A Scalable and Effective Architecture for Grid Services' Discovery

Huang Lican, Wu Zhaohui, and Pan Yunhe

College of Computer Science, Zhejiang University, Hangzhou 310027, PRC
lchuang@cs.zju.edu.cn

Abstract. This paper presents Virtual and Dynamic Hierarchical Architecture (VDHA) for discovering Grid services with high performance. Services discovery based on VDHA has scalable, autonomous, efficient, reliable, quick responsive, and fully. We propose two service discovery algorithms. Full Search Query and Discovery Protocol (FSQDP) discovers the nodes that match the request message from all N nodes, which has time complexity $O(\log N)$, space complexity $O(_{Nvg}\ )$ ($_{Nvg}$ is node numbers of each virtual group), and message-cost $O(N)$, and Domain-Specific Query and Discovery Protocol (DSQDP) searches nodes in only specific domains, which has time complexity $O(_{Nvg}\ )$, space complexity $O(_{Nvg}\ )$, and message-cost $O(_{Nvg}\ )$. In this paper, we also describe VDHA, its formal definition and related Grid Group Management Protocol.

1 Introduction

Grid [1] technology is one of the most important one come forth in recent years. The recent big progress is that scientists [2, 3, 4] have proposed service-oriented architectures such as Open Grid Services Architecture (OGSA) [2] that integrates the so called computational/data Grid architecture [1] with Web services [5].

In the service-oriented architecture of Grid, how to find Grid services is an important issue. In Globus and Web services, the services are published and discovered with centralized mode, which has bad scalability and a single point of failure. P2P [6] has good scalability, but it has some challenges such as security, network bandwidth, and architecture designs, and has difficult to search services which are described by many entities, especially by ontology terms.

We present a scalable service discovery based on Virtual and Dynamic Hierarchical Architecture (VDHA) (some ideas were formed in the paper [7,8,9]) to solve the above problems. Our method is efficient, reliable, quick responsive, and fully searching.

The structure of this paper is as following: section 2 presents related work; section 3 describes VDHA and its Grid Group Management Protocol (GGMP); section 4 gives out high performance discovery protocols and performance analysis; and finally we give out conclusions.

2 Related Work

Globus [10] defines a single, unified access mechanism for a wide range of information, called as the Metacomputing Directory Service (MDS) [10]. Building on the data representation and application programming interface defined by the Lightweight Directory Access Protocol (LDAP) [11], MDS defines a framework in which information of interest in distributed computing applications can be represented. Information is structured as a set of entries, where each entry comprises zero or more attribute-value pairs. In Globus, Grid Information Index Servers (GIISs) [1] is used to support arbitrary views on resource subsets. Grid Resource Registration Protocol (GRRP) [1] is used to register resources. Grid Resource Information Protocol (GRIP)[1] is used to access information about entities .

Web service uses WSDL [12] to describe services. WSDL is an XML format language for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure oriented information. WSFL (Web Services Flow Language) [13] and XLANG [14] describe how services can be composed together, and the behavior/interaction protocol of a Web service. UDDI (Universal Description, Discovery and Integration) [15] is used to enable online registry and the publishing and dynamic discovery of Web services offered by businesses.

The above services descriptions and discoveries are centralized, with bad scalability and with a single point of failure.

Iamnitchi et.al [16] combine P2P technologies with Grid ones to discover resources. But, algorithms they used are not effective. Although P2P technologies based on Distributed_Hash_Table (DHT) technologies such as Pastry [17], CAN [18], and CHORD [19] have effective in time complexity, they can not be used for discovering the services which are needed to be described with a lot of entities or semantic languages and the services which are discovered by partial-match searching.

Chander, A. et.al [20] propose NEVRLATE for scalable and expressive peer-to-peer (P2P) networking efficient resource discovery. They maintain two two-dimensional sets of servers, occurring in one 'horizontal' dimension for registration, and occurring in the other 'vertical' dimension for lookup. It needs to register every service into $\sqrt{N}$ (N is the number of all nodes).

