

# Resource Discovery In Grid Environment Using Ontology Based Approach

**Er. K.S. Dhindsa\***, **Er. Naveen Dogra\*\***, **Er. Mohit Kumar\*\*\***  
\*BBSBEC Fatehgarh Sahib Punjab, \*\*/\*\* RIMT-IET Mandigobindgarh

*Abstract -In today's complex world of high speed computing, computers have become extremely powerful and even home-based workstations are powerful enough for running complex applications. There is a need for numerous complex scientific experiments, advanced modeling scenarios, genome matching, astronomical research, a wide variety of simulations, complex scientific/ business modeling scenarios, and real-time personal portfolio management, which require a huge amount of computational resources. So to satisfy some of these computational requirements, Grid computing is born. Resource discovery is a very important feature in the grid computing environment. The matching framework provides a reasonable solution to resource discovery. The matchmaking framework may be decomposed into four components i.e. classad specification, matchmaking algorithm, matchmaking protocol and claiming protocols. In this paper we proposed a new algorithm for Resource Discovery using Ontology based Approach.*

## 1. INTRODUCTION

Grid computing simply stated, is distributed computing taken to the next evolutionary level. The goal is to create the illusion of a simple yet large and powerful self-managing virtual computer out of a large collection of connected heterogeneous systems sharing various resources. Grid computing spans multiple organizations, machine architectures and software boundaries to provide unlimited power, collaboration and information access to everyone connected to a grid There are lots of approaches available for resource discovery like Agent Based Approach, Query Based Approach, Parameter Based Approach, Quality of Service Based Approach, Ontology Based Approach, Peer-to-Peer Approach , De-Centralized Approach and Routing Transferring Model-Based Approach and Request Forwarding Approach. Ontology Based Approach is used in the present work. The main idea behind this approach is the advertisement of the resources. In this approach, service provider registers its service description into the service registry database . When a Grid application sends a request to service directory, matchmaker returns the matches to the service requester. Requester chooses the best resource based on the specific need.

## II. GRID TAXONOMY

Depending upon the design objectives and target applications for a particular Grid environment grid systems can

be classified into three categories as shown in figure . The design objective of a grid system can be any one or a combination of two or more of the following.

- a. improving application performance
- b. improving data access
- c. enhanced services

### A. Computational Grid

The computational Grid category denotes systems that have a higher aggregate computational capacity available for single applications than the capacity of any constituent machine in the system. These can be further subdivided into distributed supercomputing and high throughput categories depending on how the aggregate capacity is utilized. A *Distributed Supercomputing Grid* executes the application in parallel on multiple machines to reduce the completion time of a job. Typically, applications that require a distributed supercomputer are grand challenge problems. Examples of grand challenge problems are fluid dynamics, weather modeling, nuclear simulation, molecular modeling, and complex financial analysis. A *High Throughput Grid* increases the completion rate of a stream of jobs. Applications that involve parameter study to explore a range of possible design scenarios such as ASIC or processor design verification tests would be run on a high throughput Grid.

### B. Data Grid

The Data Grid category is for systems that provide an infrastructure for synthesizing new information from data repositories such as digital libraries or data warehouses that are distributed in a wide area network. Computational Grids also need to provide data services but the major difference between a Data Grid and a computational Grid is the specialized infrastructure provided to applications for storage management and data access.. The two popular DataGrid initiatives, European DataGrid and Globus are working on developing large-scale data organization, catalog, management, and access technologies.

### C. Service Grid

The Service Grid category is for systems that provide services that are not provided by any single machine. This category is further subdivided in on demand, collaborative,

and multimedia Grid systems. A *Collaborative Grid* connects users and applications into collaborative workgroups. These systems enable real time interaction between humans and applications via a virtual workspace. A *Multimedia Grid* provides an infrastructure for real-time multimedia applications. This requires supporting quality of service across multiple different machines whereas a multimedia application on a single dedicated machine can be deployed without QoS. An *On demand Grid* facilitates access to rarely used resources for short periods of time. While many organizations and individuals would benefit from resources such as a supercomputer or specialized instrumentation equipment, their infrequent usage pattern would fail to justify the procurement of such a resource. Grid services can allow seamless use of these resources, regardless of the given domain.

### III.MATCHMAKING FRAMEWORKS

The main actions involved in the matchmaking process are advertising, matching, notification and claiming. The above interactions between the matchmaker and other principals participating in the matchmaking environment motivate the definition of the following components of a matchmaking service.

