

Application of ACO for Resource Discovery in Grid Computing Environment

S.Nirmala Devi
Assistant Professor
Department of Computer Science
Government Arts College for Women
Salem, TamilNadu, India

A.Pethalakshmi
Associate Professor and HOD
Department of Computer Science
MVM Government Arts College for Women
Dindigul, TamilNadu, India

ABSTRACT

Efficient utilization of resources is the key factor for any environment. The grid environment is also more dynamic, it facilitates the movement of resources in and out of the environment flexibly. Hence maintenance of centralized registries is not sufficient for this demanding environment. Here, we propose a new approach for resource discovery which employs the Ant Colony Optimization (ACO) technique for resource discovery it also includes the routing information and enhances the resource discovery process.

Keywords

Grid Computing, Resource Discovery, Swarm Intelligence, Ant Colony Optimization, Routing Table.

1. INTRODUCTION

Grid Computing can be defined as the collection of heterogeneous resources agreed to form virtual organizations (VO's) and they cooperate, share and work to achieve the heavy computational task. Grid environment considers any kind of resources as computing resources, data resources, storage resources, service resources, network resources and web resources. All these resources are heterogeneous and dynamic in nature and efficient utilization of resources is required for the problem solving in grid environment.

There were various projects running in various parts of the world using the grid computing techniques. Here we will survey the most important organizations which provides the grid computing middleware and their resource discovery approaches[16] and propose a new approach using ACO to improve the performance of the resource discovery.

2. GRID RESOURCE MANAGEMENT IN GRID TOOLS

There were various tools available for deploying the grid environment and these tools makes use of the grid resource manager for resource discovery, management etc., This section explores the grid resource discovery process in these tools.

2.1 Globus Tool Kit

The Globus ToolKit makes use of the GRAM (Grid Resource Allocation and Management) for managing the resources. The GRAM4 consists of two services as pre- WS GRAM and WS GRAM[17]. This provides the XML based version for Resource Specification Model. The submitted job used to provide the requirements to the Local Resource Manager

(LRM). If the resources are available at the LRM itself, then the requests will be processed locally otherwise the requests would be sent to the GRAM.

The resources in the Globus are registered through the WSRF (Web Service Resource Framework) and WS Notification. The GRAM also provides two aggregator services that collect the recent state information from the registered information sources. The two aggregators implement a registry (Index) and event – driven data filter (Trigger) [18]. This collected information is stored to retrieve the necessary resources.

2.2 Legion

In Legion every element is represented as object and it follows the hierarchical architectural model for the resource management [19].

Legion objects are independent of one another, disjoint in address-space and communicate with one another through RMI (Remote Method Invocation) [20]. The Resources in Legion are represented as objects. There were various legion objects such as host object, vault object, implementation objects and proxy object. The Users can also create their own objects. The Host object is the machine representative to Legion. It is responsible for executing jobs on the machine. Vault objects are responsible for managing other legion objects. The implementation objects encapsulate the executables. The proxy objects are used to execute legacy application binaries on host and vault pairs. They do not play a role in scheduling. Here the resources in other locations are called through the RMI by making use of these objects.

The Legion makes use of distributed query resource discovery. It also employs the periodic pull resource information and the information is disseminated to other objects [21].

2.3 Condor G

The nodes connected in Condor are called the Condor Pool. The job is submitted to the centralized scheduler for execution. The resources are identified in condor and the requests are specified in the classified Ads resource specification language[22]. The central manager holds the resource information. The central manager queries the data store for information resource availability. Resource dissemination is done through periodic push mechanism.

2.4 Nimrod – G

Nimrod-G is a grid enabled resource management system, based on computational model[22]. Nimrod-G broker tries to find the cheapest resources available to solve the problem.

3. RESOURCE DISCOVERY TECHNIQUES

The resources are scattered in this large scale grid environment and also these resources can enter or leave the environment at any time. Thus this dynamic nature necessitates the discovery of this resource. Only efficient utilization of these resources provides the better solution to the grid users and there were various approaches available for discovering these resources which are discussed below.

3.1 Centralized approach

In centralized approach, the participating nodes would register their resources with the registry. The registries here are maintained using UDDI (Universal Description Discovery and Integration) or LDAP (Lightweight Directory Access Protocol) since the grid environment is built on the web services. Normally the disadvantage in this system is the failure of centralized node or link, may lead to the severe consequences and it also sometimes necessitates the maintenance of replication. And this would increase the overhead and it is also not possible. This even increases the communication cost.

In [10], the GMD Web based grid service publication directory system is maintained for resource discovery. In this method, there were two key components as GMD Portal manager and GMD query web service. The GMD portal manager is a centralized one and every entry is made in this which employs the tomcat server.

