DESIGN OF INTELLIGENT QUERY SYSTEMS FOR LARGE DATABASES

Bharat Bhargava
Computer Science Department
University of Pittsburgh
Pittsburgh, PA 15260

ABSTRACT

In this chapter, we present techniques that allow a query system to play an active (or intelligent) role in communicating knowledge in large databases to the user via meaningful and efficient feedback during query execution. Our approach is to dynamically create temporary files and access paths for information relevant to present query and inform the user of the existence of such information (if security is not compromised). Our research and experience shows that such information can be made accessible at a very low cost when the system is obtaining the data that has been requested and can be presented (if user shows interest) to the user without much effort. Four types of semantics (database semantics, database organization semantics, usage semantics, and real-world semantics) have been identified, their sources, appropriate data structures for their representation, and applicability have been presented with examples from medical and pictorial databases. Our hypothesis for this research is that most users cannot be expected to know all necessary information available in a large database and the query system must play an intelligent role and provide hints to the users.

I. INTRODUCTION AND DEFINITION OF PROBLEM:

In image processing applications, a large quantity of data is gathered to cover all possible inferences that can be drawn in the future. In present systems the cost of a meaningful conclusion derivable from this data is proportional to the size of database rather than the amount of information that is necessary for reaching the conclusion. Large databases offer the potential for analysis, retrieval, and use of complex information that is modelled by the database. Because of sheer volume of data, many problems arise that leave this potential largely unrealized. Among these problems of interest to us are summarized as follows

a) A user is usually interested in retrieving or updating a very small subset of data but the probing in the database is extensive and the system causes the query to be executed slowly.

b) Database Management System (DBMS) is unable to consider the characteristics of current users and so uses predefined, inefficient algorithms in all cases.

c) The user is unaware of potential information available in the database and presents his intent in a set of unstructured queries. The DBMS takes a passive approach and tries to respond to each individual query rather than set up a strategy for efficient retrieval or update.

In order to improve the utility of large databases, we see a need to make user-database interfaces responsive, cost-effective and efficient for all users, in particular the casual users. A casual user is defined as the user who has no information about the data model and its implementation and articulates precisely the information he needs at any particular point in an interactive mode.

A complete user-database interface system can then be expected to take up the following tasks:

a) Accept a question expressed in a natural language subset (not necessarily grammatical) in the decision-maker's own term (in some cases fuzzy).

b) Determine what information is being requested from the database. In case of syntactically incomplete or semantically ambiguous question, clarify the intent of question by talking back in the language of user or in very simple terms.

c) Determine the entities and operations that are required on the entities. Make sure the operations are valid within the data model.

d) Establish an optimal access path that will link up the entities with actual database organization files.

e) Derive the requested information by manipulating the database, and present this to the user.

f) Determine any other entities in the database that have relationship to the entities being considered in the present query.

g) At execution time, determine if any additional information in the database can be useful to the user in context of present query. If so, create auxiliary files and indices for such information and indicate the availability of such information to user if security is not compromised.

h) If desired by the user, present additional information obtained in g).

Recently several query subsystems (12, 18, 26) have allowed the user to retrieve data from modestly sized databases by means of restricted natural language queries. These systems have demonstrated that natural language access to databases in a limited domain is technically feasible and reasonably economical. This research deals with tasks a) and b). While it seems desirable to allow free-form Englishlike input, we cannot expect computers will understand, without much overhead, any language that even humans cannot express without ambiguity and use of context. Though it is true that in a formatted database that deals with a limited problem area in a well defined context, it is possible to avoid many ambiguities of natural language. In addition, the systems developed for experimentation have acquired special capabilities such as ability to talk back to the user to clarify the intent of presented query, tolerance to minor spelling and grammatical errors, interpretation of utterances in the dialogue, implementation of ellipses, paraphrases, etc. Essentially, the techniques employed are from the area of natural language parsing and formal language theory to analyze the syntactic and to some extent semantic content of user's query.

