Available online at www.sciencedirect.com
SCIENCE DIRECT®
Future Generation Computer Systems 21 (2005) 1235–1248
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

A super-peer model for resource discovery services in large-scale Grids☆

Carlo Mastroianni [a], Domenico Talia [b,*], Oreste Verta [b]

[a] *ICAR-CNR 87036 Rende (CS), Italy*
[b] *DEIS University of Calabria, 87036 Rende (CS), Italy*

Received 23 March 2005; received in revised form 3 June 2005; accepted 6 June 2005
Available online 19 July 2005

Abstract

As deployed Grids increase from tens to thousands of nodes, peer-to-peer (P2P) techniques and protocols can be used to implement scalable services and applications. The super-peer model is a novel approach that helps the convergence of P2P models and Grid environments and can be used to deploy a P2P information service in Grids. A super-peer serves a single physical organization in a Grid, and manages metadata associated to the resources provided by the nodes of that organization. Super-peers connect to each other to form a peer network at a higher level. This paper examines how the super-peer model can handle membership management and resource discovery services in a multi-organizational Grid. A simulation analysis evaluates the performance of a resource discovery protocol; simulation results can be used to tune protocol parameters in order to increase search efficiency.

1. Introduction

Grid computing and peer-to-peer (P2P) computing models share several features and have more in common than we generally recognize. The integration of the two computing models could bring benefits in both fields and could result in future integrations. In particular, the use of P2P protocols is expected to improve the efficiency and scalability of information services in large-scale Grid systems [8,19].

As Grids used for complex applications increase from tens to thousands of nodes, their functionalities should be decentralized to avoid bottlenecks. The P2P model could favour Grid scalability: designers can use P2P style and techniques to implement decentralized Grid systems. The adoption of the service oriented model in novel Grid systems (for example, the Open Grid Services Architecture (OGSA) [2], or the Web Services Resource Framework (WSRF) [20]) will support the convergence between the two models, since Web Services can be used to implement P2P interactions between hosts belonging to different domains.

☆ Extended version of the Best Paper Award winning paper in the European Grid Conference 2005.

* Corresponding author. Tel.: +39 0984 494726; fax: +39 0984 494713.
E-mail addresses: mastroianni@icar.cnr.it (C. Mastroianni), talia@deis.unical.it (D. Talia), verta@deis.unical.it (O. Verta).

0167-739X/$ – see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2005.06.001

In particular, an ongoing effort aims at studying how it is possible to drive the integration trend to efficiently handle two key services in Grid information systems: *membership management* (or simply *membership*) and *resource discovery* services. The objective of a membership management service is two-fold [8]: adding a new node to the network, and assigning this node a set of neighbour nodes. The resource discovery service is invoked by a node when it needs to discover and use hardware or software resources matching given criteria and characteristics.

In currently deployed Grid systems, resources are often owned by research centres, public institutions, or large enterprises: in such organizations hosts and resources are generally stable. Hence, membership management and resource discovery services are efficiently handled through centralized or hierarchical approaches, as in the OGSA and WSRF frameworks. As opposed to Grids, in P2P systems nodes and resources provided to the community are very dynamic: peers can be frequently switched off or disconnected. In such an environment, a distributed approach is more effective and fault-tolerant than a centralized or hierarchical one.

Recently, super-peer networks have been proposed [21] to achieve a balance between the inherent efficiency of centralized search, and the autonomy, load balancing and fault-tolerant features offered by distributed search. A super-peer node acts as a centralized resource for a number of regular peers, while super-peers connect to each other to form a network that exploits P2P mechanisms at a higher level.

The super-peer model can be advantageously adopted in large-scale Grids because it allows for a very efficient implementation of the information service and it is naturally appropriate for Grids. In fact, a large-scale Grid can be viewed as a network interconnecting small-scale, proprietary Grids; each of these Grids, which from now on will be referred to as a Physical Organization (PO), is composed of a set of hosts within one administrative domain. Within each PO, one or more nodes, e.g. those that have the largest capabilities, can act as super-peers, while the other nodes can use super-peers to access the Grid and search for resources and services. Note that a PO is not the same as a Grid Virtual Organization (VO), which is defined as a short-lived organization that spans administrative boundaries [3], even if in some cases the two concepts can coincide; in particular when a VO represents resources that are long-lived within one administrative domain.

This paper examines how the super-peer model can handle membership management and resource discovery services in a multi-organizational Grid. A simulation analysis evaluates the performance of a resource discovery protocol; presented results can be used to tune protocol parameters in order to increase search efficiency. The remainder of the paper is organized as follows. Section 2 discusses related work. Section 3 introduces the super-peer model, and shows how it can be used in large-scale Grids, in particular in service-oriented Grid frameworks. A discovery protocol based on the super-peer model is proposed and discussed. Section 4 analyzes the performance of the proposed discovery protocol by means of an event-driven simulation framework. The influence of network and protocol parameters on performance indices is evaluated, so that the protocol can be tuned to increase search efficiency. Section 5 concludes the paper.

2. Related work

Discovery services in P2P networks can be classified as using unstructured or structured approaches to search resources. Gnutella [7] and Kazaa [9] are examples of unstructured P2P networks: hosts and resources are made available on the network without a global overlay planning. Structured P2P networks, such as CAN [14], Chord [17] and Pastry [16], use highly structured overlays and exploit a Distributed Hash Table (DHT) to route queries over the network. A DHT is a data structure for distributed storing of pairs (key, data) which allows for fast locating of data when a key is given.

Peer discovery and membership services are mainly used for the construction and the start up of P2P networks. Such services can be provided using a very large variety of techniques. For example, Gnutella provides a number of well known “cache servers” that store the addresses of peers that can accept connection requests. In Chord, a peer that wants to join the network must contact a known peer, already included in the Chord ring, and request to it the addresses of its future predecessor and successor peers in the ring; afterwards, the new peer will ask these neighbour peers to be added to the ring. The FLAPPS system [15] is a P2P infras-

tructure which has the merit of flexibly combining a number of different styles of peer network construction methods as well as different resource discovery protocols.

Membership and resource discovery services are also key issues in Grid systems. Today, a centralized or hierarchical approach is usually adopted. For example, in the Globus Toolkit 2, or GT2 [1], a node that wants to connect to the Grid registers at a centralized index server, the Globus Index Information Server (GIIS), and periodically sends to that server information about the resources offered to other nodes. GIIS servers are organized according to a hierarchical approach. Query messages are delivered to a high-level GIIS and then possibly forwarded to lower-level information servers.

The information model exploited in the Globus Toolkit 3, or GT3, built upon OGSA, is based on Index Services [6], a specialized type of Grid Services. Index Services are used to aggregate and index *Service Data*, i.e. metadata associated to the resources provided by Grid hosts. There is typically one Index Service per organization but, in large organizations, several Index Services can be organized in a hierarchy. A similar approach is used in the WSRF-based Globus Toolkit 4: ServiceGroup services are used to form a wide variety of collections of WS-Resources, a WS-Resource being a Web service that is associated with a stateful resource.

