World of Computer Science and Information Technology Journal (WCSIT)
ISSN: 2221-0741
Vol. 1, No. 8, 357-363, 2011

Conniving the Information Assimilation and Retrieval (INAR) System for the Heterogeneous, Multi Related Information Sources

Prof L. Senthilvadivu, Research Scholar
Anna University of Technology
Coimbatore, Tamilnadu, India.

Dr. K. Duraiswamy, Dean (Academic)
K.S.R College of Technology
Tiruchengode, Tamilnadu, India.

Abstract— The World Wide Web (WWW) is known for being a web of documents; however, little is known about the structure or growth of such a web. Search engines such as Google have transformed the way people access and use the web and have become a critical technology for finding and delivering information .In the proposed work, the tasks such as the information assimilation and retrieval have been discussed. In the information assimilation, the data synthesis have been done from multi related and heterogeneous information sources and stored hierarchically. The data can be collected from various resources under different domains. The data are available in persistence storage by using default programmatic methodology. In this paper we propose also a searching algorithm to be used in web search engines that simply relies on information that could be extracted based on user queries from multi related and heterogeneous information resources. Hierarchical results from Heterogeneous Domain, Build Positive Set and Fetch Positive Results are the most important aspects of the searching system.

Keywords- Assimilation; Heterogeneous domain; persistence storage; Hierarchical results.

I. INTRODUCTION

Search Engine searches the Internet or selects pieces of the Internet based on query. They keep an index of the words they find, and where they find them. They allow users to look for words or combinations of words found in that index. Early search engines held an index of a few hundred thousand pages and documents, and received one or two thousand inquiries each day. Today, a top search engine will index hundreds of millions of pages, and respond to tens of millions of queries per day. The search engine follows two paradigms; they are words in page and where the word is found. One of the hottest topics in recent years in the AI community, as well as in the Internet community, is the Semantic Web. It is about making the Web more understandable by machines. It is also about building an appropriate infrastructure for intelligent agents to run around the Web performing complex actions for their users. Furthermore, Semantic Web is about explicitly declaring the knowledge embedded in many web-based applications, integrating information in an intelligent way, providing semantic-based access to the Internet, and extracting information from texts.

Semantic web extends the network of hyperlinked human-readable web pages by inserting machine-readable metadata about pages and how they are related to each other, enabling automated agents to access the Web more intelligently and perform tasks on behalf of users. The Semantic Web is a "man-made woven web of data" that facilitates machines to understand the semantics, or meaning, of information on the World Wide Web. The term was coined by Tim Berners-Lee, the inventor of the World Wide Web and director of the World Wide Web Consortium (W3C), which oversees the development of proposed Semantic Web standards. The concept of Semantic Web applies methods beyond linear presentation of information (Web 1.0) and multi-linear presentation of information (Web 2.0) to make use of hyper-structures leading to entities of hypertext. The main purpose of the Semantic Web is driving the evolution of the current Web by enabling users to find, share, and combine information more easily. The semantic web is a vision of information that can be readily interpreted by machines, so machines can perform more of the tedious work involved in finding, combining, and acting upon information on the web.

Creating a search engine which scales even to today's web presents many challenges. Fast crawling technology is needed to gather the web documents and keep them up to date. Storage space must be used efficiently to store indices and, optionally, the documents themselves. The indexing system must process hundreds of gigabytes of data efficiently. Queries must be handled quickly, at a rate of hundreds to thousands per second. These tasks are becoming increasingly difficult as the Web grows. However, hardware performance and cost have improved dramatically to partially offset the difficulty. There are, however, several notable exceptions to this progress such as disk seek time and operating system robustness. In designing Google [1], both the rate of growth of the Web and technological changes have been considered. Google is designed to scale well to extremely large data sets. It makes efficient use of storage space to store the index. Its data structures are optimized for fast and efficient access.

