RAIRO
INFORMATIQUE THÉORIQUE

C. BATINI
A. D'ATRI

Relational data base design using refinement rules

RAIRO – Informatique théorique, tome 17, n° 2 (1983), p. 97-119.
<http://www.numdam.org/item?id=ITA_1983__17_2_97_0>

R.A.I.R.O. Informatique théorique/Theoretical Informatics
(vol. 17, n° 2, 1983, p. 97 à 119)

RELATIONAL DATA BASE DESIGN USING REFINEMENT RULES (*)

by C. BATINI and A. D'ATRI (1)

Communicated by G. AUSIELLO

Abstract. – *The design of a data base schema can be organized through a set of refinement steps, that gradually introduce progressive details in the description of reality. In this approach, suitable restrictions to refinement rules allow to guarantee both conceptual and logical properties.*

Following this approach the formalism of hypergraphs and hypergraph grammars can be applied to investigate various aspects of data base schema design, in the framework of the relational model of data.

In this paper we investigate, as refinement restrictions, boundings in renamings of objects and prove several results concerning meaningful properties in the relational theory of data.

Résumé. – *La conception du schéma d'une base de données peut se structurer en étapes introduisant progressivement les détails de la description du monde réel. Dans ce travail, nous proposons des contraintes sur les étapes pour garantir certaines propriétés au niveau conceptuel et logique. Nous utilisons des hypergraphes et des grammaires d'hypergraphes pour étudier plusieurs aspects de la conception du schéma d'une base, dans le cadre du modèle relationnel. Nous considérons des contraintes liées au changement de noms d'objects, et nous présentons des résultats liés à des propriétés importantes de la théorie relationnelle.*

CONTENTS

1. INTRODUCTION

This paper is devoted to the investigation of formal methodologies for the design of the data base (DB) schema in the framework of the n-ary relational model of data.

(*) Received in 1981, revised in July 1982.

(1) Istituto di Automatica, Università di Roma and I.A.S.I.-C.N.R., via Eudossiana, 18, Roma (Italy).

R.A.I.R.O. Informatique théorique/Theoretical Informatics, 0399-0540/1983/97/$ 5.00

A data base may be seen as a collection of informations about a fragment of the real world. In the n-ary relational model the reality of interest is represented in the following way.

Let $T=\{A_1, A_2, \ldots, A_n\}$ be a finite set of *attributes;* we will denote by $\ldots X, Y, Z$ subsets of T. Let DOM be a function that associates to every attribute A_i a set of *values*; a *relation instance* R (T) over T is a subset of the Cartesian product DOM $(A_1) \times$ DOM $(A_2) \times \ldots \times$ DOM (A_n); an element of the former product is called *tuple*; the value of a tuple t corresponding to attributes $X \subseteq T$ is denoted by $t.X$.

A relation instance can be visualized by means of a table in which columns are labelled with attributes and rows depict tuples (see *fig.* 1).

Employée	Age	Salary
0505	28	30,000
0610	31	30,000
0740	26	20,000

Figure 1

The syntactic objects used to describe sets of relation instances are called *Schemata. A relational schema* $\underline{R} = \langle \text{ATTR}, \Gamma \rangle$ is defined by a relation name R, a set of attributes ATTR and a set of predicates Γ that characterize the legal relation instances over T associated to the relation schema. In the following the only kind of predicates we will consider are functional dependencies. A *functional dependency* (FD) $X \rightarrow Y$ (where $X, Y \subseteq T$) holds in R (T) iff for every pair of tuples t_1, t_2 of R, $t_1.X = t_2.X$ implies $t_1.Y = t_2.Y$.

Given that a set of FDS, holds in a relation schema $\underline{R}$, one can infer other FDS that must hold in $\underline{R}$ as well. The set of all such FDS, called the *closure* of Γ and denoted Γ^+, can be derived using the following inference rules [1]:

1. If $Y \subseteq X$ then $X \rightarrow Y$ (*trivial dependencies*).
2. If $Z \subseteq W$ and $X \rightarrow Y$ then $XW \rightarrow YZ$.
3. If $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$.