Nowadays, the Grid community agrees that it is not efficient to devise scalable Grid resource discovery based on a centralized or hierarchical approach when a large number of Grid hosts, resources, and users have to be managed, also because of the heterogeneity of such resources [8].

Recently, super-peer networks have been proposed to achieve a balance between the inherent efficiency of centralized search, and the autonomy, load balancing and fault-tolerant features offered by distributed search. In [21], performance of super-peer networks is evaluated, and rules of thumb are given for an efficient design of such networks: the objective is to enhance the performance of search operations and at the same time to limit bandwidth and processing load. In [11], a general mechanism for the construction and the maintenance of a super-peer network is proposed and evaluated. In this work, a gossip paradigm is used to exchange information among peers and dynamically decide how many and which peers can efficiently act as superpeers.

Kazaa [9] designates the more available and powerful peers as supernodes. In Kazaa, when a new peer wants to join the network, it searches for the existing supernodes, and establishes an overlay connection with the supernode that has the shortest RTT. In [4], a framework that combines the structural DHT design with a multi-level architecture based on super-peers is proposed. Peers are organized in disjoint groups, and lookup messages are first routed to the destination group, then to the destination peer using an intra-group overlay. In [12], both resources and the content stored at peers are described by means of RDF metadata. Routing indices located at super-peers use such metadata to perform the routing of queries expressed through the RDF-QEL query language. Puppin et al. [13] proposed a Grid Information Service based on the super-peer model and its integration within OGSA. The Hop Counting Routing Index algorithm is used to exchange queries among the super-peers and, in particular, to select the neighbour super-peers that offer the highest probability of success.

3. A super-peer model for Grids

The super-peer model can be advantageously exploited in Grid systems for the deployment of information and discovery services. To maximize the efficiency of the super-peer model in Grids, it is useful to compare the characteristics of Grids and P2P networks.

(i) Grids are less dynamic than P2P networks, since Grid nodes and resources often belong to large enterprises or public institutions and security reasons generally require that Grid nodes authenticate each other before accessing respective resources.

(ii) Whereas in a P2P network users usually search for well defined resources (e.g. MP3 or MPEG files), in Grid systems they often need to discover software or hardware resources that match an extensible set of resource descriptions. Accordingly, while structured protocols, e.g. based on distributed indices, are usually very efficient in file sharing P2P networks, unstructured or hybrid protocols seem to be preferable in largely heterogeneous Grids. Another consequence is that the performance of a discovery service is influenced

by the distribution of classes of resources, a class of resource being a set of resources that satisfy some given constraints on resource properties, as discussed in Section 4.

(iii) In a Grid, it is feasible to identify, for each PO, a subset of powerful nodes having high availability properties; these nodes can be used as super-peers.

These considerations guided us through the design of membership and discovery services. For the sake of simplicity, we suppose that only one super-peer is associated to each PO, i.e. we will not consider *redundant* super-peers serving the same PO. A super-peer accomplishes two main tasks: it is responsible for the communications with the other POs, and it maintains metadata about all the nodes of the local PO. The set of nodes belonging to a PO (i.e. the super-peer and the ordinary nodes) is also referred to as a *cluster* in the following.

The membership protocol exploits the characteristics of *contact nodes*. A contact node is a Grid node that plays the role of an intermediary node during the building process of the Grid network. One or more contact nodes are made available by each organization. Whenever an organization wants to connect to the Grid, the corresponding super-peer contacts a subset of contact nodes and registers at those nodes. In turn, the selected contact nodes randomly choose a number of previously registered Grid super-peers and communicate their addresses to the requesting super-peer: these super-peers will constitute the neighbour set of the new Grid super-peers. A super-peer communicates with contact nodes either periodically or whenever it detects the disconnection of a neighbour super-peer, in order to ask for its substitution.

Fig. 1 shows a schema of the membership management protocol. A number of contact nodes are depicted, and for each of them the corresponding set of registered super-peers is reported. In Fig. 1(a), the super-peer *S* wants to connect to the Grid and selects two contact nodes. In Fig. 1(b), the selected contact nodes add *S* to the list of registered super-peers and respond to it by communicating the addresses of a number of super-peers, which will constitute the neighbour set of node *S*. The membership management protocol requires a proper setting of the *contact parameter K*, i.e. the number of contact nodes at which a new super-peers registers ($K = 2$ in Fig. 1).

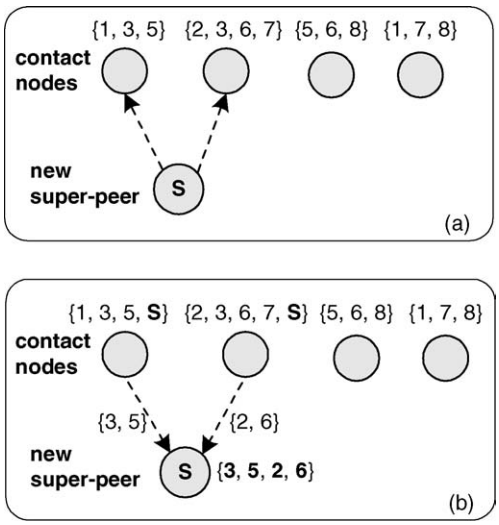

Fig. 1. The membership management protocol: (a) a new super-peer registers at a set of contact nodes and (b) receives the identities of its neighbour super-peers.

Ordinary nodes, i.e. simple peers, can be already connected to the super-peer before it initiates the joining procedure or can connect to the super-peer after it has joined the Grid.

As shown in Fig. 2, the super-peer model exploits the centralized/hierarchical information service provided by the Grid infrastructure of the local PO: e.g. the MDS-2 service of GT2 [5] or the Index Service of GT3 [6]. It is not necessary that the same Grid framework is installed in all the POs: it is only required that super-peers are able to communicate with each other using a

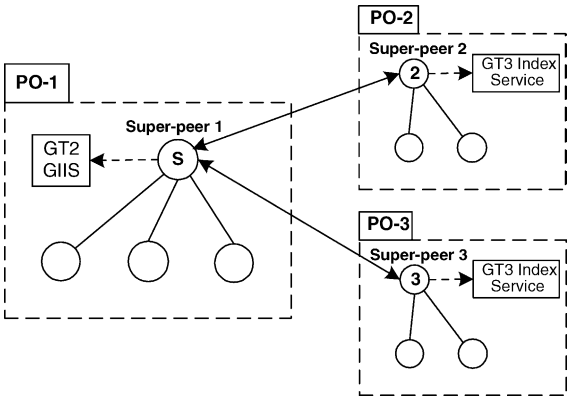

Fig. 2. A Grid network configuration exploiting the super-peer model.

standard protocol and that each super-peer knows how to interact with the information service of the local PO.

