VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

TOWARDS SELF-RESOURCE DISCOVERY AND SELECTION MODELS IN GRID COMPUTING

M. S. Alzboon, A. S. Arif and M. Mahmuddin
Inter Networks Research Laboratory, School of Computing, Universiti Utara Malaysia, UUM Sintok, Kedah, Malaysia
E-Mail: mwafaqalzboon@gmail.com

ABSTRACT

Global computational grids nowadays are suffered from ossification problems due to the following fundamental challenges related to different existing solutions in grid computing: scalability, adaptability, security, reliability, availability and manageability. The management difficulty is due to heterogeneity, dynamicity and locality of the resources within global grid networks. Large-scale grids make the fundamental problem of resource discovery a great challenge. This paper presents a self-resource discovery mechanism (SRDM) that achieves efficient grid resource discovery and takes advantage of the strengths of both hierarchy and decentralized approaches that were previously developed for grid based P2P resource discovery. P2P systems offer potential strengths such as self-organization, self-healing, and robustness to failure or attacks. Unfortunately, the majority of existing Distributed Hash Table (DHT) based P2P overlays are lacking of attributes range queries that are familiar in resource discovery lookups. The proposed model builds an effective distributed hierarchy that providing scalable, decentralized resource discovery and allocation as well as load balancing for distributed computing using large-scale pools of heterogeneous computers. Fundamentally, SRDM employs the spatial index and partitions the overlay space to build a distributed quad tree; each computational resource in the network can calculate its Nodepower. Next, it encodes the information about each node's available computational resources power in the structure of the links connecting the nodes in the network. This distributed encoding is self-organized, with each node managing its in-degree and local connectivity via its available Nodepower. Assignment of incoming jobs to nodes with the freest resources is also accomplished by sampling it.

Keywords: grid computing, resource discovery, quad-tree.

INTRODUCTION

The Internet is used to browse and exchange information, exchange mail and publishing e-commerce and different kinds of Multimedia making it as an information highway of the current world [1]. Internet has become one of the members in the commodity list in the computing world. As a result, a new research has emerged, and attracted the attention of people around the globe to use idle computers that are connecting to the Internet. Computers are described as idle when they are not being used by any program or the computers that are linked to the Internet, but actually not executing any work or computation [2]. This condition offers the researchers in this area to employ the idle computers within the Internet for improved utilization. A grid is equivalent to an Electrical Grid (EG) where the whole consumers are linked to utilize it. Every day people utilize electricity without knowing where it is produced; which path it is selects to transmit power from the generators to its destination, and the source of the EG. Grid technology is rising as the next computing stage on the Internet. So, in the nearest future it is possible to use a computer with remotely located resources, immediately getting computing resources by plugging into the socket's wall as doing it for EG. The essential idea behind Grid Computing (GC) is dividing a huge job, which usually requires months to complete on individual computers, into smaller chunks that can be executed by computers during their idle times. At the end, the completed job is collected by these computers to provide the final result. This is not as simple as it seems, this is an extraordinary task that involves conceptual model, model planning, model design, model integration, and distribution for resources [3].

Grid computing uses middle-ware to coordinate disparate IT resources across a network. A grid middleware distribution is a software stack or a set of cooperating components, services and protocols which enable users to access to the distributed resources of a grid such as Advanced Resource Connector (ARC), European Middle-ware Initiative (EMI), gLite, Globus toolkit, GridWay, Open Middle-ware Infrastructure Institute United Kingdom (The OMII-UK distribution), Oracle Grid Engine, Alchemi, and Legion [4].

The rest of this paper is structured as follows. The state of the art in resource discovery is provided in the next section. Section 3 discusses open research issues in self-resource discovery. Section 4 illustrates our proposed model Finally Section 6 conclude this paper.

STATE OF THE ART IN RESOURCE DISCOVERY

Grid Information Service (GIS) is the most critical part being utilized as a hidden part; it is responsible for collecting and to supply updated information particularly for extremely dynamic resource information about grids resources. Moreover GIS offers tools in order to record resources, to inquire databases, and to delete left nodes. Many on hand GIS similar to MDS deal with mostly static information, but lack dynamic information provisioning and well-organized caching mechanisms. Applications executing in the Grid require efficient access to information about services and available resources. This information can have a considerable impact on planning, replica chosen, and scheduling

choices for workflow execution. Information about accessible resources and their qualities is regularly gathered by data administrations and served to applications by means of publish/subscribe or query/response interfaces.

