an important contribution to the improvement and axiomatization of the service discovery process. As Semantics we understand the introduction of machine interpretable languages in the descriptions of resources. Therefore, Semantics plays an important role in reducing the participation of the user in the service discovery process. Specifically, the inclusion of Semantics in the service discovery implies the use of ontologies and semantic markup languages such as OWL-S (Martin et al., 2004), SAWSDL (Martin et al., 2007), or WSMO [1], to semantically describe the services and the use of mechanisms to interpret and reason about the semantics, which provides more accurate results in the service discovery process.

Service-Oriented Computing and Service-Oriented Architectures are gaining in importance in the industry. However, the majority of efforts have been centered on the execution of individual services. In the last few years, there is a trend in SOE to provide higher levels of functionality to the services in order to facilitate the collaboration among them. This trend brings additional considerations to the services that take part in the SOE. In order to create more complex 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 could be seen as agents that participate in a Multi-Agent System (MAS) (Huhns et al., 2005; Brazier et al., 2009). Multi-Agent Systems (MAS) are populated by agents that are aware of what is happening in their environment and decide to perform local actions (behaviors) based on their observations. Therefore, the environment influences agent decisions, and the agent actions modify the environment. Agents are able to learn about previous experiences and update and reason about their information in order to improve their decisions and actions. Moreover, agents are social entities that are aware of the existence of other agents. This awareness facilitates cooperation and collaboration among them to achieve individual or collective goals that cannot be achieved with individual capabilities or knowledge (Huhns, 2002). Accordingly, agents need to look for other agents to collaborate with. The solutions proposed in this area are close to the solutions proposed by SOE but incorporate characteristics of MAS such as organizational information, argumentation strategies, trust, or reputation among others.

---

[1] http://www.w3.org/Submission/WSMO/

Traditionally, distributed systems have a specific purpose and the entities that are part of them are combined to achieve the required results. However, there are other trends, such as Complex Systems that are composed of a large number of entities, which are capable of exchange information with one another and with the environment, and display an organization without any external organizing principle being applied. An example of Complex Systems are Complex Networks (Amaral & Ottino, 2004). Complex Networks provide more realistic structures based on features that are present in many biological, social, and technological networks (Wang & Chen, 2003). These models offer a new range of opportunities where interactions between entities are not established at design time, but rather depend on interest, trust, or reputation relationships. Links established in these networks are less rigid than the ones fixed in other traditional systems. Some of these Complex Network models have interesting properties that facilitate searches in distributed environments. Several models have been proposed to simulate the structure of these Complex Networks. These models try to reflect how links are established between entities, and they are characterized by node degree, clustering coefficient, and network diameter. The most well-known models are Small-World and Scale-Free (Watts, 2004). Moreover, there is a set of Complex Networks, called *navigable networks*, where short paths between two random entities can be found using only local information (Watts et al., 2002; Simsek & Jensen, 2005; Kleinberg, 2006). This raises the question of which criteria should be followed by the entities in these models in order to establish links and to guide the search towards the target. As in previous environments, the search strategy and its success depend considerably on the structure of the network.

The four presented areas deal with search challenges. In P2P systems, peers are looking for resources such as files. SOE try to provide reusability of services; therefore, they should provide mechanisms to locate the required services. Moreover, business processes are also seen as composition of services. To facilitate this task, service discovery plays an important role. In MAS, agents have to deal with complex goals that cannot be achieved by themselves. They need the collaboration of other agents. The challenge here is how to locate the agents that offer the required functionality. Finally, CN present model structures with interesting properties for distributed environments, and more specifically for decentralized searches. In all these areas, the search for resources, services or entities is a key issue. The following sections describe and analyze how these areas have dealt with search challenges.

3   Structures and Search Strategies

The areas of P2P, SOE, MAS, and CN confront the problem of searching in distributed environments. One of the criterion that influences how to deal with the search problem is the underlying structure of the system. There are some previous review articles that have analyzed contributions to deal with search challenges in distributed environments that have also considered this criterion to analyze the proposals. Basically, they consider two main groups: structured and un-structured systems (Lua et al., 2005; Meshkova et al., 2008; Hughes et al., 2010). In the review presented in this paper we have divided the proposals considering the underlying structure and which entities are responsible of the search task. Specifically, we have organized the approaches the following groups: centralized, distributed, or decentralized. In centralized systems the search process and the resource management rely on a central entity. In distributed environments, these duties rely on a set of entities. Finally, in decentralized structures each member of the system is responsible of the resource search and data management tasks. In each of these groups, we describe the works of P2P, SOE, MAS, and CN. Approaches from CN are only present in the decentralized group. This is because of the fact that all the proposals in the area are loosely-structured and decentralized. Note that in this review we mainly focus on decentralized and distributed approaches due to they are more appropriate than centralized approaches to deal with the management of resources in distributed environments. We include the section of centralized approaches to have a complete overview of all the approaches. At the end of each group, a discussion about the proposals, considering structural and search criteria, is presented. Before

dealing with the analysis of the works, we are going to establish a set of characteristics that will be used to analyze the approaches.

3.1 Evaluation Criteria

The following characteristics are considered in the analysis of the approaches. These features are organized in two dimensions: (i) the structural features related to the underlying system where the entities are located; (ii) the main features of the search process (see Figure 6.3.5):

- Structural: this dimension refers to the structure of the system where the entities are located and where they are going to look for resources, services, or other entities. Three main structures are considered: centralized, distributed, and decentralized. Within this dimension, we also consider a set of related aspects. Each aspect is evaluated with a symbol $(+, \sim, -)$.
  - Scalability: how the system behaves when the number of entities increases (+ scalability, $\sim$ limited scalability, - no scalability).
  - Robustness: tolerance to failures of entities that participate in the search process (+ robustness, $\sim$ limited robustness, - no robustness).
  - Structural dependence: degree of dependence between the search strategy and the structure of the system (+ structural dependence, $\sim$ partial dependence, - no dependence).

- Search: this dimension refers to the type of search used in the system: blind (i.e., flooding, broadcast, random), or informed (i.e., local or global knowledge, semantic or syntactic information, historical information). Within this dimension, we have analyzed a set of aspects. Each aspect has been evaluated with a symbol $(+, \sim, -)$ except the knowledge aspect that has been evaluated using the labels "Local" or "Global".
  - Adaptability: determines if the search approach is applicable to different resources and systems without significant changes (+ adaptable without changes, $\sim$ adaptable with some changes, - no adaptable).
  - Accuracy: determines if the search results contain relevant resources (+ accuracy, $\sim$ reduced accuracy, - no accuracy). This term is considered as precision in the area of Information Retrieval.
  - Traffic: determines the number of messages generated in the network to locate the resource (+ could overload the system, $\sim$ generates moderated traffic, - reduced traffic).
  - Semantic Information: determines if semantics are considered in the system during the search process or to establish the structure of the system (+ semantics, - no semantics).
  - Knowledge: determines if entities that participate in the system have a partial or a global view of the system (local or global).

Note that not every aspect in each dimension is explained when we describe the approaches, since some of them remain unclear or unspecified by the authors. Also, works that are similar are grouped together for their analysis. In Figure 6.3.5 a schematic overview of the analysis is shown. The table groups the proposals by areas. An approach is analyzed in each row of the table. The first column describes the structure. The second column contains the first author of the article where the proposal is described and the year of the publication ([authorYear]). Moreover, the search strategy is described briefly. The three following columns describe the criteria of the structural dimension. The rest of the columns describes the search dimension. In the next section, the approaches of the table are described in detail.

4 Centralized Approaches

These systems are characterized by an entity that has all the information about the resources and services that the rest of the entities offer to other members of the system. Moreover, this central entity has a set of capabilities for the resource location in order to facilitate coordination.

Centralized approaches are appropriate for systems with a low number of entities. The search process is fast and considers all the information available in the system. This global knowledge provides efficiency, and if a required resource or service exits, it will be found. However, central entities could be a bottleneck if they have a very limited capability, or 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 resources and services seriously affects the robustness of the system. Proposals based on this paradigm are analyzed below.

In centralized approaches for P2P systems, there is a *super-peer* that is responsible for the management of the resources. The rest of the peers ask the *super-peer* to locate the desired resource (Yang & Garcia-Molina, 2003). An example of this is a file-sharing system called *Napster* (Gummadi et al., 2002). In this system, there is a server that contains a centralized index over the data of every node. It is the only *super-peer* and all other nodes are clients. A client sends a search request of a resource that contains a set of keywords to the super-peer. The super-peers searches matches between the keywords provided by the client and the keywords in its local registry. Then, the super-peer answers with the peer that has the required resource and the client establishes a direct connection with the provider peer. A further development of this model consists on a set of *super-peers* instead of a unique super-peer.

In SOE, a clear example of a centralized approach to deal with service discovery is the UDDI registry that stores a set of service descriptions and supports a discovery mechanism that is based on keywords [2]. Registries of this type have information about all the available services in the system. A typical scenario is a registry that receives queries with keywords. The registry has an engine that engages query keywords with keywords from the stored service descriptions. The matched services are returned as a candidate answer set, and the user browses them in order to find one that fulfills the requirements. The search algorithms are very simple and do not take into consideration any cross-correlations between services or quality features. If the search process fails, the user has to repeat the query with new keywords. A summary of this type of service discovery can be found in (Bachlechner et al., 2006). The drawback of the UDDI-like registries is that there is no consideration of semantics in their discovery process. The main improvement in them has been done by the inclusion of semantics in the service descriptions and in the discovery algorithms. As an example, Srinivasan et. al (2004) extend the UDDI registry to include semantics in the service discovery process to obtain more accurate results. Semantic markup languages provide a formal and explicit specification of shared concepts. These languages facilitate the description of services and queries with a logic formalization. Markup languages exploit ontologies to facilitate sharing, reuse, composition, and mapping, which makes services computer interpretable. As a consequence, agents can reason about services to provide automatic service discovery, execution, and composition and interoperation (McIlraith et al., 2001). With regard to service discovery, semantics provide matching flexibility and accuracy considering those concepts that have the same meaning to be similar concepts even though they are syntactically different.

Many of the proposals based on centralized systems in SOE have focused their efforts on the introduction and improvement of semantic information management in order to obtain accurate and efficient discovery algorithms. One way of dealing with semantic information in the discovery process is the use of hypergraphs. Brogi et al. (2006) present a search approach that is based on semantic hypergraphs. The discovery system consists of two main modules: the *Hypergraph Builder* and the *Query Solver*. The *Hypergraph Builder* analyzes the ontology-based descriptions of the registry-published services to build a labeled directed hypergraph.This hypergraph synthesizes all the data dependencies of the advertised services. The vertices of the hypergraph correspond to the concepts defined in the ontologies that are used in the service descriptions, while the hyperedges represent relationships among these concepts (*subConceptOf*, *equivalentConceptOf*, and *intra-service dependency*). The *Query Solver* navigates the hypergraph

---

[2]http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf

considering the *intra-service* dependencies to address the discovery of (compositions of) services as well as by considering the *subConceptOf* and *equivalentConceptOf* relationships to cope with different ontologies.

Prabhu et al. (2007) present a search strategy that also uses hypergraphs to store web service information more accurately than regular graphs. The graph G is composed of vertices that represent concepts, and directed edges represent web services in W (a set of registered web services available in a central agent). The central discovery and composition algorithm works simultaneously both forwards (from input concepts provided by the user (I)) and backwards (from the set of user expected output concepts (O)) performing a breadth-first search from I and from O on G. At the end of each stage, the algorithm checks if there is a collision between the two paths. When a collision is detected, it means that a service or a service composition exists. Bailey et al. (2006) also reduce the web service discovery problem to one involving hypergraphs. The process of finding sets of web services that satisfy the user query can be reduced to finding transversals of the web service hypergraph. A transversal of the hypergraph corresponds to a set of web services that covers all the functionalities requested by the user.

The consideration of semantics improves the accuracy of the service discovery results. However, the maintenance of the service ontology graph could overload the matchmaker in highly dynamic environments where available services change frequently and, therefore, the graph with the service ontology concepts and relations also changes. Along with the idea of the efficiency improvement, Mokhtar et al. (2006) presents an approach that combines optimizations of the discovery process at reasoning and matching levels. To optimize the discovery process at the reasoning level, he uses a solution proposed by Constantinescu et al. (2003) for encoding concept hierarchies using intervals. Under the assumption that service advertisements and service requests already contain the codes corresponding to the concepts that they involve, semantic service reasoning is reduced to a numeric comparison of codes.

Regarding the task of service management in MAS, traditionally, middle-agents are the responsible entities. A middle-agent can be a *broker* or a *matchmaker* (Klusch & Sycara, 2001; Sycara et al., 2004). A *broker* agent acts as an interface between the agents that provide the services and the agents that request these services. The broker agent intermediates all the transactions. All the communications go through the broker. This broker contacts the providers of the required services, negotiates, and returns the result of the service execution to the requester (Sycara et al., 2004). In the case of a *matchmaker* agent, it has a registry with all the services, and when it receives a request, it selects the most suitable providers that could fulfill the request. This selection is sent to the requester agent that decides whether or not to contact or not the provider agents. This functionality is very similar to the registries in SOE.

