Resource discovery in grid systems: a survey

Abdelkader Hameurlain*

Institut de Recherche en Informatique de Toulouse IRIT,
Paul Sabatier University,
118, Route de Narbonne,
31062 Toulouse Cedex, France
E-mail: hameur@irit.fr
*Corresponding author

Deniz Cokuslu

Institut de Recherche en Informatique de Toulouse IRIT,
Paul Sabatier University,
118, Route de Narbonne,
31062 Toulouse Cedex, France

and

Izmir Institute of Technology,
Department of Computer Engineering,
Gulbahce, Urla, 35430 Izmir, Turkey

and

International Computer Institute
Ege University, Bornova,
35100 Izmir, Turkey
E-mail: denizcokuslu@iyte.edu.tr

Kayhan Erciyes

International Computer Institute
Ege University, Bornova,
35100 Izmir, Turkey
E-mail: kayhan.erciyes@ege.edu.tr

Abstract: Resource Discovery (RD) is one of the key issues in successful Grid systems. Yet, new methodologies for RD are constantly researched owing to the dynamicity, heterogeneity and large-scale characteristics of Grids. Recently, synergy and convergence between Grid, Agent and Peer-to-Peer (P2P) systems were pointed out clearly. This paper provides a survey and a qualitative comparison of the most promising approaches (P2P techniques and agent systems) for RD. Viability of Grid systems relies mainly on efficient integration of P2P techniques and mobile agent (MA) systems to bring scaling and decentralised control properties to Grids.

Keywords: grid systems; resource discovery; P2P techniques; mobile agents.

Reference to this paper should be made as follows: Hameurlain, A., Cokuslu, D. and Erciyes, K. (2010) ‘Resource discovery in grid systems: a survey’, *Int. J. Metadata, Semantics and Ontologies*, Vol. 5, No. 3, pp.251–263.

Biographical notes: Abdelkader Hameurlain is Full Professor in Computer Science at Paul Sabatier University, Toulouse, France. He is a member of the Institut de Recherche en Informatique de Toulouse (IRIT). His current research interests are in query optimisation in parallel and large-scale distributed environments, mobile databases and database performance. He has been the general chair of the International Conference on Database and Expert Systems Applications (DEXA’02). He is Co-editor in Chief of the *International Journal* “*Transactions on Large-Scale Data- and Knowledge-Centered Systems*” (LNCS, Springer). He was Guest Editor of two special issues of *International Journal of Computer Systems Science and Engineering* on “Mobile Databases” and “Data Management in Grid and P2P Systems”.

Deniz Çokuslu received his BSc (2004) and MSc (2007) from Izmir Institute of Technology, Department of Computer Engineering. He had worked for Yaşar Holding, Astron Project Development Office between 2003 and 2004 as a SAP Software Developer. Since 2004, he is working in Izmir Institute of Technology as a Research Assistant. Now, he is currently continuing his research for his PhD in Ege University, International Computer Institute and Paul Sabatier University, IRIT Laboratory.

Kayhan Erciyes received a BSc in Electrical Engineering and Electronics from the University of Manchester, MSc in Electronic Control Engineering from the University of Salford and a PhD in Computer Engineering from Ege (Aegean) University. He worked as faculty at Oregon State University, University of California Davis and California State University San Marcos, USA. His research interests are in parallel and distributed systems and computer networks. He works on distributed algorithms in mobile ad hoc networks, wireless sensor networks and the Grid. He is a faculty member of Computer Engineering Department and Rector of Izmir University, Izmir, Turkey.

1 Introduction

Research activities are driven by new applications, technology trends and innovative aspects. Since the 1990s, internet became the most important driving force for scientific application development. The increase in demand for resources to be shared by different sites lead researchers to propose Virtual Organisations (VOs), called Grids (Foster and Kesselman, 2004).

A resource in the Grid may correspond to several different concepts. It may be a computational resource such as CPU, memory, storage unit or network; it may be a data resource, which provides metadata and its contents such as database; or it may be a service, which is programmed to accomplish a specific task. The main objective of Grids is to provide a powerful and robust platform, which serves those resources without being affected by the dynamicity of the nodes. The resources offered by Grids have to be accessible to the users easily without any deep technical knowledge. On the other hand, the management services such as job scheduling, load balancing, RD and allocation must be realised in the background without user interaction.

Their large scale, heterogeneity and dynamicity properties characterise Grid systems (Hameurlain et al., 2008). In a large-scale environment, there might be a huge number of data sources (e.g., database, XML files), users and computing resources (CPU, memory, network and I/O bandwidth), which are heterogeneous and autonomous. Moreover, the network presents a low bandwidth on the average, and a strong latency. As far as robustness is concerned, system instability (dynamicity of nodes) means that a node can join, leave or fail at any time.

The extension of distributed and parallel systems to large scale and unstable system dimensions generates new critical problems, which cause real challenges:

- resource discovery
- resource selection and task scheduling
- data management
- autonomic computing and monitoring services
- replication and caching
- security issues.

In this paper, we focus only on the RD issue, which we consider to be the most important one for the success of Grid systems. The important characteristics of the Grid systems such as being large scale, dynamicity and heterogeneity are main reasons for our consideration. Effective usage of the resources in a Grid system relies on the discovery of the right resources for given tasks and the above-mentioned characteristics make the RD a time-consuming process, which can negatively affect the performance of the whole system.

Resource Discovery (RD) in Grids can be defined as searching and locating resource candidates, which are suitable for a job in which processing environments' constraints are clearly specified. On the other hand, the *RD problem* is defined as realising the *RD* in a reasonable time, considering the dynamicity and large scale of the environment. In this perspective, several methods have been proposed to solve the RD problem in Grid systems. They can be classified into three main categories (Hameurlain et al., 2008): methods based on centralised/hierarchical systems, methods based on P2P systems and methods based on agent systems.

In the first decade of Grid systems, web services (Antonioletti et al., 2005) emerged as suitable approaches, which provide an easy-to-use platform-independent tool for Grid services. Open Grid Service Architecture (OGSA) (Foster and Kesselman, 2004) is one of the most important examples of those Grid tools, which emerged by the use of web services. It became essential in most Grid systems. OGSA was mainly aimed to be a standard service interface in Grid technologies in which users can define inter-operable and portable services. The specific communication languages (i.e., Web Services Description Language), which came within such tools, solved many problems in Grids in terms of granting service to the users with less technical knowledge requirements. In most developments, centralised or hierarchical indexing mechanisms were proposed for the RD process

(Antonioletti et al., 2005; Elmroth and Tordsson, 2005; Kaur and Sengupta, 2007; Moltó et al., 2008; Ramos and Magalhaes, 2006; Yu et al., 2003). Hence, these methods were poorly adapted to the large scale and dynamic nature of the Grid in which nodes can leave or join the system at any moment. Most of those methods could not be adapted to today's large-scale environments. Scalability and dynamicity in Grids restrict the usage area of centralised or hierarchical systems. Other approaches to use in Grids were investigated to overcome these problems. Therefore, new areas are researched with this scope in mind, and for some Grid services, especially RD, researchers focused on agent and P2P systems to evaluate their capabilities on a Grid platform. Synergy and convergence between Grid, Agent and P2P systems were clearly pointed out in Foster et al. (2004), Iamnitchi and Talia (2005) and Talia and Trunfio (2003).

