ERGOT: Combining DHTs and SONs for Semantic-Based Service Discovery on the Grid

Giuseppe Pirró, Domenico Talia, Paolo Trunfio
{gpirro,talia,trunfio}@deis.unical.it
Rende University of Calabria, Via P. Bucci 41C, 87036 (CS), Italy

Paolo Missier, Carole Goble
{pmissier,carole}@cs.man.ac.uk
University of Manchester, Oxford Road, M13 9PL Manchester, UK

CoreGRID Technical Report
Number TR-0177
August 31, 2008

Institute on Knowledge and Data Management

CoreGRID - Network of Excellence
URL: http://www.coregrid.net

CoreGRID is a Network of Excellence funded by the European Commission under the Sixth Framework Programme

Project no. FP6-004265

ERGOT: Combining DHTs and SONs for Semantic-Based Service Discovery on the Grid

Giuseppe Pirró, Domenico Talia, Paolo Trunfio
{gpirro,talia,trunfio}@deis.unical.it
Rende University of Calabria, Via P. Bucci 41C, 87036 (CS), Italy

Paolo Missier, Carole Goble
{pmissier,carole}@cs.man.ac.uk
University of Manchester, Oxford Road, M13 9PL Manchester, UK

CoreGRID TR-0177
August 31, 2008

Abstract

The Grid has rapidly moved from a toolkit-centered approach, composed of a set of middleware tools, toward a more application-oriented Service Oriented Architecture in which resources are exposed as services. The soaring number of available services advocates distributed and semantic-based discovery architectures. Distribution promotes scalability and fault-tolerance whereas semantics is required to provide for meaningful descriptions of services and support their efficient retrieval. Current approaches exploit either Semantic Overlay Networks (SONs) or Distributed Hash Tables (DHTs) sweetened with some ”semantic sugar”. SONs enable semantic driven query answering but are less scalable than DHTs, which on their turn, feature efficient but semantic-free query answering based on ”exact” matching. This paper presents the ERGOT system combining DHTs and SONs to enable distributed and semantic-based service discovery on the Grid. We argue that these two models can benefit from each other in the sense that SONs can be constructed by exploiting DHTs mechanisms thus enlightening the way to the semantics-free content publishing and retrieval mechanisms of the latter. In particular, ERGOT allows establishing semantic links among peers, and the SONs, by scrutinizing semantic service descriptions they advertise on the DHT. As we will show semantic links can also be viewed as *semantic shortcuts* on the DHT. Finally, ERGOT exploits an ad hoc semantic similarity metric to perform service matchmaking and numerically rank results .

1 Introduction

The Grid has been conceived as a distributed platform for resource sharing and problem solving in which participants, possibly from different organizations, choose to cooperate by dynamically forming virtual organizations. The first generation of Grids was middleware-centric in the sense that it provided a set of software components and protocols definitions to form a toolkit. More recently, the attention has shifted toward the application layer and, in particular, the concept of service orientation as a way to virtualize and unify resources, services and information has been introduced. As a matter of fact, be either middleware-centric or service-oriented, the Grid requires adequate mechanisms to allow the resource orchestra to coordinate and play the same tune. In this paper we are concerned in investigating and addressing the service discovery problem. This problem can be characterized in the general context of the Service Oriented Architecture (SOA) model from which the Open Grid Service Architecture (OGSA) has been conceived. The SOA model has been widely recognized as a promising form of distributed computing on the Internet. In this architecture three main actors can be recognized: (i) a service provider, which advertises information about services it wants to make accessible; (ii) a service registry, which stores information about available services; (iii) a service

This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

requester, which queries the registry to look for services satisfying some requirements. However, this formulation suffers from some limitations:

- It provides an approach to service discovery based on centralized registries (i.e., UDDI). Such an architecture is unlikely to go through the soaring rate of incoming requests and in case of registry crash jeopardizes the whole service discovery mechanism.
- The lack of semantically-rich service descriptions and complex query mechanisms to perform service matchmaking makes it harder and harder to find services that fit one's needs.

In order to mitigate these issues, two profitable research strands are Peer-to-Peer (P2P) and the Semantic Web (SW).

P2P architectures guarantee decentralization, scalability and fault tolerance. There is a variety of P2P network models ranging from unstructured (a la Gnutella) or hybrid based on Super Peer (SP), to structured based on distributed hash tables (DHTs). In particular in the Grid context, several research strands have investigated how P2P architectures can be exploited for efficient resource (e.g., service) discovery. [19, 7, 20].

On the other hand, SW technologies allow for semantic characterization of resources through the use of ontologies that provide shared and formally defined terminologies describing knowledge domains [6]. In the Grid, a major initiative in this strand of research is the Semantic Open Grid Service Architecture (S-OGSA) [3] which extends OGSA with some services to manage the semantics of Grid resources. In a more general context of web services, some initiatives such as OWL-S (`http://www.w3.org/Submission/OWL-S/`), SAWSDL (`http://www.w3.org/2002/ws/sawsdl/`) and WSMO (`http://www.wsmo.org/`) have recently been proposed to semantically characterize web service description.

In this paper, we investigate how P2P models and SW technologies can be exploited to perform service discovery in open environments (e.g., the Grid). P2P and SW technologies, separately or together, gave birth to several approaches to service discovery. In [2, 12, 5] the semantic-based service discovery issue has been addressed but not that of centralization. Other approaches such as [1, 17] do just the opposite. More comprehensive systems combine P2P and SW technologies in different fashions. The Hypercube [16] exploits ontologies to give positions to peers in the network. The SPiDer system [15] combines ontologies and a SP-based DHT. WSPDS [9] exploits a Semantic Overlay Network [4] and WSDL-S to semantically describe services. Generally speaking, decentralized and semantic-based approaches exploit either SONs (e.g., WSPDS) or a structured architectures such as DHTs in which services are semantically characterized through ontologies (e.g., SPiDer). Both SONs and DHTs have their pros and cons. In a SON, peers choose their neighbors according to a criterion of semantic similarity between services they provide. Here, service discovery is not based on "exact" matching. Conversely, in a DHT peers are assigned neighbors algorithmically (in a semantic-free way) and services can only be discovered through "exact" matching. This allows DHTs to obtain maximum precision intended as the fraction of results that match a given key. However, since the definition of discovering [1] itself suggests that one does not exactly know in advance what it is discovering, DHTs do not guarantee maximum recall intended as the fraction of results that are relevant to a request. In this respect, SONs can perform better since they go beyond exact matching. Finally, current approaches do not feature effective service matchmaking techniques as those devised in centralized initiatives (e.g., [2, 12, 5]).

In this paper we present the ERGOT (Efficient Routing Grounded On Taxonomy) system combining DHTs and SONs to perform semantic-based service discovery and featuring a service matchmaking mechanism based on an ad-hoc semantic similarity metric. The contributions of this paper can be summarized as follows:

- ERGOT combines DHTs and SONs to perform semantic-based service discovery. We argue that these two models can benefit from each other in the sense that SONs can be constructed by exploiting DHTs mechanisms and hence the former can help to light the way to the semantics-free content publishing and retrieval approach of the latter.
- ERGOT allows peers to build semantic links, and then a SON, during their normal activities in a DHT (e.g., service advertising). As we will show, semantic links can also be viewed as semantic shortcuts on the DHT.
- ERGOT features different and flexible service discovery mechanisms that can exploit the SON and/or the DHT enhanced with semantic shortcuts.

[1]Discover: to find information, a place or an object, especially for the first time. Cambridge dictionary online, http://dictionary.cambridge.org.