Fig 1 shows the high level Google Architecture. In which there is a URL server that sends lists of URLs to be fetched to the crawlers. The web pages that are fetched are then sent to the store server. The store server then compresses and stores the web pages into a repository. Every web page has an associated ID number called a document ID which is assigned whenever a new URL is parsed out of a web page. The indexing function is performed by the indexer and the sorter. The indexer performs a number of functions. It reads the repository, uncompressed the documents, and parses them. Each document is converted into a set of word occurrences called hits. The hits record the word, position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of "barrels", creating a partially sorted forward index. The indexer performs another important function. It parses out all the links in every web page and stores important information about them in an anchors file. This file contains enough information to determine where each link points from and to, and the text of the link.

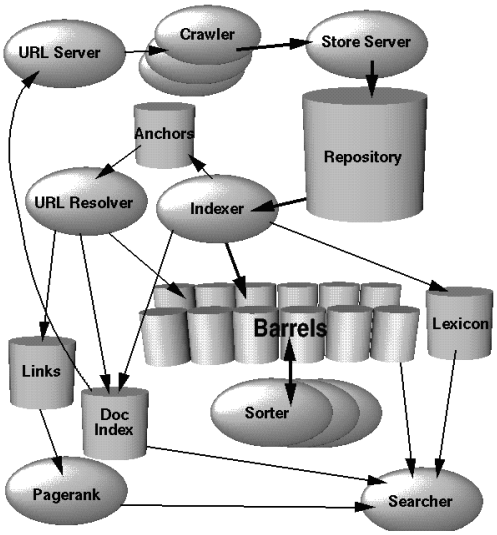

Figure 1. High Level Google Architecture.

The URL resolver reads the anchors file and converts relative URLs into absolute URLs and in turn into doc ID. It puts the anchor text into the forward index, associated with the doc ID that the anchor points to. It also generates a database of links which are pairs of doc ID. The links database is used to compute Page Ranks for all the documents. This is the most common form of text search on the Web. Most search engines do their text query and retrieval using keywords. In the proposed system, the retrieval of information has been done based on the queries of the user as keywords. The information can be retrieved as both content and the links structure of the web page for further more information.

Further, the Chapters are organized in the following manner. The Chapter 2 represents the Overview, Design and Implementation of the system. The Chapter 3 gives the Performance design and Chapter4 compares the related work. The Chapter 5 discusses about the conclusion and future enhancements in the work.

II. THE SYSTEM

2.1 AN OVERVIEW

Text search is a key technology. Search engines that index the Web provide a breadth and ease of access to information that was inconceivable only a decade ago. Text search has also grown in importance at the other end of the size spectrum. Search engines are structurally similar to database systems. Documents are stored in a repository, and an index is maintained. Queries are evaluated by processing the index to identify matches which are then returned to the user. However, there are also many differences. Database systems must contend with arbitrarily complex queries, whereas the vast majority of queries to search engines are lists of terms and phrases. In a database system, a match is a record that meets a specified logical condition; in a search engine, a match is a document that is appropriate to the query according to statistical heuristics and may not even contain all of the query terms. Database systems return all matching records; search engines return a fixed number of matches, which are ranked by their statistical similarity. Lacking full natural language understanding, we believe that in many cases common-sense knowledge and domain-specific knowledge may be used to improve the effectiveness of text categorization by generating more informative features than the mere bag of words.

Within the proposed framework, the system carries out a large number of experiments to understand better and explain why phrase-based models outperform word-based models. The empirical results, which hold for all examined language pairs, suggest that the highest levels of performance can be obtained through relatively simple means: heuristic learning of phrase translations from word-based alignments and lexical weighting of phrase translations. Surprisingly, learning phrases longer than three words and learning phrases from high-accuracy word level alignment models does not have a strong impact on performance.

The goal of the search mechanism is to support the user with finding opportunities and exact information. In order to provide an adequate support mechanism, the search mechanism needs to have access to a number of information sources. The most important sources are the data profiles, which contain structured information about the domains, and the link set, which contain structured and detailed information about the domains that can be adapted with other relevant information. All the information required for the search mechanism is made available by the metadata repository. The search mechanism will provide a number of services for accessing relevant information, ranging from a straightforward keyword based search, to a topic based search.