In the past few years, software agents, especially MAs (Fuggetta et al., 1998), became very popular software entities, which can ensure important properties to distributed systems such as autonomy, scalability and flexibility. Those properties of agents were studied widely in grid RD domain. In the agent-based studies (Cao et al., 2002; Ding et al., 2005; Kakarontzas and Savvas, 2006; Yan et al., 2007; Yu et al., 2006), the basic idea was to benefit from negotiation and cooperation policies between agents. The organisation type of agents and their internal structures play a central role in efficient resource management. On the other hand, methods based on P2P techniques benefit all the advantages of P2P systems such as scalability. P2P systems were proven to work efficiently under large-scale environments. A good survey on P2P systems can be found in Androutsellis-theotokis and Spinellis (2004). The P2P-based algorithms can be classified into three classes depending on how peers are organised: structured P2P systems, unstructured P2P systems and super-peer systems. In structured P2P systems, peers are organised into VOs. Discovery of resources in such systems is realised by using distributed hash tables (DHTs). In unstructured P2P systems, the peers are not organised to construct any topology and RD is realised by diffusion of messages to the network. In super-peer systems, some nodes are selected to act as directory services (super-peers). Resource information is held by those super-peers to find resources in the Grid system. Different methods were proposed by using all these different P2P systems (Cai et al., 2003; Cheema et al., 2005; Marzolla et al., 2007; Trunfio et al., 2007).

A very comprehensive and detailed survey on the grid RD methods can be found in Ranjan et al. (2008) and Trunfio et al. (2007). These studies examine the existing P2P-based grid RD methods in detail. They both introduce a similar classification as well as a comparison between different classes of methods according to some important criteria. Another survey can be found in Sedaghat et al. (2008). Although, this survey on the grid RD methods is based on Agent systems and P2P techniques. However, it does not examine agent-systems-based methods under a classification. Furthermore, it does not contain a comparative section between the two approaches by using well-defined evaluation criteria.

Considering our current knowledge of the literature, we can state the following remarks:

- few surveys or overviews on the RD methods based on P2P techniques are proposed, such as Ranjan et al. (2008) and Trunfio et al. (2007), which are very comprehensive and detailed on the grid RD methods
- to the best of our knowledge, there are no detailed syntheses on the RD methods based on agent systems in the literature in which classification and comparison between different classes of methods are introduced
- there are no studies that compare the two approaches, P2P-based and agents-based approaches, with respect to the qualitative criteria.

In this paper, we propose a survey of grid RD approaches, which are based on P2P and agent systems. We also provide a qualitative comparison of Grid RD approaches (Section 4) and methods (Sections 2.4 and 3.3). For each class of methods, we describe synthetically the main approach, followed by a deeper analysis and comparison with respect to the most important criteria, namely: complexity, scalability, dynamicity, reliability and support for multi-attribute, dynamic-attribute and range queries. These criteria are explained here:

Complexity is a basic measure, which determines the run-time of the algorithm. In this paper, it is considered in two aspects, message and time complexities. The message complexity deals with the number of transferred messages. Relatively higher message complexities may result in congestion in the network, which may negatively affect the performance of the algorithms. On the other hand, time complexity determines how many steps are required for the termination of the algorithm.

Scalability is a very important measure, because Grids are large-scale environments in their nature. The performance of a system, which is not scalable, degrades very rapidly as the size of the environment grows. This fact may cause the algorithm to perform poorly in such environments.

Dynamicity is another important factor in analysing Grid algorithms since nodes in Grid systems might be highly dynamic in terms of joining and leaving the system, mostly without any notice. The algorithms that tolerate the dynamicity of the environment are more suitable for Grid systems.

Reliability is also an important measure because in some cases, erroneous query results may cause irrecoverable faults. For instance, RD algorithms, which may result in false-positive errors, might not be suitable in Grid systems.

Support for multi-attribute, dynamic-attribute and range queries is a decisive criterion on selecting the methodology in most cases since the running applications may require those types of queries.

The major contributions of this paper are twofold: a detailed survey of main grid RD methods based on the most promising approaches, which are agent and P2P systems and a qualitative comparison between these methods (Sections 2.4 and 3.3) and approaches (Section 4) outlining their advantages and drawbacks.

The rest of the paper is organised as follows. Section 2 presents a synthesis of some RD methods based on P2P techniques by distinguishing the three classical classes of P2P systems. Section 3 provides an overview of RD methods based on agent systems. Section 4 points out the main advantages and drawbacks of each RD approach and summarises a qualitative comparison between them. Section 5 concludes the survey.

2 Grid resource discovery based on P2P techniques

In this section, we first describe some recent RD algorithms based on three different classes of P2P systems: Unstructured P2P Systems, Super-peer Systems and Structured P2P Systems. Next, we give a detailed analysis of these algorithms. Then, we provide a qualitative evaluation for each class of algorithms with respect to the introduced criteria. Finally, we give a comparative study between the different classes of RD techniques.

2.1 Grid resource discovery based on unstructured P2P systems

2.1.1 Overview and analysis

Iamnitchi and Foster (2004) proposed a P2P approach in which nodes construct an unstructured P2P system. In this algorithm, nodes publish their resource information to the network. The requests can be sent to any known node in the overlay. The requests have their Time-To-Live (TTL) parameters to terminate the dissemination of the query after a finite number of hops. If a node that receives the request does not have the required resources, then it forwards the query to one of its neighbours. The request is terminated either by a successive result or by a hop count limitation. Iamnitchi and Foster also mentioned the routing pattern of the queries. They proposed four different routing mechanisms, namely: *random walk*, *learning based*, *best neighbour* and *learning + best neighbour routing*. They evaluated those routing mechanisms with the test results. Although Iamnitchi and Foster made optimisations on the routing of queries, the message complexity of the flooding process is $O(N^2)$ where N is the number of nodes in the network. The time complexity of the algorithm is $O(N^2)$ since the messages are sent one at a time. Although these complexities might be a limiting factor in terms of scalability, the completely distributed nature of this algorithm eliminates bottlenecks and ensures scalability of the algorithm. The algorithm does not contain any single point of failures, but the dissemination of queries is optimised by using TTL parameters, which limits the scope of the queries. This may result in unsuccessful results even if the resource exists somewhere in the Grid (false-positive errors). The algorithm supports different types of queries such as range queries, multi-attribute queries and dynamic queries.

Filali et al. (2008) proposed an unstructured P2P-based RD method for Grids. The resource nodes send their resource information periodically to their neighbours and the neighbours store this information in their cache. The information in the cache will be later used to route the queries towards the destination. If the cache of the requested node does not contain any useful information, the query is flooded to the network. Therefore, worst-case message complexity of the algorithm is $O(N^2)$ where N is the number of nodes in the network. The time complexity is $O(N)$. Because the algorithm is completely distributed, it does not suffer from bottleneck problems. However, even if caching of resources makes the algorithm faster, in a large-scale environment, the worst-case message complexity may limit the scalability of the algorithm. The dynamicity of the network does not negatively affect the algorithm since the failure of a node does not interrupt the distribution of the flooded messages. But, the algorithm uses TTL parameters to limit the scope of flooding messages. This may result in false-positive errors. The algorithm supports both range queries, multi-attribute queries and dynamic-attribute queries since the queries are processed inside the nodes without using any hashing function.

