Boyce-Codd Normal Form Decomposition

K. K. Nambiar
School of Computer and Systems Sciences
Jawaharlal Nehru University, New Delhi 110067, India

B. Gopinath T. Nagaraj S. Manjunath
Department of Electrical and Computer Engineering
Rutgers University, Piscataway, NJ 08855

(Received September 1996)

Abstract—An algorithm is given for the lossless decomposition of a relation into Boyce-Codd Normal Form.

Keywords—Boyce-Codd Normal Form; Decomposition; Algorithm.

1. INTRODUCTION

In relational database theory [1-3], a relation is said to be in Boyce-Codd Normal Form (BCNF), if all the determinants in the relation are keys. A set of relations is called a lossless decomposition of a given relation if the join of the relations gives back the original relation. In this paper, we give a method for obtaining a lossless decomposition in which each relation is in BCNF. It is known [1] that boolean functions written in terms of its prime implicants can be used to represent functional dependencies in a relation. We make use of them to obtain the decomposition.

2. DEFINITIONS AND NOTATIONS

Brief definitions of special terms used here are given below, some conventional definitions are also briefly stated just to be complete.

Relation: A subset of a cartesian product. A relation can be visualized as a table.

Attributes: Columns of the table.

Database: A set of relations.

Dependency Function: A disjunctive boolean expression in which every term has exactly one literal complemented.

Horn Function: A disjunctive boolean expression in which every term has one or zero number of literals complemented.

Unate Function: A disjunctive boolean expression in which every term has no literal complemented.

$ABC\cdots H$: A product $ABC\cdots H$ can mean three things: a boolean term $ABC\cdots H$, a relation consisting of attributes $A,B,C,\cdots,H$ or a set $\{A, B, C, \cdots, H\}$. The meaning is to be taken from the context.

Projection: A table corresponding to a subset of the columns.

Semiboolean Lattice: The lattice corresponding to a unate function.

It has been shown that the dependencies in a relation can be represented by either a dependency function or a horn function. Also, it is known [1] that a relation is in BCNF, if its horn function is a unate function. We make use of this fact in developing an algorithm for the decomposition.

3. BOYCE-CODD NORMAL FORM

Given below are two examples to illustrate the procedure for lossless decomposition.

Example 1.

Consider a relation in which there are two fuctional dependencies

$$AB \to C,$$
$$C \to A.$$

The dependencies in the relation can be represented by the dependency function

$$AB\bar{C} + C\bar{A}.$$

If we add the term ABC to the dependency function, we get the horn function

$$AB + BC + C\bar{A}.$$

We look for a *smallest* term *with a complemented literal* in the expression. In our case, it is $C\bar{A}$. The projection AC obviously has to be in BCNF. We delete $C\bar{A}$ and the terms containing the dependent attribute A from the rest of the expression. We get BC as the only term left. The required decomposition is

$$\{AC, BC\}.$$

It is easy to see that the join of these relations will give the original relation.

Example 2.

Consider a relation in which there are four fuctional dependencies

$$C \to A,$$
$$D \to B,$$
$$AD \to C,$$
$$BC \to D.$$

The dependencies in the relation can be represented by the dependency function

$$C\bar{A} + D\bar{B} + AD\bar{C} + BC\bar{D}.$$

If we add the term $ABCD$ to the dependency function, we get the horn function

$$AD + BC + CD + C\bar{A} + D\bar{B}.$$

We look for a smallest term with a complemented literal in the expression, which in this case we can take as $C\bar{A}$. We note that the projection AC has to be in BCNF. We delete $C\bar{A}$ and the terms containing the dependent attribute A from the rest of the expression. In what is left, we look for the smallest term with a complemented literal and obtain it as $D\bar{B}$. Noting that the projection BD is in BCNF, we delete the term $D\bar{B}$ and the terms containing the dependent attribute B. We notice that CD is the only term left. The required decomposition is

$$\{AC, BD, CD\}.$$

These relations can be joined to get the original relation.

4. ALGORITHM

The algorithm for the decomposition of a relation into BCNF can be stated as follows.

Step 1. From the given dependencies, write the dependency function.
Step 2. Add to the dependency function the term $ABC\cdots H$ and get the horn function.
Step 3. Search for a smallest term with a complemented literal.
Step 4. If Step 3 is successful, get the projection corresponding to the term.
Step 5. Delete the smallest term in Step 3 from the horn function along with all the terms containing the dependent attribute.
Step 6. Repeat Step 3 to 5 until the horn function reduces to a unate function.
Step 7. Form the projection corresponding to the literals in the unate function appearing in Step 6 and the literals that might have disappeared while going from Step 1 to Step 2.

The set of projections collected while carrying out the steps gives a lossless decomposition in which all the relations are in BCNF.

5. CONCLUSION

It is known that corresponding to every dependency function there is a lattice and hence a lattice can also represent dependencies in a relation. In terms of lattices, the algorithm for lossless decomposition can be visualized as seperating out semiboolean lattices from a given lattice.

REFERENCES

1. K. K. Nambiar, *Some Analytic Tools for the Design of Relational Database Systems,* Proceedings of the Sixth International Conference on Very Large Databases, Montreal, (1980).
2. W. W. Armstrong, *Dependency Structures of Database Relationships,* Information Processing 74, North Holland, Amsterdam, (1974).
3. C. J. Date, *An Introduction to Database Systems,* Narosa Publishing House, New Delhi, (1995).