J Supercomput (2013) 66:229–248
DOI 10.1007/s11227-013-0899-6

Direction-aware resource discovery in large-scale distributed computing environments

Wu-Chun Chung · Chin-Jung Hsu · Kuan-Chou Lai · Kuan-Ching Li · Yeh-Ching Chung

Published online: 13 March 2013

Abstract As a system scales up, the peer-to-peer (P2P) approach is attractive to distributed computing environments, such as Grids and Clouds, due to the amount of resources increased. The major issue in large-scale distributed systems is to prevent the phenomenon of a communication bottleneck or a single point of failure. Conventional approaches may not be able to apply directly to such environments due to restricted queries and varied resource characteristics. Alternatively, a fully decentralized resource discovery service based on an unstructured overlay, which relies only on the information of resource attributes and characteristics, may be a feasible solution. One major challenge of such service is to locate desired and suitable resources without the global knowledge of distributed sharing resources. As a consequence, the more nodes the resource discovery service involves, the higher the network overhead incurs. In this paper, we proposed a direction-aware strategy which can alleviate the network traffic among unstructured information systems for distributed resource discovery service. Experimental results have demonstrated that the proposed approach achieves higher success rate at low cost and higher scalability.

W.-C. Chung (✉) · C.-J. Hsu · Y.-C. Chung
Dept. of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan
e-mail: wcchung@sslab.cs.nthu.edu.tw

C.-J. Hsu
e-mail: oxhead@sslab.cs.nthu.edu.tw

Y.-C. Chung
e-mail: ychung@cs.nthu.edu.tw

K.-C. Lai
Dept. of Computer Science, National Taichung University, Taichung 403, Taiwan
e-mail: kclai@mail.ntcu.edu.tw

K.-C. Li
Dept. of Computer Science and Information Engineering, Providence University, Taichung 433, Taiwan
e-mail: kuancli@pu.edu.tw

Keywords Resource discovery · Grid computing · Cloud computing · Unstructured overlay

1 Introduction

One of the essential issues in distributed computing environments is to explore and integrate the desired resources to accomplish the consigned computations and services. For example, Grid schedulers need to locate available resources to accommodate the batched computation, Cloud service providers request to discover affordable resources for service provisioning so as to provide a scalable runtime environment, etc.

In Grids [1, 2] and Clouds [3, 4], the information system plays a critical role to aggregate diverse and large-scale resource information, in which the main purpose is to provide a directory service via indexing and organizing the resource information among administrative nodes (or sites). Traditional information systems [5–7] usually adopt a centralized or a hierarchical architecture to monitor and lookup the directory of dedicated resources. These approaches can manage the global resource information easily. However, they may incur a communication bottleneck or a single point of failure problem and the system performance degrades as the scale of resources increases [8, 9].

The peer-to-peer (P2P) overlay is an abstract network that incorporates peers and logical links on top of the native or physical network topology. In such a P2P overlay, peers act as network nodes and communicate with each other through the links among any inter-connected nodes. Therefore, each peer can exploit a routing mechanism to publish or lookup the information of sharing contents or resources in a decentralized system. Regarding to the scalability and robustness, resource discovery in Grids or Clouds may apply P2P approaches to improve the system performance.

The traditional P2P techniques work well in locating exact queries based on the identifier. In the context of Grids or Clouds, queries are more complex due to a set of resource attributes with different characteristics, such as CPU speed and memory capacity, which are given for resource discovery. In addition, the status of resource attributes could change over time and the inquiring value of a desired resource attribute could be in an abstract range. Accordingly, developing an efficient resource discovery mechanism is important to facilitate the complex queries in large-scale distributed computing environments. Some novel techniques have been proposed in the literature [10–13] to support the multi-attribute range queries in distributed computing environments. Unfortunately, these structured systems impose a tight control upon the indexing information of resource locations and the connectivity of network topology. Hence, the system efficiency may be decreased under a high churn rate condition [14, 15] and the system performance may be degraded when the inquiring range of queries is large.

With the consideration of simplicity and high robustness, prior studies [16, 17] apply an unstructured P2P approach in Grids to discover the dedicated resources, and also to introduce some principles toward a generic solution for resource discovery. In such a system, the resource information is stored locally instead of being published

to a structured one. Therefore, the system does not need extra efforts to maintain a tightly coupled topology or to republish the resource information dynamically. Nevertheless, using an unstructured system may face the problem of aimlessly forwarding the query to other neighbors one after another. In addition, computing resources in distributed computing systems cannot be replicated in an arbitrary way like data files.

This study conducts a novel approach for the fully decentralized resource discovery service to resolve resource queries over an unstructured overlay. The major challenge is to locate one desired resource without any global resource information, that is, the exploration process for the desired resources highly relies on neighbors. Consequently, the heavier network traffic is produced as more nodes are involved in the discovery process. Our objective is to resolve these queries with higher efficiency and less overhead in a large scale resource sharing environment. The strategy is able to provide the direction awareness based on resource characteristics to locate the desired resources. The effectiveness of the proposed approach is evaluated through experiments with a diversity of parameters and non-uniform distribution of resource information, which show that the proposed strategies outperform conventional approaches by over 30 %, and even 40 % in some cases.

The main contributions are threefold. First, an innovative approach for the distributed resource information and discovery system comprises multiple attribute overlays, and is only built on one unstructured base overlay. Second, this paper proposes a direction-aware resource discovery strategy with the consideration of resource characteristics to alleviate the unnecessary network overhead. Finally, the proposed approach is able to locate the desired resource without any global information and to resolve the complex query over an unstructured network with higher success rate and lower cost.

The remainder of this paper is organized as follows. Related works and correlated studies are discussed in Sect. 2, while Sect. 3 presents the decentralized resource information and discovery system and the corresponding mechanisms. Experimental results are given in Sect. 4, and finally, conclusions and future research directions are summarized in Sect. 5.

2 Related work

As the amount of distributed resources scales up, scalable and robust architectures are carried out for the resource management in distributed computing systems such as Grids and Clouds. An information system ought to adopt a decentralized approach to overcome the shortcoming of centralized or hierarchical approaches. Therefore, the distributed resource discovery is a vital service for such systems to fulfill user needs.

In the past years, attention has been given to the convergence of Grid systems and P2P approaches [18–20], including also several decentralized approaches to index and discover resource information in a peer-to-peer manner [21, 22]. However, resource discovery in the context of Grid and Cloud computing resources usually faces a threefold challenge. First, resources are diverse and queries are complex. Second, the characteristics of resource attributes include static attributes and dynamic

ones whose values are varied at any time. Finally, a request may inquire an abstract range of values for the desired resources. Consequently, novel solutions come out at developing the decentralized resource discovery for such distributed computing environments.