2.1.2 Evaluation

Considering the nature of the unstructured P2P RD techniques, in most cases, because of the common routing mechanisms, the complexity of the algorithms is around $O(N^2)$, which makes the approach unscalable. The message and in some cases time complexities have higher order of growth than the scale of the network.

On the other hand, this approach can easily handle dynamicity of the Grid since both resources and indexing nodes are distributed to the entire network. In any case, even if the network is very dynamic, queries are not lost in the network and propagation of the queries continues until a TTL value is reached.

Nearly all unstructured systems suffer from false-positive errors caused by the usage of TTL limitations. Even if the searched resources exist and are available on the Grid, the system may return unsuccessful results to the queries because the TTL limit is reached. Otherwise, when TTL is set to a higher value, asymptotic increase in the messages negatively affects the bandwidth and runtime of the algorithms.

Nevertheless, the unstructured P2P systems support range, multi-attribute and dynamic-attribute queries easily.

2.2 Grid resource discovery based on super-peer systems

2.2.1 Overview and analysis

Mastroianni et al. (2005) proposed a RD mechanism in which some nodes are selected as super-peers, which act as directory services. Each resource node is connected to one super-peer to share its resource information. The queries are accepted only by the super-peers. When a query is submitted, it is routed to the nearest super-peer first and if the query cannot be satisfied by a super-peer, it is flooded to other super-peers in the network. When a super-peer realises that the query is satisfied by a node within its Physical Organisation (PO), it returns the address of the available resource to the requester. In this study, super-peers are responsible for the indexing mechanism; therefore, flooding of a query only takes place between super-peers. The message complexity of the algorithm is $O(S^2)$ where S is the number of super-peers in the network. The time complexity is $O(S)$. Considering the fact that the number of nodes in each PO is reasonable and the POs are balanced, there are no bottlenecks in terms of indexing the resources. However, as the scale of the requests increases, the super-peers may suffer from bottleneck problems, which may limit the scalability of this system. Moreover, dynamicity of nodes, especially super-peers, may negatively affect a large number of resources. Failure of a super-peer, which manages a large PO, may result in a momentary loss of indexing of a large number of resources, even if those resources are still reachable and available. On the other hand, the algorithm includes a replacement policy for the failures of super-peers, which avoids single-point failures. But, although the algorithm does not contain single-point failures, it suffers from false-positive errors, which are caused by usage of TTL parameters in flooding operations. The use of a simple indexing mechanism brings the ability to handle range and multi-attribute queries. But, because of the periodic updates of the resource information, dynamic-attribute queries are not supported.

Puppin et al. (2005) proposed a grid information service based on super-peer approach. They defined some nodes as super-peers, then created clusters by using the super-peer neighbourhoods. The cluster membership can be redundant, which makes the system more resilient to super-peer failures. The super-peers are responsible for the indexing of their clusters. The resource nodes send their resource information to the super-peers in their clusters. The super-peers are also responsible for keeping information related to the resources and replying to the queries. The queries are flooded between the super-peers until suitable resources are found. In the worst-case scenario, the query is flooded between all super-peer nodes, which have a worst-case message complexity of $O(S^2)$ where S is the number of super-peers in the network. The time complexity of the algorithm is $O(S)$. The algorithm proposes replication of super-peers to solve the scalability problems raised by bottlenecks. The replication of super-peers also solves the dynamicity problem both on the side of resources and super-peers. The usage of TTL parameters in the queries limits the reliability of this algorithm since it may cause false-positive errors. Moreover, periodic updates of resource information may require very frequent updates in highly dynamic networks. The algorithm supports range and multi-attribute queries. But, because of the discrete periodic updates of the resource information, dynamic-attribute queries are not supported.

Marzolla et al. (2005) defined the concept of Workload Management Systems (WMSs), which act as an indexing service for a subset of VOs in the Grid. The WMSs form a tree structure and each WMS is responsible for keeping information about its neighbouring resources. The queries are routed over the WMS tree by using a Breadth First Search (BFS) method. In their design, each attribute value is mapped to a digit in a binary BitMap, therefore, each node has a BitMap array, which stores its resource information. The system supports multi-attribute queries by using a divide-and-conquer approach. The multi-attribute queries are first decomposed to individual sub-queries, and after each query is issued independently, the results are merged, and intersecting resource nodes are extracted. In this study, the super-peers are connected in a tree structure, and search operations for queries are handled by a BFS tree traversal. The worst-case message complexity of the algorithm is $O(S)$ where S is the number of super-peers. The time complexity is $O(H)$ where H is the height of the tree. Since the super-peer tree is not a rooted tree, every super-peer has approximately the same load in the network, which avoids bottlenecks. The low complexity of BFS routing makes this system scalable. The construction and maintenance of the tree structure is left to the responsibility of the underlying overlay system. Nevertheless, a super-peer failure may break the connectivity of the tree, which may prevent a large portion of the Grid from being queried even if resources are available and reachable. Therefore, each super-peer in the tree can be considered as a potential single point of failure. In this study, all resources are considered to be a member of a range; therefore, range queries are naturally supported. The algorithm also supports multiple-attribute queries by using a divide-and-conquer paradigm, but it does not support dynamic-attribute queries since the resources are advertised to super-peers by periodic updates.

2.2.2 Evaluation

Most super-peer-based P2P algorithms use flooding between the super-peers. Decreasing the size of the flooding domain reduces time and message complexities of algorithms. Nearly, all of this type of algorithms have message complexities $O(S^2)$ and time complexities $O(S)$ where S is the number of super-peers in the network. Even these types of algorithms can be considered as more scalable than unstructured systems; super-peers may suffer from being bottlenecks in the system when the number of requests is large.

Moreover, the super-peers are responsible for a set of resources and failure of a super-peer will break the imaginary connection of the resources, which exist and are available. Therefore, dynamicity of a super-peer badly affects the domain of the queries. This fact also negatively affects the reliability of this approach by turning super-peers into single point of failures.

However, since the queries are resolved by super-peers by checking the index tables, this approach supports range queries and multi-attribute queries easily. But, because the resource information is collected by the super-peers at periodic intervals, this method does not support dynamic-attribute queries in its nature.

2.3 Grid resource discovery based on structured P2P systems

2.3.1 Synthesis and analysis

Cai et al. (2003) proposed a grid RD system (MAAN), based on P2P, which supports multi-attribute and range queries. MAAN is an extension of the Chord P2P system. Each MAAN node is an instance of a Chord system. The resource information is mapped to Chord key-space. Each node is responsible for maintaining resource information of its key-space. The single-attribute queries are handled by the search services provided by the Chord system. The multi-attribute queries, on the other hand, are handled in two different ways in MAAN. In the first method, the complex query is divided into sub-queries, and each sub-query is issued within the attribute's proper space. Then, the results of sub-queries are merged to find out if there are any resources, which satisfy all the attributes in the query. In the second approach, the complex query is issued as a single query, which searches a resource that satisfies all the attributes of the query. Since the algorithm uses Chord as its underlying P2P system, its query lookup message and time complexities are $O(\log N)$. The load on each node is fairly distributed. This distribution eliminates bottleneck problems. The approach is scalable since the lookup complexity has a lower order than the number of nodes in the network. Although the system is completely distributed, the queries are relayed to one neighbour at a time; therefore, the case of a node failure may result in a loss of query in the network. But, false-positive errors do not exist since the queries are distributed using a predefined schema. MAAN supports multi-attribute queries by constructing multiple DHTs for each attribute and it also supports range queries by using hashing functions. The dynamic-attribute queries are not supported since DHT is generated once and is updated at discrete intervals.