As a next step, to allow the user freedom from having to know details of data model and its implementation, user's information models are developed. These models incorporate user's knowledge of the meaning of the databases. The knowledge consists of user-defined database properties such as functional and semantic interrelationships among database entities, domain specification for the attributes, content description for files, etc. The model has been used for fuzzy query analysis (10) and establishing optimal access paths (8, 24). It is interesting to note that the semantic knowledge of the database can be managed in many different ways. In (29), Wiederhold discusses this issue in an elegant manner. The choice of the semantic knowledge is critical for the performance of the query system. One obvious choice used by IDA (24) is to store this information in the conceptual schema away from the database. For very large databases such division becomes untenable; data which are valuable for knowledge based processing are also used for routine data processing. Duplication of the information is costly and will make the updating very costly. The second choice, used by TORUS (21), is to represent all knowledge in the database. The drawback of this approach is that the representation for a large domain could be an extremely complex structure to create, and the inference mechanisms needed to answer queries and verify updates would be potentially very slow. Such research effort covers tasks c) and d).

In this paper, we will discuss the tasks from e) through h). The concept of utilizing focus in understanding dialogue (15, 16) and the work done in optimizing access paths for query execution (19) is applicable to this research. Our intention is to make the query system take an active role in understanding the user's intent and his need for information. Furthermore, the query system will analyze the database entities and their values together with the semantic knowledge of relevant portion of the database to provide useful feedback about the database yet unknown to the user.

II. DESIGN TECHNIQUE

Irrespective of whether the query language is formal or natural it is very difficult for every user to be fully knowledgeable about the structure, semantic, and content of a large database. This fact is particularly true if the database is constantly being updated and modified by different users at several sources. Moreover, an instant communication exchange among several users cannot easily be cost justified.

For example, a radiologist enquiring a medical database is not fully aware if another doctor has ordered a blood test on the patient and the test results are available or if the patient has been prescribed any new drugs.

Similarly, in picture databases, a user has to scan through the picture frames exhaustively before he can know whether the relevant information exists and if so in what part of the database. Thus, the user interested in a subset of database in current query systems has to formulate a series of specific queries and only anticipate that he is moving in the right path to the desired information.

In addition, the user may be scanning repeatedly a close vicinity of a subset of database and unknowingly the same file in successive questions, thus increasing the I/O accesses to the database. This approach is failure prone because in some cases the user may obtain incomplete information even though other useful knowledge exists in the database and was not specifically requested by the user. Moreover, this approach is time consuming from both the user's and machines point of view.

Most query systems in present use display the information specifically requested by the user. Question-answer systems using deductive inference have been developed to aid the user. But these systems limit their task to understanding the user's query and make no attempt to provide any additional input from the information stored in the database.

We believe that if the databases query system can present not only what the user has asked for but some additional information that is related to entities in the present query, the user will use the system efficiently. This idea is derived from human behavior where for each question, the intelligent interviewee (replier) does not just give an answer for the question asked but supplements it with other useful, relevant information. For example, a professor usually tells a student about related courses and research papers along with answering what students wanted to know about a particular subject.

For this type of intelligent query system, we need some semantic information along with the actual database. Semantic relationships may be predefined or may be conceived during the process of extracting information of the database. Thus, some semantic knowledge would be time-independant (static) and can be determined in the design process.

For example, functional dependencies (11) among attributes falls in this category. Other semantic knowledge can only be known at the run-time and hence, mechanisms to collect and use them have to be incorporated in the system. User's domain of interest evident from a set of his queries will be an example of such semantics.

The dynamic mode of information gathering increases the retrieval cost at the critical query processing time, but the cost can be spread over the use of the system if the same user plans to return for similar queries. The dynamic semantics can be incorporated in the static category if the frequency of use justifies.