Some studies, as those presented in MAAN [11], Mercury [10], SWORD [12], and Squid [13], adopt the structured overlay and improve the capability of multi-attribute range queries for discovering resources in distributed computing systems. In such systems, each node (or peer) is responsible for maintaining a directory index containing a subset of resource information from other nodes. These techniques strictly regulate the network topology and manage the placement of the published resource information. Hence, as the value of a dynamic attribute is changed, the resource status has to be updated and its latest state has to be re-published to the structured information service. The more information a system needs to update, the higher the total cost this system will pay. Hence, the maintenance of structure approaches is considerably expensive.

In contrast to the tightly structured system, some loosely controlled systems construct an unstructured network to link the participating peers (e.g., Gnutella and KaZaA). Such unstructured systems have no global knowledge about the network topology or resource locations. Thus, each node blindly floods or randomly forwards a request to its neighbors, so incurring large network overhead. In this way, several studies [23–25] improve the search of requested data in unstructured P2P networks. However, conventional P2P approaches miss to consider the characteristics of computing resources and cannot be applied directly to the context of an information system in Grid computing [17]. Accordingly, a new paradigm for resource discovery in distributed computing systems is presented.

A. Iamnitchi and I. Foster [16, 17] first investigate the design of a fully decentralized approach to tackle the resource location problem in Grid environments, in which they are focused on the request processing and several request propagation schemes to locate the desired resources. Four axes of architectural components have been considered on a general resource discovery solution, namely, the membership protocol, the overlay construction, the preprocessing, and the request processing. Another research study [26] further improves the performance of resource utilization and turnaround time to discover more resources in a fully decentralized resource discovery service. The main idea of previous studies is to forward a request with or without the extra information stored on nodes. In case of exploiting the extra information, a request is propagated based on the histogram information; otherwise, a request is forwarded to the uninformed node randomly chosen from its neighbors. However, these forwarding schemes do not consider the attribute type and the value of resource characteristics.

In a Grid or a Cloud system, each query can be resolved according to the requirement of resource attributes and corresponding values so that each node can forward the request to a more profitable neighbor. This paper considers a flat, fully decentralized resource discovery service based on the unstructured overlay. In this way, each node only records its own resource information instead of publishing the information to the tightly controlled network or other nodes. Therefore, there is no extra overhead to maintain the strictly structured topology and republish the information to corresponding nodes.

However, using the pure flooding in an unstructured network may involve additional nodes that are irrelevant during the discovery process. Hence, a large amount of routing messages is ineffectual, so the system performance will deteriorate when the scale of the network size gets larger. In order to eliminate the drawback of flooding, a random walk search [27] is considered to reduce the overall messages, in which each node only forwards the request to one of its neighbors in each step. Consequently, the request is randomly propagated from node to node in the overlay network. The coverage of visitors could be lower to alleviate the number of routing messages. However, the pure random-selection scheme aimlessly visits the nodes without guided directions.

Therefore, this research concentrates on the proposal of a direction-aware resource discovery scheme based on resource characteristics, to improve the performance of conventional approaches. The overall message cost can be reduced in distributed computing systems, alleviating the network overhead.

3 Resource information and discovery system

This section describes how to exploit the unstructured overlay to constitute the distributed resource sharing environment, and addresses next how the proposed strategy discovers the desired resources according to the queries and resource characteristics.

3.1 System overview

This paper considers a distributed system (e.g., a Grid or a Cloud) consisting of multiple information systems, and each information system organizes its own resources within the geographical location and manages the information directory of local resources. Thereafter, an information system acts as a node participating in an unstructured network to form the distributed resource sharing environment. In such a fully decentralized environment, each node has no global knowledge about resources shared in other nodes. That is, each node is only aware of its directly connected neighbors in the distributed system. To approach the flat and fully decentralized resource discovery, a node contacts other nodes as needed to locate the desired resources fulfilling the requested requirement.

Figure 1 depicts a logical view of the system architecture for a network of nodes connected with corresponding neighbors. The system is composed of one base overlay and multiple attribute overlays. The base overlay is constructed by an unstructured overlay acting as the foundation network of the system. Upon the base overlay, the attribute overlays represent the categories of resource information. Each node constructs an attribute overlay for each type of resource information according to the neighborhood in the unstructured overlay. Specifically, the neighboring nodes of a node in each attribute overlay are the same, but the request forwarding routes may be different in distinct attribute overlays.

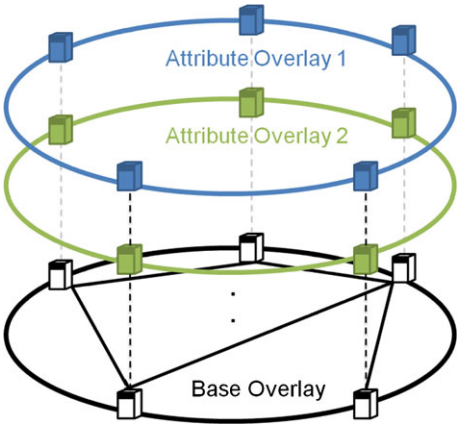

Fig. 1 Logical view of a system architecture

3.1.1 Membership

For the initialization procedure, each node builds up the membership in the base overlay. First, a new node participates in the unstructured overlay by establishing connections with other existing nodes. Precisely, the node contacts a set of existing K nodes which are selected by a bootstrapping node to provide the initial information for a joining node. Through these existing nodes, a new node can successfully participate in the unstructured overlay and become a member in the resource sharing system. Hereafter, the new node constitutes a neighboring set of neighbors and exchanges the information of local resource status with each other.

In this system, each node shares the same category of resource attributes, though the attribute value of each attribute status may be different. That is, different resource providers have a similar policy to share their dedicated resources unselfishly, but different sharing resources may have different capacities and capabilities. Moreover, the procedure of exchanging information with neighbors does not bringing extra overhead for the network traffic because the update information is possible to be piggybacked in the operation of overlay maintenance. Therefore, when the status of local resource information is changed, the corresponding node only needs to update the resource status to its neighbors, instead of republishing the information to the strictly controlled indexing service.

3.1.2 Neighbor set

Based on the resource information exchanged from neighbors, each node estimates the values of each attribute between its local resources and neighbors' resources. Without loss of generality, we are able to encode the attribute value of each computing resource into a comparable sequence, e.g., the numeric value or the bit string. Based on the compared results, each node classifies its neighbors into the large-neighbor set (denoted as L) and the small-neighbor set (denoted as S).