Andrzejak and Xu (2002) proposed a P2P grid RD mechanism based on CAN P2P system. They extend CAN to handle range queries. In their study, the Grid resources are mapped to a set of attributes. Subsets of nodes in the Grid compose the CAN network, which act as indexing servers in the system (Interval Keeper (IK) Nodes). IK nodes are responsible for a subinterval of attributes. Each node in the Grid submits its resource information to its responsible IK and the queries are processed within the IK nodes. A query is distributed to all the IK nodes in the worst-case scenario by using BFS traversal on a hypercube. Therefore, the time and message complexity of this algorithm is $O(\log_d S)$ where d is the dimension of the hypercube, and S is the number of IK nodes. It ensures the scalability by specifying a threshold value for the responsibility of IK nodes. If the threshold value is exceeded, then a new IK node is added to the same responsibility range. The algorithm also keeps replacement nodes for the IK nodes to ensure fault-tolerance against the dynamicity of IK nodes. For the dynamicity of resources, it uses periodic status updates and timeouts. Since the resource information is distributed between the IK nodes and their replicas, single point of failure situation is minimised. The queries are distributed between all relevant IK nodes, which eliminates false-positive errors. The proposed algorithm is developed for supporting different types of queries. For the range queries, they dedicate subsets of servers to be responsible for specific ranges of an attribute and for the multi-attribute queries each different type of attribute is mapped to a distinct DHT. The resources advertise themselves by reporting their resources to the appropriate DHT server.

Oppenheimer et al. (2004) proposed a structured P2P-based RD mechanism (SWORD), which supports both range queries and multi-attribute queries. In SWORD, there are two types of nodes: reporting nodes and server nodes. Reporting nodes are resources that send their resource information periodically to server nodes and the server nodes collect the resource information and provide querying mechanisms. The server nodes are connected to each other by using Bamboo P2P system and use DHT system to index the resources. When a node makes a resource request, the query is hashed and relayed to the corresponding key-space. Bamboo, therefore SWORD, has a lookup message and time complexity $O(\log N)$ (Rhea et al., 2004) where N is the number of nodes in the overlay. Each node in the overlay has the same probability of being assigned to a query, which ensures that there are no bottlenecks in the system. Moreover, the order of growth in the message and time complexities is smaller than the number of nodes in the system, which proves the scalability of the algorithm. The queries are relayed to one neighbour at a time, therefore, failure of an overlay node may result in loss of queries in the network, but since the queries are relayed in the overlay without any TTL restrictions, the false-positive errors do not exist. SWORD is designed to handle both multi-attribute and range queries. But, dynamic-attribute queries are not supported since DHTs are updated at discrete intervals.

In XenoSearch, Spence and Harris (2003) proposed a P2P-based resource allocation method for distributed environments such as Grids. The proposed algorithm is an extension of the Pastry P2P system (Rowstron and Druschel, 2001). In this system, XenoServers are organised in a tree structure. The resources are at the leaves of the tree and the inner nodes aggregate resource information of its child nodes by classifying them into different ranges for attributes. The inner XenoServer nodes periodically collect aggregate information of resources. When a query is posted, it is traversed in the tree and if suitable resources are found, the resulting XenoServers are returned to the requester. Then, the requester checks the resulting servers to see if the resource information is up-to-date, and seizes the resources it needs. The time and message complexity of the XenoSearch algorithm is $O(\log N)$ where N is the number of nodes in the XenoServer overlay (Spence and Harris, 2003). Since query-processing tasks are distributed between the XenoServer nodes, bottlenecks do not exist. Moreover, query lookup complexity has a lower order than the number of nodes in the network, which ensures scalability of the system. The resource information is updated periodically in the XenoServers by aggregating the resource information in the tree structure. Failure of a XenoServer node may result in loss of both queries and aggregated updates. Therefore, the tree nodes in the algorithm might be considered as single point of failures. On the other hand, the use of DHT eliminates any false-positive errors. The algorithm supports multi-attribute queries by independently querying each attribute, and intersecting the results. It also supports range queries by using a multi-dimensional DHT. For dynamic-attribute queries, after the query result is held, the algorithm contacts candidate resources to verify the current status.

Talia et al. (2007) proposed a DHT-based RD mechanism for large-scale Grids. In their study, they pointed out the advantages of using P2P systems for the RD in Grids. But, they also remarked the inefficient use of unstructured P2P. The proposed architecture is based on Chord Structured DHT system. They extend Chord so that it supports multi-attribute queries by utilising multiple DHTs. For the dynamic entity queries such as CPU load, they use DHT broadcasting since for dynamic resources, the cost of keeping structured DHTs is very high. They use the Chord finger tables to distribute dynamic-attribute queries in parallel. To prevent redundant messages, they incrementally increase the finger entries starting from 1. Then, the number of fingers is incremented up to an optimal value for use in the future. For the dynamic-attribute queries, the algorithm uses DHT broadcast mechanism on Chord, which has a time complexity of $O(\log N)$ and message complexity of $O(N)$. For other types of queries, the time and message complexities are $O(\log N)$ where N is the number of nodes in the overlay. Since each node in the overlay has the same probability to receive a query, bottlenecks do not exist in the system. Moreover, order of time and message complexities in all cases are less than or equal to the number of nodes in the network, which ensures the scalability of the algorithm. A node failure in the overlay may cause loss of requests since some types of queries are relayed to only one neighbour at a time. Since each node in the overlay is responsible by itself, single-point-of-failures do not exist in terms of query results. But, during a query relay, each node may become a single point of failure. On the other hand, the queries are routed using DHTs and DHT broadcast mechanisms; therefore, false-positive errors do not exist. The usage of DHT broadcast for dynamic-attribute queries ensures that the results to the queries are always up to date; hence, it supports dynamic-attribute queries. The algorithm also supports multi-attribute and range queries by using multiple DHTs.

2.3.2 *Evaluation*

Since these algorithms use topological structures, time and message complexities of the algorithms are around $O(\log N)$. In many algorithms, all resource nodes get involved in the query processing, which means that, theoretically, all nodes will have the same load. This eliminates the bottlenecks in the system and ensures the scalability of the structured P2P approach.

On the other hand, in most algorithms, the queries are distributed to the network by following a defined path in the topological structure. Therefore, failure of a node, which will forward the query, may result in loss of queries in the network. This brings the single point of failure problem in a dynamic Grid environment. But, the use of structured query routing mechanisms eliminates false-positive errors since the query is relayed to the end of its search domain.

Even if the nature of structured P2P-based RD algorithms do not support range, multi-attribute and dynamic-attribute queries, nearly all algorithms, which are developed in this scope, find reasonable solutions to support all different types of queries.