Current solutions to GIS mechanism can fall generally into three categories; centralized, decentralized, and hierarchical model. However, each of these models has severe limitations in regard to scalability, adaptability, availability, and manageability. Some authors mentioned that the process of resource discovery and allocation (or scheduling) in grid is divided into three steps: discovery, selection of resources and job execution [5]. However, just a little would give the impression to have comparatively enhanced step of development and have large increased operational deployment by existing grid operators. Each system would have its local scheduler to establish how its job queue is executed. A step further is the aim of designing resources discovery mechanisms that grant assurance performance and maintain quality of service. A noteworthy amount of grid research has paid attention on the allocation architecture and algorithm [6].

However, allocation mechanism alone is not sufficient to hold job execution in the grid. The resource discovery and assignment also play important role in grid system to cooperate with scheduling mechanism to realize large-scale resource sharing and guarantee the minimum execution time [7]. The study of optimal resources discovery, allocation and assignment in multiple architectures is a challenging because of involved technologies and complexity of communications happening at job scheduling process require extra flexible resource assignment and job scheduling. Even more combining dynamic resource sharing and resource discovery, allocation and assignment performance in grid system is also value of additional investigation [8].

The Grid Computing (GC) has an ossification problem such as scalability, component failures, dynamism and manageability. The GC most likely the most heterogeneous computing systems. It consists of computers with dissimilar operating systems, a variety of hardware and software, dissimilar storage capacities, CPU speeds, network connectivity and technologies [9]. Even though computational grids are attractive for their expected computational power, there is still a level of complexity to make use of such highly heterogeneous resources [10]. The GC is distressed from ossification problems. To solve the GC ossification problems lets dive into the five primary difficulties related to different resource discovery mechanism in GC [2]: (i) Scalability: the ability to be enlarged to accommodate that growth of the grid. (ii) Adaptability: the ability to cope with unexpected disturbances in the environment. (iii) Availability: the quantity of period time for example, one year that the resources are obtainable in the situation of element faults in the system. (iv) Manageability: the ability of being managed or controlled. Commonly these difficulties due to: (a) Heterogeneity characteristics of computational grids create the primary problem of resource discovery a vast challenge. (b) Dynamicity nature of different resources involved in grid networks. (c) Locality of the resources within grid networks [11].

OPEN ISSUES IN SELF-RESOURCE DISCOVERY

To facilitate and smooth the progress of resource sharing environment a basic problem is to discover the resource or resources entity that matches a user's job description. We assume a resource to be an entity that is already on hand or available in the system for example OS, CPU and memory. A well-organized resource discovery method is one of the basic necessities for GC, as it helps in resource management system and allocation or co-allocation of jobs.

Grid resource discovery refers to the process of locating satisfactory resources based on user requests. Resource discovery action engages searching for the suitable resource types with the intention of matching the user's job needs [12]. This process represents an important step based on which resource reservation and task scheduling can take place to enable Grid application development. A resource in the system is represented by a number of attributes. The values associated with these attributes may be either static or dynamic. Static resources have attributes that are relatively stable, while dynamic resources have volatile or dynamic attributes. Consequently finding a dynamic resource is much more challenging than finding static resources. For descriptive aims we would think about a resource to be a dynamic characteristic of a computing element (e.g., CPU utilization load, hard-disk free space, network bandwidth utilization and available utilization memory). Any node within the system is an element which holds a combination of these resources. As a result the goal of resource discovery is to find a node that would assure all resource requirements specified in a resource request. An attractive issue regarding to a resource discovery in grid computing is if a required resource does not exist but can be combined using a combination of other existing grid resources [13].

Due to the complexities inherent to the grid environment, it is challenging to develop efficient methods to discover dynamic grid resources. The solution should be fully decentralized, scalable with the number of users and resources, and tolerate intermittent resource participation (either voluntary or due to failure). Grid resource discovery methods must intelligently handle resource requests by searching a small subset of a large number of accessible resources, the status and availability of which dynamically change. Various kinds of solutions to grid resource discovery have been suggested, including centralized and hierarchical information server approaches. However, both of these approaches have serious limitations in regard to scalability, fault tolerance, and network congestion Joining and leaving nodes do not affect.