Let NE_i denote the set of neighbors of a node n_i; f_a denote a function that extracts the value of a particular attribute a in the resource information of a node or in the

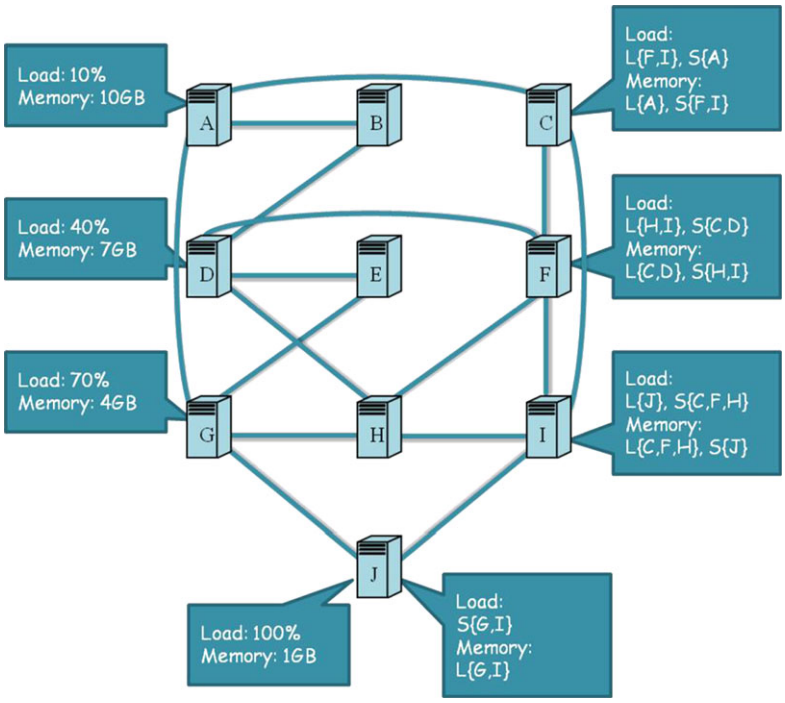

Fig. 2 Example of a decentralized information system based on an unstructured overlay network

requirement of a query. Thus, for each attribute a, n_i is able to classify the NE_i into L_a or S_a, such that

$$\begin{aligned} L_a &= \{n_l | n_l \in NE_i, f_a(n_l) \geq f_a(n_i)\} \\ S_a &= \{n_s | n_s \in NE_i, f_a(n_s) < f_a(n_i)\}. \end{aligned} \tag{1}$$

Thereupon, if the retrieved attribute value of a neighbor is equal to or larger than that of the local information, the neighbor is classified into the large-neighbor set; otherwise, classified as small-neighbor set. Notably, let n_z be one node in NE_y of node n_y, and n_y be one node in S_a of node n_x, then $f_a(n_z)$ may not be smaller than $f_a(n_x)$, and similarly in the large-neighbor set. This happens because the participating nodes in our system are organized using an unstructured overlay rather than an ordered network topology.

Figure 2 gives an example of an unstructured information system to illustrate our approach. Two types of resource attributes are chosen as examples: the system load and the memory capacity. Suppose that the attribute value of system load ranges from 10 % to 100 % and is assigned to each node from n_A to n_J incrementally. The attribute value of memory capacity ranges from 1 to 10 GB and is decrementally assigned to each node from n_A to n_J. Based on the membership management, each node exchanges the resource information with neighbors and categorizes its neighbors into the large-neighbor and the small-neighbor sets. For example, taking the resource attribute of memory capacity, the L_{memory} of n_F includes n_C and n_D while the S_{memory} includes n_H and n_I.

3.2 Direction-aware resource discovery

The investigation into the unstructured information system is carried out to conduct an efficient resource discovery service to route queries over the network. In such an information system, the generic procedure for each node to discover the required resource is to lookup the local information first. If there is no local resource satisfying the requirement, the node forwards the request to other neighbors and inquires their local indexing information for the required resource. After receiving such a request, each neighbor also looks up its local resource and then replies to the requesting node in case the matching resource is found; otherwise, the request is propagated to other neighbors until the required resource is found or a stop condition is reached.

Under this perspective, the way of selecting profitable nodes to forward the request is a key issue. Two classic routing strategies are widely used for unstructured P2P systems to locate desired resources. One is the flooding and the other is the random walk. For the flooding-based scheme, each node contacts all its neighbors for the desired resources, while the random-walk-based scheme forwards the request to a randomly chosen node in each step. Consequently, the more irrelevant nodes are involved in the routing scheme, the higher network overhead may be incurred.

3.2.1 Mechanism overview

In the proposed system, each resource is composed of a variety of characteristic attributes, such as system load, memory capacity, among others. Resource discovery in such resource sharing environments needs to resolve the queries with a set of pairs of desired attributes and values. Accordingly, the attribute-based routing strategy is preferred to resolve the complicated query and locate the desired resource with satisfied requirement.

Let A_Q denote the set of attributes in the requirement of a query Q, A_I denote the set of attributes in the local resource information I, and AO_a denote the overlay of an attribute a based on unstructured network. A multi-attribute query is passed to one AO_i to discover the adequate resource, where the attribute i is able to be randomly chosen from A_Q or be the primary attribute defined by users. Since each node has the complete information of its local resource, the query can be resolved in any one of the attribute overlay the node has joined.

The main goal of alleviating network cost is to reach fewer nodes. Since each node in our system manages its geographical resource information, a node can exploit the information of request and resource to resolve the query efficiently. With such a type of resource information, a direction-aware resource discovery for unstructured information systems is proposed. Forwarding a query relies on the compared result between the attribute value of required resource and that of local indexing resource. The principle of direction-aware routing (DAR) strategy is given as

$$\begin{cases} L_a, & \text{if } f_a(A_Q) \geq f_a(A_I) \\ S_a, & \text{if } f_a(A_Q) < f_a(A_I). \end{cases} \tag{2}$$

That is, a request is oriented to the node within the small-neighbor set if the attribute value of the required resource is smaller than that of the local indexing resource; otherwise, the request is oriented to a node within the large-neighbor set.

The proposed DAR strategy can also handle the range query for each attribute, provided that the value of a resource attribute is given in an abstract range. The common categories of a given abstract range can be inductive into two types of operators: the relational operators (i.e., $>$, $=$, $<$) and the logical operators (i.e., AND, OR). For the simple range query with only one type of relational operators, the resource discovery mechanism can make the forwarding decision as discussed in Eq. (2). On the other hand, for complex range queries, the abstract range could be given with both a relational operator and a logical operator. For instance, the value of a desired memory capacity is given in a range from 4 to 8 GB.