3.2 Decentralized approach

The decentralized approach can also be used to maintain the details of the resources. In [2], the dynamic hash tables are used to maintain the details about the resources at multiple sites. But this requires frequent data updating. This follows the query approaches as RLQ and RUQ. RLQ is the query issued to the broker service to locate the resources and RUQ is the update query sent by the participating site about its available resources. In [11], the system is divided into groups as inter groups, intra groups etc and also the penalty functions are implemented by applying the distributed approach.

3.3 Ontology based approach

Ontology means representing related information as a set of concepts. Then retrieving information based on match making process.

In [3], the match making algorithm is used to retrieve the information about the resources. The main actions involved are advertising, matching, notification and claiming for the resources. Here the matchmaker protocol is composed of the publishing protocol and notification protocol.

3.4 Peer-to-peer approach

Peer to peer refers to the approach where every participating node have equal preference and it can act both as server and client at a time. This modeling could also be used in grid as there is no client server approach available in the grid and each participating node can serve and request the process.

The peer to peer approach also makes use of DHT with indexing features. In [4], the P2P uses the network organization, Data organization and d- dimensional query routing.

In [15], the comparative study is made between the structured P2P systems and unstructured P2P systems for resource discovery. And various parameters such as scalability, dynamicity, reliability, range queries, multiattribute queries and dynamic attribute queries were considered.

3.5 Routing based approach

The routing approach makes use of message routing concepts [5]. In [13] the link state routing protocol OSPF (open shortest path first) was improved for resource discovery and location. The routing table data is considered and the shortest path to the resource is identified and made use. This facilitates the higher query efficiency and less communication traffic

3.6 Agent based approach

Current researches are focusing on agent based approaches to discover the resources. The agents collect resources that match the resources required and inform to the source and even it leaves the information for other agents [6] [7] [8]. In [15], the agent based approach is evaluated on various parameters as scalability, dynamicity, reliability, range queries. And it has been arrived that the agent based system is suitable for small or medium scale environment rather than on large scale environments. The dynamicity is tolerable but it is not reliable and all the query approaches are supported by this agent approach.

3.7 Hybrid Approaches

Certain researches are on hybrid approaches, which combine the approaches to reap the benefits of both the approaches. In [14] the P2P approach and ontology approach is combined for resource discovery. And the authors have also notified that the energy utilization is reduced to 10% from 40%.

4. PROPOSED APPROACH

Resource discovery is the key issue in the grid computing environment. The submitted job should be processed in an optimized way i.e., by considering various factors as time utilization, bandwidth utilization etc. The proposed approach makes use of the ACO (Ant Colony Optimization) technique to discover the resources in the grid environment by following the sequence of steps below.

4.1 Job Submission & Resource Identification.

The user submits the job at the participating node in the virtual organization. The heavy computational task may require more number of resources. The job will be submitted to LRM (Local Resource Manager). The LRM identifies the required resources and if the resources are available in the local host and if it is computationally feasible to handle with the local resources then the request will be processed locally.

4.2 Application of Ant Colony Optimization

If the job could not be processed locally then the resources from the grid need to be identified. For this the ant colony optimization technique is used. ACO is a class of algorithms, whose first member called ant system, was initially proposed by Colomi, Dorigo and Maniezzo[23]. It is inspired by the behavior of ants. The main idea behind employing the ACO is facilitating parallel search over several computational threads based on the given infrastructure.

4.3 Generation of Ants

The ants are generated at the machine where the job is submitted. The ACO technique normally employs two mechanisms as trail evaporation and daemon actions. Here the trail evaporation is performed as the ant reaches the TTL (Time To Live) reaches the zero value. It means that the ant has travelled enough distance and it need not have to wait more time. Hence in order to make the network free and information seeking easier the pheromone trails are made to evaporate. Similarly the ants while moving have to update the details regarding the path to the hops passing through.

4.4 Resource Allocation

Based on the details collected from the ants in the network, the resource can be allocated for the job to complete. The ants would send the information to the source machine. The information given is used to select the resource.

5. SIMULATION IN GRIDSIM

The GridSim toolkit supports modeling and simulation of a wide range of heterogeneous resources, such as single or multiprocessors, shared and distributed memory machines such as PCs, workstations, SMP's and clusters with different capabilities and configurations. In grid the resources are geographically distributed across multiple administrative domains with their own management policies and goals. Also the resource brokers in grid focus on enhancing performance of a specific application [24]. This GridSim toolkit provides facilities for the modeling and simulation of resources and network connectivity with different capabilities, configurations and domains is selected for simulating our approach.

5.1 The Design of GUI

GridSim supports the CUI, and normally it is always desirable to have the GUI to be used for giving the inputs. Since GridSim is developed on the basis of Java. We have developed the GUI using Java Swing. Various inputs can be given using this GUI and it is specified in the fig 1.