- ERGOT performs service matchmaking by an ad-hoc similarity metric that compares operation names with related inputs and outputs in a service request and profile. Results are given as numeric values. This approach differs from the state of the art techniques (e.g., [10]) which give as output semantic relations (e.g., exact, subsume) obtained through time-expensive reasoning operations.

The rest of this paper is organized as follows. In Section 2 we provide a brief background on DHTs and SONs and discuss how these two P2P models can be profitably combined. In Section 3, we present the semantic service discovery model exploited in ERGOT. Moreover here we introduce the service matchmaker based on semantic similarity. In Section 4, we present the architecture of ERGOT and its functioning principles. In particular, we thoroughly analyze the provider and requester perspectives. In Section 5, we review related work and compare ERGOT w.r.t the state of the art.

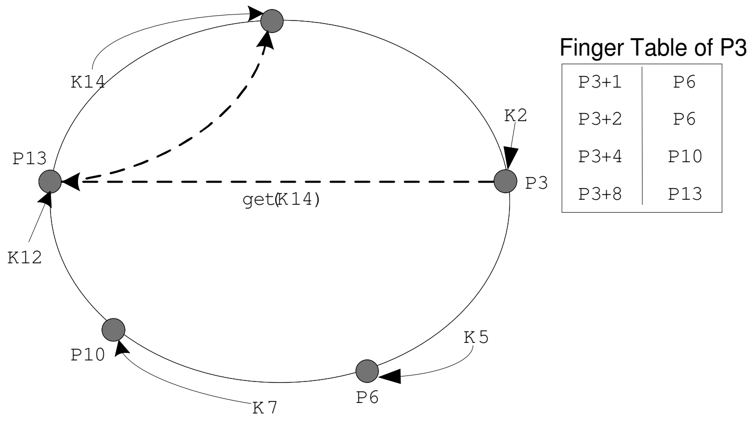
2 On combining DHTs and SONs

This section provides a brief background on DHTs and SONs and some insights on why it is useful to combine these two P2P architectures in the context of service discovery.

2.1 Distributed Hash Tables (DHTs)

DHTs have been recognized as a prominent network paradigm due to their scalability properties and efficiency in retrieving content. In this paper we focus on the Chord DHT, tough any other DHT can be used instead. Chord [18] organizes peers into an *m*-bit identifiers ring, in the interval [0, $2^{(}m - 1)$], which is the basis for routing and locating objects. Both peers and objects are assigned *m*-bit keys by exploiting consistent hashing which guarantees that the addition or removal of one component in the ring does not significantly affect the network organization. In particular, an object (or a reference to it) is stored in the peer that follows it in the ring. This peer is called the *successor*.

Figure 1 shows a simple 4-bit Chord network. For instance, the key with **id** 2 (i.e., *K2*) is assigned to its successor, that is, peer P3 which is the peer following the *id* 2 in the Chord ring proceeding in clockwise direction. Note that with 4 bits it is possible to create up to 16 keys which will be assigned to peers and objects. In order to perform efficient routing each peer maintains a *finger table* which contains the Chord *ids* of its neighbors. The number of neighbors of a peer is *O(logN)* where *N* is the number of peers in the network. Neighbors are peers located on the ring at exponentially increasing distance from a given peer. For instance, the finger table of *P3* maintains information about *P3*'s neighbors, that is, *P6*, *P10* and *P13*.



P3+1	P6
P3+2	P6
P3+4	P10
P3+8	P13

Figure 1: A 4-bits Chord network

The state of the network is maintained by a *stabilization protocol* which refreshes information in the peers' finger tables. Chord is a dynamic system where peers can join and leave the network at their will. When a peer joins the network it is assigned an identifier (e.g., obtained by hashing its IP address) to which it sends a request through an existing node in the ring. This way, the peer can reach its successor from which it obtains the keys it is responsible for. At this point the involved finger tables are updated. In case of departure, keys a peer is responsible for are assigned to its successor. Chord features two basic primitives: *put(key, value)* which is used to publish content in the network and *get(key)* which given a key finds the value associated to it. Chord performs exact look up meaning that a certain content can be retrieved only if its *id*, and then the associated key, is exactly known. In fact, a slight difference in terms of *id* (e.g., a file name) can generate completely different keys which can possibly be located in different nodes. In more detail, a request lookup requires at most *O(logN)* hops [18]. Chord routing is clockwise greedy meaning that at each hop a request is forwarded to the peer (in the finger table) whose *id* most immediately precedes the destination point proceeding clockwise. In Figure 1, the request posed by *P3* for the key with *id* 14 (i.e., *K14*) is routed at first hop to the peer in *P3*'s finger table closest to (but not higher than) 14, that is, *P13* which on its turn, by looking at its finger table can easily find the peer responsible for *K14* (i.e., *P1*).

2.2 Semantic Overlay Networks (SONs)

In early P2P systems, nodes were typically connected to each other in a "blind" way (e.g., by selecting a random number of neighbors) and queries were propagated along these links or were collected by a central server and then propagated to the peers in broadcast. Such approaches do not consider at all content stored by peers which actually can be the discriminating factor in building "intelligent" strategies for clustering peers and routing queries. Crespo and Garcia-Molina proposed the concept of Semantic Overlay Networks (SON) as a new paradigm for organizing peers and enhancing content search [4]. This approach is based on the idea that peers with similar interests, deducted by content they hold, have to be clustered together for speeding up query routing and providing better recall to their information needs. The concept of SON introduces some challenges due to the design of mechanisms to perform peer clustering, classification of content (e.g., documents) and choice of the proper SONs to which a peer has to join. As factor for partitioning an unstructured P2P network in SONs, authors proposed to exploit classification hierarchies. These furnish the semantic underpinning for classifying content, peers and queries. Authors showed the suitability of this approach both in terms of number of messages sent over the network and recall as compared to the Gnutella flooding-based approach. However, in some phases this architecture relies on flooding mechanisms as, for instance, when a peer has to choose the proper SONs to join or when a query reaches a SON (in this phase the query is broadcast within the SON). In Figure 2 an example of SON is shown. It can be noted that SON connections among peers are "logical", that is, constructed according to a criterion of semantic similarity. Conversely, physical connections do not take into account semantic aspects.

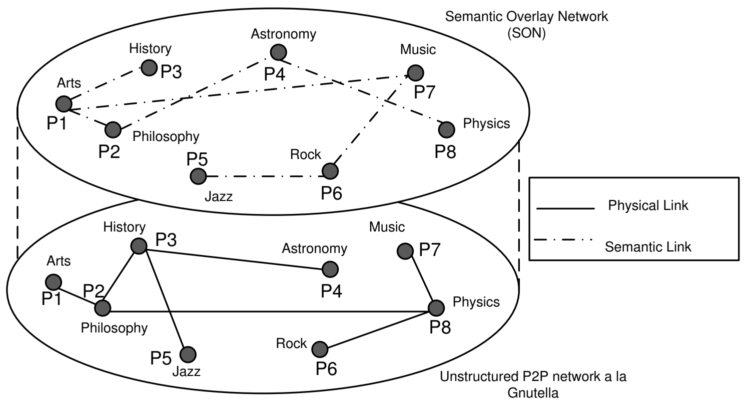


Figure 2: An example of SON. Filled rounds represent peers while labels peer interests

2.3 Why combining DHTs and SONs?

In this section, we provide a comparison of the main features of these two models. Table 1 reports the results of this analysis which is useful to motivate the design of the ERGOT system that combines DHTs and SONs.

As can be noted in Table 1, DHTs are very scalable and guarantee efficient lookup at the cost that the "identity" of what one is looking for has to be exactly known. SONs are more flexible as intrinsically perform semantic-based query answering and then go beyond exact matching. However, their performance heavily depends on how semantic links are created and the cost to create the semantic links (i.e., how to find neighbors) has also to be taken into account.