To deal with the complex range query, a straightforward way is to divide the query into two simple sub-queries. One sub-query is issued to discover the resource which $f_{\text{memory}}(A_I)$ is larger than 4 GB and the other is issued to locate the resource which $f_{\text{memory}}(A_I)$ is smaller than 8 GB. After receiving the results of each sub-query, the results are merged and then intersection is applied to extract the final list of matched resources. However, it is easily noted that the network traffic is doubled. An alternative in the proposed system is to forward the whole range query to neighbors in any one of attribute overlay, since each node maintains the geographic information about its local indexing resource. Therefore, the resource discovery mechanism is able to make forwarding decision according to

$$\begin{cases} L_a, & \text{if } f_a(\text{Min\_}A_Q) \geq f_a(A_I) \\ S_a, & \text{if } f_a(\text{Max\_}A_Q) < f_a(A_I), \end{cases} \tag{3}$$

where $\text{Max\_}A_Q$ and $\text{Min\_}A_Q$ represent the upper and the lower bounds of the abstract range of desired attribute values, respectively. Accordingly, a request is forwarded to the node selected from the small-neighbor set when the attribute value of the upper bound for the required resource is smaller than that of the local indexing resource; otherwise, the request is forwarded to a node selected from the large-neighbor set.

Based on the principle of the DAR strategy, each query can be forwarded to next nodes according to the decision making. To detail the approach, we exploit two classic schemes to describe the proposed strategies in subsequent sections.

3.2.2 Direction-aware strategy with flooding

In general, the conventional flooding scheme floods a query to all neighboring nodes for each node in a system. The more nodes the scheme involves, the more requested messages there will be, leading to the increase of network traffic. Our work aims to alleviate the disadvantage by contacting fewer useless nodes to locate dedicated resources. Here, we present a novel strategy named direction-aware routing with flooding (DAR-FLOOD) to prevail the opinion.

Since each node has the information of its own resource attributes, a node can make a forwarding decision based on comparison of attribute values between its neighbors and itself. When the attribute value of a desired resource is smaller than that of the local resource, the node floods the request to those nodes in its small-neighbor set; otherwise, the node contacts those nodes in the large-neighbor set. As for the exception case, a node may miss one of the alternative neighbor sets. This

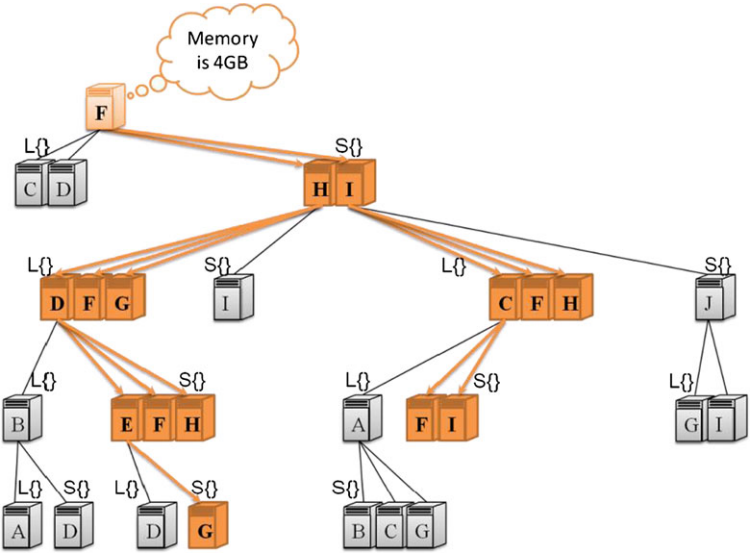

Fig. 3 Example of resource discovery visualization with direction-aware flooding

happens due to those neighbors being randomly selected without restricted control in the unstructured network, so that the value of a certain node may be the largest or the smallest among its all neighbors. In this case, the node can decide to contact the opposite neighbor set.

Figure 3 depicts a simple example of direction-aware resource discovery with flooding. To ease the understanding, we use a tree hierarchy to visualize the situation. Suppose that a query is issued to n_F, which needs to locate the desired resource with the specific value of memory capacity. Since the local resource of n_F is not qualified for such a request, it makes the flooding decision according to the compared result of attribute values between the capacity of the desired resource and the local resource of n_F. In this example, the value of desired resource is smaller than the attribute value of local resource. Thus, n_F makes a decision to flood the request to those nodes n_H and n_I in its small-neighbor set, instead of flooding to all neighbors n_C, n_D, n_H, and n_I. Each node processes the request following the previous principle and stops the request forwarding once the request has been handled by itself or a dedicated resource is found, which in this case is located at n_G. Accordingly, the resource discovery mechanism with direction awareness can flood the request to the subset of nodes that are more profitable, instead of aimlessly flooding to all nodes that may be irrelevant.

3.2.3 Direction-aware strategy with random walk

A random walk routes the request with a walker which takes a trip to only a randomly chosen node in each step to alleviate the network traffic significantly when compared to the pure flooding-based scheme. However, using the random walk approach could extend the routing path if the walker visits more useless nodes, i.e., increasing futilitarian messages. In order to overcome this inefficiency, each walker is expected to be

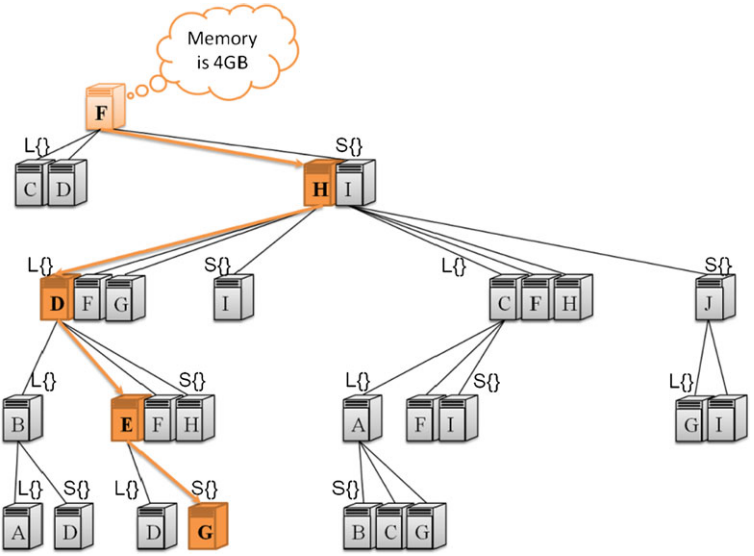

Fig. 4 Example of resource discovery visualization with direction-aware random walk

oriented to desired resources. And so, we came up with the direction-aware routing with random walk (DAR-RW) to the target.

