2009 Ninth International Conference on Intelligent Systems Design and Applications

Combining DHTs and SONs for Semantic-Based Service Discovery

Giuseppe Pirró[1], Paolo Missier[2], Paolo Trunfio[1], Domenico Talia[1,3], Gabriele Falace[1], Carole Goble[2]

[1] DEIS, University of Calabria, Rende (CS), Italy

{gpirro,trunfio}@deis.unical.it

[2] School of Computer Science, University of Manchester, UK

{pmissier,carole}@cs.man.ac.uk

[3] ICAR-CNR, Rende (CS), Italy

talia@icar.cnr.it

Abstract

The soaring number of available online services calls for distributed architectures to promote scalability, fault-tolerance and semantics; to provide meaningful descriptions of services; and to 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" match. This paper presents the ERGOT system combining DHTs and SONs to enable distributed and semantic-based service discovery. A preliminary evaluation of the system performance shows the suitability of the approach both in terms of recall and number of messages.

1 Introduction

The Service Oriented Architecture (SOA) paradigm is increasingly enjoying success as a viable model for the modular composition and reuse of third party software components on a large scale. One of the stumbling blocks towards its adoption on a Web scale, however, is the difficulty for potential service users to discover new services of interest, particularly when such services are independently deployed by a broad array of providers, on open, large-scale networks. In this scenario, the availability of effective service registries to facilitate service discovery becomes a key to the success of the SOA paradigm.

Much progress has indeed been made in the recent past towards de-centralized and scalable federations of service registries, as well as on semantic-based service annotation to increase search precision. Most proposals for distributed and federated registries rely either on unstructured Peer-to-Peer (P2P) architectures like Gnutella [1], or Distributed Hash Tables (DHTs) such as Chord [12].

At the same time, current approaches to semantics-based service discovery exploit a number of techniques, ranging from Information Retrieval [4] to Rough Set theory [7], and to logic-based reasoning to infer semantic relations (e.g., exact, subsume) between a user request and a service profile [6]. Semantically-rich service description annotation models, such as OWL-S [2], SAWSDL [3] and WSMO [4] provide the support for machine-processable annotations.

We observe that both DHTs and unstructured P2P systems present some drawbacks. DHT-based search is limited to exact terms, (e.g., the service name) and thus does not fit well with semantic similarity search models, where services are described using multiple annotations. Mapping the semantic search paradigm on unstructured P2P networks, on the other hand, can be costly in terms of network routing.

The main goal of this paper is to explore synergies between these two areas, i.e., a combination between distributed service registries and semantic service discovery. Our hypothesis is that a P2P-based federation of registries can benefit from semantic annotations of service descriptions, both in terms of search recall and of effective use of network resources, i.e., in terms of number of messages exchanged by peers during service publication and service discovery.

Our approach builds upon three main pillars: (a) a standard DHT protocol, (b) Semantic Overlay Networks (SONs) [3], and (c) semantic annotations of service descriptions using SAWSDL. SONs, are overlays designed to connect peers that are *semantically related*, in the sense that their respective content - service descriptions in our case, is

[1] http://rfc-gnutella.sourceforge.net
[2] http://www.w3.org/Submission/OWL-S
[3] http://www.w3.org/2002/ws/sawsdl
[4] http://www.wsmo.org

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.77

annotated using concepts taken from a shared ontology.

The contribution of this paper is twofold. First, we present a system called ERGOT (Efficient Routing Grounded On Taxonomy), which allows to build a SON incrementally, and at no additional cost, as a byproduct of peer interactions that occur naturally in a DHT during advertising of a new service description, as well as during search. Second, we define a measure of semantic similarity amongst service descriptions for enabling semantic-driven ranking to be exploited both in the process of construction of the SON and during service discovery.

Our preliminary experiments, presented in Section 4, confirm that this is a viable approach. In particular we will show that ERGOT obtains significant values of recall in different configurations with limited network traffic.

The rest of the paper is organized as follows. In the next section we provide a background on DHTs and SONs. The ERGOT system is presented in Section 3, followed by our experiments (Section 4) and some related work (Section 5). Section 6 concludes the paper.

2 Background on DHTs and SONs

Distributed Hash Tables (DHTs) and Semantic Overlay Networks (SONs) are the two technologies at the core of our proposal. We provide a brief summary of them here.