2.2 PROPOSED SYSTEM

2.2.1ARCHITECTURE OF INAR

The proposed system named as the Information Assimilation and Retrieval (INAR) system can be segregated into four layers. The first layer fetches static and dynamic data from multi related; heterogeneous information sources and assimilates the data hierarchically. Static data is the default data set present in the system and dynamic data can be added with the domain rule. The second layer gets the phrase from user and segregates it with NLQ parser and counts with counter process. Domain Relationship algorithm (DRA) is applied and searching mechanism initiates its task. The third layer starts retrieval from persistence by generating dynamic query depends on number of domains and sub domains found. The dynamic query can be built by application and query will be passed to persistence and result set is prepared. The fourth layer starts the presentation view to populate the result set to view logic. The user can get the information related to the phrase collection and link set. Link set is nothing but links to relevant pages to provide more information related to the query.

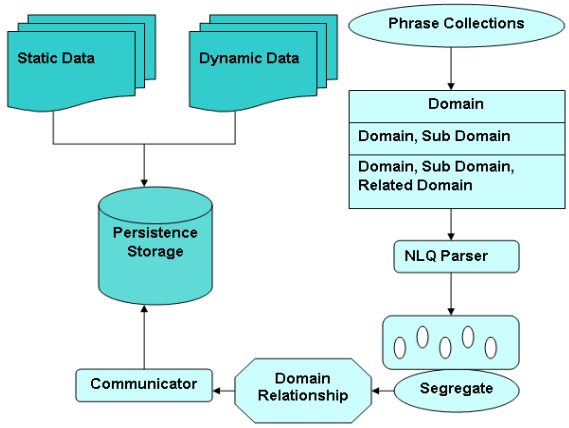

Figure 2. Architecture of INAR.

Fig 2 shows the Architecture of INAR system. The INAR system consists of the powerful structure known as Assimilation and Search configuration development (ASCD) which contains the hierarchical persistence storage and combined data structure with Domain, Child Domain, Concept, Meaning, and Related Uniform Resource Locator (URL) with Serial number (S.NO) as Primary Key. Domain is the main concept; Child Domain is the related topic of Domain. Concept is most relevant word of Domain and Child Domain. Meaning is the explanation and Related URL is the web page name that contains the comprehensive data. Information can be stored by the Administrator by himself or the information can be retrieved from the specified Servers. The default details of the INAR system are also stored by the Administrator. Administrator should authenticate with login page by selecting it from the Home. The home page has the relevant link to enter the contents which increases the heterogeneity.

2.2.1 STATIC AND DYNAMIC TEXTUAL AND PERSISTENCE TO PRESENTATION VIEW

In the proposed system, Knowledge Database Resource is the data resource which is created by the system to collect heterogeneous data. The data can be collected from various multi related, heterogeneous resources under different domains. The data are available in persistence storage hierarchically by using default programmatic methodology as shown in Fig 3.

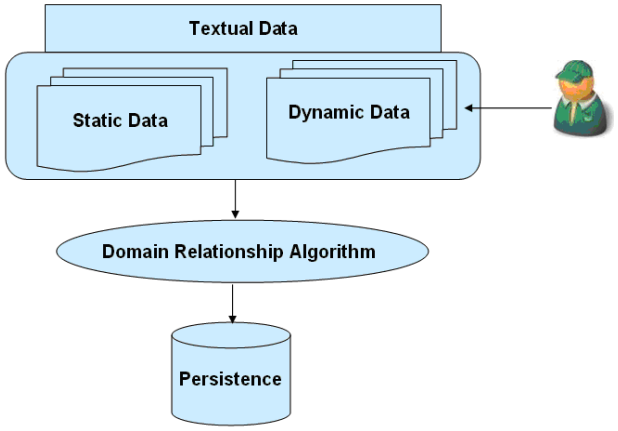

Figure 3. SD Textual and persistence with DRA

To achieve this, the system follows SD textual and persistence to presentation view where as SD stands for Static and Dynamic data. The Knowledge Database Resource is arranged using domain related meaning.

