http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 8 (1): 71-76, 2009
ISSN 1812-5638

Data Discovery in Grid Using Content Based Searching Technique

R. Renuga and Sudha Sadasivam
PSG College of Technology, Coimbatore, Tamilnadu, India

Abstract: The aim of this study is data discovery in data grid using content based search technique. This study models scientific data grid as a large peer to peer based distributed system model. Content based discovery mechanisms are applied for data discovery using this model. Grids tie together distributed storage systems and execution platforms into globally accessible resources. Data grid deals with large computational problems by providing geographically distributed resources for large-scale data-intensive applications that generate large data sets. Data grids provide collection management and global namespaces for organizing data objects that reside within a grid. This proposed mechanism has been investigated using the grid simulator gridsim.

Key words: Data grid, co-occurrence matrix, dynamic ranking, gridsim

INTRODUCTION

Grid is a form of distributed computing platform that couples geographically distributed resources. It helps in solving large-scale problems. It enables sharing, selection and aggregation of suitable computational and data resources for solving large-scale data intensive problems in science, engineering and commerce. Application schedulers in the Grid environment, termed resource brokers, perform resource discovery, selection and aggregation of a diverse set of distributed resources for an individual user. Each user in a grid environment is provided with a private resource broker. This resource broker is targeted to optimize for the requirements and objectives of its owner.

Grids are middleware that tie together storage systems and execution platforms that reside in multiple autonomous administration domains. The middleware provides a common infrastructure for creating global properties that span separate heterogeneous, independent resources. The common infrastructure is used to support single sign-on authentication environment, uniform job submission mechanisms, uniform naming conventions for grid resources and uniform scheduling systems (Foster and Kesselman, 1999). Grids therefore serve as interoperability mechanisms for turning remote, heterogeneous resources into a globally accessible system. Data grids build a data management infrastructure on top of grid environments to support potential access to billions of digital objects. A major capability of data grids is support for data discovery.

An information repository is an organized collection of information (Foster and Kesselman, 1999). Discovery of data objects can then be accomplished by querying the information repository. Since collections are typically organized around domain-specific topics, data grids must support access to a large number of information repositories. Data object discovery then involves identifying the relevant collection, specifying desired values for the collection-specific attributes, issuing a query against the collection and finally retrieving the required data object (Buntine *et al*., 2005).

For data discovery content based search is used. It is based on textual content written in natural language. It utilizes information available in the documents in a holistic manner to determine the user's interest. The documents searched are ranked according to their relevance with respect to the query. The design criteria taken into account are usability, robustness, predictability, scalability and transparency. The content used for the aforementioned purpose is obtained by extraction of description from the meta-data. This extracted data is used for the creation of the co-occurrence matrix which forms the basis for ranking. The co-occurrence matrix gives the number of times two tokens co-occur in the corpus. This approach considers all words in the formation of co-occurrence matrix except for those in the stop list.

Redundancy is taken into account while scoring. This helps to avoid unjustified conclusions. This study presents a method of dynamic ranking. Dynamic ranking is done at the query time rather than at the indexing time. Thus each time a query is fed in by the user, the ranking changes based on the query.

DATA DISCOVERY METHODOLOGY

Content based searching (Moore *et al*., 1997; Ponte and Croft, 1998) is proposed in this study for data

Corresponding Author: R. Renuga, PSG College of Technology, Coimbatore, Tamilnadu, India

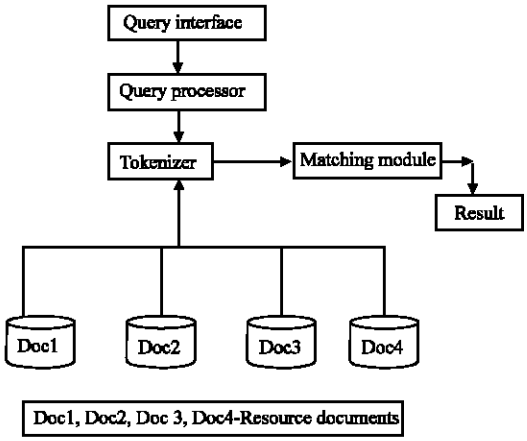

Fig. 1: Data discovery architecture

discovery. This study focuses on textual content that is written in natural language (Fig. 1). This proposed technique is used to retrieve and index documents (Oumis *et al*., 2006). It also provides mechanism to return a ranked subset of the documents according to the user's request (Page *et al*., 1998). The basis for searching and ranking is the co-occurrence matrix (Ville, 2007).

