A GRID-ENABLED INFRASTRUCTURE FOR RESOURCE SHARING, E-LEARNING, SEARCHING AND DISTRIBUTED REPOSITORY AMONG UNIVERSITIES

Anand Jumnal[1] and Dilip Kumar S M[1]

[1]Department of Computer Science Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India

ABSTRACT

In the recent years, service-based approaches for sharing of data among repositories and online learning are rising to prominence because of their potential to meet the requirements in the area of high performance computing. Developing education based grid services and assuring high availability, reliability and scalability are demanding in web service architectures. On the other hand, grid computing provides flexibility towards aggregating distributed CPU, memory, storage, data and supports large number of distributed resource sharing to provide the full potential for education like applications to share the knowledge that can be attainable on any single system. However, the literature shows that the potential of grid resources for educational purposes is not being utilized yet. In this paper, an education based grid framework architecture that provides promising platform to support sharing of geographically dispersed learning content among universities is developed. It allows students, faculty and researchers to share and gain knowledge in their area of interest by using e-learning, searching and distributed repository services among universities from anywhere, anytime. Globus toolkit 5.2.5 (GTK) software is used as grid middleware that provides resource access, discovery and management, data movement, security, and so forth. Furthermore, this work uses the OGSA-DAI that provides database access and operations. The resulting infrastructure enables users to discover education services and interact with them using the grid portal.

KEYWORDS

Grid Computing, Educational Grid, E-learning, Searching, Distributed Repository, Globus Toolkit

1. INTRODUCTION

Grid computing has brought tremendous convenience plat-form for the scientific community that allows sharing, selection and aggregation of computing resources for access, security, interoperability and new business models. The resources and services of grid are distributed with dynamic and heterogeneous characteristics. This may overcome computing power and storage capacity of conventional computing platforms. The large collaborative environments, potentially composed of multiple distinct organizations, uniform controlled access to data has become a key requirement if the organizations are to work together [1]. The data trade-off between many universities requires data file management systems to manage operations like data storage, transfer access and replication. Therefore grid computing provides various possibilities to support innovative educational and scientific applications.

DOI:10.5121/ijgca.2016.7202

In parallel with the development of grid computing, web services (WSs) are becoming widely accepted as a way of providing language and platform-independent mechanisms for describing, discovering, invoking and orchestrating collections of networked computational services [2] [3]. The emergence of service-oriented approaches on grid has changed the view of educational and scientific communities and has gained considerable attention for supporting distributed application development in e-science, e-education and e-commerce. The principal strengths of web services and grid middleware are complementary, with web services focusing on platform-neutral description, discovery and invocation, and the later focusing on the dynamic discovery and efficient use of distributed computational resources. This complementarity has given rise to the service-based grids which make the functionality of grid middleware available through web service interfaces [3].

Education is an eminence domain that can be benefited more from grid computing by collaborating web services and grid middleware. The focus is on leveraging technologies and products from many universities around the world in the area of e-learning, resource sharing, searching and repository management. Online learning has become a significant approach for modern education system without time and place restriction that is enabled by means of computer technologies like distributed computing and web based application over the network [4]. Yet to date, few universities in the world have managed to exploit the aggregate power of this seemingly infinite grid of resources for educational purposes. One can imagine the data produced by universities (academic, administration, scientific and computational data) that require a variety of heterogeneous resources that is not available on a single computer [5].

In developing countries, as observed, most of the educational content used by an institute is not shared or used by other institutes to the full extent due to the lack of infrastructure and awareness. Therefore, most of the content developed remains under the control of the creators. Though the creators are willing to share their content, they do not find an infrastructure that easily enables them to do. The grid technology can leverage the educational application provided at the current state by making available the dynamic resources to enhance the quality of education. The grid computing systems can be used to develop an efficient and economical e-learning platform. Service oriented concepts are the new platform-independent techniques with open standards and protocols that have advantage over web services [4]. In spite of the grid being a decade old, it has proven to be a promising and matured technology to provide high computing resource and security. This paper proposes a grid based system that enables easy searching and sharing education content of different universities that offers a solution for both educational content distribution and application computing environment. The platform allows users to connect and access the grid based system anywhere and anytime by using PCs, portable or mobile devices via the Internet. This system is implemented by using the functionalities associated with widely used grid middleware software Globus toolkit [6]. In order to propel the effect of such incorporation, let us consider real world example described below.

1.1. Example Scenario

Consider a real-time example of education system environment that enables students, faculty and researchers to store, search and share academic, administration and research data through a portal of respective Institute. The community (students, faculty and researchers) may also require to access data of academic and research happening around the location by searching in different repositories. This whole scenario requires provisioning and communication of data available through the PCs, laptop and mobile user's handheld devices with wireless connection. Analysis of

this kind of complex data sets for different purposes requested by different representatives may involve a whole range of techniques which are simply infeasible to perform on single machine of individual Institute.

In this scenario, it is desirable to have a better service assurance in order to share the data among multiple universities. (i.e. distributed repositories). For instance, depending upon the area of interest in variety of information and importance of the results required, one would look for searching for better and more results rather than in a single repository. Therefore, in order to solve this problem moving this scenario to grid environment that supports large numbers of distributed resources sharing and provide a good platform for the long-life, online and educational mobile applications to achieve quality data, high availability, reliability and scalability for education system.

1.2. A Broad View of the Grid Infrastructure for Education

The broader grid infrastructure facilitate seamless sharing of resources within a campus, across sister campuses and between different institutions and integrate heterogeneous platforms and resources. Grid supports a range of user groups with varying application needs and levels of quality of services. Fig. 1 presents an overview of the layered grid infrastructure for education community. The design of such grid infrastructure can benefit by providing high computing resources and middleware supports for application specific services. The rest of the paper is organized as follows. Section 2 presents the background and motivation of the work. Section 3 describes the services on educational grid. Section 4 overviews of grid and globus middleware. In Section 5, the education based grid framework model is proposed. Section 6 presents experimental setup and results. Section 7 presents conclusion and future work of the paper respectively.

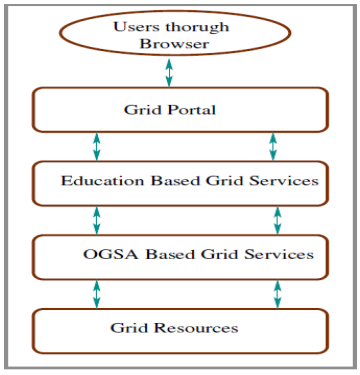

Fig.1: A broad view of the grid infrastructure for education

2. BACKGROUND

The idea of data and resource sharing in large scale with high performance is fundamental, that might be assumed a system like educational grid may surely already been widely deployed. In practice, whilst the need for these technologies is indeed widespread, we find only primitive and deficient solutions to next level of education. This work exploits the grid service technologies by

orchestrating the distributed educational resources to meet the challenges posted by the increasing demand of computing power and sharing of data resources.

