Unstructured P2P-Enabled Service Discovery in the Cloud Environment

Jing Zhou, Zhongzhi Shi

Unstructured P2P-enabled Service Discovery in the Cloud Environment

Jing Zhou[1],[2] and Zhongzhi Shi[2]

[1] Communication University of China, Beijing, 100025, China,
[2] The Key Laboratory of Intelligent Information Processing,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China
zhoujing@cuc.edu.cn

Abstract. As the Cloud computing appears to be part of the mainstream computing in a few years, the number of the services it provides, its users, and the requests of these services will be on the rise accordingly. To deliver satisfactory experience to Cloud users, it is essential that highly scalable techniques for service discovery should be available. We embarked on a preliminary study, in Cloud environments, on service discovery by adopting an unstructured P2P technique. In the context of Cloud computing, we start by examining proposed and deployed solutions to service discovery, discuss the methodology of developping efficient mechanisms for service description, description indexing, and query routing, and finally identify open issues.

1 Introduction

Armbrust *et al.* defined in [1] that Cloud computing comprises the applications delivered as services over the Internet and the hardware and system software in the datacenters that offer those services. Typically, Cloud computing comes in three kinds: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [2]. The classification emphasizes the core concept of "X as a Service", that is, software, platform, and infrastructure all can be provided to end users on demand and on a subscription basis.

Cloud services are currently available from providers such as Amazon, Microsoft, Google App Engine, Eucalyptus, and GoGrid. Users can access Cloud services by visiting service provider's website, establishing a runtime environment in response to instructions, creating a user account, and configuring related tools. There is no complicated mechanism for service discovery involved. As the Cloud computing will soon become part of the mainstream computing in a few years, the number of the services it provides, its users, and the requests of these services will increase in orders of magnitude. To deliver satisfactory experience to Cloud users, highly scalable techniques for service discovery should be available.

We observed in distributed systems that service discovery mechanisms are primarily centralized in which a central index server (or several such servers) stores and maintains all information about services being offered. Centralized

service discovery is prone to the following issues: 1) single point of failure; 2) lack of satisfactory scalability; 3) requirements for powerful computing capabilities to serve large amounts of service discovery and update queries on the central server; and 4) performance bottlenecks and network congestion. To overcome these obstacles so as to deliver a Cloud service discovery mechanism of high performance, we will rely on peer-to-peer (or P2P for short) computing [3].

The popularity of P2P computing stems from its self-organization, fault tolerance, and scalability, which make P2P computing an obvious candidate for tackling large scale service discovery in the Cloud environment. Studies [4] showed that satisfactory scalability could be achieved if P2P techniques based on DHTs (distributed hash tables) were applied to service discovery in Cloud computing. We, however, argue that such structured P2P techniques place severe constrains on the network topology and the placement of services (or their indices), which makes structured P2P unable to better model the real world.

We aim to address the issue of service discovery in the Cloud computing by exploring the Semantic Web technology to describe Cloud services and to facilitate service discovery and service matching, and investigating unstructured P2P-based techniques that support highly efficient and scalable routing mechanisms for service requests.

The balance of the paper is structured as follows. Related work is reviewed in Sect. 2. We present and discuss the primary design issues in Sect. 3 in greater details. Finally, we conclude the paper by identifying open issues in Sect. 4.

2 Related Work

2.1 Unstructured P2P and structured P2P

Unstructured P2P systems including Gnutella, Freenet, and FastTrack carry out object lookup and downloading operations in the absence of a central index server. Each peer maintains indices for the resources it currently holds. A lookup operation in such systems will not necessarily be successful and there is no upper and lower bounds on the successful operation, that is, non-deterministric query performance is offered. In addition, unstructured P2P systems support one-dimensional and multi-dimensional point queries and range queries.

