Warehousing the Web or Building Virtual Web Views

Osmar R. Zaïane
Department of Computing Science
University of Alberta
Edmonton, AB, Canada T6G 2E1
zaiane@cs.ualberta.ca

Abstract

We are so used to the ubiquitous World-Wide Web (WWW) that we take it for granted. There is no need to emphasize how dynamic, large, rich, and unstructured, yet important the Web is. From researchers and engineers to children and retired elderly, everyone uses the WWW for a variety of needs. A multitude of tools and search engines were developed to find and retrieve resources from the Web. However, everyone knows how frustrating the experience with search engines can be. It is very difficult to find, if ever found, relevant information or patterns from within resources on the Internet. The idea presented in this paper is to "warehouse" the Web in a structure that would allow efficient information retrieval and knowledge discovery from the Internet. Warehousing the Web in this context consists of creating different virtual web views with layered databases of descriptors organized hierarchicly. Using a declarative adhoc mining language, one can find and pinpoint explicit as well as implicit knowledge from the web warehouse.

Keywords: Querying the Web, Knowledge Discovery, Resource Discovery, Data Warehousing, Web Mining.

With the rapid expansion of the information base and the user community on the Internet, efficient and effective discovery and use of the resources in the global information network has become an important issue in the research into global information systems.

Although research and developments of database systems have been flourishing for many years, with different kinds of database systems successfully developed and delivered to the market, a global information system, such as the Internet, stores a much larger amount of information in a much more complicated and unstructured manner than any currently available database system. Thus, the effective organization, discovery and use of the rich resources in the global information network poses great challenges to database and information system researchers. In a recent report on the future of database research known as

the Asilomar Report [1], it has been predicted that in ten years from now, the majority of human information will be available on the World-Wide Web, and it has been observed that the database research community has contributed little to the Web thus far.

The first major challenge of a global information system is the diversity of information in the global information base. The current information network stores hundreds of tera-bytes of information including documents, softwares, images, sounds, commercial data, library catalogues, user directory data, weather, geography, other scientific data, and many other types of information. Since users have the full freedom to link whatever information they believe useful to the global information network, the global information base is huge, heterogeneous, in multimedia form, mostly unstructured, dynamic, incomplete and even inconsistent, which creates tremendous difficulty in systematic management and retrieval in comparison with the structured, well-organized data in most commercial database systems.

The second challenge is the diversity of user community. The Internet currently connects more than 40 million workstations [2, 3], and the user community is still expanding rapidly. Users may have quite different backgrounds, interests, and purposes of usage. Also, most users may not have a good knowledge about the structure of the information system, may not be aware of the heavy cost of a particular search (e.g., a click may bring megabytes of data over half of the globe), and may easily get lost by groping in the "darkness" of the network, or be bored by taking many hops and waiting impatiently for a piece of information.

The third challenge is the volume of information to be searched and transmitted. The huge amount of unstructured data makes it unrealistic for any database system to store and manage and for any queries to find *all* or even *most* of the answers by searching through the global network. The click-triggered massive data transmission over the network is not only costly and unbearable even for the broad bandwidth of the communication network, but is also too wasteful or undesirable to many users. Search effectiveness (e.g., hit ratio) and performance (e.g., response time) will be bottlenecks for the successful applications of the global information system.

There have been many interesting studies on information indexing and searching in the global information base with many global information system servers developed. Crawlers, spider-based indexing techniques used by search engines, like the WWW Worm [4], RBSE database [5], Lycos [6] and others, create a substantial value to the web users, but generate an increasing Internet backbone traffic. They not only flood the network and overload the servers but also lose the structure and the context of the documents gathered. These wandering software agents on the World Wide Web have already created controversies [7, 8]. Other indexing solutions, like ALIWEB [9] or Harvest [10], behave well on the network but still struggle with the difficulty to isolate information with relevant context, and cannot solve most of the problems posed for systematic discovery of resources and knowledge in the global information base.

In this paper, a different approach, called a Multiple Layered DataBase (MLDB) approach for building Virtual Web Views (VWV) is proposed to facilitate information discovery in global information systems. We advocate spider-less indexing of the Internet. Authors or web-server administrators send their own indexes or pointers to resources to be indexed. When documents are changed, added or removed, the indexing process is triggered again. An MLDB is a database composed of several layers of information, with the lowest layer (i.e.,

layer-0) corresponding to the primitive information stored in the global information base and the higher ones (i.e., *layer-1* and above) storing generalized information extracted from the lower layers.

The proposal is based on the previous studies on *multiple layered databases* [11, 12] and *data mining* [13, 14] and the following observations.

With the development of data analysis, transformation and generalization techniques, it is possible to generalize and transform the diverse, primitive information in the network into reasonably structured, classified, descriptive and higher-level information. Such information can be stored in a massive, distributed but structured database which serves as the layer-1 database in the MLDB. By transforming an unstructured global information base into a relatively structured "global database", most of the database technologies developed before can be applied to manage and retrieve information at this layer.

However, the layer-1 database is usually still too large and too widely distributed for efficient browsing, retrieval, and information discovery. Further generalization should be performed on this layer at each node to form higher layer(s) which can be then merged with the corresponding layered database of other nodes at some backbone site in the network. The merged database can be replicated and propagated to other remote sites for further integration [15]. This integrated, higher-layer database may serve a diverse user community as a high-level, global information base for resource discovery, information browsing, statistical studies, etc.

The multiple layered database architecture transforms a huge, unstructured, global information base into progressively smaller, better structured, and less remote databases to which the well-developed database technology and the emerging data mining techniques may apply. By doing so, the power and advantages of current database systems can be naturally extended to global information systems, which may represent a promising direction. Moreover, data mining can be put to use in such hierarchical structure in order to perform knowledge discovery on the World-Wide Web (or the Internet).

The remainder of the paper is organized as follows. In Section 1, we briefly present a general taxonomy for web mining. We survey some techniques and approaches relevant to Knowledge Discovery on the Internet. Most of these techniques are integrated in our model. A model for global MLDB is introduced in Section 2. Methods for construction and maintenance of different layers of the global MLDB are proposed in Section 3. Resource and knowledge discovery using the global MLDB and a web mining language WebML is investigated in Section 4. Finally, a discussion and conclusive remarks are presented in Section 5.

1 Data Mining or Knowledge Discovery on the Internet

Data mining, as defined in [16], is the process of non-trivial extraction of implicit, previously unknown and potentially useful information from data in large databases. Data mining is the principal core of the knowledge discovery process, which also includes data integration, data cleaning, relevant data selection, pattern evaluation and knowledge visualization. Tradition-

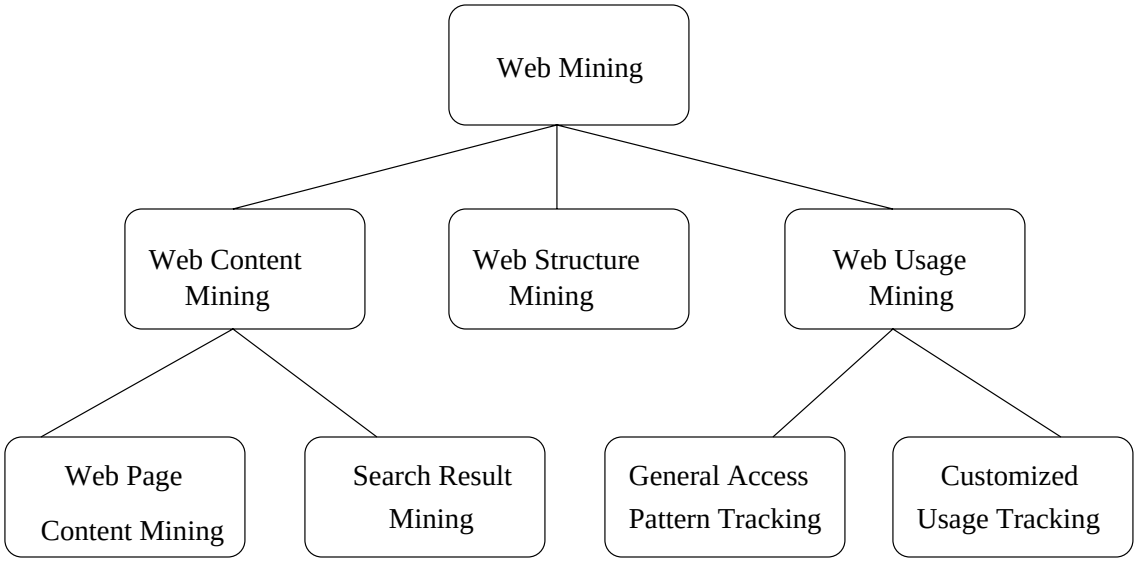

Figure 1: Taxonomy of Web Mining techniques.

ally, data mining has been applied to databases. The wide spread of the World-Wide Web technology has made the large document collection in the World-Wide Web a new ground for knowledge discovery research. In contrast to resource discovery that finds and retrieves resources from the Internet, knowledge discovery on the Internet aims at deducing and extracting implicit knowledge not necessarily contained in a resource. Traditional knowledge discovery functions put to use on databases, like characterization, classification, prediction, clustering, association, time series analysis, etc. can all be applied on the global information network. Not all these functionalities were attempted on the Internet but a few were applied to document repositories with some success.

The Internet is a highly dynamic multimedia environment involving interconnected heterogeneous repositories, programs, and interacting users. Obviously, text mining has a limited grasp on knowledge in such an environment. Data mining on the Internet, commonly called webmining, needs to take advantage of the content of documents, but also of the usage of such resources available and the relationships between these resources. Web mining, the intersection between data mining and the World-Wide Web, is growing to include many technologies conventionally found in artificial intelligence, information retrieval, or other fields. Agent-based technology[17], concept-based information retrieval, information retrieval using case-based reasoning[18], and document ranking using hyperlink features and usage are often categorized under web mining. Web mining is not yet clearly defined and many topics will continue to fall into its realm.

We define Web Mining as *the extraction of interesting and potentially useful patterns and implicit information from resources or activity related to the World-Wide Web.*

Figure 1 shows a classification of domains that we believe to be akin to Web Mining. In the World-Wide Web field, there are roughly three knowledge discovery domains that pertain to web mining: Web Content Mining, Web Structure Mining, and Web Usage Mining. Web content mining is the process of extracting knowledge from the content of documents or their

descriptions. Web document text mining, resource discovery based on concepts indexing or agent-based technology may also fall in this category. Web structure mining is the process of inferring knowledge from the World-Wide Web organization and links between references and referents in the Web. Finally, web usage mining, also known as Web Log Mining, is the process of extracting interesting patterns in web access logs. All these are relevant in our model.

