

# Grid Resource Discovery Based on Semantic P2P Communities

Juan Li      Son Vuong  
University of British Columbia  
2366 Main Mall, Vancouver, B.C., Canada  
(1)-604-822-6366  
{juanli,vuong}@cs.ubc.ca

## ABSTRACT

Grid technologies enable the sharing of a wide variety of resources. The full use of these resources requires effective resource discovery mechanisms. However, the complicated and dynamic characteristics of grid resources make sharing and discovering them a challenge. In this paper, we propose a semantic community approach to enable efficient resource discovery in grids. The system clusters nodes into communities according to their semantic properties. The community construction and maintenance is fully decentralized and self-organizing. This structure helps prune the searching space and reduce the cost of searching. The system exhibits many desirable properties: it supports complex queries and is fully decentralized, scalable, and efficient. Our simulation results show how searching the grids can take advantage of semantic communities to reduce searching costs and improve the quality of results.

## 1. INTRODUCTION

Information service, one of the key services of grids, provides resource information to users. To make information available to users quickly and reliably, an effective and efficient resource discovery mechanism is crucial. However, grid resources are potentially very large in number and variety; individual resources are not centrally controlled, and they can enter and leave the grid systems at any time. For these reasons, resource discovery in large-scale grids can be very challenging.

Traditionally, resource discovery in grids is based mainly on centralized or hierarchical models. For example, in the Globus Toolkit [6], users can get a node's resource information by directly querying a server application running on that node, or by querying dedicated information servers that retrieve and publish an organization's resource information. Although interactions between these information servers are supported, a general-purpose decentralized service discovery mechanism is still missing.

P2P techniques have been used to discover resources in more dynamic, large-scale, and distributed environments. For example, [30] organizes information nodes into a flat unstructured P2P

network, and random-walk based methods are used for query forwarding. Random-walks are not efficient in response time for a very large system. [12] proposes a hierarchical structure to organize information nodes to reduce redundant messages. However, a well-defined hierarchy does not always exist, and the global hierarchy is hard to maintain in a dynamic environment. Papers [22] and [23] present DHT-based multi-attribute resource discovery approaches, but these may incur either a high traffic load for result intersection or large overhead for multiple publication and update.

In this paper, we propose a community-based P2P model to optimize grid resource discovery. Communities are implicitly formed based on the nodes' resource ontology. Communities help to prune the searching space and disseminate information only to related nodes, thus improving the efficiency of searching. We use a SkipNet [9] overlay to assist nodes to discover their interested communities. The community discovery, construction, and maintenance are manipulated in a decentralized and automatic manner. Communities have a loose multi-layered structure that increases the flexibility and efficiency of intra-community searching. The system exhibits many plausible characteristics, such as high scalability, short latency, a low bandwidth, and support for complex queries.

## 2. COMMUNITIES AND ONTOLOGY

In social networks, a so-called small-world [4] structure has been observed. The small-world networks exhibit special properties, namely, a small average diameter and a high degree of clustering, which make them effective and efficient in terms of spreading and finding information. Because of the similarity between grid networks and social networks and the fact that human users direct grid nodes' links, grids also exhibit small-world behavior [5]. We use this theory to create communities — small-worlds — in grids. We can create communities in scientific domains such as physics, biomedicine, mathematics, and computer science. Communication and collaboration can operate on top of the communities. With communities, grid users can conveniently work in groups, even though they may not always be located in geographical proximity.

To create and make use of the interest-based community structure, we need to find the grid nodes' underlying interests. According to the extensive previous research in data mining and text retrieval [25, 26], a node's local resources reflect its interests. In addition, to communicate and collaborate, nodes in the same community should be able to understand each other, and speak the same "language". Based on these two concerns, we classify nodes by the dominant ontology of their resource possession. Ontology is defined as "a formal, explicit specification of a shared

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'06, April, 23-27, 2006, Dijon, France.

Copyright 2006 ACM 1-59593-108-2/06/0004...\$5.00.

conceptualization” [7], which can refer to the shared understanding of some domains of interests. An ontology schema always reflects the hierarchical relationships between concepts (e.g., “subClassOf”, “subPropertyOf”). As Herbert Simon [8] argues, hierarchy emerges inevitably in any complex system. In large-scale grids, simple flat structured communities cannot work efficiently in response to the changing and growing nodes; rather, a multilevel hierarchical structure is necessary. Therefore, we use the ontology hierarchy to represent the nodes’ interest category. The hierarchical design allows multilevel communities to form and offers an optimal community size and a flexible searching scope. The ontology domain can be a global taxonomy like the categories defined in Yahoo, Google, and DMOZ [27], or a general ontology that formalizes notions such as processes and events, time and space, physical objects, and so on. Figure 1 shows an example of the ontology defined in Cyc [24]. We assume the domain information is defined in a node’s data ontology and each data item can be classified into one domain (or sub-domain). We assume queries can be classified with the same policy as well.