DHTs have gained recognition as a prominent network paradigm for data distribution and indexing, due to their scalability properties and efficiency in retrieving content. For the sake of explanation we are going to use the well-known Chord DHT model [12] as a reference. Chord assigns a key to each data item. Each peer can be used both to publish new data items to the network, using a *put(key, value)* primitive, and to submit search requests by key ($get(key)$). The hashing and routing mechanism employed by Chord makes exact, key-based queries efficient. In Chord both network peers *and* data items are assigned an *m-bit* hash value (a peer's *id* can be computed by hashing the peer's IP address). With this provision, the network has a ring topology consisting of at most 2^m nodes. Chord adopts a simple rule to assign values to nodes: a *put(key, value)* operation assigns *value* to the peer whose *id* is the *successor* of *key*. This is illustrated in the left part of Fig. 1, where *key 2* (i.e., *K2*) is assigned to peer P3. A *get(key)* operation can be initiated by any peer, and is routed around the ring with the help of a *finger table* associated to each peer. The finger table for a peer contains the Chord *ids* of the peers' neighbors. This guarantees that any query is answered in $O(logN)$ hops [12]. In Fig. 1, the request posed by *P3* for the key 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*).

Neighbors are peers located on the ring at exponentially increasing distance from a given peer. The finger table of *P3* maintains information about *P3*'s neighbors, that is, *P6*, *P10* and *P13*. DHTs only support key-based, exact queries, a serious limitation for semantic-based searches where one is interested in retrieving content that are "semantically similar" to a concept provided in the search.

SONs provide a first step in this direction, although they do not directly address the problem [3]. SON is a paradigm for organizing peers and enhancing content-based search. The main idea is that, by clustering the peers according to the semantic similarity of their content, the clusters can then be exploited to speed up query routing while providing good recall. In particular, annotating the values with concepts drawn from a common taxonomy provides the necessary underpinning for partitioning an otherwise unstructured P2P network into SONs. An example of SON is shown in the right part of Fig. 1. *Semantic links* among peers are constructed according to a criterion of semantic similarity. In [3] the effectiveness of the idea as compared to the Gnutella flooding-based approach is discussed.

3 Combining DHTs and SONs

Although DHTs and SONs have been (separately) exploited in recent service discovery initiatives (e.g., [10, 13, 5]), no attempt has been made, to the best of our knowledge, to combine the two approaches.

ERGOT is a hybrid system that relies on semantic annotations of services to provide a scalable, efficient and effective service discovery mechanism by combining DHTs and SONs. In this section we begin by presenting a semantic annotation mechanism and semantic similarity function used to match semantic service profiles stored by peers, with service requests that specify ontology concepts corresponding to the various structural elements of a service interface. We then present an algorithm for publishing new semantic service profiles on the network and for establishing semantics links amongst peers based on those profiles, and finally describe a service discovery strategy based on semantic links.

3.1 Service annotation

The ability to match service descriptions to user searches is at the cornerstone of any service discovery model. Purely syntactic approaches, based on service name similarity, have been extended in several directions, using either ontology-based semantics [6] or Information Retrieval techniques [4]. On the other hand, when the matching is performed using logic-based reasoning, the match status can itself be described using a simple classification model, e.g., *exact*, *plugin*, *subsume*, *fail*. A quantitative, numerical assessment of these concepts, however, has proven difficult

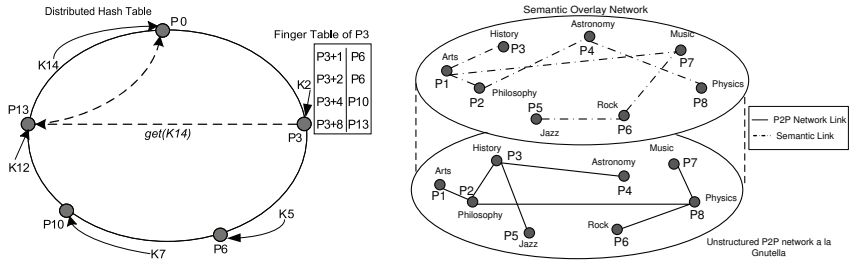

Figure 1. A DHT (left) and a SON (right)

[2] for classes like *subsume* or *plugin*, making it hard to accurately rank the matches. While result ranking has sometimes been addressed in a few cases [10, 13], this is limited to Quality of Service (QoS) indicators, and does not extend to the fine-grain semantic description of the service itself.