The resource discovery protocol, exploited by the discovery service, is defined as follows. Query messages generated by a Grid node are forwarded to the local super-peer. The super-peer examines the local information service to verify if the requested resources are present in some of the nodes belonging to the local PO, and in this case sends to the requesting node a queryHit containing the IDs of those nodes.

Furthermore, the super-peer forwards a copy of the query to a selected number of neighbour super-peers, which in turn contact the respective information systems and so on. Whenever a resource, matching the criteria specified in the query, is found in a remote PO, a queryHit is generated and is forwarded along the same path back to the requesting node, and a notification message is sent by the remote super-peer to the node that handles the discovered resource.

The set of neighbours to which a query is forwarded is determined through an empirical approach. Each super-peer maintains statistics on the number of queryHits received from all the known super-peers. The super-peer forwards a query to the neighbour super-peers from which the highest numbers of queryHits were received in the past. The maximum number of neighbours to which a query is forwarded can be tuned on the basis of the network configuration, as discussed in Section 4.

A number of techniques are adopted to decrease the network load. (i) The number of hops is limited by a Time-To-Live (TTL) parameter; the TTL is decremented only when the query is forwarded between two super-peers, i.e. between two different POs. (ii) Each query message contains a field used to annotate the nodes that the query traverses along its path. A super-peer does not forward a query to a neighbour super-peer that has already received it. (iii) Each super-peer maintains a cache memory where it annotates the IDs of the last received query messages. A super-peer discards the queries that it has already received. (iv) Whenever a super-peer, after receiving a query, finds several resources that satisfy the query constraints in the local PO, it constructs and forwards only one queryHit message containing the IDs of the nodes that own those resources.

Note that techniques (ii) and (iii) are used to avoid the formation of cycles in the query path, and are complementary, since technique (ii) can *prevent* cycles only in particular cases (i.e. when a query, forwarded by a super-peer, is subsequently delivered to the same super-peer), whereas technique (iii) can *remove* cycles in all the other cases (e.g. when *two* copies of a query, sent by a super-peer *A* to two distinct super-peers *B* and *C*, are subsequently both delivered to the remote super-peer *D*).

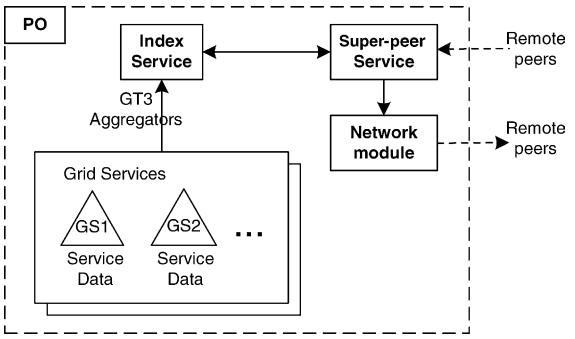

Fig. 3. Implementation of the super-peer model using the GT3 framework.

Fig. 3 shows how the GT3 information service is used in a PO to implement the super-peer model. Such architecture extends the one presented in [18]. The Index Service subscribes to the Service Data contained in the Grid Services published on the nodes of the local PO. Specialized GT3 aggregators periodically collect Service Data, which typically contain metadata information about Grid Services, and send it to the Index Service. The Superpeer Service is a static Grid Service that processes requests coming from the remote POs, queries the Index Service to find resources matching the query constraints, and forwards query and queryHit messages through the Network Module. Minor modifications will be needed in this architecture to replace the GT3 framework with the WSRF-based Globus Toolkit 4.

A simplified version of the resource discovery algorithm, executed by a Superpeer Service when receiving a query from an external PO, is reported in Fig. 4.

4. Simulation analysis

The performance of the resource discovery protocol described in Section 3 was analyzed in order to assess its effectiveness in a Grid environment and estimate

```
// Nh = max number of neighbours
// q.list: list of hosts traversed by the query q
// q.sender: neighbour super-peer from which q has been received
// q.id: query identifier
// q.ttl: current value of ttl
For each incoming query q:
  If <q.id is in the cache> then queryInCache:=true;
  Else <put q.id in the cache>
  q.ttl -= 1;
  if ((q.ttl>0) and not queryInCache)
  {
    select at most Nh best neighbours
    for each selected neighbour n:
      if <n is not in q.list> {
        <Add this super-peer to q.list>
            forward a copy of q to n
      }
  }
  <ask the local information service for resources matching q>
  if  <there are such resources> {
    send to q.sender a queryHit containing the IDs of the nodes owning the
    discovered resources;
    send notifications to the hosts owning the resources;
  }
```

Fig. 4. The resource discovery algorithm executed by the Superpeer Service.

the influence of protocol parameters on performance indices. An event-based object-oriented simulator was used both for building a super-peer network, using the membership protocol, and for simulating the behaviour of the resource discovery protocol in very large Grid networks.

In this section, we introduce and discuss the main parameters and performance indices used to evaluate the resource discovery protocol, and describe the main components of the simulator. Then, we report results obtained for two main scenarios. In the first one, the Grid network has a fixed global size, whereas the mean cluster size changes. In the second scenario, the mean cluster size is constant and the Grid size changes.

4.1. Simulation parameters and performance indices

The performance of a resource discovery protocol depends on the distribution of resources among Grid hosts. As mentioned in Section 3, in Grid systems users often need to discover resources that belong to classes of resources, rather than a specific single resource. A class of resources is defined as the set of resources that satisfy some given constraints on resource properties. For example, when building a distributed data mining application [10], a user may need to discover a software that performs a clustering task on a given type of source data. A query, containing the appropriate constraints, is generated to find such resources on the Grid; at a later time, one of the discovered resources will be chosen by the Grid scheduler for execution. This is a common problem in Grids where resources and/or services must be searched to be used in complex applications. Therefore, the performance of a resource discovery protocol in a Grid is strictly related to the categorization of heterogeneous resources in a given application domain.

We assumed, as in [8], that the average number of elementary resources offered by a single node (peer or super-peer) remains constant as the Grid size increases. This average value was set to 5, and a gamma stochastic function was used to determine the number of resources owned by each node. However, as the Grid size increases, it becomes more and more unlikely that a new node connecting to the Grid provides resources

Table 1
Simulation parameters

Parameter	Value
Grid size (number of nodes), N	10–10000
Mean cluster size, C	1–5000
Overall number of resources offered by the Grid vs. the Grid size	$5N$
Overall number of resource classes offered by the Grid vs. the Grid size	$5(\log_2 N)^2$
Maximum number of neighbours to which a super-peer can forward a query, Nh	2–8
Time to live, TTL	1–5
Mean query generation time, MQGT (s)	300
Mean internal hop time (between peer and super-peer) (ms)	10
Mean external hop time (between two super-peers) (ms)	50
Mean elaboration time at super-peers (ms)	100

belonging to a new resource class. Therefore, we assumed that the overall number of distinct resource classes offered by the Grid does not increase linearly with the Grid size. We adopted a logarithmic distribution: the number of resource classes offered by a Grid network with N nodes (where N is comprised between 10 and 10,000) is equal to $5(\log_2 N)^2$. As an example, a Grid having 1024 nodes provides 5120 resources belonging to 500 different classes.