Similarly, there is no central index server in structured P2P built upon DHTs (such as CAN, Chord, Pastry, and Tapestry). Each peer maintains the indices of data items on another $O(\log n)$ peers where n is the number of peers in the system. The keys of the data items and nodes are mapped onto the overlay network in which each node is responsible for managing a small number of data items. Whenever a peer receives a lookup request, it is able to locate the data item of interest within $O(\log n)$ hops. Therefore, the query performance delivered by structured P2P is deterministic. One-dimensional point queries can be well supported but, in order to implement one-dimensional range queries and multi-dimensional point queries and range queries, the system needs to employ spatial indices [5] such as the Space Filling Curve, kd-trees and MX-CIF quadtree, and carry out sufficient extension to the basic DHT mechanism [6].

Note that each peer in structured P2P maintains the indices of data items of others (but not the data items of its own!) and reliance among peers exists. However, each Cloud service provider maintains its own services, describing the services, creating indices according to service description, and publishing service description to facilitate service discovery. Furthermore, service discovery in Cloud computing will regularly be performed based on *multiple* (instead of single) attributes of the service in most cases. We believe that the search mechanism enabled by unstructured P2P techniques can better resolve multi-dimensional queries by incorporating support for semantics, whereas the basic DHT technique delivers satisfactory performance in keyword search.

2.2 P2P-based Service Discovery

A robust, automatic, and reliable mechanism for service discovery is indispensable to distributed systems such as Grid computing, Web services, and pervasive computing because it helps users to locate services or resources required in such systems [7–10]. To overcome the drawbacks and inefficiencies of centralized service discovery in terms of scalability, fault-tolerance, and network congestion, a number of fully decentralized solutions were proposed. Among others, the approach that adopts the P2P paradigm draws a great deal of attention (see [11] for details on P2P-based resource[3] discovery in computational grids).

The first step towards implementing a P2P-based mechanism for service discovery is to design a decentralized index system in which the appropriate data structure is selected for building indices upon the service description. Since a service is typically characterized by both static and dynamic attributes, users can specify the requirement for multiple attributes in a query, that is, a multi-dimensional query is allowed. In unstructured P2P, each peer maintains its own services and service descriptions, hence simple indices, such as the two dimensional table, can be employed to faciliate resolving multi-dimensional queries.

The routing mechanism for multi-dimensional queries in the P2P paradigm should also be developed, which helps forward service requests to their destination peers efficiently. Flooding and its variants [12, 13] are the widely used approaches to message routing in unstructured P2P. The main drawback of such techniques includes that excessive unnecessary query messages and traffic are generated. Vairous heuristics [14, 15] were therefore developed to guide message routing as well as to increase system scalability.

When a query is eventually delivered to a potential destination peer, a matching scheme is needed to confirm a perfect match between the service description and the user requirement expressed in the query message. Currently, keyword-based methods widely used in structured P2P, due to their unsatisfactory search performance, have been gradually replaced by semantics-based solutions. This topic is however not the focus of our paper and we leave it to the future work.

[3] The "resource" here may refer to clusters, supercomputers, and desktop computers and this concept has an overlap with that of the "service" in Cloud computing.

2.3 Service Description and Discovery Using Semantics

When describing a service, elements such as the service properties, capabilities, and constraints should be taken into account. To service the request, the Semantic Web offers a number of powerful tools. The integration of the Semantic Web technologies is beneficial to the realization of efficient service discovery due to the following reasons: 1) Ontology can be used to describe concepts and the relationships among the concepts within a specific domain in a disambiguous way and hence, people often employ ontology to describe services and encode the description semantically; 2) Languages such as OWL can be used to describe ontologies in order to support semantic inference for relationships among various concepts; 3) A number of tools in the Semantic Web community have been developed for service (Web Services in particular) description purposes, including OWL-S [16], WSMO (Web Services Modeling Ontology) [17] and Web Service Semantics - WSDL-S [18]; and 4) The semantic service description with powerful expresiveness is necessary to service matching based on semantics.

Semantics-based service matching comes in two forms: signature matching [19] and specification matching [20]. In signature matching, the subsumption relationship between concepts that are defined in ontologies and are used to describe the input capabilities (that is, the required services) and output capabilities (that is, the available service descriptions), is identified. Specification matching is, however, performed between the pre-conditions and post-conditions of the functional semantics of software components using automated theorem proving or query containment. Semantic matching has been applied to a number of distributed systems, including Grid computing [21, 22], P2P computing [23, 24], Web Services [25], and pervasive computing [10].