2.4 *Comparison of different P2P techniques used in grid resource discovery*

We described and analysed several grid RD algorithms, which are developed by using different P2P techniques. Although each algorithm has its own advantages and disadvantages, commonalities can be clearly distinguished when they are examined in their own classes. Regarding those commonalities, a comparison can be easily performed between three different classes of P2P techniques. A summary of comparison can be seen in Table 1.

Table 1 Summary of comparison between P2P-based resource discovery methods

	Unstructured P2P	*Super-peer P2P*	*Structured P2P*
Scalability	Not scalable due to time and message complexities	Not scalable due to bottlenecks	Scalable since complexities are low and load is distributed
Dynamicity	Tolerant to node dynamicity since queries are resolved within the nodes	Performs poorly when the Grid is dynamic	Performs poorly when the Grid is dynamic
Reliability	Not reliable because of the false-positive errors	Not reliable because of the false-positive errors and single point of failures	Reliable since no single point of failures and no false-positive errors exist

Table 1 Summary of comparison between P2P-based resource discovery methods (continued)

	Unstructured P2P	*Super-peer P2P*	*Structured P2P*
Range queries	Supported since queries are resolved within the nodes without any hashing	Supported since queries are resolved within the super-peers without any hashing	Supported using complicated techniques
Multi-attribute queries	Supported since queries are resolved within the nodes without any hashing	Supported since queries are resolved within the super-peers without any hashing	Supported using complicated techniques
Dynamic-attribute queries	Supported since queries return always up-to-date results	Not supported since resource information in the super-peers is updated in discrete intervals	Supported using complicated techniques

Regarding the analysis, we can say that the unstructured P2P-based grid RD systems are suitable for small scale, highly dynamic Grid environments in which different types of queries are required. If the scale is large, and false-positive errors are fatal, then other methods should be examined. Super-peer-based grid RD mechanisms are suitable for middle-scale Grid networks in which reliability of super-peers is strictly provided. They are not suitable for dynamic-attribute queries and if the false-positive errors cause serious problems. Structured P2P-based methods, on the other hand, are suitable for large-scale Grid systems in which reliability is important and dynamicity is low.

3 Grid resource discovery based on agent technologies

In this section, we propose a synthetic review of the state-of-the-art and analysis of some recent resource discovery algorithms, which are based on agent systems. The algorithms, which use agents, profit from using them as monitoring services. On the other hand, studies that are based on MAs profit mainly from their autonomy property, which allows the query to determine migration site by itself. Although the type of the utilised agent differs from each other, we decided to classify them according to their underlying network topologies since we believe that underlying topology has more impact on the evaluation of these algorithms. In this perspective, we defined two classes: algorithms that do not rely on a structured network topology and those that generate a network topology to accomplish RD tasks. We also provide an evaluation for each class of grid RD algorithms. Then, we show the comparison between RD methods based on different classes of agent systems.

3.1 Agent-based grid resource discovery on unstructured network topology

3.1.1 Overview and analysis

Ding et al. (2005) proposed a heuristic-agent-based RD algorithm. The agents, which are mapped into Grid nodes, cooperate to find available resources. The service information of the Grid resources are advertised within the agent graph. Each agent maintains an Agent Information Table (AIT), which records resource information of the agent itself and its neighbours. When a new resource is submitted, its agent advertises the resource information to adjacent agents. RD involves querying the contents of the AITs. When a task is received by an agent, the local AIT is checked. If the resource is not located in the local AIT, then the agent checks the AITs of the adjacent nodes. If the resource is found in one of the adjacent AITs, then the query is forwarded to that node. If the agent cannot find the resource in its local and adjacent AITs, then the query is forwarded to all its neighbours by using flooding. Since in the worst-case scenario, the query passes through all the edges between nodes, the worst-case message complexity of this algorithm is $O(E)$ and the time complexity is $O(D)$ where E is the number of edges and D is the diameter of the agent graph. Owing to its unstructured nature, the algorithm does not contain any bottlenecks and single point of failures. But, flooding in a dense network could be a limiting factor in terms of scalability. Moreover, since no rule is defined to limit the scope of the flooding, the result might take very long time to converge in some cases. On the other hand, since the queries are flooded to the network in parallel, dynamicity of the nodes does not negatively affect the querying process. Moreover, since flooded requests do not have a TTL limit, false-positive errors do not exist. The algorithm supports all multi-attribute, dynamic-attribute and range queries since the queries are resolved within the nodes without any hashing function.

Jun et al. (2000) introduced an agent-based RD model. The agents running at different nodes learn about the existence of each other using a mechanism called distributed awareness. Each agent maintains information tables about the other agents it has communicated with over a period of time and exchanges periodically this information among them. Whenever an agent needs detailed information, the information gathered by the distributed awareness mechanism is used. The agents are capable of reporting the status of their residing resources by piggy-backed messages. After a period of time, all nodes become aware of each other by merging the information tables exchanged. In their model, agents are created at remote locations to gather information, which is needed for resource management. The time and message complexities of distributed awareness algorithm is $O(E)$ where E is the number of edges in the system. Even if the algorithm does not contain any single point of failures and bottlenecks, in a highly dynamic and large-scale Grid system, the high number of update messages could limit the scalability of the algorithm. Since whole resource information exists in each node without any hashing, the system supports range and multi-attribute queries. But since dynamic-attributes are not updated, this system does not support dynamic-attribute queries.

In Yu et al. (2006), an MA-based grid resource management system is presented. In their model, an information server creates MA according to the required resources' properties. It also designates resource-query binding conditions and route rules. MA migrates according to their route rules and communicates with system agents residing on the resource nodes. The grid resources information management is held by the cooperation of MA and LDAP protocol to fetch both local and global resource information. Since the route selection in MA affects the system performance directly, Yu et al. preferred Shortest Distance Routing and Transferring algorithm in the MA. The central resource information server is used only for creating MA. The MAs migrate according to the specified route rules. Therefore, the message and time complexities of this approach is $O(N)$ where N is the number of nodes in the system. In this algorithm, the centralised information server limits the scalability of the algorithm. On the one hand, query migration is held by resource nodes. So, from the point of view of relaying nodes, bottleneck problem does not exist. On the other hand, the MA migrates according to a route strategy on a specified path. For that reason, dynamicity of resource nodes may perturb the migration of the queries. The algorithm supports range, multi-attribute and dynamic-attribute queries since the queries are resolved within the resource nodes without any hashing operation.

Tang and Huang (2006) proposed a new grid resource management algorithm based on MAs. They also described an architectural model for acquisition of Grid information. The proposed system consists of a *MessagingServer* and *ResourceNodes*. The *MessagingServer* is responsible for the generation of MA. It generates MA according to the defined constraints. It also determines the routing rules of the MA. The MA travels the network with respect to the route rules and collects information about the resources. When a user requests for a resource, it sends a query to the *MessagingServer*. Then, the messaging server generates a MA, and sends user back the results of the query. The migration strategy of MA depends on the route rules. Tang and Huang examined the migration rules in two methods: static routing and dynamic routing. In the static routing, the path of the MA is determined by the information server beforehand whereas in the dynamic routing, generally information server assigns an initial route table at first, but the MA modifies the route table during the migration according to the environment alternations. Since the queries are routed between resource nodes according to route rules, the message and time complexities of this algorithm are $O(N)$ where N is the number of resource nodes in Grid environment. The queries are processed within resource nodes in this system. This eliminates bottleneck problems in the relaying nodes, but since the *MessagingServer* is centralised, it might become a bottleneck and a single point of failure. The algorithm uses resource nodes to propagate the queries sequentially until the destination is reached. Failure of a node, which is forwarding an MA, may result in loss of queries in the network. The *MessagingServer* does not hold global information; therefore, it could be replicated to avoid single point of failures. And since the query agents are migrated according to a planned strategy, false-positive errors do not exist. The queries are processed within nodes without any hashing; so, the algorithm supports dynamic-attribute, range and multi-attribute queries.