Similar to DAR-FLOOD approach, the DAR-RW strategy also makes forwarding decisions depending on the compared result of attribute values. When the attribute value of the desired resource is smaller than that of the local resource, the walker decides to walk to one randomly chosen node from the small-neighbor set of the currently visited node; otherwise, the walker randomly chooses a node from the large-neighbor set of the currently visited node. As for the special case, the currently visited node may miss one of the neighbor sets, so that the walker may have no sense to guide the next direction to the desired resource. In this case, the walker can randomly choose a node from the opposite neighbor set, to keep exploring more candidate neighbors.

Figure 4 shows an example that illustrates our proposed approach. Since the local resource of n_F does not match the request, a node needs to make the forwarding decision according to the compared result of resource characteristics between the request and the local resource of n_F. In this case, the required resource capacity is smaller than the resource capacity of inquired n_F. When n_F forwards the request to the next node, it randomly selects a node only from the small-neighbor set of $\{n_H, n_I\}$, and not from the set of all neighbors $\{n_C, n_D, n_H, n_I\}$. According to this principle, each node forwards the request to the next profitable node until the desired resource is found or the stop condition is satisfied. Based on the proposed direction-aware resource discovery with the random walk, the request forwarding can be oriented to more relevant nodes, when compared to the aimless walking.

3.3 Maintenance overhead

In our proposed system, each node participates in the unstructured overlay by contacting a list of existing nodes. Based on the list, each node identifies its neighbors according to the resource information and the corresponding values of attributes. More precisely, each node classifies its neighbors into the set of L or S. Our approach tries to provide a tourist guide to route the request relying only on the information of resource characteristics. Based on the set of L and S, each node can seek a node either with the largest or the smallest value, and vice-versa. In other words, there exists at least one path from the node with the largest or smallest value to any node in the unstructured network. By implication, a desired resource can be located eventually if it exists. Accordingly, each node is able to guide a query to the target with a higher possibility of matched query at a lower cost over a fully decentralized resource information and discovery system.

Furthermore, to provide the robustness to a system, the proposed approach needs to consider the maintenance of membership and information update. The system architecture is composed of one base overlay and multiple attribute overlays. In such an environment, each attribute overlay is constructed upon the base overlay. Accordingly, we only need to take care of the maintenance in the underlying base overlay. Exploiting an unstructured network to construct the base overlay is beneficial due to the simplicity of the network topology and loose control on the information location. Therefore, the system robustness highly relies on the basic operations of the overlay maintenance.

In the unstructured overlay, each new node can easily join or leave the overlay by contacting the bootstrapping node. Then, each node can update its membership with neighbors in the case of normal node dynamics. To detect a node failure, a periodically keep-alive message can be used to exchange the status with neighbors. Accordingly, based on the overlay maintenance, each node can not only maintain its membership but also renew its neighbor set according to the piggybacked resource information from neighbors. That is, the maintenance cost for each node is limited to the number of neighbors when applying this proposed approach.

4 Performance evaluation

In this section, the performance of the proposed approaches is evaluated. We describe the evaluation methodology and the primary metrics of interest in our study, and following next are the experimental results of the proposed strategy.

4.1 Evaluation methods

We implemented a cycle-based simulator to evaluate the effectiveness of the proposed approach in a decentralized Grid environment. In this experimental environment, each information system acts as a node to participate in the unstructured overlay by contacting a list of existing nodes. We refer to the category of resource type in Ganglia [7] to assign the resource attributes for each node. Each node is assigned with the same categories of resource attributes, but with varied values of the same attribute.

The resource discovery module is developed to apply various forwarding mechanisms to locate the desired resource. For each round, a query requests the desired resource attributes with a randomly assigned value for each attribute, whereas a primary attribute of the requirement is also given for the query. The performance results are averaged over multiple rounds. Several scenarios are taken into consideration and simulated as experiments. First, the query with randomly assigned values is issued to the resource discovery service. Second, we are concentrated on the performance when the query is conducted in highly skewed distribution of resource information. The effectiveness of both uniform and non-uniform distributions is evaluated to locate the desired resource. We also investigate the performance impacts by varying the numbers of network nodes, neighbors, and TTL (Time-To-Live).

For the sake of estimating the performance, the experiment mainly focuses on three performance metrics: routing cost, number of routing hops, and success rate. The routing cost is evaluated by the amount of request messages forwarded among nodes while the number of routing hops is computed as the length of the shortest routing path from the initiator to the resource provider. As expected, the more nodes the resource discovery strategy covers, the more routing messages and the longer routing path there will be. On the other hand, the success rate is the percentage of success to locate at least one desired resource that is satisfied with the query. Under the given stop condition, a higher success rate stands for more efficient resource discovery approach.

4.2 Experimental results

Experimental results are presented for the proposed direction-aware resource discovery service among Grid information systems. Below, the Blind-FLOOD represents the pure flooding strategy that blindly floods the query to all neighbors for each step. On the other hand, the Blind-RW represents the conventional random walk strategy to route a request to the node that is randomly chosen from all neighbors.

4.2.1 Routing cost

First of all, the performance with respect to the average number of messages is checked in networks of varied size (N). In this experiment, the number of neighbors is set to be 20 for each node. As depicted in Figs. 5 and 6, the larger the number of nodes involved in the system, the more messages the routing scheme produces. The resource discovery mechanism using the flooding-based approaches yields a larger amount of messages to discover the desired resource than that using the random-walk-based ones. On the other hand, the proposed direction-aware routing strategies show better performance on average by 33.48 % and 43.58 % than the conventional blind flooding strategy and blind random-walk strategy, respectively. This observation is due to fact that the proposed approach considers only a subset of nodes that are more profitable. Particularly, the proposed strategies slowly raise the curve of average cost when the number of nodes increases.