To address the issue of service matching in the Cloud environment in particular, Zeng *et al.* proposed a matching algorithm which extends the keywords that describe the input and output capabilities of services by using WordNet [26]. A function is employed to evaluate the semantic similarity between the concept sets (consisting of concepts extended from the keywords) of any two services. The primary drawback of the approach is lack of support for range queries.

3 System Design

3.1 System Overview

Our proposed system consists of Cloud service providers (peers in P2P terminology) that describe their own services, create indices according to service descriptions, and publishing service descriptions for sharing purposes. Each new peer or node, upon arrival, will exchange its service descriptions with others (and maintain the information in its local index) that are already in the network, that is, 1-step replication [27] is utilized, thus forming a neighboring relationship with those nodes. When a peer voluntarily leaves the system, or is lost due to topology re-organization, the index information for that neighbor gets flushed.

According to our previous work on unstructured P2P, flooding and random walk search techniques can hardly offer scalability comparable to that of DHT-based approaches. One efficient solution to the problem is to introduce a certain degree of centralized control in the P2P network, that is, supernodes [14] [27]. We therefore allow a few number of peers with high capacity to act as supernodes as needed. For instance, when a peer finds its knowledge about services provided by other peers is rich enough[4], it can elect itself to be a supernode by informing all neighbors of the decision. If the latter agree to accept the peer as a supernode, they will update their indices to reflect the change. The primary motive underlying the introduction of supernodes is to bias query messages towards nodes that provide a shortcut to the destination peers with a high probability.

Moreover, to avoid overburdening nodes with query messages from neighbors, we allow nodes to add links to more useful neighbors and drops links to useless ones, thus leading to reorganization of the P2P overlay topology. Topology reorganization is often used by P2P systems to faciliate resource/service discovery by shortening the distance (in the form of hops) between the potential destination peers and the resource/service requester based on predictions of future requests.

We elaborate on issues comprising service description, description indexing, and routing of service requests in the following sections.

3.2 Service Description

To describe Could services, we start by using WSDL-S–currently a W3C member submission that offers a lightweight solution to creation of semantic service description. In WSDL-S, the expressivity of WSDL was augmented with semantics by adopting concepts similar to those in OWL-S. We may gradually extend and enhance WSDL-S by means of the extensibility provided by WSDL itself according to the specific requirements for Cloud service descriptions.

In a nutshell, WSDL-S provides a few extensibility elements to realize the URI reference mechanism as follows [18].

wssem:modelReference specifies the association between a WSDL entity and a concept in a semantic model.

wssem:schemaMapping handles the structural difference between the schema elements of a Web service and their corresponding concepts in a semantic model.

wssem:precondition and wssem:effect are specified as child elements of the element *operation* and describe the semantics of the operation.

wssem:serviceCategorization comprises service categorization information that could be used when publishing a service in a Web Services registry.

[4] In related work such as [14], supernodes refer to peers with higher bandwidth connectivity, whereas in our system in which service description is encoded with semantics (see Sect. 3.2) and we argue that nodes with richer knowledge should be able to better serve service requests and thus be elected as supernodes.

