SEMANTIC INDEXING FOR COMPLEX PATIENT GROUPING

Kilian STOFFEL[†‡] Joel SALTZ[†‡], Jim HENDLER[†], Jim DICK[‡], William MERZ[‡] and Robert MILLER[‡]

[†]Computer Science Department
University of Maryland
College Park, 20742, MD

[‡]Johns Hopkins Hospital
Department of Pathology
Baltimore, 20885, MD

(stoffel, saltz, hendler)@cs.umd.edu
(jdick, wmerz, rmiller)@pds.path.jhu.edu

In this paper we describe indexing techniques based on domain knowledge made available in the form of ontologies. In high level interfaces like those used in many data warehousing applications, it is advantageous to use terminology familiar to the end-user of the system. This terminology is very often different from the one incorporated in the underlying database. Much of the terminology (i.e. concepts) in an end-user ontology will map to complex collections of data warehouse attribute value pairs. We create a mapping between the end-user terminology and attribute-value pairs in the data warehouse. We optimize performance by indexing the data warehouse using an ontology. We optimize performance by indexing the whole database using an ontology that represents the end-user's terminology.

INTRODUCTION

A joint effort between computer scientists at the University of Maryland and clinicians at Johns Hopkins Hospital focuses on using high performance computing technology in support of medical applications. One focus is on providing tools that support medical data warehousing. One particular project focuses on providing database support to microbiologists and infectious disease specialists.

The taxonomy used to describe microbiological data is complex; this terminology also changes with time. A problem we have encountered is the difficulty clinicians and researchers face in formulating queries that capture the kinds of questions they wish to pose.

As a means for overcoming this sort of difficulty, we are looking at how to efficiently integrate ''semantic" knowledge, stored in the form of thesauri (or more technically, ontologies) in ways that support efficient indexing of large databases. In particular, we are working to provide medical specialists with the ability to express more complex queries without becoming experts on the underlying data model. In this paper we show how we can support this efficient indexing, facilitate complex data access, and support high-level querying for users untrained in the details of the underlying database forms.

MOTIVATING EXAMPLES

As stated above, our examples are motivated by a group of applications that involve the analysis of information in a medical data warehouse we are constructing at Johns Hopkins Hospital. The data warehouse includes in-patient and outpatient data from the microbiology, blood bank, clinical chemistry and hematology laboratories, along with pharmacy data and some clinical data. The microbiology data is obtained through a large and varied group of identification techniques, which are generally carried out in a stepwise manner. Protocols are then followed to produce a more precise identification.

The scope of this paper precludes a detailed discussion of microbiological taxonomy [1]. A bacterial species can be defined by

1. structural attributes of size, shape, Gram stain reaction and macroscopic growth,
2. physiologic traits relative to oxygen, temperature, pH,
3. biochemical and nutritional traits,
4. biochemical information on cell composition and metabolites, and
5. DNA sequence information.

Microbiological identification requires a process of iterative refinement; there is not a single battery of tests that can be applied to all specimens to obtain a precise identification.

The set of protocols required to precisely identify an organism can be represented as a directed acyclic graph. Most organisms are not precisely identified as the iterative process of organism identification continues only as long as clinically useful information is obtained. A highly imprecise identification is permissible as long as no clinical decision hinges on a more complete identification. Identical organisms may be identified to different degrees of precision as clinical requirements may vary by anatomic site or from patient to patient.

The complex taxonomy used to describe microorganisms and the non-uniform degree of microorganism identification create difficulties for those who wish to pose queries to a clinical data warehouse. We will present some realistic examples of queries that are of clinical interest at Johns Hopkins Hospital. The clinical context associated with these examples is an ongoing effort to characterize microorganism antibiotic susceptibility. This issue is of continuing concern because of the continuing emergence of antibiotic resistant microorganisms. In the examples below, note that the minimal inhibitory concentration (MIC) of an antibiotic is the lowest concentration of antibiotic that inhibits visible growth.

Produce a histogram of the minimal inhibitory concentrations to the antibiotic Ciprofloxacin for all specimens that grew out gram negative non-lactose fermenting bacteria.

More precisely, the task is to: (1) report the Ciprofloxacin minimal inhibitory concentrations of all specimens that grew out organisms that belong to each category C (sub-concept) of gram negative non-lactose fermenting bacteria, and (2) produce a histogram for each category C grouped by MIC value.

