AN ALGORITHM FOR SYNTHESIZING A LOGIC DATABASE SCHEMA

Vitalie Cotelea

A.S.E.M. Chişinău

Abstract. An improved algorithm for synthesizing a database schema is considered. The schema that results from this procedure is proved to be in Codd's third normal form and to possess lossless-join property. It produces result even though some of the universal schema's attributes are not implicate in functional dependencies.

Keywords. Database schema, third normal form, functional dependency

An improved algorithm for synthesizing a logic database schema in third normal form proposed by Philip A. Bernstein [1] is presented. The principal problem generated by Bernstein algorithm is that the result schema doesn't always possess lossless-join property. Besides this the algorithm can be applied only in the case when all the attributes of the universal schema are implicated in functional dependencies.

The main idea of improvement consists in the fact that to the set of functional dependencies one more dependency is added. Its left side is the universal schema and the right side is an artificial attribute which doesn't exists in the universal schema. This attribute disappears in the course of further algorithm development but the mentioned problems are solved.

The described algorithm is a synthesizing procedure because it starts from the set of functional dependencies F defined on the universal schema R and produces database schema $BD=\{R_1,...,R_m\}$, where $R=R_1\cup...\cup R_m$, grouping the dependencies F so as to satisfy the following four conditions:

(1) $F\equiv\pi_{R1}(F)\cup...\cup\pi_{Rm}(F)$, where $\pi_{Ri}(F)=\{X\rightarrow Y \mid X\rightarrow R_i\in F^+\}$;

(2) BD is in third normal form;

(3) $\forall BD_j, |BD_j|\geq|BD|$;

(4) $\forall r(R), r=\pi_{R1}(r)\bowtie...\bowtie\pi_{Rm}(r)$.

Condition (1) assures that the projections of relation r on schema BD conserve F. Moreover, condition (1) informs that R_i, $1\leq i\leq m$, satisfies dependencies whose determinants are superkeys of scheme R_i.

Condition (2) is the goal of normalization. Its necessity was studies in detail in [2].

Condition (3) guaranties that any other schema built on the set F doesn't contain fewer relational schemes.

Condition (4) affirms that the database schema has lossless-join property.

To describe the algorithm the notation $ATR(F)$ which presents the set of attributes involved in dependencies F will be used, i.e.

$$ATR(F)=\{V \mid V=X\cup Y \ \&\ X\rightarrow Y\in F\}.$$

For example, if $F=\{C\rightarrow E,E\rightarrow A,CE\rightarrow D,A\rightarrow B\}$, then . $ATR(F)=\{A,B,C,D,E\}$.

To the set F of functional dependencies the dependence $R\rightarrow\Omega$ is added. This fact imposes the result schema to possess lossless-join property and all universal schema attributes to be involved.

FN3_SYNT Algorithm

input:

R: universal schema

F: the set of functional dependencies on R

output:

$BD=\{R_1,...,R_m\}$: a database schema in third normal form witch a minimal number of relational schemes and which possesses lossless-join property and conserves the functional dependencies F

begin

$BD:=\varnothing$;

$J:=\varnothing$;

1 $F:=F\cup\{R\rightarrow\Omega\}$;

2 $G:=CANON(F)$;

3 $E:=CLASE\_ECHIV(G)$;

4 *if* $|E_G(R)|>1$ *then* $E_G(R):=E_G(R)\setminus\{X\rightarrow\Omega\}$;

5 *for each* $E_G(X)\in E$ *do*

for each $X\rightarrow A\ \&Y\rightarrow B\in E_G(X)$ *do* {aici $X\rightarrow A\neq Y\rightarrow B$}

$E_j(X):=E_j(X)\cup\{X\rightarrow Y, Y\rightarrow X\}$;

6 *for each* $E_G(X)\in E$ *do*

begin

for each $X\rightarrow A\in E_G(X)$ *do*

if $MEMBRU(E\setminus\{E_G(X)\}\cup\{E_G(X)\setminus\{X\rightarrow A\}\}\cup J, X\rightarrow A)$

then $E_G(X):=E_G(X)\setminus\{X\rightarrow A\}$;

$E_G(X) := E_G(X) \cup E_j(X)$;

end

7 *for each* $E_G(X) \in E$ *do*

if $\Omega \in ATR(E_G(X))$ *then* $BD := BD \cup \{ATR(E_G(X)) \setminus \{\Omega\}\}$

else $BD := BD \cup \{ATR(E_G(X))\}$;

return (BD);

end.{ FN3_SYNT}

A description of synthesizing database schema algorithm is given.