Let now $\underline{R} = \langle \text{Attr}, \Gamma \rangle$, and let $X \subseteq \text{Attr}.X$ is called a *superkey* of $\underline{R}$ if $X \rightarrow \text{Attr} \in \Gamma^+.X$ is a *key* if X is a superkey but does not properly contain a superkey.

Finally a *Data Base Schema* is a collection $S = \{\underline{R}_1, \underline{R}_2, \ldots, \underline{R}_n\}$ of relation schemata.

The approaches to relational data base design existing in the literature [9] obtain a data base schema through two different steps:

1. In the first step, the semantic of reality is formally expressed in terms of a "unique relation" schema $S_U = \{ R_U \} = \{ \langle \text{ATTR}_U, \Gamma_U \rangle \}$ (called "Universal Relation Schema").

2. In the second step, a multi-relation Data Base Schema said *decomposition* of SU is derived, "equivalent" to S_U and "good" in some specified way. Several notions of equivalence, several "normal forms" and minimality properties for data base schemata were introduced in the literature in order to define good schemata and eliminate anomalous behaviours (*see* for instance [13, 14]).

2.1. In the *synthesis approach*, elementary data dependencies are processed and clustered in relation schemata such that a data base schema satisfying the desired minimality and normalization requirements is obtained.

2.2. In the *decomposition approach* the Universal Relation Schema is step by step decomposed until the desired requirements are satisfied.

In both approaches the final schema must at least satisfy the following requirements:

(*a*) it must contain the whole information that can be expressed by the original schema;

(*b*) it must inherit all the functional dependencies (basic and derivable) defined on the original schema.

Such requirements have been originally characterized in the context of functional dependencies by Rissanen [18] who introduced the concept of *independent decomposition.*

Several authors (*see* [15, 12, 19, 10, 11, 2, 16]) used in recent years graph formalisms in the data base area.

In former papers (*see* [3, 4]) we have formally investigated, within the formalism of hypergraph and hypergraph grammars, an approach to relational data base design in which the gathering of information requirements concerning the reality to be modelled and the modelling of such a knowledge into a data base schema are obtained through a set of refinement steps. If the rules by which data base schemata are incrementally specified, are suitably restricted, then several design properties can be dynamically mantained.

Within this approach, called "top-down approach to relational data base design", restrictions on derivations were investigated and classes of grammars were shown that always produce normalized schemata. As long as, during top-down design, new functional dependencies and relations are introduced in the schema by applying such rules, the enriched schema always enjoies the desired normalization properties.

In this paper we are interested to characterize in the formalism further design properties, both concerning functional dependencies (the *independent decomposition property*), and concerning another type of attribute dependencies, that we call *conceptual dependencies*. Intuitively, two attributes are *conceptually dependent* if the objects of the real world they represent are related by some fact of interest for the enterprise.

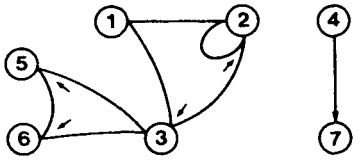
In section 2 and 3 of the paper the hypergraph and hypergraph grammars formalism suitable for our purposes is described; in section 4 several design properties concerning conceptual dependencies are characterized, by showing grammars that incrementally specify such dependencies avoiding anomalous side effects; finally in section 5 the independent decomposition property is characterized, by showing grammars that always produce schemata both independent decompositions of a given class of schemata and in (Boyce-Codd) normal form.

2. SCHEMA HYPERGRAPHS

As we said in the introduction, we aim in this section to represent in the hypergraph formalism structural properties of the relational model of data.

In the following a *Direct Hypergraph* (DH) is a pair $\langle N, S \rangle$ where: N is a finite set of *nodes* and S is a set of *direct surfaces* $\langle K_i; \bar{K}_i \rangle$ such that K_i, $\bar{K}_i$ are sets of nodes and $\cup_i (K_i \cup \bar{K}_i) = N$ (*node covering property*).