Note that in this query, the user doesn't have to know which bacteria are supposed to ferment lactose. Furthermore, as long as the ontology defines a species as being one that ferments lactose, it does not matter whether or not the protocol used to identify the bacteria actually involved evaluating whether or not a specimen fermented lactose.

Which antibiotics are effective against 80% of organisms recovered from cultures that grew out any species of Enterobacteriaceae?

In this query, we want to find out which antibiotics are effective against 80% of specimens belonging to the concept "Enterobacteriaceae" or 80% of the specimens belonging to any sub-category of Enterobacteriaceae?

Note that we need to define what is meant by an effective antibiotic. In the context of a clinical laboratory, MIC values are used to evaluate antibiotic effectiveness. No single MIC threshold defines effectiveness for all antibiotics; each antibiotic has its own MIC threshold.

Our tools support queries that characterize the effectiveness of antibiotics with respect to many possible categories of organisms. The two categorical terms (antibiotics and Enterobacteriaceae) allow us to discuss two types of categories. The first term "Antibiotics" is replaced by a list of all antibiotics used in the tests. The second term "Enterobacteriaceae" is replaced by all its sub-categories. Sub-categories of Enterobacteriaceae are not disjoint. For instance, *Citrobacter* constitutes one subcategory and *Citrobacter diversus* and *Citrobacter freundii* constitute two additional sub-categories.

Which groups of patients have been infected by organisms that have been proven to be increasingly resistant to antibiotics?

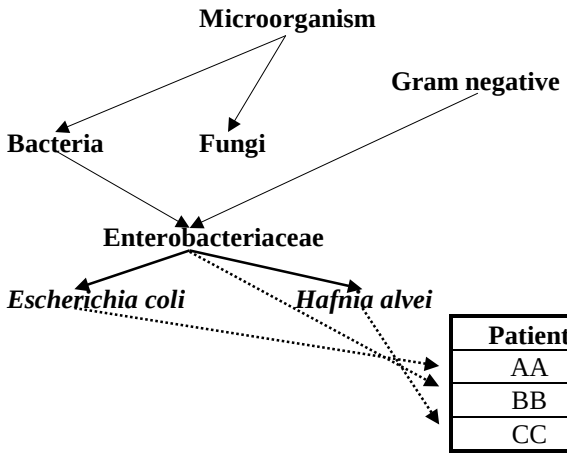
For a given group, we want to identify classes of organisms that exhibit an increase in resistance, over time, to specific antibiotics. This scenario assumes that we have an ontology that specifies a set of groups to which each patient belongs. Examples of such groups could include a neighborhood, school and workplace (in an outpatient setting) and might include hospital unit and service in an in-patient setting. Other groupings based on demographic data (e.g. age) would also be possible. Furthermore groups would also include potentially relevant diagnostic categories or current treatment with a therapy that involves immunosuppression.

SEMANTIC INDEXING

In this section we describe a sophisticated indexing schema which allows us to support the kinds of queries described here in an efficient way. Our indexing scheme is based on *ontologies*: taxonomic information with additional links that represent associated properties.

Ontologies

The most commonly used data structures employed to represent ontologies are DAGs (directed acyclic graphs). A node in the DAG is called a *concept* and represents a specific object or action. The directed links pointing from one concept to another define the concept/sub-concept relationships.



Patient	Microorganism	Age
AA	*Escherichia coli*	34
BB	*Enterobacteriaceae*	52
CC	*Hafnia alvei*	19

Figure 1: Relation between an Ontology and a data base.

Some ontologies represent very general knowledge (WordNet) [2] ,other ontologies specifically target medicine (e.g. SnoMed [3] and UMLS). [4] We define a mapping between the attributes used in the data warehouse and the terminology represented in an ontology. Users employ terms defined in an ontology when generating data warehouse queries. The use of ontologies to help users select terms needed in their queries has been described by several researchers, see for instance [5].

Ontologies can also provide a way to group records of a database in a semantically meaningful way. This type of semantic grouping can be used to optimize query performance. We anticipate that experts will frequently access data using groupings defined in an ontology. The ontology can be used to create indices which allow us to retrieve data grouped by ontological concepts. Just as b-trees permit the retrieval of a range of data, we are able to retrieve a set of records which are semantically associated with a concept in an ontology. These optimizations make it practical to develop a tool for formulating complex queries such as the queries depicted in Section 2.