2.1. Current State of Art in Grids

It is clear that, in most research fields, scholarly communication is a global activity; knowledge is developed across and between institutional and national boundaries; many academics have a stronger bond to subject peers than to institutional colleagues. Widespread sharing of data may lead to discovery and use data out of discipline in which the data were created, fostering interdisciplinary research and learning. One of the biggest international collaboration that conducts research and development to create fundamental grid technologies is the Globus Alliance [7], that implements some standards developed at a Open Grid Forum through Globus toolkit (GTK). GTK is grid middleware software includes set of service implementations for infrastructure management, java libraries for building web services and both client APIs and command line interfaces for accessing various services and capabilities. It also includes Open Grid Service Architecture (OGSA) [8] framework which supports to building systems. The latest version of the globus project comply with the Web Service Resource Framework (WSRF) [9], a set of standard web services interfaces to develop grid services.

2.2. Grid Test Beds

The current test beds of grid technology provide diverse ways to implement the applications for distributed data. In the following, some of the popular national and international research and development projects on grid are listed. Europe has achieved a prominent position in this field, particularly with its success in establishing a functional grid test bed comprising more than 20 centers. Some of the projects include DataGrid (CERN) [10], EuroGrid [11], GridLab [12], UK E-Science [13], etc. The DataGrid project aims to support the data access and computation needs of demonstration projects in high energy physics, earth observation data, and bio-sciences. EuroGrid project aims for middleware software and other software components and integrate them into EuroGrid (dynamic resource broker, accounting and billing, interface for coupled applications and interactive access). GridLab Testbed is a Pan-European distributed infrastructure which consists of heterogeneous machines from various academic and research institutions.

Nonetheless, the United States also developed several projects on grid middleware softwares and technologies funded by National Science Foundation (NSF) namely Distributed Teras-cale Facility (TeraGrid) [14], GridCenter [15], GrADS [16], iVDGL [17]. GridCenter goal was to create a stable middle-ware infrastructure to permit seamless resource sharing across virtual organizations and provisioning to define, develop d-ploy and supporting for science and engineering applications. GrADS supports for manipulation and visualization of earth science data; allows sophisticated analysis and display applications. iVDGL drives the development and transition to every day production use of Petabyte-scale virtual data applications. India also has contributed some grid projects. C-DAC's Garuda [18] aggregates geographically distributed resources enabling researchers and students from institutes across the country to conduct experiments requiring high computing facility without interruption. BIOGRID [19] disseminated bio-informatics information to researchers in the country.

As observed, most of the projects on grid technology are commercialized and found best utilization for that. But only few projects in which the functionalities are incorporated for

educational students academicians and researchers. Hence, in this paper we provide a platform for students and researchers to share the data and knowledge using grid technology.

2.3. E-learning and Search Engines

This is the new paradigm shift of teaching and learning practices in education system through e-learning services. These services are mainly based on information transfer and sharing paradigm that can more focuses on new learning strategies with appropriate software tools and environments. The more effort has been put to establish the ELeGI project [20] (European Learning Grid Infrastructure). The project focuses on developing software technologies for effective human learning and promoting and supporting a learning paradigm shift. Within the context of web based education system, the PEDC [21] system provide a grid file system to manage educational resource files and to build a writable, server-less data grid with large scale. The e-learning grid in [22] discusses peer-to-peer technologies for content distribution among independent institutes/universities. It introduces a collaborative computing platform framework that supports the creation of multi-user collaborative sessions, allowing users to self-organise and communicate, share tasks, workloads, and content, and interact across multiple different computing platforms. The work in [23] proposes a service oriented e-learning systems that include assessment, course management, grading, registration and reporting web services. e-framework in [24] is a service-oriented approach for education and research, the methodologies used helps to identify shared and common services that form a part of the information environment. Further, [25] accomplish a blended e-learning grid framework that includes the multi-agent system that can integrate with other e-learning grid systems. Unlike the other projects that are commercialized and found best utilization of grid technology, the proposed education based e-learning grid framework system provide a platform and services for students academicians and researchers to share the knowledge using grid technology.

3. PROBLEM DEFINITION

The problem is to facilitate services such as searching, e-learning and distributed operations for the education com-munity among different university repositories using grid. Setting up grid infrastructure and platform that connects many universities to share resources for e-learning applications and as well as data is to be proposed. In this view, the problem is solved using grid services, Globus toolkit, OGSA-DAI and other components.

The proposed work focuses on education teaching and learning materials management by uploading, searching and sharing among university repositories across locations. Learning materials are defined as subject notes, reports, publications etc. These are stored in respective local repositories of sites. These documents are classified by domain or department with corresponding category to represent structure information of learning materials. Term frequency of extended vector space model [26] is adopted for keyword based content retrieval and the strategy weighting scheme is used to accentuate the importance of structure. In order to reduce the complexity of index, the keywords are sorted out by their most significant bit and stored in the respective cluster. For each new occurrence of keyword, get assigned to new cluster i.e maximum of 26 clusters are created to store the possible keywords from the document. The query is constructed with domain-id and keywords to search the relevant documents in the specific domain that increases relevancy and reduces the response time. These operations are combined to implement education based grid framework. The feasibility of this work is guaranteed because of the grid services and supportive components of the grid.

4. SERVICES ON EDUCATIONAL GRID

In this section, the services are categorised into two, namely operation level services and management level services and are briefly explained below.

4.1. Operational Level Services

The operation level services consist of operations related to the education service delivery, that includes (i) searching, (ii) storing/uploading the academic and research data to the grid, (iii) e-learning services including online course, examination etc. There are dynamic interaction and inter-operation among all the services at this level.

1) **Searching:** Searching data facilitates the users to search the data in all the different repositories of the universities that are connected to the grid. Searching in collaborated repositories increase the users experience and knowledge.
2) **Storing/Uploading services:** This service allows the community to store/ upload the data in the provided space of the grid repository. The stored data is accessed through grid portal. While uploading a document, the user should provide appropriate information (metadata) of document to store in standard hierarchy such as domain, title, keywords, authors, and etc. The usability of these parameters are explained in Section 6.
3) **E-learning service:** E-learning service enables the users from remote locations to use online education services such as classes, exams, multimedia materials, publications and other e-learning applications. These services allow community to personalise the learning process according to their preferences.

4.2 Management Level Services

The services at this level are responsible for the management of operational level services and making them available on the grid. These services perform operations, such as defining new services, service access, configuring services, fault tolerance, etc. The management level services can be further categorized into admission control, registration and description, management and configuration, brokering and scheduling, ubiquitous access and retrieval, data caching and hoarding, and fault tolerance services. These services are organized in a layered taxonomy according to their order of occurrence, as shown in Fig. 2 and each layer is briefly explained in sequence as follows.

1) **Admission control:** The admission control service is the first layer in the hierarchy which is responsible for analyzing and decides to accept or reject the user requests based on the availability of resources. This service enables the utilization of grid resources effectively and examines correct metadata, standard size and relevancy of data (text/video/audio) to the domain that enables easy and secure access and, helps in improving the user satisfaction. This service prevents the system from bottleneck performances.
2) **Registration and Description:** This service layer is responsible for the registration or deregistration of any grid resources of registered institutes. This also includes authenticating the member registration and authorization of each individuals, groups or institutes/universities to utilize educational grid resources.
3) **Management and Configuration:** This service hold the responsibility for the on-demand and handling of online re-sources by adding new resources, deleting or modifying the

existing resources. The service provides to check the resources availability status for any of the registered education grid resources. In addition, it searches for academic and research related data or files that can be configured by using this service. It is also responsible for content updates.