Most Internet search-type lookup services fail to be responsive. Search engines like "Google" delay response of dynamic changed services in days.

The services discovery based on VDHA we present here is scalable, effective, fully, quick responsive, and does not register the services.

3 Overview of VDHA

3.1 Description of VDHA

VDHA is a virtual and dynamic hierarchical architecture (see Fig.1) in which Grid nodes are grouped virtually. Nodes can join the group and leave the group dynamically. The groups are virtually hierarchical, with one root-layer, several middle-layers, and many leaf virtual groups (these groups are called VOs). Among these nodes of VOs, one(just one) node (called as gateway node) in each group is chosen to form upper-layer groups, from the nodes of these upper-layer groups to form upper-upper-layer groups in the same way, and this way is repeated until to form one root-layer group. In the same group all nodes are capable to be gateway node. Gateway node is the node which is not only in low-layer group, but also in up-layer group. Gateway nodes will forward the low-layer group's status information to all the nodes in the up-layer group, and distribute the upper-layer group's status information to all the nodes in the lower-layer group. The numbers of nodes in a VO can be dynamically changed by the way that the node can dynamically join and leave the VO. A VO may join and leave the Grid system as a whole, and this autonomous property makes the large scalable systems possible.

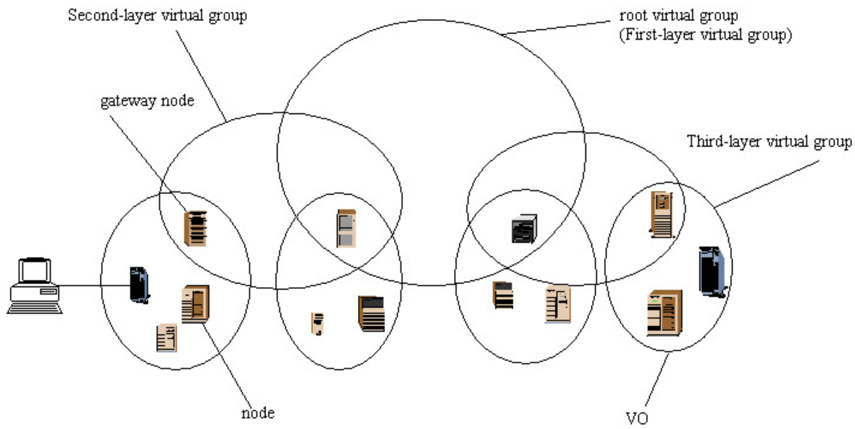

Fig. 1. Structure of VDHA
Note: There are 13 nodes in the grid system. These nodes are grouped as 4 VOs. The number of nodes in each VO is 4,3,3,3 respectively. From each VO we choose one node as gateway node to form two up-layer groups with each having 2 nodes. Then from these two groups, one node each was chosen to form a root group.

3.2 Formal definition of VDHA

Definition 1. *Grid node (denoted by p) is the node in the Grid system. All p form a set PS, that is,* $PS = \{p_i | i \in N\}, N = \{1 \ldots n\}$,*here, n is the number of the Grid nodes, each* p_i *has ID (usually Internet IP address).*

Definition 2. *Entrance node (denoted by ent) is a Grid node, which is an entrance point for users to login into the Grid system.*

Definition 3. *Client host (denoted by cli) is an apparatus (such as desktop computer, palm, PDA, mobile computer, etc), which are used by users to login into the Grid system and to do the business.*

Definition 4. *Gateway node (denoted by gn) is a Grid node which takes coordinate functions in several different layer virtual groups.*

Definition 5. *Virtual group (denoted by VG) is formed virtually by the Grid nodes. VG^i_α means the group is in the ith layer and the name of this virtual group is α. The virtual group is identified by its group name and layer number.*

Definition 6. *Coordinator of virtual group (denoted by cvg) is a gateway node taking coordinate functions in the virtual group. The symbol $cvg^i_\alpha(cvg^i_\alpha \in VG^i_\alpha)$ means that it is a gateway node in the ith layer $\alpha -$ named virtual group which functions as coordinator.*

