A Super-Peer Model for Building Resource Discovery Services in Grids: Design and Simulation Analysis*

Carlo Mastroianni[1], Domenico Talia[2], and Oreste Verta[2]

[1] ICAR-CNR 87036 Rende (CS), Italy
mastroianni@icar.cnr.it
[2] DEIS University of Calabria, 87036 Rende (CS), Italy
{talia,verta}@deis.unical.it

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 Virtual Organization (VO) in a Grid, and manages metadata associated to the resources provided by the nodes of that VO. Super-peers connect to each other to form a peer network at a higher level. This paper examines how the super-peer model can be used to 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. 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 favor 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 [1]), or the Web Services Resource Framework (WSRF) [12]) will support the convergence between the two models, since Web Services can be used to implement P2P interactions between hosts belonging to different domains.

P2P techniques can be particularly useful to manage two key services in Grid information systems: *membership management* (or simply *membership*) and *resource discovery*. The objective of a membership management service is twofold: 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 having given characteristics.

* This work was partially supported by the Italian MIUR FIRB Grid.it project RBNE01KNFP on High Performance Grid Platforms and Tools and by the project KMS-Plus funded by the Italian Ministry of Productive Activities.

P.M.A. Sloot et al. (Eds.): EGC 2005, LNCS 3470, pp. 132 – 143, 2005.

In currently deployed Grid systems, resources are often owned by research centres, public institutions, or large enterprises: in such organizations hosts and resources are usually 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.

Super-peer networks have been proposed [13] 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 allows for a very efficient implementation of the information service and it is naturally appropriate for large-scale Grids. A Grid can be viewed as a network composed of small-scale, proprietary Grids, called Virtual Organizations (VOs). Within each VO, one or more nodes, e.g. those that have the largest capabilities, can act as super-peers, while other nodes can use super-peers to access the Grid.

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 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

P2P membership and discovery services can be classified as using unstructured or structured approaches to search resources. Gnutella [4] is an example of unstructured P2P network: hosts and resources are made available on the network without a global overlay planning. Structured P2P networks, such as Chord [10], 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.

Membership and resource discovery services are also key issues in Grid systems. A centralized or hierarchical approach is usually adopted. The information model exploited in the Globus Toolkit 3 (GT3) the version of Globus built upon OGSA, is based on Index Services [3], 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 Virtual 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.

Today, 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.

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 [13], 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 [7] a general mechanism for the construction and the maintenance of a superpeer network is proposed and evaluated. A gossip paradigm is used to exchange information among peers and dynamically decide how many and which peers can efficiently act as superpeers.

In [8] 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. [9] 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 VO, 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 VO, i.e. we will not consider *redundant* super-peers. Whenever a VO wants to join the Grid, the corresponding super-peer must know the address of at least another super-peer and explores the topology of the system to constitute its *neighbour set*. A super-peer accomplishes two main tasks: it is responsible for the communications with the other VOs, and it maintains metadata about all the nodes of the local VO. The set of nodes belonging to a VO (i.e. the super-peer and the ordinary nodes) is also referred to as a *cluster* in the following.

As shown in Figure 1, the super-peer model exploits the centralized/hierarchical information service provided by the Grid infrastructure of the local VO: e.g. the MDS-2 service of GT2 [2] or the Index Service of GT3 [3]. It is not necessary that the same Grid framework is installed in all the VOs: it is only required that the super-peers are able to communicate with each other using a standard protocol and that each super-peer knows how to interact with the information service of the local VO.

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 VO, 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 VO, 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.

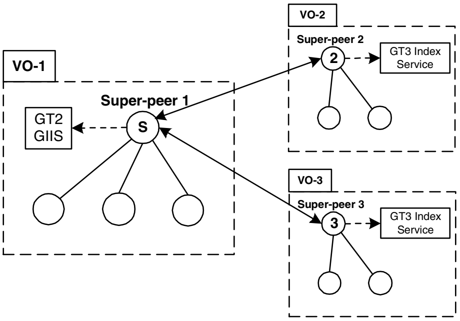

Fig. 1. A Grid network configuration exploiting the super-peer model

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 when the query is forwarded between two super-peers, i.e. between two different VOs. (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 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 VO, it constructs and forwards only one queryHit message containing the IDs of the nodes that own those resources. Techniques (ii) and (iii) are used to avoid the formation of cycles in the query path: 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).