Co-occurrence matrix: The co-occurrence matrix gives the number of documents in which the given pair of tokens co-occurs. The basic atom is a token. The tokenizer component determines a token. Tokenizer splits documents to tokens as a part of the preprocessing pipeline. Depending on its parameterization, it may allow digits, dashes, apostrophes, or other special characters to occur in tokens.

A document is considered as a finite sequence of tokens length N_d is given in Eq. 1:

$$d = (t_1, \ldots t_{Nd}) \quad (1)$$

A document may naturally contain multiple occurrences of the same token. If order is lost document, it gives a multiset:

$$D' = (t_1, \ldots, t_{Nd}) \quad (2)$$

Thus a bag of words representing a document, which is dominant in the information retrieval tradition. If token frequencies are ignored within a document, multiset D' is cast to an ordinary set D. The set of all D is called a corpus C. The set of all distinct tokens that occur in C is denoted by T. The tokens co-occur when two distinct tokens belong to the same bag of words. The co-occurrence matrix depicts the number of times any two tokens can co-occur in the corpus.

For each token $t \in T$, inverted set can be defined i.e., the set of all documents in which t occurs. Let I_t denote the inverted set for the token t, formally I_t is given by:

$$I_t = \{D \in C | t \in D\} \quad (3)$$

By definition, it is non-empty for each $t \in T$. Let m, n $\in T$ be two tokens. Their intersection is given by Eq. 4:

$$\Lambda = I_m \cap I_n \quad (4)$$

If Λ is non-empty, words m and n co-occur (i.e., they appear at least once in the same document). We can enumerate all possible token pairs in a T×T matrix Λ. Any element Λ_{ij} is given by:

$$\Lambda_{ij} = |I_i \cap I_j| \quad (5)$$

Thus Λ represents co-occurrence matrix. If token m co-occurs with token n, n co-occurs with m equally often. Thus the upper and lower triangular matrices contain the same information, thus making the co-occurrence matrix symmetric. Since a token always co-occurs with itself, the diagonal values correspond to the number of documents in which the token appears in the corpus C.

Technically, the co-occurrence matrix grows quadratically with respect to the number of tokens. Even with medium-scale corpora, its space requirements are enormous. So, the following measures are taken to reduce the size of the co-occurrence matrix:

- Storing only the upper triangular matrix as the co-occurrence matrix is symmetric
- Eliminating frequently occurring tokens which are presented for grammatical purpose but do not convey any meaning
- Distributing the files in the grid environment

The following empirical probabilities that used to score the document can be derived from the matrix.

The prior probability for seeing a token m in the corpus is:

$$P(m) = \frac{\sum_{n \in T} \Lambda_{mn}}{\sum_{l \in T} \sum_{n \in T} \Lambda_{nl}} \quad (6)$$

The conditional probability of locating token m given that token n is located:

$$P(m \mid n) = \frac{P(m, n)}{P(n)} = \frac{\Lambda_{mn}}{\sum_{l \in T} \Lambda_{nl}} \quad (7)$$

```
2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1
2 2 2 1 1 1 1 1 1 1 1 1 1 1 1
2 2 1 1 1 1 1 1 1 1 1 1 1 1
2 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 0
1 1 1 1 1 1 0
1 1 1 1 1 0
1 1 1 1 0
1 1 1 0
1 1 0
1 0
1
```

Fig. 2: Co-occurrence matrix

This probability will have a central role in the ranking scheme presented. Items of Λ are not convenient for visualization as they are unbounded sizes of intersections. So, co-occurrence data is visualized as P (m|n) distributions (Amati *et al*., 2002). But, in contrast to raw co-occurrence matrices the conditional distribution tables are not symmetrical, since in general P (m|n) ≠ P (n|m). The example of the co-occurrence matrix of a file with 29 tokens is shown in Fig. 2.

Ranking method: Let C be a corpus of documents. The user submits a query, q ∈ T to the system. It is required that the query token occurs in the corpus at least once. Assume that the query consists of only one token, q. The ranking method assigns a score, S_q (D) for each document D ∈ C. The system returns a ranked sequence of documents (D_1, D_2, ...,D_N) so that S_q (D_i) > S_q (D_{i+1}).