TABLE 1 SD TEXTUAL AND PERSISTENCE TO PRESENTATION VIEW ALGORITHM

1	Static data to persistence
2	Dynamic data view
3	Define w1, w2, w3
4	Define Domain, Sub Domain, Related Domain
5	Connect with Persistence
6	Assign the phrase to Communicator

The Static data are collected from the multi related and heterogeneous information sources where as the Dynamic data are collected with Domain rule, both are stored in the persistence storage in a hierarchical manner. The Static and Dynamic textual and persistence view algorithm can be utilized with Domain relationship where as static data are managed in application by default data structure come with system.

Table 1, represents the SD textual and persistence to presentation view algorithm. Dynamic data can be managed by administrator by adding new data structure and the data structure will be added with existing data structure. Textual data will be managed in persistence after the classification of domain, sub domain and related domain. For example, suppose the administrator is storing the Inheritance property in the topic C++ in the main topic Object Oriented Programming Concept, it is done in the INAR system by storing first the main topic Object Oriented Programming Concept in the Domain, C++ in the

Sub domain, Inheritance in the Concept and the detailed explanation in the Meaning.

2.2.2 DOMAIN RELATIONSHIP ALGORITHM (DRA)

Domain Related Meaning is the new semantic based technology. The pages are arranged in web using web languages and starts with authentication pages. The domain consists of collection of sub domains, concepts and they have the interrelationship meaning between them. To achieve this, the system uses Domain Relationship Algorithm (DRA) to fetch the data, based on the given phrase sequences.

Table 2 represents the Domain Relationship Algorithm. Based on the number of phrases in the query, the results can be fetched by Domain Relationship Algorithm which is the algorithm to fetch the information after divide the sequence collection. The persistence memory already indexed the data collections stored and DRA retrieves the data to the user. Domain Relationship is the relationship between phrases which defines the methodology. Three important aspects mainly implemented in this Domain Relationship Algorithm. The first one is getting the keyword for the Domain from the phrase collections, the second one is getting the keyword for the Sub domain and the third one is getting for the Concept. Three parts are managed in the version and more versions can be added. First version is the domain, second version is the sub domain and third version is the relationship.

TABLE 2 DOMAIN RELATIONSHIP ALGORITHM

1	Data Counter
2	Found number of items
3	Define w1, w2, w3
4	Define Domain, Sub Domain, Related Domain
5	Connect with Fetcher process
6	Assign the phrase to Fetcher process

2.2.3 CLEAVING OF PHRASES

Table 3, represents the Cleaving of phrases. Phrase collections are separated to make easy analysis and assist the application in which the NLQ (Natural Language Query) parser comes to divide the data and make them as partitioned blocks. The blocks can be identified as Inter Relationship task. The task identifies the domain and related items. After the segregation, the communicator communicates with persistence memory. Actual persistence memory holds the data and prepares itself to provide the data based on domain counts.

TABLE 3 CLEAVING OF PHRASES

1	User data arrives
2	NLQ Parser counts the phrase and parse them
3	Apps divider divides the task based on the count
4	Communication Apps connects with persistence for data retrieval
5	Data results set

The goal of the search mechanism is to support the user with finding opportunities and exact info. In order to provide an adequate support mechanism, the search mechanism needs to have access to a number of information sources. The most important sources are the data profiles, which contain structured information about the domains, and the link set, which contain structured and detailed information about the domains that can be adapted with other relevant information. All the information required for the search mechanism is made available by the metadata repository.

Fig 4 represents the Cleaving of Phrases with NLQ Parser which will provide a number of services for accessing relevant information, ranging from a straight forward keyword based search, to a topic search.

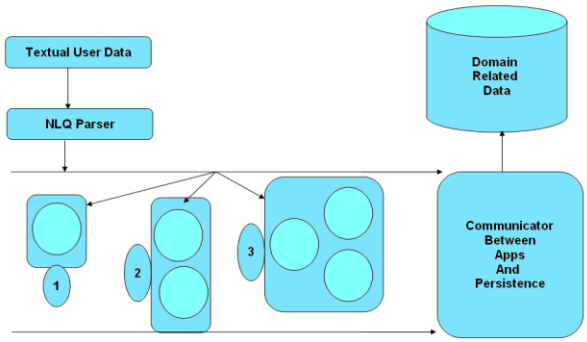

Figure 4. Cleaving of Phrases.