In some MAS, agents functionality is seen as a set of services (Brazier et al., 2009). The entity responsible for the service descriptions management and service discovery is a matchmaker. For instance, Argente et al. (2011) present the THOMAS architecture, which contains a SF (Service Facilitator) responsible for this task. The semantic service descriptions are managed by the SF that offers a set of services to store, query, and modify information about services. To facilitate the collaboration among agents, provider agents register their services in the SF. If an agent is looking for a service that cannot be provided by a single service registered in the SF, it tries to find a service composition. A similar proposal is also presented in the work of Vazquez-Salceda et al. (2010).

There are other approaches where service discovery is carried out by the interaction of different entities. In the architecture, the service discovery is done by the collaboration of three entities: a service discovery agent (SDA), a project distributed repository (WSDir), and a semantic service matchmaker (SMA). Initially, the SDA looks for services in its own repository and also in the WSDir. This process is a first selection of services based on key-words that could be refined by the SMA. The SMA is composed by three types of matchmakers: (i) a role-based matchmaker (Fernández et al., 2006, 2008), that takes into account organizational aspects such as roles and

interactions to improve the matchmaking process; (ii) a hybrid matchmaker OWLS-MX (Klusch et al., 2006), which combines semantic I/O matching with syntactic measures from information retrieval; and (iii) Precondition-Effect matchmaker, which converts preconditions and effects of services into logic predicates to determine relations.

The described approaches deal with the problem of resource location, service discovery, or agent location in an efficient way and ensure that, if the requested resource exits in the system, it will be found. The entity responsible for the search has different names but similar duties. In P2P, it is called 'super-peer', in SOE it is a registry, and in MAS it is a broker or matchmaker. With regard to the *structural dimension*, centralized approaches are efficient when the number of entities and the workload of the system are low. If the number of services and the workload increases a registry is considered to be a bottleneck. Moreover, centralized approaches have a structural dependence since all the information is stored in a central entity or a reduced number of entities. Another drawback is that the existence of a single entity that deals with the search process in the system makes it vulnerable to deliberate attacks. This is solved partially in the approaches where there is a set of entities responsible for this task. With regard to the *search dimension*, centralized approaches could consider all the available information and they obtain accurate results. A significant difference has been introduced in SOE with the inclusion of semantics in service descriptions and in the service discovery process. Semantics improves the precision of the results and facilitates the interoperation between heterogeneous entities and the service composition process. In MAS, semantics have also been included in the descriptions of services offered by agents, as well as information about organizational aspects. Furthermore, centralized approaches are based on global information. In distributed environments such as P2P, SOE, and MAS, global information is not always available. For all these reasons, alternatives such as distributed and decentralized proposals are more appropriate.

5   Distributed Approaches

Distributed systems assign the responsibility of the information management about resources or services to a set of specific entities. These entities are also responsible for answering search requests. An important issue in these systems is how these entities are selected and organized to cope with search tasks.

5.1   Peer-to-Peer.

Approaches in P2P systems where the search system is distributed are based on a hierarchy of peers. The proposals can be divided into two main groups: one based on a set of *super-peers*, and one based on *DHT*.

An example of super-peer approaches is *Morpheus*. In *Morpheus*, super-peers store the collections of their clients and answer queries on their behalf, while the clients never answer any queries. A Morpheus peer is selected automatically to be a super-peer if it has high bandwidth and processing power. When a peer enters in the system, it queries a central server to get a list of super-peers. In the same way, *Gnutella* presents a schema based on a hierarchy of two levels: *leaf-nodes* and *super-nodes*. Super-nodes have better performance capabilities and store the documents of a set of leaf-nodes. Leaf-nodes periodically send a message to their super-nodes in order to update the information about the available documents. The idea of this organization is to reduce the flooding of the network. Moreover, there are other approaches such as *Kazaa* (Liang et al., 2005) that are based on two levels and flooding algorithms. These approaches take advantage of the capabilities of the peers in the network. Taking into account their capabilities, they distribute the information and the workload among the best-qualified peers. The existence of several *super-peers* avoids the problem of a single point of failure. The main problem is when several *super-peers* fail and other peers that are less qualified must replace them. In this situation, the performance of the system would be affected. Furthermore, in these approaches, the search

is based on keywords, which limits the possibility of finding meaningful documents that have different keywords.

In P2P systems, there is a set of proposals that base their structure on *Distributed Hash Tables* (DHT). The DHT approaches are based on hash functions that associate a numeric key (identifier) to a document or resource. There are several works that are based on DHT: *Chord* (Stoica et al., 2001), *Pastry* (Rowstron & Druschel, 2001), *Kademlia* (Maymounkov & Mazieres, 2002), and *CAN* (Ratnasamy et al., 2001) among others. Basically, the main differences among them are: (i) how they distribute the keys in the space; (ii) the mechanisms that peers use to join and leave the network; and (iii) the criteria to guide the search process. In the majority of these proposals, the cost of the search process is $O(log(n))$ where n is the number of peers in the network.

Chord system creates a ring of 2^m positions, where m is the ring dimension (Stoica et al., 2001). When a new node enters in the system, it receives an key n, which is obtained by a hash function applied to its IP address. A position identified by k in the ring is assigned to a node if the node key n is equal to or follows the position k. When a new node gets into the network, it receives all the keys that its successor in the ring has but now correspond to the new node. If a node leaves the network, it will transfer all the keys to the closest successor. When a new document is introduced in the system, it has associated a key d obtained by the application of a hash function. The node that must be responsible for the document will have the same key $n=d$. If the node is not active, the document d is assigned to the closest node with the highest key. In the search process, when a node receives a message with a key, it first looks in its finger table to determine if the node that is responsible for that key is there. If it is not, it asks to the node in the finger table that has the key that is closest to the target key. This is repeated until document d is found. The main drawback of this proposal is the robustness when the number of peers that enter and leave the system increases. In this situation, the pointers to other nodes in the system structure should be updated efficiently to guarantee the correctness of the search protocol. Otherwise, with outdated information, the performance of the system will decrease seriously.

Pastry is similar to *Chord*, but this protocol is prefix-based (Rowstron & Druschel, 2001). The node identifier (n) and the keys (k) are considered to be sequences of digits based on their IP address or public key. Nodes are distributed in a ring. Each node contains the following information about other nodes in the system: a leaf set (numerically closest nodes), a routing table (prefix-based nodes), and a neighborhood set (physically closest nodes). When a new node enters in the system, it sends a query to a predetermined server to get the address of an existent node k in the ring. Then, the new node sends a join message to the k node with its identifier (n). The message keeps all the nodes of the routing process. With this information, the new node initializes its routing table. The response to the message contains the closest node c to the new node n. The new node requests the leaf set to the closest node c and informs the leaf set nodes about its arrival. Each period T, the nodes send a message to check which nodes are alive. This process has a cost $N \cdot a \cdot T$, where N is the number of nodes, and a is the average number of neighbors. If a node leaves the ring, the leaf set is updated. The search process is performed as follows: when a node receives a message with key d, the node first asks its leaf set. If the key is in the range of the leaf set, the node forwards the query to the numerically closest leaf. Otherwise, the node looks in its routing table to forward the query to a node that shares at least one more digit with d in its prefix than the current node identifier. If that node does not exist, the current node forwards the query to a node that shares at least as many digits as its own identifier with d, but that is numerically closer than the current node identifier. The number of routing steps needed is $log(n)$, where n is the total number of nodes.

Kademlia system is based on the calculation of the 'distance' between two nodes (Maymounkov & Mazieres, 2002). This distance is calculated with XOR of the two identifiers. The XOR operation gives higher values if the numbers have differences at higher order bits. Keys and node identifiers have the same format and length. Therefore, the distance between them is calculated

in the same way. The network is seen as a binary tree of nodes organized according to their identifiers. Each node must have a contact in each subtree in which it is not contained. In the searching process, a node that is searching for a target node determines the subtree that will contain the target based on the first numbers of the key. This procedure is applied iteratively finding contacts closer to the target. Each step reduces the set of candidate nodes by 50%. Search results are obtained in $O(\log n)$ time, where n is the number of nodes in the network.

Content-Addressable Network (CAN) is a distributed system based on DHT (Ratnasamy et al., 2001). Keys of documents or nodes are hashed into d dimensional space. This space is divided into areas. Two areas are neighbor if d-1 dimensions overlap. Each node owns an area in the space and maintains a state for its immediate neighbor nodes. To explain the performance of this algorithm, a 2-dimensional space is considered. When a new node joins the system, it first discovers some node I already present in the CAN. After that, it selects a random point in the space (x, y). Then I routes to (x, y) and discovers the target node J that is at the coordinates (x, y). The area of the node J is divided, and a part of this area is assigned to the new node. After this process, all the routing tables of the J's area neighbors should be updated. When the new pair $(Key, Value)$ is inserted in the system, the process is similar to a new node insertion. The algorithm applies a hash function to the *key*, which corresponds to the coordinate x, and then applies a hash function to the *value*, which corresponds to the coordinate y. Then the node I routes the information to the correspondent node. The retrieval method is very similar to the insertion method. The length of the routing path is $(d/4) \cdot n^{1/d}$, where d is the dimensions and n the number of zones. The number of neighbors that a node maintains is independent of the number of nodes in the system. Each node maintains a list with 2·d number of neighbors. However, the average path length increases as $O(n^1/d)$.

P2P systems have also introduced the use of semantics in the search process to improve their accuracy in the results provided. An example of this is the work presented by Yu et al. (2004). They propose the use of DHT and semantics to manage the information related to available services in the system. More specifically the DHT catalog stores semantic indexes for direct and flexible service management. The DHT routing protocol that is used is Chord. This is because an item can be efficiently located in $O(\log n)$, where n is the number of nodes. The authors present an extension of the structure used in Chord. The proposal is to maintain two levels of rings: the *Category Chord ring* and the *Domain Chord ring*. The Category Chord ring is composed of super nodes. Each node refers to a semantic concept in an ontology of service categories. The Domain Chord ring depends on one of the super peers of the Category ring and contains services within that category.

5.2   Service-Oriented Environments.

A variety of approaches have been proposed to deal with the centralization problem in SOE. There are several approaches that provide a scalable web service discovery such as federations of registries, communities, or coalitions (Satyanarayanan, 2001). The majority of the proposals are based or inspired on previous works in P2P systems. An example of this is the work presented by Sivashanmugam et al. (2004). The authors present the METEOR-S Web Service Discovery Infrastructure (MWSDI). The infrastructure is based on a federation of registries. The federation has an ontology for describing the domains of the registries that take part in the federation. This information is stored in the Extended Registries Ontology (XTRO). The infrastructure of the MWSDI is a peer-to-peer network where the peers are not equal. There are four types of peers: *Gateway*, which is the entry point for registries to MWSDI and updates the XTRO when new registries join the system; *Operator*, which acts as a UDDI registry and provides extra functionality such as semantic discovery and publication of Web services; *Auxiliary*, which acts as providers of the XTRO to make it highly available; and *Client* which allows users to utilize the capabilities of the MWSDI. To model the federation, the authors propose and extension of the TModel (metadata used in UDDI to describe business and their services) for the federation

(TModel directory). In MWSDI, registries join a federation one by one providing their individual TModel to be added to the TModel directory. A service that is going to be published should provide a WSDL document annotated with ontological concepts and should provide certain criteria (names of federations, domains, ontologies) for registry selection. A GUI tool is provided to facilitate this task to the user. MWSDI supports semantic and syntactic service discovery. The process of service discovery consists on a service template sent by the user to an auxiliary peer to choose the most appropriate registry. Then, the template is sent to the operator peers of the selected registries. Moreover, the templates could be translated and propagated to other registries in the federation using the TModel directory. This approach is compatible with previous versions of UDDI and with syntactic service discovery. Although the information of the services is distributed in several registries, the Gateway peer acts as supervisor of the federation so the management of this structure is centralized and partially solves the problems of centralized approaches.

There are other approaches that could be applied to centralized and distributed structures. For instance, Skoutas et al. (2008) present a method for improving the efficiency of the service search and selection process at query time. This approach is described in a centralized and distributed environment. To improve the efficiency of the service discovery, service requests and advertisements are represented by intervals. These intervals are defined considering the input and output concepts. With this encoding, the establishment of the degree of match between two service parameters is reduced to checking the relationship between the corresponding intervals. This is similar to the work of Mokhtar et al. (2006). The approach in centralized environments is based on a single registry that contains the information about all the advertised services and is responsible for performing the matchmaking and ranking process. The registry encodes all service descriptions using multi-dimensional indexes. The approach in distributed environments is based on a grid-partitioned space. Each peer of the grid knows its own coordinates and the coordinates of the areas that are directly connected to it. This is similar to the structure presented in the P2P system CAN (Ratnasamy et al., 2001). In each area, there is a peer that stores the service descriptions whose identifier is closer to its value in the 2-dimensional space. The idea of this approach is that similar services are stored by the same peer or neighboring peers and that the results offered by peers in a particular direction subsume the ones previously found. The main problem with this approach is the lack of adaptability since the peer structure and distribution is based on intervals that are defined using a pre-defined ontology.