	DHTs	**SONs**
Content Placing	Fixed, based on hashing	Each peer is responsible for its content
Content Retrieval	Exact lookup	Flexible
Network Structure	Fixed	Variable
Lookup cost	At most O(logN)	Variable
Topology Management	Stabilization protocol	Peers need to rearrange semantic link
Semantics	None	Exploit some semantic artifact (e.g., ontologies)

Table 1: Comparison between DHTs and SONs

DHTs and SONs have been (separately) exploited in recent service discovery initiatives (e.g., [15, 21, 9]). We claim that these two models can profitably be combined in the context of service discovery by observing that:

- Peers, during their normal interaction in the DHT (e.g., service advertising) can discover peers with similar content with which establishing semantic links. This way the SON can be constructed without additional costs. However, the definition of some shared semantic artifact on which the notion of ”similar content” can be defined is required.
- Content lookup can be performed by considering both the exact lookup techniques of the DHT and semantic-similarity based of SONs. In particular, when searching for a particular service, a peer can look at its semantic neighbors to see if they have something which is *similar*, in semantic sense, to the requested service. That complies with the notion of discovery thus going beyond exact lookup.
- The construction of additional links (i.e., semantic links) to those provided by the DHT topology can also improve the DHT’s exact lookup. In fact, a peer when searching for a service can consult its semantic links to see if one of the semantic neighbors is responsible (or it is closer than a traditional DHT neighbor) to the key it is looking for. These additional links can be viewed as semantic shortcuts in the DHT.

In the light of these intuitions we devised the ERGOT system that exploits ontologies as semantic artifacts and an ad-hoc service matchmaker to numerically quantify the similarity between service requests and profiles.

3 A Service Matchmaker based on Semantic Similarity

As discussed in Section 1, a number of pitfalls can be recognized in the semantic-free web service description and discovery mechanism exploited in the SOA. In order to overcome these issues and support distributed service discovery, we devised a semantic-based approach to characterize services that exploits two kinds of ontological knowledge: one to describe service functionalities and the other to annotate service operations with related inputs and outputs. Moreover, an ad-hoc service matchmaker based on semantic similarity supports numeric ranking of results related to a request. In this section we elaborate on these aspects.

3.1 Category and Domain Ontology annotations

In ERGOT, services are advertised by providers in the form of semantically-enhanced profiles. In particular, a provider can perform two ”levels” of annotations. A higher level is exploited to associate a service with one or more concepts belonging to a Category Ontology (CO). This approach resembles the service categorization mechanism provided by UDDI in which some standard taxonomies (e.g., UNSPSC, NAICS) are exploited. The aim of this kind of annotations is to ”summarize” service functionalities. Finer-grained annotations are exploited to annotate operations with related inputs and outputs to concepts belonging to a Domain Ontology (DO) with the aim to provide a more detailed characterization of web services. DO annotations result particularly useful to distinguish between services belonging to the same category.

Figure 3 shows and excerpt of a CO in the bioinformatics domain extracted from the ^{my}Grid ontology [22]. According to our model, a service can be annotated to one or more CO concepts which provide a semantic summarization of the tasks carried out by the service. In Figure 4 it is shown an excerpt of DO even in this case extracted from the

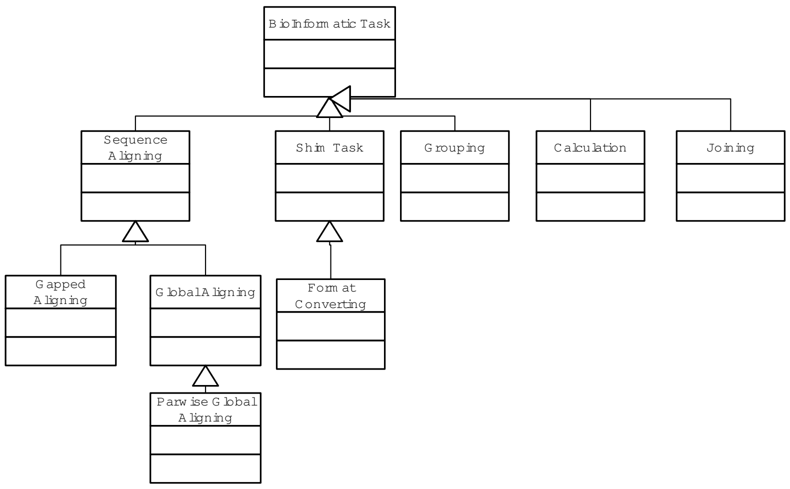
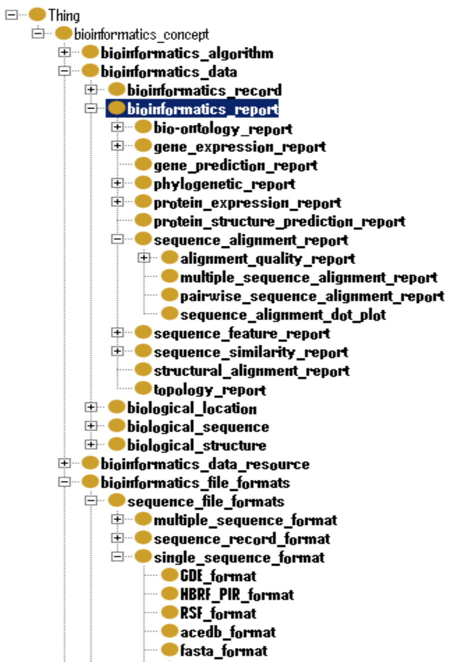


Figure 3: An excerpt of the ^{my}Grid ontology used as Category Ontology

^{my}Grid ontology. As can be noted, concepts in DO are more specific than those in the CO and can profitably be exploited to annotate service operations with related inputs and outputs.



```
Thing
  bioinformatics_concept
    bioinformatics_algorithm
    bioinformatics_data
      bioinformatics_record
      bioinformatics_report
        bio-ontology_report
        gene_expression_report
        gene_prediction_report
        phylogenetic_report
        protein_expression_report
        protein_structure_prediction_report
        sequence_alignment_report
          alignment_quality_report
          multiple_sequence_alignment_report
          pairwise_sequence_alignment_report
          sequence_alignment_dot_plot
        sequence_feature_report
        sequence_similarity_report
        structural_alignment_report
        topology_report
      biological_location
      biological_sequence
      biological_structure
    bioinformatics_data_resource
    bioinformatics_file_formats
      sequence_file_formats
        multiple_sequence_format
        sequence_record_format
        single_sequence_format
          GDE_format
          HBRF_PIR_format
          RSF_format
          acedb_format
          fasta_format
```

Figure 4: An excerpt of the ^{my}Grid ontology used as Domain Ontology

Figure 5 shows an example of annotated service. This service whose name is *nucleotide alignment* has been annotated to the CO concept *global alignment*. The service features an operation named *nucleodite alignment* that takes as input parameter a *nucleotide alignment request* which has been annotated to the DO concept *fasta format* and has as output parameter a *nucleotide alignment return* which has been annotated to the concept *multiple sequence alignment report*. Service discovery is performed by exploiting both CO and DO concepts. In particular, CO concepts are useful to identify a possible set of candidate matching services as they are used to summarize service functionalities. On the other hand, DO concepts are exploited to perform finer-grained analysis of results to find out the most relevant