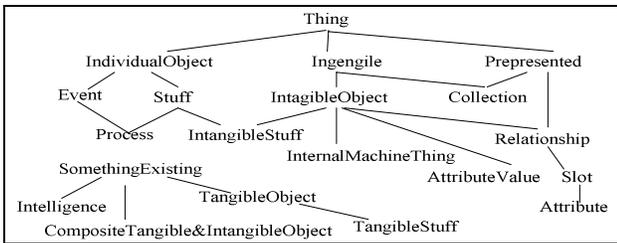


Figure 1. Top-level categories in Cyc [24].

### 3. COMMUNITY DISCOVERY

After the grid nodes’ interests have been determined, the system can cluster them accordingly to form communities. Roughly, there are three existing approaches to discovering and organizing communities. The first approach is to use a centralized index server, such as Yahoo and MSN chat rooms, to organize communities. Centralized indexing is efficient, but the server may become a potential bottleneck and point of failure when the number of clients increases. Another method is to broadcast the community advertisement. For example, in JXTA [31], a node creates a node group by publishing group advertisements in the network. Nodes receiving the advertisement can join the group. The disadvantage of this approach is that nodes far from the group initiator may not find the group. We call the third approach the history-based friend-list approach. Many recent applications, e.g., [28] and [15], use this approach to improve searching performance in unstructured networks. A node adds other nodes that answered previous queries to a friend list (or shortcut list) and forwards queries to them in the future. This approach is decentralized and scalable. However, the acquired knowledge may be inaccurate and incomplete. What is more, nodes that do not have history experiences cannot take advantage of the communities. In this section, we introduce a community discovery and formation algorithm that overcomes the problems of these existing approaches.

#### 3.1 SkipNet Category Overlay

We use a SkipNet [9] overlay, which we also call a category overlay, to assist nodes to find other nodes sharing similar

interests. SkipNet is a scalable overlay network. Similar to DHTs [16–19], SkipNet can locate a data item with logarithmic steps. Moreover, it provides controlled data placement and a guaranteed routing locality by organizing data primarily by string names rather than hashed identifiers. Nodes in the SkipNet are organized into a circular distributed data structure, which contains multiple levels of rings. Each SkipNet node has a numerical ID and a name ID. In this paper, we generate the numerical ID by hashing the node’s IP address, as existing DHTs have done. The name ID adopts a hierarchical format as in the DNS system. We create the name ID by concatenating the node’s interest and its identification. For example, node  $n_0$  interested in *Computer Science (CS)* / *Artificial Intelligence (AI)* has a name ID “CS.AI. $n_0$ .” Each node in the category overlay is in charge of one category, which it is interested in. For example, node  $CS.AI.n_0$  is in charge of domain *CS.AI*. Nodes are lexicographically ordered in each SkipNet ring according to the name ID. Paper [9] describes the algorithms for dealing with node joining, leaving and routing in SkipNet.

#### 3.2 Interest Registration

Nodes in the system form two layers of overlay: all nodes are in the community overlay, and a small part of them form a SkipNet category overlay. Now let us see how the SkipNet category overlay can help nodes form flexible multi-layered communities. When a node wants to join a community, it registers its interest to the category overlay. (A node with multiple interests can register with several communities.) Specifically, the interest of the joining node is routed along the overlay and registers to all “related” overlay nodes in the routing path, until up to the destination node that manages the interest. Here, “related” nodes are nodes that are in charge of predecessors of the registered interest. For example, in Figure 2, a new node registers its interest *CS.AI* to the overlay from a node in the *EE* domain. The register message is routed through the SkipNet ring to the *CS* domain. Inside the *CS* domain, the first node the message gets to is  $CS.n_1$ . Because *CS* is the predecessor of *CS.AI*, the new node registers to  $CS.n_1$ . Finally, the message registers to the destination node  $CS.AI.n_0$ . The joining node knows other nodes that share a similar interest from the category nodes to which it registers. Therefore, during the registration, the new node finds neighbors in different levels of the interest hierarchy, and joins their community. On average SkipNet routes a message to its destination in  $O(\log M)$  hops, where  $M$  is the number of nodes in the SkipNet overlay. In our system,  $M$  is much smaller than  $N$ , the total number of nodes in the system. Therefore, a node can quickly locate its community.