Definition 7. *Virtual group tree (denoted by VGT) is hierarchical tree formed by virtual groups. In VGT there is a root virtual group (denoted by RVG), many leaf VGs called as virtual organization (denoted by VO). $VO^m_{\alpha_m}$ means that the virtual organization is in the m-th layer and its name is α_m. The order of layers is counted from RVG, which is defined as the first-layer VG. VG except VO is formed purely by gateway nodes. VO is formed by Grid nodes with one (and just one) gateway node. RVG can not be a VO, and VO can be within all the layers except the first layer. N^i_α is the numbers of the nodes in VG^i_α. N^i_g is the number of virtual groups in the ith-layer of VGT.*

Definition 8. *VDHA is a virtual group tree with depth of at least two layers. VDHA has dynamic properties in the number of Grid nodes, layers and virtual groups, virtual group compositions, and so on. In VDHA, we have following properties:*

1. $VG^i_\alpha = \{gn \in VG^{i+1}_\beta | \beta \in A^i\}, i > 0, VG^i_\alpha$ is not a VO, here, A^i is the subset of the names of the i-th layer virtual groups. (This sentence means that the VG is formed from lower-layer groups.)

2. If $gn_1 \in VG^i_\alpha \cap gn_1 \in VG^{i+1}_\beta$ and $gn_2 \in VG^i_\alpha \cap gn_2 \in VG^{i+1}_\beta$, then $gn_1 = gn_2$.

3. Each VG has one and only one node (cvg) which takes coordinate functions.

4. Grid node p can join more than one VO

5. $PS = VO_1 \cup VO_2 \cup \ldots \cup VO_{n1}$, Here, n1 is the number of virtual organization.

6. If p satisfies the following condition: $p \in VO^m_{\alpha_m} \cap p \in VO^{m-1}_{\alpha_{m-1}} \ldots p \in VO^{m-k}_{\alpha_{m-k}}, m \geq 2, k \geq 1$,the p is gateway node. It is expressed with symbol $gn(m, k, \alpha_{m-k} \ldots \alpha_{m-1}\alpha_m)$. The meanings of parameter values are: m is the layer order of VO in VGT $(gn \in VO)$; k is the number of layers in which the gateway node functions; $\alpha_{m-k} \ldots \alpha_{m-1}\alpha_m$ are the names of the virtual groups from $V^m_{\alpha_m} to VO^{m-k}_{\alpha_{m-k}}$. Symbol $gn^i_{\alpha_i} \in VG^i_{\alpha_i}$ means that the gateway node is in the ith layer group with name α_i.

3.3 Grid Group Management Protocol(GGMP)

GGMP has two functions. Firstly, it manages membership of virtual groups and the dynamic virtual group tree. Secondly, when a gateway node fails or leaves, it selects a new one with the maximum weight value from all the on-line nodes in the group the gateway node is involved with. The details of the algorithm are shown in paper [9].

To improve fault-tolerance every member in a virtual group logically contains the group membership list, name, and cvg, and the membership list, name, and cvg of the groups immediately above and below. When there are any changes to the membership of a virtual group, such as a node joining or leaving, these changes are forwarded to the coordinator of the group, which forwards the information to all the members in the group, and to the groups in the neighboring levels. When one coordinator in a virtual group fails, another node in the same virtual group will replace it.

A node can use Query and Discovery Protocols, which are described in the following sub-sections, to find a suitable group to join. By using the Query and Discovery Protocols a VO can see the virtual group tree, and find the right place in the tree structure to add itself. A VO can also create a new partial tree from any point of the VGT and add itself to the new tree path. For example, if in VGT there is a partial tree path as ALL_Science:Biology (Biology is not VO), and a VO with the domain Fish wants to create the ALL_Science:Biology:Animal:Fish tree path, the VO first adds Animal partial tree path to ALL_Science:Biology, and then adds itself to the ALL_Science:Biology:Animal tree path. Then GGMP sets the gateway node of this VO as a member of the Biology and Animal groups.