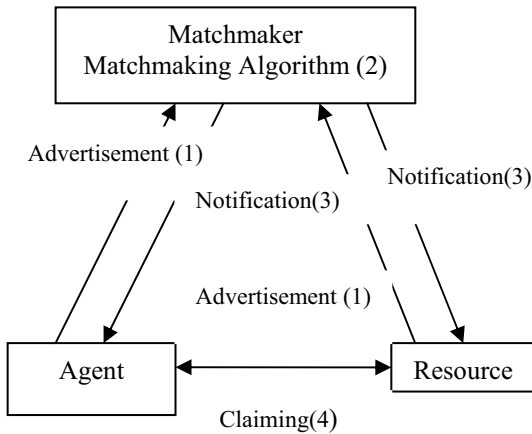


Figure 3.1 Matchmaking Processes

A language for specifying the characteristics, constraints and preferences of principals. Our framework uses the classified advertisement (classad) language for this purpose. Figure shows a classad representing a workstation. The Constraint attribute indicates that the workstation only runs jobs when it is idle (as determined by load average and keyboard idle time) and if the job’s image size is 15 megabytes less than the physical memory size of the machine, with a preference for jobs submitted by user “administrator” The classad language is a symmetric description language; both servers and customers use the same language to describe their respective characteristics, constraints and preferences. Among other constructs that allow entities to be easily represented, the language supports semi-structured “records”

that are used as the descriptions themselves, and dynamic typing with special undefined and error values that enable robust evaluation semantics in heterogeneous environments.

```

[
  Type      = "Machine";
  Activity  = "Idle";
  Disk      = 323.4 M;
  Memory    = 64 M;
  State     = "Unclaimed";
  LoadAvg   = "0.0426";
  Arch      = "Intel50";
  OpSys     = "Win 2000";
  Name      = "rimt1.ac.in";
  Constraint = other.Type == "Job"
              && LoadAvg < 0.3
              && Memory=
              other.ImageSize >= 15 M
]
    
```

Figure 3.2 ClassAd describing a workstation

The Matchmaker Protocol is composed of the publishing protocol and notification protocol that respectively describe how agents communicate with the matchmaker to post advertisements and receive notifications. The Matchmaking Algorithm is used by the matchmaker to create matches. In the abstract, the matchmaking algorithm relates the contents of submitted classads and the state of the system to the matches that will be created. As part of this process, the algorithm defines a set of conventions (an advertising protocol), which binds meanings to certain classad attributes that will be used for special purposes. For example, a matchmaker may define that in any classad, the attributes named Constraint and Rank will be respectively treated as the constraints and preferences defined by the advertising entity. The Claiming Protocol is activated between the matched parties to confirm the match and establish a working relationship. In our resource management framework, we require that the locus of control for claiming reside in the matched agents themselves. An important motivation for this requirement is that, in our framework, a match between A and B is not the same as allocating A to B. Instead, the match is permission for A and B to cooperate — it is the agents’ responsibility to verify the match and decide if cooperation is still desirable. Either entity may choose to not go further and reject the match altogether

#### IV. CLASSAD MATCHING ALGORITHM CRITERIA

In the ClassAd Matching Algorithm the matching criteria are:

- a. Architecture
- b. Operating system
- c. Disk space accessible from the execution machine
- d. Swap space of the execution machine

Normally a job is compiled and linked for only one type of architecture (1) and operating system (2). Therefore, an execution machine with the same architecture and operating system has to be found by the CM. A sufficient amount of free disk space accessible from the execution machine (3) is needed to store the job's executable before it is started. Finally, after restoring the job from the checkpoint file, the job's image must fit in the swap space of the execution machine (4). These criteria only define whether the job can run at all on a particular machine. Whether the job will run efficiently is another question. Keeping this point in view, along with above criteria we can add one more criteria, largest available run time.

#### V. NEW ALGORITHM FOR MATCHING

The matching criteria, which will be used to find a suitable match for a given job now becomes

- a. Architecture
- b. Operating system
- c. Disk space accessible from the execution machine
- d. Swap space of the execution machine
- e. Largest available run time

From the ClassAds used by resources in my Condor pool, we have identified that only first four criteria are not sufficient for resource discovery in Condor pool. A simple algorithm would just choose a machine, which satisfies all criteria, and rejects a job if no appropriate match is available. But starving should not be there i.e. it might occur that while a job is waiting for an execution machine while a less suitable machine is available. In order to avoid this situation, if at least one execution machine satisfies criteria #1 to #4, the job will be scheduled anyway. The new matching algorithm for finding an execution machine for a given job is listed below:

```
P := (list of machines which satisfy criteria #1 to #4)
IF (P = ∅) THEN
  reject job
ELSE
  Q := (list of machines in list P which satisfy criteria #5)
  IF (Q = ∅) THEN
    choose a machine from list P at random
  ELSE
    choose the machine from list P
```

```
ENDIF
ENDIF
```

If all the five criteria are considered while discovering resources via matchmaking an efficient resource discovery will be done. This conclusion is based on the fact that if a criterion #5 is considered there will be lesser chances of checkpointing and scheduling job to some available resources. So our precious time is saved which is used for saving the states of the jobs and resume at the same point in the some other computer. By considering the fifth criteria we can get the best resource in the resources pool.

