A Note on Minimal Covers

by

John Atkins
Department of Statistics and Computer Science
West Virginia University
Morgantown, WV 26506

Introduction

Minimal covers play an important role in the design of relational database schemes. They are used in determining a lossless decomposition of a relation scheme R into relation schemes in third normal form while preserving dependencies. It is well known that if F is a minimal cover for a relation scheme R, then the decomposition of R consisting of relations S(X,Y) where X and Y are sets of attributes of R and $X \to Y$ is in F is a decomposition of R into relations in third normal form which preserves the dependencies. The decomposition is lossless if we include a relation scheme which consists of a key for R.

If R is a relation scheme with a set of functional dependencies T that is not in third normal form, we can provide a dependency preserving, lossless decomposition of R into relations in third normal form if we can determine a minimal cover F for the set of dependencies T. This paper corrects a wide-spread misconception regarding the algorithm for extracting a minimal cover from a given set of functional dependencies.

Definitions

Let R be a relation scheme. If F and T are two sets of functional dependencies for R, we say that F covers T (and vice versa) if the closures [3, pg. 216] of F and T are equal. A minimal cover is defined as follows [3]:

Definition: A set of dependencies F for a relation scheme R is minimal if:

1. Every right side of a dependency in F is a single attribute.
2. For no $X \rightarrow A$ in F is the set $F - \{X \rightarrow A\}$ a cover for F.
3. For no $X \rightarrow A$ in F and proper subset Z of X is

$(F - \{X \rightarrow A\}) \cup \{Z \rightarrow A\}$ a cover for F.

The algorithm to convert a set of dependencies T to a minimal cover F is erroneously described [2, 3, 4, 5] by:

1. If $X \rightarrow Y$ where $Y = A_1 A_2 \ldots A_n$ is in T, replace $X \rightarrow Y$ by n dependencies having singleton right sides, i.e. $X \rightarrow A_1$, $X \rightarrow A_2, \ldots$.

2. For each dependency $X \rightarrow A$ in F, if $X \rightarrow A$ can be derived using the remaining dependencies in F, remove $X \rightarrow A$ (immediately) from F.

3. "Having satisfied condition 2", consider each dependency $X \rightarrow A$ remaining in F and each attribute, in some order, in X. If we can eliminate an attribute from X and maintain an equivalent set of dependencies, we do so until no further reduction of any dependency is possible.

In [5, page 95], Yang writes that "...we need only to decide whether to delete $X \rightarrow A$ if it is redundent or to delete any extraneous attributes in X otherwise". Again, the order of operations appears, erroneously, to be immaterial.

Counter example

The above algorithm for converting a set of dependencies to a minimal cover does not always result in a minimal cover for a set of dependencies. Consider a relation scheme R(A, B, C) with a set of dependencies $F = \{A\ B \rightarrow C, C \rightarrow B, A \rightarrow B\}$.

F clearly satisfies the condition which requires singleton right sides. We can now satisfy condition 2 by removing extraneous dependencies from F. However, no dependency appears to be extraneous. This is seen by considering closures of left sides:

(1) AB closure relative to $\{C \rightarrow B, A \rightarrow B\}$ is AB.
(2) C closure relative to $\{AB \rightarrow C, A \rightarrow B\}$ is C
(3) A closure relative to $\{AB \rightarrow C, C \rightarrow B\}$ is A.

To satisfy condition 3, we need only consider the dependency AB → C. We can see that the attribute B is redundant by considering A closure relative to F. Since A closure is ABC relative to F, B is redundant. Thus, we replace AB → C by A → C to get a minimal cover {A → C, C → B, A → B}. Unfortunately, this is not a minimal cover since the dependency A → B is now extraneous since A → B can be derived from A → C and C → B.

Corrected Algorithm

The algorithm to correctly replace an arbitrary cover by a minimal cover is essentially the previous algorithm but with steps 2 and 3 interchanged. The first step in the algorithm is to again ensure that every functional dependency in F has a single attribute on the right side.

The second step is to consider each dependency X → A in F in some order. If Z is a subset of X such that F is contained in the closure of (F - {X → A}) ∪ {Z → A}, then immediately replace X → A in F by Z → A. This step continues until no left side of any dependency in F can be reduced.

The third step is to again consider the dependencies in F in some order. If X → A is in F and X → A is in (F - {X → A}) closure, then immediately remove X → A from F. This step continues until every dependency in F has been tested.

The algorithm produces a minimal cover since if it did not, a dependency $X \rightarrow A$ would remain in F with either extraneous attributes in X or is derivable from the remaining dependencies in F. Clearly $X \rightarrow A$ cannot be derivable from the remaining dependencies in F since this would have been discovered in step 3. The only alternative then is for $X \rightarrow A$ to have redundant attributes in X that appeared as a consequence of step 3. Since step 3 only eliminates dependencies and does not modify any dependencies, no left side of a dependency can be reduced after step 3 that could not be reduced in step 2. Thus, the resulting cover satisfies the condition for a minimal cover.

Conclusion

It is interesting to note that at least one reference has observed that the order of the algorithm is pertinent. Maier [1, page 75] introduces the notion of reduced covers which leads to his definition of a canonical (minimal) cover. In Maier's description of the technique for finding a reduced cover, he notes that reducing right sides first before reducing left sides "will not work." In terms of our algorithm, Maier is saying that we cannot remove extraneous dependencies from F before removing redundant attributes from left sides of dependencies and expect to have a set of dependencies with no extraneous dependencies.

References

1. David Maier, The Theory of Relational Databases, Computer Science Press, Rockville, Maryland, 1983.

2. Betty Salzberg, Third Normal Form Mode Easy, Sigmod Record, 15(1986), 2-18.

3. Quentin F. Stout and Patricia A. Woodworth, Relational Databases, The American Mathematical Monthly, 90 (1983) 101-118.

4. Jeffrey D. Ullman, Principles of Database Systems, Computer Science Press, Rockville, Maryland, 1982.

5. Chao-chih Yang, Relational Databases, Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

RESPONSE from Betty Salzberg:

"I would like to thank John Atkins for pointing out this error."