We have identified several types of semantics that are involved in a user-database interaction. They are classified in four categories:

a) Database Semantics
b) User's Knowledge (Usage Semantics)
c) Real-world Semantics
d) Database Organization Semantics

In the following sections, we describe them in detail and give examples to illustrate their impact in the design of an intelligent query system. Research in artificial intelligence (AI) that deals with knowledge representation, deduction and inference procedures, and implementation of semantic networks is relevant to our work but has not been included in this paper. For example, symbolic modelling, a well known technique in artificial intelligence can be both efficient and easy to use. Similarly, semantic networks have been used in both AI and database modelling.

(II.1) Database Semantics

The translation from information to data is not completely reversible. Through one attempt to capture the semantics or contexts in which the information is generated, yet some of the semantics is lost in the conversion to data. Where large databases are used, this issue is particularly important since we expect the database to outlive the database generators and its users.

For the datamodel to reflect the real-world structure, it is necessary that not only we include in our model the database description but some related semantics. The normalization theory discussed in relational model is one vehicle to capture the database semantics to eliminate update and delete anamolies in the database use. The concept of "database skeleton" developed by Chang (10) is another approach to incorporate semantic information about the relationship among database states. The "database skeleton" has been used for fuzzy query analysis. There are two types of data semantics:

a) Semantics of Data Description
b) Semantics of Data Values

(II.1a) Semantics of Data Description

This knowledge consists of relationships among database entities, among attributes, and among entities and attributes. The functional dependencies among attributes is an example of this. Another example where such knowledge is employed is the production rules of a picture grammar.

Semantics of data-description is a well studied topic and all database models try to make use of it. The hierarchical model of database utilizes the partial ordering among entities to establish hierarchical structure. Wiederhold et. al, (30) have developed a structural model for databases which considers four types of entities,

a) Primary
b) Reference
c) Association
d) Nest

The relationships among these entities model the correct update rules for them. For example, primary entities can be updated independently of other entities, but a deletion of a reference entity must ensure that all primary entity references must be also deleted. Thus Wiederhold's model utilizes data description semantics to affirm structural integrity of the databases. This type of semantics has also been employed in designing Intelligent Data Access (IDA) module of Ladder system developed by SRI (24). Utilizing such semantics, IDA builds a complete sequence of navigational paths necessary for a query execution, before an interaction with the database takes place. It also selects an optimal sequence of file accesses based on cost which is a function of the number of relations (files) and links among these that are involved.

The data description knowledge is basically time-independent and can be easily incorporated in the conceptual scheme. Another approach is to store such semantics as a set of axioms or frames as utilized in IDA. Chang (10) proposes the utilizing of an information graph where nodes represent the attributes and edges represent the semantics of relationships. In Wiederhold's model, the data structure to store these relationships is the same as that required for storing the actual data and hence, a common data definition can be used.

(II.1b) Semantics of Data Values

Semantics of data values signifies the inter-relationships among information that these values represent. These inter-relationships are essential so that the query system does not just present the information requested by the user but also determines intelligently the interest domain of the user-query session. Such domain helps the system in presenting additional information to the user at very low cost and in some ways leads the user to a set of useful database values (without diverting user's attention to unnecessary details).

A) Time Domain
B) Aggregation Domain
C) Generalization Domain
D) All of the above

For example, if the weather (e.g., amount of rain) in a particular area has a relationship to the density of crops, when analyzing satellite photographs of the area's agriculture fields, this knowledge can be effectively used. Similarly when studying an x-ray picture, if the size of heart has an effect on the curvature of ribs which in turn has an effect on the functioning of lungs, whenever an enlarged heart is observed, its effect on breathing should be brought into immediate focus.

The data value semantics is very useful in guiding the semantic checking of database updates and thus helps in maintaining database integrity. If for some reason, certain data values are erroneous, the computer feedback that utilizes interrelationships among different types of entities, can draw immediate feedback to such facts. This is particularly useful in instances where users are making retrievals as well as updates. For example, in a medical information system, if a clerk enters by mistake the daily drug dosage of 500 mg (abnormal) instead of 200 mg, the clerk can be immediately informed of abnormality. In case the clerk ignored it, the physician will know it when he inquires the database for any entity value (such as a blood test) related to the entity drug. Similarly in a speech database if a particular type of utterance is now allowed, this information is treated as data value semantics, and can help in removing noise from the database.