Next, the effectiveness of routing cost is evaluated with different numbers of neighbors. In this experiment, we estimate the average cost of messages with a varied number of neighbors (K) and set up the network size to be 3000. Figures 7 and 8

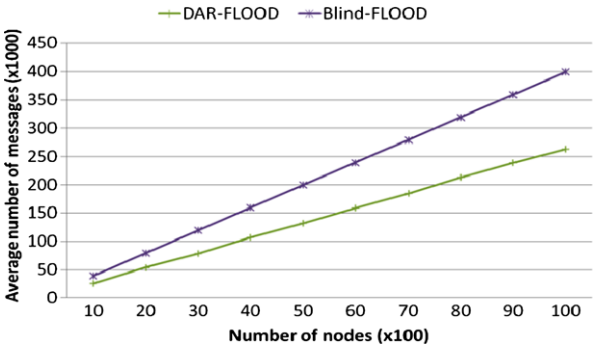

Fig. 5 Average number of messages vs number of nodes using the flooding ($K = 20$)

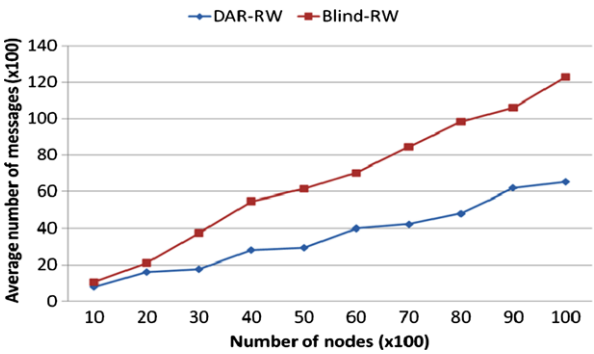

Fig. 6 Average number of messages vs number of nodes using the random walk ($K = 20$)

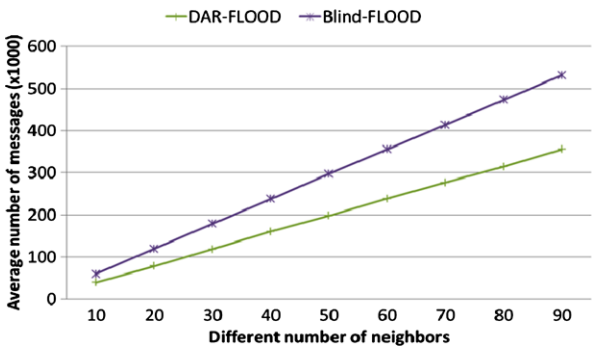

Fig. 7 Impact of varied number of neighbors on facilitating the flooding ($N = 3000$)

depict different phenomena. Regarding the flooding approaches, the more neighbors the strategy needs to cover, the higher the number of messages there will be. As in Fig. 7, both routing strategies increase the number of messages when the number of neighbors gets larger. However, the proposed strategy grows slowly and performs over 30 % better than the blind flooding one. That happens due to the proposed strategy only flooding the request to a subset of more useful neighbors with direction awareness.

Regarding the random walk approaches, the number of neighbors does not seriously degrade the system performance since the walker only selects a random node from the neighbor set to forward the request. Figure 8 illustrates the notion and shows

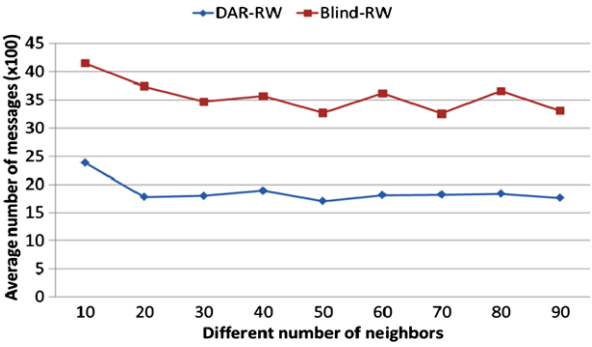

Fig. 8 Impact of varied number of neighbors on facilitating the random walk ($N = 3000$)

that the number of neighbors does not dramatically affect the average message cost for both random-walk-based strategies. Our direction-aware strategy outperforms the conventional routing strategy by approximately 50 % and is more scalable.

4.2.2 Routing hops

Figures 9 and 10 present the performance metric of routing hops. As shown in the experimental results, resource discovery approach with the flooding-based scheme outperforms that with the random-walk-based strategy. With regard to the random-walk-based strategy, the resource discovery approach with direction awareness performs more efficiently.

In the case of the flooding-based scheme, the effectiveness of our proposed direction-aware routing strategy is similar to the conventional approach. That is because the flooding-based mechanisms cover the most of neighboring nodes to locate the desired resource and find a routing path as short as possible. In our proposed approach, the DAR-FLOOD scheme can only cover a subset of neighbors to locate the desired resource; however, the Blind-FLOOD scheme will cover both large- and small-neighbor sets. That is, the shortest routing path in the DAR-FLOOD scheme could be also found in the Blind-FLOOD one. Thus, the average length of a routing path to locate the same resource between these two schemes is similar. However, as in previous experiments, the pure flooding-based scheme is more costly to achieve higher efficiency. On the contrary, the proposed approach can achieve higher efficiency with lower cost.

4.2.3 Non-uniform distribution

In this experiment, the effectiveness of the proposed approach is evaluated in a highly skewed distribution of resource information. The Zipf ($\alpha = 0.95$) distribution is adopted to simulate an environment with a non-uniform distribution of attribute values. The network size is set to be 10000 and the number of neighbors is set to be 20 for each node. Different performance metrics are evaluated under the varied TTL in this experiment. For each round, the query is issued to locate the desired resource with a randomly given value. The resource discovery process will stop the request forwarding either if the desired resource is found or the TTL is expired.

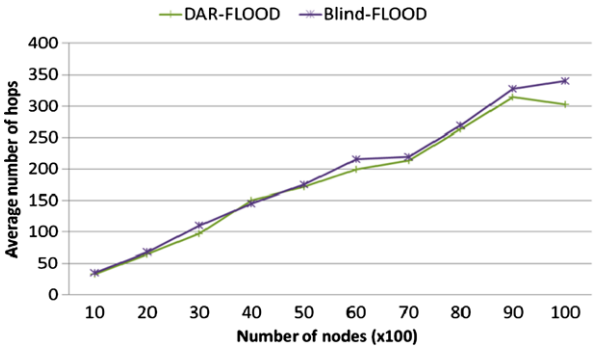

Fig. 9 Average number of hops vs number of nodes using the flooding ($K = 20$)

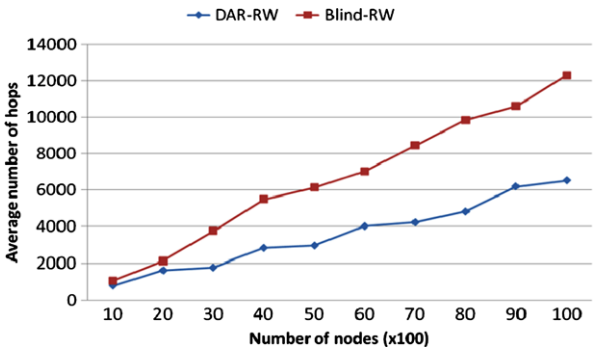

Fig. 10 Average number of hops vs number of nodes using the random walk ($K = 20$)