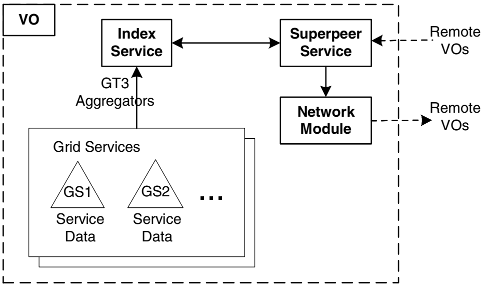

Fig. 2. Implementation of the super-peer model using the GT3 framework

```
// v = 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 v 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. 3. The resource discovery algorithm executed by the Superpeer Service

Figure 2 shows how the GT3 information service is used in a VO to implement the super-peer model. Such architecture extends the one presented in [11]. The Index

Service subscribes to the Service Data contained in the Grid Services published on the nodes of the local VO. 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 VOs, 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 that is going to be released.

A simplified version of the resource discovery algorithm, executed by a Superpeer Service when receiving a query from an external VO, is reported in Figure 3.

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

4.1 Simulation Parameters and Performance Indices

The performance of a resource discovery protocol depends on the distribution of resources among the hosts of a network. As mentioned in Section 3, in Grid systems users often need to discover resources that belong to classes of resources, rather than well defined resources. 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 [6], a user may need to discover a software that performs a clustering task on a given type of source data. 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 [5], that the average number of elementary resources offered by a single node (peer or super-peer) remains constant as the network 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 network size increases, it becomes more and more unlikely that a new node connecting to the network provides resources belonging to a new resource class. Therefore, we assumed that the overall number of distinct resource classes offered by a network does not increase linearly with the network 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 10000) is equal to `5*(log₂N)^2`. As an example, a Grid having 1024 nodes provides 5120 resources belonging to 500 different classes.

Table 1 reports the simulation parameters and the performance indices used in our analysis. During a simulation run, a node randomly selects, with a frequency determined by the mean query generation time `MQGT`, a resource class, and forwards a query for resources belonging to that class. 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, a resource discovery operation could be 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 sometimes are only 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 msec for an internal hop (i.e. a hop between a peer and the local super-peer) and to 50 msec for an external hop (i.e. a hop between two super-peers).

Table 1. Simulation parameters and performance indices

Parameter	**Value**
Network size (number of nodes) **N**	10 to 10000
Mean cluster size **C**	1 to 5000
Overall number of resources offered by the network vs. the network size	5N
Overall number of resource classes offered by the network vs. the network size	$5(\log_2 N)^2$
V: maximum number of neighbours to which a super-peer can forward a query	2 to 8
Time to live **TTL**	1 to 5
Mean query generation time **MQGT**	300 sec

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 (queries, queryHits, notifications) received by a node.
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 that elapses between the generation of a query and the reception of a generic queryHit (average response time), of the kth queryHit, of the last queryHit.

4.2 Performance Evaluation