1.1 Web Content Mining

Most of the knowledge in the World-Wide Web is buried inside documents. Current technology barely scratches the surface of this knowledge by extracting keywords from web pages. This has resulted in the dissatisfaction of users regarding search engines and even the emergence of human assisted searches[1] on the Internet. Web content mining is an automatic process that goes beyond keyword extraction. Since the content of a text document presents no machine-readable semantic, some approaches have suggested to restructure the document content in a representation that could be exploited by machines. Others consider the Web structured enough to do effective web mining. Nevertheless, in either case an intermediary representation is often relied upon and built using known structure of a limited type and set of documents (or sites) or using typographic and linguistic properties. The semi-structured nature of most documents on the Internet helps in this task. Essence[19], the technology used by the harvest system[10] relies on known structure of semi-structured documents to retrieve information. The usual approach to exploit known structure in documents is to use wrappers to map documents to some data model. Many declarative languages have been proposed to query such data models. Weblog[20] relies on Datalog-like rules to represent web documents. WebOQL[21] uses graph trees to extract knowledge and restructure web documents. WebML[22, 23], presented later in this paper, uses relational tables to take advantage of relational database power and data mining possibilities. Systems like Ahoy![24] and the shopping agent described in [25] use heuristics and learning techniques to recognize some document structure. Techniques using lexicons for content interpretation are yet to come.

There are two groups of web content mining strategies: Those that directly mine the content of documents and those that improve on the content search of other tools like search engines [26, 27].

1.2 Web Structure Mining

Thanks to the interconnections between hypertext documents, the World-Wide Web can reveal more information than just the information contained in documents. For example, links pointing to a document indicate the popularity of the document, while links coming out of a document indicate the richness or perhaps the variety of topics covered in the document. This can be compared to bibliographical citations. When a paper is cited often,

[1] Some sites like http://www.humansearch.com, http://www.searchmill.com and http://www.searchforyou.com offer search services with human assistance.

it ought to be important. The PageRank[28] and CLEVER[29] methods take advantage of this information conveyed by the links to find pertinent web pages (hubs and authorities).

The virtual web views presented in this paper also benefit from the structure of the Web by abstracting relevant information from web resources and keeping relationships between them. The virtual web views exploit the knowledge conveyed by the information network but not explicitly stated in documents. By means of counters, higher levels cumulate the number of resources subsumed by the concepts they hold. Counters of hyperlinks, in and out of documents, retrace the structure of the web resources summarized.

1.3 Web Usage Mining

Despite the anarchy in which the World-Wide Web is growing as an entity, locally on each server providing the resources there is a simple and well structured collection of records: the web access log. Web servers record and accumulate data about user interactions whenever requests for resources are received. Analyzing the web access logs of different web sites can help understand the user behaviour and the web structure, thereby improving the design of this colossal collection of resources. There are two main tendencies in Web Usage Mining driven by the applications of the discoveries: General Access Pattern Tracking and Customized Usage Tracking. A survey paper [30] presents a detailed taxonomy for web usage mining methods and systems.

2 A Multiple Layered Database Model for Global Information Systems

In this section we present the multi-layered database structure underlying the virtual web views. This structure takes into account the three aspects of web mining presented above: web content mining, web structure mining, and web usage mining.

Although it is difficult to construct a data model for the primitive global information base (i.e., layer-0), advanced data models can be applied in the construction of better structured, higher-layered databases. To facilitate our discussion, we assume that the nonprimitive layered database (i.e., layer-1 and above) is constructed based on an extended-relational model with capabilities to store and handle complex data types, including set- or list- valued data, structured data, hypertext, multimedia data, etc. Multiple layered databases can also be constructed similarly using other data models, including object-oriented and extended entity-relationship models.

Definition 2.1 *A global* multiple layered database (MLDB) *consists of 3 major components:* $\langle \mathcal{S}, \mathcal{H}, \mathcal{D} \rangle$*, defined as follows.*

1. $\mathcal{S}$: a database schema*, which contains the meta-information about the layered database structures;*

2. $\mathcal{H}$: a set of concept hierarchies*; and*

3. $\mathcal{D}$: a set of (generalized) database relations at the nonprimitive layers of the MLDB and files in the primitive global information base. □

The first component, a database schema, outlines the overall database structure of the global MLDB. It stores general information such as structures, types, ranges, and data statistics about the relations at different layers, their relationships, and their associated attributes as well as the location where the layers reside and are mirrored. Moreover, it describes which higher-layer relation is generalized from which lower-layer relation(s) (i.e., a route map) and how the generalization is performed (i.e., generalization paths). Therefore, it presents a route map for data and metadata (i.e., schema) browsing and for assistance of resource discovery.

The second component, a set of concept hierarchies, provides a set of predefined concept hierarchies which assist the system to generalize lower layer information to high layer ones and map queries to appropriate concept layers for processing. These hierarchies are also used for query-less browsing of resources such as drill-down and roll-up operations.

The third component consists of the whole global information base at the primitive information level (i.e., layer-0) and the generalized database relations at the nonprimitive layers. In other words, it contains descriptions of on-line resources summarized in each layer.

The third component is, by definition, dynamic. Note that the first, as well as the second component, can also dynamically change. The schema defined in the first component of the MLDB model can also be enriched with new fields, and new route maps can be defined after the system has been initially conceived. The updates are incremental and are propagated, in the case of the schema update, from lower layers to higher ones. New concept hierarchies can be defined as well, or updated. While updates to current concept hierarchies imply incremental updates in layered structure, new concept hierarchies may suggest the definition of a new set of layers or an analogue MLDB.

We first examine the database schema. Due to the diversity of information stored in the global information base, it is difficult, and even not realistic, to create relational database structures for the primitive layer information base. However, it is possible to create relational structures to store reasonably structured information generalized from primitive layer information. For example, based on the accessing patterns and accessing frequency of the global information base, layer-1 can be organized into dozens of database relations, such as *document, person, organization, images, sounds, software, map, library_catalogue, commercial_data, geographic_data, scientific_data, games*, etc. The relationships among these relations can also be constructed either explicitly by creating relationship relations as in an entity-relationship model, such as *person-organization*, or implicitly (and more desirably) by adding the linkages in the tuples of each (entity) relation during the formation of layer-1, such as *adding URL* [2] *pointers pointing to the corresponding authors ("persons") in the tuples of the relation "document" when possible.*

To simplify our discussion, we assume that the layer-1 database contains only two relations, document and person. Other relations can be constructed and generalized similarly.

[2]Uniform Resource Locator. Reference is available by anonymous FTP from ftp.w3.org as /pub/www/doc/url-spec.txt

Example 2.1 Let the database schema of layer-1 contain two relations, document and person, as follows (with the attribute type specification omitted).

1. document(*file_addr, authors, title, publication, publication_date, abstract, language, table_of_contents, category_description, keywords, index, multimedia_attached, num_pages, format, first_paragraphs, size_doc, timestamp, access_frequency, URL_links_in, URL_links_out, ...*).

2. person(*last_name, first_name, home_page_addr, position, picture_attach, phone, e-mail, office_address, education, research_interests, publications, size_of_home_page, timestamp, access_frequency, ...*).

Take the *document* relation as an example. Each tuple in the relation is an abstraction of one *document* from the information base (layer-0). The whole relation is a detailed abstraction (or descriptor) of the information in documents gathered from a site. The first attribute, *file_addr*, registers its file name and its "URL" network address. The key could have been a system generated object identifier *doc_id*, used to identify the documents which may be duplicated and have different URL addresses, such as in [31]. However, for simplicity we chose to retain the URL of a document as a key and duplicate the entries in *document* if necessary, allowing documents to evolve independently. There is a possibility to have two URLs for the same on-line document, especially with virtual domain addresses, but it is difficult to identify and thus we chose not to add another identifier other than the document URL. There are several attributes which register the information directly associated with the file, such as *size_doc* (size of the document file), *timestamp* (the last updating time), etc. There are also attributes related to the formatting information. For example, the attribute *format* indicates the format of a file: .ps, .dvi, .tex, .troff, .html, text, compressed, uuencoded, etc. One special attribute, *access_frequency*, registers how frequently the entry is being accessed. This is either access relative to the record in layer-1 or access collected from the web log file of the web server where the document resides. URL_links_in, URL_links_out, register the number of known pointers pointing to the document (i.e. popularity of the document), and the number of pointers coming out of the documents (i.e. number of URLs in the document). The popularity of a document can be weighted relatively to the importance of the initial document that point at it. If the initial document (i.e. parent document) is in the same topic or is popular itself, the counter is multiplied by a higher coefficient, however, if the initial document is not relevant or from the same site as the current document, the counter is multiplied by a low coefficient. $URL\_Links\_in \equiv \sum_{i,j} C_i$, where j is the number of distinct URLs pointing to the document, and C_i is 1 for an irrelevant parent page from the same web site, and higher otherwise. Relevance in this context can be measured by intersection of the document keyword sets (topics). The same applies for URL_Links_out: $URL\_Links\_out \equiv \sum_{i,k} C_i$, where k is the number of distinct URLs in the document, and C_i their "importance". Other attributes register the major semantic information related to the document, such as *authors, title, publication, publication_date, abstract, language, table_of_contents, category_description, keywords, index, multimedia_attached, num_pages, first_paragraphs*, etc. □

Note that getting the Links_out list of a document is straightforward, however, the Links_in list can be difficult if we look at the links as a sparse matrix between all existing URLs. Such a matrix for the Web can not be computed in a realistic manner. In the VWV context, Links_in contains only "known links", that is links from documents in the

VWV. When a document is added, its Links_out is divided into two sets: the known links and the outside VWV links. The known links are used to update the Links_in of those documents in the VWV. The Links_in of the new document is computed by checking for the URL of the new document in the Links_out lists of the VWV.

Layer-1 is a detailed abstraction (or descriptor) of the layer-0 information. The relations in layer-1 are substantially smaller than the primitive layer global information base but still rich enough to preserve most of the interesting pieces of general information for a diverse community of users to browse and query. Layer-1 is the lowest layer of information manageable by database systems. However, it is usually still too large and too widely distributed for efficient storage, management and search in the global network. Further compression and generalization can be performed to generate higher layered databases.

Example 2.2 Construction of an MLDB on top of the layer-1 global database.

The two layer-1 relations presented in Example 2.1 can be further generalized into layer-2 database which may contain two relations, doc_brief and person_brief, with the following schema,

1. doc_brief(*file_addr, authors, title, publication, publication_date, abstract, language, category_description, keywords, num_pages, format, size_doc, access_frequency, URL_links_in, URL_links_out*).

2. person_brief (*last_name, first_name, publications, affiliation, e-mail, research_interests, size_home_page, access_frequency*).

The resulting relations are usually smaller with less attributes and records. The least popular fields from layer-1 are dropped, while the remaining fields are inherited by the layer-2 relations. Relations are split according to different classification schemes, while tuples are merged relying on successive subsumptions according to the concept hierarchies used. General concept hierarchies are provided explicitly by domain experts. Other hierarchies are built automatically and stored implicitly in the database. We have proposed and implemented a technique for the construction of a concept hierarchy for keywords extracted from web pages using an enriched WordNet semantic network[32]. This approach [33] is presented later in this paper.