ERGOT introduces a similarity measure between service descriptions and requests. It exploits both coarse-grain service functionality descriptions and at a finer level, when available, annotations on individual elements of a service signature. We use two sources of knowledge for these annotations: a high-level *Category Ontology* (CO) for service description and a *Domain Ontology* (DO) that accounts for semantic data types description.

In this paper we do not introduce new annotation mechanism, and rely instead on existing annotations. For the sake of our implementation we assume that service annotations follow the SAWSDL model, endorsed by W3C. Other annotation frameworks are described in [8].

3.2 Preliminary definitions

In ERGOT, matching of service descriptions is based on service profiles and service requests. A service profile $P = \langle sn, \mathcal{O} \rangle$ consists of a service name sn and a set $\mathcal{O}$ of operations. An operation $O \in \mathcal{O}$ has a name $O.n$ and a set $O.IN$ and $O.OUT$ of inputs and outputs: $O = \langle n, IN, OUT \rangle$. We use a dot notation to refer to elements of a service profile's hierarchical structure. For example, $P.sn$ is the service name, and $P.\mathcal{O}_i.IN_j$ is the j-th input of the i-th operation. Each of the sn, n, and each $I \in IN$, $O \in OUT$ can be annotated with ontology concepts. We write $ann(x)$ to denote the concept that annotates a generic element x. As mentioned, we use one or more CO concepts for describing the overall function of a service, and DO concepts to annotate elements of its internal structure. Thus, we assume that $ann(P.sn) \subseteq$ CO and $ann(P.\mathcal{O}_i.IN_j) \subseteq$ DO.

A service request $R = \langle \mathcal{C}, \mathcal{O} \rangle$ matches the structure of a service profile, but ontology concepts representing user requirements for matching services appear in lieu of service component annotations. In particular, concepts $\mathcal{C}$ from the CO express service functionality requirements, while the $\mathcal{O}$ structure can be used to specify semantic types for operations inputs and output, as well as specific operations names. The level of precision of a request may vary, from simply a set $\mathcal{C}$ with no constraint on the service operations, to a request for any operation that satisfies a given set of input or output types, to a full set of requirements that include named operations with given input and output types.

A request R is evaluated against a collection of service profiles $\mathcal{P} = \{P_1 \dots P_n\}$, by matching each of the concepts found in R's structure with the corresponding annotations in each profile $P \in \mathcal{P}$, using a semantic similarity function $RPsim(R, P)$. The function is defined inductively on P's structure, using the requirement specifications found in R.

3.3 On matching service profiles

As a common starting point for the definition of $RPsim(R, P)$, we employ a baseline function, first proposed in earlier work by some of the authors [9], to measure the similarity between two concepts c_1, c_2 that are part of the same ontology. Let $msca$ be the most specific common ancestor of c_1, c_2, and $IC(c)$ the *information content* of a concept c [9]. The similarity measure is defined as:

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

The definition of $RPsim(R, P)$ is inductive on P's structure, and it accounts for the varying specificity of the request R, as anticipated. Initially, let us consider a fully specified request for a single operation type, i.e., $R = \langle \mathcal{C}, \mathcal{O} \rangle$ where $\mathcal{O} = \{O\}$, and a CO concept c as well as a set of input and output type requirements IN, OUT associated to O, i.e., $O = \langle c, IN, OUT \rangle$.

We match this request to a profile P by matching the annotations of each operation $P.O_i$ to $R.O$ and taking the similarity value of the best-matching operation. In turn, matching one operation structure $P.O_i$ requires matching the set $P.O_i.IN$ (resp. $P.O_i.OUT$) to set $R.IN$ (resp., $R.OUT$), and the operation names $P.O_i.n$ to $R.n$. The similarity of input semantic types, $Tsim(P.O_i.IN, R.O.IN)$, is the sum of the best matches amongst all possible pairs $(ann(x), y)$

with $x \in P.O_i.IN$, $y \in R.O.IN$:

$$Tsim(P.O_i.IN, R.O.IN) = \sum_{y \in R.O.IN} \max_{x \in P.O_i.IN} Csim(ann(x), y)$$

The same function applies to output types. This function rewards services that contain an operation whose input (resp. output) types best match the concepts in $R.O.IN$ (resp. $R.O.OUT$). Matching of $P.O_i.n$ to $R.O.c$ is a straightforward application of the $Csim$ function:

$$Nsim(P.O_i.n, R.O.c) = Csim(ann(P.O_i.n), R.O.c)$$

The overall similarity between $P.O_i$ and $R.O$ is the weighted sum of the three components just defined:

$$\begin{aligned} OPsim(P.O_i, R.O) = \; & \alpha \; Nsim(P.O_i.n, R.O.c) + \\ & \beta \; Tsim(P.O_i.IN, R.O.IN) + \\ & \gamma \; Tsim(P.O_i.OUT, R.O.OUT) \end{aligned}$$

This last definition accounts for the possibility that the user request contains incomplete requirements, as in this case where the corresponding component is simply zero (a minimal request may only include a concept for the operation name, or for the input/output semantic data types).