A Cloud service can be semantically annotated by borrowing all these constructs from WSDL-S. Figure 1 presents an example Cloud service description. Note that a precondition is a set of statements that should be true before a service can be successfully invoked. Hence, we can describe part of a user's requirement for a service by means of preconditions. For instance, if a user is looking for a disk storage service, she can send out a "DiskStorage" request and specify the capacity should be 200GB and the transfer rate of the disk should be 50Mb/s. However, in proposal [18] at most one precondition (as well as one effect) is allowed so multiple preconditions should be captured into one high level precondition. We can combine those two statements via "AND" into one precondition since **WSSemantics.xsd** defines an attribute "expression" of type "string" for **wssem:precondition**.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<definitions name="DiskStorage"
   targetNamespace="http://www.example.com/wsdl-s/examples/diskstorage.wsdl"
   xmlns="http://www.w3.org/2004/08/wsdl"
                .....
   xmlns:wssem="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
   xmlns:DSOntology="http://www.example.com/wsdl-s/ontologies/DiskStorage.owl"
   <types>
                .....
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
        xmlns="http://www.example.com/wsdl-s/examples/diskStorage.wsdl">
        <xs:complexType name="processDiskStorageRequest">
             <xs:all>
               <xs:element name="billingInfo" type="xsd1:DSBilling"/>
               <xs:element name="storageItem" type="xsd1:DSItem"/>
             </xs:all>
        </xs:complexType>
      <!–Semantic annotation is added directly to leaf element –>
        <xs:element name="processDiskStorageResponse" type="xs:string"
        wssem:modelReference="DSOntology#StorageConfirmation"/>
      </xs:schema>
   </types>
   <interface name="DiskStorage">
      <operation name="processDiskStorage" pattern="wsdl:in-out"
        wssem:modelReference="DSOntology:RequestDiskStorage">
        <input messageLabel ="processDiskStorageRequest"
             element="tns:processDiskStorageRequest"/>
        <output messageLabel ="processDiskStorageResponse"
             element="processDiskStorageResponse"/>
        <wssem:precondition name="AvailableStoragePrecond"
             wssem:modelReference="DSOntology#StroageAvailable"
             expression="DSOntology#Capacity=200GB&&
               DSOntology#Transferrate=50Mb/s"/>
        <wssem:effect name="StorageOccupiedEffect"
             wssem:modelReference="DSOntology#StorageOccupied"/>
      </operation>
   </interface>
</definitions>
```

Fig. 1. An example Cloud service description

3.3 Description Indexing

Among all the constructs used to describe a Cloud service, the precondition is the most important resource upon which we can build indices for all the services in a P2P network. Suppose a user issues a Cloud service request and we can then compile part of the request (which is specified in **wssem:precondition**) as (*attri_1*, *value_1*), (*attri_2, value_2*), ..., (*attri_i*, *value_i*).

As such, a user request can be converted to a multi-dimensional query by which the user specifies multiple conditions should be met on several attributes. We adapted one of the database approachs–the kd-tree–to establish a data space for all service descriptions. Once the data space is constructed, we set out to develop the routing strategy (in the following section) by which a multi-dimensional query can be efficiently forwarded to the relevant peers in such a space.

The kd-tree is a data structure that decomposes a multi-dimensional data space into hyperrectangles and each node corresponds to a hyperrectangle. The fields of a kd-tree node include the splitting dimension number, splitting value, left kd-tree, and right kd-tree. According to the first two fields, each node splits the space into two subspaces. Searching for a point in the dataset represented in a kd-tree can be carried out in a traversal of the tree from root to leaf ($O(\log n)$ if there are n data points). In the following, we demonstrate a kd-tree representation of 4 points (200GB, 50Mb/s), (300GB, 80Mb/s), (600GB, 30Mb/s), and (800GB, 90Mb/s). For simplicity, a 2-d space is used for illustration.

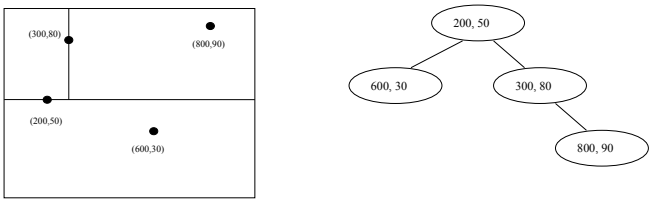

Fig. 2. A k-d tree example

In Fig. 2, the root node (200GB, 50Mb/s) splits the space in the y-axis into two subspaces. The point (600GB, 30Mb/s) lies in the lower space, that is, $\{(x, y)|y < 50\}$ and hence is in the subtree of the root node. By traversing such a kd-tree, the hyperrectangle leave that potentially contains the target point can be located.