In a surface $s_i = \langle K_i; \bar{K}_i \rangle$, K_i is said the set of *source nodes and* $\bar{K}_i$ the set of *target nodes*. A DH is said *atomic* if $|S| = 1$; *see* in figure 2 a graph representation of a DH.



$$N = \{1, 2, 3, 4, 5, 6, 7\}$$

$$S = \{\langle 1, 2; 2, 3 \rangle, \langle 3; 5, 6 \rangle, \langle 4, 7 \rangle\}$$

Figure 2

$D' = \langle N', S' \rangle$ is *contained* in $D = \langle N, S \rangle$ (and we write $D' \sqsubseteq D$) if $N' \subseteq N$ and $S' \subseteq S$.

Given $D = \langle N, S \rangle$ and $D' = \langle N', S' \rangle$ such that $D' \sqsubseteq D$, D' is said *maximally connected component* (mcc) of D iff:

(1) N' is a maximal set of pairwise connected nodes.

(2) S' is the set of surfaces of S with nodes all in N' where two nodes n_1, n_2 are said *connected* if they belong to the same surface, or if a node n_3 exists connected both to n_1 and n_2.

Let now Σ be an alphabet of labels. A *Node Labelled direct Hypergraph* (NLH) is a pair $H = \langle D, \varphi \rangle$ where D is a DH and φ: $N \to \Sigma$ is a *node labelling function.* The set of labels of H (of the set $N' \subseteq N$) *is denoted* Φ (H) $(\Phi\ (N'))$.

An NLH is said *compact* if no pair of nodes share the same label.

Two NLHs H, H' are said *isomorphic* $(H \equiv H')$ if a one to one correspondence exists between pairs of nodes of H and H' such that for every surface $\langle K_i; \overline{K_i} \rangle$ of H a surface $\langle K_j; \overline{K_j} \rangle$ of H' exists such that $\Phi(K_i) = \Phi(K_j)$ and $\Phi\ (\overline{K_i}) = \Phi'\ (\overline{K_j})$ (and vice versa). In the following $[H]$ will denote the equivalence class of NLHs isomorphic to H.

Let $H = \langle D, \varphi \rangle$ and $H' = \langle D', \varphi \rangle$ be NLHs. We say that H' is a *subhypergraph* of H $(H' \subseteq H)$ if $H'' \in [H']$ exists such that $D'' \sqsubseteq D$ and $\varphi'' \subseteq \varphi$.

Finally, let H be an NLH and μ a labelled surface of H; the mcc $M' \subseteq H$ that contains μ is denoted M (H, μ).

Several operations on NLHs may be defined.

Given $H = \langle N, S, \varphi \rangle$ and $H' = \langle N', S', \varphi' \rangle$ let $H'' = \langle N'', S'', \varphi'' \rangle$ be such that $H'' \equiv H'$ and $N'' \cap N' = \varphi$. We call *union* of H and H' $(H \cup H')$ any $\overline{H}$ isomorphic to the SH: $\langle N \cup N'', S \cup S'', \varphi \cup \varphi'' \rangle$.

Let H, H', H'' be such that $H' \subseteq H$ and $H'' \subseteq H$, H'' is the *complement* of H' with respect to H (and vice versa), in the following H-H', iff:

$$N' \cup N'' = N, \qquad S' \cup S'' = S, \qquad S' \cap S'' = \varphi.$$

A *Data Base Schema Hypergraph* (SH) is a NLH whose mccs are all compact. In the following $\mathscr{H}$ is the set of all SHs.

In table I the correspondence between the hypergraph and relational terminology is shown.

Notice that:

1. Compacteness of mccs corresponds to the uniqueness of attribute names inside a relation.

2. We do not need to represent names of relations in the hypergraph formalism: this corresponds to assume that every attribute has the same meaning in every relation schema in which it appears.

3. To semplify our investigation we allow only one kind of surfaces, that represent in the following *functional dependencies.* We could extend the

formalism replacing S with a collection of sets of surfaces $\{S_1, S_2, \ldots, S_n\}$ corresponding to n different kinds of data dependencies (e. g. multivalued dependencies, *see* [13]).