Tables 1 and 2 report, respectively, the simulation parameters and the performance indices used in our analysis. During a simulation run, each peer or super-peer node generates queries at random times. In particular, the mean query generation time MQGT is the average interarrival time between two successive query requests issued by the same node. MQGT is a stochastic variable having a gamma probability distribution function with a shape parameter equal to 2, and a mean value equal to 300 ms. For each generated query, the simulator randomly selects the class of the resources that the user needs to discover in accordance with a uniform stochastic distribution.

Among the performance indices, Nres is deemed to be more important than the probability of success Psucc, since it is often argued that the *satisfaction of the query* depends on the number of results (i.e. the number of discovered resources) returned to the user that issued the query: for example, in [22] a resource discovery operation is considered *satisfactory* only if the number of results exceeds a given threshold. The message load L should obviously be kept as low as possible. This performance index often counterbalances the success indices, in the sense that high success probabilities are achievable at the cost of having high elaboration loads. The ratio R is an index of efficiency: if we succeed in increasing the value of R, a higher relative number of queryHit messages, containing useful results, is generated or forwarded with respect to the overall number of messages. Response times are related to the *time to satisfaction* experienced by a user: to calculate them, we assumed that the mean hop time is equal to 10 ms for an internal hop (i.e. a hop between a peer and the local super-peer) and to 50 ms for an external hop (i.e. a hop between two super-peers). Furthermore, we assume that a super-peer spends a mean elaboration time of 100 ms to process an incoming query.

4.2. Description of the simulation environment

The simulation environment includes two components: a network generator and a discrete-event continuous-time simulator.

The network generator, written in Java, is used to simulate the construction of a Grid network. Such a construction is driven by a set of network parameters among which: the Grid size, the mean cluster size, the resource distribution (Section 4.1), the contact parameter. The contact parameter, defined in Section 3, is

Table 2
Performance indices

Performance index	Definition
Probability of success, Psucc	Probability that a query issued by a peer will succeed, i.e. will be followed by at least one queryHit
Mean number of results, Nres	Mean number of resources that a node discovers after its query
Message load, L	Frequency of all messages received by a node (messages/s), including queries, queryHits, notifications
queryHits/messages ratio, R	Number of queryHits received by a node divided by the overall number of messages received by that node
Response times, Tr, Tr(K), Tr(L)	Mean amount of time (s) that elapses between the generation of a query and the reception of a generic result (average response time), of the kth result, of the last result

set to 2. The network generator adopts an incremental approach: at each step a new super-peer is added to the Grid, according to the membership protocol described in Section 3, and a number of peers are connected to that super-peer, thus forming a new physical organization connected to the Grid. The output of the network generator is a simple script describing the Grid topology and the distribution of resources on Grid nodes; this script is passed as an input to the discrete-event simulator.

The discrete-event simulator, written in C++, is fully object-oriented and designed upon objects which simulate the behaviour of Grid/P2P components and are able to exchange messages among them. Every time an object receives a message, it performs a related procedure and possibly sends new messages to other objects. The objects types defined in the simulator are the following:

- `SuperPeer`, which models a super-peer serving a PO. In particular, if the deployed Grid framework is the Globus Toolkit 3, a `SuperPeer` object simulates the behaviour of a Superpeer Service (see Fig. 3) and executes the algorithm shown in Fig. 4. Each `SuperPeer` object is connected to a number of similar objects serving the neighbour POs, to a number of Peer objects corresponding to the local peers, to a `UserAgent` object and to an `InfoService` object.
- `Peer`, which models a simple peer located within a PO. Such an object is connected to the local `SuperPeer` object and to a `UserAgent` object.
- `UserAgent` generates queries on behalf of a user. Each `UserAgent` object is connected to a `Peer` or `SuperPeer` object, in order to model the behaviour of a user attached to a Grid node. If such a node is a simple peer, queries are forwarded by the `Peer` object to the local `SuperPeer` object; otherwise queries are forwarded by the `SuperPeer` object to a number of adjacent `SuperPeers`.
- `Event`, which embodies a message exchanged among `UserAgent`, `Peer` and `SuperPeer` objects. An `Event` objects is characterized through its source and destination objects, its message delivery time and its type. Possible `Event` types are:
  (a) `GeneratedQuery` (message sent by a `UserAgent` to the attached `Peer` or `SuperPeer` object when a new query is generated by the user);
  (b) `Query` (message exchanged between `Peer` and `SuperPeer` objects to search for resources belonging to a given class);
  (c) `queryHit` (message exchanged between `Peer` and `SuperPeer` objects to carry results to the user that originated a query);
  (d) `Notification` (message sent by a `SuperPeer` object to a local `Peer` that owns a requested resource; such a `Peer` is so notified that it can receive a download request from the node that originated the query message).
- `InfoService`, which simulates the behaviour of the Grid information service of a PO (e.g. the GT3 Index Service) and is connected to the `SuperPeer` serving that PO. An `InfoService` manages information about the resources located in the local PO and is contacted by the `SuperPeer` object to search for the resources specified by a query message.
- `Event Dispatcher`, which manages events, stores them in a queue ordered by message delivery times, and dispatches them to destination objects.

4.3. Performance evaluation

The proposed discovery protocol was first analyzed for a Grid network having a fixed number of nodes (10,000, including super-peers and simple peers), and a mean cluster size C ranging from 1 (corresponding to a fully decentralized P2P network, in which peers and super-peers coincide) to 5000 (corresponding to a Grid composed of only two clusters). We tested different values of `Nh` and `TTL`, in order to analyze how those parameters can be tuned to improve performance when the mean cluster size is known. Notice that an estimated value of the mean cluster size can be computed by exchanging information among super-peers.

It should be remarked that, though the numerical values of the performance curves reported in this section are obviously driven by the values of the simulation parameters that have been chosen, the general shape of such curves can give a useful insight into the qualitative behaviour of the resource discovery protocol. However, for the sake of clearness, our discussion about the simulation results will often refer to exact numerical values of parameters and performance indices represented in these curves.