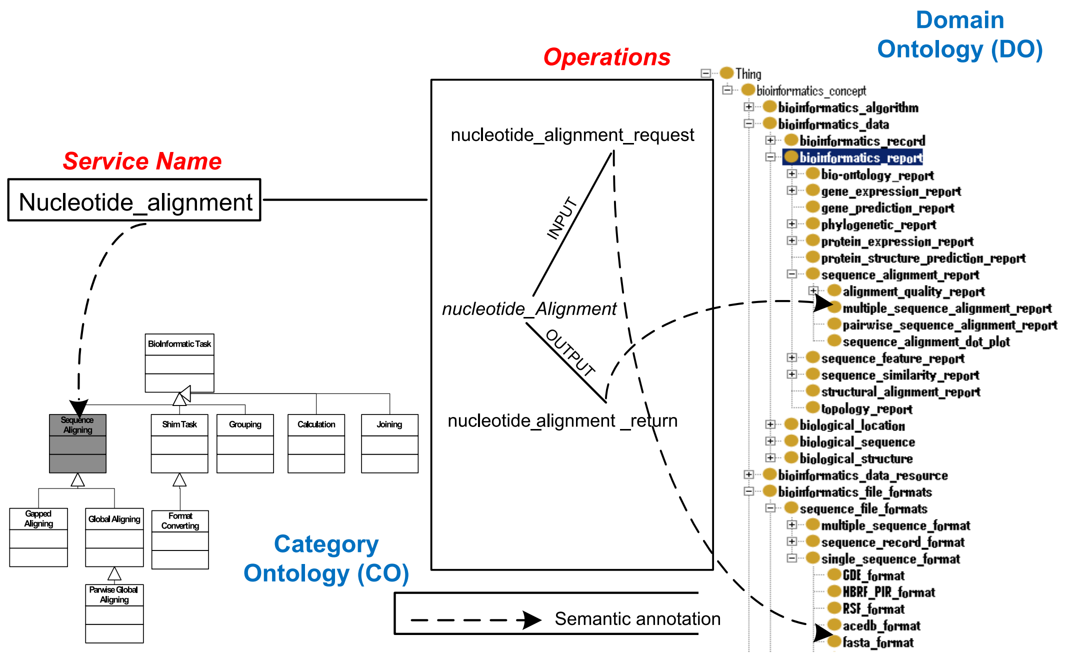


Figure 5: An example of service annotation

through the mechanism described in Section 3.5. Finally, to represent annotations we exploit the lightweight approach of SAWSDL (`http://www.w3.org/2002/ws/sawsdl/`).

3.2 Preliminary Definitions

After introducing the annotation mechanism, in this section we provide some formal definitions adopted throughout the paper.

A service profile $P = \langle sn, \mathcal{O}p \rangle$ is defined by a service name sn and a set of operations $\mathcal{O}p$. An operation $Op \in \mathcal{O}p$ has a name n and a set $\mathcal{I}$ and $\mathcal{O}$ of inputs and outputs, respectively:

$$Op = \langle n, \mathcal{I}, \mathcal{O} \rangle$$

Here sn, n, and each $I \in \mathcal{I}$, $O \in \mathcal{O}$ can be annotated with ontology concepts. We write $ann(x)$ to denote the concept that annotates a generic element x.

A service profile forms a hierarchical structure, and we will use a dot notation to refer to its elements. For example, $P.sn$ is the service name, $P.\mathcal{O}p_i.\mathcal{I}_j$ is the j-th input of the i-th operation, and so forth. A service request R has the following structure: $R = \langle \mathcal{C}, \mathcal{O}p \rangle$ where $\mathcal{O}p$ is optional. It has a structure similar to that of a service profile apart for the fact that instead of having a single service name it has a set of concepts (i.e., $\mathcal{C}$) belonging to the Category Ontology (CO) presented in Section 3.1.

3.3 On semantic service matchmaking

The task of comparing a service request with a service profile is generally referred to as service matchmaking. UDDI adopts a simple approach based on string comparison. In order to improve the poor accuracy of this technique, several approaches have been proposed. These range from pure logic-based approaches [10] exploiting ontologies to non logic-based that can exploit different techniques such as Information Retrieval [5] or Rough Set Theory [12] just to cite a few. Logic-based initiatives, through reasoning operations identify logic relations, such as *exact*, *plugin*, *subsume*, *fail*, between a request and a service profile. Conversely, non-logic-based approaches provide numeric assessments. However, as also observed by [2], it is difficult to numerically quantify some semantic relations such as *subsume* or *plugin*. Therefore, ranking and interpretation of the relevance of results becomes more challenging. More specifically, in distributed contexts only a few initiatives ([21, 15]) have addressed the problem of result ranking but only taking into account Quality of Service (QoS) indicators and not the semantics associated to operation names with related inputs and outputs. In ERGOT, we devised an ad-hoc mechanism for service matchmaking based on semantic similarity that will be described in the next section.

3.4 A service matchmaker based on semantic similarity

The notion of semantic similarity has been widely recognized as important in many research areas ranging from Artificial Intelligence to Cognitive Sciences. In particular, it aims at quantifying the similarity between two terms by exploiting one or more information sources that can be for example a well-defined ontology (e.g., WordNet [13]). We aim at exploiting semantic similarity for service matchmaking. In this case, the sources of knowledge are the CO and the DO which are exploited for service annotation and discovery. In the literature several approaches have been proposed to assess similarity between concepts in the same or different ontologies. In this work we adopt the approach proposed in [14]. In particular, the semantic similarity between two concepts $Csim(c_1, c_2)$ is computed as follows:

$$Csim(c_1, c_2) = \begin{cases} 3 \cdot IC(msca(c_1, c_2)) - IC(c_1) - IC(c_2) & \text{if } c_1 \neq c_2 \\ 1 & \text{otherwise} \end{cases}$$

where $msca$ is the most specific common abstract between c_1 and c_2 and IC represents the information content of a given concept which quantifies the information a concept expresses in terms of the number of hyponyms (i.e., subconcepts) it has in the ontology. In particular the more hyponyms a concept has the less information it expresses. For further details refer to [14]. Our service matchmaker is based on this metric.

3.5 Measuring semantic similarity between service request and profile

The aim of our matchmaker is to perform finer-grained comparison between a service request and a service profile thus allowing to numerically quantify in what extent a request R fits with a service profile P. The similarity function $RPsim(R, P)$ between P and R is defined inductively on their common structure, as follows. Initially, we consider the semantic similarity between two concepts above presented. Next, we consider an operation $Op^P \in P.\mathcal{O}p$ defined as part of a profile P and an operation $Op^R \in R.\mathcal{O}p$ in a request R, and let $\mathcal{I}^P = Op^P.\mathcal{I}$ and $\mathcal{I}^R = Op^R$ be their respective annotated sets of inputs.

The similarity $Isim(\mathcal{I}^P, \mathcal{I}^R)$ between the two input sets is obtained by comparing each concept associated to an input in the request, $I_i^R \in \mathcal{I}^R$, with each concept associated to an input in the profile, $I_j^P \in \mathcal{I}^P$. The similarity is the sum of the best matches, normalized by the number of inputs in the request. Formally:

$$Isim(\mathcal{I}^P, \mathcal{I}^R) = \frac{\sum_{I_i^R \in \mathcal{I}^R} \max_{I_j^P \in \mathcal{I}^P} Csim(ann(I_i^R), ann(I_j^P))}{|\mathcal{I}^R|}$$

The normalization factor $|\mathcal{I}^R|$ is a measure of specificity of the request. To see why this is important, consider two requests R_1 and R_2, where R_1 is vague and contains a strict subset $\mathcal{I}^{R_1} \subset \mathcal{I}^{R_2}$ of the input concepts of R_2. Suppose that both requests are matched with a profile P, and that the similarity between input terms in $\mathcal{I}^{R_2} \setminus \mathcal{I}^{R_1}$ and the inputs of P is very low. In this case, it is reasonable to expect the vaguer request R_1 to have a better match with P than R_2, because P does not offer any of the additional inputs requested by R_2.