TABLE I

Hypergraph formalism	Relational formalism
Node	Column number inside a table
Label	Attribute name
Surface	Data dependency among attributes
Isomorphism	Immateriality of column order
Connection among a set of labelled nodes	Membership of a set of attributes to the same relations
Isomorphism class of a maximally connected component	Relation schema
Isomorphism class of a Data base Schema Hypergraph	Data base schema

Example 2.1. The n-ary schema

Student (Student #, Course #, Grade, Student Age)

Course (Course #, Course Credit)

may be represented as an SH where:

$N = \{1, 2, 3, 4, 5, 6\}$

$S = \{\langle 1; 1\rangle \langle 1, 2; 3\rangle, \langle 1; 4\rangle, \langle 5; 6\rangle\}$

$= \{(1, \text{Student \#}), (2, \text{Course \#}), (3, \text{Grade}), (4, \text{Student Age}), (5, \text{Course \#}), (6, \text{Course Credit})\}$

See in figure 3 the corresponding graphical representation.

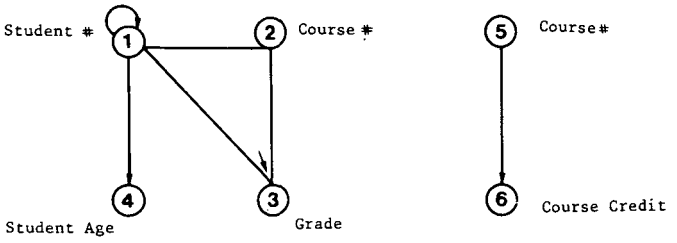


Figure 3

Till now we have represented in the hypergraph formalism only *relational structures.* In the relational theory a set of operators has been defined to manipulate relation instances. Since we are interested to transform relation schemata, such transformations may be defined in the hypergraph formalism by the following operators.

Let $\{M_1, \ldots, M_n\}$ be a set of maximally connected SHs; the *Join* $(M_1, \ldots, M_n)$ is a compact SH, $H = \langle N, S, \varphi \rangle$ where:

$\forall i \ (1 \leqq i \leqq n)$, $\hat{M}_i = \langle \hat{N}_i, \hat{S}_i, \varphi_i \rangle \in [M_i]$ exists such that $\hat{M}_i \subseteq H$ and $S = \bigcup_{i=i}^{n} \hat{S}_i$.

A *compact SH associated to H* ($C(H)$) is a SH obtained from H joining together all mccs of H.

If the *Join* $(M_1, \ldots, M_n)$ is not connected then it is useful to define a *Full Join* as the SH obtained by the *Join* adding a new trivial surface $s = \langle \bigcup_i N_i; \Phi \rangle$. In figure 4 an SH, its *Join* and *Full Join* are shown.

Notice that from now on we generally omit node numbers in the graphical representation of NLHs.

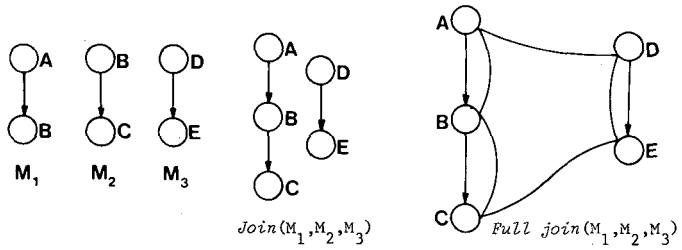


Figure 4

Let $M = \langle N, S, \varphi \rangle$ be a maximally connected SH and let $N' \subseteq N$, we define the *Projection of M over N'* (*Proj* $(M; N')$) in the following way:

Proj $(M; N') =$ *if* there exists an mcc $M' = \langle N', S', \varphi' \rangle$ s.t. $M' \sqsubseteq M$ and M' is maximal with respect to $\sqsubseteq$,

then M'

else Proj $(\langle N, S \cup \langle N'; \Phi \rangle, \varphi; N')$.

Let $\Sigma' \subseteq \Phi(M)$. We define *Proj* $(M; \Sigma')$ as *Proj* $(M; N')$, where $N' = \{N \mid \varphi(N) \in \Sigma'\}$.

See in figure 5 an SH an one of its *Projs*.

As we said in the introduction, functional dependencies are characterized in the relational theory by set of inference rules. A partial order among SHs, that can be seen as an index of notational redundancy, is now introduced by using such rules in our formalism. Intuitively, H' is redundant with respect to H if it contains all the surfaces of H plus a set of surfaces derivable from them.