4 Query and Discovery Protocols

In VDHA, query and discovery protocols are used for querying and discovering some entities such as resources and services, virtual group name, node status, etc. Every node has resources and services which are described by WSDL or ontology languages, etc. Matching the request message is done by the agent of node which has the services. There are two kinds of QDP: Full Search Query and Discovery Protocol (FSQDP), which searches all nodes to find nodes that match the request message, and Domain-Specific Query and Discovery Protocol (DSQDP), which searches nodes in only specific domains.

4.1 Full Search Query and Discovery Protocol (FSQDP)

FSQDP first finds the root virtual group, and then the coordinator of virtual group of this group forwards the query message to its all members. All of these members execute parallel forwards of the message down to the members of their low-layer groups until leaf virtual groups as Fig. 2 shows.

Before describing the FSQDP algorithm, we list the definition of message routing primitives in Table 1, and give the algorithms for the Route_To_RVG

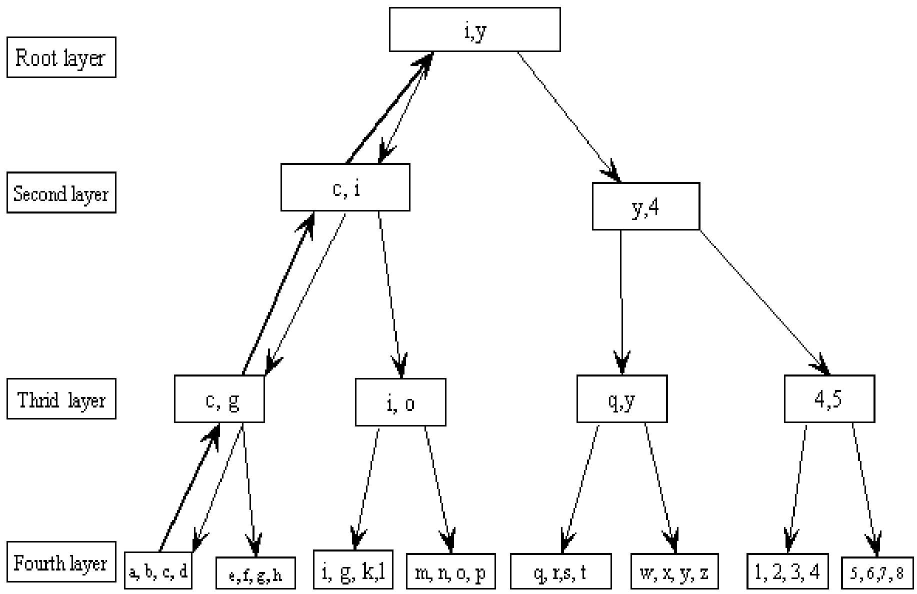

Fig. 2. FSQDP searching process

primitive which routes a query message up to the root virtual group, Route primitive which routes message in parallel to all nodes , and compare_service which compares the service at a node with the service requested in the message. In these algorithms, the query message is denoted by qmessage, and the result message is denoted by rmessage. We also assume that client host (cli) attaches entrance node (ent), and this entrance node is within a VO whose virtual group coordinator is cvg.

cvg.Route_To_RVG(qmessage)
  1. If $(Pnode(UPcvg(cvg)) == \phi)$ Return cvg;
  2. If $(Pnode(UPcvg(cvg)) <> Pnode(cvg))$ cvg.send (qmessage, UPcvg(cvg));
  3. UPcvg(cvg).Route_To_RVG(qmessage);

cvg.Route(qmessage)
  1. cvg.send (qmessage, $p_i \in VGroup(cvg) | p_i \neq cvg$);
  2.$if(Type(VGroup(cvg)) == VO)$
    $for(\forall p_i \in VGroup(cvg))$
      $if(p_i.compare\_service(qmessage) == true)$
        $p_i.send(rmessage, ent), ent.send(rmessage, cli)$;
  $else$ for$(\forall p_i \in VGroup(cvg))$ $LOWcvg(p_i, Layer(cvg)).Route(qmessage)$;

function pi.compare_service(qmessage) {

Table 1. Primitives and Functions

Description	Meaning
sender.send (message, receiver)	sender sends message to receiver.
sender.send (message,receiver $\in Set$)	sender sends message to all the receivers belonging to Set.
Pnode(cvg)	returns node ID of cvg.
Layer ($var^m_{\alpha_m}$)	returns the level order of cvg or gn.
UPcvg($cvg^m_{\alpha_m}$)	returns cvg in the next highest level.
VGroup ($cvg^m_{\alpha_m}$)	returns the virtual group containing $cvg^m_{\alpha_m}$.
Type(VG)	returns VG's type (VG or VO).