The proposed discovery protocol was first analyzed for a Grid network having a fixed number of nodes (10000, 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 network composed of two clusters). We tested different values of v 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.

Results shown in Figures 4-6 are obtained with v equal to 4. Figure 4(a) reports the mean number of discovered resources versus *C*, for TTL values ranging from 1 to 5. It appears that performance values increase with the TTL value as long as *C* is lower than 1000. Beyond this threshold, curves related to different values of TTL tend to converge. 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. 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 comes from the use of the super-peer model. From Figure 4(b) we see that a high processing load at super-peers is a toll to pay if a high number of results are desired. Indeed the curves of message load show a trend similar to the trend observed in Figure 4(a), except for networks having very large clusters, in which a high percentage of resources is discovered within the local VO.

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 Figure 5 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 are internal, i.e. situated within the local VO. Remind that a super-peer receives queryHits only from remote VOs, since it knows the resources offered by the local VO. Values of R converge to 1/3 with *C*=5000, i.e. with only two clusters in the network, 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.

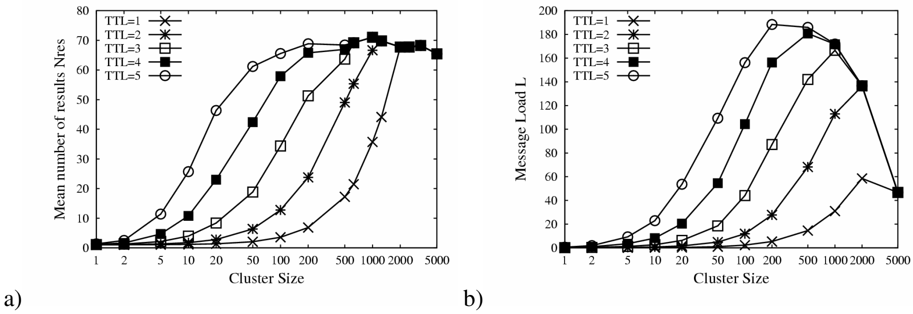

Fig. 4. Mean number of results (a) and message load at super-peers (b) w.r.t. the cluster size, for different values of TTL and v=4

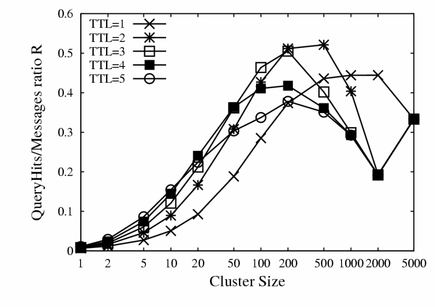

Fig. 5. QueryHits/messages ratio R at super-peers w.r.t. the cluster size, calculated for different values of TTL and v=4

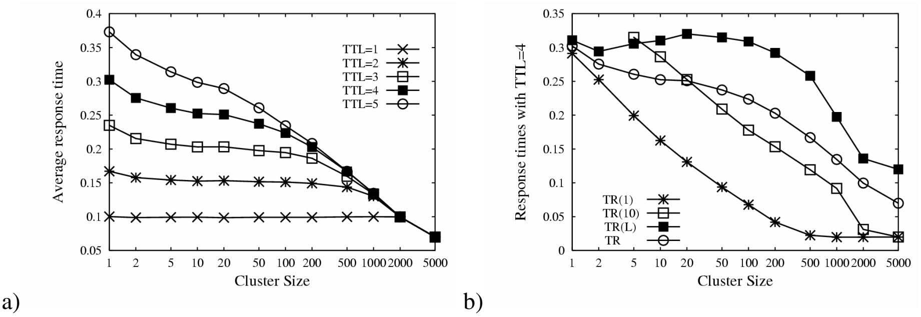

Fig. 6. Response times w.r.t. the cluster size, with v=4. (a): average response times for different values of TTL; (b): comparison between Tr, Tr(1), Tr(10) and Tr(L), with TTL=4

Figure 5 helps to identify, for a given value of *c*, the TTL value that maximizes the efficiency of the network. As the mean cluster size increases, the optimal value of TTL becomes smaller and smaller. For example, the optimal value of TTL is equal to 3 if *c* is 100, and to 2 if *c* is 1000. 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 Figure 6. Figure 6(a) shows that the average response time increases with the TTL value and decreases with the cluster size. Moreover, it is confirmed that a high value of TTL is advantageous only for small/medium cluster sizes. In Figure 6(b), different response time indices are compared (TTL is set to 4): plotted curves are related to the average response time, the response times of the first and the 10th queryHit, and the response time of the last queryHit received for a given query. The values of these indices 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 ones, which are statistically received earlier than external queryHits. The Tr(L) index is an exception: it slightly increases as the cluster size increases form 2 to 100. The reason is that within that range of the cluster size the number of results increases very rapidly, therefore there is a higher and higher probability that the last query, which is normally an external query, experiences a long response time.

Figure 7 reports the values of Nres and R obtained for a TTL value equal to 4 and a variable number of neighbours v, in order to evaluate how v can be tuned for a fixed value of TTL. The number of results significantly increases with the value of v only if the cluster size is lower than 100; with larger clusters, a value of v equal to 4 is sufficient to achieve a high number of results. Figure 7(b) shows that the values of R are maximized with v equal to 4. We can conclude that it is not convenient to set v to a value higher than 4 if the cluster size exceeds 100, because we would increase the network and processing load without increasing the number of results.