3.1.2 Evaluation

In agent-based grid RD approaches in which the underlying network topology is unstructured, the system does not suffer from bottleneck problem. The main factor that affects the scalability of the system is diffusion technique of the requests. When agents are used, the diffusion is handled by using flooding approach, which is unscalable because of its message complexity. On the other hand, if MAs are used, because of their autonomy and self-decision properties, more clever routing techniques are applied to increase the scalability.

On the other hand, when the Grid is dynamic, since in agent-based approaches flooding is used to distribute the queries, dynamicity of the nodes does not perturb the dissemination of queries. But when the MAs are used, the queries are routed on a single path, and the failure on any node on this path may cause loss of queries in the network.

The algorithms do not have single point of failures in general since central managers do not exist. Moreover, the distribution of queries is not limited by a TTL value, which eliminates false-positive errors. But, we believe that in large-scale environments in which an unstructured network topology exists, TTL values are essential to avoid extremely long query response durations.

Nevertheless, nearly all analysed algorithms, which belong to this classification, support range, multi-attribute and dynamic-attribute queries easily since they process queries within the nodes without any discrete mapping function such as hashing.

3.2 Agent-based grid resource discovery on structured network topology

3.2.1 Overview and analysis

Yan et al. (2007) proposed a system in which the resources are divided into some VOs. Each VO constitutes an overall index server and several nodes. In this model, each node is both the client that queries the resource information and the server of resource information management. The agents monitor the changes in resources to the local servers;

the dynamic-attributes are dynamically updated on the LDAP server of VO by using Update Agent. At the same time, each VO renews the resource information on the index server of central domain through LDAP's copy mechanism periodically. The RD process is held in the same order: first, the local servers are queried, then the VO local LDAP servers, and finally central domain global LDAP server is queried. The RD is held by using MAs. The proactive migration of MAs is a simple tree traversal mechanism. If a node cannot response to a query, it sends the query agent to an upper level (VO LDAP Server), if the query cannot be satisfied within the VO, then it is forwarded to the global LDAP server. From this point, the query agent migrates through the node, which satisfies the query. In the worst case, central VO LDAP Server sends the MA to all VOs in the network. Therefore, the message complexity of this algorithm is $O(D)$ where D is the number of VOs. The time complexity is $\Theta(c)$ where c is a constant since distribution of MA to the VOs is issued in parallel. The use of a global LDAP server and some VO LDAP servers may result in bottlenecks in the system, which could negatively affect the scalability of the algorithm. The VOs and Global LDAP server could also become single point of failures, therefore, dynamicity of central servers may cause false-positive errors. The algorithm supports multi-attribute and range queries. For the dynamic-attribute queries, the system uses update-agents, which update nodes' status dynamically.

Kakarontzas and Savvas (2006) presented an agent-based approach to grid RD and selection process. The approach is based on client agents, which act on behalf of Grid users. It searches for resources in a network of Resource Representatives (RRs), which are registries of resource characteristics. The resources are first registered to the system by using an agent named Local System RMS Agent. Another agent, named RR, maintains a repository of registrations. The RRs form a network; they match requests and resources by searching suitable resources within their name spaces. An RR may represent a large number of nodes. To effectively search a resource among the RR nodes, non-overlapping clusters are built. If a query cannot be satisfied within an RR, it is forwarded to the neighbours until a resource is found or the request is timed out. The neighbourhood of RRs can be defined by any distance measure such as hop-counts. The clients generate resource constraints and submit the query by using the Client Agent. The client agent sends the query to the RR and collects the potential resources and after collecting resource candidates, it communicates directly to the resources. In that stage, resources send their current up-to-date status, then the client selects the most suitable resource, and sends a cancel message to the rest of the candidates. The RD process is realised by disseminating the query into a graph of RRs. Flooding of the query is limited to a hop count, which is determined heuristically. Since flooding is required, the message complexity of the algorithm is $O(E)$ and the time complexity is $O(D)$ where E is the number of edges between RRs and D is the diameter of the graph. Use of a large number of resource management agents and RRs in this algorithm eliminates bottleneck problem and constructs virtual clusters, which makes the algorithm more scalable. The dynamicity of resource nodes is updated to their Local RMS agent; therefore, dynamicity of resource nodes does not negatively affect this algorithm. But, failure of a local RMS node or an RR may cause a large number of nodes to become invisible even if they exist and are available. In consequence, the RRs are single point of failures in this algorithm. Moreover, queries are flooded between RRs by using a TTL parameter, which may result in false-positive errors. Since RRs process the queries centrally without any hashing, multi-attribute and range queries are supported. Moreover, since the up-to-date nodes' statuses are obtained by communicating with resource nodes, dynamic-attribute queries are also supported.

In Manvi et al. (2005), an Agent-based Resource Allocation Model (ARAM) is presented. Manvi, Brije and Prasad presented three types of agents in their study. Resource Brokering Agents (RBAs) act as resource schedulers and try to match the requests with the resources. An RBA maintains a resource database of each Grid and the information is updated by interacting with the grid information servers. Job Agents (JAs), which are mobile, search for the available resources. The migration of unallocated JAs is directed by RBAs. Resource Monitoring Agents (RMAs) inform the node's current state to the local cluster server. In ARAM, the master server divides the tasks submitted by the user into subtasks and distributes them to the local cluster servers. Local cluster servers try to find available resources and assign each subtask to a resource if it exists. The migration of JAs can be handled in three different topologies. In single-step topology, an unallocated JA migrates consecutively to the next RBA node in a circular fashion. In random transfer topology, an unallocated JA migrates to a random node to find the suitable resource and in tree shape topology, RBAs are arranged as a tree-like structure. The unallocated JAs always migrate through the parent nodes. The message and time complexities of the RD process are $O(C)$ where C is the number of clusters in the system. The use of clusters increases the scalability of system by distributing the load on the local cluster servers. But, dynamicity of the local cluster servers or master cluster servers may badly affect the system since failure of such nodes will cause a large part of the resources to become invisible even if they exist and are available. The algorithm supports range and multi-attribute queries. For the dynamic-attribute queries, Manvi, Brije and Prasad propose two update strategies: continuous and periodic. The algorithm can support dynamic-attribute queries only if continuous updates are used.