#### VI. COMPARISONS

*Addition of New Parameter:* In the classad matching algorithm the matching criteria are: Architecture, Operating System, Disk space accessible from the execution machine and Swap Space of the execution machine.. These criteria only define whether the job can run at all on a particular machine. Whereas in the new algorithm the matching criteria to find the suitable match for a given job now becomes Architecture, Operating System, Disk space accessible from the execution machine, Swap Space of the execution machine and Largest available run time. It helps in finding the perfect match of machine for the job.

*Reduction in Checkpoints:* The criteria of classad matching algorithm only define whether the job can run on a particular machine or not .But it doesn't tells whether the job will run efficiently. The matching process should be efficient which means that it should not burden the requester with the excessive delays that would prevent its effectiveness. The new criteria added in the algorithm find the suitable match of machine and job that reduces the number of checkpoints and increase the effectiveness.

*Faster Execution:* In the new algorithm there is less number of checkpoints. The checkpointing of a program means the saving of the state of the program so that its execution can be restarted. Since there are less number of checkpoints in the new algorithm it results in faster execution

*Cost Reduction:* As there is less number of the checkpoints using new algorithm, the speed of processing of the job is more in the new algorithm as compared to the classad matching algorithm which results in cost reduction.

*Manageability:* The new algorithm is quite easy to manage because by using this algorithm we get perfect match for the machine and most of working is depend on that particular node.

## CONCLUSION

This paper emphasizes the Resource Discovery Problem in the grid environment. The goal is to create the illusion of a simple yet large and powerful self-managing virtual computer out of a large collection of connected heterogeneous systems sharing various resources. Grid computing spans multiple organizations, machine architectures and software boundaries to provide unlimited power, collaboration and information access to everyone connected to a grid. In other words, Grid is a network of heterogeneous resources working in collaboration to solve problems that cannot be addressed by the resources of any one organization. This sharing is necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. A set of individuals and/or institutions defined by such sharing rules form what we call a Virtual Organization

Matchmaking is done on the base of criterion like Architecture, Operating system, Disk space accessible from the execution machine and Swap space of the execution machine. But these criterions are not sufficient. As the computation needs are growing, there are other factors also which must be considered. In this paper, we have proposed a New Approach for matchmaking which makes use of a criterions i.e. largest available run time along with the above specified four criterions. Addition of this criteria leads to fewer checkpoints and results in cost reduction

## FUTURE SCOPE

Conventional resource discovery systems are very efficient at managing static and dedicated resources for high-performance computing, but cannot handle the complexity and dynamism of distributively owned high-throughput computing environments. The geographically diverse institutions show lot of interest for discovery of resources in the Grid Environment. Condor is software that provides high throughput computing and can be used for discovering resources in Grid Environment. Large jobs have high checkpointing and restarting costs. Running the large jobs on execution machines which are available for the long periods of time results the fewer checkpoints and hence the cost reduction. Resource discovery through matchmaking is a relatively new area, and lot of work can be done in this field. In this paper we have added one criteria i.e. largest available run time which reduces the number of checkpoints and hence results in cost reduction. In future the new parameter can be found in the same field which further reduces the cost and increases the speed of processing.

## REFERENCES

- Foster, I. and Kesselman, C., "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann Publishers, San Francisco, CA, pp. 1-677, 1999
- Foster, I. "The Grid: A New Infrastructure for 21st Century Science", Physics Today, Vol. 55, pp. 42-47, February 2002
- Foster, I., "What is the Grid? A Three Point Checklist", GRIDToday, Vol. 1, pp. 1-4, July, 2002
- Kl Krauter, Rajkumar Buyya and Muthucumaru Maheswaran, "A taxonomy and survey of Grid resource management systems for distributed computing", Copyright John Wiley & Sons, Ltd. Pp. 135-164, 17 September 2001
- Marvin Solomon "The ClassAd Language Reference Manual", pp. 1-25, May, 2004
- S. Ludwig, P. Santen, "A Grid Service Discovery Matchmaker based on Ontology Description," Euroweb — The Web and the GRID: from e-science to e-business, pp. 17-18, Dec. 2002