Database values can be used to trigger chains of relationship to increase the focus of interest. Thus the query system can bring forth finer details of database characteristics that may be interesting to user more than abstract level detail. For example, a query system using graphics for pictorial database can identify several parts of the picture as of interest to user and thus can present relevant frames in succession. The user can then select useful frames and expand his focus from there.

We have identified three dimensions of the focus domain and they are described in the following paragraphs.

A) <u>Time Domain</u>

A component of the knowledge that the system can make use of already exists in all databases. This knowledge is obtained via the time-domain and can be referred to as the trend in database values. If a particular entity is referred in the user's query, the query system can automatically check values of this entity over a period of time. The user can then be presented the data value as well as the trend. An example where this knowledge is frequently used is the chest x-ray reports. Not only is the current abnormalities important to a physician but also if any new one has appeared or else vanished.

B) <u>Aggregation Domain</u>

This component of the data value semantics can be predefined or else dynamically created at the time of data gathering. For example, the knowledge that several different entities and their value are created at a particular instance of (an instance could be defined as the same day) can be used for creating auxiliary files. If the user's query makes a reference to one of these entities, certain characteristics of other simultaneously created entities could be automatically checked. For example, if chest x-ray and blood cholesterol were ordered by a physician on the same day, a query for chest x-ray result can also present blood cholesterol level if it is in abnormal range.

C) <u>Generalization Domain</u>

The third dimension of data value semantics is the generalization view of the data. All entities in the database can be grouped into classes on the basis of functional dependencies, usage patterns, etc. These classes can be ordered to give hierarchical view of the database. The entities of higher levels could be abstraction of the entities at the leaf level. Now if the user asks for a data value at one level under a particular node, the query system can automatically prepare itself to present to the user similar answers for other entities at the same level. We illustrate this point by an example of the database of household goods. The various entities are organized hierarchically as shown in Figure (1). If a customer needs a table, the system automatically determines, the customer may be interested in the bedroom furniture and the kitchen appliances, too.

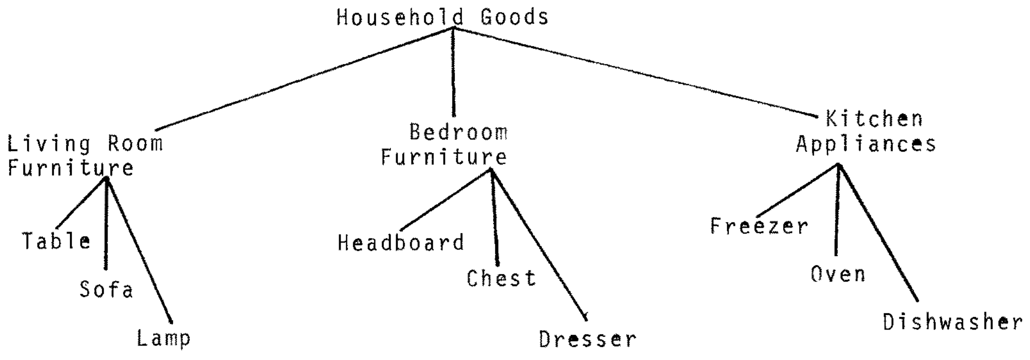

Figure (1)

Finally, a query system can be designed to take into account one or all three dimensions of the data value semantics. Thus this knowledge is in three dimensional domain as illustrated in figure (2).

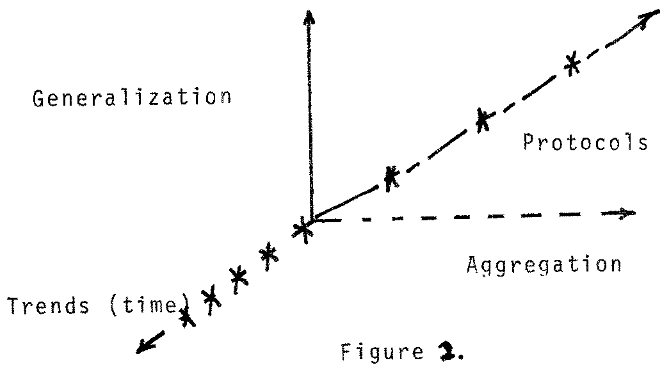

Figure 1.

<u>Representation of Data Value Semantics in the Query System:</u>

One representative of data value semantics is through a set of protocols predefined and stored in a module in the database management system (6,20). A protocol is very similar to the production rules used in Artificial Intelligence representation. The protocols monitor the database during the execution of the query and trigger printing of suggestions for the user if certain conditions are satisfied. A protocol consists of a condition and an action clause.

We briefly describe a knowledge representation that we used in a query system used for a medical information system. The condition clause is either the presence of an entity in the database, or an entity value in a particular range, or a change in entity value over a defined period of time. The condition clauses are linked by boolean operators "and","or", in addition to temporal connectives "and followed by", "and preceded by", and "coincide with." The temporal connections are used to relate two conditions on a time domain. For example, "BUN greater than 25 and patient on DIGOXIN" is a valid condition clause.

The action clause consists of a suggestion to do something as a result of a condition and the justification for that suggestion. For example "Reduce DIGOXIN because of increased risk of toxicity" is a valid action clause.

"THEN" and "ELSE" can be used to relate conditional clauses to action clauses. An example of a protocol is

Condition	IF HCT 38 or HGB 11.5
Suggested Action	THEN ORDER BLOOD INCIDES TO $\underbrace{\text{diagnose Anemia}}_{\text{Justification}}$

The conditions that are checked pertain specifically to the entities referred by the user in this query. For example if a user requested a retrieval on a blood-glucose test result, the protocols based on entity blood-glucose will be triggered. But if the request was for a seven-day summary report from the patient's medical record, the system will do condition-checking on all entities (blood tests, radiology examinations, etc.) that are retrieved. The protocol's corresponding to these entities will be triggered at run-time and a list of suggested actions is printed along with the summary report (see Figure 3).

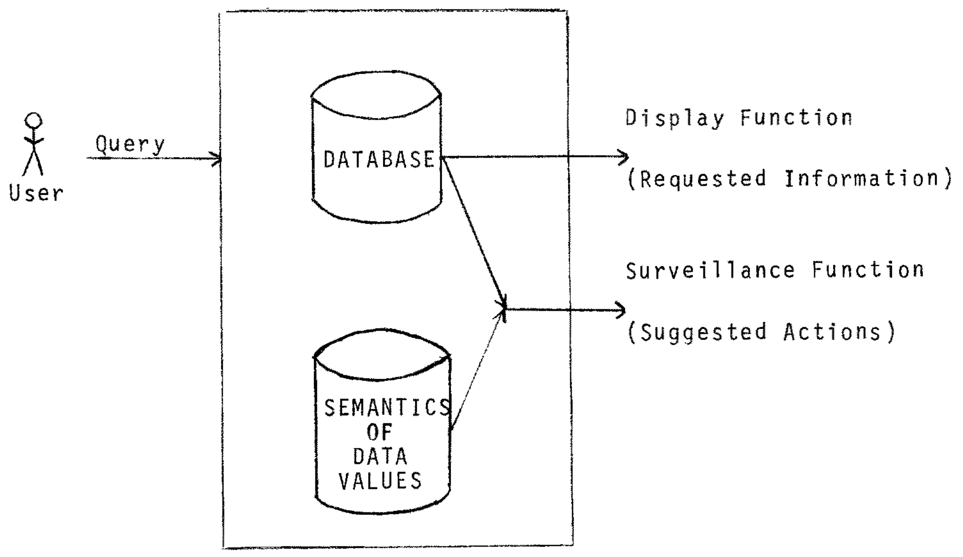

Figure (3)
QUERY SYSTEM

The protocols are defined with the help of individual users of the data values. The task of defining protocols can be assigned to the database administrator.

Another representation could be a casual network or a graph. The nodes could represent an entity value and the arc could represent a relationship with another node. Of course a network or graph could easily become very large if all possible data values for a given item are represented. But if a few selected values for each item and their impact on other entities is considered, the graph can be easily manipulated.

II.2 User's Knowledge (Usage Semantics)

In most database systems, the system treats the user's query as if it is being presented to him for the very first time. Even if a query is repeated a hundred times a day, the current systems have no provision to allow a mechanism that will facilitate the query to be executed faster the next time around. Obviously this is not true in human dialogue where the answer to the same question is given much more quickly.

We also observe that many query systems do not relate one user's query to the next query to capture the context of the user session or more simply the intent of the user. The relationship among queries in a given session is an important concept and has been studied by Grosz (15, 16). It has been demonstrated that interactions in a task-oriented dialogue can be used to establish the focus, or current area of interest, within the task.

We believe that by monitoring the usage patterns of different users, enough empirical data and knowledge can be inferred so as to capture different views of the database. These views can then be used for the logical organization of the database. Thus, whenever the query system identifies the user's interest domain, it can quickly relate it to the logical view and hence answer queries much more efficiently. This subject is also being studied by Chang.

Interestingly a good source of this knowledge is the audit trails (schedules) or copies of user's transactions that are already maintained by current database systems for recovery from crashes.

II.3 Database Organization Semantics

This is the knowledge about the physical characteristics of the file organizations and access methods available on a particular computer system. This knowledge can help us determine the cost of implementing various access structures and also help us in designing appropriate algorithms for database reorganization as the user's access patterns change.

An intelligent query system can also use database organization semantics in determining the cost of creating and maintaining temporary files during a user session. Moreover, it might be possible for the system to inform the user the amount of time it will take to execute a given query. The user can then decide to defer the query in some cases for a later convenient time. Most current systems provide no such information and hence user has no idea about which queries will execute faster.

If the query system is provided with the knowledge of access costs, some optimization of database operations is possible before query execution. For example, Smith and Chang (7) have suggested an optimizer for relational operations such as join, project, and select by considering the cost of implementing them both logically and physically.

In very large databases, the reorganization of databases cannot take place because of lack of free time from ongoing operations. We also believe that a total reorganization policy is difficult to define because of changes in usage patterns and requirements. A solution to this problem would be to create auxiliary structures which can be dynamically created and removed to improve response time. The creation of such structures will have to be cost justified and the database organization semantics, if available will be necessary. The Knowledge Based Management System (KBMS) is being studied at Standord University under the direction of Professor Wiederhold and it should provide interesting investigation in this subject.

A simple way to represent this information will be to employ the same data structures as used to store the database. This will allow easy manipulation of this information and no new languages to access this will be required.

II.4 Real World Semantics

By this, we mean the knowledge that is called laws of science, for example, Newton's Law, Area of a Circle, Pythagorous Theorem, ships have to sail in water, the lower bound of a particular search algorithm is $\log_2(n + 1)$, etc. This semantics can be used for validating updates and retrievals and can reject erroneous requests.

The real-world semantics is static in nature but is context-sensitive to a particular application. The problem, of course, is to determine what knowledge to include.

III. A SCENARIO OF AN INTELLIGENT QUERY SYSTEM

If a physician makes a query for the blood glucose level of a patient in a large medical database, the system checks if the result is normal. If so, the system employs the data semantic to check for all other tests that were performed concurrently or during a small interval of time (for example, the same day). If no other tests were done, the computer simply reports back the glucose count. If other tests were done and have values within normal range, the computer displays the glucose count and provides a feedback to physician by printing "all other tests normal." If these other tests are abnormal, and/or glucose count is abnormal, the system proceeds to detect causes. It checks the protocol (based on data-value semantics) for each abnormal test and derives a list of suggested actions and their justification from the database. This information is kept available in an auxiliary file. If the physician's next query indicates a desire for further exploration on an abnormal result, this auxiliary information is presented to him after security checks. The system also proceeds to check for glucose counts on previous dates to check for any trend. If values have been abnormal or in a critical range, no immediate warning is given.

The system proceeds in certain logical paths depending on the application. The logical paths are followed by using subroutines that define each of the above data value semantic information.

IV. CONSIDERATION OF SECURITY, COST AND INTEGRITY IN THE ABOVE SCENARIO

In our scenario the security problem gets complicated and becomes a very important issue. The additional information obtained by the system cannot be made available to all users. For example, a user may be allowed access to current value but not the previous one. In some cases, the user does not care for conditions in the database and the suggested actions. Thus, the system must incorporate a mechanism to selectively present the different levels of supplementary information to the user. In our design, the security checks can be established via bit maps for all users at each entity node. The bits are used to determine if the user has access to the current and any further nodes in the database. The user is identified to the system at the time of logging to the system by the user's job number, pass-word, and application category. This information is mapped to a unique bit position in the security attributes of each node. Thus the system predetermines the window (subschema) of the whole data semantics that an individual user's queries can employ. For example, the system may be organized to hide certain patient information from a nurse but may provide totally free access to the patient's physician.

The second problem with our approach arises because additional costs are involved when system starts providing supplementary information. We consider that response time (turn around time) for a query is the representation of the cost function. The response time is a function of the number of additional accesses and processing that are required when the system is employing the data value semantics to determine related information. We will set up a routine that interrupts the system when the additional response time exceeds 50% (or any other fraction) of response time required for answering the query without any data value knowledge. At this time the system will be ready to present the user with the additional information. If the user encourages the system to explore further, the system will proceed or else it will execute the next query.

In spite of the above problems of cost and security, one automatic advantage that is achieved by the intelligent query system is data integrity. If for some reason, certain data values are erroneous, the computer feedback draws immediate attention of the user. This is particularly useful in instances where users who are making queries are also entering data. In the medical information system, this situation occurs in the ward, clinical laboratory, and pharmacy. For example, if the front desk clerk enters by mistake the daily drug dosage of 500 mg (abnormal) instead of 200 mg, the physician knows it when he inquires the database, at anytime for any reason, for information of this patient. An immediate feedback from the system for all abnormal dosages can warn the pharamcist when he fills the prescription.

V. FUTURE RESEARCH

Further research is continuing on the problem of how to dynamically collect semantic information from a user session and to create optimal access paths, indexes and temporary files to make present and similar future accesses most efficient. Furthermore, the analysis of the effects of database size on the efficiency of these techniques and thorough cost-benefit study is under progress.

Related research on a much larger scale is being pursued under the project "Knowledge Base Management System" under the guidance of Professor Gio Wiederhold at Stanford University.

REFERENCES

1. Bhargava, B.K., "Crossfertilization of DBMS Research with Other Disciplines of Computer Science," ACM National Conference, December 3-5, 1978, Washington, D.C.

2. Bhargava, B. K., "Data Abstraction, Data Search, and Query Languages," presented at Computer Science Conference, Detroit, February 1978, Technical Report.

3. Bhargava, B. K., "An Intelligent Query System," Annual Pattern Recognition and Image Processing Conference, Chicago, 1978.

4. Bhargava, B. K., and C. J. McDonald, "Tree Systems for Medical Information Processing," Eleventh Annual Rocky Mountain Bioengineering Symposium, USAF Academy, April 15-17, 1974, Colorado.

5. Bhargava, B. K., et. al., "Development and Implementation of a Computerized Clinical Laboratory System," LABORATORY MEDICINE, November 1976.

6. Bhargava, B. K. et. al., "A Computer Record and Clinical Monitoring System for Ambulatory Care," American Journal of Public Health, March, 1977.