Puh et al. (2007) proposed a multi-agent RD algorithm for Grids. The proposed system is composed of five agents: User Agent, Directory Facilitator (DF) Agent, Service Agent, Multi-Operation Agent and Monitoring Agent. Each

Grid service is registered to a service agent and the Grid services are registered to the DF at the service boot time. The users are represented by user agents, which are mobile. When a user needs a resource, the user agent checks the DF, and gets a list of resources. Then, it chooses some resources from the resulting list and completes the RD. The queries are processed in the central DF node and the results are verified by communicating the candidate nodes. Since the query is distributed to all nodes in the system in parallel, the message complexity of the algorithm is $O(N)$ and the time complexity of the algorithm is $O(c)$ where c is a constant. Despite the low complexities, this algorithm is not scalable because of the bottleneck problem, which is raised in the central DF agent in large-scale Grids. The existence of a RMA allows the resources to be dynamic but the dynamicity of a central DF agent may negatively affect the algorithm since it is a single point of failure. Since the queries are resolved in the central DF agent without any hashing, the algorithm easily supports range and multi-attribute queries. Moreover, the monitoring agent reports all current resource information to the DF agent in real time, which makes this algorithm support dynamic-attributes too.

3.2.2 Evaluation

In structured networks, the bottleneck problem is eliminated since the load on nodes is distributed. Moreover, because of the structured nature, migration of MAs or distribution of queries is held with more efficient routing techniques, which increase the scalability of the system.

But, this advantage brings some problems such as single point of failures. When the Grid is dynamic, since some nodes act as relay nodes or manager of a subset of resource nodes, the dynamicity of the network may perturb the dissemination of queries. The failure of a node in the structure will cause loss of queries in the network. Moreover, failure of a node, which manages a large subset of resources, will cause a large number of resources to become invisible to the queries even if they exist.

On the other hand, nearly all analysed algorithms, which belong to this class, support range and multi-attribute queries easily since they are processed within the nodes without any discrete mapping function such as hashing. The dynamic-attribute queries are also supported by the use of agent technology, which allows the resources to send updates about their dynamic-attributes continuously instead of at periodical intervals.

3.3 Comparison between different agent-based resource discovery techniques

Several agent-based grid RD algorithms are described and analysed. In this section, a comparison of each class of those algorithms is explained with respect to the defined criteria. A summary of the comparison can be seen in Table 2.

Table 2 Summary of comparison between agent-based resource discovery methods

	Unstructured network topology	*Structured network topology*
Scalability	Not scalable due to time and message complexities	Scalable because of the hierarchical distribution of load
Dynamicity	Tolerant to node dynamicity since queries are distributed in parallel, but not tolerant in some approaches in which query migrates on a single path	Not tolerant since dynamicity of nodes in the structure may result in disconnectivity of a large portion of the resources
Reliability	Not reliable because of either false-positive errors or single point of failures	Not reliable because of the single point of failures
Range queries	Supported since queries are resolved within the resource nodes without any hashing	Supported since queries are resolved within the nodes without any hashing
Multi-attribute queries	Supported since queries are resolved within the resource nodes without any hashing	Supported since queries are resolved within the nodes without any hashing
Dynamic-attribute queries	Supported since queries are resolved within the resource nodes without any hashing	Supported since agents update the resource information dynamically

4 Global comparison of grid resource discovery approaches

We analysed and evaluated some recent studies related to different methodologies used in grid RD in the previous sections. We focused on Agent systems and P2P-based grid RD techniques. In Table 3, a summary of comparison of the methodologies can be found, which is explained in detail here.

Both Agent-based and P2P-based techniques are being broadly studied in today's grid RD systems. Agent-based systems are attractive, mainly because of their autonomy property. They have capabilities to determine new migration sites according to their migration policies. This property might easily be used to accomplish efficient route selection for queries, which converges to the result in each step. On the other hand, their non-deterministic nature results in false-positive errors in many recent studies in which inefficient flooding techniques are avoided. Most of the agent-based RD techniques process the queries inside the resource nodes while the query is traversing the network. This brings the advantage of having up-to-date resource information all the time, which makes this class of algorithms support dynamic-attribute queries. Regarding these attributes, agent-based RD methods are suitable for dynamic middle-scale Grid environments in which some false-positive errors are acceptable. On the other side, most recent P2P techniques use structured DHT systems, which increase the performance of the queries drastically. Moreover, usage of the DHT mappings brings the scalability and reliability of the P2P systems since all nodes in the system involve the RD process.

Table 3 Summary of comparison between P2P and agent-based resource discovery methods

	RD based on agent systems	*RD based on P2P systems*
Scalability	Suitable for small-medium scale Grid environments	Suitable for large-scale Grid environments
Dynamicity	Tolerant because of the autonomy property of agents	Perform poorly when the environment is dynamic
Reliability	Not reliable because of either false-positive errors or single point of failures	Reliable since false-positive errors and single point of failures do not exist
Range queries	Supported since queries are resolved within the resource nodes without any hashing	Supported using complicated techniques
Multi-attribute queries	Supported since queries are resolved within the resource nodes without any hashing	Supported using complicated techniques
Dynamic-attribute queries	Supported since queries are resolved within the resource nodes without any hashing	Supported using complicated techniques

But, on the other hand, DHTs are not miraculous and have some disadvantages for the RD domain. The usage of DHTs limits the RD algorithm in terms of support for dynamic-attribute queries. Since dynamic-attributes of resources are changing in time, keeping these attributes in DHTs is not feasible. To solve this problem, many algorithms use the topological structure of the overlay to efficiently distribute the query directly between the resource nodes. Inheriting all properties of overlay systems, P2P-based grid RD methods are suitable for large-scale dynamic environments in which reliability of queries is important.

5 Conclusion

Resource management centred on RD is the key to the success of Grid systems. Initially, the proposed RD methods were based on centralised/hierarchical topology. These methods have the advantages of being OGSA compliant. However, with respect to the main characteristics of Grid systems, being large scale and unstable (dynamicity of nodes), it is clear that the proposed methods become unattractive in such an environment. The RD methods should be scalable and robust. In this perspective, the synergy and convergence between Grid, P2P and Agent systems have motivated many authors to design and develop different classes of RD methods: methods based on P2P techniques and methods based on agent systems. Both these paradigms can help for scaling and reliability. In this paper, we proposed a survey of main RD methods. For each class of methods, we described synthetically the main methods, with a detailed analysis. With respect to introduced criteria (e.g., scalability, dynamicity, etc.), a qualitative evaluation of described methods was provided. This enables to appreciate the behaviour of these methods in a highly dynamic large-scale environment. We also suggested a comparative study between both RD methods (P2P techniques and agent systems), which allowed pointing out their advantages and drawbacks.

References

Androutsellis-theotokis, S. and Spinellis, D. (2004) 'A survey of peer-to-peer content distribution technologies', *ACM Computing Surveys*, Vol. 36, pp.335–371.

Andrzejak, A. and Xu, Z. (2002) 'Scalable, efficient range queries for grid information services', *2nd International Conference on Peer-to-Peer Computing, P2P 2002*, Linköping, Sweden, pp.33–40.

Antonioletti, M., Atkinson, M., Baxter, R., Borley, A., Hong, N.P.C., Collins, B., Hardman, N., Hume, A.C., Knox, A., Jackson, M., Krause, A., Laws, S., Magowan, J., Paton, N.W., Pearson, D., Sugden, T., Watson, P. and Westhead, M. (2005) 'The design and implementation of grid database services in ogsa-dai: research articles', *Concurr. Comput.: Pract. Exper.*, Vol. 17, Nos. 2–4, pp.357–376.

Cai, M., Frank, M., Chen, J. and Szekely, P. (2003) 'Maan: a multi-attribute addressable network for grid information services', *4th Int. Workshop on Grid Computing, GRID 2003*, pp.184–191.