Finally, the overall similarity between P and $R.O$, i.e., when R contains requirements for one single operation, is simply the best similarity value over all operations in $P.\mathcal{O}$:

$$RPsim(R.O, P) = \max_{O_i \in P.\mathcal{O}} OPsim(R.O, O_i)$$

We extend this function to the case where R specifies requirements for multiple operations $R.\mathcal{O}$, simply by adding up the similarities values that result from the best matches between each $O_j \in R.\mathcal{O}$ and each $O_i \in P.\mathcal{O}$:

$$RPsim(R, P) = \sum_{R.O_j \in R.\mathcal{O}} \max_{O_i \in P.\mathcal{O}} OPsim(O_j, O_i)$$

3.4 Publishing semantic service profiles

Consider a service profile $P = \langle sn, \mathcal{O} \rangle$, and let $C_P = ann(P.sn) \subseteq$ CO be the set of its service-level annotations.

P is published by invoking the standard DHT primitive $put(key, value)$, using each concept $c_P \in C_P$ as the key [5]. Thus, publishing requests are of the form $put(c_P, P)$. In addition, however, we enhance the DHT behavior to support semantic links among peers, and we then exploit these links to facilitate service discovery. In this section we explain the publication algorithm and describe the process of building semantic links.

For each published profile P we distinguish between its *hosting peer* $hp(P)$, i.e., the peer that requests the publishing of P; and the *responsible peer* $rp(P)$ to which P is assigned according to the standard DHT routing algorithm. To recall, in general a DHT defines a mapping function $p = publish(k)$ that maps a key k to a peer p (since in our experiment we have used Chord, we assume this function to be $publish(k) = successor(k)$). This means that, since we are using CO concepts as keys, each peer is responsible for a set of concepts $c \in$ CO:

$$concepts(p) = \{c \in \text{CO} | publish(c) = p\}$$

3.5 Building semantic links

In ERGOT, each peer maintains a *Semantic Annotations Table* (SAT), in addition to its finger table. If $c \in concepts(p)$, then p's SAT records all peers that host profiles that are annotated using c, i.e.,

$$SAT(p) = \{hp(P) | c \in ann(P.sn) \cap concepts(p)\}$$

In practise, each peer now keeps a record of peers that publish content with the specific semantic annotations they are responsible for.

We use SATs to establish semantic links amongst peers. Suppose that hosting peer $p = hp(P)$ publishes P using $put(c_0, P)$. When the request is routed to $p' = rp(P)$ (through the usual finger table mechanism), p' now responds by sending to p the entries of its SAT that correspond to the concept c_0 used as key. We call the peers in these entries the *semantic neighbors* $sn(p, c)$ of p:

$$\begin{aligned} sn(p, c) = \{ & hp(P) \; \forall \; P \text{ s.t.} \\ & c \in ann(P.sn) \wedge c \in concepts(p')\} \subseteq SAT(p') \end{aligned}$$

Intuitively, the peers in $sn(p, c)$ host services that have been annotated using the same concept c used to describe P. The SAT for p is populated incrementally each time it publishes a new service profile, with no additional routing as we are piggybacking on the original DHT routing strategy (only a new message back from $rp(P)$ to $hp(P)$ is needed).

A peer p can establish semantic links $p \rightarrow p'$ with each neighbor $p' \in sn(p)$. Not all of the peers in $sn(p, c)$ are interesting neighbors, however. It may be the case that one of these peers hosts many services, only one of which is annotated using c. As the semantic links are used to create a SON, which is in turn then used for semantic similarity search, installing a link to such a neighbor would actually be misleading for many of the searches. We can deal with this variability in two main ways. Firstly, we can limit the extent of the set $sn(p, c)$ received by p, for example by using a predefined threshold on the minimal number of services that

[5] Note that P gets published multiple times, once for each concept in the annotation.

each $p' \in sn(p)$ annotates using c. And secondly, we can associate a measure of *strength* to each new semantic link $p \rightarrow p'$, based on the semantic similarity function between the services hosted by p and p', as described in Section 3.2.