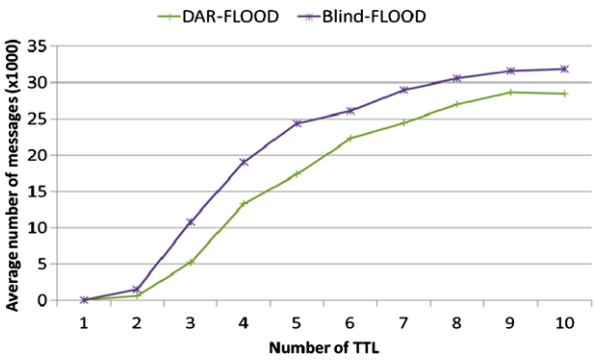

Fig. 11 Average number of messages vs TTL using the flooding ($K = 20$, $N = 10000$)

Regarding the routing cost, as depicted in Figs. 11 and 12, the average number of messages increases as the TTL gets larger. Deservedly, the larger the TTL is assigned to the resource discovery process, the more nodes the routing scheme covers, so incurring more messages to the system. Experimental results show that our proposed strategies present better performance on average by 26.6 % and 22.9 % compared with the blind flooding strategy and blind random-walk strategy, respectively.

As observed in the experiment, under the same stop condition, the resource discovery mechanism using the flooding-based approach yields more messages than the

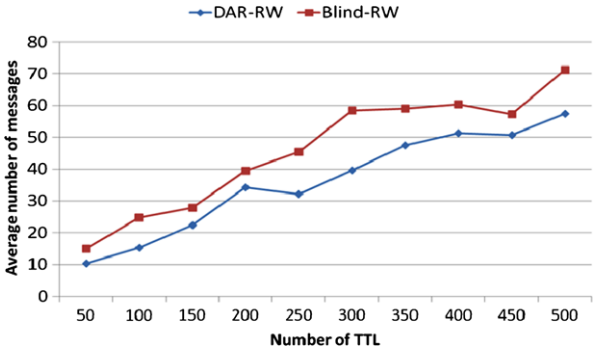

Fig. 12 Average number of messages vs TTL using the random walk ($K = 20$, $N = 10000$)

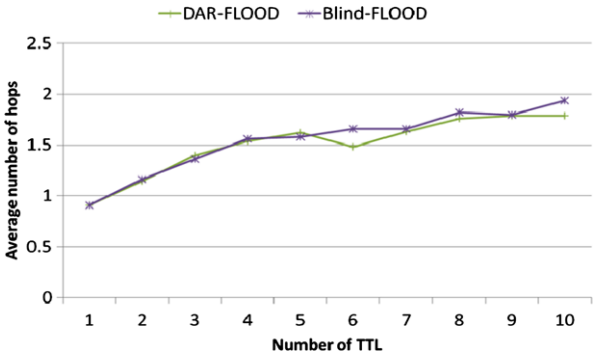

Fig. 13 Average number of hops vs TTL using the flooding ($K = 20$, $N = 10000$)

random-walk-based approach to locate the desired resource. However, the flooding-based approach is able to achieve fewer routing hops than the random-walk ones under the same TTL. In particular, as depicted in Fig. 13, performance results show that the effectiveness with respect to the routing hops for the proposed direction-aware flooding is similar to the blind flooding approach. That is, the conventional flooding approach has a higher routing cost to provide high efficiency. On the contrary, our proposed strategy can achieve high efficiency with low cost.

On the other hand, the performance with respect to the success rate is also evaluated in the experiment as depicted in Figs. 14 and 15. Experimental results show that the flooding-based approach can achieve higher success rate with a smaller TTL, while the random-walk-based approach needs a larger TTL to achieve higher success rate. Figure 14 illustrates that both flooding-based approaches can locate the desired resource with high probability. In other words, as shown in previous experiments, our direction-aware strategy can achieves high success rate to discover the resource with low overhead. Moreover, Fig. 15 shows that both random-walk-based approaches need a larger TTL to locate the desired resource. As the TTL gets larger, the success rate increases. Additionally, our proposed strategy can achieve a higher success rate with a smaller TTL than the conventional random walk.

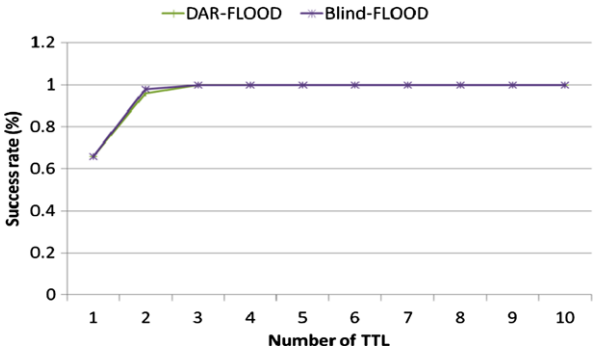

Fig. 14 Success rate vs TTL using the flooding ($K = 20$, $N = 10000$)

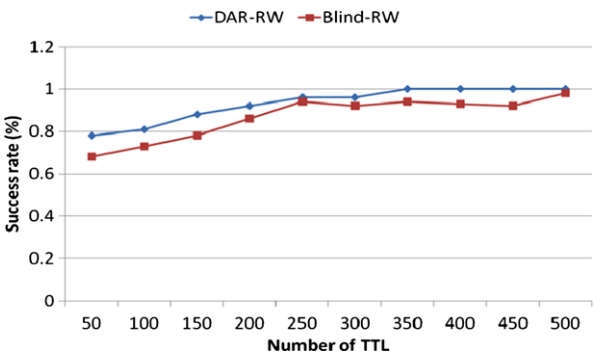

Fig. 15 Success rate vs TTL using the random walk ($K = 20$, $N = 10000$)

5 Conclusions and future work

One of most essential services in large-scale distributed computing systems, such as Grids and Clouds, is to index the scattered resource information and discover the dedicated resources. A flat, unstructured resource information and discovery system is carried out in this research. To provide an efficient resource discovery service, this paper introduces the direction-aware strategies to resolve resource queries among the unstructured information systems. The proposed approach not only reduces the amount of unnecessary messages, but also locates the desired resource with a higher success rate at low cost and high scalability. The effectiveness of system performance is evaluated through a diversity of parameters and the non-uniform distribution of resource information. Experimental results show that the proposed approach is able to outperform the blind flooding and random-walk schemes by over 30 % and 40 %, respectively.