In our system, each new node locally creates an index in the form of a kd-tree for all the service descriptions it is aware of. The index will gradually grow and evolve as nodes join and depart from the system. The description indices are therefore distributed among individual nodes and will be used for routing service requests.

3.4 Routing of Service Requests

Part of a user's service request (encoded in **wssem:precondition**) is used to formulate a multi-dimensional query. To execute the query over a P2P network, it must be routed to the set of nodes that contain data relevant to the query. We describe routing as follows.

1. According to the content of the query, the node issueing the query will first traverse its kd-tree to locate the set of potential target nodes and subsequently forward the query to those nodes. If no such nodes are found, the node simply sends the query to its associated supernodes, if any. Otherwise, it will flood the query message to all neighbors.
2. Upon receipt of an incoming query, the receiving node checks its local index to find the target service that might satisfy the user query[5] and forwards the query to the potential targets. If no such nodes are found, the node routes the query to its associated supernodes, if any. Otherwise, it will flood the query message to all neighbors if the message was not flooded in its last hop. A flooded message, in this case, will be sent to a randomly chosen neighbor.

This process is repeated until: 1) there are no other potential targets to which a query can be further routed, 2) the TTL (Time-To-Live) in the query message is decreased to zero, or 3) the service requester explicitly claims that enough results have been collected.

3.5 Discussions

- **Decentralized indices**: By locally creating a kd-tree to index all service descriptions that a peer node is aware of, we obtain a fully distributed index (consists of all the local indices) for service discovery in the proposed P2P system. Neighbors simply delete index information about services of a leaving node from their kd-tree, whereas in a DHT-based structured P2P system, node departure incurs much more operations.
- **Support for point and range queries**: Thanks to the use of the kd-tree, both a point query and a range query can be similarly handled in our system. This is in contrast to other approaches in which it is easy to resolve the point query but sufficient extensions should be made to the basic mechanism (DHTs for instance) if range queries are also supported.
- **Localized scheme**: We proposed a localized scheme when constructing the kd-tree index for each peer node, that is, no global knowledge is required. However, in [28] when "skip pointers" (shortcuts used for optimized routing based on kd-trees) are built, the set of all nodes should be known in advance.

4 Future Work

In our preliminary work on unstructured P2P-enabled service discovery in Cloud computing, much still is left to be investigated. For instance, we have yet to experimentally demonstrate the efficency and the scalability of the routing algorithm. Moreover, developing matching and sorting algorithms for semantics-based service matching is also indispensable for realizing effective and efficient service discovery in the context of Cloud computing. We are currently exploring these issues and will report our onging work in a forthcoming paper soon.

[5] A perfect match is only confirmed after the semantics-based service matching has been carried out.

Acknowledgment

This work is funded by 382 Research Foundation for Talented Scholars (No. G08382320), the Engineering Disciplines Planning Project (No. XNG0921), and the Leading Academic Discipline Program (3rd phase of 211 Project for the Communication University of China). We also acknowledge the input of the National Natural Science Foundation of China (No. 60775035, No.60933004, No.60970088), the National High-Tech Research and Development Plan of China (No. 2007AA01Z132), National Basic Research Priorities Programme(No. 2007CB-311004) and National Science and Technology Support Plan (No.2006BAC08B06), Dean Foundation of Graduate University of Chinese Academy of Sciences(O85101-JM03).

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley (2009)
2. Buyya, R., Pandey, S., Vecchiola, C.: Cloudbus toolkit for market-oriented cloud computing. In: Proceedings of the 1st International Conference on Cloud Computing, Beijing, China (2009) 24–44