Further generalization can be performed on layer-2 relations in several directions. One possible direction is to partition the *doc_brief* file into different files according to different classification schemes, such as category description (e.g., *cs_document*), access frequency (e.g., *hot_list_document*), countries, publications, etc., or their combinations. Choice of partitions can be determined by studying the referencing statistics. Another direction is to further generalize some attributes in the relation and merge identical tuples to obtain a "summary" relation (e.g., *doc_summary*) with data distribution statistics associated [14]. The third direction is to join two or more relations. For example, *doc_author_brief* can be produced by generalization on the join of *document* and *person*. Moreover, different schemes can be combined to produce even higher layered databases.

A few layer-3 relations formed by the above approaches are presented below.

1. cs_doc(*file_addr, authors, title, publication, publication_date, abstract, language, category_description, keywords, num_pages, format, size_doc, access_frequency, URL_links_in, URL_links_out*).

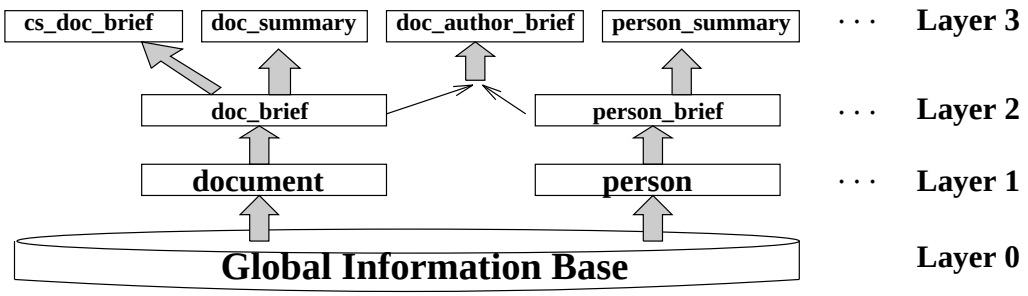

Figure 2: A conceptual route map of the global information base

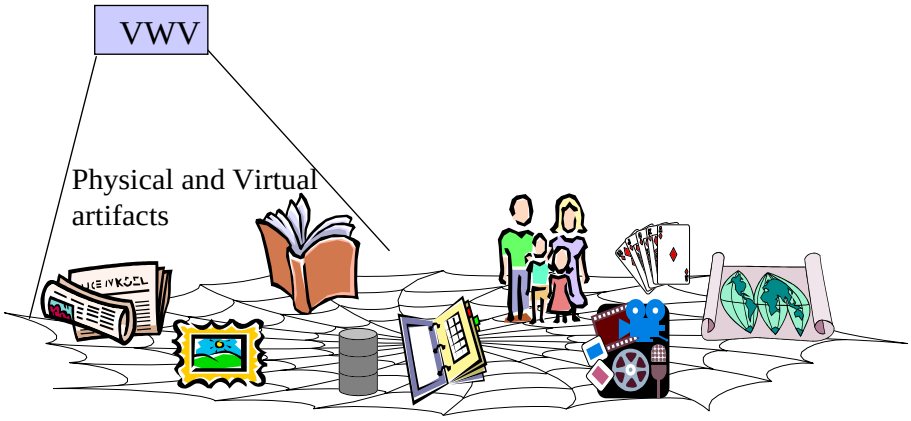

Figure 3: A VWV abstracts a selected set of resources and makes the WWW appear as structured.

2. doc_summary(*affiliation, field, publication_year, first_author_list, file_addr_list, average_popularity, count*).

3. doc_author_brief(*file_addr, authors, affiliation, title, publication, pub_date, category_description, keywords, num_pages, format, size_doc, access_frequency, URL_links_in, URL_links_out*).

4. person_summary (*affiliation, research_interest, year, num_publications, count*).

The attribute *count*, is a counter that reckons the records from the lower layer generalized into the current record. *average_popularity* averages the *URL_links_in* count of the generalized records from the lower layer.

In general, the overall global MLDB structure is constructed based on the study of frequent accessing patterns. It is also plausible to construct higher layered databases for a special-interest community of users (e.g., ACM/SIGMOD, IEEE/CS) on top of a common layer of the global database. This generates partial views on the global information network, hence, the name Virtual Web View (VWV). A VWV provides a window to observe a subset of Web resources, and gives the illusion of a structured world.

This customized local higher layer acts as cache which may drastically reduce the overall network traffic [34, 35]. Some systems like Lagoon[3] "mirror" remote documents, but we

[3]Lagoon Caching Software Distribution, available from ftp://ftp.win.tue.nl/pub/infosystems/www/README.lagoon

believe that caching indexes (i.e. high layers containing descriptors) would be definitely more profitable.

One possible schema of a global MLDB (containing only two layer-1 relations) is presented in Figure 2. □

The first step, and probably the most challenging one in the construction of the layered structure of the VWV, is the transformation and generalization of the unstructured data of the primitive layer into relatively structured data, manageable and retrievable by databases. The challenge is mostly due to the common and persistent absence of information describing information in the primitive layer: Metadata. However, the eminent appearance and acceptance of standards and recommendations for metadata availability such as the Dublin Core metadata initiatives [36], the eXtensible Markup Language (XML) and the Resource Description Framework (RDF)[37], will simplify this task.

3 Construction and maintenance of MLDBs

A philosophy behind the construction of MLDB is information abstraction, which assumes that most users may not like to read the details of large pieces of information (such as complete documents) but may like to scan the general description of the information. Usually, the higher level of abstraction, the better structure the information may have. Thus, the sacrifice of the detailed level of information may lead to a better structured information base for manipulation and retrieval.

Figure 4 presents the general architecture of a multiple layered global information base, where the existing global information base forms layer-0, and the abstraction of layer-0 forms layer-1. Further generalization of layer-1 and their integration from different sites form layer-2, which can be replicated and propagated to each backbone site, and then be further generalized to form higher layers.

3.1 Construction of layer-1: From Global Information Base to Structured Database

The goal for the construction of layer-1 database is to transform and/or generalize the unstructured data of the primitive layer at each site into relatively structured data, manageable and retrievable by the database technology. Three steps are necessary for the realization of this goal: (1) standardization of the layer-1 schema, (2) development of a set of softwares which automatically perform the layer-1 construction, and (3) layer construction and database maintenance at each site.

Obviously, it is neither realistic nor desirable to enforce standards on the format or contents of the primitive layer information in the global information network. However, it is desirable to construct a rich, shared and standardized layer-1 schema because such a schema may lead to the construction of a structured information base and facilitate information management and sharing in the global information network.

While enforcing a standard for describing document content, which would considerably help in the construction of the first layer of the MLDB, is a difficult and probably, and

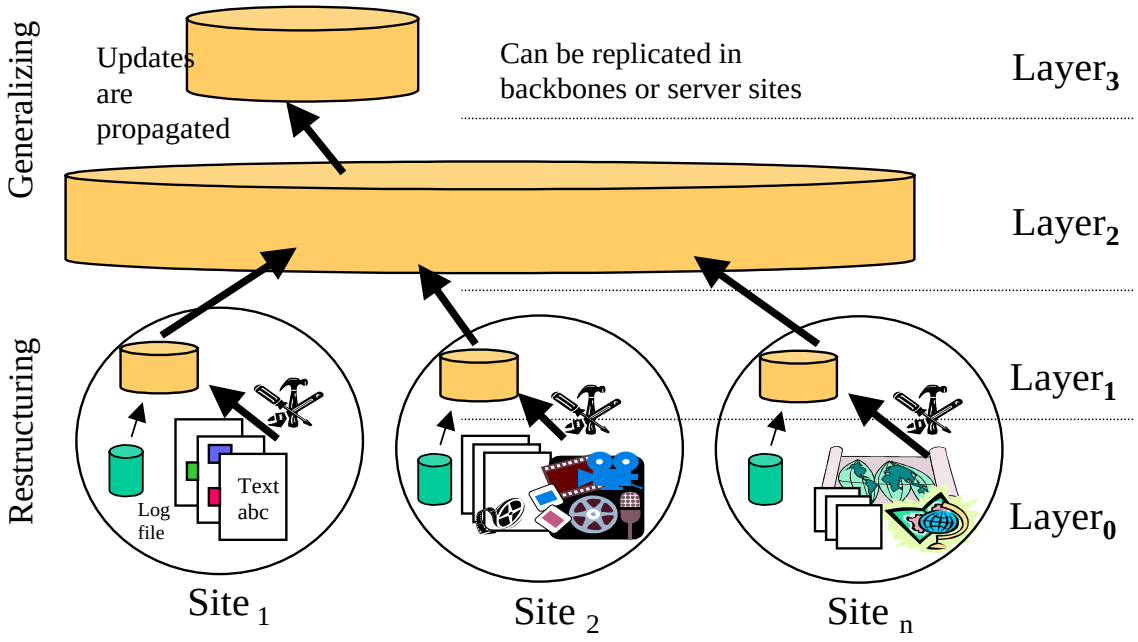

Figure 4: Using tools to generate and update the first layer of the MLDB structure.

arguably, an utopian task, some industry standards such as XML (and XML-based applications) and other recommendations such as the Dublin Core Metadata initiative and the W3C Resource Description Framework, are gaining momentum and will become in the near future a great asset that will facilitate the implementation of the first MLDB stratum in an efficient and economical way. Metadata and a uniform interpretation of descriptors of metadata are key issues in this respect. Initially, a Virtual Web View can either be restricted to a niche of resources with metadata attached, or compromise with information extracted or deduced by specialized tools and agents.

A standard layer-1 schema can be worked out by studying the accessing history of a diverse community of users and predicting the future accessing patterns by experts. Such a schema is constructed incrementally in the process of increasingly popular use of the information network, or standardized by some experts or a standardization committee. However, it is expected that such a schema may need some infrequent, incremental modifications, and the layer-1 database would be modified accordingly in an automatic and incremental way.

To serve a diverse community of users for flexible information retrieval, the layer-1 schema should be rich enough to cover most popular needs, and detailed enough to reduce excessive searches into the layer-0 information base. Notice that different VWVs can coexist to serve different community needs. Because of the diversity of information in a global information base, there often exist cases in which data have complex structures or cannot match the specified schema.