By harmonizing the user's application requirements, suitable resources for requested task are found in the grid computing system. This can also be defined as a system in which despite of large scale and dynamicity; suitable resources are identified for execution

of a job. In a grid, resource discovery is a complex process. This is due to meeting and sharing requirement of resource owners and users and involvement of multi attribute and dynamic queries. Resource discovery mechanisms are divided into three categories, and are explained below as shown in Figure-1 [12] [14].

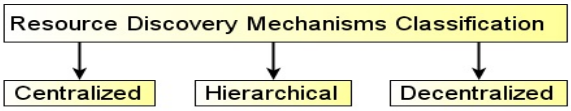

Figure-1. Resource discovery mechanisms classification.

A. CENTRALIZED MECHANISMS

Client architecture which discovers resources by using selected group of controllers in a system is known as the centralized mechanism. The information about different available services can be obtained from the server in these systems. In this mechanism request is generated by the entity for a certain service and is forwarded to the server. The server searches the required resources and then provides theses resource to the concerned entity [15]. They can search the required resources in less time as compared to other mechanisms. This mechanism face bottleneck when the number of nodes in a system goes beyond several hundred because in this architect assigned set of controllers process all queries. Now we discuss some centralized mechanisms in a Grid which are used in determining available resources.

In central mechanism, the information is updated and stored in central servers. It increases the traffic of the network which does not support dynamic attribute queries. On the other hand most of the mechanism support multi attribute queries. Due to centralized control of information, security and authorization issue is handled easily. Another disadvantage is low scalability. Due to huge traffic of nodes, system suffers the bottleneck problem. Due to central data base parallel access is limited thus high work load decrease time efficiency. These drawbacks were removed in decentralized mechanism which changes the database to avoid failure.

Client/server structural design which discovers resources by using selected group of controllers in a system is known as the centralized mechanism [16]. This mechanism face bottleneck when the number of nodes in a system goes beyond several hundred because in this architecture a set of controllers is assigned to process all queries. It increases the traffic of the network which does not support dynamic attribute queries. On the other hand, most of the mechanism support multi attribute queries. Due to centralized control of information, security and authorization issue is handled easily. Another disadvantage is low scalability. Due to huge traffic of messages, system suffers the bottleneck problem. Parallel access in central data base results in high work load this decrease time efficiency [17].

B. DECENTRALIZED MECHANISMS

In a decentralized mechanism, grid resources are allowed to join or leave at any time and at the same time disconnecting and connecting to the routers is also allowed [18]. These mechanisms support load balancing, fault tolerance, high scalability and remove Single-Point-of-Failure (SPF). These systems also do not support dynamic attribute of query, requires update of information for resource periodically and network traffic is very high [19]. In a decentralized mechanism Grid resources are allowed to join or leave at any time and at the same time disconnecting and connecting to the routers is also allowed. These mechanisms do not support balancing of load, fault tolerance, high scalability and single point of failure. These systems also does not support dynamic attribute of query. These mechanisms also required update of information for resource periodically.

C. HIERARCHICAL MECHANISMS

Like the Globus resource discovery mechanism, the information of resources are indexed, updated and sorted under a set of hierarchical nodes. The drawback of this system is that queries are not answered at the lower level and it cannot provide fast response to the queries [20]. At times, for small and medium sized grids, the hierarchical mechanisms are practical and for very large grids super peer mechanisms are more effective. Sufficient time is required to update resource information from the leaf to the root. SPF occurs in tree root system and it minimizes the system fault tolerating feature.

Hierarchical mechanisms are more scalable than centralized systems, offer some proximity search capabilities which are raised from the system's design and reduce the network's traffic. The drawback of this system is that queries are not answered at the lower level and it cannot provide fast response to the queries. At times for small and medium sized Grids the hierarchical mechanisms are practical and for very large Grids super peer mechanisms are more effective. Sufficient time is required to update resource information from the leaf nodes to the root node. Single point of failure occurs in tree root system and it decreases the system fault tolerating feature.