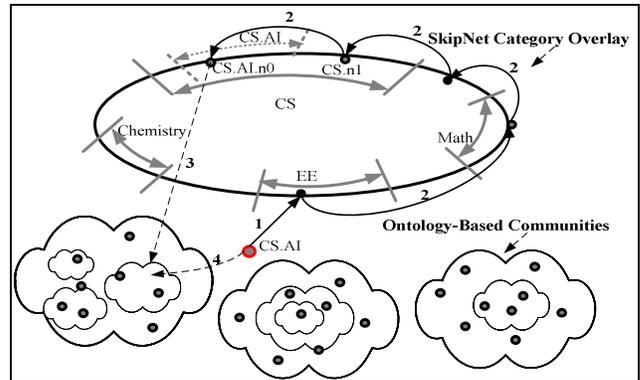


Figure 2. Interest registration.

There are several special cases for a node’s registration: (1) If a new node cannot get enough neighbors from the registration path, it explicitly routes to the upper- and lower-level categories to register and get more neighbors. (2) If a node cannot find its interested category in the overlay, i.e., it is the first node registering this interest, and if it satisfies the qualification requirements for a category node, it joins to the SkipNet overlay and takes charge of this category.

### 3.3 Remarks

Readers may wonder where the SkipNet nodes come from. In fact, they are also user nodes. We use the heterogeneity of grid nodes, and promote those stable and powerful ones to join the SkipNet overlay. Excluding ephemeral nodes from the SkipNet overlay avoids unnecessary maintenance costs. A node joins the SkipNet only when three conditions are satisfied: (1) It satisfies the capacity requirements, i.e., it is powerful enough. (2) It is stable for a threshold time period. (3) It is the first node that registers this interest, or it is not but the load of this interest is high inside the category overlay.

SkipNet is not chosen arbitrarily; its locality properties make it a good implementation for the category overlay. DHTs are an alternative, but they cannot provide the advantages SkipNet provides: First, a node has no incentive to store unrelated nodes’ information, so it is necessary to guarantee that a node is in charge of only interested information. We add a node’s interest in its name ID, and routing is based on the name ID. Therefore, a SkipNet category node stores only the information of nodes that share the same interest as its own. In addition, the naming and routing schemes guarantee that interests in the same branch of the ontology tree remain in the same arc of the SkipNet ring. For example, domains *CS.AI.neural-nets* and *CS.AI.nat-lang* are both inside domain *CS.AI*. In this way, a node can quickly locate nodes sharing similar interests. This property is very useful for the category overlay. As we explain in the following section, nodes need to register to different hierarchies of an interest domain to form multi-layered communities.

Unlike super-peers in Gnutella [11], SkipNet category nodes do not deal with detailed content searching; therefore, the load of the overlay is not that high. Its load comes from the nodes’ register, update, and community lookup operations. Both the register and update messages are lightweight, basically only the node’s IP address and its interested domain. Besides, only the most representative interests, usually only a few, are registered for each node. To control the update overhead, the update period can be adaptively adjusted. To reduce the community lookup overhead, nodes will first check their local and neighboring cache before they lookup the SkipNet overlay.

## 4. VIRTUAL COMMUNITIES

### 4.1 Community Topology

From the category overlay, registering nodes know neighbors from different levels of the interest hierarchy. Then they connect with each other, forming a multi-layered community. The community is an unstructured network like the Gnutella network, except that nodes discriminate neighbors from different levels.

Figure 3 illustrates the center node’s neighborhood topology. All other nodes are the center node’s neighbors. The center node’s interest is closer to the inner neighbors’ interests than to those of

the outer neighbors. In Figure 3, the center node and its neighbors in the white ring are interested in domain *a.b.c*; neighbors in the light gray ring are interested in domain *a.b*, and neighbors in the dark gray ring are interested in *a*. Inside the community, nodes may use the Gnutella Ping/Pong protocol to update their neighbor tables. A node contacts inner neighbors more frequently than it contacts outer ones. Therefore, each node gets accurate information about nearer neighbors, and coarse information for further ones. This approach scales well to a large-scale community as the overhead is controlled.

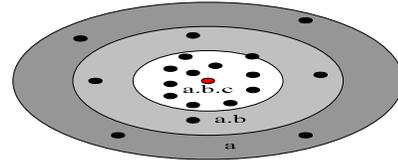


Figure 3. A node’s neighborhood topology in the community.

### 4.2 Query

Various strategies can be used for query forwarding inside the community overlay. For example, if controlled flooding such as the Gnutella searching protocol is used, the query is forwarded to neighbors until the time-to-live (TTL) value reaches zero. We do not focus on how queries are routed within a community overlay, since it has been well studied in the literature [2, 5, 11, 14].

With this layered community architecture, searching can be focused on the small region most related to the querying node’s interest, but it will not be restricted to this region. Searching can break through the small region to a wider yet related world by forwarding through the outer neighbors. Since nodes with similar interests are in the same community overlay, we can expect that most queries will be satisfied within the local community.

It is possible, though infrequent, that a node may want to search content in different communities. The category overlay facilitates query forwarding among communities, just as it helps nodes join their interested communities. To alleviate the lookup overhead on the category overlay, a node first checks its own local host cache to see whether it has cached nodes falling into the target community. Moreover, it also queries neighboring nodes on their local host caches. Only after all these attempts fail will the lookup request be forwarded to the category overlay.

## 5. EXPERIMENTS

We performed extensive simulations to evaluate the performance of our searching scheme. To make comparisons, we simulate our SkipNet-assisted multi-level (SAML) community in conjunction with the learning-based friend-list (LFL) community [28] and a randomly connected Gnutella network. We use Gnutella protocol as the basic searching protocol for all the three approaches. Initially, nodes are connected in a random manner. A SAML node registers its interests when it joins the network. (In practice, this is not necessary.) For the LFL community, as the simulation continues, each node updates its neighbors based on query experience. The resource set includes 500,000 resources falling into 8 big domains. Every domain has 3 to 8 sub-domains, and there are up to 5 levels of hierarchy. Every node keeps 10 to 50 resources. Requesters are randomly chosen from the network. Queries created by a node with interest *i* have a 70% possibility of

falling into domain  $i$ , 20% possibility into ancestor (or descendent) domains of  $i$ , and 10% into any arbitrary domains. The dynamic network behaviors are simulated as follows: in every unit simulation session, an active node has a 20% possibility of creating a query, a 1% possibility of updating its resources, a 1% possibility of leaving the system, and the same number of offline nodes join the system, starting functioning without any prior knowledge. Our evaluation metrics are (1) The recall rate, which is defined as the number of results returned divided by the number of results actually available in the network. (2) The clustering coefficient of a node, which is the ratio of the number of existing edges and the maximum number of possible edges connecting its neighbors. (3) The scope of a query, which is defined as the fraction of nodes involved in the query processing and the total nodes in the system.

First, we ran the simulation on a network with 10,000 nodes. The average node degree is 5 for all these three schemes. The TTL value is set to 5. Results are collected when all nodes join the network. Figure 4 compares the three routing strategies in terms of query recall rate. We can see that SAML outperforms both of the other two searching schemes. The recalls of our SAML searching and the Gnutella searching are constant over the simulation period. The recall of LFL increases as a result of the nodes' learning, and then it becomes stable and remains constant. Because new joining nodes have no knowledge to forward the query, and the knowledge learned from experiences is not as comprehensive as the community category knowledge in SAML, the recall of the LFL is not as good as that of SAML.

Figure 5 plots the time-evolution of clustering coefficient ( $cc$ ). The same as the recall rate in Figure 4, the  $cc$  of SAML and Gnutella (its value is too small to see in the figure) remains constant, and the  $cc$  of the LFL increases at first and then it converges to a constant value. Unlike the recall rate in Figure 4, the constant  $cc$  value of LFL is higher than that of SAML. Generally speaking, a higher value of  $cc$  means a well connected cluster and a good searching performance. Therefore, random Gnutella's small  $cc$  value (around 0.0003) can explain its poor recall rate. However, LFL's  $cc$  is high, even higher than that of SAML. Then why is its recall lower than SAML's? To answer this question, we looked into the communities created by LFL. We found that nodes tend to form very small and tight communities. Communities with similar interests may be distributed in the network like small scattered islands. This deteriorates the searching performance.

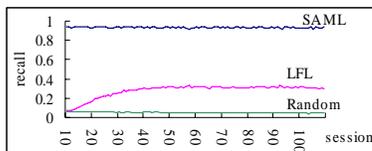


Figure 4. Time evolution of the recall rate.

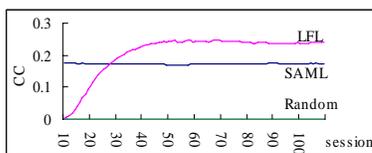


Figure 5. Time evolution of the clustering coefficient.

We also compared the three searching schemes in different sized networks. The results in Figures 6–8 are average values during the whole simulation period. Figure 6 compares the three routing strategies in terms of query recall rate. With the same TTL, Gnutella and LFL's recall rates decrease substantially as the network size increases, but SAML is not directly affected by the network size. We can see that our routing scheme achieves a very high recall rate even when the network size is large. As Figure 7 shows, SAML and LFL have a large clustering coefficient, much larger than that of the random network (too small to see in the figure) with the same number of nodes and edges. Figure 8 shows the average query searching scope for each query strategy. The scope for a query is the fraction of nodes that see the query. It is closely related to the load of the system. A smaller query scope increases system scalability. For example a scope of 100% means that all nodes see the query, and the load caused by the query is high. The result shows that SAML achieves load reduction by forwarding queries to only a small number of nodes. Figures 6 and 8 indicate that with a network size of 150,000, SAML can achieve about a 93% recall rate with only 1% scope. (Note: the recall and scope is achieved with TTL=5).

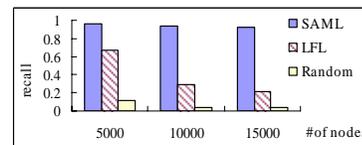


Figure 6. Recall vs. network size.

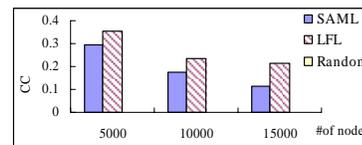


Figure 7. Clustering coefficient vs. network size.

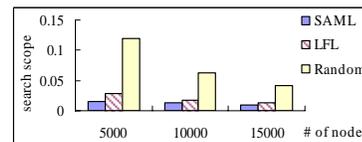


Figure 8. Searching scope vs. network size.

## 6. RELATED WORK

Many recent P2P-based searching techniques relate to our research. Flooding is the predominant search method in unstructured P2P networks. This method, though simple, does not scale well in terms of message overhead. There have been numerous attempts [1–3] to enhance its scalability. Random walks [14, 20] are an alternative to flooding for unstructured searches. They can reduce network traffic but at the cost of query latency. Recently, hierarchical super-peer systems [13] have been proposed to improve searching efficiency. DHTs [16–19] have received a lot of attention in the last few years. These systems are scalable and efficient. However, a missing feature of DHTs is support for rich queries. Another hurdle to DHTs' deployment is their tight control of both data placement and network topology, which makes them sensitive to failures and less likely to keep content and path locality [9]. More recently, a few studies [10, 21, 22] extend the DHT scheme to support keywords or multi-

attribute queries. The basic idea is to map each keyword to a key. A query with multiple keywords then uses the DHT to lookup each keyword and returns the intersection. To do that, large amounts of data must be transferred from one node to another, and the traffic load may be high. Systems like [23] avoid this multiple lookup and intersection by storing an object's complete keyword list on each node, but this may incur more overhead from publishing and storing the keywords.

Much of this paper draws upon the structure and evolution of small-world networks as presented by Watts and Strogatz [4]. They picture a small-world as a loosely connected set of highly connected sub-graphs, having a small average path length and a large clustering coefficient. There are various applications that use the small-world idea and partially overlap with this paper's main idea. Sripanidkulchai et al. [28] propose a mechanism that improves Gnutella's flooding-based search by adding interest-based shortcuts between nodes. A node adds shortcuts to other nodes that replied to its queries. In FLASK [15], two nodes maintain an open connection if they request at least a number of common files during a fixed time period. In [29], nodes push the advertisement of one's expertise to other nodes, and they select neighbors according to semantic properties.

## 7. CONCLUSION

Grid technology is receiving increasing attention from research communities, industries, and governments. As more and more resources appear in grids, there is an increasing need to find an effective and efficient way to discover and query these resources. This paper presents an ontology-based community architecture to optimize searches in grids. This architecture adopts a decentralized technique for identifying groups of nodes with common interests and for building overlays that mirror shared interests. It combines the efficiency and scalability of structured P2P networks with the connection flexibility of unstructured P2P networks.