The similarity $Osim(\mathcal{O}^P, \mathcal{O}^R)$ between the annotated sets of outputs for an operation is defined similarly to that of the inputs:

$$Osim(\mathcal{O}^P, \mathcal{O}^R) = \frac{\sum_{O_i^R \in \mathcal{O}^R} \max_{O_j^P \in \mathcal{O}^P} Csim(ann(O_i^R), ann(O_j^P))}{|\mathcal{O}^R|}$$

We also define the similarity between two annotated operations names $n^P = Op^P.n$ and $n^R = Op^R.n$:

$$Nsim(n^P, n^R) = \begin{cases} Csim(ann(n^P), ann(n^R)) & \text{if } n^P \neq \bot \text{ and } n^R \neq \bot \\ 0 \text{ otherwise} \end{cases}$$

We can now proceed to define the similarity between a service operation $Op^R \in R.\mathcal{O}p$ and an operation request $Op^P \in P.\mathcal{O}p$, as follows:

$$\begin{aligned} OPsim(Op^R, Op^P) = \; & \alpha \; Nsim(Op^R.n, Op^P.n) + \\ & \beta \; Isim(Op^R.\mathcal{I}, Op^P.\mathcal{I}) + \\ & \gamma \; Osim(Op^R.\mathcal{O}, Op^P.\mathcal{O}) \end{aligned}$$

The weights α, β, and γ are defined by the provider of the annotations for profile P, and account for the different importance that the provider associates to the various annotations of the profile structure.

Finally, the similarity function $RPsim(R, P)$ between a request and a profile is computed by matching each operation request $Op^R \in R.\mathcal{O}p$ with all profile operations $Op^P \in P.\mathcal{O}p$ and adding up the best matches. As we have done with input and output similarity, the sum is normalized by the number of operations specified in the request:

$$RPsim(R, P) = \frac{\sum_{Op^R \in R.\mathcal{O}p} \max_{Op^P \in P.\mathcal{O}p} OPsim(Op^R, Op^P)}{|R.\mathcal{O}p|}$$

4 The ERGOT architecture

In this section we present the ERGOT system that brings together the discussion done in Section 2.3 on combining DHTs and SONs with the service discovery model and matchmaker presented in Section 3. In describing ERGOT, we distinguish between the service provider and requester perspective.

4.1 Publishing semantic service profiles in ERGOT: the provider perspective

A service provider is allowed to publish service profiles annotated as described in Section 3.1. Service profiles are at the basis of the construction of semantic links among peers with similar interests. In this section we elaborate more on these aspects and in particular how to publish service profiles and how to build semantic links.

4.1.1 On publishing service profiles

In order to provide a deeper insight on the ERGOT service publishing mechanism we suppose the knowledge domain to be bioinformatics and services to be annotated by exploiting the ^{my}Grid ontology [22] although any other ontology can be used. The provider annotates services by exploiting two shared ontologies as described in Section 3. Briefly, a Category Ontology (CO) is used to summarize service functionalities, whereas a Domain Ontology (DO) is exploited to annotate operations with related inputs and outputs. We assume a peer receives both the CO and DO from the peer it contacts in its join phase in the DHT. We also assume CO concepts to be distributed among the participants to the DHT. In practice, each peer will be responsible for a subset of keys obtained by hashing CO concept identifiers (e.g., their path from the root). Note that concepts that are ”close” in the ontology will be possibly dispersed in different nodes as the DHT hashing mechanism does not preserve locality.

Service profiles are published by exploiting the DHT's *put(key, value)* primitive where each *key* is assigned a *value*. In our context, the *key* is one CO concept. Note that a service can be described by more than one CO concept and therefore it will be published several times. It is interesting also observing that CO concepts are used to publish service profiles whereas DO concepts are used in a second phase to perform finer-grained service matchmaking (see Section 3.4). In order this publishing mechanism to work properly a slightly modification to the DHT storing mechanism is required. In fact, we need to keep trace of multiple services annotated to the same concept along with peers that annotated them. In Figure 6 it is shown an example of publication of the *nucleotide alignmen*t service profile (see Figure 5) belonging to the peer P13.

As can be noted, the peer responsible for the CO concept *Global Aligning* to which the service is annotated is *P3*. Hence, *P3* is responsible for keeping track, through the *Semantic Annotation Table*, of all the services annotated by peers to that concept. The second column of the *Semantic Annotation Table* keeps track of the peers that have services annotated to a given concept. Information in the *Semantic Annotation Table* will provide support to the creation of create semantic links as will be discussed in the next section.

4.1.2 On building semantic links

In order to allow building SONs grouping peers with similar content and enhancing the DHT with meaningful semantic links ERGOT exploits the information stored in the *Semantic Annotation Table*. CO concepts, used to publish services in the DHT, can also be viewed as high level semantic descriptions of peer content in terms of services. For instance, in the *Semantic Annotation Table* in Figure 6 both *P13* and *P10* have services annotated to the *Global Aligning* CO concept. As this concept is given a well-defined semantic meaning, one can infer that *P13* and *P10* are semantically-similar to some extent. In ERGOT peers act as rendezvous points for the CO concepts they are responsible for and

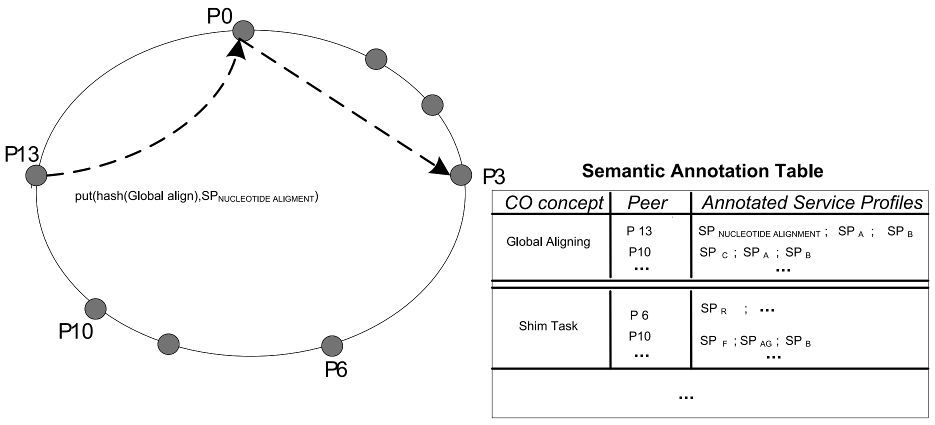

P0
P13
P3
P10
P6
put(hash(Global align),SP_NUCLEOTIDE ALIGMENT)
Semantic Annotation Table
CO concept
Peer
Annotated Service Profiles
Global Aligning
P 13
P10
...
SP NUCLEOTIDE ALIGNMENT ; SP A ; SP B
SP C ; SP A ; SP B
...
Shim Task
P 6
P10
...
SP R ; ...
SP F ; SP AG ; SP B
...
...

Figure 6: An example of service annotation

enable peers with similar interests (in terms of CO concepts) to get in touch with each other. This consideration is at the basis of the construction of semantic links.