USING ONTOLOGIES FOR INDEXING

Indexing

In order to use the ontologies for indexing we have to establish links between the data in the data warehouse and the concepts in the ontology. All pairs of attributes and values in the database[†] are mapped to concepts in the ontology. This mapping requires the use of a data dictionary to translate database attribute value pairs to ontology concepts. An example is given in Figure 1. The solid links are ontological links and the dashed links are indexing links.

Thus, for example, we have indexing links from the ontology concepts ***Escherichia coli*** and ***Hafnia alvei*** to tuples 1 and 3. We also have an indexing link from the ontology concept **Enterobacteriaceae** to tuple 2. Tuples 1 and 3 each record a specific microorganism genus and species. Tuple 2 records a microorganism that is identified only as a member of the family Enterobacteriaceae. Note that *Escherichia coli* and *Hafnia alvei* are also members of the family Enterobacteriaceae, but we only index the concept that maps directly to an attribute value pair.

Data Structures and Algorithms

Data structures: Each concept consists of three components: (1) The *concept ID*, (2) a list of sub-concepts, and (3) a list of pointers to database tuples. The sub-concept list associated with concept C is a list of pointers into the ontology file that identifies all sub-concepts of C. The list of pointers to database tuples identify *instances* of concept C. These basic data structures are used by the following algorithms:

Instances: This operation retrieves all the *direct* instances of a concept.

[†] Throughout this paper we will make the simplifying assumption that data is stored in a universal relation. This is not a necessary condition, but simplifies the discussion of the basic ideas.

```
1: A := TRANSITIVE_SUBCONCEPT(Enterobacteriaceae)
2: B := TRANSITIVE_SUBCONCEPT(ANTIBIOTIC)
3: C := {susceptible, very susceptible }
4: for each c in C
       D := D union TRANSITIVE_INSTANCE(c)
5: for each a in A
6:    for each b in B
      E[a,b] := TRANSITIVE_INSTANCE(b)∩ TRANSITIVE_INSTANCE(a) ∩D
      report (a,b) if    (count(E[a,b])/count(TRANSITIVE_INSTANCE(a)))>0.8
```

Figure 2: Pseudocode for Example 2.

Sub-Concept: This operation retrieves the *sub-concept-list*.

Transitive Sub-Concept: This operation returns all sub-categories reachable from a concept by following all possible directed links.

Transitive Instance: Transitive instance is a combination of the transitive sub-concept operation and the instance operation. In the first step, all reachable sub-concepts are collected. In the next step, all instances of these concepts are gathered.

DISCUSSION OF THE EXAMPLES

In this section we present a more detailed description of Example 2. Space limitations prevent us from presenting detailed discussions of the other examples.

Example 2

Which antibiotics are effective against 80% of organisms recovered from cultures that grew out any species of Enterobacteriaceae?

Figure 2 depicts the steps that must be carried out to implement Example 2. In Step 1 we calculate all transitive subconcepts of "Enterobacteriaceae" and assign this to Set A. Set A will then contain all microorganism names that are subconcepts of "Enterobacteriaceae". In Step 2 we assign to set B the transitive sub-concepts of the concept "Antibiotic". In Step 3 we create a simple set with the two concepts "susceptible" and "very susceptible". The union of all the transitive instances of "susceptible" and "very susceptible" is then assigned to set D in Step 4. Set D thus contains all patient records where susceptible bacteria were found. We then intersect the set D with: (1) all transitive instances of each sub-concept of "Antibiotic" and (2) all transitive instances of all sub-concepts of "Enterobacteriaceae". This yields a collection of sets E(a,b). Each set E(a,b) contains pointers to patient records for each Enterobacteriaceae sub-concept a and each antibiotic sub-concept b. Finally, in step 6, we report (a,b) only if 80% of the cultures that grew out Enterobacteriaceae sub-concept a proved to be susceptible or very susceptible to antibiotic b.