Results shown in Figs. 5–14 are obtained with `Nh` equal to 4. In Fig. 5, the probability of success is plotted

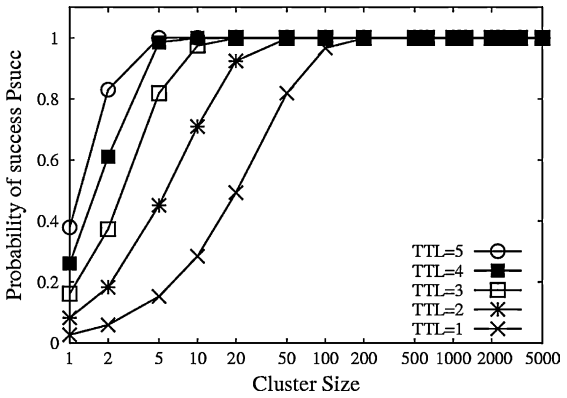

Fig. 5. Probability of success w.r.t. the cluster size, for different values of TTL and Nh = 4.

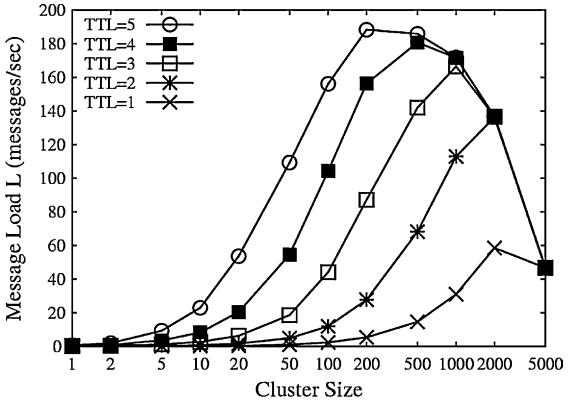

Fig. 8. Message load at super-peers w.r.t. the cluster size, for different values of TTL and Nh = 4.

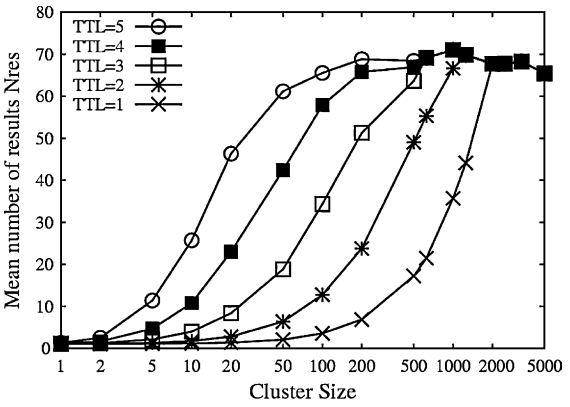

Fig. 6. Mean number of results w.r.t. the cluster size, for different values of TTL and Nh = 4: overall results.

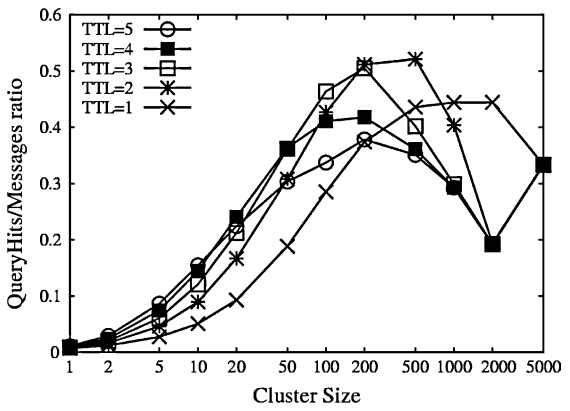

Fig. 9. QueryHits/messages ratio at super-peers w.r.t. the cluster size, for different values of TTL and Nh = 4.

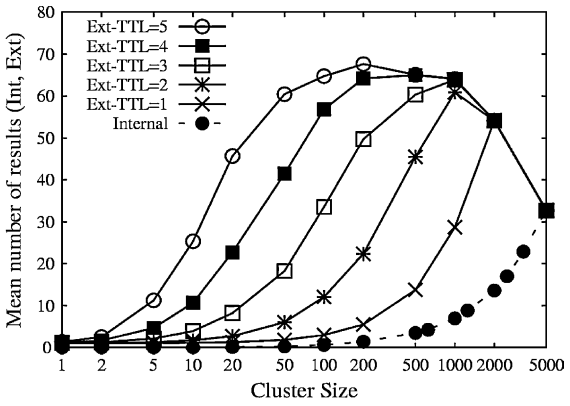

Fig. 7. Mean number of results w.r.t. the cluster size, for different values of TTL and Nh = 4: internal and external results.

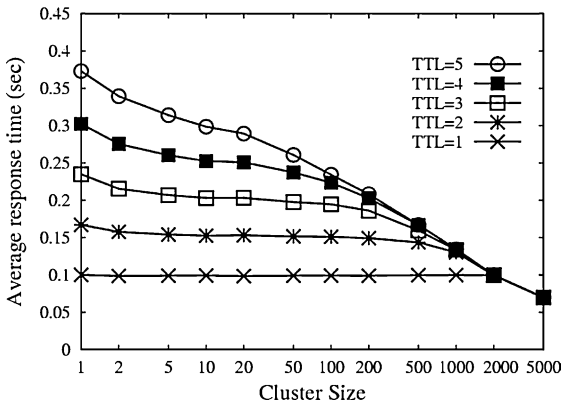

Fig. 10. Average response times w.r.t. the cluster size, with Nh = 4, for different values of TTL.

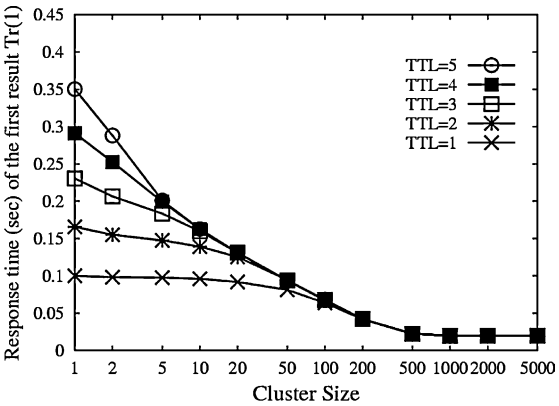

Fig. 11. Response times of the first result w.r.t. the cluster size, with Nh = 4, for different values of TTL.

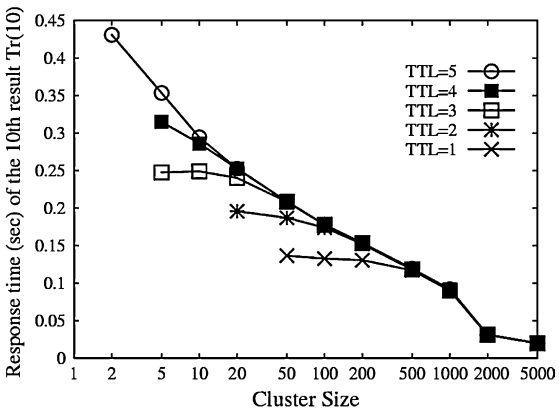

Fig. 12. Response times of the 10th result w.r.t. the cluster size, with Nh = 4, for different values of TTL.