In more detail, the process of semantic links creation can be summarized as follows. A peer *p* when publishing semantic service profiles can get, from the peers responsible for the CO concepts to which these profiles are annotated, a list *Ls* of candidate semantic neighbors. This list will be constructed by exploiting the information in the *Semantic Annotation Table*. The number of possible candidates depends on the strategy adopted. A conservative strategy would highlight as possible semantic neighbor each peer that has at least one service annotated to a CO concept to which the peer is annotating its services. A less conservative strategy can fix a threshold in terms of minimum numbers of services so that a peer can be pointed out as a possible semantic neighbor. Again, another strategy could take into account not the number of services but their similarity obtained by exploiting the mechanism defined in Section 3.5. The process of finding a set of candidate semantic neighbors can also be performed in the discovering phase. In this case, a peer by scrutinizing the results of a discovery process can establish semantic links with those that provide interesting results.

The peer *p*, sends to each peer *pi* in *Ls* a request for establishing establish semantic links that are constructed by comparing service profiles of *p* with those of *pi* through the matchmaker described in Section 3.5. The result of this comparison is a number representing the strength of the semantic link. Moreover, a peer *pi* can suggest to *p* a set of other relevant peers with which the latter has already established semantic links. Note that a shallower approach to semantic link creation could not involve the matchmaker and consider as semantic neighbors all the peers in *Ls* or compare them only in terms of CO concepts to which they have annotated their services. Besides, the frequency of semantic neighbor discovery can be set by the user and the number of neighbors can be fixed apriori. At this point each peer in addition to the links imposed by the DHT topology (in our case Chord) will have an additional set of links, stored in a *Semantic Link Table* as shown in Figure 7. Peers that are related by semantic links form a SON. For instance, in Figure 7, *P3* and *P0* are not neighbors in the DHT but they are semantic neighbors since they have services published under the CO concept *Shim Task*. To summarize, the DHT coupled with the CO is the basis for building semantic links that allow the building of SONs. The process of service discovery will be detailed in the next section.

4.2 Exploiting semantic links: the requester perspective

ERGOT offers different types of service discovery with different level of flexibility: (i) semantic based discovery (ii) category based discovery; (iii) combined discovery. In the rest of this section we briefly describe them in more detail.

4.3 Semantic based discovery

Semantic-based service discovery involves the peers belonging to SONs to which the requester belongs. A peer poses a service discovery request by choosing one or more CO concepts and possibly specifying operations with related inputs and outputs. The request is then forwarded to semantic neighbor peers that are relevant to the subject (in terms of CO concepts) of the request. The relevance of a request with a semantic neighbor can be computed in different

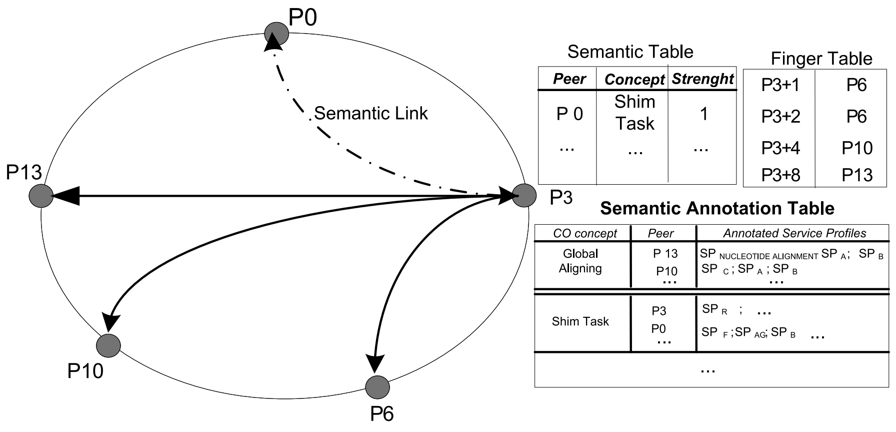

P0
Semantic Link
P13
P3
P10
P6
Semantic Table
Peer | Concept | Strenght
P 0 | Shim Task | 1
... | ... | ...
Finger Table
P3+1 | P6
P3+2 | P6
P3+4 | P10
P3+8 | P13
Semantic Annotation Table
CO concept | Peer | Annotated Service Profiles
Global Aligning | P 13 P10 ... | SP NUCLEOTIDE ALIGNMENT SP A; SP B SP C ; SP A ; SP B ...
Shim Task | P3 P0 ... | SP R ; ... SP F ;SP AG; SP B ...
...

Figure 7: Construction of semantic links

ways. For instance, one selection criterion could be the strength of the semantic links whereas another one could measure the semantic similarity, through the metrics described in Section 3, between the request and concepts in the *Semantic Link Table*.

When a peer receives a request it tries to locally fulfill it and then looks in its local *Semantic Link Table* for peers whose interests, in terms of CO concepts, are semantically close to the concepts in the request. The request is then forwarded to the relevant peers. Note that the search in the SON is not "exact" as in the case of DHT lookup but involves the similarity metric and possibly the matchmaker described in Section 3 (depending on how detailed is the request).

4.4 Category based discovery

It could be that a peer that poses a service request has no, or not enough, semantically related peers in its *Semantic Link Table* to send the request to. This could for example happen when the peer does not share any service profile on the network, resulting in no semantic links creation. When this happens, the request has to be resolved exclusively on the DHT. Even in this case the request comprises one or more CO concepts and possibly some operations with related inputs and outputs. The cost of routing such kind of request in terms of hops is $O(klogN)$ where k is the number of CO concepts in the request. Note that this approach relies on "exact" matching as it involves only the DHT. An approach similar to this has been adopted by the SPiDer system [15]. However, differently from SPiDer, ERGOT offers a more flexible mechanism since it can possibly distinguish relevant services by performing, on the set of results retrieved through exact lookup, finer-grained matchmaking.

4.4.1 Combined discovery

A peer can also pose a combined search discovery request. In this case, the query will involve both SONs and the DHT thus combining the flexibility of SONs with exact and efficient lookup of the DHT.

4.5 Enhancing DHTs with semantic shortcuts

In ERGOT, semantic links can be viewed as (possibly) additional links to those imposed by the DHT topology (Chord in our case). In particular, semantic neighbors are also peers in the DHT terminology each of which is given an *id*. As an example let's consider the scenario depicted in Figure 8.

P7 stores in the finger table Chord neighbors and in the *Semantic Link Table* its semantic neighbors discovered as detailed in Section 4.1.2. Here can be noted that *P7* in addition to its Chord neighbors has additional semantic neighbors, that is, *P30* and *P24*. To show what benefits can bring semantic links to the traditional DHT routing, suppose *P7* poses a search on the DHT for a key with *id* 31. According to the Chord protocol, the successor of this key is *P0*. If we consider the standard DHT routing algorithm, this request from *P7* to reach *P0* will involve the following hops: *P21*, *P29*, *P30*, *P0*. Now, by noting *P30* is a semantic neighbor of *P7* we can save some hops. In fact, in this case the request will be routed though the path *P30*, *P0*. Therefore, when performing exact lookup (i.e., looking for a

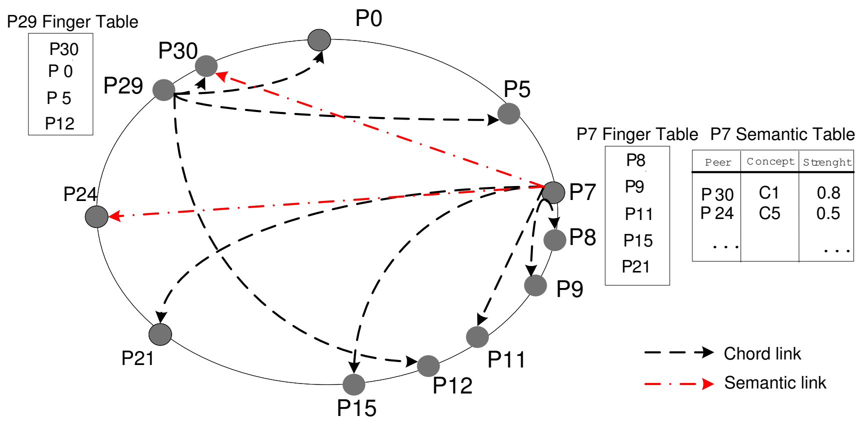


Figure 8: A semantically-enhanced DHT

particular key) in the DHT it is possible that a semantic neighbor is given an *id* which is closer to the searched key (in the DHT terminology) than every node in the finger table. This can help to reduce the number of hops needed to reach the node responsible of a given key. In addition, as done in semantic-based discover, transitivity can also be allowed between semantic links, that is, it would be possible to look at semantic neighbors of a peer's semantic neighbors.