In our database, we maintained results for 27 antibiotics and we store 65 sub-concepts of Enterobacteriaceae. Thus Example 2 allows us to evaluate antibiotic susceptibilities for 1755 different combinations of organism and antibiotic. This query required roughly 40 minutes on a 150 MHz Pentium PC using a database with 20521 records of organisms. The scope of this paper precludes a detailed discussion of results obtained. We found, for instance, that over 80% of our Enterobacteriaceae were susceptible or very susceptible to amikacin, ciprofloxacin, ceftazidime, cefuroxime, gentamicin, piperacillin, trimethoprim/sulfa, ticarcillin, and tobramycin. For all species of *Citrobacter* taken together, only amikacin, ciprofloxacin, gentamicin, and tobramycin were effective against 80% of specimens. However, for *Citrobacter diversus* three additional antibiotics were effective in 80% of specimens - ceftazidime, cefuroxime and trimethoprim/sulfa.

In order to execute this query without using the indexing scheme, most of the 1755 different combinations of organism and antibiotic would have to be accounted for through large disjunctive queries.

RELATED WORK

The integration of ontologies into data base systems is of growing interest. Due to the space restrictions we are not able to provide here an overview of all the different approaches. Ullman presented a selection of these ideas in [6].

There are a wide variety of projects that address issues associated with making it easier for users to formulate medical database queries (e.g. . [5, 7, 8]). Our focus is somewhat different as we provide a powerful (but not simple) user interface for automatic generation of very large sets of related queries.

Somewhat similar approaches to the work presented in this paper can be found in some of the

more powerful search engines that target text data bases; these search engines are mainly used in WEB browsers. [9] [10] In our approach, we apply ontology indexing schemes to relational databases rather than to text databases (although in future work we will extend our work to a combination of relational and text databases). Another important difference is that we support transitive closure operations on concepts and instances. This functionality increases flexibility and makes it possible to carry out sets of closely related queries in an optimized fashion.

Complex indexing schemes are also known in the deductive data base community. A good overview of these systems is given in [11]. These systems offer more functionality than the system we propose but do not scale well with increasing database and ontology size. .

CONCLUSION AND FUTURE WORK

We presented an efficient semantic indexing scheme for complex grouping operations. We showed this scheme could be used for integrating ontological and relational data. A prototype of the system is currently being tested in the Johns Hopkins Hospital, the prototype is being used to track the spread of antibiotic resistant bacteria, evaluate patterns of antibiotic use, and to screen for nosocomial infections. In the near future, we plan to parallelize the prototype to allow it to function in an interactive manner. In past work, we have parallelized many of the computational components employed in this prototype[12] and we anticipate that parallelizing our prototype should be relatively straighforward. We are also currently exploring the use of these indexing techniques in other domains in which complex data retrieval is used and where ontologies can be generated.

REFERENCES

[1] I.Bergey. "Systematic Bacteriology". Williams and Wilkins, 1984.

[2] George A. Miller. "Human language technology". Technical report, Psychology Department, Green Hall, Princeton University, 1996.

[3] Roger Code. "Systemized Nomenclature of Human and Veterinary Medicine: SNOMED". College of American Pathologists, American Veterinary Medical Association, 1993.

[4] UMLS. "Unified Medical Language System". National Library of Medicine, 1994.

[5] George Hripcsak, Barry Allen Janes J. Cimino and Rober Lee. Access to Data: Comparing AccessMed with Query by Review. Journal of the American Medical Informatics Association, Volume 3, Number4, Jul/Aug 1996.

[6] Jeffry D. Ullman. The database approach to knowledge representation. Proceedings of the 13th National Conference of the American Association for Artificial Intelligence. AAAI Press, MIT Press, 1996.

[7] C. Safran, D. Porter, J. Lightfoot et al. CliQuery: a system for online searching of data in a teaching hospital. Ann Intern Med. 1989;111;751-756

[8] B. Leao, E. Reategui. HYCONES: a Hybrid Connectionist Expert System. SCAMC proc. 1993;461-465.

[9] altavista-web. http://altavista.digital.com/av/lt/help.html, Altavista.

[10] medline-web http://www.medline.com, Medline.

[11] Jose Alberto Fernandez, Jarek Gryz, and Jack Minker, Disjunctive deductive databases: Semantics, updates and architecture, In Proceedings of the 4th Bar -Ilan Symposium on Foundations of AI, pages 256--274. AAAI Press, 1996.

[12] J. Saltz, G. Agrawal, C. Chang, R. Das, G. Edjlali, P. Havlak, Y.S. Hwang, B. Moon, R. Ponnusamy, S. Sharma, A. Sussman and M. Uysal, Programming Irregular Applications: Runtime support, Compilation and Tools, In Advances in Computers,Academic Press, Vol 45, 1997**.**