Unless the author of the Web document specifies the keywords for the required document by using Meta tags, it is up to the search engine to determine them. Essentially, this means that search engines pull out and index words that appear to be significant. Since engines are software programs, not rational human beings, they work according to rules established by their creators for what words are usually important in a broad range of documents. The title of a page, for example, usually gives useful information about the subject of the page.

Words that are mentioned towards the beginning of a document are given more weight by most search engines. The same goes for words that are repeated several times throughout the document. Some search engines index every word on every page. Others index only part of the document. Full-text indexing systems generally pick up every word in the text except commonly occurring stop words. Some of the search engines discriminate upper case from lower case; others store all words without reference to capitalization. Keyword searches have a tough time distinguishing between words that are spelled the same way, but mean something different. This often results in hits that are completely irrelevant to the query. Search engines also cannot return hits on keywords that mean the same, but are not actually entered in the query. A query on heart disease would not return a document that used the word "cardiac" instead of "heart."

In the proposed system, NLQ parser separates the phrases given by the user. In Domain based search, the input given by the user is separated as Domain, Child Domain and Concept with the NLQ parser. The result set

can be known as hierarchical result set which means, if the user gives Domain only the user gets the result found within Domain. If the user puts the query to search the Child domain the results are fetched by the system from the Domain and Child domain and so on. Hence the searching techniques are trying to refine the searching area in respond to the queries by the user. NLQ Parser first counts the given term. Different counts define the application way. The terms can be understood by the application and the terms will be sent to persistence and matched with appropriate fields to build the query.

III. PERFORMANCE DESIGN

In order to determine the concepts that best describes a query, the relatedness scores between the concepts and the query need to be determined. The approaches chosen for our experiments are explained in greater detail. First, a term-level phrase based approach is presented, and then three level graph-based approaches are introduced. In both cases, it is assumed that the concepts have been manually mapped to the correct corresponding data. Meta tags allow the owner of a page to specify keywords and concepts under which the page will be indexed. Classical search engines provide the links of the search terms and not content of search terms. Some of this research using statistical analysis on pages containing the words or phrases search for the new algorithm is natural language queries. It reaches semantic web based cluster response of web server instead of providing just links.

Phrase collection are separated to make easy analysis and assist the application the NLQ parser comes to divide the data and make them as partitioned blocks. The blocks can be identified as Inter Relationship task which identifies the domain and related items. After the segregation, the communicator communicates with persistence memory that holds the data and prepares itself to provide the data based on domain counts. In persistence memory, lots of static and dynamic information are stored for the web data retrieval to the user. Domain relationship Algorithm fetches the data and Finder process correctly defines the phrases using counter process. More Finders provides the sample search, among this Correct Finder gets the positive result set. The important commitment is that assigning the phrase to the phrase matcher process which yields the result with link set. The added advantage of the system is the link set provides detailed information page. Information retrieval from the link set is another focal process and it helps the navigation process very effectively. One of the major areas of this search engine is concept based searching with Heterogeneous data. In order to avoid the burden of useless pages, Time Consumption, Assistance to user, the proposed system provides accurate information, Positive Map, Link set to connect for further resources.

IV. RELATED WORK

Semantic data transformation plays an important role in realizing the vision of the semantic web. It supports the transformation of data in different representations into ontology. In order to allow the task to be achieved effectively, the instructions on how to realize transformation should be well specified, preferably in a declarative and reusable format, thus allowing the construction of robust tools which on the one hand assist users to generate and maintain mappings at design time and on the other hand perform semantic data transformation at run time. Furthermore, the transformation instructions should not only allow the generation of semantic data objects but also allow the creation of rich semantic relations between them. This context [2] developed a comprehensive instance mapping ontology. One distinctive feature of the instance mapping ontology is that it provides comprehensive support for the specification of complex mappings. Another feature is that the instance mapping ontology is representation independent, which does not limit itself to data sources in particular representations. This ontology has been applied in generating a semantic layer for the web site of the Knowledge Media Institute (KMi) at the Open University.