if (the semantic description of service in p_i suits the semantic description of the service in qmessage) return true;
else return false; }

The algorithm of FSQDP is as following:
Step 1 cli .send (qmessage, ent)
Step 2 ent.send (qmessage, cvg)
Step 3 cvg_RVG = cvg.Route_To_RVG(qmessage);
Step 4 cvg_RVG.Route(qmessage);

4.1.1 Performance Analysis of FSQDP

Let $N^i_{vg_{max}}$ be a maximum number of nodes in the i-th layer virtual groups , that is,
$N^i_{vg_{max}} = MAX(N^i_{\alpha 1}, \ldots N^i_{\alpha j} \ldots N^i_{\alpha n}), \alpha j \in$ name set of $VG^i, n = N^i_g$
,and suppose that the maximum time of sending a message is T_{max}, so , due to the parallel message forwarding , maximum total time, space needs and message costs of searching all nodes with FSQDP is as formula (1), (2) and (3).
$T_{all_{max}} = n \times T_{max} + T_{max} \times \sum_i^n (n^i_{vg_{max}} - 1)$ (1)
Here n is the number of layers.
$S_{max} = L \times n_{vg_{max}}$ (2)
where $n_{vg_{max}}$ is the maximum number of nodes in a virtual group, and L is the maximum number of layers
$Message\_cost = n + N - 1$ (3)
Because the number of layer is small, so the time complexity, space complexity and message cost are as formula (4), (5) and (6)
$T_{complexity} = O(n_{vg_{max}})$ (4)
$S_{complexity} = O(n_{vg_{max}})$ (5)
$Message\_cost_{complexity} = O(N)$ (6)
If we suppose all virtual groups have the same numbers of nodes (n_{vg}) and the time for sending a message (T) is constant, then the number of layers is $log_{nvg}N$, from (1) and (2) we have (7) and (8).
$T_{all} = log_{n_{vg}}N \times T \times n_{vg}$ or $T_{all} = n \times T \times \sqrt[n]{N}$ (7)

$S = \sqrt[n]{N} \times n_{vg}$ or $S_{max} = (\sqrt[n]{N})^2$ (8)

Figure 3 shows the influence of number of nodes in a virtual group on the time response. If we suppose that the time for transferring a message is 0.1 second, the algorithm needs less than 100 seconds to search all 10^{10} nodes with 200 nodes per virtual group and 5 layers ($log_{200}10^{10}$) layers.