7. Smith, J. M. and P. Chang, "Optimizing the Performance of a Relational Algebra Database Interface," CACM. Vol. 18, Oct., 1975.

8. Carlson, C. R. and Kaplan, R. S., "A Generalized Access Path Model and Its Application to a Relational Data Base System," in Proc. ACM SIGMOD, 1976, pp. 143-154.

9. Chang, C. L., "DEDUCE: A Deductive Query Language for Relational Data Bases," in Pattern Recognition and Artificial Intelligence, Academic Press, 1976, pp. 134-188.

10. Chang, S. K., and J. S. Ke, "Database Skeleton and Its Application to Fuzzy Query Analysis," IEEE Trans. on Software Engineering, Vol. SE-4, No. 1, January 1978.

11. Codd. E. F., "A Relational Model of Data for Large Shared Data Banks," CACM, Vol. 13, No. 6, June, 1978, pp. 377-387.

12. Codd, E. F., "Rendezvous Version I: An Experimental English-Language Query Formulation System for Casual Users of Relational Databases" in Databases: Improving Usability and Responsiveness," edited by Ben Shneiderman, Academic Press 1978.

13. Codd, E. F., "Seven Steps to Rendezvous with the Casual User," in Data Base Management, J. W. Klimbie and K. I. Koffemen, eds., North Holland, 1974. pp. 179-200.

14. Ghosh, S. P., "Consecutive Storage of Relevant Records with Redundancy," CACM, Vol. 16, No. 8, August, 1975, pp. 464-471.

15. Grosz, B. J., "The Representation and Use of Focus in Dialog Understanding," Ph.D. Thesis, the University of California at Berkeley, June 1977.

16. Grosz, "The Representation and Use of Focus in a System for Understanding Dialogs," Proc. 5th IJCAI, Cambridge, MA, 1977, Vol. 1, pp. 67-76.

17. Hammer, M. and A. Chan, "Index Selection in a Self-Adaptive DBMS' Proc. of ACM-SIGMOD, Washington, 1976.

18. Hendrix, G. G. et. al., "Developing a Natural Language Interface to Complex Data," ACM Transaction on Database Systems, Vol. 3, No. 2, 1978.

19. Hevner, A. R., and Yao, S. B., "Optimization of Data Access in Distributed Systems, TR-281, Computer Science Department, Purdue University, July, 1978.

20. McDonald, Clement, Bharat Bhargava and David Jeris, "A Clinical Information System for Ambulatory Care," National Computer Conference, May 19-23, 1975, Anaheim, California.

21. Mylopoulos and N. Roussopoulos, "Using Semantic Networks for Data Base Management," Proc. International Conference on Very Large Data Bases, Framingham, MA, September 1975, pp. 144-172.

22. Petrick, S. R.,"On Natural Language Based Computer Systems," IBM Journal of Research & Development, Vol. 28, No. 4, July, 1976, pp. 114-125.

23. Robinson, A., "Investigating the Process of Natural-Language Communication" A Status Report, Technical Note, 165, SRI, July, 1978.

24. Sagalowicz, "IDA: An Intelligent Data Access Program,"Proc. Third Int'l Conference on Very Large Databases, Tokyo, Japan, October 1977.

25. Walker, D.E., et. al., "An Overview of Speech Understanding Research at SRI," Proc. 5th International Joint Conference on Artificial Intelligence, Cambridge, MA, August 1977.

26. Waltz, D. L., "An English Language Question Answering System for a Large Relational Database," CACM, Vol. 21, No. 7, 1978.

27. Wiederhold, G., "Database Design," McGraw-Hill, 1977, Chapter 7.

28. Wiederhold, G., "Binding in Information Processing," Technical Report, Computer Science Department, Stanford University.

29. Wiederhold, G., "Management of Semantic Information for Databases," 3rd USA-Japan Conference, San Francisco, October 1978.

30. Wiederhold, G. and R. Elmasri, "A Structural Model for Database Systems, Technical Report CSD, Stanford University.