4) **Brokering and scheduling:** This service is responsible for the brokering of heterogeneous systems and scheduling of jobs over grid and selecting the most appropriated resource for the requested task. This service layer supports in increasing the resource utilization of the grid by using resource filter and rankers by prioritizing the requests and obtaining the various operations performed by other resources. Such services could be designed to interact with grid middleware components such as Storage Resource Broker (SRB) system. The broker system provides a storage service by managing users, file locations, storage resources and metadata catalogue information. SRB with metadata catalogue provides to access data sets and resources based on their logical names rather than physical locations i.e. the functionality hides the physical location of the file by providing a logical view to the user's files.

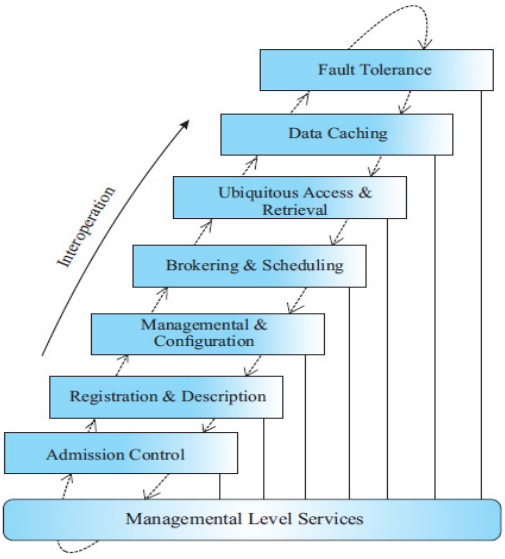

Fig.2: Management level services

5) **Ubiquitous Access and Retrieval:** The ubiquitous ac-cess and retrieval service allows ubiquitous access, storage, retrieval, analysis, management and sharing of all types of academic and research-specific data and files, such as academic notes, seminar/project reports, research publications and all other education-related and technological types of data.
6) **Data Caching:** This caching service is responsible to improve the scalability and performance of applications that access data by caching frequently used data on middle tier system.
7) **Fault Tolerance:** The fault tolerance service is responsible for error-handling tasks such as malfunctioning of another service, broken service link, or identification of the fault. For example, this service records the failure reporting on academic reports, file searching error, visualization ambiguity etc. This service can also facilitate for data availability by employing the information redundancy that seeks to provide fault tolerance through replicating the data.

5. GLOBUS AND OGSA-DAI OVERVIEW

The key strategy of the grid computing is to use middleware to divide and apportion pieces of a program among several computers.

5.1 Components of Globus

The Globus toolkit [27] is a community-based, open-architecture, open-source set of services and software libraries that support grids and grid applications. The toolkit includes information infrastructure, resource management, data management, communication, fault detection, security and portability. It is packaged as a set of components that can be used either independently or together to develop useful grid applications. GTK include the following:

5.1.1 Grid Security Infrastructure (GSI) [28]

The GSI provides essential building blocks for other grid protocols. This authentication and authorization system makes it possible to authenticate a user just once using public key infrastructure (PKI) mechanisms to verify a user-supplied "grid credential." GSI handles the mapping of the grid credential to the diverse local credentials and authentication/authorization mechanisms that apply at each site. Hence, users need not re-authenticate each time they access a new remote resource. GSI's PKI mechanisms require access to a private key that they use to sign requests. While in principle, a user's private key could be cached for use by user programs. This approach exposes this critical resource to considerable risk. Instead, GSI employs the user's private key to create a proxy credential, which serves as a new private-public key pair that allows a proxy to make remote requests on behalf of the user.

5.1.2 Monitoring and discovery service (MDS) [29]

MDS provides a framework for publishing and accessing information about grid resources by using the lightweight directory access protocol (LDAP) as a uniform interface to such information. MDS provides two types of directory services: namely the grid resource information service (GRIS) and the grid index information service (GIIS). A GRIS can answer queries about the resources of a particular grid node; examples include host identity (e.g., operating systems and versions), as well as more dynamic information such as current CPU load and memory availability. GIIS combines the information provided by a set of GRIS services managed by an organization giving a coherent system image that can be explored or searched by grid applications.

5.1.3 Grid Resource Access and Management (GRAM) [30]

This component is used to locate, submit, monitor, and cancel jobs on grid resources. It is set of services and clients for communicating with a range of different batch/cluster job schedulers using a common protocol. It is meant to address a range of jobs where reliable operation, stateful monitoring, credential management, and file staging are important. The components have dependency with GSI and GridFTP [31].

5.1.4 GridFTP

This protocol defines a general purpose mechanism for secure, reliable, high-performance data movement and used for efficiently transferring large volumes of data. It is based on the Internet FTP protocol and thus involves two communication channels: a control channel and a data channel. The commands and responses flow over the control channel, and the data is transmitted over the data channel. The GridFTP design supports secure authentication of control channel requests via the GSI, Kerberos, or an SSH security mechanism.

5.2 Open Grid Service Architecture-Data Access Integration (OGSA-DAI)

In service-oriented grid, the principal objective is to enable data resources and computational resources to be accessed and managed in a secure and systematic manner [32]. OGSA-DAI [33] is a service based architecture for database access over the grid and allow consumers to discover the properties of structured data stores and to access their contents and is extensible to accommodate different storage paradigms. OGSA-DAI provide a standard service based interface through which databases can be presented to OGSA grid applications. The service supports both the management of data and management of the computational resources used to store and process the data. The facilities include query and update, programming interface, indexing, high availability, recovery, replication, uniform access to data and schema, archiving and security. OGSA-DAI is designed to provide generic database access capabilities within grid and it is better resulted in providing enhanced functionality compared with other database middleware by supporting access to multiple database paradigms. Furthermore, it support flexible data de-livery, providing facilities for incremental and bulk delivery, to and from services or files in a synchronous or an asynchronous manner [33]. The Fig. 3 shows the overview of OGSA-DAI.

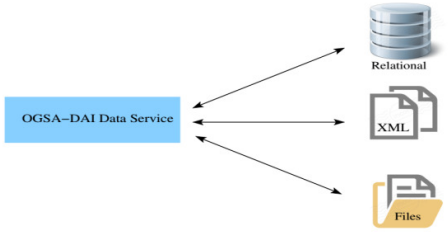

Fig.3: Overview of OGSA-DAI

6. PROPOSED MODEL

The main focus of this work is to propose a prototype model that combines the academic and research data of the educational institutes and universities to support the educational community. The proposed model is a grid enabled education based framework architecture that provides promising platform to support sharing of geographically dispersed learning content and it allows students, faculty and researchers to share and gain knowledge in their area of interest by using e-learning, searching and distributed repository services among universities from anywhere anytime. The proposed model makes use of the architecture components and services explained in earlier sections. The layered architecture of the proposed model is depicted in Fig. 4 consisting of six layers namely grid network layer, grid resource layer, resource system software, OGSA service layer, web service layer, application layer and software utilized in the respective layers.

The application layer supports for implementing content retrieval and e-learning services. The grid middleware environment holds web service layer, service architecture layer and middleware components. These are in-charge of dispatching application's information as stateful service to the underlying layers. Communication service of OGSA layer employs the globus components for respective operations. Resource system software and grid re-source layer represents the hosting environment, resource layer deals with computing devices like high-performance computer, file server and compute cluster. The communication protocol of system software enables data exchange between the end resources and grid middleware. The layered architecture of proposed model eases the implementation process and improves the prospective of component reuse.

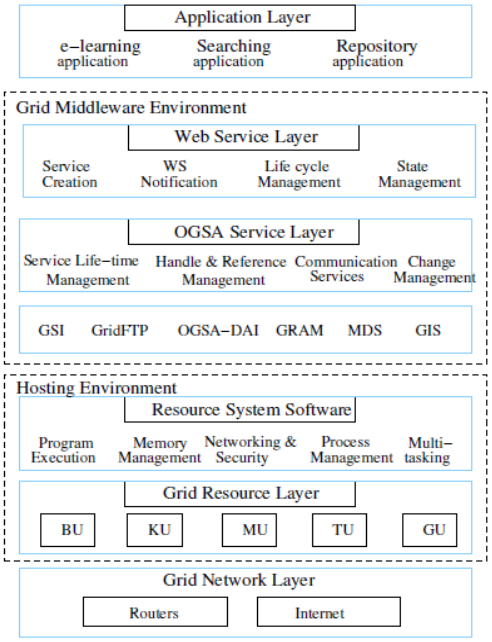

Fig.4: Layered architecture of proposed model

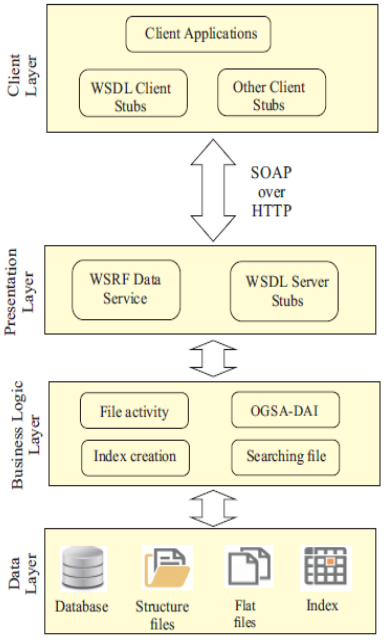

Fig.5: High-level Framework Architecture

6.1 High-level Framework Architecture

The high level architecture in Fig. 5 describes the actions and operations at different levels including client level, presentation level, business logic level and data level.

1) **Client Layer:** This layer provides client application (normally web browsers) that allows end users to request for available services from grid system. It enables the flexibility for end user to access services from any computing devices.
2) **Presentation Layer:** This layer provides graphical user interface that allows user to understand and interact with the things easily. It accepts the necessary inputs from user to process. The main function of this layer is to translate the tasks and results from the neighbour layers so that the users can understand.

Fig.6: Data flow design of upload operation

3) **Business Logic Layer:** This layer is composed of searching, e-learning and repository services. This layer holds the logic of coordinating the application process tasks, brings out logical decisions, assesses and performs the computation. The layer also carries and processes the data between surrounding layers.
4) **Data Layer:** This lies at the underneath of the frame-work, consisting of data resources that deal with all kinds of data formats. OGSA-DAI enable the support of uniform access or store to other data service resources.

6.2 Operations

This section describes the individual operations of proposed work.

6.2.1 Uploading

This operation allows the authorized users to upload the verified and validated document. The user has to provide details of uploading document such as domain, title, authors, keywords, etc. Then, the document is stored in the respective local repositories and local index is constructed for the stored data along with required meta information. The *deliverFromGFTP* activity is invoked by the remote server to update the global index and this is repeated when local indexes are added with new entry or any new update on previously stored document. The step-by-step procedure of upload operation is illustrated in Fig. 6.

6.2.2 Indexing

In order to speed up the searching mechanism, the idea is to use a centralized index, which is generated by reorganizing the existing documents based on the bottom up approach. The local index at the bottom level maintains the individual repository information of the documents stored in different universities. The global index at top level is constructed based on the information (metadata) of local indexes of all repositories. The indexing structure stores the metadata and structural information which increases the efficiency and precision of searching. In the case of a new entry in any of the local indexes, the global index is updated. Even when the global index is crashed due to some reasons, it can be reconstructed from the local indexes and vice versa. In such cases, the system should compromise with cost of recreating index and index creation time. The Fig. 7 shows the working architecture of index creation and search operation along with the interaction and inter-operation between the global and local indexes.

The most significant digit radix sort [34] is used to cluster the similar keywords based on most significant letter (alphabet). This mechanism partition the main index file `G` into `Z` pieces and groups the elements with same alphabet into a cluster. The process is carried out recursively to arrange the strings from left to right in each cluster and finally all clusters are arranged in order.
The counting sort [35] is adopted to count the number of objects stored in each cluster and one

auxiliary array `AUX[i]` which is maintained to hold each cluster range (i.e. integer values) dynamically. An array Z of n most significant elements is taken from the keywords in the range `[1, 2,..., x]`. For each element '`i`' of input array `Z`, increment `AUX[i]` by 1.

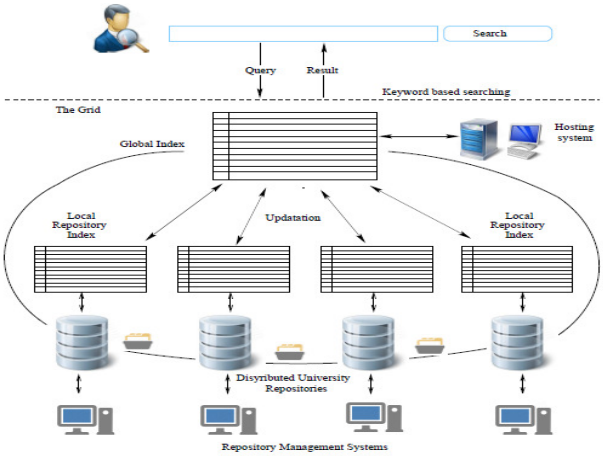

Fig.7: Overview of index creation and searching operation

The process is iterated for n elements of each cluster array with time complexity of O(n) and updates AUX. The index j values of AUX presents the number of times that j appears in Z. Next, insert each j with the total of AUX[j] times in the new AUX' list as presented in Algorithm 1. This adds up the computation with the complexity of O(x). Thus, the overall computation time of counting sort is O(n+x).

Algorithm 1: Count Sort

```
1  for each i ← 1 to x do
2      AUX[i] ← 0;
3  for each j ← 1 to n do
4      AUX[Z[j]] ← AUX[Z[j]] + 1;
5  for each i ← 2 to x do
6      AUX[i] ← AUX[i] + AUX[i-1];
7  for each j ← 1 to n do
8      C[AUX[Z[j]] ← Z[j];
       AUX[Z[j]] ← AUX[Z[j]]-1;
```

Vector space model along with term frequency (tf) and inverted document frequency (idf) method is broadly used in keyword based information retrieval which effectively supports multi-keyword search. tf is the number of times a given keyword appears within document and idf is obtained through dividing cardinality of documents by the number of documents containing the keyword. In vector space model, each document is represented as a vector, whose elements are the normalized tf values of keywords in the document. For each keyword k appearing in the document d, W_{ki} is a weight representing the dominance of the keyword k in the document d and is given by:

$$W_{ki} = \frac{tf(k_i, d)}{MaxFreq(d)} . \log_{10} \frac{|N|}{n_k} \quad (1)$$

where $tf(k_i,d)$ is the frequency of the keyword ki occurred in document d, MaxFreq(d) is the maximum frequency appeared in the collection of documents, |N| is the total number of documents and n_k is the number of documents containing keyword k_i.

The procedure of sorting of index G by most significant digit for the extracted keywords K = (k_1, k_2, . . ., k_n) from the document is as follows,

1) Identify the MSD of each k_i.
2) Sort the K based on the first digit of each keyword using count sort.
3) Grouping elements with the same digits into a cluster C_i where i = 1, 2, . . . n
4) Concatenate the clusters C_1, C_2, . . .,C_n together in order.

After sorting all the extracted keywords from the main index, each keyword placed in respective cluster in the form of (k||fid||wk), where k is the keyword, fid is the file identifier of the keyword and w_k is the weight calculated by tf and idf values.

6.2.3 Searching

Searching module executes the user query in the grid node (i.e virtual organization) where global index is constructed and returns the results relevant to the user query. This mechanism of search operation provides centralized search execution that prevents the distributed query execution at each node and time required to execute each query.

Each virtual organization is set up with middleware and associated modules that construct local index and shall update in global index. The resource management module of each virtual organization determines the resources that perform data source operations at run time.

Vector similarity function is used to measure the similarity between queries. Vector similarity shows each query as a vector in vector space model on query values. The symmetric matrix is constituted for the queries that are to be compared. Then, the cosine similarity between upper or lower triangular matrix query elements are calculated and the equation is given by:

$$Sim\ (q_i, q_j) = \frac{\sum_{i,j=1\ k\in(K_i\cap K_j)}^{m,n} q_{i,k} \times q_{j,k}}{\sqrt{\sum_{i=1\ k\in(K_i\cap K_j)}^{m} q_{i,k}^2} \times \sqrt{\sum_{j=1\ k\in(K_i\cap K_j)}^{n} q_{j,k}^2}} \quad (2)$$

where K_i is a set of keywords of query q_i and $K_i \cap K_j$ is set of commonly searched keywords by both queries q_i and q_j. For better understanding, the above equation is considered for two successive queries. Therefore, the vector similarity and cosine angle between two queries q_1 and q_2 is given by:

$$Sim\ (q_1, q_2) = \frac{\sum_{k\in(K_1\cap K_1)} q_1 \times q_2}{\sqrt{\sum_{k\in(K_1\cap K_1)} q_1^2} \times \sqrt{\sum_{k\in(K_1\cap K_1)} q_2^2}} \quad (3)$$

If the similarity between two or n successive queries are found similar or relatively similar then search module fetches the information from data cache segment. This reduces the query execution time. Otherwise, keyword matching proceeds in global index and retrieves the information of documents that are matches the keywords.

The notion of ranking the documents over total retrieved documents provides better and accurate list of results. In this work, retrieved documents are ranked based on (1) number of query keywords occurred in document, (2) keywords scores in the document, and (3) maximum the query keywords occurred in the document higher the ranking order. Based on above criteria, we define a collection of documents in terms of D_{all} and D_{rel} where D_{all} is the set of all documents and D_{rel} is set of documents relative to the domain. Let d' be the document belongs to D_{rel}. Consider a query of n keywords Q = $\{k_1, k_2, \ldots k_n\}$ and $D_{qk} = \{d | d \in D_{all} \wedge \forall k \in Q\}$ a set of documents that contains all the query keywords. The result of the query is defined as:

$$Q_{res} = \{d | \forall k \in Q \; \exists d \in D_{all}((d, d') \in D_{rel} \wedge d' \notin D_{qk} \wedge \text{contain}(d', k) Q\}$$

Thus, the result Q_{res} contains the set of documents that has at leat one occurrence of document including all of the query keywords. The function for the number documents retrieved is given by:

$$N_d = f(D_x) = \sum_{i=1}^{|V|} D_{xi} \tag{4}$$

Where D_x states documents containing of n query keywords, |V| is the number documents in the index consist of x query keywords. The recursive function that returns documents containing any number of query keywords is given by:

$$F(x) = f(D(x)) + F(x-1) \tag{5}$$

The total number of documents denoted as D_{tot} containing all and any number of query keywords is calculated as:

$$D_{tot} = D_x + D_{x-1} + \cdots + D_{x-(x-1)} = \sum_{i=x-(x-1)}^{x} D_i \tag{6}$$

These documents employ ranking function that deals with w_{ki} scores of individual document with respective keyword in the index and cosine similarity between documents associated with number of keywords occurred in it (according similarity equation). The ranking of document d_i to a user query Q is given by:

$$Rank(d_i, q) = \{rank(d_{xi}, k_j), rank(d_{(x-1)i}, k_j), \cdots rank(d_{x-(x-1)i}, k_j)\} \tag{7}$$

Where $Rank(d_i, q)$ is the set of documents arranged in the order of maximum to minimum occurrence of query keywords. The ranking of individual document in the respective set is calculated by:

$$rank(d_{xi}, k_j) = \left[\sum_{i=1, j=1}^{p,q} w_{xi}(d_{xi}, k_j) \right] \times Sim(d_{xi}, d_{xi+1}) \tag{8}$$

Where p is number of query keywords and q is number of documents in d_x list. The overall ranking is the sum of w_{ki} score of each query keyword with respect to document multiplied by a measure of cosine similarity between documents belongs to respective sets of number of keywords occurrence. The procedure of searching and ranking of documents is given in Algorithm 2

6.2.4 Downloading

The user can opt for download or view option provided in the user interface. Download operation initiate the task execution of file retrieval by referring the information stored in database like IP/distinguished name of node, port, port type, etc. GridFTP protocol is used to transfer the file by coordinating with OGSA-DAI component of that grid node.

Algorithm 2: Document searching and ranking

```
1   q_v  -  Vector of the query q
2   r    -  The number of documents to be returned
3   G    -  Global index of documents
4   for each incoming query q_i do
5       compute Sim (q_i, q_j);
6       if any found similar then
7           return r;
8       for each keyword k_i of query qi do
9           look up into the cluster based on MSD in G
10          list(D_x ⋯ D_x−(x−1))
11          Compute overall rank Rank(d_i, q)
12          for each D_x to D_x−(x−1) do
13              for each d_i in the set do
14                  Compute rank(d_xi, k_j)
15  return r
```

6.3 Use of OGSA-DAI

The key characteristic of OGSA-DAI is the integration of data resources that enables to access relational databases and XML repositories managed in secure and systematic way.

OGSA-DAI extends its support to service based infrastructure for accessing databases and repositories through web services platform i.e via WSRF included in GTK middleware. It also facilitates Java based APIs and CLIs to use and test the functionalities on the platform. Java based APIs are used to interact with data operations of OGSA-DAI and web service operations as well. For each task execution, OGSA-DAI construct an activity corresponding to the task type that needs to be performed. The varieties of activities are defined by OGSA-DAI to deal with different data sources. In this work relational, deliver and file activities are used for the operations. In order to update the global index, the relational database activity is performed from the local index where the document is uploaded. For every update, the local indexes are updated in global index performed by relational database activity.

The delivery activity of OGSA-DAI defines a protocol with corresponding port type for transfer of data such as grid data transport (GDT), Grid file transport protocol (GridFTP), and unified resource locator (URL). Any of the protocol can be used for transfer operation and that supports the grid data service (GDS). These activities hold the ability to deliver data to and from a GDS. For this work, the GridFTP protocol is adopted for the transfer of file when user opt for download or transfer of data and internal operations of grid. The GridFTP consist of two activity types: *deliverFromGridFTP* that receives data from external location and *deliverToGridFTP* that deliver the data to external location. The GDS performs the insert and update operations for indexes of respective repository attached to institute or university using the activities of OGSA-DAI.

It is learnt that the complexity of decentralized searching methods in distributed data sources increases with in-crease in grid nodes. For information searching, distributing number of queries to all distributed indexes and join joining them may lead to unnecessary resource utilization and consume more time. Instead, a centralized application that retrieves the information from centralized index saves time and resources. Other grid resources are utilized in the case of data transfer and e-learning applications.

6.4 E-Learning

To provide e-learning contents in collaborative and distributed fashion using the grid is component of this work. The proposed framework architecture model manage and support e-learning services like curriculum activities, virtual classes, tests, presentations, and educational games (puzzles, quiz, etc.). Furthermore, the model provides multimedia data in respective area of subject and research. To manage and use these learning services, users are distinguished by their roles like administrator and user learner. The administrator has all privileges to supervise and handle e-learning services by deploying/hosting new ones and removing. The respective hosted service provides synchronous and asynchronous facilities for collaboration and discussions. Synchronous holds virtual classes, chats etc. and asynchronous is responsible for emails, forums etc. The GTK environment utilizes its components and orchestrates computing resource to execute these applications on gird.

7. EXPERIMENTAL RESULTS

7.1 Experimental Environments

The experimental environment is set up using desktop machines and laptops with high end dedicated server. The details are given in Table I.

Host	CPU Type	Processor Cores	Clock (Ghz)	Memory (GB)	Storage (TB)	NIC (Gbps)	Linux Kernel
IBMServer	Intel Xeon X5670	12	2.93	32	4	1	3.19.0-25
HpEliet01	Core i7-3770	8	3.4	4	1	1	3.19.0-25
HpEliet02	Core i7-3770	8	3.4	4	1	1	3.19.0-25
HpProbook01	Core i7-4702	8	3.4	4	1	1	3.13.0-48
HpProbook02	Core i7-4702	8	3.4	4	1	1	3.13.0-48

TABLE I: Specification of hardware resources on the test-bed

A grid test bed is established to experiment grid operations using Globus toolkit 5.2.5 and OGSA web services operations to operate through Internet. Each machine in the test bed consists of GTK with GRAM, GridFTP and GSI components with Ubuntu 12.04 Lts OS. All machines are connected via local area network and assigned static IP addresses. The proposed services are developed as grid services according to the OGSA using the libraries of GTK. The services of the proposed framework are implemented as stateful services and meet the standards of WSRF and communicated each other through well defined interfaces. The essential necessity of defined services should be compatible with other components of GTK. The components of the framework are developed using the libraries provided by Globus Alliance. The user interfaces are designed for end-users that provide a web-based interface and allows users interact with the grid application via GUI. The main portals are admin portal and user portal. Admin portal allow authorised user to operate core level grid operations. User portal provide searching and e-learning service for all community.

7.2 Results

To verify the proposed work, the framework is implemented and evaluated in a lab level of grid test bed. To begin with, experimental results of keyword based document retrieval from connected system repositories are conducted and the results are discussed.

7.2.1 Efficiency

The response time metric is to measure the efficiency of searching the document(s) or time taken to process each query. The experiment is conducted in two phases. In the first phase, response time is measured for each single query passed for searching in global index. In the second phase, average response time is measured for every batch of queries over a time period. Fig. 8 represents the response time for first 30 randomly generated queries. The response time varies according to keyword length or numbers of keywords in the query. The query number 6 and 15 showed with less response time because of repeated or identical keywords by successive queries. The average response time of search method is 0.840 seconds for these first 30 queries including different queries lengths. The results are measured for average response time against the number of queries a over period of time as illustrated in Fig 9. In addition, the Fig. 10 shows that the average response time for single keyword queries and multiple keyword queries are almost linear with the size of the document collection.

On the other hand, the storage complexity of the each cluster sorted out from the main index is O(R×C), where R indicates the number of row in the index and C indicates the number of columns in the index (i.e C = 3). At the initial test, extracted R = 1246 keywords for 10 files and C = 3, R×C = 1246 3 = 3738 elements that would consume 13.3kb storage space. The storage space consumption for index is proportional to the number of keywords in the dictionary.

7.2.2 Accuracy

In the experiment, we analyze the performance of information retrieval system in terms of accuracy. The accuracy of fetched documents is measured by its precision and recall.
Let,

R_{rt}	-	number of relevant documents retrieved
R_{nrt}	-	number of relevant documents not retrieved
NR_{rt}	-	number of non-relevant documents retrieved
NR_{nrt}	-	number of non-relevant documents not retrieved

Thus, the total number of retrieved documents (T_{rt}) includes: $T_{rt} = R_{rt} + NR_{rt}$
and total number of relevant documents (T_{rl}) includes: $T_{rl} = R_{rt} + R_{nrt}$
using T_{rt} and T_{rl} the precision and recall are defined respectively as follows:

$$Precision = \frac{R_{rt}}{T_{rt}} \tag{9}$$

$$Recall = \frac{R_{rt}}{T_{rl}} \tag{10}$$

The performance of search method in terms of recall function is evaluated as shown in Fig. 11. By considering the numbers of relevant documents corresponding to some keywords are measured before applying the recall function and then it is employed on the retrieved result.

In Fig. 12, the accuracy is measured in terms of precision function parameters against the number of keywords embedded in query. The mean average precision (MAP) is used for these first 30 random queries and is defined as:

$$MAP = \sum_{i=1}^{N} \frac{\left(\sum_{j=1}^{n} \frac{Prec(q_i)}{n}\right)_i}{N} \tag{11}$$

Where Prec(q_i) is the precision value of query q_i, n is the number queries and N is the number of average precisions. Hence, it is observed that the proposed approach attains good results in precision and recall.

7.2.3 Resource Utilization

In the proposed approach, searching, uploading and e-learning are services that utilize resources effectively. Each of the services require resources to execute its tasks, sending queries to search in local indexes of each institute's repository that demands the resource from that particular grid node to execute and return the retrieved results. Performing local search increases the response time of overall retrieval of result and resource utilization rate.

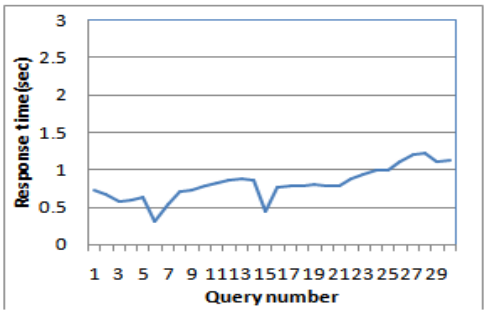

Fig. 8: The response time for 30 queries

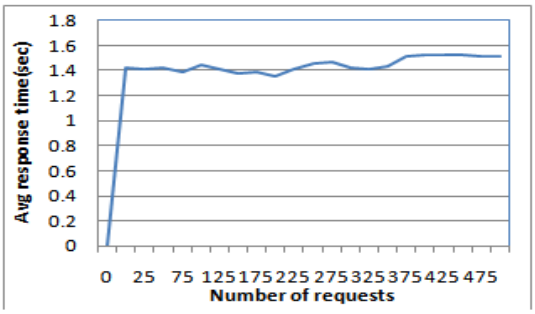

Fig. 9: Average response time for number of queries

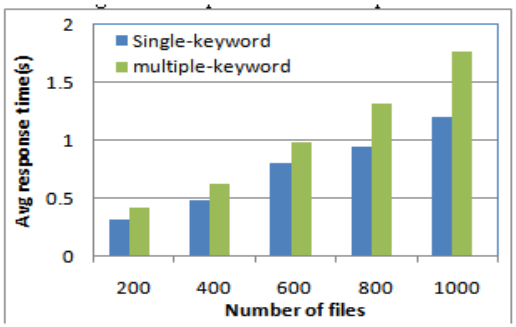

Fig. 10: Average response time for single and multiple keywords with increasing number of files

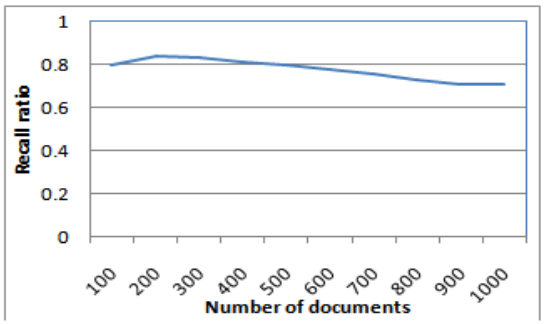

Fig. 11: Recall: Similarity ratio

Fig. 13 presents the comparison of resource utilization rate for centralized and decentralized searching techniques. Decentralized utilization rate attains almost equal in all sites, because this method keeps other site resource busy with execution of queries distributed by main server site. Whereas centralized method keep its only resource active for searching operation, so its utilization at server site is more than other sites.

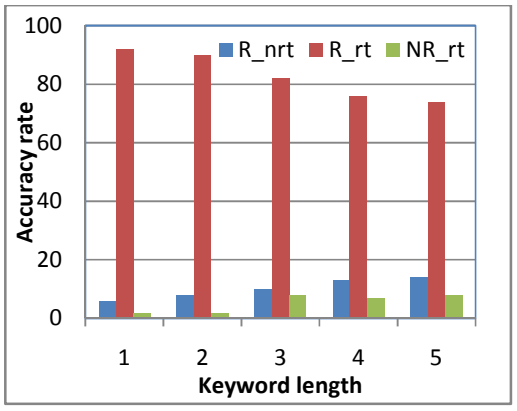

Fig. 12: Accuracy of the searching method

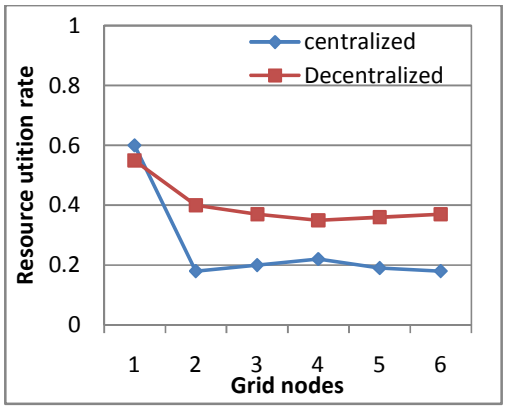

Fig. 13: Resource utilization: centralized and decentralized method

7.3 Discussions

After the demonstration of experiments, we indicate the implications of the results. The purpose of the experimental design is to build a framework architecture that facilitate the grid services for educational community and utilizes resources efficiently. Due to its hierarchical index creation of storing document, the cost of index creation at that particular time period is lesser. The two level hierarchies are maintained at the index for various domains results in faster query execution. Furthermore, it is observed that searching function attains better results by retrieving relevant documents among the many files of different repositories.

As explained in Section 6.2, the experiment is conducted by hypothesising the fresh documents to be stored on empty repository and indexed. The query and task execution is dependent on CPU

speed and main memory size. So, the global/centralized index is deployed in a site with more powerful computer system to deal with maximum number of requests. The security is maintained to preserve data securely in distributed repositories. This concept makes framework to utilize grid resource efficiently.

8. CONCLUSION

The grid technology enhances e-learning in new and potential ways. It provides the support for adopting the web services to build grid services. The proposed work highlighted the issues of utilizing the grid resources for education system or community. The framework architecture focused on the services that allow education community to easy search and share access to educational contents of different institutes/universities. The system enables the end user to access the services through a web portal by any devices. The framework architecture uses Globus toolkit to construct the grid environment and WSDL, WSRF, and OGSA-DAI for service interoperability.

The architecture is engineered in layers of functionalities; the services coordinate and aggregate the functionality from services in successive layers. The operation level services provide explicit task execution capabilities and fine-grained model with dynamic interaction and inter-operation. The management level services provides their extendibility to manage the pre-configured operation control, a coarse grained model to control over tasks with admission control and a failure handling mechanism. This system would serve as a prototype for developing education and knowledge based application for universities across number of disciplines. Furthermore, the prototype can be focused and categorized to specific domains such as engineering, science, and business management, etc. would lead to community in the collaboration to investigate data and acquire more precise results about specific domain from many repositories located over different locations.

8.1 Future Work

A number of possible future extensions to the framework have been identified and are under investigation.

Data management: The framework architecture is envisioned to be complemented with service based; middle-ware and data management consideration that constitutes on top of mechanism such as OGSA-DAI, GridFTP and GRAM integrates consistently with proposed services and clients. Support for multiple documents from multiple sites data management of index would need to be provided by implementing service customization in constructing all indexes. For large data files, the system needs to consider effective keyword extraction method which reduces the complexity as well as space.

Resource management: The system that has powerful resource capacity exposing the centralized index for clients by offering fast response with relevant information. Even though admission control service insures the maximum requests, the framework architecture lacks in support of resource at that particular site. Interesting research question include, investigation of employing resource for attaining the maximum requests with advanced task management mechanism.

Furthermore, the system should adopt the replica management method to assurance of fault tolerance for data produced at multiple sites and this can be speed up the content retrieval and loading process. This approach can be extended for multiple domains as well as multiple

Universities. The user will have enhanced GUI's for better choice and experience. Adaption of the framework architecture to new technologies to facilitate new optimized services is also of interest.

ACKNOWLEDGEMENTS

This research work is fully supported by the SERB - DST New Delhi, under grant reference number: NO SB/EMEQ-250/2013.

REFERENCES

[1] K. Karasavvas, M. Antonioletti, M. Atkinson, N. C. Hong, T. Sugden, A. Hume, M. Jackson, A. Krause, and C. Palansuriya, "Introduction to ogsa-dai services," in Scientific applications of grid computing, pp. 1–12, Springer, 2005.

[2] K. Gottschalk, S. Graham, H. Kreger, and J. Snell, "Introduction to web services architecture," Journal of IBM Syst., vol. 41, no. 2, pp. 170–177, 2002.

[3] S. Lynden, A. Mukherjee, A. C. Hume, A. A. Fernandes, N. W. Paton, R. Sakellariou, and P. Watson, "The design and implementation of ogsadqp: A service-based distributed query processor," Future Generation Computer Systems, vol. 25, no. 3, pp. 224–236, 2009.

[4] M. ZadahmadJafarlou, M. Karimi, et al., "A dependable and economic service for long-life e-learning applications in grid environments," Procedia-Social and Behavioral Sciences, vol. 28, pp. 784–789, 2011.

[5] S. M. D. Kumar and A. Jumnal, "A real-time grid enabled test bed for sharing and searching documents among universities," in 2nd IEEE International Conference on Advances in computing and communication & Engineering (ICACCE), May 2015.

[6] G. Project, http://toolkit.globus.org/,

[7] G. Alliance, http://toolkit.globus.org/alliance/,

[8] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," Globus Project, 2004.

[9] I. Foster, J. Frey, S. Graham, S. Tuecke, K. Czajkowski, D. Ferguson, F. Leymann, M. Nally, T. Storey, W. Vambenepe, et al., "Modeling stateful resources with web services," Globus Alliance, 2004.

[10] E. Commission, http://www.datagrid.cnr.it/,

[11] E. Commission, http://www.eurogrid.org/,

[12] E. Commission, http://www.gridlab.org/,

[13] T. Hey and A. E. Trefethen, "The uk e-science core programme and the grid," Proc. of the Future Generation Computer Systems, vol. 18, pp. 1017–1031, Oct 2002.

[14] N. S. F. (NSF), http://www.teragrid.org/,

[15] N. S. F. (NSF), http://www.grids-center.org/,

[16] N. S. F. (NSF), http://nhse2.cs.rice.edu/grads/,

[17] N. S. F. (NSF), http://www.ivdgl.org/,

[18] K. Prasad, H. Gupta, N. Mangala, C. Subrata, H. Deepika, and P. Rao, "Challenges of monitoring tool for operational indian national grid garuda," in Proc. of the National Conf. on Parallel Computing Technologies (PARCOMPTECH), pp. 1–5, Feb 2013.

[19] S. F. for Bioinformatics and I. D. Computational Biology,http://www.scfbio-iitd.res.in/biogrid/biogrid.htm,

[20] M. Gaeta, P. Ritrovato, and S. Salerno, "Elegi: The european learning grid infrastructure," in 3rd International LeGE-WG Workshop: GRID Infrastructure to Support Future Technology Enhanced Learning, Berlin, Germany. 3 December, 2003, p. 9, 2003.

[21] L. Qinghu, W. Jianmin, K. Y. Lam, and S. Jiaguang, "Gridfs: A webbased data grid for the distributed sharing of educational resource files," in Advances in Web-Based Learning-ICWL 2003, pp. 81–92, Springer, 2003.

[22] V. Alexandrov, N. Alexandrov, I. Bhana, and D. Johnson, "The elearning grid: Peer-to-peer approach," in EduTech Computer-Aided Design Meets Computer-Aided Learning, pp. 133–142, Springer, 2004.

[23] M. T. Su, C. S. Wong, C. F. Soo, C. T. Ooi, and S. L. Sow, "Serviceoriented e-learning system," in First IEEE International Symposium on Information Technologies and Applications in Education (ISITAE), pp. 6–11, IEEE, 2007.

[24] N. Jacobs, A. Thomas, and A. McGregor, "Institutional repositories in the uk: the jisc approach," Library Trends, vol. 57, no. 2, pp. 124–141, 2008.

[25] S. Roy, A. De Sarkar, and N. Mukherjee, "An agent based e-learning framework for grid environment," in E-Learning Paradigms and Applications, pp. 121–144, Springer, 2014.

[26] A. Trotman, "Choosing document structure weights," Information processing & management, vol. 41, no. 2, pp. 243–264, 2005.
[27] I. Foster, "Globus toolkit version 4: Software for service-oriented systems," in Network and parallel computing, pp. 2–13, Springer, 2005.
[28] B. Lang, I. Foster, F. Siebenlist, R. Ananthakrishnan, and T. Freeman, "A multipolicy authorization framework for grid security," in Fifth IEEE International Symposium on Network Computing and Applications (NCA), pp. 269–272, IEEE, 2006.
[29] X. Zhang and J. M. Schopf, "Performance analysis of the globus toolkit monitoring and discovery service, mds2," in IEEE International Conference on Performance, Computing, and Communications, pp. 843–849, IEEE, 2004.
[30] M. Feller, I. Foster, and S. Martin, "Gt4 gram: a functionality and performance study," in TeraGrid Conference, 2007.
[31] G. Khanna, U. Catalyurek, T. Kurc, R. Kettimuthu, P. Sadayappan, and J. Saltz, "A dynamic scheduling approach for coordinated wide-area data transfers using gridftp," in IEEE International Symposium on Parallel and Distributed Processing (IPDPS), pp. 1–12, IEEE, 2008.
[32] S. Lynden, A. Mukherjee, A. C. Hume, A. A. Fernandes, N. W. Paton, R. Sakellariou, and P. Watson, "The design and implementation of ogsadqp: A service-based distributed query processor," Future Generation Computer Systems, vol. 25, no. 3, pp. 224–236, 2009.
[33] M. Antonioletti, M. Atkinson, R. Baxter, A. Borley, N. P. Chue Hong, B. Collins, N. Hardman, A. C. Hume, A. Knox, M. Jackson, et al., "The design and implementation of grid database services in ogsa-dai," Concurrency and Computation: Practice and Experience, vol. 17, no. 24, pp. 357–376, 2005.
[34] B. A. Wagar, "System for msd radix sort bin storage management," Aug. 8 1995. US Patent 5,440,734.
[35] S. Ruggieri, "Efficient c4. 5 [classification algorithm]," Knowledge and Data Engineering, IEEE Transactions on, vol. 14, no. 2, pp. 438–444, 2002.

AUTHORS

Anand Jumnal received the BE and M.Tech degree in Computer Science and Engineering, from Visvesvaraya Technological University, India in 2010 and 2013 respectively. He is currently a Research Scholar in UVCE, Bangalore University. His research interest includes grid computing, clou d computing, resource scheduling and allocation.

Dilip Kumar S M received the B. E degree in 1996 and the M. Tech degree in 2001 from Kuvempu University and Visvesvaraya Technological University respectively. He obtained the Ph. D degree from Kuvempu University in April 2010. All the three degrees are in Computer Science and Engineering discipline. Currently working as Associate Professor in the Department of Computer Science & Engineering, UVCE, Bangalore University, Bangalore. He is involved in research and teaching and has more than 19 years of teaching experience and guiding eight Ph. D students. He has published over 30 papers in International Journals and Conferences. Principal Investigator for a research project of Rs. 18.31 Lakhs sponsored by Department of Science and Technology, Govt. of India is ongoing and handling two consultant projects. His current research lies in the areas of Grid and Cloud computing.