There has been a large body of related work on data integration. Early work on multi-database systems [6] focused on relational or object-oriented database views for integrated access to data from several relational databases. More recently, mediators [21] and wrappers have been developed for information integration from multiple data repositories (including semi-structured and unstructured data). Examples include the TSIMMIS project at Stanford University [4], the SIMS project [17], the Ariadne project at the University of Southern California, the Hermes project at the University of Maryland [7], NIMBLE – a commercial system based on research at the University of Washington [20], and the TAMBIS project in UK [5]. INDUS has been inspired by, and builds on previous work by several groups on data integration, and in particular, logic-based and ontology-based approaches to data integration [6]. A major goal of the INDUS project is to provide modular, extensible, open source software environment for ontology-based information integration and data-driven knowledge acquisition from heterogeneous, distributed, autonomous information sources.

The work [15] on understanding and classifying web queries presents algorithms and techniques for increasing a search service's understanding of web queries. Provided within are a set of techniques and metrics for performing temporal analysis on query logs. The metrics proposed for the log analysis are shown to be reasonable and informative, and can be used to detect changing trends and patterns in the query stream, thus providing valuable data to a search service. Continued with an algorithm for automatic topical classification of web queries, results are presented showing that the classification approach can be successfully applied to a significant portion of the query stream, making it possible for search services to leverage it for improving search effectiveness and efficiency.

When humans approach text processing tasks, such as text categorization, they interpret documents in the context of their background knowledge and experience. On the other hand, conventional information retrieval systems represent documents as bags of words, and are restricted to learning from individual word occurrences in the training

set. To enrich document representation through automatic use of vast repositories of human knowledge end, Wikipedia and the Open Directory Project, the largest encyclopedia and Web directory, respectively are used in this work [13]. Wikipedia articles and ODP categories represent knowledge concepts. In the preprocessing phase, a feature generator analyzes the input documents and maps them onto relevant concepts. The latter give rise to a set of generated features that either augment or replace the standard bag of words. Feature generation is accomplished through contextual analysis of document text, thus implicitly performing word sense disambiguation.

In dynamic environments, such as the World Wide Web, a changing document collection, query population, and set of search services demands frequent repetition of search effectiveness (relevance) evaluations. Reconstructing static test collections, such as those used in TREC [14], requires considerable human effort, as large collection sizes demand judgments deep into retrieved pools. In practice it is common to perform shallow evaluations over small numbers of conditions (often binary, A vs. B) without system pooling, not intending to construct reusable test collections. This work focuses on making repeatable evaluation possible in such an environment by providing a method for determining the number of queries that must be evaluated for reliable results, introducing resource-oriented automatic evaluation techniques, and providing a framework for integrating these automatic techniques with manual judgments to reduce the amount of manual judgment required. This semiautomatic framework is validated against a manual evaluation of the top ten results of ten web search engines across 896 queries in navigational and informational tasks. Leveraging bootstrapped observed power enables us to compare conclusions drawn across evaluations, focusing on only those that are reproducible across query sets. Augmenting manual judgments with pseudo-relevance judgments mined, even naively, from web taxonomies reduces both the chances of missing a correct binary conclusion and those of finding an errant conclusion by approximately 50%. These semiautomatic techniques provide reliable insight into conclusions before an evaluation is fully completed, enabling intelligent evaluation strategies to reduce effort.

Most research on Text Categorization TC [16] approach classification as a binary categorization problem. Rather than determining the category of a document from a multitude of categories, typical TC algorithms classify a document as relevant or not relevant to a given category. However, most real world entities, such as Web documents, are composed of a variety of topics (i.e. multi-class) and can belong to more than one class (multi-label). The common approach of binary TC algorithms to multi-class categorization is to transform a multiple-category assignment problem into multiple binary decision problems by breaking the classification task into multiple disjoint binary tasks. In other words, a document is classified for each category, and the binary results are pooled to arrive at a single decision based on ranking of possible categories. The main problem with such an approach is that it ignores correlation between classes and trivializes multi-label categorization.