A VWV can also be limited to a subset of documents accompanied with metadata and progressively augmented with new documents as metadata becomes available. This strategy might encourage authors and resource creators to describe their documents with established standards if they wish their documents to be accessible in a VWV-like system. In our

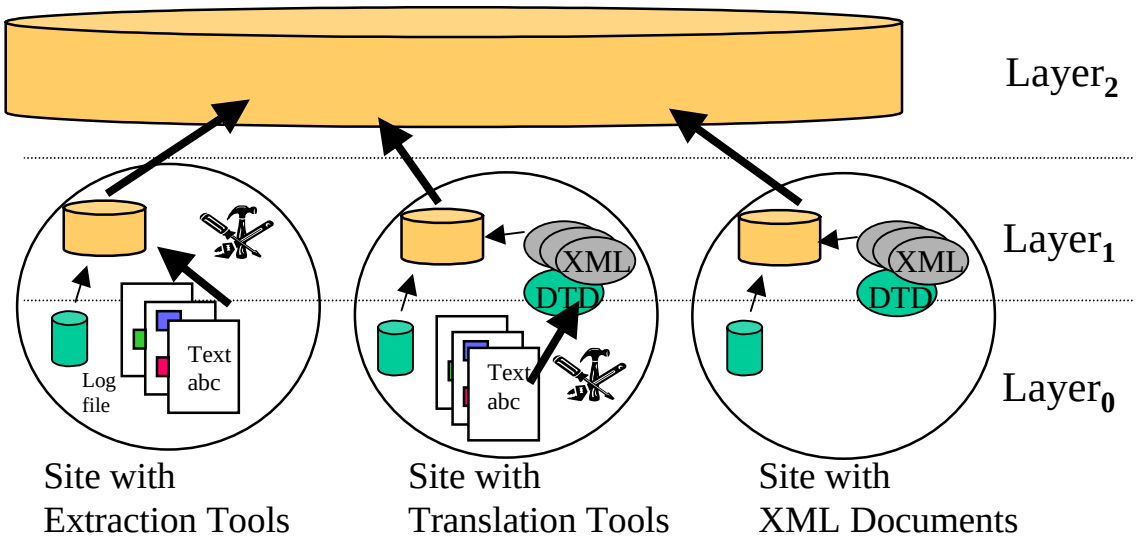

Figure 5: Extraction of metadata for Layer-1 construction.

experiments, we have restricted our VWV to a set of documents for which we had metadata available.

Figure 5 shows two types of layer-1 construction software: Extraction tools and translation tools. Web sites with XML-like documents that follow metadata semantic guidelines would just need an XML parser to build the first layer. However, for other sites, documents can either be translated to XML form with translation tools for the parser to process, or extraction tools are used to retrieve the relevant and available information from the documents to be directly added to the first layer. Notice that these tools can be managed and executed on site by the site administrators (or even the document authors) when even they feel necessary or possible. This would avoid unnecessarily overloading the servers to retrieve the needed information.

Ideally, the layer-1 construction is done using on-the-shelf tools. There are several powerful query languages specifically developed for querying semistructured data. These mostly declarative languages are used to retrieve information from semistructured or unstructured data, or managing web sites by accessing clever data models. A good survey of some of these tools can be found in [38]. Lorel (for Lightweight Object Repository Language) based on labeled graph data model, is developed for querying XML or other semistructured data [39]. Other query languages like UnQL [40], StruQL from the Strudel system [41], and WebOQL [21] also use labeled graphs or hypertrees as a flexible data model to represent semistructured data. While some of them are designed as part of systems for web site management or web restructuring, their general purpose is to query semistructured data in Web context, and thus can be used to extract relevent information from web documents in order to populate the layer-1 of the MLDB. However, there is a need for wrappers describing the underlying structure of web documents in order to query web documents in a database-like fashion. Web document structuring languages, like WebOQL or WebLog [20], are capable of retrieving information from on-line pages such as news sites like CNN, tourist guides, or conference lists, but are limited to semi-structured Web sources that have wrappers defined around

them. New promising research on automatic generation of wrappers for web documents has been proposed. For example, [42] proposes an approach for generating wrappers around web documents based on the underlying structure of a Web page.

3.2 Generalization: Formation of higher layers in MLDB

Since local layer-1 databases are all connected via Internet, they collectively form a globally distributed, huge layer-1 database. Although information retrieval can be performed directly on such a global database, performance would be poor if global-wide searches have to be initiated frequently.

Higher layered databases are constructed on top of the layer-1 database by generalization techniques. Generalization reduces the size of the global database, makes it less distributed (by replicating the smaller, higher-layer databases at, for example, network backbone sites or local server sites), while still preserving the general descriptions of the layer-1 data.

Clearly, successful generalization becomes a key to the construction of higher layered databases. Following studies on attribute-oriented induction for knowledge discovery in relational databases [14, 43], an attribute-oriented generalization method has been proposed for the construction of multiple layered databases [12]. According to this method, data in a lower layer relation are generalized, attribute by attribute, into appropriate higher layer concepts. Different lower level concepts are generalized into the same concepts at a higher level and are merged together, which reduces the size of the database.

We examine in detail the generalization techniques for the construction of higher layered databases.

3.2.1 Concept generalization

Nonnumeric data (such as keywords, index, etc.) is the most commonly encountered type of data in the global information base. Generalization on nonnumerical values should rely on the concept hierarchies which represent necessary background knowledge that directs generalization. Using a concept hierarchy, primitive data is expressed in terms of generalized concepts in a higher layer.

Concept hierarchies are provided explicitly by domain experts or stored implicitly in the database. For the global MLDB, a set of relatively stable and standard concept hierarchies should be provided as a common reference by all the local databases in their formation of higher layered databases and in their browsing and retrieval of information using different levels of concepts.

The concept hierarchies for keywords and indexes can be obtained by referencing a standard concept hierarchy catalogue which specifies the partial order of the terms frequently used in the global information base.

A portion of the concept hierarchy for *keywords* that we used in our experiments is illustrated in Figure 6, and the specification of such a hierarchy and alias is in Figure 7. Notice that a **contains**-list specifies a concept and its immediate subconcepts; and an **alias**-list specifies a list of synonyms (aliases) of a concept, which avoids the use of complex lattices in the "hierarchy" specification. The introduction of alias-lists allows flexible queries and

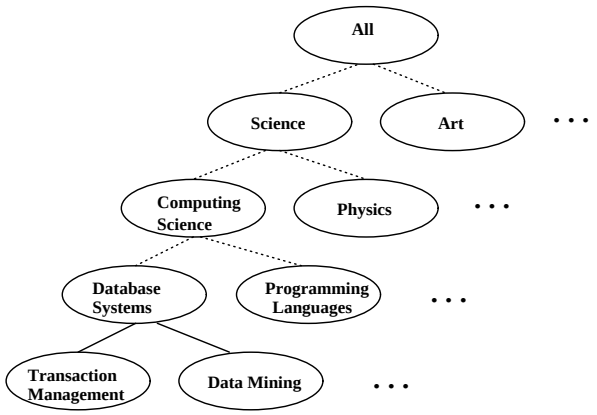

Figure 6: A possible concept hierarchy for *keywords*

helps dealing with documents using different terminologies and languages. Also, the dashed lines between concepts in Figure 6 represent the possibility to have other layers of concepts in between.

Such a concept hierarchy is either provided by domain experts, or constructed as follows:

1. Collect the frequently used words, technical terms and search keys for classification.
2. Build up a skeleton classification hierarchy based on the technical term specification standards in each field, such as *ACM Computing Review: Classification System for Computing Reviews*, etc. Notice that most of such classification standards are on-line documents.
3. Consult on-line dictionaries (such as Webster dictionary and thesaurus, etc.) for automatically attaching the remaining words to appropriate places in the hierarchy (which may need some human interaction).
4. Consult experts in the fields to make sure that the hierarchy is reasonably complete and correct.
5. Incrementally update such a hierarchy, when necessary, due to the introduction of new terminologies.

In the preliminary experiments, the used concept hierarchy (also shown in Figure 7) was built manually using the set of all keywords extracted from our document collection. We had also first hand experience automatically building a concept hierarchy for the MultiMediaMiner project [44]. The hierarchy was built using WordNet semantic network [32, 45], a collection of more than 95,000 English words with their relationships, commonly used in cognitive science and computational linguistics.

Algorithm 3.1 Creating a concept hierarchy of recognized keywords using WordNet semantic Network.

Input: (i) List of keywords $\mathcal{L}$kw; (ii) List of domain specific terms and phrases *domain*, (iii) enriched WordNet $EWordNet$.

All	**contains**:	Science, Art, ...
Science	**contains**:	Computing Science, Physics, Mathematics, ...
Computing Science	**contains**:	Theory, Database Systems, Programming Languages, ...
Computing Science	**contains**:	database systems, Programming Languages, ...
Computing Science	**alias**:	Information Science, Computer Science, Computer technologies, ...
Theory	**contains**:	Parallel Computing, Complexity, Computational Geometry, ...
Parallel Computing	**contains**:	Processors Organization, Interconnection Networks, PRAM, ...
Processor Organization	**contains**:	Hypercube, Pyramid, Grid, Spanner, X-tree, ...
Interconnection Networks	**contains**:	Gossiping, Broadcasting, ...
Interconnection Networks	**alias**:	Intercommunication Networks, ...
Gossiping	**alias**:	Gossip Problem, Telephone Problem, Rumor, ...
Database Systems	**contains**:	Data mining, transaction management, query processing, ...
Database Systems	**alias**:	Database technologies, Data management, ...
Data mining	**alias**:	Knowledge discovery, data dredging, data archaeology, ...
Transaction management	**contains**:	concurrency control, recovery, ...
Computational Geometry	**contains**:	Geometry Searching, Convex Hull, Geometry of Rectangles, Visibility, ...
...		

Figure 7: Specification of hierarchies and aliases extracted from an experimental concept hierarchy for computer science related documents.

Output: (i) List of rejected keywords $\mathcal{R}$, (ii) Concept hierarchy $\mathcal{CH}$.

Method. For all given words, accept only those that are domain specific, recognized in WordNet or their canonical form is recognized by WordNet. Organize the accepted words in a hierarchy given the parent-child relationship in WordNet. The pseudo-code for creating the keyword hierarchy is as follows:

```
begin
(1)  R ← ∅ ; L ← ∅
(2)  foreach word in Lkw do {
(3)      if (word ∈ domain) add word to L; next word
(4)      accept ← false
(5)      CanonicalForms ← MorphologicalAnalysis(word)
(6)      foreach form in CanonicalForms do {
(7)          if (form ∈ EWordNet) add form to L; accept ← true
(8)      }
(9)      if (¬ accept) add word in R
(10) }
(11) foreach word in L do {
(12)     if (word ∉ CH)
(13)         parent ← lookup(ParentOf(word),EWordNet)
(14)         while parent ∉ CH do {
(15)             descendant ← parent; parent ← lookup(ParentOf(descendant),EWordNet)
(16)         }
(17)         D ← GetChildren(parent, CH
(18)         add word to CH
(19)         draw arc from parent to word in CH
(20)         foreach descendant in D do {
(21)             if (IsParent(word, descendant, EWordNet)
(22)                 remove arc from parent to descendant in CH
(23)                 draw arc from word to descendant in CH
```

```
(24)                endif
(25)            }
(26)        endif
(27)  }
end                                                                    □
```

Lines 1 to 10 are the cleaning step retaining in $\mathcal{L}$ only recognized words. The rejected words are put in $\mathcal{R}$ (line 9) which is later consulted by an ontology expert who would either discard the words or consider them new domain related terms by adding them manually to the concept hierarchy and WordNet for future runs. *MorphologicalAnalysis* is a procedure that extracts all possible forms and declinations of a given term. Lines 11 to 27 are the hierarchy tree building.