INITIAL PROPOSAL FOR SELF-RESOURCE DISCOVERY MECHANISM

We propose a layered algorithm that achieves dynamic and self-resource discovery mechanism (SRDM) as shown in Figure-2 and Figure-3. It is based on a Quad-Trees model. A Quad-Tree is a type of tree structure in which each node has up to four children. Quad-Trees are commonly used to divide 2D spaces into smaller areas. They are similar to octrees. The advantage of using Quad-Trees, like other trees, is that it can be quickly searched. For instance, a tree storing sixteen nodes can be searched in only two search iterations. A tree storing 64 nodes can be searched in only three iterations.

DQT consists of the following main features: (i) It is layered; (ii) it supports heterogeneity and scalability; (iii) it is independent from any physical architecture such

as grid, (iv) it achieves self-load balancing depending only on local knowledge for each node. Our vision is to design a logical mapping to the physical grid network with a new distributed Quad-Tree model. Our initial distributed Quad-Trees model can be visualized as in Figure-2.

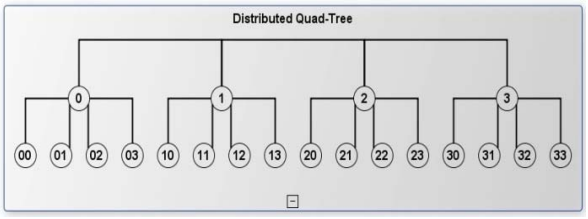

Figure-2. Distributed quad-tree model.

The algorithm makes use of spatial indexing; it starts by partitioning the geographical area of the network into equal sections. Each subsection partitioned further into smaller equal subsections. It will continue partitioning until it reaches a predefined goal or until each node resides in a single partition. Each partitioning stage will create a new level in the quad tree system. We envision developing a model that has a very minimum level, thus the discovery process will be completed with a very minimum time.

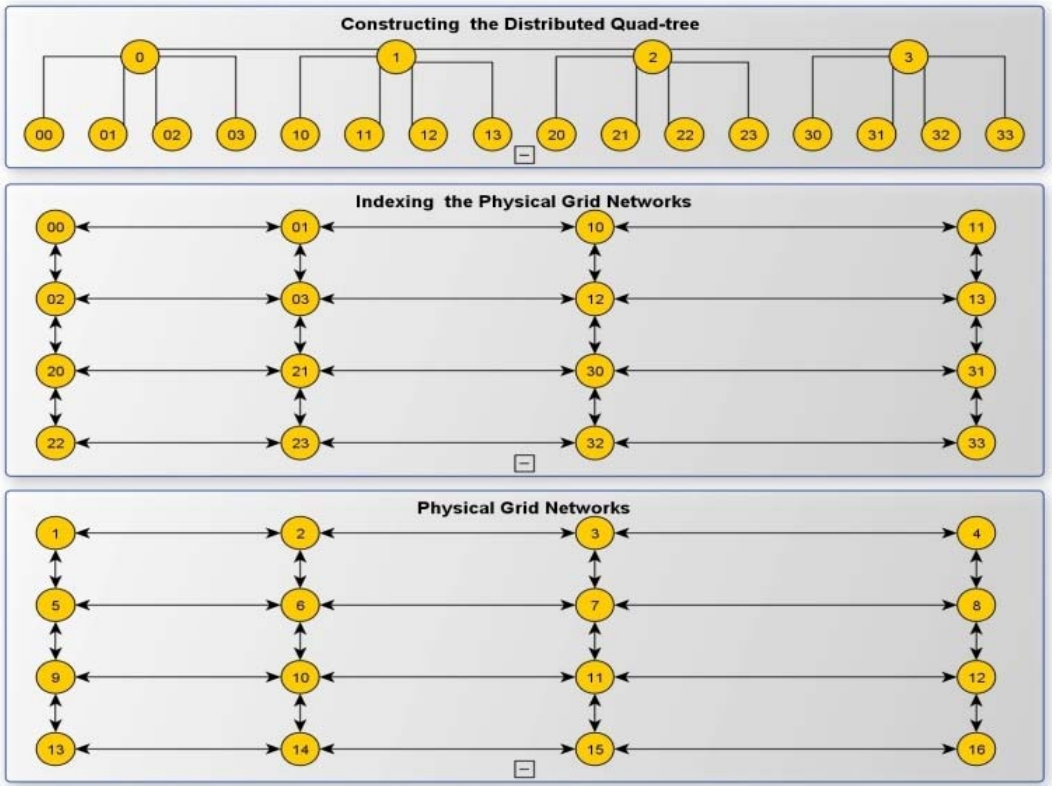

Figure-3. Conceptual model for self-resource discovery mechanism (SRDM).