Cao, J., Spooner, D., Turner, J.D., Jarvis, S., Kerbyson, D.J., Saini, S. and Nudd, G. (2002) 'Agent-based resource management for grid computing', *CCGRID'02: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, IEEE Computer Society, pp.350.

Cheema, A.S., Muhammad, M. and Gupta, I. (2005) 'Peer-to-peer discovery of computational resources for grid applications', *In GRID05: Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing, IEEE Computer Society*, pp.179–185.

Ding, S., Yuan, J., Ju, J. and Hu, L. (2005) 'A heuristic algorithm for agent-based grid resource discovery', *Intl. Conf. on e-Technology, e-Commerce and e-Service*, Hong Kong, pp.222–225.

Elmroth, E. and Tordsson, J. (2005) 'An interoperable, standards-based grid resource broker and job submission service', *First International Conference on e-Science and Grid Computing*, pp.212–220.

Filali, I., Huet, F. and Vergoni, C. (2008) 'A simple cache based mechanism for peer to peer resource discovery in grid environments', *CCGRID08, Eighth IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society*, pp.602–608.

Foster, I., Jennings, N.R. and Kesselman, C. (2004) 'Brain meets brawn: Why grid and agents need each other', *AAMAS04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, IEEE Computer Society, New York, USA, pp.8–15.

Foster, I. and Kesselman, C. (2004) *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, CA, USA.

Fuggetta, A., Picco, G.P. and Vigna, G. (1998) 'Understanding code mobility', *IEEE Transactions on Software Engineering*, Vol. 24, No. 5, pp.342–361.

Hameurlain, A., Morvan, F. and Samad, M.E. (2008) 'Large scale data management in grid systems: a survey', *IEEE International Conference on Information and Communication Technologies: from Theory to Applications (ICTTA)*, pp.1–6.

Iamnitchi, A. and Foster, I. (2004) *A Peer-to-Peer Approach to Resource Location in Grid Environments*, Kluwer, Norwell, MA, USA.

Iamnitchi, A. and Talia, D. (2005) 'P2P computing and interaction with grids', *Future Generation Computer Systems*, Vol. 21, No. 3, pp.331–332.

Jun, K., Boloni, L., Palacz, K. and Marinescu, D.C. (2000), 'Agent-based resource discovery', *Proc. 9th IEEE Heterogeneous Computing Workshop*, Cancun, Mexico, pp.43–52.

Kakarontzas, G. and Savvas, I.K. (2006) 'Agent-based resource discovery and selection for dynamic grids', *Proc. of the 15th IEEE Intl. Workshops on Enabling Technologies*, pp.195–200.

Kaur, D. and Sengupta, J. (2007) 'Resource discovery in web-services based grids', *Proceedings of World Academy of Science, Engineering and Technology*, pp.284–288.

Manvi, S.S., Birje, M.N. and Prasad, B. (2005) 'An agent-based resource allocation model for grid computing', *IEEE SCC*, pp.311–314.

Marzolla, M., Mordacchini, M. and Orlando, S. (2005) 'Resource discovery in a dynamic grid environment', *DEXA Workshop 2005*, IEEE Press, pp.356–360.

Marzolla, M., Mordacchini, M. and Orlando, S. (2007) 'Peer-to-peer systems for discovering resources in a dynamic grid', *Parallel Computing*, Vol. 33, Nos. 4–5, pp.339–358.

Mastroianni, C., Talia, D. and Verta, O. (2005) 'A super-peer model for resource discovery services in large-scale grids', *Future Generation Computer Systems*, pp.1235–1248.

Moltó, G., Hernández, V. and Alonso, J.M. (2008) 'A service-oriented wsrf-based architecture for metascheduling on computational grids', *Future Generation Computing Systems*, Vol. 24, No. 4, pp.317–328.

Oppenheimer, D., Albrecht, J., Patterson, D. and Vahdat, A. (2004) *Scalable Widearea Resource Discovery*, Technical report csd04 -1334, University of California Berkeley.

Puh, M., Jezic, G. and Kusek, M. (2007) 'Multi-agent system for resource discovery in grid network', *16th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, Paris, France, pp.320–324.

Puppin, D., Moncelli, S., Baraglia, R., Tonelotto, N. and Silvestri, F. (2005) 'A grid information service based on peer-to-peer', *EuroPar, Springer LNCS*, pp.454–464.

Ramos, T.G. and Magalhaes, A.C. (2006) 'An extensible resource discovery mechanism for grid computing environments', *CCGRID06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society*, pp.115–122.

Ranjan, R., Harwood, A. and Buyya, R. (2008) 'Peer-to-peer-based resource discovery in global grids: a tutorial', *Communications Surveys and Tutorials, IEEE*, Vol. 10, No. 2, pp.6–33.

Rhea, S., Geels, D., Roscoe, T. and Kubiatowicz, J. (2004), 'Handling churn in a dht', *ATEC04: Proceedings of the annual conference on USENIX Annual Technical Conference*, Boston, MA, USA, pp.127–140.

Rowstron, A. and Druschel, P. (2001) 'Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems', *Lecture Notes in Computer Science*, pp.329–350.

Sedaghat, M., Othman, M. and Sulaiman, M. (2008) 'Agents role in grid environments during resource discovery: a review', *Information Technology, ITSim08*, Vol. 3, pp.1–9.

Spence, D. and Harris, T. (2003) 'Xenosearch: Distributed resource discovery in the xenoserver open platform', *Proceedings of HPDC*, IEEE Computer Society, p.216.

Talia, D. and Trunfio, P. (2003) 'Toward a synergy between p2p and grids', *IEEE Internet Computing*, Vol. 7, No. 4, pp.95–96.

Talia, D., Trunfio, P. and Zeng, J. (2007) 'Peer-to-peer models for resource discovery in large-scale grids: a scalable architecture', *High Performance Computing for Computational Science – VECPAR 2006*, pp.66–78.

Tang, X. and Huang, L. (2006) 'Grid resource management based on mobile agent', *WISE Workshops*, pp.230–238.

Trunfio, P., Talia, D., Papadakis, H., Fragopoulou, P., Mordacchini, M., Pennanen, M., Popov, K., Vlassov, V. and Haridi, S. (2007) 'Peer-to-peer resource discovery in grids: Models and systems', *Future Gener. Comput. Syst.*, Vol. 23, No. 7, pp.864–878.

Yan, M., Yu-hui, Q., Gang, W. and Ju-Hua, Z. (2007) 'Study of grid resource discovery based on mobile agent', *Proc. of the 3rd Intl. Conf. on Semantics, Knowledge and Grid*, pp.570–571.

Yu, J., Venugopal, S. and Buyya, R. (2003) *Grid Market Directory: A Web Services Based Grid Service Publication Directory*, Technical Report, Grid Computing and Distributed Systems (GRIDS) Lab, Dept. of Computer Science and Software Engineering, The University of Melbourne, Melbourne, Australia.

Yu, J., Zhao, C. and Pan, Y. (2006) 'Grid resource management based on mobile agent', *Proc. of the Intl. Conf. on Computational Intelligence for Modeling Control and Automation and Intl. Conf. on Intelligent Agents Web Technologies and Intl. Commerce*, Sydney, Australia, pp.255–256.