This work [18] investigates whether the automatic grouping of similar documents (document clustering) is a feasible method of presenting the results of Web search engines. Several key requirements for document clustering of search engine results: clustering quality, concise and accurate cluster descriptions, and speed. In response, A novel clustering algorithm – Suffix Tree Clustering (STC) [18] is specifically designed for this task in several respects. First, STC groups documents based on shared phrases. Second, it allows overlapping clusters. Finally, STC is a fast, incremental, and linear-time (in the number of documents) algorithm. The clustering quality of STC is evaluated and showed it superior to other commonly used algorithms on Web search engine results. STC is to be significantly faster than other linear-time algorithms when clustering search engine snippets.

Web data are the item sets that define the effective of web structure. The addition of data set increases the ease of use of data availability in web structure. For the positive analysis, it can be viewed to mobile network [10] increases the user smoothness data fetching way. The negative map added the discovery process to the user not only understanding about the positive structure and also the negative structure to avoid in future and share with other networks. Adding the domain words in search terms and undefined phrase inputs are the key terms. Unlike keyword search systems, INAR system try to determine what it means, not just what it says. In the best circumstances, this searching system returns hits on documents that are "about" the subject/theme exploring, even if the words in the document do not precisely match the words enter into the query. When the working of this method is concerned, there are various methods of building clustering systems, some of which are highly complex, relying on sophisticated linguistic and artificial intelligence theory that we won't even attempt to go into here. INAR system is comparably performs its assimilation hierarchically and searches for the content by executing queries.

V. CONCLUSION AND FUTURE WORK

The present work explains the search based on the queries posted by the user to the multi related, heterogeneous information sources which gives out the content merely matching the queries. The contents are stored hierarchically in the information sources which are heterogeneous and multi related in nature. The information can be retrieved from the sources with heterogeneous map using Static and Dynamic textual retrieval and persistence to presentation view with Domain Relationship algorithm. This kind of searching is trying to give simply the exact content retrieval for the user based on the queries posted by the user which can eliminate the searching of millions of links for the queries. This work can be extended by including the dictionary as such the related words shown in the system which may assure the exact meaning and anchoring the user to get the positive result within no time.

REFERENCES

[1] Sergey Brin and Lawrence Page (1998) The Anatomy of a Large-Scale Hyper textual Web Search Engine Proc. Seventh Int'l Conf.World Wide Web (WWW '98), pp. 107-117.

[2] Doan, A., Madhavan, J., Domingos, P. and Halvey, A., (2002) "Learning to map between ontologies on the semantic web", in Proceedings of WWW'02, pp. 662-673.

[3] T. Berners-Lee, J. Hendler, and O. Lassila (2001) "The Semantic Web," scientific Am.

[4] Garcia-Molina , Y. Papakonstantinou , D. Quass, A. Rajaraman , Y.Sagiv , J. Ullman , V. Vassalos , J. Widom (1996). The TSIMMIS approach to mediation: Data models and Languages. Journal of Intelligent Information Systems

[5] Paton, N.W., Stevens, R., Baker, P.G., Goble, C.A., Bechhofer, S., (1999). Query processing in the TAMBIS bioinformatics source integration system. In: Proc. 11th Int. Conf. on Scientific and Statistical Databases (SSDBM), IEEE Press, 138-147.

[6] Castillo, J.A.R.; Silvescu, A.; Caragea, D.; Pathak, J.; Honavar, V.G.; (2003). Information extraction and integration from heterogeneous, distributed, autonomous information sources - a federated ontology-driven query-centric approach, Proceedings of IEEE International Conference on Information Reuse and Integration.

[7] Subrahmanian, V.S., Sibel Adali, Anne Brink, James J. Lu, Adil Rajput, Timothy J. Rogers, Robert Ross, Charles Ward (2000). HERMES A Heterogeneous Reasoning and Mediator System.