As for directions for future research, developing a comprehensive system comprising the resource information monitoring and the service provisioning is fascinating. An interesting issue is to handle the dynamic information update. There is a trade-off between the information accuracy and the network cost for sustaining the latest resource status. Based on the resource information retrieving protocol [28], the distributed resource sharing system could achieve a high degree of information accuracy with a low cost of updating messages. Another considerable research is to exploit the resource characteristics to benefit other informed approaches to discover the desired

resource in such a distributed computing system. Our strategy can be further applied to the emerging Cloud platform to provide an efficient resource information and discovery service.

Acknowledgements The authors would like to express their gratitude to the anonymous reviewers for their reviews and suggestions. This paper is partially supported by Delta Electronics, Inc. under grant 101F2289A8.

References

1. Chervenak A, Foster I, Kesselman C, Salisbury C, Tuecke S (2000) The data grid: towards an architecture for the distributed management and analysis of large scientific datasets. J Netw Comput Appl 23(3):187–200
2. Foster I, Kesselman C, Tuecke S (2001) The anatomy of the grid: enabling scalable virtual organizations. Int J High Perform Comput Appl 15(3):200–222
3. Marinos A, Briscoe G (2009) Community cloud computing. In: Proceedings of the 1st international conference on cloud computing, Beijing, China, pp 472–484
4. Dikaiakos MD, Katsaros D, Mehra P, Pallis G, Vakali A (2009) Cloud computing: distributed Internet computing for IT and scientific research. IEEE Internet Comput 13(5):10–13
5. Fitzgerald S, Foster I, Kesselman C, Laszewski G, Smith W, Tuecke S (1997) A directory service for configuring high-performance distributed computations. In: Proceedings of the 6th IEEE international symposium on high performance distributed computing, Portland, OR. IEEE Press, New York, pp 365–375
6. Czajkowski K, Fitzgerald S, Foster I, Kesselman C (2001) Grid information services for distributed resource sharing. In: Proceedings of the 10th IEEE international symposium on high performance distributed computing, San Francisco, CA, USA. IEEE Press, New York, pp 181–194
7. Massie ML, Chun BN, Culler DE (2004) The ganglia distributed monitoring system: design, implementation, and experience. Parallel Comput 30(7):817–840
8. Zhang X, Freschl JL, Schopf JM (2003) A performance study of monitoring and information services for distributed systems. In: Proceedings of the 12th IEEE international symposium on high performance distributed computing, Washington, DC, USA, pp 270–281
9. Mastroianni C, Talia D, Verta O (2008) Designing an information system for grids: comparing hierarchical, decentralized P2P and super-peer models. Parallel Comput 34(10):593–611
10. Bharambe AR, Agrawal M, Seshan S (2004) Mercury: supporting scalable multi-attribute range queries. In: Proceedings of the 2004 conference on applications, technologies, architectures, and protocols for computer communications, Portland, Oregon, USA. ACM Press, New York, pp 353–366
11. Cai M, Frank M, Chen J, Szekely P (2004) MAAN: a multi-attribute addressable network for grid information services. J Grid Comput 2(1):3–14
12. Oppenheimer D, Albrecht J, Patterson D, Vahdat A (2004) Scalable wide-area resource discovery. Technical Report UCB/CSD-04-1334, EECS UoC, Berkeley
13. Schmidt C, Parashar M (2008) Squid: enabling search in DHT-based systems. J Parallel Distrib Comput 68(7):962–975
14. Gummadi KP, Dunn RJ, Saroiu S, Gribble SD, Levy HM, Zahorjan J (2003) Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In: Proceedings of the 9th ACM symposium on operating systems principles, Bolton Landing, NY, USA. ACM Press, New York, pp 314–329
15. Rhea S, Geels D, Roscoe T, Kubiatowicz J (2004) Handling churn in a DHT. In: Proceedings of the annual conference on USENIX annual technical conference, Boston, MA. USENIX Association, Berkeley, pp 127–140
16. Iamnitchi A, Foster I (2001) On fully decentralized resource discovery in grid environments. In: Proceedings of the second international workshop on grid computing, Denver, CO, pp 51–62
17. Iamnitchi A, Foster I (2004) A peer-to-peer approach to resource location in grid environments. In: Nabrzyski J, Schopf JM, Weglarz J (eds) Grid resource management. Kluwer Academic, Dordrecht, pp 413–429
18. Foster I, Iamnitchi A (2003) On death, taxes, and the convergence of peer-to-peer and grid computing. In: The 2nd international workshop on peer-to-peer systems, Berkeley, CA, pp 118–128

19. Talia D, Trunfio P (2003) Toward a synergy between P2P and grids. IEEE Internet Comput 7(4):95–96
20. Chung W-C, Hsu C-J, Lin Y-H, Lai K-C, Chung Y-C (2010) G2G: a meta-grid framework for the convergence of P2P and grids. Int J Grid High Perform Comput 2(3):1–16
21. Trunfio P, Talia D, Papadakis H, Fragopoulou P, Mordacchini M, Pennanen M, Popov K, Vlassov V, Haridi S (2007) Peer-to-peer resource discovery in grids: models and systems. Future Gener Comput Syst 23(7):864–878
22. Ranjan R, Harwood A, Buyya R (2008) Peer-to-peer-based resource discovery in global grids: a tutorial. IEEE Commun Surv Tutor 10(2):6–33
23. Kalogeraki V, Gunopulos D, Zeinalipour-Yazti D (2002) A local search mechanism for peer-to-peer networks. In: Proceedings of the 11th international conference on information and knowledge management, McLean, Virginia, USA. ACM Press, New York, pp 300–307
24. Yang B, Garcia-Molina H (2002) Improving search in peer-to-peer networks. In: Proceedings of the 22nd international conference distributed computing systems, Vienna, Austria, pp 5–14
25. Filali I, Huet F (2010) Dynamic TTL-based search in unstructured peer-to-peer networks. In: Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing, Melbourne, VIC, Australia, pp 438–447
26. Tangpongprasit S, Katagiri T, Kise K, Honda H, Yuba T (2005) A time-to-live based reservation algorithm on fully decentralized resource discovery in grid computing. Parallel Comput 31(6):529–543
27. Lv Q, Cao P, Cohen E, Li K, Shenker S (2002) Search and replication in unstructured peer-to-peer networks. In: Proceedings of the 16th international conference on supercomputing, New York, USA. ACM Press, New York, pp 84–95
28. Chung W-C, Chang R-S (2009) A new mechanism for resource monitoring in grid computing. Future Gener Comput Syst 25(1):1–7