5 Related Work and Discussion

The problem of service discovery has been addressed from different, not necessarily disjoint, perspectives thus giving birth to several strands of research. Some of these focus on mitigating centralization by adopting decentralized architectures (e.g., structured, unstructured, hybrid). Others focus on defining semantically-rich data models such as OWL-S (`http://www.w3.org/Submission/OWL-S/`), SAWSDL (`http://www.w3.org/2002/ws/sawsdl/`) and WSMO (`http://www.wsmo.org/`) to describe and discover services. Again, others focus on performing efficient discovery by adopting, for instance, Information Retrieval to exploit as much as possible information encoded in service descriptions. We investigated the state of the art of service discovery initiatives and identified a list of desiderata a service discovery mechanism should fulfill:

- Decentralization: as the number of services grows up, decentralization becomes a mandatory requirement to avoid bottlenecks of centralized service repositories such as UDDI.
- Semantic rich representation: the more a service description is semantically characterized, the more it is possible by a "machine" to understand if it satisfies a user need. This way, more complex tasks such as service composition can be meaningfully carried out.
- Semantic based discovery: similar to service representation, semantic-based service discovery allows better expressing requests and giving them a precise meaning. In particular, the similarity between a request and a service profile can be performed on a semantic basis.
- Ranking mechanism: ranking is important to present results to a user and allows her to distinguish among a multitude of services all claiming to fulfill her needs.

In the following table, on the basis of the list of desiderata we identified, a set of service discovery approaches are compared. In particular in our subsequent analysis we employ as distinctive factor the network architecture they adopt.

5.1 Centralized approaches

In this section, we examine centralized architecture for web service discovery. The ROSSE system [12] exploits Rough Set Theory to perform service discovery in the Grid. In particular, ROSSE identifies some dependences among service properties and is able to compute the Lower and Upper approximation of Grid services that match a user request. It also

	Network Architecture	**Semantic Support**	**Ranking Mechanism**	**Specific techniques**
	Centralized			
ROSSE	Registry-based	No	Yes (numerical)	Rough set theory
Woogle	Registry-based	No	Yes	Ad-hoc clustering technique
COMPAT	Registry-based	Yes	Yes (numerical)	Semantic matching
Matchmaker	Registry-based	Yes	Yes (logic relations)	Reasoning
	Decentralized			
DUDE	DHT	No	No	Prefix Match of service names
Schmidts and ParaShar	DHT	No	No	Hilbert Space Filling curves
Meteor-S	JXTA	Yes	-	-
Hypercube	Ad-hoc topology	Yes	-	-
SPiDeR	DHT-Super peer based	Yes	Yes (based on Qos)	Behavior-based search
Vu et al.	DHT	Yes	Yes (based on Qos)	Ontology partitioning. Bloom filters
ATLAS	DHT	RDF	No	RDF-based
WSPDS	SON	Yes	No	Similarity between peers
ERGOT	DHT+SON	Yes	Yes (numerical)	Combines DHTs and SONs

Table 2: Comparision among service discovery architectures

quantifies the similarity between relevant properties by converting them to numeric values on the basis of a predefined set of possible semantic relations. ROSSE also takes into account QoS by defining a set of heuristics each of which will be weighted toward a final similarity ranking. The system also supports subsumption reasoning by exploiting an ad-hoc reasoner.

Woogle [5] is a system meant to perform efficient service discovery by exploiting textual information encoded in web service descriptions. In particular, authors devised a novel clustering algorithm which groups parameter names into semantically meaningful concepts and performs similarity computations by an Information Retrieval approach where service descriptions are viewed as vectors of terms. The COMPAT system [2] exploits an ad-hoc ontology framework based on Description Logics and a thesaurus to enable semantic-similarity based service discovery. In particular, service profiles are defined by exploiting both a service ontology, which allows to group services with similar features, and a domain ontology used to annotate operation names, input and output. The system support ranking of results by computing, through the thesaurus, the semantic similarity between a request and a service profile.

The Matchmaker [10] has been one of the first semantic matchmakers for web services. It allows, through reasoning, to define different semantic relations (e.g., exact, subsume, plug-in) between a user request and a service profile.

ERGOT differs from ROSSE and Woogle since it exploits ontologies to annotate services. It shares some semantic features with COMPACT as it performs semantic based service matchmaking. However, differently from all these approaches it is based on a fully decentralized architecture.

5.2 Decentralized approaches

To cope with the pitfalls of centralized service discovery architectures several decentralized initiatives have been proposed. However, while being decentralized these can adopt different approaches to reach decentralization.

The DUDE system [1] extends the UDDI centralized service discovery mechanism by allowing multiple registries to form a federation with a DHT as a rendezvous point. Here, service information (on the basis of the service name) is distributed among the participants; however the DHT querying mechanism limits the scope of the queries only to relevant registries. To support prefix based querying a service name is hashed different times, one for each prefix, and published on the DHT.

In [17] a DHT based web service discovery system is proposed. Here a service description is viewed as a set of points in a multidimensional space identified by the possible keywords found in service descriptions. In order to map the multidimensional space to DHT keys, authors exploit Hilbert Space Filling Curves (HSFCs) which ensure that the locality in the multidimensional space will be preserved after the reduction. However, that jeopardizes the hashing mechanism of the original DHT thus leading to load imbalance. In practice, with the dimensional reduction, data elements are not uniformly distributed in the index space, i.e., certain keywords can be more popular and hence the associated index subspace can be more populated. To cope with this issue authors propose two load balancing mechanisms: Load Balancing at Node Join and Load Balancing at Runtime. The system supports wildcard queries and partial keyword queries.

Both DUDE and the system described in [17], differently from ERGOT, while coping with the scalability and fault tolerance issues, neither provide semantic characterization of web services nor result ranking. Besides, the approach described in [17] uses Hilbert Space Filling Curves as a mechanism to ensure locality preserving hashing. This mechanism, on the one side ensures that service descriptions which are similar in the space of keywords describing them will be mapped into similar keys to be stored in the DHT. But, on the other side destroys the nice properties of consistent hashing thus not ensuring that keys will be distributed evenly among nodes. To cope with this issue an ad-hoc load balancing technique has been devised by authors as discussed in [17]. ERGOT avoids this problem by combining DHTs and SONs, that is, efficient lookup and semantic based service discovery.

The Meteor-S system [11] support semantic based organization of web services in a federation of registries. This system, developed in JXTA, is based on an unstructured P2P network and is mainly meant to organize service publications by identifying the most suitable registry to host a service description.