The score to a document is assigned as follows:

$$S'_q(D) = \sum_{t \in D} P(q \mid t) \tag{8}$$

The formulation of scoring of all D ∈ C in matrix form is described as follows. Let D be a C×T matrix containing the documents. A row from the T×T conditional distribution table corresponding to the query q is denoted by q.

The idea behind the above formulation is to give a high score to documents that contain either the query token q or many co-occurring tokens {t| Λ_{tq} ≠ ø, t ∈ T}. Since co-occurring tokens contain many synonyms, hypernyms and hyponyms of q, this scheme rewards relevant documents even though they would not contain any occurrences of the query token q.

Naturally the co-occurrences include non-relevant tokens as well. So, the relevant tokens should be weighted against the non-relevant ones with P (q|t). The idea is that for relevant tokens, Λ_{tq} should be large and Λ_{tt} (the frequency of token t) not be much larger than that. In other words, to gain a high weight, token t should appear only in the same document as q, i.e., preferably $I_t \subseteq I_q$. Very frequent words have large Λ_{tq} but also their Λ_{tt} is large and thus P (q|t) becomes small. In probabilistic terms, high weight is given to term t if seeing it makes seeing q in the same document probable. Correspondingly, a high score is given to document D if it contains many tokens related to q.

The scheme favors long documents. To overcome this, the description from the metadata is taken for this purpose. Generally the length of the description does not vary to a great extent. Then depending on the score computed for each of the documents they are ranked from highest score to the lowest score document.

SYSTEM DESIGN

The following sequence diagrams show the design of the system.

The system has two modules:

- Meta data extraction and construction of co-occurrence matrix
- Establishing grid environment and returning ranked list of documents

Module 1: This module is used for meta data extraction and construction of co-occurrence matrix. The functions of the components in this module and their interaction (Fig. 3) are given below:

Meta data: The meta data class takes care of obtaining an html document from the reserve and retrieves the metadata description which is used for the formation of co-occurrence matrix.

Tokenizer: The tokenizer class splits text files equivalent of the corresponding html files (considering the meta description) into tokens. Here, space is considered to be the delimiter for splitting. Corpus and co-occurrence matrix are updated accordingly.

Read file: The read file class reads the file and forms a bag of words. It also appends the corpus with words after eliminating redundancy. This corpus forms the basis for the co- occurrence matrix.

Module 2: This module is used for establishing grid environment and returning ranked list of documents. The functions of the components in this module and their interaction (Fig. 4.) are given below:

Data grid: The data grid class simulates the grid environment by creating resources, users, network and assigning values to all parameters of the various grid components.

File user: File user class creates users and assigns files to each of the resources. Then resource can be run in parallel.

Parallel execution: The class parallel execution runs all the resources in parallel and each of the resources scores the files allocated to it.

Scoring: The scoring class receives the query and calculates the score based on how many times the query as well as co-occurring tokens are present in the document. The weight for co-occurring tokens is based on in how many files they co-occur.