## 8. REFERENCES

- [1] Chawathe, Y., Ratnasam, S., Breslau, L. Lanhan, N. Shenker, S. "Making Gnutella-like P2P Systems Scalable," In *Proceedings of ACM SIGCOMM'03*.
- [2] B.Yang, H.Garcia-Molina, "Efficient search in peer-to-peer networks," *Proc. of CDCS'02*, Vienna, Austria, July.
- [3] Banaei-Kashani, F. and C. Shahabi. "Criticality-based analysis and design of unstructured peer-to-peer networks as complex systems". *Proc. of the 3rd IEEE/ACM CCGrid*.
- [4] Watts, D.J., Strogatz, S.: Collective dynamics of 'small-world' networks. *Nature* 393 (1998).
- [5] Iamnitchi, A., Foster, I., Nurmi, D.: A Peer-to-Peer Approach to Resource Location in Grid Environments. In: *Symp. on High Performance Distributed Computing*. 2002.
- [6] Globus Toolkit: <http://www.globus.org/toolkit/>.
- [7] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, 1993.
- [8] H. Simon, *The Sciences of the Artificial*, MIT pres, 1996.
- [9] N. J.A. Harvey, M.B. Jones, S. Saroiu, M. Theimer, and A. Wolman. "SkipNet: A Scalable Overlay Network with Practical Locality Properties". In *Proc. of the Fourth USENIX USITS '03*.
- [10] S. Shi, Y. Guanwen, D. Wang, J. Yu, S. Qu and M. Chen "Making Peer-to-Peer Keyword Searching Feasible Using Multi-level Partitioning". *Proc. Of the 3rd International Workshop on Peer-to-Peer Systems*.
- [11] Gnutella website. <http://gnutella.wego.com/>.
- [12] H. Lican, W. Zhaohui, and P. Yunhe. "A scalable and effective architecture for Grid Services discovery". In *Proc. of the First Workshop on Semantics in Peer-to-Peer and Grid Computing*, 2003.
- [13] B.Yang and H.Garcia-Molina, "Designing a Super-Peer Ntrwork," *Proc. 19th Int'l Conf. Data Engineering*, Los Alamitos, CA, March 2003.
- [14] Lv, C., Cao, P., Cohen, E., Li, K., Shenker, S. "Search and replication in unstructured peer-to-peer networks". In: *ACM, SIGMETRICS 2002*.
- [15] Iamnitchi, A., *Resource Discovery in Large Resource Sharing Environments*, PhD Thesis, Department of Computer Science, The University of Chicago, 2003.
- [16] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. "Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing," *Technical Report, UCB/CSD-01-1141*, April 2000.
- [17] A. Rowstron and P. Druschel. "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems," *Proceedings of the IFIP/ACM Middleware*, 2001.
- [18] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *ACM SIGCOMM*, 2001
- [19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. "A Scalable Content-Addressable Network," *ACM SIGCOMM*, August 2001.
- [20] Adamic, L., Huberman, B., Lukose, R., Puniyani, A.: "Search in power law networks". *Physical Review* (2001)
- [21] P. Reynolds and A. Vahdat. "Efficient Peer-to-Peer Keyword Searching". In *Proceedings of ACM/IFIP/USENIX Middleware*, June 2003.
- [22] M. Cai, M. Frank, J. Chen and P. Szekely, "MAAN: A Multi-Attribute Addressable Network for Grid Information Services" *The 4<sup>th</sup> International Workshop on Grid Computing*, 2003.
- [23] C. Tang and S. Dwarkadas. "Hybrid Global-Local Indexing for Efficient Peer-to-Peer Information Retrieval". In *Proceedings of USENIX NSDI*, March 2004.
- [24] Lenat, D., and R. V. Guha, (1990), *Building Large Knowledge Based Systems*, Addison Wesley, NY.
- [25] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic indexing. *Journal of the American Society for Information Science*, 1990.
- [26] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, S.R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Hypersearching the web. *Scientific American*, 1999.
- [27] DMOZ website. <http://www.dmoz.org>.
- [28] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient content location using interest-based locality in peer-to-peer systems, In *INFOCOM'03*.
- [29] Haase, P., Siebes, R.: Peer selection in peer-to-peer networks with semantic topologies. In: *Proc. WWW2003*.
- [30] Iamnitchi A, Foster I, "On Fully Decentralized Resource Discovery in Grid Environments," *Proc. The 2nd IEEE/ACM International Workshop on Grid Computing*, 2001.
- [31] JXTA website. <http://www.jxta.org/>.