The WSPDS system [9] aims at constructing an overlay network of peers (here called servents, that is, server and client at the same time) by comparing their data content (web service descriptions). In particular, nodes create links by comparing the inputs and the outputs of their services by exploiting the matchmaker described in [10]. Besides, the similarity between a query and the peers to whom forward it is computed by the same matchmaker. Each WSDL description has associated a WSIL which contains a pointer to a WSDL-S exploited to semantic annotate a service. The matching between requests and services is enhanced by annotating both with globally shared concepts.

The Hypercube system [16] adopts an ad-hoc network topology. Here, a globally known ontology is exploited to determine the organization of peers.

The Spider system [15] organizes participants into a super peer (SP) based P2P structured network in order to take into account the different computational power of nodes. Each peer is connected to a SP with which it interacts for performing operations of service advertising and discovery. In particular, service discovery is performed by using three different techniques. A keyword based approach, a category-based approach and a behavioral approach. The system supports a reputation component to perform QoS ratings of web services.

The system proposed in [21] combines ontologies, DHTs and a reputation mechanism based on trusted agents to perform service discovery. One of its key features is the partitioning of a shared ontology, with which describing and querying for services, in concept groups. Each concept group is summarized by a Bloom filter to enable quick concept-membership checking. Hence, a service is described as a set of unordered concepts each of which is represented by the Bloom filter to which each concept belongs. To map service descriptions in the underlying DHT a special hash function is applied to the concatenation of all the keys the description. A QoS component allows rating the quality of a web service and is used to rank results. Even in this case, the use of a particular hash function that does not guarantee consistent hashing causes that keys will not be distributed evenly among nodes.

The ATLAS system [8] has been designed as a decentralized mechanism for resource discovery in S-OGSA [3]. ATLAS adopts a DHT-based architecture to publish and discovery information about Grid resources on the form of RDF triples. The system allows to pose two types of queries: (i) on-time queries which will be resolved on the network on-the-fly and; (ii) publish/subscribe queries which are continuous queries in the sense that if a resource specified in the query does not exist, the request will be stored in the network and when the resource is published the initial requester is notified. ATLAS allows resolving conjunctive queries expressed in a logic language based on triple patterns.

ERGOT shares some characteristics with the above-mentioned systems. In particular, it provides semantic characterization of services through ontologies. ERGOT, differently from other systems offering a similar feature (e.g., [21, 15, 9]) provides two levels of annotation. A category ontology is used to categorize services and guiding their publishing on the DHT, while a domain ontology is used to semantically characterize operation names with related inputs and outputs. Overall, the main differences w.r.t these systems can be summarized as follows: (i) ERGOT combines DTHs and SONs. To the best of our knowledge ERGOT is the only system combining these two architectures for the purpose of service discovery; (ii) ERGOT adopts a ranking mechanism based on semantic similarity. In particular, after locating services that match a user need by scrutinizing their categorization, it performs fine-grainer matchmaking by computing the semantic similarity between operation names, inputs and outputs in a request and those in a service description. This provides the user with a more immediate interpretation of results since most of current approaches either do not perform ranking of results or only provide ranking based on QoS (e.g., [21]).

References

[1] Sujata Banerjee, Sujoy Basu, Shishir Garg, Sukesh Garg, Sung-Ju Lee, Pramila Mullan, and Puneet Sharma. Scalable Grid service discovery based on uddi. In *MGC '05: Proceedings of the 3rd international workshop on Middleware for Grid computing*, pages 1–6, New York, NY, USA, 2005. ACM.

[2] Devis Bianchini, Valeria Antonellis, and Michele Melchiori. Flexible semantic-based service matchmaking and discovery. *World Wide Web*, 11(2):227–251, 2008.

[3] Oscar Corcho, Pinar Alper, Ioannis Kotsiopoulos, Paolo Missier, Sean Bechhofer, and Carole A. Goble. An overview of s-ogsa: A reference semantic Grid architecture. *J. Web Sem.*, 4(2):102–115, 2006.

[4] Arturo Crespo and Hector Garcia-Molina. Semantic overlay networks for p2p systems. In *AP2PC*, pages 1–13, 2004.

[5] Xin Dong, Alon Y. Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Simlarity search for web services. In *VLDB*, pages 372–383, 2004.

[6] T. R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

[7] A. Iamnitchi and D. Talia. P2p computing and interaction with Grids. *Future Generation Comp. Syst.*, 21(3):331–332, 2005.

[8] Z. Kaoudi, M. Koubarakis, K. Kyzirakos, M. Magiridou, I. Miliaraki, and A. Papadakis-Pesaresi. Publishing, discovering and updating semantic Grid resources using dhts. In *CoreGRID Workshop on Grid Programming Model, Grid and P2P Systems Architecture, Grid Systems, Tools and Environments.*, 2007.

[9] Farnoush Banaei Kashani, Ching-Chien Chen, and Cyrus Shahabi. Wspds: Web services peer-to-peer discovery service. In *International Conference on Internet Computing*, pages 733–743, 2004.

[10] Takahiro Kawamura, Jacques-Albert De Blasio, Tetsuo Hasegawa, Massimo Paolucci, and Katia P. Sycara. Public deployment of semantic service matchmaker with uddi business registry. In *International Semantic Web Conference*, pages 752–766, 2004.

[11] Verma Kunal, Kaarthik Sivashanmugam, Amit Sheth, Abhijit Patil, Swapna Oundhakar, and John Miller. Meteor-s wsdi: A scalable p2p infrastructure of registries for semantic publication and discovery of web services. *Inf. Technol. and Management*, 6(1):17–39, 2005.

[12] Maozhen Li, Bin Yu, Omer Rana, and Zidong Wang. Grid service discovery with rough sets. *IEEE Transactions on Knowledge and Data Engineering*, 20(6):851–862, 2008.

[13] G. Miller. Wordnet an on-line lexical database. *International Journal of Lexicography*, 3(4):235–312, 1990.

[14] Giuseppe Pirro' and Nuno Seco. Design, implementation and evaluation of a new semantic similarity metric combining features and intrinsic information content. In *ODBASE 2008*, pages 1270–1287, 2008.

[15] Ozgur D. Sahin, Cagdas Evren Gerede, Divyakant Agrawal, Amr El Abbadi, Oscar H. Ibarra, and Jianwen Su. Spider: P2p-based web service discovery. In *ICSOC*, pages 157–169, 2005.

[16] Mario Schlosser, Sintek Michael, Decker Stefan, and Nejdl Wolfgang. A scalable and ontology-based p2p infrastructure for semantic web services. In *P2P '02: Proceedings of the Second International Conference on Peer-to-Peer Computing*, page 104, Washington, DC, USA, 2002. IEEE Computer Society.

[17] Cristina Schmidt and Manish Parashar. A peer-to-peer approach to web service discovery. *World Wide Web*, 7(2):211–229, 2004.

[18] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.

[19] D. Talia and P. Trunfio. Toward a synergy between p2p and Grids. *IEEE Internet Computing*, 7(4):94–95, 2003.

[20] Paolo Trunfio, Domenico Talia, H. Papadakis, P. Fragopoulou, Matteo Mordacchini, M. Pennanen, Konstantin Popov, Vladimir Vlassov, and Seif Haridi. Peer-to-peer resource discovery in Grids: Models and systems. *Future Generation Comp. Syst.*, 23(7):864–878, 2007.

[21] Le-Hung Vu, Manfred Hauswirth, and Karl Aberer. Towards p2p-based semantic web service discovery with qos support. In *BPS 2005*, 2005.

[22] K. Wolstencroft, P. Alper, D. Hull, C. Wroe, PW. Lord, RD Stevens, and CA Goble. The (my)Grid ontology: bioinformatics service discovery. *Int J Bioinform Res Appl.*, 3(3):303–325, 2007.