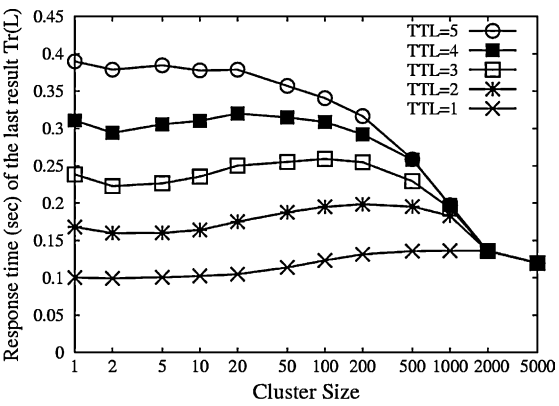

Fig. 13. Response times of the last result w.r.t. the cluster size, with Nh = 4, for different values of TTL.

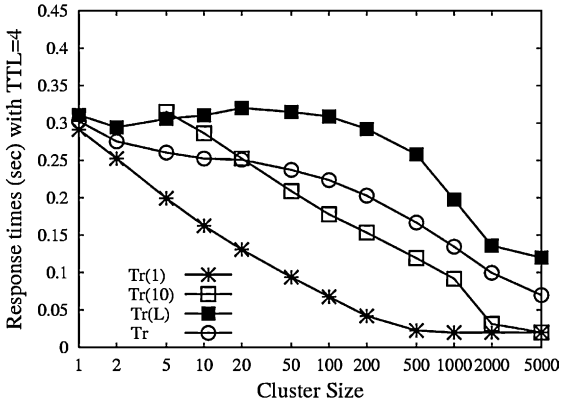

Fig. 14. Response times w.r.t. the cluster size, with Nh = 4 and TTL = 4: comparison between Tr, Tr(1), Tr(10) and Tr(*L*).

versus *C*, for TTL values ranging from 1 to 5. It is observed that, if the mean cluster size is low, a high TTL value is necessary to explore a significant portion of the Grid and achieve an acceptable probability of success. With larger cluster sizes (higher than 100), the probability of success is close to 1, but the figure does not allow for figuring out how many results can be expected on average.

Fig. 6 reports the mean number of discovered resources versus *C*, for TTL values ranging from 1 to 5. It appears that, as long as *C* is lower than 1000, the number of results can be notably increased by increasing the TTL value. Beyond that threshold, curves related to different values of TTL tend to converge, meaning that the protocol allows for exploring the entire Grid, irrespective of the TTL value. This information can be exploited when tuning the value of TTL. For example, if we want to maximize the number of results, with a value of *C* equal to 100, the TTL value should be 5 or higher, whereas if the value of *C* is higher than 500, it is almost ineffective to increase the TTL value beyond 3. Moreover, the very small number of results obtained for a decentralized network, i.e. with a cluster size equal to 1, demonstrates the great advantage that derives from the use of the super-peer model.

While Fig. 6 reports the overall number of results expected when issuing a query, Fig. 7 distinguishes between internal results (corresponding to resources discovered within the local cluster) and external results (coming from remote clusters). With a fixed TTL, the number of external results increases as the cluster size

increases, because a higher number of peers is searched in remote clusters. However, in a Grid composed by very large clusters, it is observed that the number of external results begins to decrease because a considerable number of peers is located in the local cluster; accordingly, the contribution of internal peers to the overall number of results becomes significant. Note that the number of internal results is obviously independent from the TTL value.

From Fig. 8, we see that a high processing load at super-peers is a toll to pay if a high number of results is desired. Indeed, the curves of message load show a trend similar to the curves, observed in Fig. 7, related to the number of external results.

A trade-off should be reached between maximizing the number of results and minimizing processing load; to this aim, we calculated the R index at super-peers. From Fig. 9 we see that R, for a fixed value of TTL, initially increases with C, as a result of the fact that the number of incoming queryHits experiences a higher increase rate than the overall number of messages. Beyond a threshold value of C, which depends on the value of TTL, an opposite trend is observed; the number of received queryHits falls down, due to the fact that a consistent percentage of peers is located within the local cluster. Remind that a super-peer receives queryHits only from remote clusters, since it already knows the resources offered by the local cluster. Values of R converge to 1/3 with $C = 5000$, i.e. with only two clusters in the Grid, for the following reason: each super-peer receives comparable numbers of internal queries (queries from local peers), external queries (queries from the other super-peer) and queryHits, because the numbers of peers in the two clusters are similar and almost each query directed to the other super-peer is followed by one queryHit message.

Fig. 9 helps to identify, for a given value of C, the TTL value that maximizes protocol efficiency. As the mean cluster size increases, the optimal value of TTL becomes smaller and smaller. For example, if C is set to 100, the value of TTL that maximizes the ratio R is equal to 3, whereas if C is set to 500, the optimal TTL value is 2. It is interesting to note that the highest values of R are obtained for cluster sizes comprised between 200 and 500 and a TTL value equal to 2.

Values of response times versus the cluster size are reported in Figs. 10–14. Fig. 10 shows that the average response time Tr increases with the TTL value and decreases with the cluster size. The values of response times decrease as the cluster size increases, for two main reasons: (i) queries and queryHits traverse a smaller number of super-peers and (ii) a higher fraction of queryHits are internal queries, which are statistically received earlier than external queryHits.

The response time related to the first result Tr(1) is depicted in Fig. 11; if compared to Tr, curves of Tr(1) have a steeper slope because, as the cluster size increases, it is more and more likely that the first queryHit comes from a peer located within the local cluster, thus taking a shorter time to be delivered.

The response time related to the 10th response Tr(10) is reported in Fig. 12. Note that the curves of Tr(10) are depicted only for the values of cluster size and TTL that allow for obtaining at least 10 results with a significant probability. Fig. 13 shows the response time related to the last result received for a given query, Tr(L). This index corresponds to the amount of time after which all results are received; note the mean number of results was shown in Fig. 6. If compared with the other response times discussed so far, Tr(L) shows a dissimilar behaviour, since it slightly increases as the cluster size increases from 2 to a value that depends on the TTL value. The reason is that the mean number of results increases very rapidly with the cluster size, therefore there is a higher and higher probability that the last query, which is usually an external query, experiences a long response time.

In Fig. 14, all the discussed response time indices are compared for a fixed value of TTL, which is set to 4. It is observed that Tr(10) presents higher values, with respect to Tr, as long as the cluster size is smaller than 20. The reason is that the 10th result is not actually received for all queries, as can be derived from Fig. 12; Tr(10) is calculated only for the queries that are followed by at least 10 results, and for those queries the 10th result usually experiences a quite long delay. For larger cluster sizes, Tr(10) becomes lower than Tr and Tr(L) because the number of expected results is higher than 10.