3.6 Query processing

The ERGOT architecture discussed so far enables service discovery based either on the SON, or on the underlying plain DHT topology, or on a combination of both. In each case, we assume that a service query has the form $SQ := \langle id_{SQ}, id_p, R, TTL \rangle$, where id_{SQ} is a unique *id* assigned to the query, id_p is the *id* of the peer that generated the query, R is a request of the form $R = \langle \mathcal{C}, \mathcal{O} \rangle$ and TTL is the Time-to-Live, which represents the maximum number of hops that the query can traverse. Finally, we assume that service profiles can be ranked according to their similarity to the request, as explained in Section 3.2. The semantic search strategy is straightforward: a peer p that receives a request begins by matching it within its local service profiles; it then forwards the request over its semantic links. Alternatively, the peer may choose to use the underlying DHT to route the request. This may happen when there are no semantic neighbors that satisfy the strength criteria, for example. In this case, the request becomes a collection of standard $get(c)$ operations, with $c \in R.\mathcal{C} \subseteq$ CO. The routing in this case involves $O(k \log N)$ hops where k is the number of concepts in the request. This approach relies on exact matching of the concepts, and is similar to [10]. However, in ERGOT, upon reaching the peer responsible for the concept a similarity-based search can additionally be performed on that peer's service profiles.

4 Experiments

This section discusses a preliminary evaluation of the system focused on annotation of services to Category Ontology (CO) concepts while aspects related to annotations to Domain Ontology (DO) concepts and fine-grain service matchmaking will be addressed in a future work. We exploit Chord as underlying DHT although any other implementation can be used. Our aim is to investigate the behavior of the system in different network configurations by evaluating its performance in terms of *recall* (i.e., the number of found services vs. the number of existing relevant services) and the *number of messages*. These two indicators are useful to investigate the efficiency of ERGOT in finding relevant results and network traffic generated in a service lookup.

In the simulator, we adopt a CO with 2000 concepts distributed over a depth of 10. This structure of the CO has been chosen to have characteristics similar to existing service categorization taxonomies such as NAICS [6] adopted by UDDI. We set up a networks of 1000 peers with a number of services varying from 10000 to 50000. Each service is given a number of annotations ranging from 1 to 6 and is assigned to a randomly chosen hosting peer. We set the forward threshold to 0.51 and the TTL to 4. We computed the set of relevant services for each query before publishing the services over the DHT. After the service publishing, we pose the same query and resolve it following semantic links.

Fig. 2 shows the results in terms of recall and number of messages. The horizontal axis represents the number of hops, that is, the depth to which the original request has been spread over the SON. After one hop, the recall increases from 0.032 with 10000 services to 0.222 with 50000 services, as shown in the left graph of Fig. 2. Similarly, after two hops the recall passes from 0.131 to 0.361, and after three hops it increases from 0.771 to 0.871. Hence, for any fixed number of services the recall increases with the number of hops since the request gets more propagated over the SON. On the other hand, fixing the number of hops, the recall increases with the number of services. This is due to the fact that a higher number of services implies a higher number of semantic links among peers. As shown in the right graph of Fig. 2, also the number of messages follows an increasing trend depending both from the number of hops and the number of services. For instance, in the configurations with 10000 and 50000 services, the number of messages passes from 35 after one hop to 315 at the third hop, and from 85 to 880, respectively. Therefore, the number of messages increases with the number of services, according to the correlation between number of semantic links and number of services discussed above.

Overall, the preliminary evaluation results demonstrate the suitability of ERGOT, which achieves significant values of recall with low impact in terms of network traffic.

5 Related Work

Service discovery has been addressed from different, not necessarily disjoint, perspectives thus giving birth to several strands of research. Decentralized service discovery architectures have been proposed to cope with the pitfalls (e.g., scalability, fault tolerance) of centralized one. DUDE [1] extends the UDDI centralized service discovery mechanism by allowing multiple registries to form a federation with a DHT as a rendezvous point. In [11] a DHT based web service discovery system is proposed. 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 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

