8-25-1995

Design Reusability and Learning in CABSYDD (Case-Based System for Database Design)

W. Amber Lo
Millersville University

Joobin Choobineh
Texas A&M University

Design Reusability and Learning in CABSYDD (Case-Based System for Database Design)

W. Amber Lo
Department of Business Administration
Millersville University

Joobin Choobineh
Department of Business Analysis and Research
Texas A&M University

1. Introduction

A database is a collection of related data that represents some relevant reality. The database design process consists of four steps: Requirements Collection and Analysis, Conceptual Database Design, Logical Database Design, and Physical Database Design. Recently, knowledge-based computer-aided design tools have been introduced. A list of them can be found in Lo (1994). None is empowered with machine learning to improve its permanent knowledge base or inference abilities. This research applies case-based reasoning to Conceptual Database Design. Section 2 reviews related theories. Section 3 discusses design and implementation of the application. Lastly, section 4 is the conclusion.

2. A Review of Related Theories

Section 2.1 describes the Enhanced Entity Relationship Model. Section 2.2 discusses case-based reasoning.

2.1 The Enhanced Entity-Relationship

Model The Enhanced Entity-Relationship (EER) Model of Elmasri and Navathe (1989) is the data model used in this research. The three basic constructs are entity type, attribute, and relationship type. An entity type (ET) is a group of objects that have the same kind of properties, e.g., ET BOOK-CIR is all the books for regular circulation in a library. Clusters of subclass ETs can be defined from a superclass ET. The superclass at the highest level is the "root-level" ET, and, if taken with all its subclasses at different levels, it is called "a chunk of entity types" (Lo 1994). Two descriptions are used to provide information on an ET. The entity type role describes the role that an ET plays in organizations of the same industry (Lo 1994). For example, ET USER in a library plays the role "MEMBERS." The entity type domain describes its scope (Navathe et al. 1986). For example, HOLDING, with domain "ALL HOLDINGS" covers all library materials. An attribute (ATTR) is a property of an entity type or a relationship type. The three basic attribute domains are string, numeric, and logical. Each attribute has a cardinality ratio which denotes the minimum and maximum numbers of possible values for that attribute (Batini and Lenzerini 1984). The key attribute value identifies each particular instance of an ET. A relationship type (RT) is an association among ETs. For example, "a user checks out a book for regular circulation" is the activity statement of the RT CHECKS-OUT-C between USER and BOOK-CIR. All RTs are restricted to be binary here as in Storey and Goldstein (1988).

2.2 Case-Based Reasoning and Case- Based Systems

Case-based reasoning is the process of solving a problem by retrieving the solution of a previously solved problem and adapting it to that of the problem under study. Past experiences are used to avoid having to start from scratch every time. This new solution along with the problem statement (together called a "case") can be saved and properly indexed in a case base for future use. Multiple solutions to a problem form a category of cases. The process of case-based reasoning can be divided into four major steps as shown in Figure 1 (Kolodner and Riesbeck 1990, Riesbeck and Schank 1989): Problem Description, Base Case Searching and Retrieval, Base Case Adaptation and New Solution Evaluation, and Learning. The Problem Description step obtains the index values of the problem. Then, the Base Case Searching and Retrieval step tries to find and retrieve the closest case to serve as the base case. The search involves going through and measuring the usefulness of all relevant cases and the one with the highest score is chosen. The Base Case Adaptation and New Solution Evaluation step adapts the chosen base-case solution and evaluates the result until a satisfactory new solution is derived. Evaluation methods and standards are specific to individual problem domains. The last step, Learning, retains useful experiences to increase future problem solving power. Case-based learning often results in changes in the knowledge base. Various case-based expert systems have been implemented (Kolodner 1993). No attempt has been reported in using this approach for database design. The next section discusses such application.

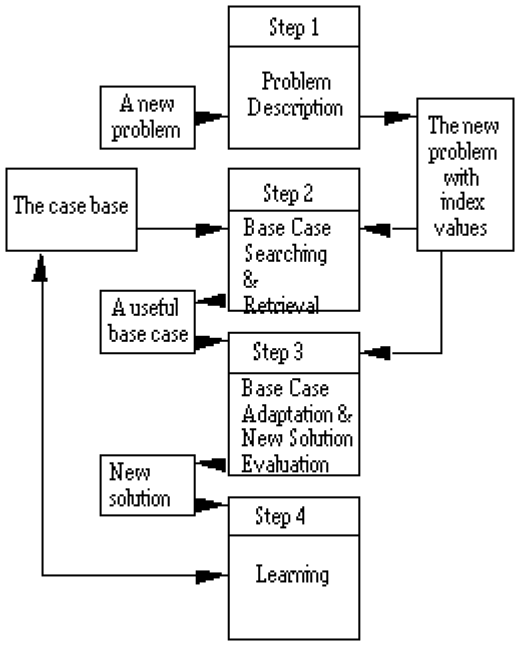
Step 1
Problem Description
A new problem
The new problem with index values
The case base
Step 2
Base Case Searching & Retrieval
A useful base case
Step 3
Base Case Adaptation & New Solution Evaluation
New solution
Step 4
Learning

Figure 1. The Process of Case-Based Reasoning

3. Design and Implementation of the Case-Based Database Design System

The concept of maintaining a schema library was first proposed by Di Battista et al. (1989). A case-based database design tool can further take over the human activity of selecting a base case and help transform it into the desired schema. Section 3.1 describes the case base, section 3.2 discusses case-based conceptual design, and section 3.3 describes implementation.

3.1 The Case Base

As functional departments in enterprises of the similar industries share many common objects and associations among objects, conceptual schemata defined at this level should be highly reusable. A case is a complete and stand-alone conceptual schema for one particular functional department. They are classified by industry according to the Standard Industrial Classification Manual (1987). As shown in Figure 2, the case base has four levels of primary indexes: division, major group, industry group, and functional department with cases at the lowest level. Each department is created as a new conceptual schema is stored as the first case for that department. All cases of a department of a particular industry group form a category of cases. If there are more than one case in a category, secondary indexes (acquired through learning) will become necessary below the departmental level. Each secondary index is a root-level ET name of some case in that category. The term "index" here refers to a single element and not a list. An index value of "y" (meaning "yes") for a case states the fact that this case has that ET. A value of "n" (meaning "no") states that the ET is not needed by this case. Each case has to be defined by all secondary indexes of that category. For example, in Figure 2, there are three cases in the "library" category with their own values for the three secondary indexes: 1) locker, 2) study-room, and 3) special-program.

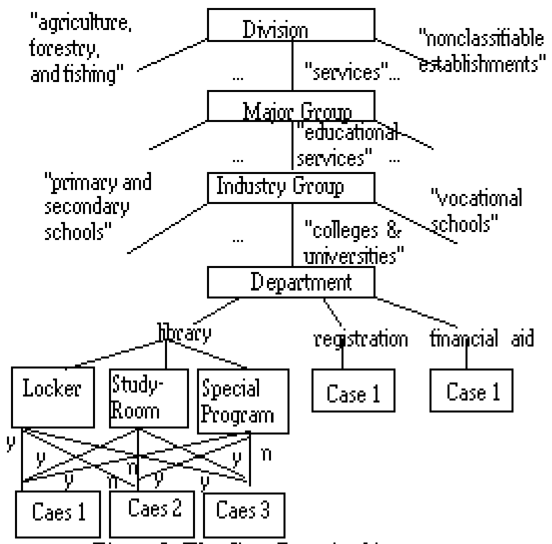

Figure 2. The Case Base Architecture

3.2 The Process of Case-Based Conceptual Design

Following the process of case-based reasoning shown in Figure 1, case-based conceptual database design consists of four steps. This paper concentrates on the learning of new cases and secondary indexes. Further details of the first three steps are in Lo (1994).

The schema to be defined is called the new case. The first step, New Case Description, describes this new case in terms of the relevant indexes of the case base. A value should be supplied for each index. The second step, Base Case Searching and Retrieval, is carried out in a best-first search manner. The corresponding category should be reached by following a branch according to the division, major group, industry group, and department to which this new case belongs. There are three possible outcomes of this search. First, if there is only one case in the category, this only case will be retrieved for deriving the conceptual schema of the new case. Second, if there are more than one case in the category, the case with the most matching secondary index values will serve as the base case. If there is a tie, then one case will be chosen randomly. Third, if the category of this new case is not in the case base, then a case for the same functional department has to be found among other industry groups in the same major group. Each category may have its own set of secondary indexes to be searched. The one case with the highest final score will become the base case. If no industry group in the same major group has this departmental case, then this new schema will be defined from first principles.

The third step, **Base Case Adaptation and New Schema Evaluation** , follows an interactive parameter adjustment strategy (Kolodner and Riesbeck 1990). In order to transfer constructs in an orderly fashion and to maintain the consistency of ET roles and domains, adaptation is performed in an old-to-new system-initiated manner. The procedure first selects an old ET according to a set of priorities and then transforms it to the equivalent new construct. It continues until all the ETs of the new case have been derived. The same procedure is then applied to RTs. Besides maintaining consistency as the new schema evolves, overall evaluation also includes the detection and elimination of dangling entity type chunks and unnecessary subclasses.

Learning occurs when a new case and any new secondary indexes are added into the case base. With more indexes in the case base, the system has more knowledge about classifying schemata. Secondary indexes are specific to a category because each one records a difference in a root-level ET among all the cases. The decision of adding a new case and its justification depends on the observed structural differences between the base and the new cases. A new index is automatically created at the first discovery of the difference and the corresponding index values of all cases are also automatically determined by the system.

The very first case (either defined from scratch or from a neighboring case) to be added to a new category does not need any secondary index as there is no other case to which it is compared. Any secondary index of a neighboring base case is not transferred as it does not denote a structural difference of this category.

If both the base and the new cases belong to the same category, then the new case is added only if it causes the creation of one or more new secondary indexes in either of the two situations: (1) the base case is the only existing case with no secondary index and (2) there are more than one existing cases and some secondary indexes and both the base and the new cases have exactly the same set of secondary index values. There are two ways in which new secondary indexes are created. First, any root-level ET not transferred to the new view (not yet used as a secondary index) is used as a new secondary index with the exact ET name as the index name. All existing cases have an index value of "y" and the new case has an index value of "n." Second, similarly, any new root-level ET defined from scratch for the new case is used as a new secondary index. All existing cases have a value of "n" and the new case has a value of "y."

If a new case has partially matching secondary index values in its own category, then it is retained because there is no other case in that category with the same set of secondary index values. The creation of any new index is the same as described above.

3.3 Implementation

The prototype "CABSYDD" is implemented in the expert system shell Clips (Version 6.0). It aids a designer in performing external view definition either through case-based reasoning or from scratch (through a subsystem called SYDD) if no relevant case is available. SYDD implements a step-by-step conceptual design methodology from first principles (Lo 1994).

4. Conclusion

This research was the first in considering past experiences for database design. This research demonstrated the feasibility of applying case-based reasoning to conceptual database design through the development of the prototype, CABSYDD. It increases its own knowledge with experiences. Evaluation, which includes system validation and verification, has also been performed (Lo 1994). References and the full version of this abstract are also available upon request.