Figs. 15 and 16 report, respectively, the values of Nres and L obtained for a TTL value equal to 4 and a variable number of neighbours Nh, in order to evaluate how Nh can be tuned to optimize performance. The number of results, depicted in Fig. 15, significantly increases with the value of Nh only if the cluster size is

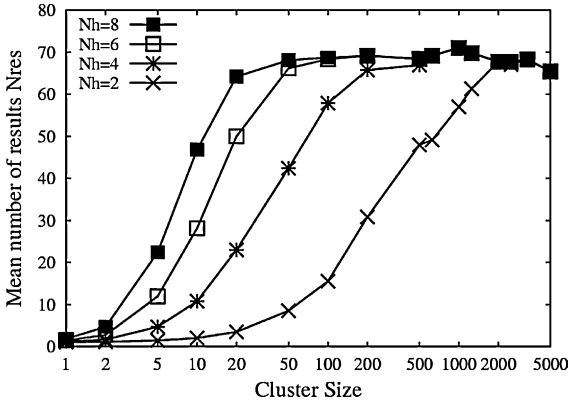

Fig. 15. Mean number of results w.r.t. the cluster size, for different values of Nh, with TTL = 4.

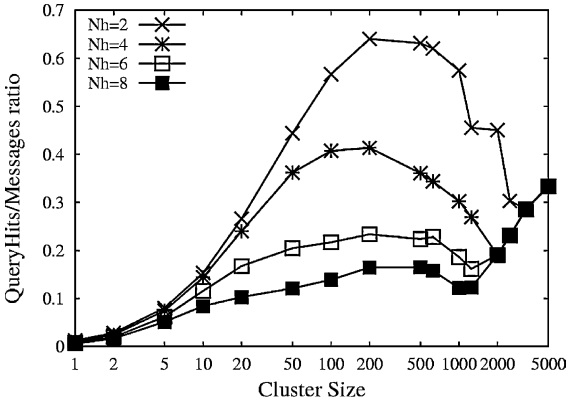

Fig. 17. QueryHits/messages ratio at super-peers w.r.t. the cluster size, for different values of Nh, with TTL = 4.

lower than 100; with larger clusters, a value of Nh equal to 4 is sufficient to achieve a high number of results. As expected, from Fig. 16 it appears that the processing load is highly increased if queries are forwarded to a large number of neighbours.

Fig. 17 shows that the values of *R* are maximized with Nh equal to 2, and decrease as the number of neighbours increases. From an analysis of Figs. 15, 16 and 17, we can conclude that it is not convenient to set Nh to a value higher than 4 if the cluster size exceeds 100, because we would increase the network and processing load without significantly increasing the number of results. For example, from Fig. 15, we can deduce that, with *C* equal to 200, the mean number of results does not appreciably increase if Nh is set to a value greater than 4, whereas by using Nh = 8 the message load increases significantly (see Fig. 16) and the value of *R* decreases (see Fig. 17).

Finally, the performance of the super-peer model was analyzed for different Grid sizes. The mean cluster size was set to 10, while the number of Grid nodes was varied from 10 (corresponding to a super-peer network having only one cluster) to 10,000. The value of Nh was set to 4. It appears from Fig. 18 that an increase in the TTL value allows for discovering a notably higher number of resources only with networks having more than 1000 nodes. As usual, a similar effect is observed on the processing load (Fig. 19); as discussed above, a trade-off may be obtained by analyzing the efficiency index *R*.

Fig. 20 shows that the optimum TTL value, i.e. the value that maximizes *R*, increases with the Grid size.

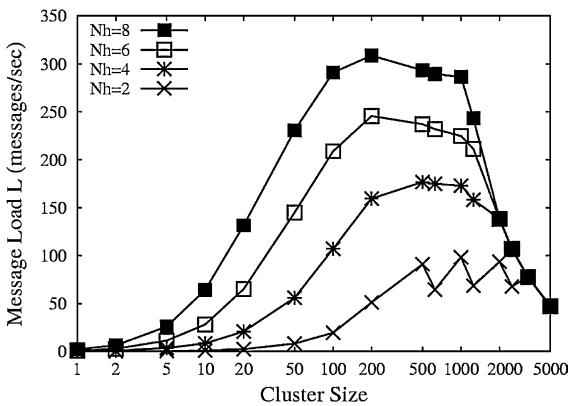

Fig. 16. Message load at super-peers w.r.t. the cluster size, for different values of Nh, with TTL = 4.

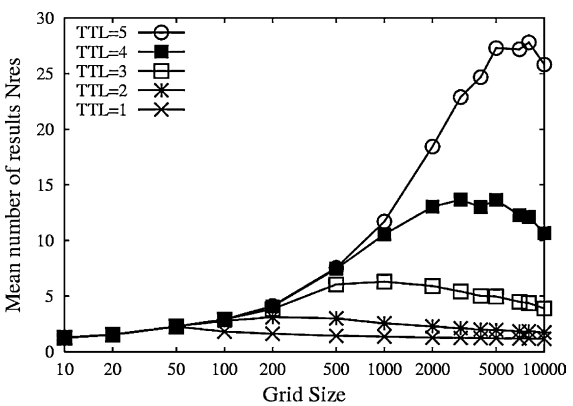

Fig. 18. Mean number of results w.r.t. the Grid size, for different values of TTL, with Nh = 4.

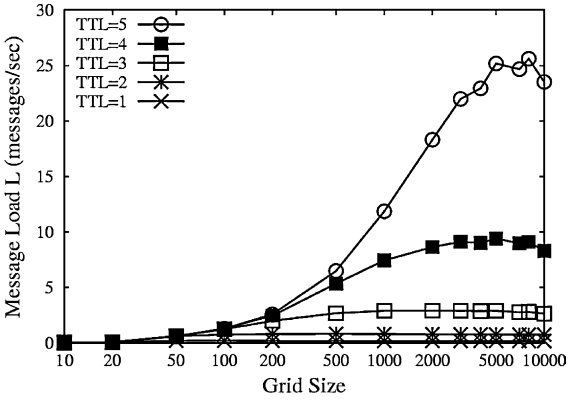

Fig. 19. Message load at super-peers w.r.t. the Grid size, for different values of TTL, with Nh = 4.

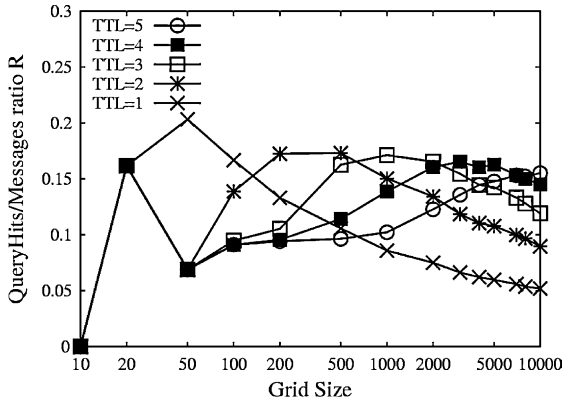

Fig. 20. QueryHits/messages ratio at super-peers w.r.t. the Grid size, for different values of TTL, with Nh = 4.