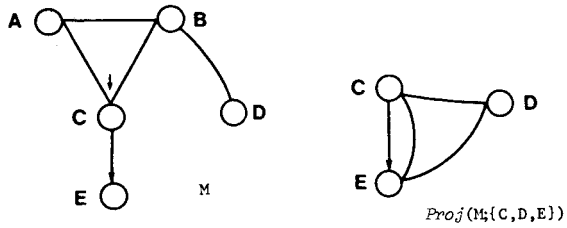


Figure 5

Formally let $H = \langle N', S', \varphi' \rangle$, we denote $H \leqq H'$ if H' is isomorphic to an SH obtained from H enriching S with a set of surfaces obtained using FD's inference rules:

(*a*) $\forall\, \bar{K} \subseteq K \subseteq N \Rightarrow s = \langle K; \bar{K} \rangle$ (*reflexivity*);

(*b*) $\forall \{K; \bar{K}\rangle \in S$ and $K' \subseteq \bar{K}' \subseteq N \Rightarrow s = \langle K \cup K'; \bar{K} \cup \bar{K}' \rangle$ (*augmentation*);

(*c*) $\forall \langle K; \bar{K}\rangle \in S$ and $\langle \bar{K}; \bar{\bar{K}} \rangle \in S \Rightarrow s = \langle K; \bar{\bar{K}} \rangle$ (*transitivity*).

H' is a *nonredundant cover* of H if $H' \leqq H$ and H' is a minimal element with respect to $\leqq$. We usually represent a SH with one of its nonredundant covers.

We say that H' is 1-*equivalent* to H ($H' \overset{1}{\equiv} H$) *if* $H' \leqq H$ or $H \leqq H'$or $\hat{H}$ exists such that $H' \overset{1}{\equiv} \hat{H}$ and $\hat{H} \overset{1}{\equiv} H$.

We call *complete* (*closure*) *SH associated to* H (H^+) an SH 1-equivalent to H, maximal element with respect to the partial order $\leqq$. Trivially, for every pair H, $\hat{H}$ such that $H \overset{1}{\equiv} \hat{H}$, $H^+ \equiv \hat{H}^+$ (*uniqueness property*).

We can now recall the definition of *independent decomposition* in the relational theory.

Intuitively, given a "unique relation" schema

$$S_u = \{R_u\} = \{\langle \mathrm{ATTR}_u, \Gamma_u \rangle\}$$

a schema S, decomposition of S_u is said *independent* [5] if:

(1) all the facts that can be represented in S_u, are representable in S (*lossless join property*);

(2) all the functional dependencies derivable in S_u, are derivable in A (*faithful property*).

In our formalism, let $M = \langle N, S, \varphi \rangle$ be an mcc and N', $N'' \subseteq N$ where $N' \cup N'' = N$ and $N' \cap N'' \neq \Phi$. We say that an SH $H_d = Proj\ (M;\ N') \cup Proj\ (M;\ N'')$ is a *decomposition* of H.

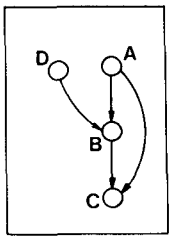
Furthermore, if the following properties hold:

(*a*) $C\ (H) \overset{1}{\equiv} C\ (H_d)$;

(*b*) $\langle N' \cap N''; N' \rangle$ or $\langle N' \cap N''; N'' \rangle$ is in S^+ (the set of surfaces of H^+), the decomposition is said *independent.*

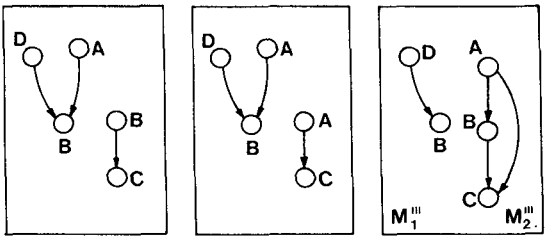
Given H, $H' \in \mathscr{H}$ we say that H' is an *independent decomposition of* H if its mccs can be obtained from mccs of H through a set of independent decompositions.

For example, given the SH of figure 6, H', H'' and H''' in figure 7 are all decompositions of H but only H' is independent.



H

Figure 6



H' H'' H'''

Figure 7

Infact, surface $\overset{B}{\bigcirc} \rightarrow \overset{C}{\bigcirc}$ appears in C (H') but is not derivable in C (H''), while B, the only label shared by the two mccs of H''', is a key neither in M_1''' nor in M_2''' of H'''.

Trivially, the former definitions can be shown to be equivalent to the property of independent decomposition given in [8] and the property Rep 4 given in [5].

3. THE FORMALISM OF HYPERGRAPH GRAMMARS

As we said in the introduction, the aim of this paper is to formally investigate an approach to relational data base design in which the gathering of information requirements concerning the reality to be modelled, and the modelling of such a knowledge into a data base schema are obtained through a set of refinement steps, and to study suitable restrictions to the rules by which the data base schema is incrementally specified in order to dynamically mantain design properties.

In the hypergraph formalism we need a concept of (context free) rewriting rule and *hypergraph grammar* to characterize top-down schema generation.

Informally, a rewriting rule describes the way in which an atomic object of the SH, for example a surface, can be replaced by a more complex structure.

We choose labelled surfaces as the atomic objects to be expanded and schema hypergraphs as replacing structures.

We define a *rewriting rule* as a 3-ple $\langle p, \varphi_\alpha, \varphi_\beta \rangle$ where:

$p = \langle \alpha, \beta, h \rangle$ is a *pattern production* such that:

$-\alpha = \langle N_\alpha, S_\alpha \rangle$, the *left member*, is a direct surface;

$-\beta = \langle N_\beta, S_\beta \rangle$, the *right member*, is a DH.

$-h$: $N_\alpha \to P(N_\beta)$, is the *embedding function*, such that $\forall n, m \in N_\alpha$:

(i) $n \neq m \Rightarrow h(n) \cap h(m) = \varphi$;

(ii) an mcc, the *link*, exists in β containing the codomain of h.

$\varphi_\alpha: N_\alpha \to \Sigma$ and $\varphi_\beta: N_\beta \to \Sigma$ are *labelling functions*.

In the following β° is $\langle \beta, \varphi_\beta \rangle$ and λ is the *Proj* of β° with respect to the nodes of the link. Furthermore, we represent a pattern production as in figure 8, in which equal symbols mark nodes of α and β that are in the correspondence defined by h.

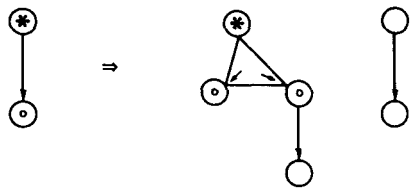


Figure 8

Let now H be an SH, $d = \langle \langle \alpha, \beta, h \rangle, \varphi_\alpha, \varphi_\beta \rangle$ be a rewriting rule and γ be a labelled surface of H s. t. $\gamma \equiv \langle \alpha, \varphi_\alpha \rangle$. The NLH H' is said *directly derivable* from H via the rewriting rule d applied to γ ($H \overset{d, \gamma}{\to} H'$) if H' can be obtained by H replacing γ with β°, using function h to specify for every node of γ the sets of nodes of β° that inherit membership to preexisting surfaces of H adjacent to γ.

The following theorem gives necessary and sufficient conditions for H' to be an SH.

THEOREM 2.1: *Let H be an* SH, $d = \langle p, \varphi_\alpha, \varphi_\beta \rangle$ *a rewriting rule, γ a labelled surface, H' the NLH such that $H \overset{d, \gamma}{\to} H'$. H' is an* SH *iff:*

$$\Phi(\lambda) \cap \Phi(M(H, \gamma) - \gamma) = \Phi.$$

Proof of the if part: Trivially from the hypothesis, in the table expansion of $M(H, \gamma)$ no pair of nodes can have the same label; so H' is an SH.

Proof of the only if part: If H' is an SH, no label for the link can be choosen in $\Phi(M(H, \gamma))$ except for labels already in γ.

Q. E. D.

We assume in the following that property expressed by theorem 2.1. always holds in derivations.

Remark: Notice that if several $\gamma_i \equiv \langle \alpha, \varphi_\alpha \rangle$ exist in H the rewriting rule can be simultaneously applied to all γ_i, since at last one γ_i for every mcc exists.