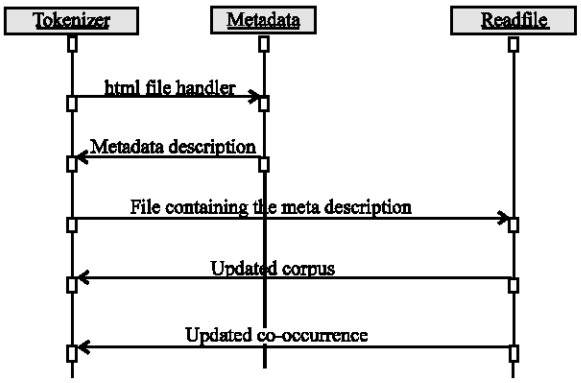

Fig. 3: Sequence diagram for module 1

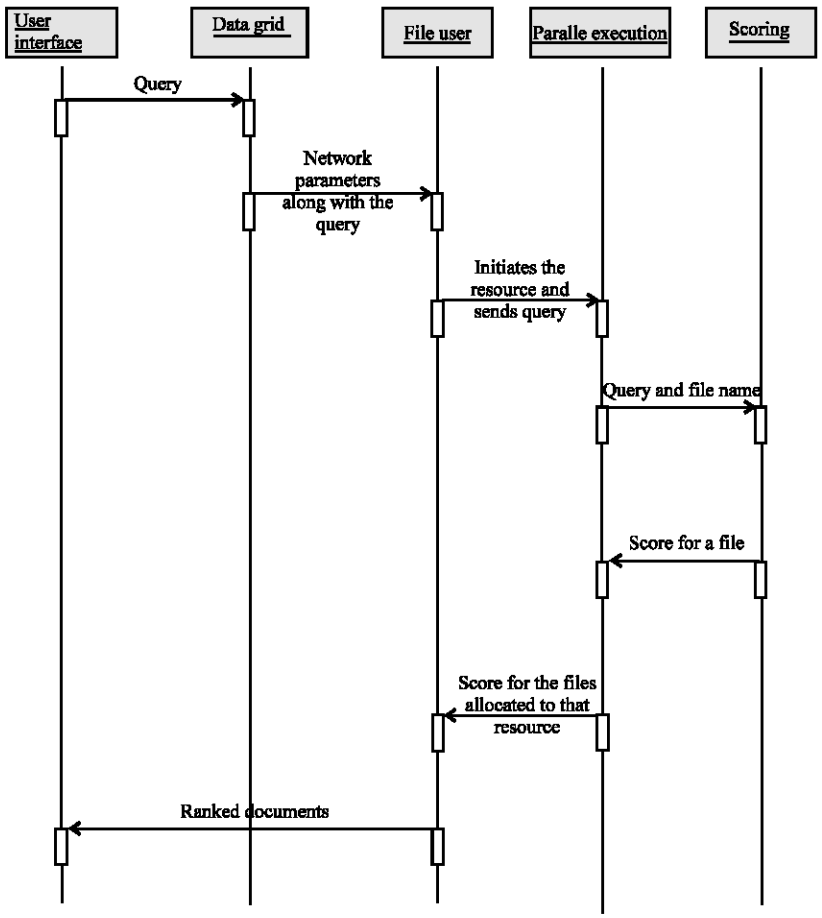

Fig. 4: Sequence diagram for module 2

EXPERIMENTAL RESULTS

The implementation of this study is done using Java, Net Beans (www.netbeans.org/download/flash/netbeans_55/nb_overview) and GridSim, a simulator package for grid environment. This application is tested in WINDOWS XP Platform working under Pentium IV, 1.4 GHz. Processors with 0.99 GB RAM. Grid Sim is used for simulating a simple framework for deterministic modeling and simulation of resources and applications.

The GridSim toolkit (www.buyya.com) supports modeling and simulation of a wide range of heterogeneous resources, such as single or multiprocessors, shared and distributed memory machines such as PCS, workstations, symmetric multiprocessors and clusters with different capabilities and configurations. It provides facilities for the modeling and simulation of resources and network connectivity with different capabilities, configurations and domains. It supports primitives for application composition, information services for resource discovery and interfaces for assigning application tasks to resources and managing their execution.

In case of scoring it is required to get the co-occurrences of the query with all the words in the document. While doing this only one line of the co-occurrence matrix is considered at a particular time. For multiple tokens the score for each of the query tokens is calculated and they are summed to give a final score which is considered for ranking. Threading concept is used in GridSim package to simulate the existence of multiple resources.

GridSim performs resource allocation, scoring and shutting down of all entities. The grid entities are first initialized. In resource allocation each file that is to be scored is allocated to a resource. Consider there are 6 files and 5 resources in the grid environment. The allocation of files to resources is shown below:

```
Initialising...
Starting GridSim version 4.0
Entities started.
File_User.addMaster(): /cbse_final/src/automata.htm1 has been added to Res_0
File_User.addMaster(): /cbse_final/src/checksum.html2 has been added to Res_1
File_User.addMaster(): /cbse_final/src/deshaw1.htm3 has been added to Res_2
File_User.addMaster(): /cbse_final/src/deshaw2.htm4 has been added to Res_3
File_User.addMaster(): /cbse_final/src/selectpoll.htm5 has been added to Res_4
File_User.addMaster(): /cbse_final/src/Poultry.html6 has been added to Res_0
Resource 0 started at 0 : 56 : 53
Resource 1 started at 0 : 56 : 53
Resource 2 started at 0 : 56 : 53
```

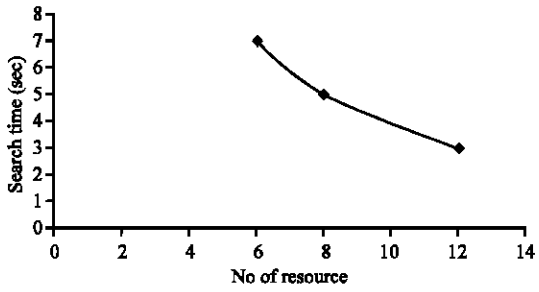

Fig. 5: No. of resources versus search time

```
Resource 3 started at 0 : 56 : 53
```

Once each file is allocated to a resource then the scoring method is called. This is done simultaneously for all resources. The time at which score is computed by each resource is displayed. As the number of documents used is small, there is no distinction in time taken for calculation. The score of each document is stored in a file and until the next query is fed the score will remain in the secondary storage. This process is illustrated below

```
Resource number : 1-----> Filename : /cbse_final/src/automata.htm----->score : 0.0
Resource 1 ends at 0 : 56 : 53
Resource number : 1-----> Filename : /cbse_final/src/Poultry.html----->score : 0.0
Resource 1 ends at 0 : 56 : 53
Resource number : 2-----> Filename : /cbse_final/src/checksum.html----->score : 0.053097345
Resource 2 ends at 0 : 56 : 53
Resource number : 0-----> Filename : /cbse_final/src/selectpoll.htm----->score : 0.016260162
Resource 0 ends at 0 : 56 : 53
Resource number : 4-----> Filename : /cbse_final/src/deshaw2.htm----->score : 1.3046523
Resource 4 ends at 0 : 56 : 53
Resource number : 3-----> Filename : /cbse_final/src/deshaw1.htm----->score : 1.3046523
Resource 3 ends at 0 : 56 : 53
```

The last phase is the shutting down of various entities. Once the entities are shut down the results are consolidated and passed to the front-end. These ranked documents are displayed to the user the order according to their scores on the user interface.

Figure 5 depicts number of resources vs. search time for the queries for number of documents are 3 and sizes of the documents are 2K, 1K and 3K. It can be seen that as the number of resources increases then the search time is reduced.

CONCLUSION

This study has presented a method of discovery of data in a grid using Content Based Search method. The search technique is implemented by doing the following steps:

- Splits the given documents into tokens eliminating redundancy
- Forms a co-occurrence matrix

- Scores the documents based on the co-occurrence matrix
- Ranks the documents based on the score computed

This technique has been implemented and tested in the grid environment.

ACKNOWLEDGMENTS

The authors would like to thank Dr. R. Prabhakar, Principal, Coimbatore Institute of Technology, Dr. RudraMoorthy, Principal, PSG College of Technology and Mr. Chidambaram Kollengode, YAHOO Software Development (India) Ltd, Bangalore for providing us the required facilities to complete the research successfully. This research is carried out as a consequence of the YAHOO's University Relation Programme.

REFERENCES

Amati, G. and C.J. Van Rijsbergen, 2002. Probabilistic models of information retrieval based on measuring divergence from randomness. ACM Trans. Inform. Syst., 20: 357-389.

Buntine, W., J. Löfström, S. Perttu and K. Valtonen, 2005. Topic-specific scoring of documents for relevant retrieval. Workshop on Learning in Web Search, (LWS'05), Bonn, Germany, pp: 34-41.

Foster, I. and C. Kesselman, 1999. The grid: Blueprint for a new computing infrastructure. Morgan Kaufmann and San Francisco.

Moore, R., C. Baru, P. Bourne, M. Ellisman, S. Karin, A. Rajasekar and S. Young, 1997. Information based computing. Proceedings of the Workshop on Research Directions for the Next Generation Internet, May 13-17, Vienna, DC., http://www.cra.org/Policy/NGI/papers/reaganWP.

Ounis, I., G. Amati, V. Plachouras, B. He, C. Macdonald and C. Lioma, 2006. Terrier: A high performance and scalable information retrieval platform. Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval, August 10, Seattle, WA., pp: 18-24.

Page, L., S. Brin, R. Motwani and T. Winograd, 1998. The page rank citation ranking: Bringing order to the Web. Technical Report, Stanford Digital Library Technologies Project.

Ponte, J.M. and W.B. Croft, 1998. A language modeling approach to information retrieval. Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28, Melbourne, Australia, pp: 275-281.

Ville Tuulos, H., 2007. Design and implementation of a content-based search engine. M.Sc. Thesis, University of Helsinki.