Quad tree model has been widely used in area of image processing, Wireless Sensor Networks, P2P computing and implemented for collision detection. The distributed hierarchy is used for the node discovery that can locate the nodes very fast, and support the network's high dynamicity in terms of joining and leaving node as well as removing the single point of failure so there is no any bottleneck. Hence, it achieves distance-sensitive in-network querying (locality). Through the distributed hierarchy, will support efficient information storage, maintain a minimalist structure, which considered as stateless and the construction of distributed hierarchy is local and does not require any communication at all

Although this model is very interesting and has a very good potential to be applied in a huge grid system, we found that it lacks of robustness, especially in terms of single point of failure. Thus, we want to propose a new distributed quad tree in order to address this problem. We

will find ways to enable each top main node dynamically passing the grid information among each other, so that when certain top nodes fail, it can automatically elect another node to be the top node.

To enhance further the speed of discovery process, particularly in the aspect of logical and physical mapping, we will devise certain special index to each individual partition. Each partition will be given a unique ID. By using encoding trick and assigning DQT addresses for DQT nodes, we can start constructing the DQT structure.

However, these nodes IDs are logical and it has to be tied to a physical node address. To achieve this each node broadcast its logical and physical IDs to its neighbors that are 2 hops away. Once the physical and logical IDs are linked, each node can know its parent. And thus the tree is formed. Besides that, to achieve a self-resource discovery and self-load balancing, we plan for each node to computes its strength index, which we will refer as Nodepower.

The Nodepower is computed locally by each node and it is a weighted value for the computing resources of that node such as memory, CPU speed, available harddisk space, and bandwidth in addition to the bus speed. Each node then decides based on its power the number of outgoing edges for it. The outgoing edges are few for nodes that have low power and increases with the high node power. To guarantee connected resources the minimum outgoing edges is restricted to 2 while the maximum depends on the Nodepower. It is important to note that the outgoing edges to a certain node are computed for nodes that are expected to provide a service. This results in another logical layer in the network. The first layer is the physical grid networks, next the quad-tree layer and the third one is the incoming and outgoing edges for resources. Each outgoing edges to a certain node has to be linked with another node from the other end. This will results in having all node that provide the same service connected together using the incoming and outgoing edges which will facilitates the process of distributing the load between them later. The problem of selecting the other end of each outgoing edge is solved using the DQT. Each node is a member in the DQT. Therefore, it will send a message to its parent in the tree requesting a certain number of nodes of the same type to be linked to itself. The parent starts assigning from its children, if it is not enough, it will send the same message to its upper parent until we have a full assignment. After the construction of the third layer is complete, the network will be ready to achieve self-resource discovery and load balancing. Once a job enters the network, it will be analyzed to find out the type of resources it is requesting. This will decide which type of resource will be used to serve the job. Using the DQT, the job will be forwarded to nearest node of the requested type resource. Then the receiving resource will follow the incoming edges to direct the job to the least busy resource. Once the job is started processing, the available Nodepower of the selected node is decreased based on the current job needs. Therefore, the node incoming edges will be recomputed and decreased to reflect the actual available Nodepower. The decrease of the outgoing edges implies that some edges have to be deleted. The nodes that will be deleted will be selected using the quad-tree

CONCLUSIONS

Resource discovery and allocation is a big challenge because of the increased cost, heterogeneity in both (software and hardware), and complexity of current networks technologies. In this paper we reviewed the state of the art in resource discovery mechanisms. Our study shows that conventional mechanisms are not adequate to face the ever increasing challenge and complexity of technology. To this end, we elaborated our vision in designing a novel self- resource discovery mechanism for computational grid is proposed. The mechanism employs the spatial index and partitioning the space of the network to equal partitioning quadrants. We have proposed a DQT with an additional logical layer that is built based on the available resources. This layer is connecting resources of the same type thus facilitating the process of resource discovery mechanisms. Local knowledge is exploited to achieve a better response time and performance. We believe that all these plans can be actualized and contributes towards a robust, extremely fast and self-recovery resource discovery for a huge size grid

ACKNOWLEDGEMENTS

The authors wish to thank the Ministry of Education, Malaysia for funding this study under the Long Term Research Grant Scheme (LRGS/bu/2012/UUM/ Teknologi Komunikasidan Infomasi).