The constructed concept hierarchies are replicated to each server participating in the VWV, together with the higher layered databases, for information browsing and resource discovery. Managing very large concept hierarchies is a challenge. An efficient encoding taxonomies and managing of dynamic partial orders techniques for reasoning with taxonomies (concept hierarchies, lattices, or complex semantic networks) in computer applications have been proposed in [46, 47].

Generalization on numerical attributes is performed in a more automatic way by the examination of data distribution characteristics [48, 49, 50]. In many cases, it may not require any predefined concept hierarchies. For example, the size of document can be clustered into several groups, such as {*below 10Kb, 10Kb-100Kb, 100Kb-1Mb, 1Mb-10Mb, over 10Mb*}, according to a relatively uniform data distribution criteria or using some statistical clustering analysis tools. Appropriate names are assigned to the generalized numerical ranges, such as {*tiny-size, small-size, middle-size, large-size, huge-size*} to convey more semantic meaning.

With the availability of concept hierarchies, generalization can be performed to produce the strata of the MLDB structure.

3.2.2 Attribute-oriented generalization

Data generalization refers to generalizing data within an attribute in a relational tuple, such as merging generalized data within a set-valued data item, whereas relation generalization refers to generalizing a relation, which often involves merging generalized, identical tuples in a relation.

By removing nongeneralizable attributes (such as long text data, etc.) and generalizing data in other attributes into a small set of values, some different tuples may become identical at the generalized concept level and can be merged into one. A special attribute, *count*, is associated with each generalized tuple to register how many original tuples have been generalized into the current one. This process reduces the size of the relation to be stored in a generalized database but retains the general description of the data of the original database at a high concept level. Such a summarized view of data may facilitate high-level information browsing, statistical study, and data mining.

With data and relation generalization techniques available, the next important question is how to selectively perform appropriate generalizations to form useful layers of databases. In principle, there could be a large number of combinations of possible generalizations by

selecting different sets of attributes to generalize and selecting the levels for the attributes to reach in the generalization. However, in practice, a few layers containing most frequently referenced attributes and patterns are sufficient to balance the implementation efficiency and practical usage.

Frequently used attributes and patterns are determined before generation of new layers of an MLDB by the analysis of the statistics of query history or by receiving instructions from users and experts. It is wise to remove rarely used attributes but retain frequently referenced ones in a higher layer. Similar guidelines apply when generalizing attributes to a more general concept level. For example, for a document, the further generalization of layer-1 *document* to layer-2 *doc_brief* can be performed by removing the less frequently inquired attributes *table_of_contents, first_paragraphs*, etc.

Notice that a new layer could be formed by performing generalization on one relation or on a join of several relations based on the selected, frequently used attributes and patterns. Generalization [14] is performed by removing a set of less-interested attributes, substituting the concepts in one or a set of attributes by their corresponding higher level concepts, performing aggregation or approximation on certain attributes, etc.

Since most joins of several relations are performed on their key and/or foreign key attributes, whereas generalization may remove or generalize the key or foreign key attributes of a data relation, it is important to distinguish between the following two classes of generalizations.

1. key-preserving generalization, in which all the key or foreign key values are preserved.
2. key-altering generalization, in which some key or foreign key values are generalized, and thus altered. The generalized keys should be marked explicitly since they usually cannot be used as join keys at generating subsequent layers.

It is crucial to identify altered keys since, if the altered keys were used to perform joins of different relations, it may generate incorrect information [12]. Notice that a join on generalized attributes, though undesirable in most cases, could be useful if the join is to link the tuples with *approximately* the same attribute values together. For example, to search documents, one may like to consider some closely related but not exactly the same subjects. Such kind of join is called an approximate join to be distinguished from the precise join.

Usually, only *precise join* is considered in the formation of new layered relations using joins because approximate join may produce huge sized joined relations and may also be misleading in the semantic interpretation at different usages. However, approximate join will still be useful for searching some weakly connected concepts in a resource discovery query.

3.2.3 An MLDB construction algorithm

Based on the previous discussion, the construction of an MLDB can be summarized into the following algorithm, which is similar to attribute-oriented generalization in knowledge discovery in databases [14].

Algorithm 3.2 Construction of an MLDB.

Input: A global information base, a set of concept hierarchies, and a set of frequently referenced attributes and frequently used query patterns.

Output: A multiple layered database (MLDB) abstracting a given subset of the WWW.

Method. A global MLDB is constructed in the following steps.

1. Determine the multiple layers of the database based on the frequently referenced attributes and frequently used query patterns.

2. Starting with the global information base (layer-0), generalize the relation step-by-step (using the given concept hierarchies and generalized schema) to form multiple layered relations (according to the layers determined in Step 1).

3. Merge identical tuples in each generalized relation and update the *count* of the generalized tuple.

4. Construct a new schema by recording the definitions of all the generalized relations, their relationships and the generalization paths.

Rationale of Algorithm 3.2.

Step 1 indicates that the layers of an MLDB should be determined based on the frequently referenced attributes and frequently used query patterns. This is reasonable since to ensure the elegance and efficiency of an MLDB, only a small number of layers should be constructed, which should provide maximum benefits to the frequently accessed query patterns. Obviously, the frequently referenced attributes should be preserved in higher layers, and the frequently referenced concept levels should be considered as the candidate concept levels in the construction of higher layers. Steps 2 and 3 are performed in a way similar to the attribute-oriented induction, studied previously [14, 12]. Step 4 constructs a new schema which records a route map and the generalization paths for information browsing and knowledge discovery. □

Example 3.1 A portion of relation *doc_brief* is presented in Table 1.

file_addr	authors	title	publication	pub_date	key_words	···
http://fas.sfu.ca/9/cs/research /projects/HMI-5/documents /papers/han/coop94.ps.gz	J. Han Y. Fu R. Ng	Cooperative Query Answering Using Multiple Layered Databases	Proc. 2nd Int'l Conf. Cooperative Info. Systems	May 1994	data mining, multiple layered database, ···	···
···	···	···	···	···	···	···
ftp://ftp.cs.colorado.edu /pub/cs/techreports /schwartz/FTP.Caching-PS	P.B.Danzig R.S.Hall M.F.Schwartz	A Case for Caching File Objects Inside Internetworks	Proc. SIGCOMM	Sept. 1993	caching, ftp, ···	···
···	···	···	···	···	···	···
http://sobolev.mit.edu/people /jphill/publications/shap.dvi	J.R.Phillips H.S.J. Zant	Influence of induced magnetic fields on Shapiro steps in Josephson junction arrays	Physical Review B 47	1994	magnetic fields, Josephson array, Shapiro step, ···	···
···	···	···	···	···	···	···

Table 1: A portion of *doc_brief* extracted from *document* at layer-1.

By extraction of only the documents related to *computing science*, a layer-3 relation *cs_doc* can be easily obtained. Also, performing attributed-oriented induction on *doc_brief* leads to another layer-3 relation *doc_summary*, a portion of which is shown in Table 2.

affiliation	field	pub_year	count	first_author_list	file_addr_list	···
Simon Fraser Univ.	Database Systems	1994	15	Han, Kameda, Luk, ···	···	···
Univ. of Colorado	Global Network Systems	1993	10	Danzig, Hall, ···	···	···
MIT	Electromagnetic Field	1993	53	Bernstein, Phillips, ···	···	···
···	···	···	···	···	···	···

Table 2: A portion of *doc_summary* extracted from *doc_brief* at layer-2.

Notice that backward pointers can be stored in certain entries, such as *first_author_list* and *file_addr_list*, in the *doc_summary* table, and a click on a first author or a file_address will lead the presentation of the detailed corresponding entries stored in layer-2 or layer-1. □

3.3 Distribution and maintenance of the global MLDB

3.3.1 Replication and distribution of the global MLDB

A global MLDB is constructed by extracting extra-layers from an existing (layer-0) global information base using generalization and transformation techniques. A higher layer database is usually much smaller than the lower layered database. However, since the layer-1 database is resulted from direct, detailed information extraction from the huge global information base, its size is still huge. It is unrealistic to have this layer replicated and distributed to other servers. A possible implementation is to store each local layer-1 database at each local network server site, but to replicate the higher layered databases, such as layer-2 and above, and propagate them to remote backbone and/or ordinary network servers. Load can be further partitioned between backbone and ordinary servers. For example, one may store a complete layer-2 database at the backbone site but only the relatively frequently referenced portions of layer-2 and/or higher layers at the corresponding sites. Also, specifically projected layers (e.g., medical database) can be stored at the closely relevant sites (e.g., hospitals and medical schools). By doing so, most information browsing and brief query answering can be handled by searching within the local network. Only detailed requests will be forwarded to the backbone servers or further to the remote sites which store the information. Only when the full document is explicitly requested by a user (with the awareness of its size), will the full layer-0 document be sent across the network to the user site. This will substantially reduce the amount of data to be transmitted across the network and thereby improve the response time.

Moreover, some higher layered databases could be defined by users for easy reference. For example, a user may define a new database at a high layer as "*all the documents related to heterogeneous databases published in major conferences or journals since 1990*". An information manager cannot construct a new database for every user's definition. Most such definitions will be treated like views, i.e., no physical databases will be created, and queries on such views will be answered by the query modification technique [51, 52]. Only if such a view is shared and frequently referenced, may it be worthwhile to create a new database for it.

3.3.2 Incremental updating of the global MLDB

The global information base is dynamic, with information added, removed and updated constantly at different sites. It is very costly to reconstruct the whole MLDB database. Incremental updating could be the only reasonable approach to make the information updated and consistent in the global MLDB.

In response to the updates to the original information base, the corresponding layer-1 and higher layers should be updated incrementally. Incremental updates can be performed on every update or at regular times at the local site and propagate the updates to higher layers.

We only examine the incremental database update at insertion and update. Similar techniques can be easily extended to deletions. When a new file is connected to the network, a new tuple t is obtained by the layer-1 construction algorithm. The new tuple is inserted into a layer-1 relation R_1. Then t should be generalized to t' according to the route map and be inserted into its corresponding higher layer. Such an insertion will be propagated to higher layers accordingly. However, if the generalized tuple t' is equivalent to an existing tuple in this layer, it needs only to increment the count of the existing tuple, and further propagations to higher layers will be confined to count increment as well. When a tuple in a relation is updated, one can check whether the change may affect any of its high layers. If not, do nothing. Otherwise, the algorithm will be similar to the deletion of an old tuple followed by the insertion of a new one.