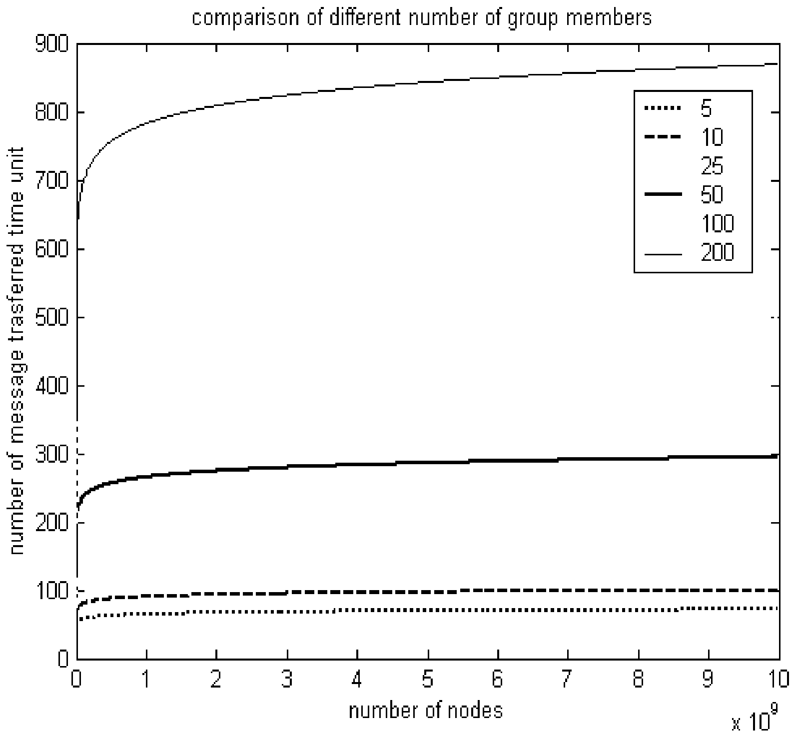

Fig. 3. influence of number of nodes in a virtual group on the time response

Figure 4 shows the influence of number of layers on the time response. Except for 3 layers, the response time varies little with the increase of nodes. Because more layers will increase the cost of Grid Group Management, the number of layers with range of 4-6 is the best.

How about the traffic? Does the algorithm cause a network jam? This problem is still under study. If we suppose that a message length is L_p , and N_l links distribute traffic load, then we have traffic per link as formulae (9).
$Traffic_{link} = (L_p \times (N + n - 1))/(n \times T \times \sqrt[n]{N} \times N_l)$ (9)

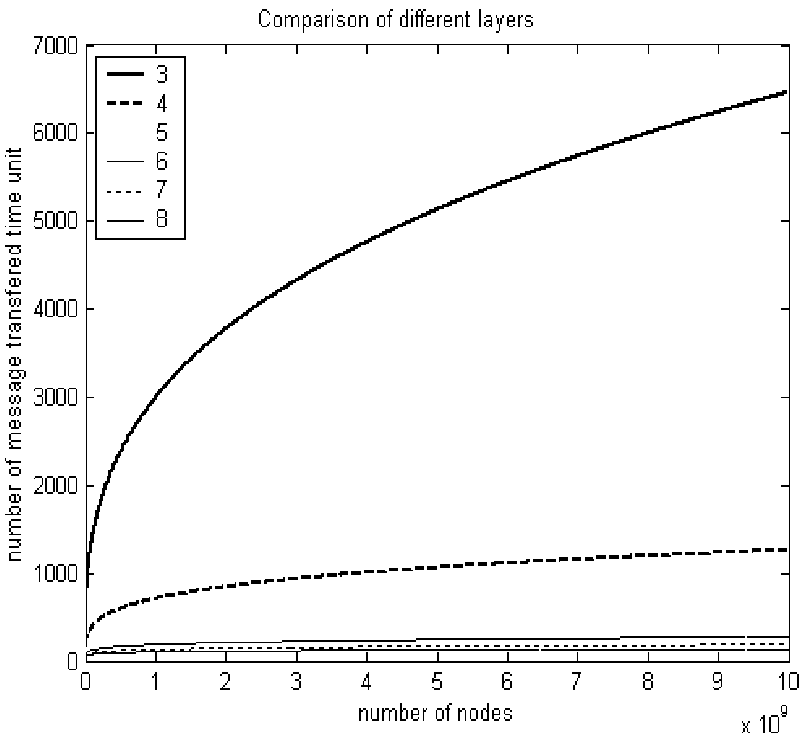

Fig. 4. influence of number of layers on the time response

From (9), we can approximately calculate the use of bandwidth. For example, if L_p is 1000 bits, n is 5, N is 1,000,000, T is 0.1 second, and N_l is 1000, then traffic per link is 126kbit/s.

4.2 Domain-Specific Query and Discovery Protocol (DSQDP)

FSQDP is effective, but may cause much traffic. Domain-Specific Query and Discovery Protocol has not this problem. To use this protocol, the object of virtual group must maintain the catalogue with classifying services from general to detail. It may be done by the nodes' joining the proper virtual group of Grid system. The protocol only searches the nodes whose catalogue matches the request group keywords as Fig. 5 shows.