3. Clark, D.: Face-to-face with peer-to-peer networking. Computer Journal **34**(1) (2001) 18–21
4. Ranjan, R., Zhao, L., Wu, X., Liu, A.: Peer-to-peer cloud provisioning: Service discovery and load-balancing. The Computing Research Repository **abs/0912.1905** (2009)
5. Samet, H.: The Design and Analysis of Spatial Data Structure. Addison-Wesley Publishing Company (1990)
6. Cai, M., Frank, M., Chen, J., Szekely, P.: Maan: A multi-attribute addressable network for grid information services. Journal of Grid Computing **2** (2004) 3–14
7. Toma, I., Iqbal, K., Roman, D., Strang, T., Fensel, D., Sapkota, B., Moran, M., Gomez, J.M.: Discovery in grid and web services environments: A survey and evaluation. Multiagent and Grid Systems **3**(3) (2007) 341–352
8. Bachlechner, D., Siorpaes, K., Lausen, H., Fensel, D.: Web service discovery - a reality check. In: Demos and Posters Session of the 3rd European Semantic Web Conference, Budva, Montenegro (June 2006)
9. Zhu, F., Mutka, M.W., Ni, L.M.: Service discovery in pervasive computing environments. IEEE Pervasive Computing **4**(4) 81–90
10. Mokhtar, S.B., Preuveneers, D., Georgantas, N., Issarny, V., Berbers, Y.: Easy: Efficient semantic service discovery in pervasive computing environments with qos and context support. Journal of Systems and Software **81**(5) 785–808
11. Ranjan, R., Harwood, A., Buyya, R.: Peer-to-peer-based resource discovery in global grids: A tutorial. IEEE Communications Surveys and Tutorials **10**(1-4) (2008) 6–33
12. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer systems. In: Proceedings of the 16th international conference on Supercomputing, New York, New York, USA (2002) 84–95

13. Iamnitchi, A., Foster, I., Nurmi, D.C.: A peer-to-peer approach to resource location in grid environments. In: Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing. (2002) 419
14. Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., Shenker, S.: Making gnutella-like p2p systems scalable. In: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, Karlsruhe, Germany (2003) 407–418
15. Zhou, J., Hall, W., Roure, D.C.D., Dialani, V.K.: Supporting ad-hoc resource sharing on the web: A peer-to-peer approach to hypermedia link services. ACM Transactions on Internet Technology **7**(2) (May 2007)
16. 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.: Owl-s: Semantic markup for web services. http://www.daml.org/services/owl-s/1.2/ (2010)
17. Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., , Fensel, D.: Web service modeling ontology. Applied Ontology **1**(1) (2005) 77–106
18. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.T., Sheth, A., Verma, K.: Web service semantics - wsdl-s. http://www.w3.org/Submission/WSDL-S/, W3C Member Submission (2005)
19. Zaremski, A.M., Wing, J.M.: Signature matching: a tool for using software libraries. ACM Transactions on Software Engineering and Methodology **4**(2) (1995) 146–170
20. Zaremski, A.M., WING, J.M.: Specification matching of software components. ACM Transactions on Software Engineering and Methodology **6**(4) (1997) 333–369
21. Harth, A., Decker, S., He, Y., Tangmunarunkit, H., Kesselman, C.: A semantic matchmaker service on the grid. In: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, New York, NY, USA 326–327
22. Ludwig, S.A., Reyhani, S.M.S.: Introduction of semantic matchmaking to grid computing. Journal of Parallel and Distributed Computing **65**(12) 1533–1541
23. Zhou, G., Yu, J., Chen, R., Zhang, H.: Scalable web service discovery on p2p overlay network. In: Proceedings of the IEEE International Conference on Services Computing (SCC 2007), Salt Lake City, Utah, USA (2007) 122–129
24. Li, Y., Zou, F., Wu, Z., Ma, F.: Pwsd: A scalable web service discovery architecture based on peer-to-peer overlay network. In: Proceedings of the 6th Asia-Pacific Web Conference. (2004) 291–300
25. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.P.: Semantic matching of web services capabilities. In: Proceedings of the 1st International Semantic Web Conference on The Semantic Web, Sardinia, Italy (2002) 333–347
26. Zeng, C., Guo, X., Ou, W., Han, D.: Cloud computing service composition and search based on semantic. In: Proceedings of the 1st International Conference on Cloud Computing. (2009) 290–300
27. Gkantsidis, C., Mihail, M., Saberi, A.: Hybrid search schemes for unstructured peer-to-peer networks. In: Proceedings of IEEE INFOCOM. (2005) 1526–1537
28. Ganesan, P., Yang, B., Garcia-Molina, H.: One torus to rule them all: multi-dimensional queries in p2p systems. In: Proceedings of the 7th International Workshop on the Web and Databases, Paris, France (2004) 19–24