4 Web Mining Language

Similar to other extended-relational database systems, a Virtual Web View (VWV) system treats the requests for information browsing and resource discovery like relational queries. However, since concepts in a VWV are generalized at different layers, search conditions in a query may not match exactly the concept level of the currently inquired or available layer of the database. For example, to find documents related to a particular topic, such as "attribute-oriented induction", a query may put this term as a search key. However, the current layer may only contain terms corresponding to a higher concept level, such as "induction techniques", or "data mining methods". In this case, it is unlikely to find in the current layer an exact match with the provided search key, but is likely to find a more general concept that subsumes the search key. On the other hand, a search key in a query may be at a more general concept level than those at the current layer. For example, a search key "sports", though conceptually covers the term "baseball", does not match it in the database. Therefore, a key-oriented search in a VWV leads us to introduce four additional relational operations to extend the semantics of traditional selection and join. These operators, *coverage, subsumption, synonymy*, and *approximation*, have their correspondent built-in language primitive in WebML defined respectively as COVERS, COVERED BY, LIKE and CLOSE TO. Other primitives could be defined by users and written as external programs accessing the concept hierarchy.

Definition 4.1 In the global MLDB system, four additional intrinsic relationships: *coverage, covered_by, synonym*, and *approximation*, are defined as follows.

1. coverage ($\supset$): A concept A covers another concept B, denoted as $A \supset B$, if A or A's synonym is an ancestor of B or B's synonym in the same concept hierarchy.

2. covered_by ($\subset$): A concept A is covered by another concept B, denoted as $A \subset B$, if A or A's synonym is a descendant of B or B's synonym in the same concept hierarchy.

3. synonym ($\simeq$): A concept A is a synonym of another concept B, denoted as $A \simeq B$, if A and B are in the same alias list in the same concept hierarchy.

4. approximation ($\sim$): A concept A is an approximate of another concept B, denoted as $A \sim B$, if A or A's synonym is a sibling of B or B's synonym in the same concept hierarchy. □

Based on these relationships, additional selection and join operations can be defined as follows.

Definition 4.2 Let σ be a selection performed on the i-th attribute (column) of relation R using the selection constant c. Four addition selection operations are defined as follows,

1. coverage-selection: if the selection predicate is $c \supset \$i$, i.e., the selection operation is $\sigma_{c \supset \$i} R$,

2. covered_by-selection: if the selection predicate is $c \subset \$i$, i.e., the selection operation is $\sigma_{c \subset \$i} R$,

3. synonym-selection: if the selection predicate is $c \simeq \$i$, i.e., the selection operation is $\sigma_{c \simeq \$i} R$, and

4. approximation-selection: if the selection predicate is $c \sim \$i$, i.e., the selection operation is $\sigma_{c \sim \$i} R$. □

Similarly, one can define four corresponding join operations in the global MLDB systems by replacing the query constant c in the selection predicate of the definition with the j-th column of a relation S.

4.1 A query language for information discovery in the global MLDB

With the construction of the global MLDB, a query language, WebML, can be defined for resource and knowledge discovery using a syntax similar to the relational language SQL [51, 52]. Four newly introduced operators have their correspondent language primitives in WebML, as shown in Table 3.

WebML borrows heavily from a data mining query language DMQL [53]. The top-level WebML query syntax is presented in Table 4. A more formal grammar of WebML can be found in [54] At the position for the keyword select in SQL, an alternative keyword list can

WebML primitive	operation	Name of the operation
COVERS	$\supset$	coverage
COVERED BY	$\subset$	covered_by
LIKE	$\simeq$	synonym
CLOSE TO	$\sim$	approximation

Table 3: New WebML primitives for additional relational operations.

<WebML> ::= <MINE HEADER> **FROM** relation_list
[**RELATED TO** name_list] [**IN** location_list]
WHERE where_clause
[**ORDER BY** attribute_name_list]
[**RANK BY**] {inward | outward | access}
<MINE HEADER>::=
{{ **SELECT** | **LIST** } { attributes_name_list | * }
| <DESCRIBE HEADER> | <CLASSIFY HEADER>}
<DESCRIBE HEADER>::= **MINE DESCRIPTION**
IN RELEVANCE TO { attributes_name_list | * }
<CLASSIFY HEADER>::= **MINE CLASSIFICATION**
ACCORDING TO attributes_name_list
IN RELEVANCE TO { attributes_name_list | * }

Table 4: The top level syntax of WebML.

be used when the search is to browse the summaries at a high layer, mine description can be used when the search is to discover and describe the general characteristics of the data, mine classification is used to find classifications of web objects according to some attributes, whereas select remains to be a keyword indicating to find more detailed information. Two optional phrases, "related-to *name_list*" and "in *location_list*", are introduced in WebML for quickly locating the related subject fields and/or geographical regions (e.g., Canada, Europe, etc.). They are semantically equivalent to some phrases in the where-clause, such as "*keyword* covered-by *field_names*" and/or "*location* covered-by *geo_areas*", etc. But their inclusion not only makes the query more readable, but also helps the system locate the corresponding high layer relation if there exists one. The phrase "according-to *attributes_name_list* in-relevance-to attributes_name_list" is only used for classification with mine classification. It indicates the attributes upon which to classify web objects. The where-clause is similar to that in SQL except that new operators may be used.

While this query language is simple, users do not have to learn it and write queries. A Java-based or HTML-based user interface can easily be developed on top of WebML to avoid heavy instruction queries, and to provide a means for interaction based on field-filling and button-clicking. This is one of our future projects.

4.2 WebML Operational Semantics

WebML queries are applied and pertain only to a given VWV which uses an MLDB structure. According to Definition 2.1, a VWV has three major components: $\langle \mathcal{S}, \mathcal{H}, \mathcal{D} \rangle$. $\mathcal{S}$ contains the schema of the virtual view, $\mathcal{H}$ contains a set of concept hierarchies which are a set of partial orders of the form $(P, a, \leq)$ where a is an attribute defined in the schema, P is a set of values in the domain *dom(a)* and $\leq$ is a reflexive, transitive and anti-symmetric binary relation representing subsumption of values in P, and $\mathcal{D}$ is a set of relations containing descriptors and abstractions of resources on the Web. Relations in $\mathcal{D}$ are organized in levels where relations in each level abstract the relations in the lower levels. The relationships among the relations at different levels of $\mathcal{D}$ are outlined in a route map in $\mathcal{S}$. Besides the route map and the conventional schema definitions of the relations in $\mathcal{D}$, $\mathcal{S}$ contains a set of generalization paths each of which shows how a higher layered relation is generalized from one or a set of lower layered relations; formally, $r = \Pi_A(r_1 \bowtie r_2 \bowtie ... \bowtie r_n)$ where Π and $\bowtie$ are respectively the projection and join operators in the relational algebra, r is the relation at level L, $r_1 ... r_n$ are relations at level l such that $l < L$ and A is the attribute set of r. $A = \{a_1 ... a_k\}$ where a_i is an attribute from one of $r_1 ... r_n$ and the value of a_i is either its value at level l or an upper bound in its partial order $(P, a_i, \leq)$. A value x of attribute a_i is an upper bound of y in $P = dom(a_i)$ if $y \leq x$.

WebML queries pertain to the relations in $\mathcal{D}$ and utilize the partial orders in $\mathcal{H}$ as well as the generalization paths in $\mathcal{S}$. Note that the result of a WebML query is always a relation. The result relation can be at any level of the MLDB structure, and can be "intercepted" by a data mining process for further computation. Processing WebML queries is made straightforward by translating them into corresponding SQL queries and mapping values in query conditions into upper or lower bounds in the pertinent partial orders. Since WebML takes advantage of concept hierarchies using the four additional intrinsic relationships: *coverage, subsumption, synonymy*, and *approximation* presented in Definition 4.1, these primitives are converted into disjunctions of concepts as follows:

1. *COVERS* ($\supset x$): The coverage is replaced by a disjunction of ancestors of x, $Q \supset x$ such that $\forall\, q \in Q, x \leq q$ in partial order $(P, a, \leq)$ where a is the attribute of concept x. Q is the set of least upper bounds of x, noted $\dashv \{x\}$.

2. *COVERED BY* ($\subset x$): The subsumption is replaced by a disjunction of descendants of x, $Q \subset x$ such that $\forall\, q \in Q, q \leq x$ in partial order $(P, a, \leq)$ where a is the attribute of concept x. Q is the set of all lower bounds of x, noted $\models \{x\}$.

3. *LIKE* ($\simeq x$): The synonymy is replaced by a disjunction of synonym concepts Q from the alias list of concept x in the partial order $(P, a, \leq)$ where a is the attribute of concept x such that $\forall\, q \in Q, x \leq q \wedge q \leq x$. Q is the set of all synonyms of concept x, noted $\simeq \{x\}$.

4. *CLOSE TO* ($\sim x$): The approximation is replaced by a disjunction of sibling concepts of x, $Q \sim x$ such that $\forall\, q \in Q, q \leq u \wedge x \leq u$ and u is least upper bound of x and q in partial order $(P, a, \leq)$ where a is the attribute of concept x, $u = \sqcup\{x, q\}$. Q is the set of all direct siblings of x, noted $\sqcap \sqcup \{x\}$.

4.2.1 Processing WebML Queries

WebML queries are translated into SELECT-FROM-WHERE SQL queries with identical structure, except for additional conditions in the WHERE clause from the RELATED TO and IN WebML clauses. RELATED TO F, adds conditions on the subject field and can be substituted by "$\wedge$ $a$coveredby$F$" in the WHERE clause, where a is the attribute of F. IN L, adds conditions on the geographical location (i.e. Internet domain) and can also be substituted in the WHERE clause by "$\wedge$ $l$coveredby$L$" where l is the attribute location or web address. LIST, like SELECT, is translated into an SQL SELECT statement with the exception that LIST aims at the highest MLDB layer possible while SELECT directs its results to the lowest layer. MINE DESCRIPTION and MINE CLASSIFICATION queries are also translated into SELECT statements with the attribute list from the IN RELEVANCE TO clause. In both cases, the information is retrieved at the lowest layer (Layer-1) and either collected in a data cube for OLAP purposed in the case of MINE DESCRIPTION, or a decision tree is constructed for the retrieved data, based on class labels from the ACCORDING TO clause in the case of MINE CLASSIFICATION.

The major challenge of processing WebML queries in a VWV is to find the appropriate relations in the appropriate layer of the MLDB structure to execute the equivalent SQL queries. While the FROM clause indicates the lowest level relation containing descriptors of a given resource on the Internet (i.e. document, person, image, game, etc.), the RELATED TO and IN clauses add conditions to help pinpoint the appropriate MLDB layer to execute the query. If the field or location specified in the RELATED TO or IN clause is known in the route map in $\mathcal{S}$, the relevant generalized relation is selected as source for the query. Moreover, the highest level in the partial orders of the different concepts used in the query is used to indicate the MLDB layer to use. For a query W, C is the set of all concepts and terms used in the query W and H the set of all partial orders of concepts in C, c is the highest concept used in hierarchy P if $c \in C \wedge \forall\ q \in C, q \leq c$ with $(P, a, \leq)$ where a is attribute of c. $\forall\ P \in H, P$ has only one highest concept level in C. The set of all highest concept levels in C and the route map in $\mathcal{S}$ identify the MLDB layer to use as source of the query W.

4.3 WebML Examples

As mentioned earlier, the MLDB structure provides ground for resource discovery on the Internet (i.e. pinpointing relevant documents) as well as knowledge discovery (i.e. implicit knowledge extraction). Following are examples of queries for resource discovery and for data mining from the Web which illustrate the semantics of WebML.

Example 4.1 (*Query for Resource Discovery*) The query, *list the documents published in Europe and related to "data mining"*, is presented as follows.

LIST $*$
FROM document **IN** Europe
RELATED TO computing science
WHERE **ONE OF** keywords **COVERED BY** "data mining"

Notice that the keyword LIST indicates that the query is to briefly browse the information, and therefore, it searches the relations using the where-clause as a constraint. Using SELECT instead of LIST would locate a set of URL addresses of the required documents, together with the important attributes of the documents. The keyword LIST, however, allows to display document attributes at a high conceptual level and provides and OLAP-like interaction. "FROM *document*" does not indicate to find the document relation at layer-0 or layer-1, but indicates to find the top-most layer of the *document* relation which fits the query. Therefore, "*document*" is a clue to the system to find the appropriate relation at a high layer. We adopt this convention since it is the system's responsibility to find the best match, and it is unreasonable to ask users to remember all the relation names at different layers. Moreover, the RELATED TO clause can help the system locate the appropriate top layer relation in case the relations are split by topic. To execute this query, the VWV system uses the phrase "FROM *document*" and "related-to *computing science*" to locate the top layer relation, *cs_document* for example. The phrase, "ONE OF *keywords* COVERED BY *'data mining'*" means that there exists an entry in the set *keywords* which is subsumed under '*data mining*'. Moreover, the phrase "IN *Europe*" confines the search to be within *Europe* which will be mapped into concrete countries using a concept hierarchy for Internet domains. In this case, a relatively large set of answers will be returned. An interactive process to deepen the search will usually be initiated by users after browsing the answer set. □

Example 4.2 (*Query for Resource Discovery*) To locate the documents related to data mining topics and linked from Osmar's webpage, and then rank them by *importance*, a simple WebML query is presented as follows.

```
SELECT   *
FROM     document
WHERE EXACT "http://www.cs.sfu.ca/~zaiane" IN links_in
         AND ONE OF keywords COVERED BY "data mining"
         AND "Ted Thomas" IN authors
RANK BY INWARD, ACCESS
```

Notice that "SELECT *" means to print all the important attributes in a relation at a high layer, and moreover, "*'Ted Thomas'* IN *authors*" means that *'Ted Thomas'* is in the set of *authors*, whereas "ONE OF *keywords* COVERED BY *'data mining'*" means that there exists an entry in the set *keywords* which is subsumed under '*data mining*'. "RELATED TO *computing science*" is not necessary in this particular query since '*data mining*' is subsumed under '*computing science*', however, this clause can alleviate ambiguity in case the search term is subsumed under more than one ancestor. Moreover, the RELATED TO clause can help the system locate the appropriate top layer relation in case the relations are split by topic. The EXACT keyword specifies that the URL should be used as given in the comparisons, and not as prefix of potential URLs as we shall see in the next example. To execute this query, the VWV system uses the phrase "FROM *document*" and "RELATED TO *computing science*" to locate the top layer relation *cs_document* for example, and then uses the two search keys in the where-clause as well as the links-in set to locate a set of URL addresses of the required documents, together with the brief descriptions: *authors, title, publication, publication date,*

keywords, etc. The documents would be ranked by the number of hyperlinks linking to them from other resources and how often these documents are accessed. This translates the subjective term *importance* stated in the request. Users have the choice to either find more detailed layer-1 descriptions (i.e. drill-down), or directly access the documents by clicking at different buttons (drill-through). □

Example 4.3 (*Query for Resource Discovery*) To locate the documents about "Intelligent Agents" published at Simon Fraser University (SFU) and that link to Osmar's web pages in at least two depth link paths, the following query could be written:

SELECT $*$
FROM document IN "www.sfu.ca"
RELATED TO "computer science"
WHERE "http://www.cs.sfu.ca/~zaiane" IN links_out $(->$ OR $->->)$
AND ONE OF keywords LIKE "Intelligent Agents"

In this query the EXACT keyword was not used. This means that the URL given in the query could be used as a prefix to any address in the links_out set. Moreover, $\rightarrow$ and $\rightarrow\rightarrow$ are used to check recursively the links_out sets at a depth 2 for other links that verify the condition. Only local links are used since the URL and the SFU domain match; global links are not necessary. The 'IN "www.sfu.ca" ' phrase is used to limit the retrieval to the SFU domain, extracted from the Internet domain hierarchy. The LIKE keyword is used to match "Intelligent Agents" to other synonyms defined in the concept hierarchy.

This query returns a list of URL addresses together with important attributes of the documents that match. □

Example 4.4 (*Query for Knowledge Discovery*) To inquire about European universities *productive* in publishing on-line *popular* documents related to database systems since 1990, a WebML query is presented as follows:

SELECT affiliation
FROM document IN "Europe"
WHERE affiliation COVERED BY "university"
AND ONE OF keywords COVERED BY "database systems"
AND publication-year $>$ 1990
AND count = "high"
AND h(links-in) = "high"

In this query, "productive" is measured as those that published a *high* number of papers. The term "high" is a generalization of the numeric value of access-frequency along its concept hierarchy. While constructing the layers of the MLDB structure, concept hierarchies for numerical attributes are automatically built and labeled later by users. The label "high" would, for example, correspond to a count greater than 20, in other words affiliation with more than 20 published papers. "Popular" is measured by the high number of hyperlinks coming from other resources towards these papers. *links-in* in this case is not just counted

(cardinality of links-in set), but is surveyed by a heuristic function h, provided by the user, which calculates the popularity based on the importance of the links. For example, a local link would get the weight 0.5, a link from a resource related to the condition in the where-clause would get 2, while other global links get 1.

It is interesting to note that the execution of this query does not return a list of document references, but rather a list of universities (publishing popular documents about databases), which is implicit information (or knowledge) extracted from a conglomerate of documents. □

Example 4.5 (*Query for Knowledge Discovery*) Suppose the query is to "*describe the general characteristics in relevance to authors' affiliations, publications, etc. for those documents which are popular on the Internet and are on "data mining"*. A knowledge discovery query to answer this request, characterized by the keyword "MINE DESCRIPTION" is shown below:

MINE DESCRIPTION
IN RELEVANCE TO authors.affiliation, publication, pub_date
FROM document **RELATED TO** Computing Science
WHERE **ONE OF** keywords **LIKE** "data mining"
AND access_frequency = "high"

The discovery query will be first executed as a retrieval to collect from *cs_document* the data which are relevant to "*authors.affiliation, publication, pub_date*" and satisfy the where-clause. Then the attribute-oriented induction is performed on the collected data, which generalizes "*publication*" into groups, such as major AI journals, major database conferences, and so on, and generalizes publication date to year, etc. The generalized results are collected in a data cube and can be interactively manipulated by the user using OLAP operations. □

Example 4.6 (*Query for Knowledge Discovery*) To classify according to update time and popularity the documents published on-line in sites in the Canadian and commercial Internet domain after 1993 and about information retrieval from the Internet, a WebML query can be presented as follows:

MINE CLASSIFICATION
ACCORDING TO timestamp, access_frequency
IN RELEVANCE TO $*$
FROM document **IN** Canada, Commercial
WHERE **ONE OF** keywords **COVERED BY** "information retrieval"
AND ONE OF keywords **LIKE** "Internet"
AND publication_year $>$ 1993

The phrase MINE CLASSIFICATION requests a classification tree from the system. The query first collects the relevant set of data from the VWV relations, executes a data classification algorithm to classify documents according to their access frequency and their last modification date, then presents each class and its associated characteristics in a tree. The user can navigate the tree representation and drill through to the documents if needed. □

5 Conclusion and Discussion

Different from the existing global information system services, a new approach, called *virtual web views (VWV)* using *multiple layered database (MLDB)* structure, has been proposed and investigated for resource and knowledge discovery in global information systems. The approach is to construct progressively a global multiple layered database by generalization and transformation of lower layered data, store and manage multiple layered information by database technology, and perform resource and knowledge discovery by query transformation, query processing and data mining techniques. The Virtual Web View plays the role of a data warehouse for web content.

While in theory it is possible to create a unique global virtual web view that would summarize and represent the entire content of the World-Wide Web, it is neither practical nor desirable [54]. A VWV is based on concept hierarchies and it is very difficult to find a consensus on a general ontology. It is more realistic to build different VWVs specializing in different topics or restricted geographically, etc. VWVs can also share the same primitive data but use different ontologies (i.e. concept hierarchies). In such a context a software agent that plays the role of a mediator and broker between VWVs is necessary.

A web document querying language like WebOQL [21] or FLORID [55] is capable of retrieving information from HTML documents using graph tree representations or Datalog-like rules. Their powerful expressions can extract interesting and useful information from within a given set of web pages. We intend to use the power of such query languages to build our system's data model. In the preliminary experiments we conducted, we built the Virtual Web View of web pages containing computer science technical reports. The different layers were generalized using a pre-built ontology of computer science terms (Figure 7). WebML queries were prototyped on top of SQL and the results were very encouraging, demonstrating the resource dicovery and knowledge discovery capabilities of WebML. The ontology used allowed an interactive browsing of the corpus of documents with roll-up and drill-down options. In the future, we plan to use WebOQL and StruQL to build a VWV of a larger scale with a concept hierarchy automatically derived from the semantic network WordNet.

The major strength of the VWV approach is its promotion of a tight integration of database and data mining technologies with resource and knowledge discovery in global information systems. With the dynamically growing, highly unstructured, globally distributed and huge information base, the application of the mature database technology and promising data mining techniques could be an important direction to enhance the power and performance of global information systems.

Our study shows that the web data warehousing can be performed and updated incrementally by integration of information retrieval, data analysis and data mining techniques, information at all of the non-primitive layers can be managed by database technology, and resource and knowledge discovery can be performed efficiently and effectively in such a multiple layered database.

Enforcing a consistent standard for metadata on the Internet will simplify data exchange and the effective information extraction from on-line documents. XML and the Dublin Core initiative are new standards endorsed by many organizations. With these standards web data warehousing can start with a niche of documents and progressively add new resources

and sites when these standards are more widely used.