For example, in a Grid with 1000 nodes the maximum value of R is obtained with a `TTL` equal to 3, whereas, in a Grid with 10,000 nodes, R is maximized with a `TTL` equal to 5. As a consequence, `TTL` should be set to a value equal or greater than 5 only if the number of nodes exceeds 5000; for smaller networks, tuning decisions should take into account that a high value of `TTL` can slightly increase the number of results but surely decreases the overall efficiency.

5. Conclusions

The P2P model is a distributed computing paradigm that is used to harness the computing, storage, and communication power of hosts in the network to make their underutilized resources available to others. P2P shares this goal with the Grid, which is designed to provide access to remote computing resources for high-performance and data-intensive applications.

Resource discovery in Grid environments is currently based on centralized or hierarchical models. Because such information systems are built to address the requirements of organizational-based Grids, they do not deal with more dynamic, large-scale distributed environments. The number of queries in such environments makes a client–server approach ineffective. Future large-scale Grid systems should implement a P2P-style decentralized resource discovery model.

The super-peer model is a novel approach that facilitates the convergence of P2P models and Grid environments, since a super-peer serves a single organization in a Grid and at the same time connects to other super-peers to form a peer network at a higher level. This paper proposed an approach based on the super-peer model for handling membership management and resource discovery services in a Grid. In particular, a resource discovery protocol was presented and discussed. We reported simulation results, obtained with different network configurations, which give some general insight into how real implementations of the discovery protocol might behave. In particular, we evaluated the influence of protocol parameters (such as the number of neighbour super-peers and the time to live) on performance indices. Performance evaluation allows for efficiently tuning the values of such protocol parameters when a real Grid must be deployed.

Acknowledgements

This work has been partially supported by the Italian MIUR project Grid. It (RBNE01KNFP), and the research work of D. Talia is also carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

References

[1] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, Grid information services for distributed resource sharing, in: Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing, San Francisco, CA, USA, 2001.

[2] I. Foster, C. Kesselman, J.M. Nick, S. Tuecke, Grid services for distributed system integration, IEEE Comput. 35 (6) (2002) 37–46.
[3] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, Int. J. Supercomput. Appl. 15 (3) (2001).
[4] L. Garces-Erce, E. Biersack, P. Felber, K. Ross, G. Urvoy-Keller, Hierarchical peer-to-peer systems, in: Proceedings of EuroPar, Klagenfurt, Austria, 2003.
[5] The Globus Alliance: Information Services in the Globus Toolkit 2 Release, http://www.globus.org/mds/mdstechnologybrief_draft4.pdf.
[6] The Globus Alliance: Information Services in the Globus Toolkit 3 Release, http://www.globus.org/mds.
[7] The Gnutella Protocol Specification v.0.4. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.
[8] A. Iamnitchi, I. Foster, J. Weglarz, J. Nabrzyski, J. Schopf, M. Stroinski, in: Grid Resource Management (ed.), A Peer-to-Peer Approach to Resource Location in Grid Environments, Kluwer Publishing, 2003.
[9] Kazaa, http://www.kazaa.com.
[10] C. Mastroianni, D. Talia, P. Trunfio, Managing heterogeneous resources in data mining applications on Grids using XML-based metadata, in: Proceedings of Heterogeneous Computing Workshop, Nice, France, 2003.
[11] A. Montresor, A robust protocol for building superpeer overlay topologies, in: Proceedings of the International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 2004.
[12] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst, A. Loser, Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks, in: Proceedings of World Wide Web Conference, Budapest, Hungary, 2003, pp. 536–543.
[13] D. Puppin, S. Moncelli, R. Baraglia, N. Tonellotto, F. Silvestri, A peer-to-peer information service for the Grid, in: Proceedings of International Conference on Broadband Networks, San Jose, CA, USA, 2004.
[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, A scalable content-addressable network, in: Proceedings of ACM SIGCOMM, San Diego, CA, USA, 2001.
[15] P. Reiher, M. Scott, Forwarding Layer for Application-level Peer-to-Peer Services (FLAPPS), http://flapps.cs.ucla.edu/.
[16] A. Rowstron, P. Druschel, Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems, in: Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms, 2001.
[17] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup service for internet applications, in: Proceedings of ACM SIGCOMM, San Diego, CA, USA, 2001.
[18] D. Talia, P. Trunfio, A P2P Grid services-based protocol: design and evaluation, Proceedings of the European Conference on Parallel Computing (EuroPar), LNCS 3149, 2004.
[19] D. Talia, P. Trunfio, Towards a synergy between P2P and Grids, IEEE Internet Comput. 7 (4) (2003) 94–96.
[20] The Web Services Resource Framework, http://www.globus.org/wsrf/.
[21] B. Yang, H. Garcia-Molina, Designing a super-peer network, in: 19th International Conference on Data Engineering, IEEE Computer Society Press, Los Alamitos, CA, USA, 2003.
[22] B. Yang, H. Garcia-Molina, Efficient search in peer-to-peer networks, in: Proceedings of ICDCS, Wien, Austria, 2002.

Carlo Mastroianni is a researcher at the Institute for High Performance Computing and Networks of the Italian National Research Council (ICAR-CNR) in Cosenza, Italy. He received a PhD in computer engineering from the University of Calabria, Italy, in 1999. His research interests include distributed systems and networks, grid computing, peer-to-peer networks, multimedia distribution networks. As a lecturer, he teaches computer networks at the University of Calabria and Databases at the "Parthenope" University of Naples, Italy.

Oreste Verta is a research associate at the Faculty of Engineering at the University of Calabria, Italy. He received the Laurea degree in computer engineering from the University of Calabria in 2004. His research interests include grid computing, peer-to-peer networks, web service orchestration.

Domenico Talia is a full professor of computer science at the Faculty of Engineering at the University of Calabria, Italy, a research associate at ICAR-CNR in Rende, Italy and a partner at Exeura s.r.l. He received the Laurea degree in physics at University of Calabria. His research interests include grid computing, distributed knowledge discovery, parallel data mining, parallel programming languages, cellular automata and peer-to-peer systems. Talia published three books and more than 150 papers in international journals such as Communications of the ACM, IEEE Computer, IEEE TKDE, IEEE TSE, IEEE TSMC-B, IEEE Micro, ACM CS, FGCS, Parallel Computing, IEEE Internet Computing and conference proceedings. He is a member of the editorial boards of the Future Generation Computer Systems journal, the International Journal on Web and Grid Services, the Parallel and Distributed Practices journal, and the Web Intelligence and Agent Systems International journal. He is a member of the European Commission expert panel that wrote the "Next Generation Grid" report and is a member of the Executive Committee of the CoreGRID Network of Excellence and member of the European Knowledge Discovery Network of Excellence. He is serving as a program committee member of several conferences and is a member of the ACM and the IEEE Computer Society.