The algorithm is similar to FSQDP. The different is described as following:

Function **keyword (qmessage, Layer (cvg))** returns keyword of cvg in the layer of Layer (cvg).
Function **keyword (Group (cvg))** returns group's keyword.

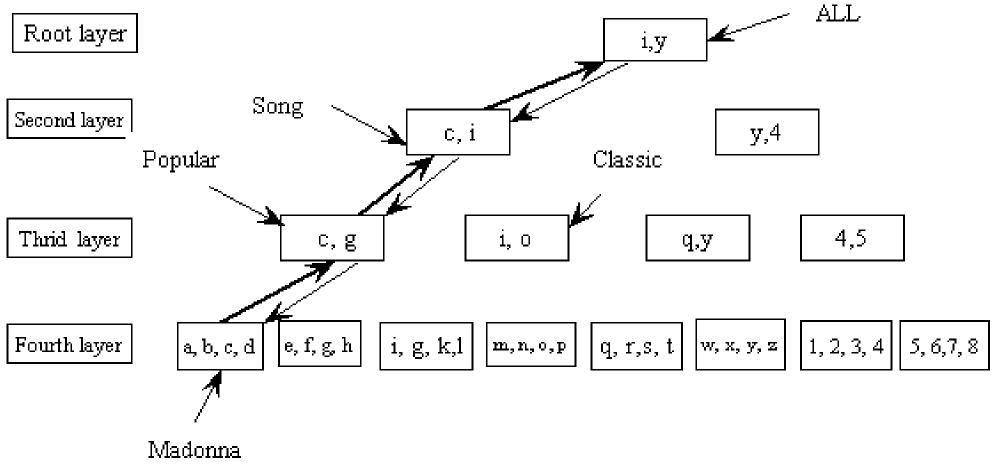

Fig. 5. DSQDP searching process

cvg.Route(qmessage)/* slight different with FSQDP*/
1. cvg.send (qmessage, $p_i \in VGroup(cvg)|p_i \neq cvg)$;
2.$if(Type(VGroup(cvg)) == VO)$
$for(\forall p_i \in VGroup(cvg))$
$if(p_i.compare\_service(qmessage) == true)$
$p_i.send(rmessage, ent), ent.send(rmessage, cli)$;
else for ($\forall p_i \in VGroup(cvg))$
if(keyword(qmessage, Layer(cvg)) == keyword(Group(cvg)))
$LOWcvg(p_i, Layer(cvg)).Route(qmessage)$;

4.2.1 Performance Analysis of DSQDP

The terms are the same as 4.1.1. We have:
$T_{all_{max}} = 2n \times T_{max} + T_{max} \times (n_{vg_{max}} - 2)$ (10)
$S_{max} = L \times n_{vg_{max}}$ (11)
$Message\_cost_{max} = 2n + n_{vg_{max}} - 2$ (12)
$T_{complexity} = O(n_{vg_{max}})$ (13)
$S_{complexity} = O(n_{vg_{max}})$ (14)
$Message\_cost_{complexity} = O(n_{vg_{max}})$ (15)

This protocol is effective and message cost is low. But, the resources and services must cluster according to hierarchical keywords.

5 Conclusion

The services discovery based on VDHA we presented is scalable, autonomous, efficient, reliable, quick responsive, and fully. FSQDP discovers the nodes that match the request message from all N nodes, which has time complexity $O(logN)$, space complexity $O(n_{vg})$ (n_{vg} is node numbers of each virtual group), and message-cost $O(N)$. Whereas, when the services are clustered as classification, we can use DSQDP to discover services with time complexity $O(n_{vg})$, space complexity $O(n_{vg})$, and message-cost $O(n_{vg})$. With each virtual group having 200 nodes and a 4 to 6 layer virtual group tree, the protocols are suitable to more than 10 billions of nodes. The services discovery based on VDHA is full decentralized, and avoiding of single point of failure. There are no needs for nodes to know all global names of groups or node identification, etc, because the groups are organized as a virtual group tree and the group and node properties can be obtained by Query and Discovery Protocol. The services discovery based on VDHA is un-related to services description languages, because it uses local agents of nodes to match the services. The only thing for agents to do is to obey the specification of service discovery message format and match services according to the local service description.