Pirró et al. (2010) present a system for service discovery called ERGOT (Efficient Routing Grounded On Taxonomy). This system is based on DHTs and Semantic Overlay Networks (SON). SON are flexible network organizations where nodes with semantically similar content are clustered together (Crespo & Garcia-Molina, 2004). ERGOT provides a semantic-based service discovery in distributed infrastructures such as Grids and Clouds. The system is composed of several layers: *concept*, which contains concepts used in the semantic annotations; *service*, which contains services annotated using SAWSDL; *DHT*, which follows the structure of Chord and is responsible of the service profile publication considering the semantic annotations; and *Semantic Overlay Network*. Basically, the system consists of a set of peers that are responsible for a set of service descriptions. Peers establish links following the criteria used in chord Stoica et al. (2001) and links based on the similarity of the service descriptions that they store. Each peer has a finger table that manages the Chord links, and a Semantic Annotation Table (SAT) that associates a concept with a set of service descriptions and provider peers. SAT is used by the peers to establish semantic links with other peers. The number of semantic links is limited by using a pre-defined threshold on the minimal number of service descriptions that the peer should have annotated with a similar concept of the SAT. When a service provider publishes a service in a peer, there is a mapping function that maps a key (concept) from the service description to a peer. For the discovery process, the ERGOT system allows either the use of SON, the use of DHT overlay, or a combination of both. When a peer receives a request, it checks in its own service profiles.

If it does not have the service, it looks its SAT and forwards the request using its semantic links. In order to decide which neighbors the query has to be forwarded, the peer calculates the similarity between the query and the semantic links stored in the SAT. The peer can also use of the underlying DHT to route the request. He et al. (2008) present a similar approach that is also based on DHT and semantics to deal with distributed service discovery.

There are other approaches that use other kinds of structures to distribute the registries in service-oriented environments. Cao et al. (2010) propose a tree-like structure to organize the distribution of registries. The structure presented has two kind of nodes: *registry proxy* (RP) and *registry center* (RC). The RP nodes maintain the tree structure and forward registration and discovery messages to the appropriate RC. The RP nodes have associated a set of concepts of a domain ontology and maintain a neighbor table with references to their father and children nodes. The RC nodes store the service descriptions related to a set of domain concepts. To insert a new service description advertisement, providers send a request to the root of the tree, which is a RP node, and, through a 'judge algorithm' the advertisements are sent to the appropriate RP nodes. These nodes determine if the advertisement has concepts that match their concepts and, in that case, the RP nodes send it to the children nodes (RC) where the description will be stored. Otherwise, the RP nodes discard the advertisements. The service discovery process is divided into three steps. The first step is to allocate the user request in the appropriate RP, and the second step is to map the user request terms to the concept in the OWL ontology. The third step is to forward the discovery request to the appropriated RC node. The main drawback of this approach is that it relies on a pre-defined ontology tree. The authors present algorithms to maintain a balanced tree, but the problem of centralization remains at the root of the tree since all the service requests should be processed first by the root in order to decide which RP it would be forwarded to.

Perryea et al. (2006) present a service registry as a community of services that compounds a service knowledge base at publication time. They use the idea of populations in ecology systems as the basis for their proposal. The community in the proposal is seen as a directed graph where the set of vertices represents service populations and the directed edges represent composition relationships. Providers publish their services, which are classified into a service population according to their semantics. This creates pre-composed services that could be reused for future discoveries. Although service descriptions are stored in a distributed way, the main problem is that the publishing and discovery processes start in centralized engines, which could be overloaded depending on the system demand.

5.3 Multi-Agent Systems.

As we have stated above, there is currently a trend in MAS where agent functionalities are seen as services; therefore, the structure and search techniques used in SOE and P2P are also used in these environments. Thus, in the following proposals we can find a set of similarities with the works described above.

The use of *coalitions or clusters* is an alternative for distributing the management of services in an open MAS. Nevertheless, the choice of what coalitions are going to be formed is a difficult task. This entails recursively calculating the values of the coalitions and later selecting the coalition with the best result. The calculation of the coalition values can be made in parallel, but this phase requires each agent to know the rest of the agents in the system (global knowledge). In addition to determining the best value, they have to use broadcast. Therefore, in some situations, the system could be overloaded. Ogston and Vassiliadis (2001b; 2001a) present an algorithm for consumer agents that are looking for service providers based on coalitions. In this algorithm, each agent has a number of tasks and needs to delegate to other agents. Agents are grouped with other agents that have similar tasks. The search process is carried out following a random search in the neighborhood. When a matching between the task of an agent and the service provided by other agent is found, it is considered that these agents are able to cooperate. These

agents get into a coalition that allows each agent to extend its neighbors, and the scope of search is extended for future tasks. In the case that the agent does not find another agent with the desired characteristics, the agent will look for one in other coalitions. The main drawback is that the clusters have a cluster controller that swaps free tasks each turn. Moreover, the size of the clusters is not limited and the links between agents that collaborate are fixed independently of the duration of the task.

Another way for agents to locate services in a more efficient way is the *distribution of the middle-agents or facilitators* (Mullender & Vitanyi, 1988). This approach consists of the distribution of the service directory, its memory, and the traffic of messages. Jha et al. (1998) propose to splitting the function of the facilitator among a group of agents. The system designer assigns a local matchmaker to a segment of the system in order to provide matchmaking services in its segment. The local matchmaker consults its peers or a central matchmaker whenever it cannot provide an answer to a local query. This type of solution reduces communication traffic and confines it to network segments (in which communication is fast). Moreover, it reduces message queue sizes, improving scalability and fault tolerance. This approach is applicable in systems that have a hierarchical topology, in which information sharing can be confined to local segments. In systems with very large segments, the problems of scalability are only marginally relieved by this approach. For instance, matchmakers in large segments are overloaded. Another case in which this approach is not efficient is in systems with many cross-links between segments where the coordinating tasks among local matchmakers could be greater than the benefit obtained from their distribution. Moreover, if the system structure changes, segments that previously were populated by few agents could be populated by a large number of agents and collapse the matchmaker with their requests.

Sigdel et al. (2005) present an adaptive system. The suggested framework allows automatically adaptable matchmaking methods for service location depending on the network structure and characteristics. This approach is based on two levels: the *system adaptation level* and the *node adaptation level*. In the *system adaptation level*, the system adapts itself to the changing circumstances of the network, the number of nodes, and the service load. If any one of these circumstances increases, the system introduces new matchmakers that will reduce the service load of the central matchmaker. These new matchmakers are defined in a segment of consumers and suppliers where they could be created. When some of the previous circumstances decrease, for example the service load, a mechanism unifies the segments and eliminates the created matchmakers to increase the productivity of the original matchmaker. In the *node adaptation level*, nodes suppliers or consumers could be promoted to matchmakers with small modifications. When matchmakers are not required in the system, they could return to being consumers or suppliers. In each segment, there is a matchmaker that is in charge of looking for the matching between consumers and suppliers. If the matchmaker cannot find a suitable matching, it sends the request to others matchmakers. The communication and cooperation between matchmakers is fundamental.

Bromuri et al. (2009) present a system that integrates three components: (i) an argumentation framework for decision-making called CHARGO; (ii) a P2P platform that organizes the information using a K-Dimensional tree structure and supports multi-attribute and range queries called PLATON; and (iii) an ontological environment for agents called GOLEM. In this system, requestor agents discover provider agents through the P2P platform PLATON or using multiple and distributed registries that store semantic descriptions. As a result of the discovery process, the requestor agents have a set of services that fulfill high-level requirements. However, these agents are able to use argumentation strategies to internally select the most appropriate services according to a set of preferences. Moreover, once a requestor agent has selected a service provider agent, it starts a negotiation process with the provider. This system covers the whole process of service discovery, including negotiation with the providers. Moreover, it takes advantage of the agents capabilities such as argumentation or negotiation skills.

5.4   Discussion.

In the area of P2P systems two approaches have been presented: super-peers and DHT. With regard to super-peers and considering the *structural dimension*, these approaches offer limited scalability to the system that could be reduced if the number of peers increases. Moreover, tasks of search and resource management rely on the structure of the super-peers in the system. This affects the robustness of the system. There is another problem when several super-peers fail and other peers less qualified must replace them. With regard to the *search dimension*, the flooding algorithm is the most widely used strategy. Algorithms of this type could be applied to different domains easily; however, the traffic generated could overload the system. Another drawback of these systems is that the search is based on keywords; therefore, some results could be missed.

Concerning P2P systems based on DHT and their *structural dimension*, they improve the scalability and robustness through the distribution of search tasks among a set of peers following an specific criteria that varies in the different approaches. Nevertheless, the main disadvantage of these rigid structures is the maintenance of the indexes when the peers enter and leave the system. Updates imply the interchange of messages among peers; therefore, during a period of time the system could be in an inconsistent state due to outdated references. With regard to the *search dimension*, these approaches rely on the fact that peers know the exact key of the resource that they are looking for. These mechanisms are not so effective for locating resources with partial information. Moreover, the accuracy in the search is reduced since the search is based on numeric keys and does not consider semantic information. The main advantage of these algorithms is that, in the majority of the proposals, the search process is bounded to $O(\log(n))$ where n is the number of peers in the network.

With regard to SOE distributed proposals and the *structural dimension*, we can conclude that they offer limited scalability. Some of them distribute the content of the service descriptions in several registries, but the figure of a central entity that coordinates, supervises, and is responsible for the maintenance of the structure still remains. This implies that the search process relies on this central entity and could be a critical point of failure. However, there are proposals that use Semantic Overlay Networks or DHT structures to partially avoid this problem. Considering the *search dimension*, the presented approaches rely on a predefined ontology to describe the categories of the resources and registries. This could be a drawback for the arrival of new services that do not belong to a category that is defined in this ontology. Moreover, the entities in the system know where the central entity or the set of registries are; therefore, the queries are directly sent and the search process does not generate much traffic.

In the MAS approaches described above, the structural and search features are similar to SOE approaches. From the *structural* point of view, there is a set of approaches that makes use of coalitions or adaptive matchmakers to provide more scalability. Considering the *search* in the proposals, the process is very similar to the distributed strategies in P2P and SOE. Agents know a matchmaker to send the query and then this matchmaker contacts other matchmakers if it has not found the required service. The systems are able to adapt the number of matchmakers taking the demand of the system and the traffic into account. However, this implies a coordination effort that increases the traffic in the system. In the majority of the proposals the matchmaking process is based on keywords and does not include semantic information. Therefore, the search process is less accurate.

6   Decentralized Approaches

In decentralized systems, all the entities are considered to be equal and there is an arbitrary topology. There is no central control on how entities should be connected or disconnected when they join or leave the network. Moreover, there is no centralized maintenance of the network structure, although there are systems where nodes are responsible of maintaining its local structure (Ko et al., 2008; Kota et al., 2009). These features provide more flexibility and adaptability. However, the structure of the system cannot provide information that guides the

search. Entities have no global knowledge of the system structure or service organization. The entities need the collaboration of the rest of the system to succeed in the search process. Locating a resource or service efficiently is one of the most important challenges in unstructured networks (Bisnik & Abouzeid, 2005). Algorithms for resource location in these systems can be classified into *blind* (if entities do not consider information during the search), or *informed* (if the entities consider local information from previous searches or provided by their neighbors). The following sections are organized considering blind and informed algorithms. In each section, we analyze the proposals in P2P, SOE, MAS, and CN. Note that SOE and CN are not present in blind algorithms sections. In SOE, the majority of the proposals use informed algorithms that consider information of service descriptions. In CN, the approaches make use of structural information, such as connection degree, or similarity between nodes to guide the search.

6.1 Blind Algorithms: Flooding

Blind algorithms do not consider any information about resource locations. These algorithms could be applied in several domains because they do not require specific domain knowledge. However, their efficiency depends on the structure of the overlay network and the number of available copies of the resources. In order to select the most appropriate neighbors to forward the request, a peer uses the following algorithms: flooding, or random-walks. In this section we describe blind algorithms that use flooding techniques. In the following section the algorithms that use random-walks are described.

6.1.1 Peer-to-Peer.

The original Gnutella implementation is an example of a *flooding broadcast* discovery mechanism [3]. When a peer makes a query, the query is sent to all its neighbors. If the neighbors cannot answer the query, the query is forwarded to their neighbors and so on. If the resource is found, the peer sends a message to the peer who sent the query and they establish a peer-to-peer connection. The query has a time-to-live (TTL) that is decremented each time a peer processes it. When the TTL of the query is equal to 0, the query is not forwarded anymore. The use of TTL associated to the queries is to control the traffic in the network. Since all the queries are forwarded to all the neighbors, the system does not scale well. The workload of the network grows exponentially with a linear increase in the number of peers. If the number of peers in the system increases considerably, it will cause the saturation of the network. This approach has the advantage of flexibility in the processing of queries. Peers can determine how it will process the query and respond accordingly. It is simple to design as well as efficient. However, this type of mechanism is very susceptible to sabotage; malicious peers can send out a large number of queries that produces a significant workload on the network.

An improvement of flooding algorithms in P2P systems is presented by Yang et al. (2002). The authors present the *Iterative deepening* technique which is based on policies. A policy determines the depth level that should be reached in each iteration. Given a policy P = {3, 5, 9}, the source node starts a Breadth-First Search of a depth 3 (TTL=3) and sends the message to all its neighbors. When the message arrives to a node in level 3 (frontier of the search) the query is stopped at all nodes in that level and these nodes send a message to the source node. If the source node finds that the query has been satisfied, then the search process is stopped. Otherwise, the source node starts the next iteration, but now the search starts with the next policy (TTL=5).

6.1.2 Multi-Agent Systems.

Flooding algorithms have also been used in MAS. Ouksel et al. (2004) present a P2P approach that uses a flooding algorithm to locate agents with the needed capability. An agent broadcasts a query to its neighbors and the agent that receives the request either offers its services to the

[3]http://www.stanford.edu/class/cs244b/gnutella_protocol_0.4.pdf

original requester, if they match with the query requirements, or broadcasts the request to its neighbors. As in P2P systems that use flooding strategies, the main drawback of this approach is the overall communication traffic overhead if the services are not replicated in the system.

A variant of the flooding algorithm is proposed by Campo et al. (2002). They propose a push-pull solution for service discovery in pervasive multi-agent systems. In these environments, agents cannot rely on a single agent that is permanently present in the system to act as a central server, and none of the agents that are present at a certain moment are suitable to act as a service facilitator. As an alternative, they propose a combination of push and pull solutions: an agent announces its services only when another agent requests one of its services. In that case, the provider agent broadcasts to all the agents in the system. The main drawback is that agents waiting for an answer to their requests receive many queries that are not related to their interests.

There are other proposals that refine the flooding mechanism in different ways. In particular, Lopes et al. (2008) present an approach based on P2P strategies to facilitate the cooperation of agents. Specifically, the authors present two algorithms for resource location: *Priority-base flooding* (PBF), and *Iterative Branching Depth First Search* (IBDFS) algorithms. The first algorithm establishes a priority between the queries that arrive to a peer in the network. By following this algorithm, a peer manages its workload by establishing a priority order among the queries. A peer prefers to forward a query that comes from a node that is close rather than a query that comes from a node that is far away. Therefore, the priority is inversely proportional to the number of hops of the query. This criteria is based on the idea that queries from far-away nodes have been distributed among the network and have a high probability of being processed by other nodes, while queries from closer nodes have not yet been processed by many nodes. Furthermore, the authors present a second algorithm: the *Iterative Branching Depth First Search*, which introduces the use of an iterative process in the depth first search to increase the coverage of the network. If a node cannot provide an answer, it selects one neighbor to forward the request to. If that neighbor cannot satisfy the request, then this node forwards the query to the rest of its neighbors. This approach increases the branching level iteratively and, therefore, the chances of finding the answer faster. Both algorithms are evaluated over a Semantic Overlay Network that dynamically takes advantages of semantic dependences between peers and their resources. The tests conclude that the IBDFS improves the performance of PBF due to the branching factor, which increases the parallel power of the search. The main drawback of the system is the time required to establish the Semantic Overlay Network.

6.2   Blind Algorithms: Random Walks

Random walks have been presented as an alternative search algorithms to flooding ones in P2P systems. A *random walk* algorithm selects a set of the neighbors 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 be: the TTL has been consumed or the query has been satisfied. This algorithm reduces the number of messages considerably when compared to flooding algorithms. In the worst case, it produces k*TTL messages, where k is the number of walkers. The disadvantage of these types of algorithms is that the percentage of success varies depending on the network topology (Gkantsidis et al., 2006), the popularity of the resource, the number of walkers, and the TTL (Bisnik & Abouzeid, 2005).

6.2.1   Peer-to-Peer.

In P2P some experiments have been done to show that random walks offer better results in searches than flooding techniques (Lv et al., 2002). The results of the experiments conclude that the adaptability in termination conditions and granularity in coverage of the search spaces are the attributes that make random walks more suitable than flooding. Bisnik et al. (2005) propose an algorithm to adapt these parameters. Moreover, the authors integrate a feedback algorithm for maintaining an estimation of the popularity of the resources. In general, random-walk algorithms

have been improved considering local information, therefore, they could be considered as a hybrid proposals of random and informed techniques. For example, in (Zhong, 2006) the authors present an algorithm based on the content popularity of neighbors to improve the performance of random-walk algorithms. In this method, at each step of the walk, the next hop is selected form the neighbors of the current peer with probabilities biased towards their content popularity. The content popularity is calculated as the number of queries satisfied divided by the number of the total queries received. Another relevant work is GIA (Chawathe et al., 2003). In this work the authors propose an algorithm that refines random-walks. Each peer in the network forwards tokens (walkers) depending on its capacity. The distribution of the tokens among its neighbors is not equal. This distribution depends on the neighbor capacity. Moreover, GIA has a flow-control and adaptation mechanism to converge to states where peers receive and send the same number of tokens. Random algorithms have also been used in adaptation mechanisms. Cholvi et al. (2007) use a random algorithm in an adaptation mechanism. In this mechanism each peer i chooses a set of peers C_i. This set is selected using random-walks with a bounding TTL. Each peer i reconnects their links to peers situated in C_i by analyzing their weights. The weight of each peer is calculated considering its capacity and the average time spent by a search query at the peer.

6.2.2   Multi-Agent Systems.

There are other algorithms that integrate several blind algorithms. An example is presented by Dimakopoulos et al. (2003). Basically, each agent has a local registry with k resources and the agent that offers them. Each agent is connected to a set of neighbors. The system is modeled as a directed graph. When an agent receives a query about a resource and it does not have it, the agent redirects the query to its neighbors following a flooding-based algorithm. The authors present three algorithms: (i) pure flooding algorithm; (ii) teaming algorithm (the agent propagates the message to only a set of its neighbors with a certain probability); and (iii) random paths (the agent limits the number of neighbors to a set in the first propagation and then each neighbor continues the propagation to only one of its neighbors). Furthermore, the proposal considers the problem of updates when agents move or leave the system. Two mechanisms are proposed: inverted cache, and update flooding. The first one implies that each agent should maintain a list with all the agents that have references to it. This could create maintenance problems if the documents of agents are spread in many registries. The second mechanism is based on flooding update messages until a TTL limit; therefore, some entries could be obsolete due to the TTL limit.

6.3   Informed Algorithms

Informed algorithms use local information to forward the request to the most promising neighbor (the closest neighbor to the target) and to reduce the network overhead. The entities store information about their direct neighbors or about statistics of previous searches in local registries.

6.3.1   Peer-to-Peer.

An example of this type of algorithms is presented by Crespo et al. (2002). 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 the following information about each neighbor: the number of documents along the path and the number of documents on each topic of interest. The storage space per neighbor can be adjusted increasing or decreasing the level of summarization of the index. 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 select or flooding the network. The set of neighbors to forward the query to are selected according to their *goodness* for the query. The notion of *goodness* reflects the number of documents of a certain category in nearby nodes. This reduces the number of messages that forward the query to the nodes that have a high potential of obtaining results. The problem could be the information maintenance. The

number of messages 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. For instance, node A knows that the best neighbor to redirect a query of category 'x' is C because, through that route, there are 200 documents of that category. However, if all these documents are from a single node in that route and this node leaves the system, node A forwards the request through the wrong route until the update message arrives. Moreover, the precision of the method depends on the number of categories that are considered in the search process. For instance, if you are looking for a document of a certain category that is not considered in the node, you do not have any information about what the best route to follow is.

There are other approaches that use information obtained from previous experiences to guide the search in the network. This information is used to obtain statistics that could be considered in heuristics. The *Directed Breath First* search is a technique presented by Yang et al. (2002) that forwards the queries only to a subset of neighbors. These statistics can be very simple. They present several heuristics: (i) select the neighbor with the highest success in previous searches; (ii) select the neighbor that finds the shortest paths (this means that the neighbor is close to useful nodes); (iii) select the most stable neighbor (the neighbor that has been selected the greatest number of times since it was connected to the network); (iv) select the neighbor with the shortest message queue. The main drawback of this approach is that, in order to follow these heuristics, the network needs a training period to obtain significant information. Moreover, some of the heuristics could overload some peers and leave other potential peers without traffic.

Adaptive Probabilistic Search is a proposal presented by Tsoumakos et al. (2003). It is based on the combination of the k-random walk algorithm and a 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 has to be selected the next time. The algorithm works as follows. Initially the requester peer forwards the query to k of its neighbors. In the next steps, the peers only forward the request to one of their neighbors. The selection of this neighbor is made by using the probabilities given by the index values. The index values are updated by considering the information obtained in the search process. During the search, the peers add their identifiers to the request message. There are two versions of this algorithm: *optimistic* and *pessimistic*. The *optimistic* version always increases the value of the index, and it only decreases the index in the case that the search fails. In that case, the algorithm sends a message to all the nodes whose identifier is in the query (the reverse path) to update the indexes. The *pessimistic* version always decreases the value of the index and, if the search is successful, then the indexes are updated. Moreover, the authors present two extensions to improve the performance of the algorithm: (i) swapping the strategy considering the probability of success to minimize the number of backward messages; and (ii) the consideration of the distance between a peer and the object in order to update the indexes. The last improvement is only applied to the pessimistic version. To improve the performance of the algorithm, a learning period is needed to obtain more precise values for indexes and more efficient searches. Something similar happens in the Intelligent Search Mechanism proposed by Kalogeraki et al. (2002). This approach allows peers to identify links that are likely to have relevant information. To establish this metric, the peer collects the queries that peers reply to. The drawback is that the algorithm needs a period of time to collect the information that improves the search. Moreover, if the links between peers change frequently, this information becomes useless.

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. (2006) 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 pheromone quantity. 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 one selects the best neighbor considering the quantity of pheromone. The second one 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 according to the 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.

Upadrashta et al. (2005) 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 friend list and learns about the interests to obtain more relevant sources faster and with less traffic. The list reflects similarity of interests (semantic categories) between peers. For each neighbor in the list, it associates a category and a value that reflects the strength of the relationship between the peers that are related to the category. The main contribution is a "semantic-social" routing approach. When a peer sends a request, it decides which of its neighbors has the same category of the query. If the number of selected neighbors is high, then the algorithm selects only the best peer taking into account the strength.

6.3.2   Service-Oriented Environments.

In decentralized SOE, semantics have also been used as an instrument to provide information in the search process. Basters et al. (2006) present an algorithm for decentralized service retrieval in unstructured networks. The algorithm is based on semantic information. Each agent in the network has a local training set that contains previous queries and their results. When an agent receives a query about a service, it first looks up semantic similar services using its local matchmaker. The agent keeps the services returned by the matchmaker and redirects the query to the most promising neighbor. The selection of the most promising neighbor 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. Each agent that receives the query repeats this process and redirects its selected services to the agent who forwarded the query. By this method, the result set that contains all the relevant services for the query is generated while it is propagated back to the initial agent. An agent can reject a query if it has forwarded the query previously, if the TTL of the query is reached, or if the risk of forwarding the query is maximal for each of its neighbors. 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 initially will use flooding algorithms that could overload the system.

Bianchini et al. (2009) present the SERVANT architecture for providing a decentralized service registry (DSR) that facilitates the service-based semantic search. The DSR is organized in three layers: *logical* (where connections are defined as in a P2P network), *semantic* (where the semantic overlay network is maintained), and *mapping* (where mappings between similar services are defined to support interoperability). In the semantic layer, each peer maintains a semantic service description about the services it offers. Similar peers (peers that offer similar services) are connected through semantic links. The authors distinguish between two types of semantic links: functional similarity links, which relate similar semantic service descriptions, and coupling similarity links, which semantically relate the outputs from one service to the inputs of other service. Each query has a TTL associated that is decremented each time a peer forwards it. When a service request arrives to a peer, its Semantic Search Assistant component (SSA) checks if it has the required service. Specifically, the SSA contacts to a local service matchmaker that has

a hybrid matchmaking model to compare semantic service descriptions based on peer ontologies and terminological knowledge contained in a thesaurus. If a service that matches the request is found, the peer sends the service to the peer that originally sent the request. If the query TTL is not 0, the forwarding process continues based on its semantic neighbors. 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 avoid the formation of isolated clusters in the semantic layer. The authors propose two policies to stop the search process although the TTL associated to the query is not 0. One policy determines that the search process ends when services that satisfy the request are found. The other policy determines that although services that satisfy the request are found, the search continues looking for neighbors that offer services until services with better non-functional properties are found. Once the user receives the retrieved services and selects one, the services linked to the selected one through coupling similarity links are proposed to the user. The drawback of this approach is that the peers are organized in clusters of similar services, therefore, it is probably that a peer cannot find services that are semantically different to their 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.

Ding et al. (2010) present a decentralized model for service discovery. The structure of the model is based on links established when a new node joins the system. The new node broadcasts its service advertisement to all the existing nodes. Each node in the system analyzes the service offered by the new node and determines if it could be one of its potential service providers. Whether the new node is considered as a potential provider, the node stores its information in a local registry. Otherwise, the advertisement is ignored. Each node in the system has a service matchmaker and a local service registry. The matchmaker is composed by a syntactic and semantic matchmaker. The syntactic matchmaker analyzes the text description and parameters of service operations using information retrieval techniques. The semantic matchmaker considers the information of the OWL-S descriptions. The main drawback of this approach is the use of broadcast to establish the potential service providers of the nodes. This fact affects to the scalability of the system. Moreover, it is not clear how nodes deal with the search process when the provider is not in its local registry.

6.3.3 Multi-Agent Systems.

In MAS algorithms that simulate the behaviors of ants have also been used for service location in unstructured systems. Babaoglu et al. (2002) present a middleware that is based on the MAS paradigm. They consider that ant colonies are natural instances of MAS, which are capable of solving complex problems in a completely decentralized way. Anthill is a network that is composed of nests. Each nest implements a hash function that associates a key to a keyword and a document. Nests can send requests generating one or more ants (autonomous agents) that navigate the network trying to satisfy the request. Ants communicate with other ants indirectly through the information stored in the nests. This indirect communication is called *stigmergy*.

Zhang et al. (2004a) propose a completely decentralized MAS without mediators. The system is based on a P2P structure where each agent has the following local information: a *collection*, a *collection descriptor*, *search engine*, an *agent view* and a *control center*. The *collection* is the set of documents that the agent has available to share with the rest of the agents in the system. The *collection descriptor* characterizes the distribution of the vocabulary in the collection. The *control center* decides how the distributed search is going to be performed. The *agent view* structure contains information about other agents in the system. Initially, agents are connected randomly. The authors propose an agent-view reorganization algorithm (AVRA) based on the initial agent-view. The objective is for each agent to contain agents with similar documents in its agent-view. In order to avoid isolated clusters of agents, the algorithm establishes a percentage of similar and dissimilar agents that should be in the agent-view. 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 a 'good starting agent' if its similarity with respect to the query is over a threshold. Whether the initial agent is a 'good starting agent', the algorithm performs as KNN. Otherwise, agent A selects the most similar neighbor B to the target, and a message with the similarity information is sent to B. This process is repeated n times. Taking the highest similarity value into account, the last agent will choose the agent to restart the search using the KNN algorithm. The main disadvantage of this approach is the high communication cost required to organize agents into communities.

6.3.4 Complex Networks.
Approaches from the area of Complex Networks were not mentioned in previous sections. This fact is due to they are only applicable to decentralized systems. Complex Networks have been considered to model decentralized systems where the search for resources or services is carried out considering local information (Boccaletti et al., 2006). The most common network models present in the area of Complex Networks are the *small-world* and *scale-free* models. In the following paragraphs, we describe these two models and the different proposals for dealing with the construction of these models and the search strategies.

Small-World networks are characterized by two main features: they have a high clustering coefficient, and small average path. A high clustering coefficient reflects that the neighbors of a node are also neighbors with each other (connection degree between the members of a neighborhood). These networks also usually have small average path, which is related to the diameter of the network. The diameter in small-world networks is exponentially smaller than the size of the network and can be bounded to $\log(n)$, where n is the number of nodes. There are several mechanisms that have been used to generate small-world structures. The main difference among them is the initial structure that is considered:

- Lattice-Ring models: have a regular structure (ring, lattice or grid) that is modified by randomly rewiring some existent links or by adding new ones.
- Hierarchical models: are more realistic than regular models and, instead of using a regular structure (lattice or ring), they use hierarchical structures that reflect the organizational structure of a certain domain.
- Grid and Hierarchical models: take features from both models in order to build the network.

Watts and Strogatz (1998) propose a method for constructing a small world network that starts with a regular graph with n nodes that are located in a ring and that has k neighbors for each node. Then, some links (which are randomly selected) are rewired with a probability ρ, $0 \leq \rho \leq 1$. The authors conclude that, with intermediate values of ρ, a network with high degree of clustering and small path length between nodes can be obtained. Another method for constructing a network with small-world properties was proposed by Kleinberg in (2000). The network is based on a 2-dimensional regular lattice of $n \times n$ dimensions where all the nodes of the lattice are connected to the closest neighbors (short connections). Then, long random connections are established with a probability p_{ij} that is inversely proportional to the square of the distance between them:

$$p_{ij} \propto r_{ij}^{-\gamma} \tag{1}$$

where r is the Manhattan distance (number of links between nodes) in the lattice between nodes i and j. Kleinberg concludes that when γ is equal to the dimension of the lattice, the network is searchable and a greedy algorithm can be used to navigate the network and find the target with paths bounded by $\log(n)^2$.

The main problem with these models of networks is that they are based on a lattice structure and does not reflect how the real complex networks are created. A more natural model could be

created if occupational aspects were considered to establish links. In these models, the distance between nodes is calculated based on a hierarchy.

Watts et al. (2002) present a *hierarchical network model* that is based on social structures (hierarchical structures). A node can be member of several social structures. The authors define the similarity between two individuals as the height of their lowest common ancestor level in the hierarchy. Based on that distance, they define the probability that an individual i establishes a link with an individual j (which is randomly chosen) that is located at distance x from i is:

$$p(x) = c^{-\alpha x}, \qquad\qquad\qquad (2)$$

where c is a normalizing constant and α is the homophily factor (tendency to be associated with similar individuals). The authors propose a greedy search algorithm. They consider that each individual has a vector with social dimensions (H) and define a social distance between two individuals based on those vectors. An individual selects one of its neighbors considering which one is the closest to the target individual (minimal social distance) to forward the message. They also determine the most suitable number of social dimensions ($H=2$) and the value of the homophily parameter ($\alpha=1$) in order to obtain an average path length that is similar to Milgram's experiment (Travers & Milgram, 1969).

A similar proposal is presented by Kleinberg. He presents a *hierarchical model with exponent β* (Kleinberg, 2006). The hierarchy is modeled as a b-ary tree having n leaves. The distance between two leaves u and v is defined as the lowest common predecessor $h(u,v)$ in the tree. Based on that, Kleinberg defines a network where, for a node v with degree k, the probability to establish a link with a node w is proportional to $b^{\beta h(v,w)}$. Moreover, Kleinberg determines that a decentralized search algorithm with poly-logarithmic time is possible.

Adamic et al. (2005) test different search strategies in an e-mail network with small-world properties. Using different criteria, each strategy selects the next contact to forward the message. The criteria are: (i) the degree of the contact; (ii) if the contact is close to the target in the organizational hierarchy; or (iii) if the contact is physically close to the target. The first method does not perform well in the small-world network since there is no a significant difference between the node degrees. The second strategy is based on the work of Watts et al. (2002). The results confirm that the strategy of using the organizational hierarchy works well in the e-mail network, and the relationship between separation in the hierarchy and probability of correspondence fits well with the model presented. The last strategy uses the physical position of the individuals, which is similar to Milgram's experiment (Travers & Milgram, 1969). In the experiment of Adamic et al., the physical position is the floor and the office. The results show that the probability of two individuals having a link between them follows the relation $1/r$ instead of the relation $1/r^2$ presented by Kleinberg. The reason for this difference is the limiting geometry of the building.

Another model to consider is the combination of a grid model that is based on distances and hierarchical model. This idea has been proposed by Kleinberg in (2001) where he defines a new model based on a group structure. He said that people who belong to the same small group have more probabilities of being connected. For instance, in a grid model, groups can be found (subgrids) where nodes are closer. In hierarchies, groups can be found in subtrees where two nodes are inside. The model is built based on the distance between two nodes v and w, which is defined as the size of the smallest group that contains both nodes. A link between two nodes v and w is established with a probability that is inversely proportional to the size of the smallest group to which both nodes belong ($g(v, w)^\gamma$). An efficient decentralized algorithm with polylogarithmic delivery time is possible with $\gamma = 1$ and out-degree $k = c\, log^2 n$.

In addition to the models proposed by Kleinberg and Watts, there are other proposals for constructing small-world graphs with O(log n) as the expected diameter (Nguyen & Martel, 2005; Slivkins, 2005). Nguyen and Martel (2005) present a general framework for the construction of small-world networks. Simsek and Jensen (2005) present a new model to generate power-law networks and small-world networks. These networks are built considering two parameters:

homophily and degree disparity. Homophily is a social concept that represents that individuals have more probability of establishing links with similar individuals than with dissimilar ones. Degree reflects that some people have more neighbors than others and may act as a connexion between different social circles. Moreover, the authors propose a new algorithm that confronts the task of searching for a target node in a large network with local information. The proposed algorithm is called Expected-Value Navigation, and homophily and degree parameters are used to guide the search. The main advantage of this algorithm is that if the network shows no homophily, it can reduce the search by only considering the degree. On the other hand, if the degree information is not available, or if all nodes have the same degree, the algorithm searches by only considering the similarity between nodes.

The other commonly used model in the area of Complex Networks is the *Scale-Free* model, which is characterized by its connectivity distributions, which are in a power-law form that is independent of the network scale (Wang & Chen, 2003). The degree of connection of most of the nodes is low, while there are a few nodes that present a high degree. This feature is reflected through several functions that indicate the probability that a randomly selected node has exactly k edges, in other words, how the node degrees are spread over the network. For a number of networks, the degree distribution could be described by the following function:

$$p_k \sim \quad k^{-\alpha} \tag{3}$$

where k is the average degree of the network, and α indicates the rate of decay. Barabasi and Albert (1999) present a mathematical model to create a network with power-law characteristics. The method for creating the network is performed as follows: when a new node arrives to the network, it tends to establish a connection with the highest connected node rather than nodes that have a low degree of connection. The probability for a node that is already in the network to receive a new link is $p(k_i) = c\,k_i$, where c is a normalizing constant and k_i is the degree of the node i.

Thadakamalla et al. (2007) present networks that are called spatial scale-free networks. In these networks, nodes are situated in a n-dimensional space and are connected based on node degree and geographical location. The authors investigate the performance of several algorithms in a U.S. airline network. Some of these algorithms take advantage of both, distance and degree, and the others only consider one of these characteristics. In general, the algorithms that use both features to guide the search process have a good performance and get similar results compared to algorithms that use the global information of the network.

Dell'Amico (2006) presents a social network model based on preferences that considers the degree of connection of nodes and distance. Therefore, the probability of a node i to select a neighbor j with degree k_j and d_{ij} the distance between i and j:

$$\Pi(k_j, d_{ij}) \sim \quad \frac{(k_i)^{\alpha}}{(d_{ij})^{\sigma}} \qquad \alpha \geq \sigma \geq 0 \tag{4}$$

The parameters α and σ represent the influence of degree and distance, respectively, for preferential attachment. The author presents several experiments to evaluate the influence of these two parameters in the structure of the network. The main conclusion is that the best network properties (high clustering, scale-free distribution, and low diameter) appear when $\frac{\sigma}{\alpha} = 1$.

Xiao et al. (2006) present an algorithm to search scale-free networks based on a simple method presented in (Adamic et al., 2001). During the search, each node that receives a request checks the information that the neighbors at one-hop, two-hops, and three-hops can provide. If the target belongs to the neighborhood, then the search is finished. Otherwise, if among their immediate neighbors there are some nodes that have not yet been visited, the request is forwarded to the node with the highest degree. In some situations it is difficult to obtain the three-hop information. For instance, a highly connected node could have too many nodes within three-hop distance from itself. For those situations, the authors propose a partial,

three-hop information-based searching method. In this method, if the node needs to query the third-hop neighbors, it decides to select only a few adjacent nodes. They evaluate the methods considering the information of one-hop, two-hops, and three-hops. The simulations show that having more local information (three-hops) helps to improve the efficiency of the search process. The modification of the method with the partial three-hop information does not improve the performance of the case with complete three-hop information. In order to reduce the path length obtained with partial three-hop information, the authors propose a refinement method that adopts a simple greedy strategy to take 'shortcuts' of the route.

The structures and algorithms used in Complex Networks have also been included in search proposals in P2P systems, SOE, and MAS. An example of the inclusion of Complex Networks in P2P is Symphony (Manku et al., 2003). Symphony is a distributed hash protocol that is inspired by Kleinberg's Small world construction. Moreover, the small-world structure has also been used to improve Freenet (Zhang et al., 2004b). In MAS, a discovery mechanism that is inspired in small-world structures is presented by Moore and Suda (2002). By using keyword similarity, historical information, and clustering each agent determines which relations should be chosen. To forward the query, similarity is calculated as the ratio of keywords that an agent and its partner of relation have in common. Agents with similar keywords will be located near each other in the network. Furthermore, it could be possible to create new sub-clusters inside the clusters where the second keyword is shared among the members. Moreover, in order to connect clusters, agents establish a set of random connections. They try to create a small-world with these long connections. In this proposal, the historical information associated to the relations between agents is considered. This historical information is the ratio of the number of successful queries and all the forwarded queries through the relation. This is used to consider the utility of the relation in order to forward future queries. Moreover, there are other approaches in MAS where Complex Networks have also been used to find the optimal path between two individuals on small-world and scale-free networks by taking into account trust values (L et al., 2010).

6.3.5   Discussion.

This discussion is focused on the blind and informed algorithms used by the approaches from the areas of P2P, SOE, MAS and CN. With regard to the *structural dimension*, in approaches based on informed algorithms, the majority of the proposals are scalable since they use information from previous searches or from neighbors in the search process. Moreover, some CN offer high scalability due to their structure guarantees that greedy strategies could be use and the search can be bounded to $O(logn)$. However, in flooding approaches, the scalability is limited since the algorithms generate too much traffic if the resources are not replicated or the TTL is high. Since all the information is distributed among all the entities, the system robustness is not prone to failures, except to intentional ones that could affect to scale-free structures more seriously. The search does not depend on the structure. However, there are approaches where the links between entities are not established randomly. The information to establish the links is also used to guide the search; therefore, there is a small dependence between the structure and the search process. Moreover, with regard to the *search dimension*, the majority of the approaches that use blind algorithms could be applied in different scenarios since the search is domain-independent. With regard to approaches that use informed algorithms, the search process relies on specific information that may not be available in all the scenarios. The recall of the search results using blind algorithms is very high, because these algorithms cover a great part of the network during the search. However, semantic information is not considered and this reduces its precision (accuracy). In informed algorithms, the accuracy is reduced since the search is based on the majority of proposals in keywords and the area covered by the algorithm is smaller.

Area/Dim	Structure		Search Strategies	Structural Dimension			Search Dimension					
				Scalability	Structural	Robustness	Adaptability	Accuracy	Traffic	Semantic	Search cost	Knowledge
Service Oriented Environments	Decentralized		[Bianchini09] Search based on semantic similarity between service descriptions + Random.	+	-	+	+	~	~	+		Local (neighbors in different layers +semantic service descriptions +Syntactic information)
			[Ding10] Local semantic and syntactic matchmaking.	-	+	+	+	+	+	+		Local (registry with potential providers of semantic services and service consumers)
			[Basters06] Flooding + probabilistic algorithm based on previous experiences stored in a local registry that consider semantic information and communication cost.	~	-	+	+	+	~	+		Local (neighbors+number of messages and semantic service descriptions)
	Distributed	Federation	[Sivashanmugam04] Search based on semantic federation domains.	~	+	~	-	~	-	~		Global (semantic inf of registries + services)
		Grid	[Skoutas08] Search based on semantic intervals.	+	+	+	+	+	~	+		Local (semantic inf about services)
		DHT	[Pirro10, He08] Search based on semantic similarity between query and services stored in peers or , DHT Chord protocol, or the combination of both.	~	~	+	+	+	~	+		Local (Finger table + Semantic Annotation Table)
		Tree	[Cao10] Search based on semantic matching in a tree of registries.	-	+	-	+	+	~	+		Global (Semantic information about services)
		Cluster	[Perryea06] search based on semantic matching in a directed graph.	-	+	-	+	+	-	+		Global (Semantic information about services)
	Centralized		[UDDI02] Direct request to the registry. Syntactic similarity between service descriptions.	-	+	-	+	+	-	-		Global (Syntactic information about services)
			[Srinivasan04] Direct request to the registry. Semantic similarity between service descriptions.									Global (semantic service descriptions)
		Hypergraphs	[Brogi06] Search based on semantic relations in a hypergraph.	-	+	-	+	+	-	+		Global(semantic service descriptions)
			[Prabhu07] Breadth First Search algorithm forward and backward simultaneously until a collision is found.									Global(semantic service descriptions)
			[Bailey06] Search based on transversals in a hypergraph.									Global(semantic service descriptions)
			[Mokhtar06] Semantic service search reduced to a numeric comparison of intervals.									Global(semantic service descriptions)
P2P Networks	Decentralized	Informed	[Crespo02] Routing indexes based on number of documents of certain category in nearby nodes.	~	+	~	+	~	~	-		Local – Global (topics of interest and documents along paths)
			[Yang02] Directed breadth first where the queries are forwarded to a subset of neighbors.	+	~	+	+	~	~	-		Statistics about previous searches
			[Tsoumakos03] Adaptive probabilistic search based on the results of previous searches.	+	-	+	~	~	~	-		Local (neighbors and probabilities)
			[Michlmayr06] Ant algorithm based on keyword similarity and quantity of pheromone.									Local (keywords, documents)
		Blind	[Zhong06] random-walks biased by content popularity of the nodes.	+	-	+	+	~	+	-		Local (queries received and attended)
			[Yang02] iterative deepening: breadth-first search of a limited depth established by a policy.	~	-	+	+	~	+	-		Local (Files)
			[gkantsidis05, Bisnik05] k-random walk									Local
	Centralized		[Napster06] direct search request based on keywords.	-	+	-	+	~	-	-	O(1)	Global (Files – peers)
		Super-peers	[Kazaa04, Morpheus, Gnutella01] broadcast among super-peers. Search based on keywords.	-	+	~	+	~	+	-	O(N)	Global (Files – peers, and references to other super-peers)
	Distributed Hierarchical	Structure based on DHT / mesh	Tapestry [Lua05] (mesh network) search based on the degree of match between suffix digits in different levels.								O(log N)	Local (Files and references to other Super-peers)
			Pastry [Rowstron01] (mesh network) algorithm based on a routing table and prefix-based comparisons between IP addresses.								O(log N)	Local (leaf set, routing table, Neighborhood set)
		circular	Chord [Stoica01] (circular node Id space) finger table alg. Based on distance between keys (node id, data key).	~	+	~	-	~	-	-	O(log N)	Local (Finger Table)
		D-space	CAN (multidimensional id coordinates space): greedy algorithm that forwards the query to the closest coordinates in the space.								$O(d. N^{1/d})$	Local (a set of neighbors in the coordinate space)
		tree	Kademlia [Maymounkov02] (binary tree): algorithm based on XOR distance between identifiers of nodes.								O(log N)+c	Local (Set of tuples of near peers)
		ring	[Yu04] (Chord+semantics): 2 levels of rings. Algorithm based on semantic indexes.	~	+	~	-	~	-	+	O(log N)	Local (Finger Table)
			Symphony [Manku03] (small-world): algorithm based on distances.	+	+	+	-	~	-	-	O(1/k logN)	Local (near peers and long distance links)
MAS	Decentralized	Informed	[Zhang04] K Nearest Neighbors + Gradient Search Scheme.	+	-	+	~	~	+	-		Local (Vocabulary, documents, and agents With similar docs)
			[Dimakopoulos03] flooding + random algorithms.	~	~	+	~	~	+	-		Local (Agents, resources)
			[Babaoglu02] ant algorithm.	+	-	+	+	~	~	-		Local (keywords, Urls)
		Blind	[Campo02] push & pull+flooding.	-	-	+	~	~	+	-		Local (service descriptions)
			[Ouksel04] flooding algorithm.	~	~	+	~	~	+	-		Local (Ag, resource)
			[Lopes08] priority-base flooding and iterative branching depth first search.	~	-	+	+	+	+	+		Local (Files)
	Distributed		[Ogston01] random search and grouping procedure.	~	~	+	~	~	~	-		Local (Tasks)
			[Jha98] search in distributed matchmakers.	~	+	~	+	~	-	-		Local (Services and other matchmakers)
			[Sidgel05] search in distributed matchmakers.	+	+	~	+	~	-	-		Local (Services and other matchmakers)
			[Bromuri09] semantic search in a K-dimensional tree. The search process includes argumentation and negotiation processes.	~	+	+	~	~	-	+	O(log N)	Local (multiple attributes about services)
	Centralized		[VazquezSalceda10] semantic search in a single matchmaker agent.	-	+	-	~	+	-	+		Global (semantic service descrip.)
			[Argente11] semantic search in a single matchmaker agent.	-	+	-	~	+	-	+		Global (semantic service descrip.)
			[Caceres06] [Klusch06][Fernandez07] semantic search in a single matchmaker agent or in a sequential combination of matchmakers.	-	+	-	+	+	-	+		Global (semantic/syntactic service descrip.)
Complex Networks	Decentralized – Loosely Structured	Scale-Free & Small-World	[Barabasi99] [Adamic01] algorithm based on hop distance between nodes and connection degree.	~	~	~	+	~	~	-		Local (degree, number of hops)
			[Xiao06] algorithm based on three-hop information and degree.									Local (degree and three-hop neighbors)
			[Thadakamalla07] algorithm based on connection degree and distance.									Local (node degree and geographical situation)
			[Dell06] algorithm based on connection degree and distance between nodes.	+	-	+	-	~	-	-		Local (degree and distance)
			[Watts02] algorithm based on social distance between two nodes.	+	-	+	-	~	-	-		Local (Vector of social dimensions)
		Social groups	[Kleinberg06] [Kleinberg01] [Adamic05] alg. based on the distance of nodes in a hierarchy.	+	~	+					Poly-Logarithmic	Local (social distance) Global (Hierarchy)
		ring	[Watts98]	+	~	+		~	-	-		Local
		lattice	[Kleinberg00] greedy algorithm based on Manhattan distance.	+	~	+		~	-	-	$log(n)^2$	Local (Manhattan distance (links) Between neighbors)
			[SimsekJ05] search algorithm based on connection degree and similarity between nodes.	+	~	+	+	~	~	-		Local (similarity and degree)

Figure 1 Overview of search approaches in distributed environments. The table groups the proposals by areas. An approach is analyzed in each row of the table. The first column describes the structure. The second column contains the first author of the article where the proposal is described and the year of the publication ([authorYear]). Moreover, the search strategy is described briefly. The three following columns describe the criteria of the structural dimension. The rest of the columns describes the search dimension.

7   Conclusions and Final Remarks

In this article, an analysis of different approaches for dealing with the search challenge in distributed environments has been presented. These approaches are from different areas that tackle the same problem in different scenarios. First, we described these four areas: P2P, SOE, MAS, and Complex Networks in order to put the proposals in context. Next, we introduced the dimensions that were considered in the analysis of different works. The works have been grouped into three main sets taking into account the underlying structure of the systems: structured, distributed, and unstructured.

The four areas that we have considered present different scenarios where search strategies are applied. For instance, P2P systems are populated by reactive nodes that offer and request resources such as audio, video, or text files. The aim of these systems is to facilitate resource sharing among peers. In the case of SOE, systems are populated by services that are considered to be basic building blocks. Services are platform-independent and facilitate interoperability. The aim of these systems is to facilitate the reusability and adaptability of existing services. To deal with this task, the service discovery process plays a critical role. In the case of MAS, systems are populated by agents. Agents have social and pro-active capabilities that make them flexible and adaptable to changes in the system. These agents have resources and capabilities that can be offered to others. Under certain circumstances, agents have to deal with complex goals that require the collaboration of other agents with certain capabilities. For this reason, the task of agent location becomes important in the context of MAS. Moreover, the area of Complex Networks has proposed models for distributed environments that allow greedy strategies to be used and also to obtain short paths in the search process.

The analysis shows that, from the *structural* point of view, the areas of P2P, SOE, and MAS follow similar structures. In systems where the number of entities is limited, centralized approaches are responsible for resource location (Gummadi et al., 2002; Brogi et al., 2006; Klusch et al., 2006; Prabhu, 2007; Argente et al., 2011). These approaches generate less traffic, are more efficient and the results are more accurate since all the information is considered. In order to avoid bottlenecks and to provide robustness and scalability, if the system is larger, distributed approaches such as: super-peers (Liang et al., 2005) or DHT (Stoica et al., 2001; Rowstron & Druschel, 2001; Ratnasamy et al., 2001; Mokhtar et al., 2006; Maymounkov & Mazieres, 2002) in P2P; federations of registries (Sivashanmugam et al., 2004) in SOE; coalitions of agents (Ogston & Vassiliadis, 2001b,a) or distributed middle-agents (Mullender & Vitanyi, 1988; Sigdel et al., 2005) in MAS, have been proposed. Moreover, there are some proposals that integrate structures from different areas such as DHT and Semantic Overlay Networks based on semantic service descriptions (Manku et al., 2003; He et al., 2008; Pirr and et al., 2010). Finally, if the systems are highly dynamic, with a large number of heterogeneous entities that only have partial knowledge, the search process relies on each entity. In these approaches, there are two types of search strategies: blind or informed. Blind strategies generate more traffic since they do not rely on domain-specific information. Informed strategies use statistical information from previous searches in order to guide the search. The main problem with the informed strategies is that they need a training period in order to have enough information to guide the search (Yang & Garcia-Molina, 2002; Basters & Klusch, 2006). For this reason, there are decentralized approaches that try to facilitate the search process following certain criteria to establish links between entities. An example of this is the use of Semantic Overlay Networks (Lopes & Botelho, 2008; Bianchini et al., 2009). Moreover, Complex Network provide models where short paths can be found following greedy search strategies (Simsek & Jensen, 2005; Adamic & Adar, 2005; Kleinberg, 2006). These models are considered in proposals in P2P, SOE and MAS to organize and improve the resource location in decentralized and loosely structured systems (Manku et al., 2003; L et al., 2010; Moore & Suda, 2002).

With regard to the *search dimension*, in general, the four environments present similar solutions for dealing with the search for resources or services. In P2P systems, the majority of the proposals

are oriented to the location of resources such as files, and the search process is based on keys (Stoica et al., 2001; Rowstron & Druschel, 2001; Ratnasamy et al., 2001; Maymounkov & Mazieres, 2002). SOE approaches use similar structures to deal with the service discovery. However, this area introduces an important improvement: the inclusion of semantics in the service descriptions and in the search process. Semantics enhances the discovery process by providing more flexibility and precision (Mokhtar et al., 2006; Brogi et al., 2006; Bailey, 2006; Prabhu, 2007). Moreover, semantics has also been introduced as a criterion to establish links between different entities (Bianchini et al., 2009; Pirr and et al., 2010; Val et al., 2011). In the case of MAS, the agents that populate the system offer their capabilities through services. For this reason, some of the works presented in SOE could be directly applied to solve the problem of service discovery in MAS. Moreover, agent features such as organizational roles, trust, or argumentation and negotiation capabilities have been included to improve the selection process or guide the search process (Fernández et al., 2008; Bromuri et al., 2009; L et al., 2010).

7.1 Final Remarks

Distributed systems are populated by a large number of heterogeneous entities that act as clients and providers. Therefore, there are a great number of different types of services and resources that could be considered during the search process. Moreover, the entities join and leave the system dynamically which makes the management of updated information about resources difficult. In distributed systems, in most situations, entities only have a partial view of the system. Taking into account these features, the systems should provide search mechanisms that: (i) provide scalability and robustness when entities that participate on them change; (ii) locate a required resource only considering local information and do not require flooding strategies; (iii) adapt themselves as the environmental conditions changes (i.e., user demand, business requirements); (iv) manage different types of information (i.e., syntactic and semantic data); (v) promote the cooperation in systems where self-interest or malicious entities are present in order to improve the search results; (vi) integrate functional and non-functional information in the selection process. In the following paragraphs we describe each feature:

Robustness and Scalability. In many proposals, systems are based on rigid hierarchical structures where the content is placed on a set of entities according to hash functions. Moreover, these entities are also responsible for the search process. The most appropriate systems for providing robustness and scalability in distributed environments should be decentralized, where all the entities are equal and each one manages its own information and carries out the search process. Complex Networks provide decentralized and loosely structured models. In these models, links that follow more flexible criteria than in structured systems are established. In some approaches, semantics has been introduced to establish these links, this provides flexible self-organization and improves the query routing and search performance, facilitating the adaptation to environmental conditions as well as the search process. Moreover, there is a set of Complex Network models that have a structure where greedy algorithms can locate the target resource in short paths.

Local knowledge. In distributed systems, in most situations, entities only have a partial view of the system. Therefore, the search process should use blind strategies or informed strategies that rely on local knowledge. Blind strategies such as flooding are inefficient since they generate too much traffic. Informed algorithms are more scalable since they have information that guides the search and the number of generated messages in the process is lower than in blind strategies. For these reasons, the use of informed strategies is more appropriate. However, not all the entities have enough information in the system to consider informed strategies. Therefore, entities should be able to choose which strategy is more appropriate by taking into account their information. In this situation, structures that consider semantic information such as Semantic Overlay Networks

can guide the search when the entities do not have enough information to decide the best neighbor to forward the query to.

Self-Adaptation. The system structure should not be rigid: systems are not static; collaborations between entities may change; service demand can change; unexpected failures might appear; or entities might leave the system. Therefore, the structure should facilitate the adaptation at run time in order to improve the efficiency of the search process (Weyns & Georgeff, 2010).

Data Heterogeneity. Although there are semantic-free approaches such as DHT systems that provide good performance for key discovery, they are not as efficient as semantic approaches for other types of queries such as text queries. Moreover, considering the heterogeneity of entities, an important issue is the inclusion of semantics in the search process. Semantics provides a mechanism to facilitate the interoperability of entities that require and offer services or resources (McIlraith et al., 2001; Wei & Blake, 2010) and to improve the results of the searches. Semantics could not only be included in the service descriptions and in the search process, but also could be included in the structure of the network.Moreover, not all the semantic service descriptions are annotated using the same ontological language. Even the service descriptions that use the same language could use different ontologies that should be aligned (Shvaiko & Euzenat, 2008). It is nevertheless important to consider that not all the entities of the system provide semantic annotated information. For this reason, the integration of both semantic and syntactic information, as well as the use of mechanisms to align ontologies and translate different descriptions to a common model, facilitates the integration of heterogeneous entities.

Collaboration issues. In systems where only local knowledge is available and there is no a predefined structure, the success of the search process relies on the collaboration of the entities that are part of the system. Nevertheless, this is not a very common situation in open and dynamic systems where the entities that belong to the system frequently change. A common problem in P2P systems are the free riders. Free riders are peers that only download resources from other peers. Moreover, in MAS, self-interested agents that decide to pursue its own goals and not to collaborate to accomplish other goals are also present. Therefore, the system should provide mechanisms to deal with this problem and encourage the collaboration among agents. Malicious agents could also appear and the entities that are part of the system should be able to detect and isolate them in order to improve the efficiency of the search process.

Reputation. Entities cooperate with other entities in order to forward requests and provide services. In the absence of a central entity or set of entities that handles the queries and the data the entities must have mechanisms to determine which of their neighbors are trustful and whether or not to forward the query. In decentralized environments, reputation and recommendation mechanisms have been proposed to deal with this task. These mechanisms determine the trustworthiness of other entities considering direct interactions or the information received from other entities. Reputation mechanisms could be considered during the search process as a criteria to determine which entity is more suitable to forward a query or provide a service.

Beyond functional parameters. In the majority of the proposals presented, the search process is reduced to finding an accurate result considering simple functional criteria. However, in SOE, since there is a large number of entities a set of similar service providers is easily found. Therefore, more information is necessary to determine the best provider. To carry out this ranking, non-functional parameters should be included in the service descriptions (Papazoglou et al., 2008; Chaari et al., 2008). However, there is no a standard way to include these parameters inside the semantic descriptions. This makes their usability in the discovery process difficult.

Entities as agents. Entities that populate current systems could be seen as agents with complex capabilities that interact with others in order to achieve common or individual goals. Agent capabilities allow them to be aware of their situation in the system and act in consequence. Therefore, they can incorporate some of the features that we have mentioned (i.e., trust and reputation models) in order to establish more reliable links with other entities in the system. They can also use their previous experience to improve the search process. Another interesting point to consider is the inclusion of MAS features in the search process. Most of the approaches are based on statistical or semantic information about the resources. Trust and reputation introduce new information that could give more flexibility and efficiency to the search process (L et al., 2010). Moreover, negotiations (Bromuri et al., 2009; Liu & Schmeck, 2010), organizational information (Fernández et al., 2006), or behavioral aspects (Cong & Fernández, 2010) could be included in the system to enhance the search process.

An approach that provides an structure and a search mechanism that includes all these features could be considered suitable to deal with the search of resources in open, dynamic and distributed environments in an efficient way.

8 Acknowledgements

References

Adamic, L. A. & Adar, E. (2005). How to search a social network. *Social Networks*, **27**, 2005.

Adamic, L. A., Lukose, R. M., Puniyani, A. R., & Huberman, B. A. (2001). Search in power-law networks. *Physical Review*, **64**(4).

Amaral, L. & Ottino, J. (2004). Complex networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, **38**, 147–162.

Argente, E., Botti, V., Carrascosa, C., Giret, A., Julian, V., & Rebollo, M. (2011). An abstract architecture for virtual organizations: The thomas approach. *Knowledge and Information Systems*, **29**, 379–403.

Babaoglu, O., Meling, H., & Montresor, A. (2002). Anthill: a framework for the development of agent-based peer-to-peer systems. In *22nd International Conference on Distributed Computing Systems*, pages 15 – 22.

Bachlechner, D., Siorpaes, K., Fensel, D., & Toma, I. (2006). Web service discovery – a reality check. In *Proceedings of the 3rd European Semantic Web Conference*.

Bailey, J. (2006). Fast discovery of interesting collections of web services. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 152–160. IEEE Computer Society.

Barabasi, A. L. & Albert, R. (1999). Emergence of scaling in random networks. *Science (New York, N.Y.)*, **286**, 509–512.

Basters, U. & Klusch, M. (2006). Rs2d: Fast adaptive search for semantic web services in unstructured p2p networks. In *International Semantic Web Conference*, volume 4273 of *Lecture Notes in Computer Science*, pages 87–100. Springer.

Ben-Ami, D. & Shehory, O. (2005). A comparative evaluation of agent location mechanisms in large scale mas. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, pages 339–346, New York, NY, USA. ACM.

Bianchini, D., Antonellis, V. D., & Melchiori, M. (2009). Service-based semantic search in p2p systems. In *Proceedings of the 2009 Seventh IEEE European Conference on Web Services*, ECOWS '09, pages 7–16, Washington, DC, USA. IEEE Computer Society.

Bisnik, N. & Abouzeid, A. (2005). Modeling and analysis of random walk search algorithms in p2p networks. In *Proceedings of the 2nd International Workshop on Hot Topics in Peer-to-Peer Systems*, pages 95–103. IEEE Computer Society.

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., & Hwang, D.-U. (2006). Complex networks: Structure and dynamics. *Physics Reports*, **424**(4-5), 175 – 308.

Brazier, F. M. T., Kephart, J. O., Parunak, H. V. D., & Huhns, M. N. (2009). Agents and service-oriented computing for autonomic computing: A research agenda. *IEEE Internet Computing*, **13**.

Brogi, A., Corfini, S., Aldana, J., & Navas, I. (2006). Automated discovery of compositions of services described with separate ontologies. *ICSOC 2006*.

Bromuri, S., Urovi, V., Morge, M., Stathis, K., & Toni, F. (2009). A multi-agent system for service discovery, selection and negotiation. In *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1395–1396. International Foundation for Autonomous Agents and Multiagent Systems.

Campo, C., Martin, A., Garcia, C., & Breuer, P. (2002). Service discovery in pervasive multi-agent systems. In *AAMAS Workshop on Ubiquitous Agents on embedded, wearable, and mobile agents*.

Cao, J., Yao, Y., Zheng, X., & Liu, B. (2010). Semantic-based self-organizing mechanism for service registry and discovery. In *Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 345 –350.

Chaari, S., Badr, Y., & Biennier, F. (2008). Enhancing web service selection by qos-based ontology and ws-policy. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 2426–2431. ACM.

Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., & Shenker, S. (2003). Making gnutella-like p2p systems scalable. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, pages 407–418. ACM.

Cholvi, V. & Rodero-Merino, L. (2007). Using random walks to find resources in unstructured self-organized p2p networks. In *Proceedings of the IEEE Workshop on Dependable Application Support in Self-Organizing Networks*, pages 51–56.

Cong, Z. & Fernández, A. (2010). Behavioral matchmaking of semantic web services. In *Proceedings of the 4th International Joint Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2)*, volume 667, pages 131–140.

Constantinescu, I. & Faltings, B. (2003). Efficient matchmaking and directory services. In *Web Intelligence*, pages 75–81. IEEE Computer Society.

Crespo, A. & Garcia-Molina, H. (2002). Routing Indices For Peer-to-Peer Systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 23. IEEE Computer Society.

Crespo, A. & Garcia-Molina, H. (2004). Semantic overlay networks for p2p systems. In *Proceedings of the 3rd International Workshop on Agents and Peer-to-Peer Computing*, volume 3601 of *Lecture Notes in Computer Science*, pages 1–13. Springer.

Dell'amico, M. (2006). Highly Clustered Networks with Preferential Attachment to Close Nodes. In *Proceedings of the European Conference on Complex Systems 2006*.

Dimakopoulos, V. V. & Pitoura, E. (2003). A peer-to-peer approach to resource discovery in multi-agent systems. In *Proceedings of Cooperative Information Agents*, volume 2782 of *Lecture Notes in Computer Science*, pages 62–77. Springer.

Ding, D., Liu, L., & Schmeck, H. (2010). Service discovery in self-organizing service-oriented environments. In *Proceedings of the 2010 IEEE Asia-Pacific Services Computing Conference*, pages 717–724. IEEE Computer Society.

Fernández, A., Vasirani, M., Cáceres, C., & Ossowski, S. (2006). Role-based service description and discovery. In *AAMAS-06 Workshop on Service-Oriented Computing and Agent-Based Engineering*, pages 1–14.

Fernández, A., Ossowski, S., & Vasirani, M. (2008). General architecture. In *CASCOM: Intelligent Service Coordination in the Semantic Web*, Whitestein Series in Software Agent Technologies and Autonomic Computing, pages 143–160.

Gkantsidis, C., Mihail, M., & Saberi, A. (2006). Random walks in peer-to-peer networks: Algorithms and evaluation. *Performance Evaluation*, **63**(3), 241–263.

Gummadi, P. K., Saroiu, S., & Gribble, S. D. (2002). A measurement study of napster and gnutella as examples of peer-to-peer file sharing systems. *SIGCOMM Comput. Commun. Rev.*, **32**, 82–82.

He, Q., Yan, J., Yang, Y., Kowalczyk, R., & Jin, H. (2008). Chord4s: A p2p-based decentralised service discovery approach. In *IEEE International Conference on Services Computing*, volume 1, pages 221 –228.

Hughes, D., Coulson, G., & Walkerdine, J. (2010). *A Survey of Peer-to-Peer Architectures for Service Oriented Computing*, pages 1–19. IGI Global.

Huhns, M. N. (2002). Agents as web services. *IEEE Internet Computing*, pages 93–95.

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. (2005). Research directions for service-oriented multiagent systems. *IEEE Internet Computing*, **9**, 65–70.

Jha, S., Chalasani, P., Shehory, O., & Sycara, K. (1998). A formal treatment of distributed matchmaking. In *Proceedings of the 2nd International Conference on Autonomous Agents*, pages 457–458, New York, NY, USA. ACM.

Kalogeraki, V., Gunopulos, D., & Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. In *Proceedings of the eleventh international conference on Information and knowledge management (CIKM '02)*, pages 300–307, New York, NY, USA. ACM.

Kleinberg, J. (2001). Small-world phenomena and the dynamics of information. In *In Advances in Neural Information Processing Systems (NIPS)*, pages 431–438. MIT Press.

Kleinberg, J. (2006). Complex networks and decentralized search algorithms. In *Proceedings of the International Congress of Mathematicians (ICM)*.

Kleinberg, J. M. (2000). Navigation in a small world. *Nature*, **406**, 845.

Klusch, M. & Sycara, K. (2001). *Brokering and matchmaking for coordination of agent societies: a survey*, pages 197–224. Springer-Verlag, London, UK.

Klusch, M., Fries, B., & Sycara, K. (2006). Automated semantic web service discovery with owls-mx. In *Proceedings of the 5th international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, pages 915–922, New York, NY, USA. ACM.

Ko, S. Y., Gupta, I., & Jo, Y. (2008). A new class of nature-inspired algorithms for self-adaptive peer-to-peer computing. *ACM Trans. Auton. Adapt. Syst.*, **3**(3), 11:1–11:34.

Kota, R., Gibbins, N., & Jennings, N. R. (2009). Self-organising agent organisations. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 797–804, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

L, G., Wang, Y., & Orgun, M. (2010). Optimal social trust path selection in complex social networks. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1391–1398. AAAI Press.

Liang, J., Kumar, R., & Ross, K. (2005). Understanding kazaa. In *Proceedings of the 5th New York Metro Area Networking Workshop (NYMAN)*.

Liu, L. & Schmeck, H. (2010). Enabling self-organising service level management with automated negotiation. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT '10, pages 42–45. IEEE Computer Society.

Lopes, A. L. & Botelho, L. M. (2008). Improving multi-agent based resource coordination in peer-to-peer networks. *Journal of Networks*, **3**, 38–47.

Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., & Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, **7**, 72–93.

Lv, Q., Cao, P., Cohen, E., Li, K., & Shenker, S. (2002). Search and replication in unstructured peer-to-peer networks. In *Proceedings of the 16th international conference on Supercomputing*, ICS '02, pages 84–95, New York, NY, USA. ACM.

Manku, G. S., Bawa, M., Raghavan, P., & Inc, V. (2003). Symphony: Distributed hashing in a small world. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, pages 127–140.

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., & Sycara, K. (2004). Owl-s: Semantic markup for web services. `http://www.w3.org/Submission/OWL-S/`.

Martin, D., Paolucci, M., & Wagner., M. (2007). Towards semantic annotations of web services: Owl-s from the sawsdl perspective. In *Proceedings of Workshop OWL-S: Experiences and Directions at 4th European Semantic Web Conference*.

Maymounkov, P. & Mazieres, D. (2002). Kademlia: A peer-to-peer information system based on the xor metric. *Proceedings of the 1st International Workshop on Peer-to Peer Systems (IPTPS02)*.

McIlraith, S. A., Son, T. C., & Zeng, H. (2001). Semantic web services. *IEEE Intelligent Systems*, **16**, 46–53.

Meshkova, E., Riihijärvi, J., Petrova, M., & Mähönen, P. (2008). A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, **52**, 2097–2128.

Michlmayr, E. (2006). Ant algorithms for search in unstructured peer-to-peer networks. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*.

Mokhtar, S., Kaul, A., Georgantas, N., & Issarny, V. (2006). Towards efficient matching of semantic web service capabilities. In *Proceedings of International Workshop on Web Services - Modeling and Testing*.

Moore, M. & Suda, T. (2002). A decentralized and self-organizing discovery mechanism. In *Proceedings Of the 1st Annual Symposium on Autonomous Intelligent Networks and Systems*.

Mullender, S. & Vitanyi, P. (1988). Distributed match-making. *Algorithmica*, **3**, 367–391.

Nguyen, V. & Martel, C. (2005). Analyzing and characterizing small-world graphs. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics.

Ogston, E. & Vassiliadis, S. (2001a). Local distributed agent matchmaking. In *Proceedings of the 9th International Conference on Cooperative Information Systems*.

Ogston, E. & Vassiliadis, S. (2001b). Matchmaking among minimal agents without a facilitator. In *Proceedings of the 5th International Conference on Autonomous Agents*, pages 608–615.

Ouksel, A., Babad, Y., & Tesch, T. (2004). Matchmaking software agents in b2b markets. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*.

Papazoglou, M. P., Traverso, P., Dustdar, S., Leymann, F., & Krämer, B. J. (2006). Service-oriented computing: A research roadmap. In F. Cubera, B. J. Krämer, and M. P. Papazoglou, editors, *Service Oriented Computing (SOC)*, number 05462 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

Papazoglou, M. P., Traverso, P., Dustdar, S., & Leymann, F. (2007). Service-oriented computing: State of the art and research challenges. *Computer*, **40**, 38–45.

Papazoglou, M. P., Traverso, P., Dustdar, S., & Leymann, F. (2008). Service-Oriented Computing: A Research Roadmap. *International Journal of Cooperative Information Systems*, **17**(02).

Perryea, C. & Chung, S. (2006). Community-based service discovery. In *Proceedings of the International Conference on Web Services*, pages 903 –906.

Pirr and, G., Trunfio, P., Talia, D., Missier, P., & Goble, C. (2010). Ergot: A semantic-based system for service discovery in distributed infrastructures. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, pages 263 –272.

Prabhu, S. (2007). Towards distributed dynamic web service composition. In *ISADS '07: Proceedings of the 8th International Symposium on Autonomous Decentralized Systems*, pages 25–32, Washington, DC, USA. IEEE Computer Society.

Rao, J. & Su, X. (2004). A survey of automated web service composition methods. In *Proceedings of the 1st International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004*, pages 43–54.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content-addressable network. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '01)*, New York, NY, USA. ACM.

Risson, J. & Moors, T. (2006). Survey of research towards robust peer-to-peer networks: search methods. *Computer Networks*, **50**, 3485–3521.

Rowstron, A. I. T. & Druschel, P. (2001). 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, pages 329–350. Springer-Verlag.

Satyanarayanan, M. (2001). Pervasive computing: Vision and challenges. *IEEE Personal Communications*, **8**, 10–17.

Schmidt, C. & Parashar, M. (2004). A peer-to-peer approach to web service discovery. *World Wide Web*, **7**, 211–229.

Shvaiko, P. & Euzenat, J. (2008). Ten challenges for ontology matching. In R. Meersman and Z. Tari, editors, *On the Move to Meaningful Internet Systems: OTM 2008*, volume 5332 of *Lecture Notes in Computer Science*, pages 1164–1182. Springer Berlin / Heidelberg.

Sigdel, K., Bertels, K., Pourebrahimi, B., Vassiliadis, S., & Shuai, L. (2005). A framework for adaptive matchmaking in distributed computing. In *Proceedings of GRID Workshop*.

Simsek, Ö. & Jensen, D. (2005). Decentralized search in networks using homophily and degree disparity. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 304–310.

Sivashanmugam, K., Verma, K., & Sheth, A. (2004). Discovery of web services in a federated registry environment. volume 0, page 270, Los Alamitos, CA, USA. IEEE Computer Society.

Skoutas, D., Sacharidis, D., Kantere, V., & Sellis, T. (2008). Efficient semantic web service discovery in centralized and p2p environments. In A. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. Finin, and K. Thirunarayan, editors, *The Semantic Web - ISWC 2008*, volume 5318 of *Lecture Notes in Computer Science*, pages 583–598.

Slivkins, A. (2005). Distance estimation and object location via rings of neighbors. In *Proceedings of the 24th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 41–50. ACM Press.

Srinivasan, N., Paolucci, M., & Sycara, K. (2004). Adding owl-s to uddi, implementation and throughput. In *First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*.

Stoica, I., Morris, R., Karger, D., Kaashoek, F., & Balakrishnan, H. (2001). Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. *Computer Communication Review*, **31**(4), 149–160.

Sycara, K., Paolucci, M., Soudry, J., & Srinivasan, N. (2004). Dynamic discovery and coordination of agentbased semantic web services. *IEEE Internet Computing*, **8**, 66–73.

Thadakamalla, H. P., Albert, R., & Kumara, S. R. T. (2007). Search in spatial scale-free networks. *New Journal of Physics*, **9**.

Travers, J. & Milgram, S. (1969). An experimental study of the small world problem. *Sociometry*, **32**.

Tsoumakos, D. & Roussopoulos, N. (2003). Adaptive probabilistic search for peer-to-peer networks. In *Peer-to-Peer Computing*, pages 102–109.

Upadrashta, Y., Vassileva, J., & Grassmann, W. (2005). Social networks in peer-to-peer systems. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*.

Val, E. D. & Rebollo, M. (2007). Service Discovery and Composition in Multiagent Systems. In *Proceedings of 5th European Workshop On Multi-Agent Systems (EUMAS 2007)*, pages 197–212. Association Tunisienne D'Intelligence Artificielle.

Val, E. D., Rebollo, M., & Botti, V. (2011). Introducing homophily to improve semantic service search in a self-adaptive system. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*.

Vanthournout, K., Deconinck, G., & Belmans, R. (2005). A taxonomy for resource discovery. *Personal Ubiquitous Comput.*, **9**, 81–89.

Vázquez-Salceda, J., Vasconcelos, W. W., Padget, J., Dignum, F., Clarke, S., & Roig, M. P. (2010). Alive: an agent-based framework for dynamic and robust service-oriented applications. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1 - Volume 1*, AAMAS '10, pages 1637–1638, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Wang, X. F. & Chen, G. (2003). Complex networks: small-world, scale-free and beyond. *Circuits and Systems Magazine, IEEE*, **3**(1), 6 – 20.

Watts, D., Dodds, P., & Newman, M. (2002). Identity and search in social networks. *Science*, **296**(5571), 1302 – 1305.

Watts, D. J. (2004). The "New" Science of Networks. *Annu. Rev. Sociol.*, **30**, 243–270.

Watts, D. J. & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, **393**, 440–442.

Wei, Y. & Blake, M. B. (2010). Service-oriented computing and cloud computing:challenges and opportunities. *IEEE Internet Computing*, **14**, 72–75.

Weyns, D. & Georgeff, M. (2010). Self-adaptation using multiagent systems. *Software, IEEE*, **27**(1), 86 –91.

Xiao, S. & Xiao, G. (2006). On degree-based decentralized search in complex networks. *CoRR*.

Yang, B. & Garcia-Molina, H. (2002). Efficient search in peer-to-peer networks. In *Proceedings of the International Conference on Distributed Computing Systems (ICDCS)*.

Yang, B. & Garcia-Molina, H. (2003). Designing a super-peer network. *International Conference on Data Engineering*, page 49.

Yu, S., Liu, J., & Le, J. (2004). Decentralized web service organization combining semantic web and peer to peer computing. In *ECOWS*, volume 3250 of *Lecture Notes in Computer Science*. Springer.

Zhang, H., Croft, W. B., Levine, B., & Lesser, V. (2004a). A multi-agent approach for peer-to-peer based information retrieval system. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '04, pages 456–463, Washington, DC, USA. IEEE Computer Society.

Zhang, H., Goel, A., & Govindan, R. (2004b). Using the small-world model to improve Freenet performance. *Computer Networks*, **46**(4), 555–574.

Zhong, M. (2006). Popularity-biased random walks for peer-to-peer search under the square-root principle. In *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS)*.