[8] L. Ding, T. Finin, A. Joshi, R. Pan, R.S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs, "Swoogle: A Search and Metadata Engine for the Semantic Web," Proc. 13th ACM Int'l Conf. Information and Knowledge Management (CIKM '04), pp. 652-659, 2004.

[9] L. Ding, T. Finin, A. Joshi, Y. Peng, R. Pan, and P. Reddivari, (2005) "Search on the Semantic Web," Computer, vol. 38, no. 10, pp. 62-69.

[10] Risvik, K. M. and Michelsen, R. (2002). Search Engines and Web Dynamics. Computer Networks, vol. 39, pp. 289–302, June 2002.

[11] Gómez-Pérez, A., & Corcho, O. Ontology Languages for the Semantic Web. IEEE Intelligent Systems 17(1), 54-60, 2002.

[12] Heflin, J., & Hendler, J. A Portrait of the Semantic Web in Action. IEEE Intelligent Systems 16(2), 54-59, 2001.

[13] Gabrilovich E. (2006). Feature Generation for Textual Information Retrieval Using World Knowledge. Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)

[14] Jensen E.C. (2006). Repeatable Evaluation of Information Retrieval Effectiveness in Dynamic Environments. Illinois Institute of Technology. May 2006.

[15] Beitzel S.M. (2006). On Understanding and Classifying Web Queries. Illinois Institute of Technology. May 2006.

[16] Yang K. (2002). Combining Text-, Link-, and Classification-based Retrieval Methods to Enhance Information Discovery on the Web. University of North Carolina, May 2002.

[17] Arens, Y., Chee, C., Hsu, C., and Knoblock, C. (1993) Retrieving and Integrating Data from Multiple Information Sources. International Journal

of Intelligent and Cooperative Information Systems. Vol. 2, No. 2. Pp. 127-158

[18] Zamir O.E. (1999). Clustering Web Documents: A Phrase-Based Method for Grouping Search Engine Results. University of Washington, 1999.

[20] Draper, D., Halevy, A., and Weld, D. (2001). The NIMBLE XML Data integration System. In: Proceedings of the International Conference on Data Engineering (ICDE 01).

[21] Wiederhold, G. and M. Genesereth (1997) the Conceptual Basis for Mediation Services, IEEE Expert, Vol.12 No.5 pp. 38-47

[22] McBryan Oliver A. McBryan.(1994) GENVL and WWWW: Tools for Taming the Web. First International Conference on the World Wide Web. CERN, Geneva (Switzerland), May 25-26-27 1994.

[23] Shkapenyuk, V. and Suel, T. (2002). Design and implementation of a high performance distributed web crawler. In Proceedings of the 18th International Conference on Data Engineering (ICDE), pages 357-368, San Jose, California. IEEE CS Press.

AUTHORS PROFILE

Prof L.Senthilvadivu received M.Sc., M.Phil in Physics, MCA, and M.Phil in Computer Science. She is currently doing research in the field of Data mining in Computer Applications under Anna University of Technology, Coimbatore. She has 17 years of experience in the field of Physics and Computer Science and shown her excellence by attending short term course in the field of Mobile Computing organized by K.S.R College of Engineering and a workshop in the area of Data Mining using Memetic Algorithm organized by Periyar University, Salem. She has published number of papers in national and international conferences and journals. She is a member of ISTE and IEEE.

Dr.K. Duraiswamy B.E., M.Sc. (Engg.), Ph.D, MISTE, SMIEEE is currently working as Dean (Academic) in K.S. Rangasamy College of Technology, Tiruchengode, Tamilnadu, India. He has 42 years of teaching and research experience. He has guided 16 Ph.Ds in the area of Computer Science and Engineering in addition to 14 M.Phil students in Computer Science. He is currently guiding more than 12 students for Ph.D. He has also guided more than 100 M.E. students in the area of Computer Science and Engineering. He has published 51 papers in International Journals and 12 papers in National Journals in addition to participating more than 72 National and 43 International Conferences. His area of interest are : Image Processing, Network Security, Data Communication, Soft Computing, State estimation, Power System load forecasting and scheduling, Computer Architecture character recognition Data mining etc.