Acknowledgments

This paper is supported by virtual cooperative research project granted by the Ministry of Education of PRC.

References

1. I. Foster, C. Kesselman and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International Journal of High Performance Computing Applications*,15(3).200-222.2001, http://www.globus.org/research/papers/anatomy.pdf
2. I. Foster, Kesselman, J. M. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", 2002.2.17 http://www.globus.org/ research/papers/ogsa.pdf
3. David De Roure, Nicholas Jennings and Nigel Shadbolt, "Research Agenda for the Semantic Grid: A Future e-Science Infrastructure" ,2001,9 http://www.semantic grid.org/v1.9/semgrid.pdf
4. Rana O.F. and Walker D.W., "Service Design Patterns for Computational Grids" , *Patterns and Skeletons for Parallel and Distributed Computing*, edited by Fethi A. Rabhi and Sergei Gorlatch. Published by Springer-Verlag, 2002. ISBN 1-85233-506-9.
5. Grid Web Services Workshop.2001, http://gridport.npaci.edu/workshop/webserv01/agenda.html
6. D. Clark, "Face-to-Face with Peer-to-Peer Networking", *Computer*, Vol. 34, No.1, January 2001, pp.18-21

7. Huang Lican, Zhou xuezhong, Wu Zhouhui, et.al. "Virtual and Dynamic Hierarchical Architecture and Its Usage in an e-Science Application for Providing Knowledge Services", *In the proceedings of The Joint International Computer Conference 2002(JICC2002)*, ZheJiang University Press, pp159-165
8. Huang Lican, Wu Zhaohui and Pan Yunhe, "Virtual and Dynamic Hierarchical Architecture for e-Science and Related Protocols", *In the proceedings of 2002 International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES2002)*
9. Huang Lican, Wu Zhaohui and Pan Yunhe, "Virtual and Dynamic Hierarchical Architecture for Chinese University e-Science Grid", *In the proceedings of 2002 International workshop on Grid and Cooperative Computing(GCC2002)*, Publishing House of Electronics Industry ,pp297-311
10. I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", *International Journal of Supercomputer Applications*, 11(2): 115-128, 1997
11. M. Wahl, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3)", IETF RFC 2251, December 1997.
12. Christensen, E., Curbera, F., Meredith, G. and Weerawarana.,S., "Web Services Description Language (WSDL) 1.1". W3C, Note 15, 2001, http://www.w3.org/TR/wsdl.
13. "Web Services Flow Language (WSFL) Version 1.0", http://www4.ibm.com/ software/ solutions/ Webservices/pdf/WSFL.pdf
14. "Web Services for Business Process Design", http://www.gotdotnet.com/team/ xml_wsspecs/xlang-c/default.htm
15. "UDDI: Universal Description, Discovery and Integration", http://www.uddi.org/
16. A. Iamnitchi and I. Foster, " On Fully Decentralized Resource Discovery in Grid Environments", *International Workshop on Grid Computing 2001*
17. A. Rowstron and P. Druschel. "Pastry:Scalable, distributed object location and routing for large-scale peer-to-pear systems". *In Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms(Middleware)*,2001
18. S.Ratnasamy, P.Francis, K.Handley, R.Karp, and S.Shenker. "A scalable content-addressable network". *In Proceedings of ACM SIGCOMM 2001",2001*
19. *I.Stoica, R.Morris, D.Karger, M.F.Kaashoek, andH.Balakrishnan. "Chord: a scalable peer-to-peer lookup service for internet applications".* In Proceedings of ACM SIGCOMM2001*, 2001*
20. *Chander A.; Dawson, S.; Lincoln, P.; Stringer-Calvert, D., "NEVRLATE: Scalable Resource Discovery",* Proceedings CCGRID 2002. 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid