Semantic-based Grid Resource Discovery and its Integration with the Grid Service Broker

Thamarai Selvi Somasundaram[1], R.A.Balachandar[1], Vijayakumar Kandasamy[1], Rajkumar Buyya[2], Rajagopalan Raman[3], N.Mohanram[4] and S.Varun[1]

[1]Department of Information Technology
Anna University, MIT Campus
Chromepet, Chennai – 600044, India
Email : stselvi@annauniv.edu

[2]Grid Computing and Distributed Systems Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne, VIC 3010, Australia
Email : raj@csse.unimelb.edu.au

[3]Centre for Development of Advanced Computing
Block-11, 6/13, Park Avenue,
Keshava Perumal Puram, Chennai - 600 028, India
Email :mrr@cdacindia.com

[4]Centre for Development of Advanced Computing
No-1, Old Madras Road,
Byappanahalli, Bangalore - 600 028, India
Email :mohanram@cdacb.ernet.in

Abstract: *This paper addresses the need of semantic component in the grid environment to discover and describe the grid resources semantically. We propose semantic grid architecture by introducing a knowledge layer at the top of Gridbus broker architecture and thereby enabling broker to discover resources semantically. The semantic component in the knowledge layer enables semantic description of grid resources with the help of ontology template. The Ontology template has been created using Protégé-OWL editor for different types of computing resources in the grid environment. The Globus Toolkit's MDS is used to gather grid resource information and Protégé-OWL libraries are used to dynamically create knowledge base of grid resources. Algernon inference engine is used for interacting with the knowledge base to discover suitable resources.*

1. Introduction

Grid technologies enable the sharing, exchange, discovery, selection and aggregation of geographically or Internet-wide distributed heterogeneous resources – such as sensors, computers, databases, visualization devices and scientific instruments [1, 23]. The grid computing infrastructure defined in [2] is only part of much larger picture that also includes information handling and support for knowledge processing within the distributed scientific process. This broader view is adopted for semantic grid which can be described as an extension of the current Grid where information and services are given well defined meaning, better enabling computers and people to work in cooperation [14, 15]. The semantic Grid is a service oriented architecture in which entities provide services to one another under various forms of contract. In such an environment the ability to describe the Grid resources needed by applications is essential for developing seamless access to resources on the Grid [26]. The Globus Toolkit's Monitoring and Discovery System (MDS) defines and implements mechanisms for service discovery and monitoring in distributed environments. However, MDS supports traditional service matching which is based on symmetric, attribute based matching [9] and does not support semantic descriptions of Grid services or resources[6, 7]. In grid environment where resources are generally owned by different people, communities or organizations with varied administration policies and capabilities, obtaining and managing these resources is not a simple task. Resource brokers simplify this process by providing an abstraction layer for users to access heterogeneous resources transparently [20]. Gridbus broker is a resource broker designed to support scheduling of both computational and data grid applications [10]. However, the resource discovery module implemented in the Gridbus broker does not support semantic description and discovery of grid resources, as it uses the Globus Grid Index Information Services (GIIS) [23] or Grid Marker Directory (GMD) [24] to gather grid resource information.

In this paper, we propose a knowledge layer on the top of Gridbus broker architecture for semantic description and discovery of resources. The ontology based knowledge base has been created using Protégé-OWL libraries to describe grid resources semantically. Algernon inference engine is used to interact with the knowledge base for semantic

retrieval of grid resource information [16]. The rest of the paper is organized as follows: Section 2 discusses the related work in this field. The proposed semantic grid architecture is discussed in section 3. The definition of ontology with respect to our work is presented in section 4. The knowledge layer implemented in the architecture is discussed in detail in section 5. In section 6, the design and implementation of various modules of the proposed knowledge layer are described. Experimental results are presented in section 7. Section 8 concludes the paper highlighting the advantages and future scope of this research work.

2. Related Work

In [17], an ontology based Matchmaker Service is proposed that supports dynamic resource discovery and resource descriptions. However, the request is expressed using request ontology and hence there is a need to compile the user request as ontology descriptions.

Searching by traditional search engines based on keywords [18] has its own problems. As a result they do not check semantic of search objects and simply treat them as character strings. A lot of irrelevant information will be returned to the user as long the keywords appear somewhere in their files.

In [19], concept-weight algorithm is proposed in which similarities between two Ontologies have been determined.

The literature [9] proposes Semantic Grid Node Framework in which structuring of the ontology of nodes are motivated by the need to provide several essential types of knowledge about a node.

The literature [21] proposes a meta search engine called “guided google”, that is built using the Google web services. This search engine guides and allows the user to view the search results with different perspectives, which achieved through simple manipulation and automation of Google functions. However, the functionalities provided in this engine are based on “combinational keyword searching” and it neither supports semantic description nor performs semantic search.

The literature [26] addresses the problem of resource description in the context of resource broker to broker for resources described by several Grid middlewares including Unicore.

Our approach for semantic description of grid resources exploits Grid Information Service of Globus Toolkit to aggregate resource information and dynamically creates knowledge base using resource ontology template. The template is extensible to accommodate new type of resources, and the discovery module discovers more accurate results than conventional mechanism.

3. Proposed Layered Architecture

We proposed a five layered architecture that implements a knowledge layer on top of Gridbus broker as shown in Figure 1 and it can be used for building semantic grid infrastructure.

Fabric Layer

The Grid Fabric layer provides the resources to which shared access is mediated by Grid protocols. The resources may be computational resources, storage systems, catalogs, network resources and sensors or may be a logical entity, such as a distributed file system, computer cluster, or distributed computer pool.

Core Middleware Layer

This layer consists of low-level middleware that provides a secure and unified access to remote resources. Depending on the type of resources we can choose different middlewares such as Globus, Unicore, Alchemi, SRB. Using services of such low-level middleware layer, one can create high-level middleware services that support rapid creation and deployment of applications on global Grids.

High Level Middleware layer

In our architecture, this layer is implemented using Gridbus broker. The Gridbus broker follows a service-oriented architecture and is designed on object-oriented principles with a focus on the idea of promoting simplicity, modularity, reusability, extensibility and flexibility [10]. The broker uses the semantic discovery services offered by knowledge layer. The broker has been designed to operate with different grid middleware framework and toolkits such as Globus that primarily runs on Unix-class machines and Alchemi, which is a .NET based grid computing platform for Microsoft Windows enabled computers. Hence it is possible to create a cross-platform grid implementation using the Gridbus broker.

Knowledge layer

The knowledge layer is the top layer that provides services, which can look for patterns in existing data repositories and manage information services. The knowledge layer provides knowledge discovery from a huge amount of data aggregated from underlying information services layer. Moreover, this layer is domain oriented and usually uses domain knowledge built with domain ontology.

Application layer

The application layer enables the use of resources in a grid environment through various collaboration and resource access protocols. The semantic portlet

present at this layer allows the resource requester to submit the query and the semantic retrieval of suitable resource from the service ontology using reasoner.

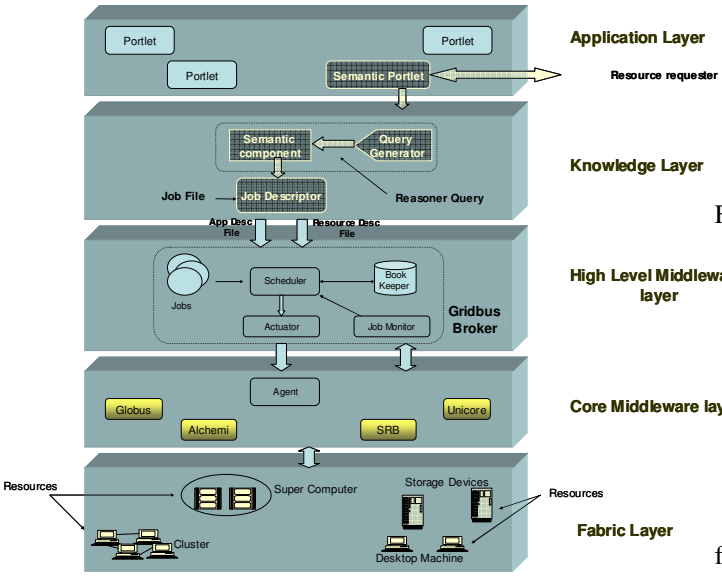


Figure 1: The Semantic Grid Architecture.

4. Defining Ontology

The modules implemented in the knowledge layer use semantic web's approach of making information understandable by computers. Information must therefore be described in such a way that computers can interpret it and derive its meaning. Computer understandable information is one that is annotated with semantics which describes the meaning of the information. The annotations themselves have to be defined so that computers can interpret and reason with them [4, 13]. A collection of annotations where their meanings are described is called an ontology. Ontologies are used to capture knowledge about some domain of interest. Ontology describes the concepts in the domain and also the relationships that hold between those concepts [4].

According to Karlsruhe Ontology model [13], an ontology with datatypes is a structure O:= (C, T, $\leq_C$, R, A, σ_R, σ_A, $\leq_R$, $\leq_A$, I, V, t_C, t_T, t_R, t_A) consisting of

- Six disjoint sets C,T,R,A,I and V called *concepts, datatypes, relations, attributes, instances* and *data values respectively.*
- Partial orders $\leq_C$ on C called *concept hierarchy* or taxonomy and $\leq_T$ on T called *type hierarchy*.
- Functions σ_R: R→ C^2 called *relation signature* and σ_A: A→CxT called *attribute signature*.
- Partial orders $\leq_R$ on R called *relation hierarchy* and $\leq_A$ on A called attribute hierarchy, respectively.
- A function t_C: C→ 2^I called *concept instantiation.*
- A function t_T: T→ 2^V called *datatype instantiation.*
- A function t_R: R→ 2^{IxI} called *relation instantiation.*
- A function t_A: A→ 2^{IxV} called *attribute instantiation.*

Further, we can define following relations viz,

- Subsumption relation $\leq_c$: (C x C) U (R x R) → {true, false} for example "$\leq$c (g,h) = true" means: "g is a subtype of h"[22].
- Conformity relation, which relates each individual instance in I to one, and one only, concept type in C. conf: I → C

Different ontology languages provide different facilities. The most recent development in standard ontology language is Web Ontology Language (OWL) from the World Wide Web Consortium. OWL is developed as a vocabulary extension to RDF and is derived from DAML+OIL with more description power than RDF, such as disjoint and cardinality [25].It is based on a different logical model which makes it possible for concepts to be described and thereby complex concepts can be built up in definitions out of simpler concepts [4, 5]. Protégé, an OWL editor developed by Stanford University, facilitates the creation of ontology and to build knowledge base. An inference engine can be used to query the knowledge base and retrieves information. In this paper, Algernon inference engine is used to interact with Protégé knowledge base for semantic retrieval of information.

5. Proposed Knowledge Layer

In the Grid environment, users and software agents should be able to discover, invoke, compose and monitor grid nodes offering required services and having particular properties. To address this issue, we propose a knowledge layer that implements various modules for semantic description and discovery of grid resources.

It also provides an interface, called ***job descriptor***, to the Gridbus broker for job execution on the resource discovered. Figure 2 identifies various modules implemented in knowledge layer. The two main modules present in semantic component namely the resource description and discovery module are the

central part of the knowledge layer. We describe various modules implemented in the knowledge layer in the following sections in detail

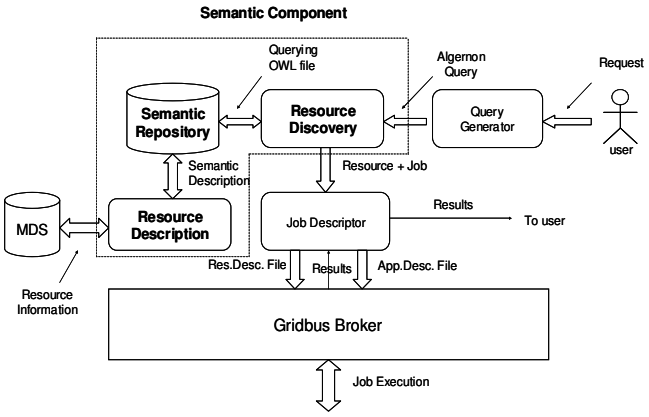


Figure 2: Knowledge Layer.

5.1 Resource Description using Ontology Template

To date, ontology creation has been a manual process. In [12], common sense knowledge was extracted manually from different sources and expressed using Ontologies. This is inevitably a very laborious process, and there is a need to atleast partially automates the process of ontology creation and knowledge extraction. Hence, we can imagine a predefined ontology of concepts and relationships, plus a knowledge base of instances [11]. Our resource description module defines resource ontology template created using Protégé editor and it provides necessary concepts and properties with which a resource can be described. Different possible computing resources are considered for creating ontology template. Our structuring of the ontology of resources is motivated by the need to provide semantic information about a resource and also to provide transparent access to Grid resources. We propose the following precise definitions to explain the motivation behind the creation of ontology template and how it can be used for semantic description.

Definition 1: *An ontology template is a domain specific ontology that provides hierarchy of concepts along with properties to define their characteristics.*

Definition 2: *Any resource can be modeled as an instance of a specific concept provided that the resource can be described using the properties defined in that concept.*

Once the ontology template is created, we can build knowledge base with the instances and the specific property instantiations. Together the ontology and the knowledge base make up a ***semantic repository***. Whether, the two parts are stored separately, e.g., in two distinct relational databases, depends on the practicalities of the implementation [11]. When a resource is registered in the grid, its information can be obtained using grid resource monitoring tool like MDS of Globus Middleware [8, 23]. With this information, instance of appropriate resource concept in the ontology template is created for every computing resource in the grid. The characteristics of the resource for ex, freeRAMSpace, is also defined in the respective property of the appropriate resource concept in the ontology template. For example, an existence of a computer with Linux OS can be represented in the ontology template by creating an instance for the concept "computer" and the "hasOS" property of the concept "computer" will be assigned the value "Linux".

Protégé-OWL APIs is used to dynamically create instance of a particular concept and also to assign values to appropriate properties in the resource ontology template. With these features, the resource information of the entire grid environment can be described semantically which in turn enables semantic discovery of grid resources.

5.2. Resource Discovery Module

This module allows the users to submit the information about required resource to execute their job. It then generates appropriate Algernon query depending on the requirements specified by the user and executes these queries over the ontology knowledge base to obtain best possible resources closely matching to the request. The discovery module retrieves resources that exactly match with that of request. It also retrieves resources that exhibit subsumption relation when exact matching is not found. Once suitable resource is obtained, the resource discovery module submits the resource information along with the user's job to the job descriptor. With this information, the job descriptor creates an application description file and resource description file. Both these files are required by the broker to successfully run the application in the specified resource. The broker executes the job on the specified resource and generates results. The discovery module aggregates the results from the broker and delivers to the user.

6. Design and Implementation

The architecture has been implemented and tested in Grid Computing Laboratory at Anna University. The

necessary softwares including Globus Toolkit 4.0 and protégé have been successfully installed in twelve machines. All the components of globus toolkit have been successfully configured and tested for proper operation. Also, the MDS component has been tested properly so that the *grid-info-search* tool of MDS retrieves the resource information of the local host and stores it in the *ldap* server. All prerequisite libraries have been installed and tested for proper functioning.

6.1 Creation of Ontology Template

Protégé has been installed in one of the machine (which doesn't have to be a resource provider) and ontology template has been created by considering different computing resources in the grid. The concept of these resources has been defined properly using relations and properties so that the characteristics of any resource can be defined by its properties. For illustration purpose, we define the concept of "RAM" as

C = {owl:Thing, RAM, C1024, C128, C256, C512},. For simplicity, we here described in detail about the concept of "C1024".

T = {Integer},

$\leq_C$ = {{owl:Thing, RAM},(RAM, C1024},{RAM, C128},{RAM, C256},{RAM, C512}},

A = {hasFreeMB}, R = {presentInComputer}, σ_A = {(hasFreeMB, (Parameter, Integer))}

σ_R = {(presentInComputer, (RAM, WorkStation))}, I = {C1024_0}, V = {192},

t_C = {C1024, {C1024_0}}, t_T = {Integer, {192}}, t_R = {presentInComputer, {C1024, (g06.grid)}})

t_A = {hasFreeMB, (Integer, {192})}

Also, the set of concept types {C1024, C128, C256, C512} exhibits subsumption relation with concept type {RAM}. Hence, we can define the subsumption relation for the concept C1024 with RAM as follows:-

"≤c (RAM, C1024) = true"

Figure 3 shows the ontology template with concept hierarchy considered. The values of the properties considered to define concepts, can be retrieved from MDS.

6.2 Creation of Knowledge Base

The MDS component offers tool to retrieve resource information of the host, which is accessed through remote machines. The *grid-info-search* tool aggregates various properties of grid nodes and stores it in the *ldap* server which is retrieved using suitable *ldap* query. The resource description module is developed using Java programming language. The module accesses the grid nodes and retrieves resource information by executing *ldap* queries on those nodes and updates into the ontology template. The Protégé editor offers versatile libraries called Protégé-OWL APIs with which one can manage ontology and perform several operations over the ontology those include creating and deleting the instances of concepts, assigning values to the properties etc. For every type of information retrieved from the grid node, we create instances of appropriate concept in the ontology template forming conformity relation between instances and their respective concept types. Further, an instance will be exactly related to only one concept type**.** Also, the values of various properties retrieved are assigned to respective properties of the appropriate concepts in the ontology template.

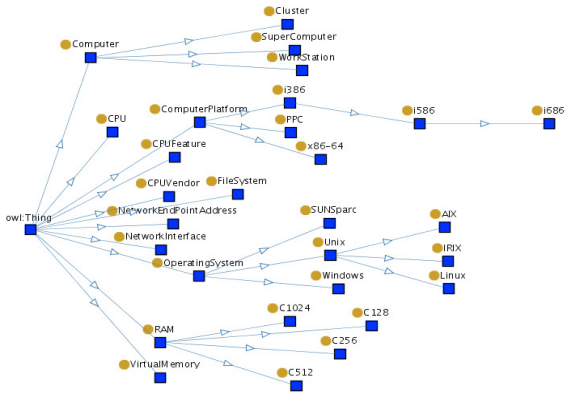


Figure 3: The ontology template shows concepts and properties.

At this point, the ontology template with concepts and properties and corresponding instances and property values together constitutes knowledge base of the grid resources. This semantic description of resources facilitates the use of inference engine to interact with the knowledge base and retrieves information semantically. Moreover, the description module is made to execute periodically so that addition and removal of resources is accounted in the knowledge base dynamically.

6.3 Resource Discovery Mechanism

The discovery module relies on the power of Algernon inference engine. In order to make the conversion of user query into Algernon query and also to provide flexible mechanism of querying, we propose query tags with the format ***label:label_value*** in which the properties of the resource are denoted as ***label*** and requested value as ***label_value***. We also include operators in query label for flexible querying. For Ex., If the user wants to search for machines with free RAM value greater than 200 MB, the query should be ***RAM:>200***. Currently, the system supports >, <, = and also NOT operators. Also, the query

mechanism is designed in such a manner that it can query a resource with multiple constraints. For Example, if the user wants to query a machine with free RAM of 200MB and free Harddisk space of 10000MB, then the query *"freeRAM:200 freeHDD:10000"* will retrieve all resources with 200MB and harddisk space with 10000MB.The Query generator module parses the user query using regular expression, stores **lefttag** and **righttag** in a vector and converts it into suitable Algernon query. The screenshot of the resource discovery porlet is shown in Figure 4.

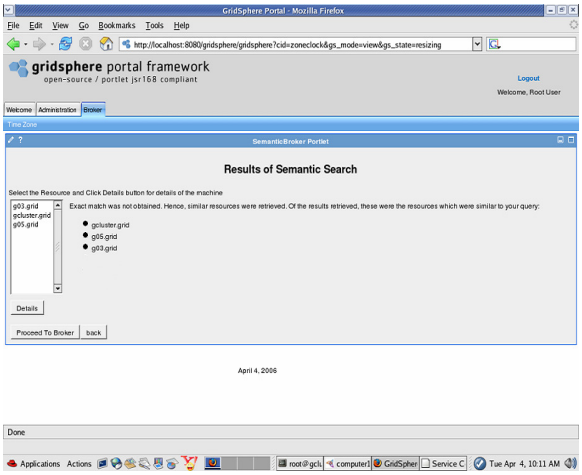


Figure 4: Resource Discovery Module.

We maintain a database where we store the ***label***s used in ontology template and its corresponding possible synonymous words to reduce the complexity of providing exact ***label*** while querying. Still, if the user specifies a ***label*** which is not present in the database, the discovery module will fail, even though, the intended resource is available. In this case, we provide an interface where we list all ***label***s used in the ontology template and allows the user to update new synonymous words for the label the user intends as shown in Figure 5.

The module will then execute the queries over the knowledge base of the grid and obtains the resource that is matching with the user's request. Algernon reasoner not only discovers the resource that exactly matches with that of the request, it also retrieves resources that exhibit subsumption relation when the exact match is not found.

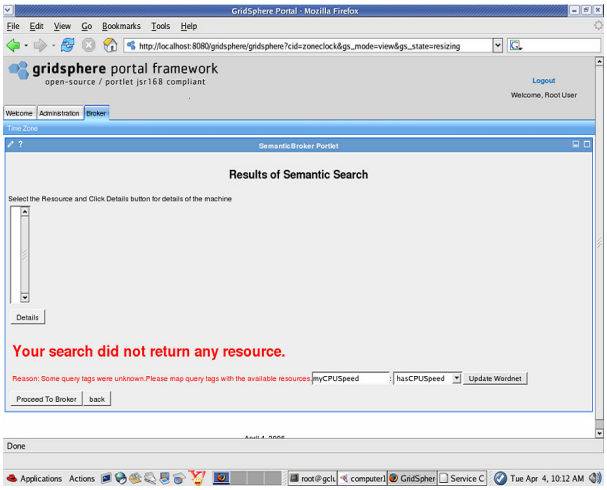


Figure 5: Interface to update database.

For example, if the user requests a computer with IRIX Operating system, the module retrieves all instances of Unix. This is because the knowledge base does not possess instance of IRIX machines and infers from the ontology template that IRIX is a subconcept of Unix that is,

"≤ (Unix, IRIX) = true"

as shown in the Figure 6.

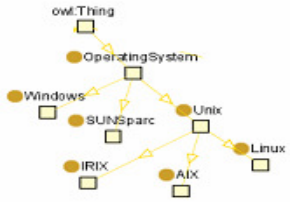


Figure 6: Concept Hierarchy-Operating System

Hence, the discovery module retrieves instances of Unix as it is compatible with IRIX Operating System.

The discovery also provides necessary user interface to submit the job and execute it in the resource obtained from the discovery module. Both the resource and job information are submitted to the job descriptor for execution.

6.4 Job Descriptor

The job descriptor generates two files namely the application description file and resource description file using the information provided by the discovery module. These files are needed by the broker for execution of the job on the specified resource. In this paper, we consider only compute resources which can be local or remote. We have created a simple application that performs multiplication of two numbers. The application has been compiled successfully and class file is created. The user

searches for the resources using the semantic component which in turn discovers suitable resource, providing the job descriptor with the hostname of the resource. Meanwhile, the user's job and the Unix command needed to execute the job are submitted to descriptor, which will submitted to the broker to initiate scheduling of jobs. Once the execution is over, the results will be collected and presented to the user.

7. Experimental Results

The performance of the discovery module has been evaluated by comparing the results obtained from various queries as shown in Figure 7. Even though, the resource requested by the user is not available in the knowledge base, the module retrieves more closely matching results.

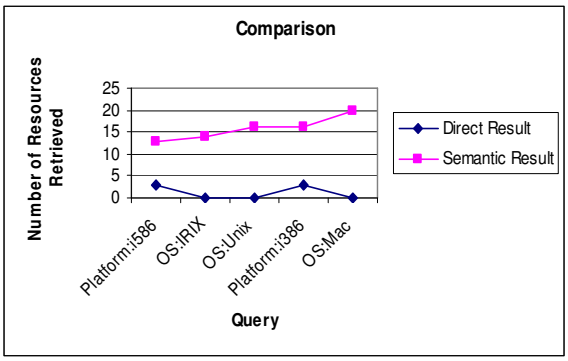
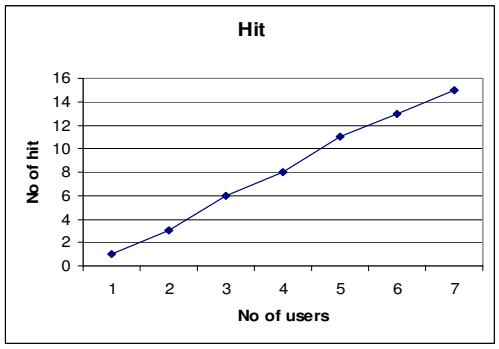


Figure 7: Direct Vs Semantic Results.

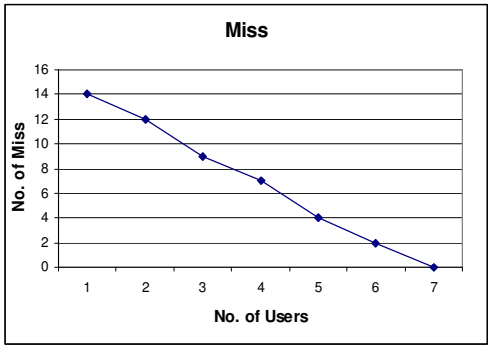
Similarly, we executed several queries and observed that the module performing context based searching retrieved more suitable match, when direct inference failed.

We also evaluate the effect of updating database with new ***label*** through the user. The performance of the module has been estimated by considering 'hit' and 'miss' while searching for a particular query. We tested our discovery module with several naïve users to navigate and search for a particular resource using its various characteristics for about fifteen iterations. Queries were fired on the *knowledge base* by specifying single property of the resource or more than one property and then the result is obtained. Whenever, the user provides a new label which is unknown to the discovery module, the searching process fails. Consequently, the discovery module allows the user to map the unknown label to the corresponding label used in the ontology template and thereby update the database with new label. Initially, the database of synonymous words is empty and it is updated by the user whenever he has provided a new label.



a) Hit Proportion.

Hence, with the number of users increasing and the amount of synonymous words are also increasing, the semantic search becomes more efficient as the user grows as shown in Figure 8.



b)Miss proportion

Figure 8: Effect of database updation in semantic discovery of resources.

8. Conclusion

The knowledge layer implemented in the proposed architecture enables the Gridbus broker to discover resources semantically. The resource description module has been designed so that any entry and removal of resources is reflected in the ontology template making the system more flexible. With this ontology template, we overcome the difficulty of resource provider to have the knowledge of Protégé ontology editor. However, the ontology template developed depends on MDS component and hence it may not support middleware other than Globus. Since, the discovery module relies on the power of Algernon inference engine, incorporating forward and backward chaining rules will make the system more efficient.

Acknowledgements

This work was undertaken as part of "Garuda: The National Grid Computing Initiative", India and is financially supported by the Centre for Development of Advanced Computing, India. The development of

the Gridbus Grid Service Broker at the University of Melbourne is partially supported by the Australian Research Council via a Discovery Project grant.

References

1. I. Foster and C. Kesselman (eds), "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, 1999, 259-278.
2. I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Virtual Organizations", International Journal of High Performance Computing Applications, 15(3), 200-222, 2001.
3. I. Foster, X. Kesselman, J. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.
4. OWL Web Ontology Language Overview. W3C Recommendation, 10 February 2004
5. T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web", Scientific American, May 2001.
6. R. Akkiraju, R. Goodwin, P. Doshi, S. Roeder, "A Method for Semantically Enhancing the Service Discovery Capabilities of UDDI", Proceedings of IJCAI Information Integration on the Web Workshop, Acapulco, Mexico, August 2003.
7. "UDDI: Universal Description, Discovery and Integration", www.uddi.org
8. www.globus.org/toolkit/mds.
9. Y. Zhang and W. Song, "Semantic Description and Matching of Grid Services Capabilities". Proceedings of the 3rd UK e-Science All Hands Meeting, Sept. 2004, UK.
10. S. Venugopal, R. Buyya and L. Winton, "A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids", Proceedings of the 2nd International Workshop on Middleware for Grid Computing (Co-located with Middleware 2004, Toronto, Canada, October 18, 2004), ACM Press, 2004, USA.
11. J. Davies, R. Studer, Y Sure and P W Warren "Next Generation Knowledge Management" BT Technology Journal, Vol. 23, No. 3, July 2005.
12. B. Lenat and V. Guha, "Building Large Knowledge Based System: representation and interface in the Cyc project", Addison-Wesley (1990).
13. G. Stumme, M. Ehrig et. al., "The Karlsruhe view on ontologies", Technical report, University of Karlsruhe, Institute AIFB (2003).
14. D. Roure, N. Jennings and N. Shadbolt, "The Semantic Grid: A future e-Science Infrastructure", Grid Computing – Making the Global Infrastructure a reality, John Wiley & Sons, Ltd, 2003.
15. M. Li, P.Van Santen, D. Walker, O. Rana, M. Baker, "SGrid: a service-oriented model for the Semantic Grid", Future Generation Computer Systems 20, July 2004, PP 7-18.
16. http://algernon-j.sourceforge.net/tutorial
17. A. Harth, Y. He, H. Tangmunarunkit, S. Decker, C. Kesselman, "A Semantic Matchmaker Service on the Grid", WWW2004, May 17-22, 2004, 326-327.
18. E. Alberdi D. Sleeman, ReTAX: "A Step in the Automation of Taxonomic Revision, Artificial Intelligence", 1997, pp257-279.
19. M. Gao, C. Liu, F. Chen, "An Ontology Search Engine Based on Semantic Analysis**",** Proceedings of the Third International Conference on Information Technology and Applications, 2005.
20. P. Asadzadeh, R. Buyya, C. Kei, D. Nayar, and S. Venugopal, "Global Grids and Software Toolkits: A Study of Four Grid Middleware Technologies", *High Performance Computing: Paradigm and Infrastructure*, Laurence Yang and Minyi Guo (eds), ISBN: 0-471-65471-X, Wiley Press, New Jersey, USA, June 2005.
21. D. Hoong and R. Buyya, "Guided Google: A Meta Search Engine and its Implementation using the Google Distributed Web Services", International Journal of Computers and Applications, Volume 26, No.1, ISSN: 1206-212X (202), ACTA Press, Calgary, Canada, 2004.
22. P. Nguyen and D. Corbett, "A Basic Mathematical Framework for Conceptual Graphs", IEEE Transactions on Knowledge and Data Engineering, Vol 18, No2, February 2006.
23. K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, "Grid Information Services for Distributed Resource Sharing", Proceedings of 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10), San Francisco, IEEE Computer Society Press, Los Alamitos, CA, USA, 2001.
24. J. Yu and R. Buyya, "Grid Market Directory: A Web and Web Services based Grid Service Publication Directory", Technical Report, GRIDS-TR-2003-0, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, January 2003.
25. F. Tao, S. J Cox, L. Chen, N. R. Shadbolt, F. Xu, C. Puleston, C. Goble, W. and Song, "Towards the Semantic Grid: Enriching Content for Management and Reuse", e-Science AHM 2003, Nottingham, 2-4 September 2003.
26. J. Brooke, D. Fellows, K. Garwood, and C. Goble, "Semantic Matching of Grid Resource Descriptions", 2nd European Across-Grids Conference (AxGrids 2004), 28-30 January 2004, Cyprus.