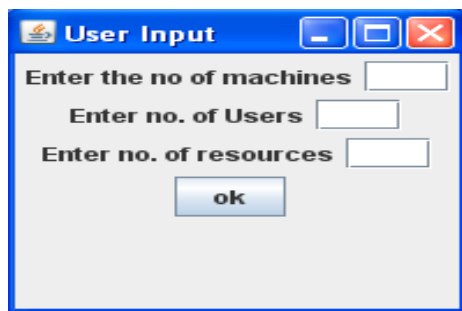


Fig 1: Input Screen Design.

5.2 The machine list class

The machine list class is used to create the LAN or WAN, depending upon the request from the user. In the list, a machine can be added by specifying the machine id, the number of processing elements (PE's) and also the MIPS rating. The machine can be identified by its id. Other methods are also available to know about the status of the PE's.

5.3 The Gridlet Class

Through the Gridlet class the job to be executed can be given. Here the text file is given as input, which takes in the details of the job to be done. The job details include the gridlet id, the length of the gridlet, the filesize and also the gridoutputsize which are given as input through the constructor.

5.4 The Resource Characteristics Class

The resource characteristics class is used to define the properties of the resource. Through this class the information such as the system type, operating system, the machine list it is associated to are maintained. The resources may be time-shared or space shared, this can also be defined. Here the resource sharing can be done through time sharing.

5.5 Generation of Ants

The ants are generated, by the machine object. These agents would access to the machine that are created using the machine class. These agents would fetch the information about the resources what have been specified through resource characteristics class. The ants designed include various factors TTL (Time to Live). This is the evaporation operation of the ants.

5.6 Selection of Resources

The local machine object would decide the resources from the details submitted by the ant. The selection is based the factors as the bandwidth utilization and resource location. The jobs can be processed using the resources developed.

6. EXPERIMENTAL ANALYSIS

The experiment is generated with the machines, users and resources that were dynamically given by the user. The number of agents generated depends upon the machines that were available at that time in the grid, whereas the other existing approach does not support scalability. Even the number of participating nodes increases, the ants increases proportionally and provides the better result in our experiment which is shown in the following fig 2.

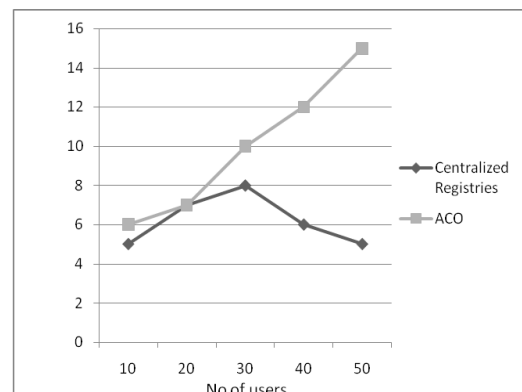


Fig 2: The no of hits increases in ACO

When the number of users or hops increases, it simultaneously increases the performance. The performance degradation is found only in centralized registries due to the bottleneck in communication. The grid also functions efficiently.

The algorithm also handles the performance of the system, as the number of queries increases the performance also increases when employing the ACO since ants are based on the trained data. And this is shown in the below fig 3.

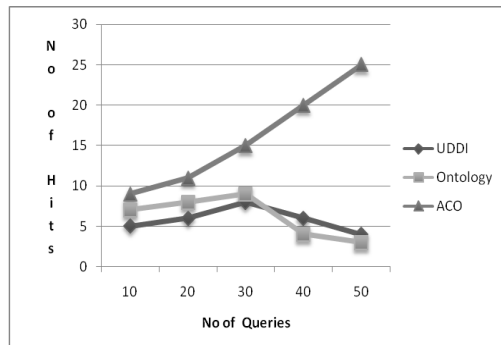


Fig 3: Comparison based on no of queries.

The ACO algorithm which is based on the colony behavior increases the performance in resource discovery in the dynamically changing grid environment.

7. CONCLUSION

Thus the grid computing environment which is more promising area in this internet era, need to be upgraded in all front to reap the benefits. The resource utilization is one of the key issues in grid computing. And this paper proposes the approach to efficiently discover the grid resources using ACO. It is also simulated using the GridSim simulator tool and have been proven that those provides the efficient way of retrieving the details. The algorithm can also be improved by retrieving the routing information.