Step 1 adds to the initial set of functional dependencies F the functional dependency $R \rightarrow \Omega$, where the determinant is the universal attribute set and the determinat is an attribute doesn't belong to the field of interest. The aim of including of this artificial dependency is to elaborate a database schema which possesses the lossless-join property.

Step 2 builds a canonical cover of the set F (a set is canonical if it's nonredundant, left reduced and the right sides of the dependencies are single attributes). At this stage the foundation of the database schema with the minimal relation schemes and sets of keys for these schemes is built.

Step 3 divides the canonical set of functional dependencies G into classes of equivalence (an class of equivalence is a set of functional dependencies with equivalent left sides).

Step 4 verifies if the functional dependency with artificial attribute Ω is placed into a class with other functional dependencies. In the affirmative case the dependency with Ω attribute is eliminated from the respective class.

Step 5 builds a set J of functional dependencies in the following way. At the beginning $J = \varnothing$ is set, then J is modified for each two different functional dependencies with X and Y determinants (where $X \leftrightarrow Y$) from $E_G(X)$ as follows $E_j(X) := E_j(X) \cup \{ X \rightarrow Y, Y \rightarrow X\}$.

Step 6 eliminates transitive dependencies of nonprime attributes. A set $E^l \subseteq E$, which satisfies $(E^l \cup J) \equiv (E \cup J)$ is found, so as none proper subset of set E^l doesn't satisfy the given condition. Then the dependencies from J are included into respective classes of equivalence of set E^l.

Step 7 forms the relational schemes $R_1,...,R_m$. Each scheme R_i includes the attributes involved by the functional dependencies from the i class of equivalence. At the end the database schema $BD=\{R_1,...,R_m\}$ is obtained. At the same time, if the artificial attribute Ω is a part of a relational scheme R_i then it is dropped from this scheme.

It is necessary to mention that algorithm for synthesizing database schema in Codd's third normal form has a polynomial temporal complexity. Moreover it builds a schema with a minimal number of relational schemes.

Theorem. If *BD* is the database schema synthesized from set *F*, then *BD* contains $|\mathbf{E_G}|$ relational schemes.

Proving. The applied dependencies for R_i ($1 \le i \le m$) relational scheme construction have equivalent determinants, i.e. they forms a class of equivalence. Therefore *BD* contains so many relational schemes as in how many classes of equivalence the set *G* is divided. But as the schema is canonical, i.e. and nonredundante then there doesn't exist a set equivalent to it with fewer classes of equivalence. The proving is evident

This theorem confirms that the synthesizing algorithm satisfies the condition (3).

Condition (1) is respected because $E \equiv E \cup J$ and *E* is *G* divided in classes of equivalence and $G \equiv F$.

Condition (2) is satisfied by step 6 of the algorithm that excludes transitive dependencies.

The inserting of the artificial dependency $R \rightarrow \Omega$ in set *F* makes synthesizing algorithm to generate the schema, which has the losless-join property, which satisfies condition (4). Besides the $R \rightarrow \Omega$ dependency links the attributes which eventually are not involved into initial set of functional dependencies *F*.

Here the lossless-join property is assured by the universal key introduced into database schema together with $R \rightarrow \Omega$. dependency (naturally if this key doesn't exist).

Definition. Let $BD=\{R_1,...,R_m\}$ be a database schema, where $R=R_1 \cup ... \cup R_m$, and let *F* be a set of functional dependencies on *R*. The set $K \subseteq R$ is called database universal key if $F \models K \rightarrow R$ and doesn't exist K^1, where $K^1 \subset K$, that satisfies $F \models K^1 \rightarrow R$.

Theorem. Let *R* be a relational scheme and *F* - a set of functional dependencies on *R*. Let *BD* be a decomposition of the scheme *R*. Then database schema $BD:=BD \cup \{K\}$ is a lossless decomposition of scheme R.

*Proving***.** The proving is evident because the tuple represented by the universal key in decomposition tableau is a goal tuple (which consist of distinct variables only).

Therefore the adding of $R \rightarrow \Omega$. dependency to *F* really introduces the universal key. Without this dependency the synthesizing algorithm doesn't always generate a database schema which possesses losslees-join property

It is necessary to mention that this idea can be extended to each user view so as they possess the lossless-join property. With this aim, for each V_i user view a dependency $V_i \rightarrow \Omega_i$ is added to the initial set *F*.

REFERENCES:

1. Philip A. Bernstein. Synthesizing Third Normal Form Relations from Functional Dependencies. *ACM Transaction on Database Systems*, Vol.1, No.4, 1976, p.277-298.
2. Codd, E.F. Further normalization of the database relational model. In *Database Systems, Courant Inst. Comptr. Sci.Symp. 6*, R.Rustin, Ed., Prentice-Hall, Englewood Cliffs, 1972, p.33-64.