[6] http://www.census.gov/eos/www/naics

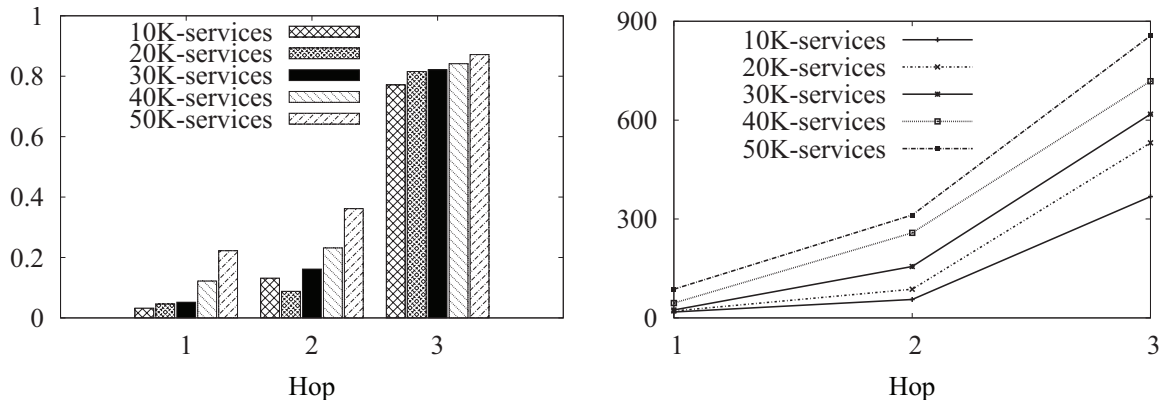

Figure 2. Recall and number of messages

leading to load imbalance. Meteor-S [8] supports 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 [5] aims at constructing an overlay network of peers by comparing their data content (web service descriptions). Nodes create links by comparing the inputs and the outputs of their services by exploiting the matchmaker described in [6]. The similarity between a query and the peers to whom forward it is computed by the same matchmaker. ERGOT shares some characteristics with the above-mentioned systems. In particular, it provides semantic characterization of services through ontologies. The main differences can be summarized as follows: (i) To the best of our knowledge, ERGOT is the only system combining for the purpose of service discovery; (ii) ERGOT adopts a ranking mechanism based on semantic similarity.

6 Conclusions

The ERGOT system proposed in this paper enables semantic-driven query answering in DHT-based systems by building a SON over a DHT. To enable semantic service matchmaking a measure of semantic similarity amongst service descriptions has been also defined. The preliminary evaluation of ERGOT demonstrates its efficiency both in terms of recall achieved and number of messages generated by each search. As a future work, we will study how to optimize system performance considering service density and other network/service parameters.

References

[1] S. Banerjee, S. Basu, S. Garg, S. Garg, S. Lee, P. Mullan, and P. Sharma. Scalable Grid Service Discovery based on UDDI. In *MGC*, 2005.

[2] D. Bianchini, V. De Antonellis, and M. Melchiori. Flexible Semantic-Based Service Matchmaking and Discovery. *World Wide Web*, 11(2):227–251, 2008.

[3] A. Crespo and H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. In *AP2PC*, 2004.

[4] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang. Similarity Search for Web Services. In *VLDB*, 2004.

[5] F.B Kashani, C. Chen, and C. Shahabi. WSPDS: Web Services Peer-to-Peer Discovery Service. In *ICOMP*, 2004.

[6] T. Kawamura, J. De Blasio, T Hasegawa, M. Paolucci, and K. Sycara. Public Deployment of Semantic Service Matchmaker with UDDI Business Registry. In *ISWC*, 2004.

[7] M. Li, B. Yu, O. Rana, and Z. Wang. Grid Service Discovery with Rough Sets. *IEEE TKDE*, 20(6):851–862, 2008.

[8] A. Patil, S. Oundhakar, A. Sheth, K. Verma Meteor-s web service annotation framework. In *WWW*, 2004.

[9] G. Pirró and N. Seco. Design, Implementation and Evaluation of a new Semantic Similarity Metric combining Features and Intrinsic Information Content. In *ODBASE*, 2008.

[10] O. Sahin, C. Gerede, D. Agrawal, A. El Abbadi, O. I. Ibarra, and J. Su. SPiDeR: P2P-Based Web Service Discovery. In *ICSOC*, 2005.

[11] C. Schmidt and M. Parashar. A Peer-to-Peer Approach to Web Service Discovery. *World Wide Web*, 7(2):211–229, 2004.

[12] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *SIGCOMM*, 2001.

[13] L. Vu, M. Hauswirth, and K. Aberer. Towards P2P-based Semantic Web Service Discovery with QoS Support. In *BPS*, 2005.