8. REFERENCES

- [1] Deniz Cokuslu, Abdelkader Hameurlain, KAYhan Erciyes, “Grid Resource discovery based on centralized and hierarchical architecture”, International journal for Infonomics. Volume 3, Issue 1, March 2010.
- [2] Rajiv Ranjan, Lipo Chan, Aaron Harwood, Shanika Karunasekera, Rajkumar Buyya, “Decentralized Resource Discovery service for Large scale federated database”.
- [3] Er.K.S.Dhindsa, Er.Naveen Dogra, Er. Mohit Kumar, “Resource Discovery in grid environment using ontology based approach”, proceedings of 2nd national conference on challenges and opportunities in information technology, RMIT – IET March29th 2008 pp. 100 to 103
- [4] Rajiv Ranjan, Aaron Harwood, Rajkumar Buyya, “A study on peer-to-peer based discovery of grid resource information”
- [5] Roddrigo Fernandes demello, Luciano Jos ‘e Senger, Laurence Tianro Yang, “ A Routing Load Balancing Policy for Grid Computing Environments”, Proceedings of 20th international Conference on Advances information networking and Applications(AINA’06)
- [6] Agostino Forestiero, Carlo Mastroianni and Giandomenico Spezzano, “ An Agent based Semi-informed protocol for resource discovery in grids”, ICCS 2006, part IV, LNCS 3994, pp.1047 – 1054, 2006
- [7] Bhupinder Singh and Seema Bawa, “ Grid Resource Discovery using Natural Mobile Agents”, International journal of computer science and network security”, Vol. 8. No. 11. November 2008
- [8] T.Srinivasan, J.B. Siddharth Jonathan, Jayesh Seshadri and Arvind Chandrasekhar, “A Swarm Intelligence based Task Allocation Algorithm (SITA) for the computational Grid”. Joshy Joseph, Craig Fellenstein, “Grid Computing”, IBM Press.
- [9] Jia Yu, Srikumar Venugopal and Rajkumar Buyya, “Grid Market Directory: A Web Services based Grid Service Publication Directory” [online] <http://arxiv.org/ftp/cs/papers/0302/0302006.pdf>
- [10] David Oppenheimer, Jeannie Albrecht, David Patterson and Amin Vahdat, “Design and implementation tradeoffs for wide – area resource discovery”, [online] <http://sword.cs.williams.edu/pubs/hpdc05.pdf>
- [11] Agostino Forestiero, Carlo MASTROIANNI and Giandomenico Spezzano, “An Agent Based Semiformed protocol for resource discovery grids”, ICCS 2006, LNCS 3994 pp.1047 – 1054, 2006.
- [12] Ying Liu, Shengrong Wang, “Grid Resource Discovery Mechanism based on OSPF routing protocol”. Control and Design Conference, 2008
- [13] Zenggang Xiong, Xuemin Zhang and Li Liu, “Resource Management Model and Resource discovery algorithm for P2P Grid”, Journal of Networks. Vol 6, No.8, August 2011, pp. 1187 – 1194
- [14] Abdelkader Hamemeurtain, Deniz Cokuslu, “ Resource Discovery in Grid Systems: A Survey” Int. j. Metadata, Semantics and ontologies, Vol 5, No.3, 2010.
- [15] C.S.R. Prabhu “Grid and Cluster Computing”, PHI Learning Pvt. Ltd, 2010 pp.31 – 32.
- [16] Martin Feller, Ian Foster and Stuart Martin, “ GT4 GRAM: A Functionality and Performance Study”, TeraGrid 2007 Conference, Madison.
- [17] Ian Foster, “Globus Toolkit Version 4: Software for Service – Orinted Architecture”. International Conference on Network and Parallel Computing, 2005, pp2-13.
- [18] Rajkumar Buyya, Steve Chapin and David DiNucci, “Architectural Models for Resource Management in the Grid”[online]
- [19] Michael J.Lewis, Adam J.Ferrari, Marty A. Humphrey, John F.Karpovich, Mark M.Morgan, Anand NATRAJAN, Anh Nguyen – Tuong, Glenn S.Wasson and Andrew S.Grimshaw, “Support for Extensibility and Site Autonomy in the Legion Grid System Object Model”.
- [20] Anand Natrajen, Marty A.Humphrey and Andrew S.Grimshaw, “Grid Resource Management in Legion”.
- [21] http://www.cs.virginia.edu/~grimshaw/publications/GSR_M03.pdf
- [22] Chaithanya Kandagatla, “Survey and Taxonomy of Grid Resource Management Systems”. [online]
- [23] <http://www.cs.utexas.edu/~browne/cs395f2003/projects/KandagatlaReport.pdf> Vittorio Maniezzo, Luca Maria Gambardella and Fabio de Luigi, “Ant Colony Optimization”. [online]
- [24] RajKumar Buyya and Manzur Murshed, “GriSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing”, Concurrency and Computation: Paractice and Experience, Concurrency Computation: pract. Exper. 2002 pp/1175 – 1220