REFERENCES

[1] G. Strawn. 2014. Masterminds of the Arpanet. IT.66-68.

[2] F. Mimhashemi, A. Albadavi, A. Asosheh, M. G. Tajgardoon. 2011. A service-based grid computing model to apply in the universities and institutes. 2011 Sixth Int. Conf. Digit. Inf. Manag. 332-336.

[3] Q. Duan. 2010. Resource allocation in buffered crossbar switches for supporting network virtualization. 2010 Int. Conf. High Perform. Switch. Routing. 147–152.

[4] H. Casanova. 2002. Distributed computing research issues in grid computing. ACM SIGAct News.

[5] S. N. M. Shah, A. K. Bin Mahmood, A. Oxley. 2010. Hybrid Resource Allocation Method for Grid Computing. Second International Conference on Computer Research and Development. 426-431.

[6] L. Tom, B. Caminero, C. Carri. 2013. Opportunistic Energy-Aware Rescheduling in Desktop Grid Environments. 178-185.

[7] D. Abramson. 2011. Mixing cloud and grid resources for many task computing. Proc. 2011 ACM Int. Work. Many task Comput. Grids Supercomput. 1.

[8] M. Z. Ali, S. Chakrabarty, M. Akther, and M. A. N. Bikas, 2012. A single sign on mechanism for multiple grid manager on Alchemi .NET based grid framework. 15th Int. Conf. Comput. Inf. Technol. 436-440.

[9] R. Rajan, G. K. Kamalam, 2013. Priority based heuristic job scheduling algorithm for the computational grid. 2013 Int. Conf. Inf. Commun. Embed. Syst. 448-451.

[10] G. Le, K. Xu, J. Song. 2013. Dynamic Resource Provisioning and Scheduling with Deadline Constraint in Elastic Cloud. Int. Conf. Serv. Sci. 113-117.

[11] V. K. Soni, R. Sharma, M. K. Mishra. 2010. An analysis of various job scheduling strategies in grid computing. 2nd Int. Conf. Signal Process. Syst. 2: 162-166.

[12] A.Belbekkouche, M. M. Hasan, A. Karmouch. 2012. Resource Discovery and Allocation in Network Virtualization. IEEE Commun. Surv. Tutorials. 14(4): 1114-1128.

[13] S. Valente, A. Grimshaw. 2011. Replicated Grid Resources. 2011 IEEE/ACM 12th Int. Conf. Grid Comput. 198–206.

[14] S. M. Fattahi, N. M. Charkari 2009. Distributed resource discovery in grid with efficient range query. 2009 14th Int. CSI Comput. Conf. 335-340.

[15] M. M. Klaus-Krauter, R. Buyya, K. Krauter, R. Buyya, M. Maheswaran. 2000. A Taxonomy and Survey of Grid Resource Management Systems," Softw. Pract. Exp., 32(2): 1-29.

[16] S. Sotiriadis, N. Bessis, Y. Huang, P. Sant, C. Maple. 2010. Towards decentralized grid agent models for continuous resource discovery of interoperable grid Virtual Organisations. Fifth Int. Conf. Digit. Inf. Manag. 530-535.

[17] R. Ranjan, A. Harwood, R. Buyya. 2008. Peer-to-peer-based resource discovery in global grids. Surv. Tutorials, IEEE. 6-33.

[18] M. Chowdhury, F. Samuel, R. Boutaba, 2010. PolyViNE: policy-based virtual network embedding across multiple domains," Proc. Second ACM. 49-56.

[19] P. Vashisht, A. Sharma. 2010. Decentralized P2P Grid Resources Discovery model in LC-Trie structured overlay. 2010 First Int. Conf. Parallel, Distrib. Grid Comput. (PDGC 2010). pp. 330-333.

[20] L. Khanli, A. Niari, S. Kargar, 2011. An efficient grid resource discovery mechanism based on tree structure. Comput. Sci. 48-53.