References

[1] Phil Bernstein, Michael Brodie, Stefano Ceri, David DeWitt, Mike Franklin, Hector Garcia-Molina, Jim Gray, Jerry Held, Joe Hellerstein, H. V. Jagadish, Michael Lesk, Dave Maier, Jeff Naughton, Hamid Pirahesh, Mike Stonebraker, and Jeff Ullman. The asilomar report on database research. *ACM Sigmod Record*, 27(4), December 1998. also available at: http://www.acm.org/sigmod/record/issues/9812/asilomar.html.

[2] Merit data. ftp://ftp.merit.edu/statistics/nsfnet.

[3] Robert H'obbes' Zakon. Hobbes' internet timeline. available http://www-personal.umd.umich.edu/~nhughes/htmldocs/timeline.html.

[4] Oliver A. McBrian. GENVL and WWWW: Tools for taming the web. In *Proc. 1st WWW Conf.*, May 1994. http://www.cs.colorado.edu/home/mcbryan/mypapers/www94.ps.

[5] David Eichmann. The RBSE spider - balancing effective search against web load. In *Proc. 1st WWW Conf.*, May 1994.

[6] M. L. Mauldin. *Lycos: Hunting WWW Information.* CMU, 1994. Available from *http://lycos.cs.cmu.edu/.*

[7] Martjin Koster. Guidelines for robot writers. Nexor Corp, http://web.nexor.co.uk/mak/doc/robots/guidelines.html.

[8] Martjin Koster. A proposed standard for robot exclusion. Nexor Corp, http://web.nexor.co.uk/mak/doc/robots/norobots.html.

[9] Martijn Koster. Aliweb - archie like indexing the web. In *Proc. 1st Int. Conf. on the World Wild Web*, May 1994. http://www.nexor.com/public/aliweb/.

[10] C. Mic Bowman, Peter B. Danzig, Darren R. Hardy, Udi Manber, and Michael F. Schwartz. Harvest: A scalable, customizable discovery and access system. Technical Report CU-CS-732-94, University of Colorado, 1994. ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/Harvest.FullTR.ps.Z.

[11] R.L. Read, D.S. Fussell, and A. Silberschatz. A multi-resolution relational data model. In *Proc. 18th Int. Conf. Very Large Data Bases*, pages 139–150, Vancouver, Canada, Aug. 1992.

[12] J. Han, Y. Fu, and R. Ng. Cooperative query answering using multi-layered databases. In *Proc. 2nd Int. Conf. Cooperative Information Systems*, pages 47–58, Toronto, Canada, May 1994.

[13] G. Piatetsky-Shapiro and W. J. Frawley. *Knowledge Discovery in Databases.* AAAI/MIT Press, 1991.

[14] J. Han, Y. Cai, and N. Cercone. Data-driven discovery of quantitative rules in relational databases. *IEEE Trans. Knowledge and Data Engineering*, 5:29–40, 1993.

[15] P. Danzig, K. Obraczka, D. DeLucia, and N. Alam. Massively replicating services in autonomously managed wide-area internetworks. Technical report, Technical report, 1994. Available from *ftp://catarina.usc.edu/pub/kobraczk/ToN.ps.Z.*

[16] G. Piatetsky-Shapiro, U. Fayyad, and P. Smith. From data mining to knowledge discovery: An overview. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–35. AAAI/MIT Press, 1996.

[17] Oren Etzioni. The wold-wide web: Quagmire or gold mine? *Communications of the ACM*, 39(11):65–68, 1996.

[18] Jody J. Daniels and Edwardina L. Rissland. A case-based approach to intelligent information retrieval. In *Proc. ACM SIGIR'95 Conf.*, Seattle, WA, USA, 1995.

[19] D. Hardy and M. F. Schwartz. Essence: A resource discovery system based on semantic file indexing. In *Proc. of the USENIX Winter Conf.*, pages 361–374, Berkeley, CA, 1993. ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/Essence.Conf.ps.Z.

[20] L. Lakshmanan, F. Sadri, and I. Subramanian. A declarative language for querying and restructuring the web. In *Proc. 6th Int. Workshop on Research Issues in data Engineering*, New Orleans, 1996.

[21] Gustavo O. Arocena and Alberto O. Mendelzon. WebOQL: Restructuring documents, databases and webs. In *Proc of ICDE Conf.*, Orlando, Florida, USA, February 1998.

[22] Jiawei Han, Osmar R. Zaïane, and Yongjian Fu. Resource and knowledge discovery in global information systems: A scalable multiple layered database approach. In *In Proc. Conf. on Advances in Digital Libraries*, Washington, DC, May 1995.

[23] Osmar R. Zaïane and Jiawei Han. Resource and knowledge discovery in global information systems: A preliminary design and experiment. In *Proc. First Int. Conf. On Knowledge Discovery and Data Mining*, Montreal, Canada, 1995.

[24] J. Shakes, M. Langheinrich, and O. Etzioni. Ahoy! the home page finder. In *Proc. Sixth World Wide Web Conf.*, Santa Clara, CA, USA, April 1997.

[25] R. Doorendos, O. Etzioni, and D. Weld. A scalable comparison-shopping agent for the world-wide web. In *Proc. Autonomous Agents ACM*, 1997.

[26] Alberto Mendelzon, George Mihaila, and Tova Milo. Querying the world wide web. In *Proc. PDIS'96*, Miami, December 1996.

[27] Oren Zamir and Oren Etzioni. Web document clustering: A feasibility demonstration. In *Proc. ACM SIGIR'98*, 1998.

[28] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *7th Int. Conf. WWW*, Brisbane, Australia, April 1998.

[29] S. Chakrabarti, B. Dom, D. Gibson, S.R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Experiments in topic distillation. In *ACM SIGIR workshop on Hypertext Information Retrieval on the Web*, Melbourne, Australia, 1998.

[30] J. Srivastava, R Cooley, M. Deshpande, and P.N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, 1(2):12–23, January 2000.

[31] Ellen Spertus and Lynn Andrea Stein. A hyperlink-based recommender system written in squeal. In *Proc. ACM CIKM'98 Workshop on Web Information and Data Management (WIDM'98)*, pages 1–4, Washington DC, November 1998.

[32] WordNet - a lexical database for english. http://www.cogsci.princeton.edu/~wn/, 1998.

[33] Osmar R. Zaïane, Eli Hagen, and Jiawei Han. Word taxonomy for on-line visual asset management and mining. In *Fourth International Workshop on Application of Natural Language to Information Systems (NLDB99)*, Klagenfurt, Austria, June 1999.

[34] P. B. Danzig, R. S. Hall, and M. F. Schwartz. A case for caching file objects inside internetworks. In *Proc. SIGCOMM'93*, pages 239–248, ACM, New York, September 1993.

[35] R. Alonso, D. Barbara, and H. Garcia-Molina. Data caching issues in an information retrieval system. In *ACM Transactions on Database Systems*, pages 359–384, 1990.

[36] Stuart Weibel, Jean Godly, Eric Miller, and Ron Daniel. OCLC/NCSA metadata workshop report (the essential element of network object description), March 1995. http://www.oclc.org:5046/conferences/metadata/dublin_core_report.html.

[37] Ora Lassila and Ralph R Swick. Resource description framework (rdf) model and syntax specification. W3C Working Draft, October 1998. http://www.w3c.org/TR/1998/WD-rdf-syntax-19981008/.

[38] Daniela Florescu, Alon Levy, and Alberto Mendelzon. Database techniques for the world-wide web: A survey. *ACM SIGMOD Record*, 27(3):59–74, September 1998.

[39] Serge Abiteboul, Dallan Quass, Jason McHugh, Jennifer Widom, and Janet L. Wiener. The lorel query language for semistructured data, 1997. http://www-db.stanford.edu/~abitebou/pub/jodl97.lorel96.ps.

[40] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A query language and optimization techniques for unstructured data. In *Proc. ACM SIGMOD Conf. on Management of Data*, 1996.

[41] Mary Fernandez, Daniela Florescu, Jaewoo Kang, Alon Levy, and Dan Suciu. Catching the boat with strudel: Experiences with a web-site management system. In *Proc. ACM SIGMOD Conf. on Management of Data*, Seattle, WA, June 1998.

[42] N. Ashish and C. Knoblock. Wrapper generation for semi-structured internet sources. In *Workshop on Management of Semistructured Data*, Tucson, Arizona, 1997.

[43] J. Han, Y. Fu, Y. Huang, Y. Cai, and N. Cercone. DBLearn: A system prototype for knowledge discovery in relational databases. In *Proc. 1994 ACM-SIGMOD Conf. Management of Data*, page 516, Minneapolis, MN, May 1994.

[44] Osmar R. Zaïane, Jiawei Han, Ze-Nian Li, Jenny Y. Chiang, and Sonny Chee. Multimediaminer: A system prototype for multimedia data mining. In *Proc. 1998 ACM-SIGMOD Conf. on Management of Data*, pages 581–583, Seattle, Washington, June 1998.

[45] R. Beckwith, C. Fellbaum, D. Gross, K. Miller, G.A. Miller, and R. Tengi. Five papers on WordNet. *Special Issue of Journal of Lexicography*, 3(4):235–312, 1990. ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps.

[46] Andrew Fall. *Reasoning with Taxonomies.* PhD thesis, School of Computing Science, Simon Fraser University, December 1996.

[47] Andrew Fall. The foundations of taxonomic encoding. *Computational Intelligence*, 14(4):598–642, 1998.

[48] R. Agrawal, S. Ghosh, T. Imielinski, B. Iyer, and A. Swami. An interval classifier for database mining applications. In *Proc. 18th Int. Conf. Very Large Data Bases*, pages 560–573, Vancouver, Canada, August 1992.

[49] D. Fisher. Improving inference through conceptual clustering. In *Proc. 1987 AAAI Conf.*, pages 461–465, Seattle, Washington, July 1987.

[50] B. de Ville. Applying statistical knowledge to database analysis and knowledge base construction. In *Proc. 6th Conf. on Artificial Intelligence Applications*, pages 30–36, Santa Barbara, CA, 1990.

[51] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems, 2nd ed.* Bemjamin/Cummings, 1994.

[52] H. F. Korth and A. Silberschatz. *Database System Concepts, 2ed.* McGraw-Hill, 1991.

[53] J. Han, Y. Fu, W. Wang, K. Koperski, and O. R. Zaïane. DMQL: A data mining query language for relational databases. In *Proc. 1996 SIGMOD'96 Workshop Research Issues on Data Mining and Knowledge Discovery (DMKD'96)*, pages 27–34, Montreal, Canada, June 1996.

[54] Osmar R. Zaïane. *Resource and Knowledge Discovery from the Internet and Multimedia Repositories.* PhD thesis, School of Computing Science, Simon Fraser University, March 1999.

[55] Wolfgang May. Modeling and querying structure and content of the web. In *Workshop on Internet Data Management in Proc. DEXA'99*, Firenze, Italy, 1999.