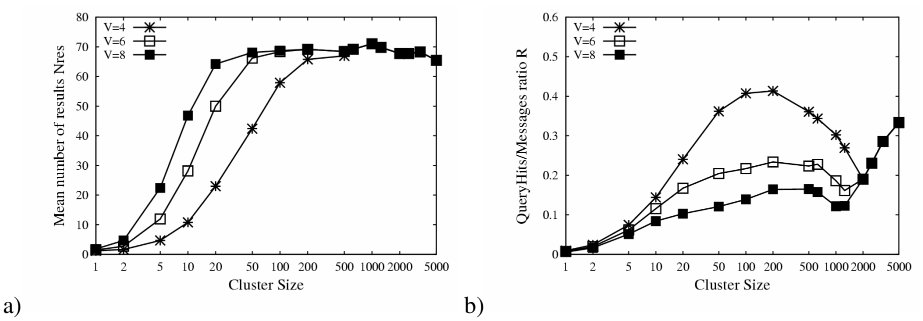

Fig. 7. Mean number of results (a), and queryHits/messages ratio R at super-peers (b) w.r.t. the cluster size, for different values of v, and TTL=4

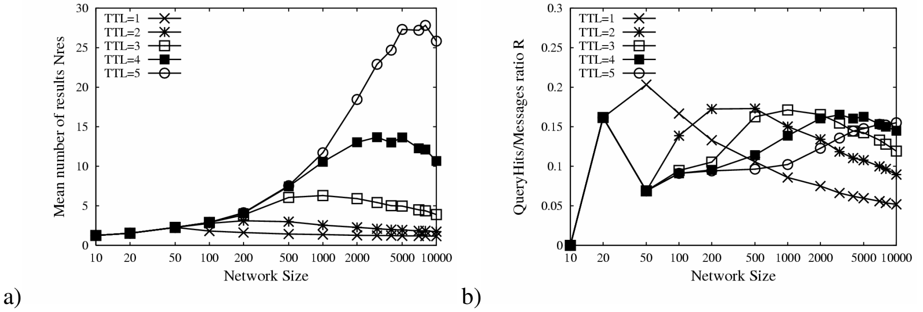

Fig. 8. Mean number of results (a) and queryHits/messages ratio at super-peers (b) w.r.t. the network size, for different values of TTL, and v=4

Finally, the performance of the super-peer model was analyzed for different network 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 10000. The value of v was set to 4. It appears from Figure 8(a) that an increase in the TTL value allows for increasing the number of results only in networks having more than 1000 nodes. Moreover, Figure 8(b) shows that the optimum TTL value, i.e. the value that maximizes R, increases with the network size. For example, in a network with 1000 nodes the maximum value of R is obtained with a TTL equal to 3, whereas in a network with 10000 nodes R is maximized with a TTL equal to 5. Thus 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

Resource discovery in Grid environments is currently based on centralized 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 Virtual Organization (VO) 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 and 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 protocol parameters when a real Grid must be deployed.

References

1. Foster, I., Kesselman, C., Nick, J., S. Tuecke, S.: Grid services for distributed system integration, IEEE Computer, 35(6) (2002) 37-46
2. The Globus Alliance: Information Services in the Globus Toolkit 2 Release, http://www.globus.org/mds/mdstechnologybrief_draft4.pdf
3. The Globus Alliance: Information Services in the Globus Toolkit 3 Release, http://www.globus.org/mds
4. The Gnutella Protocol Specification v.0.4. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf
5. Iamnitchi, A., Foster, I., Weglarz, J., Nabrzyski, J., Schopf, J., Stroinski, M.: A Peer-to-Peer Approach to Resource Location in Grid Environments, eds. Grid Resource Management, Kluwer Publishing (2003)
6. Mastroianni, C. Talia, D., Trunfio, P.: Managing Heterogeneous Resources in Data Mining Applications on Grids Using XML-Based Metadata, Proc. of International Parallel and Distributed Processing Symposium (IPDPS), Nice, France (2003)

7. Montresor, A.: A Robust Protocol for Building Superpeer Overlay Topologies, Proc. of the International Conference on Peer-to-Peer Computing, Zurich, Switzerland (2004)
8. Nejdl, W., Wolpers, M., Siberski, W., Schmitz, C., Schlosser, M., Brunkhorst, I., Loser, A.: Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks, Proc. of World Wide Web Conference, Budapest, Hungary (2003) 536-543
9. Puppin, D., Moncelli, S., Baraglia, R., Tonellotto, N., Silvestri, F.: A Peer-to-peer Information Service for the Grid, Proc. of International Conference on Broadband Networks, San Jose, CA, USA (2004)
10. Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup service for internet applications, Proc. of ACM SIGCOMM, San Diego, CA, USA (2001)
11. Talia, D., Trunfio P.: A P2P Grid Services-Based Protocol: Design and Evaluation, Proc. of the European Conference on Parallel Computing (EuroPar), LNCS 3149 (2004)
12. The Web Services Resource Framework, http://www.globus.org/wsrf/
13. Yang, B., Garcia-Molina, H.: Designing a Super-Peer Network, 19th Int'l Conf. on Data Engineering, IEEE Computer Society Press, Los Alamitos, CA, USA (2003)