In the following we denote H' with the notation $\{H; d, \gamma\}$.

See in figure 9 an example of derivation.

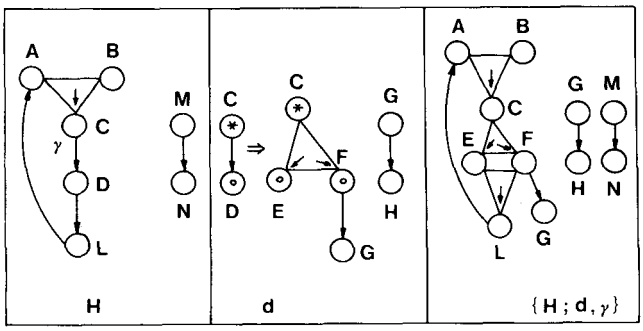


Figure 9

Since we are interested to "structurally" characterize classes of rewriting rules that guarantee particular design properties, we have to define a concept of grammar.

A *schema hypergraph grammar* is a 2-pla $G = \langle A, R \rangle$ where:

- A is a finite set of SHs (*axioms*);
- R is a set of a rewriting rules.

We say that H' is *derivable* in the grammar $G = \langle A, R \rangle$ if a finite chain of SHs $H_0, \ldots, H_n$ exists such that $H_0 \in A$, $H_n = H'$ and $\forall i\ (0 \leqq i \leqq n-1)$ d_i, γ_i exist with d_i in R such that $H_i \xrightarrow[d_i, \gamma_i]{} H_{i+1}$.

Given a grammar G, we define *language* associated to G the set of all SHs derivable in G.

4. RENAMING BOUNDS AND DEPENDENCY PROPERTIES IN TOP-DOWN RELATIONAL DATA BASE DESIGN

We will use now the formal machinery introduced in section 3 to characterize in the top-down approach to data base schema design, several design properties concerning data dependencies. In particular we are interested to properties that can be achieved with suitable restrictions on renamings of objects in derivations. Such restrictions can be formally represented as properties of labelling functions used in schema hypergraph grammars.

In this section, we are interested to a special kind of dependencies, that we call *conceptual* and *logical dependencies.*

Intuitively, two attributes are *conceptually* (*c*-) *dependent* if the objects of the real world they represent are related by some facts of interest for the enterprise. Otherwise, we call them *conceptually independent.*

Two attributes are said *logically* (*l*-) *dependent* (*l-independent*) if they appear (do not appear) in the same relation schema.

See in table III the correspondence between relational and hypergraph formalism.

Table III

Property	Relational formalism	Hypergraph Formalism
Conceptual dependence	Attributes are related by some facts	Nodes in the compact DBH are connected (belong to the same mcc)
Logical dependence	Attributes are in the same relation	Nodes are connected

In figure 10, for instance, A and B are l-dependent, A and C are l-independent and c-dependent, A and D are c-independent.

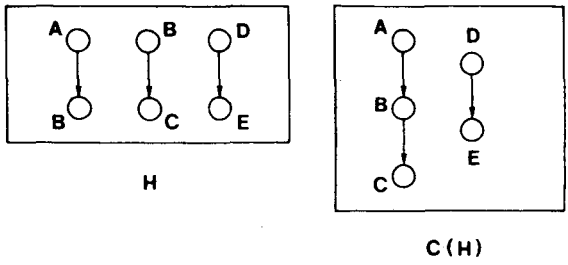


Figure 10

During the design of a schema, one of the most relevant decisions concerns *l*- and *c*-dependencies between concepts; if, for instance, a derivation step splits one mcc into several mccs, the motivation can be either to achieve *l*-independence (for instance, to get some kind of normalization) or to achieve *c*-independence (according to the semantics of the real world).

In a top-down design we may impose that in such derivations a discipline is imposed to avoid the occurrence of undesirable side effects.

With regard to *l*-dependencies, notice that in the definition of rewriting rule property (i) guarantees that two *l*-dependent nodes cannot collapse in a derivation step, while property (ii) guarantees that all preexisting *l*-dependencies are mantained.

Finally, the context free character of rewriting rules (the left member is a surface), guarantees that *l*-independencies too are mantained.

With regard to *c*-dependencies let us examine now in figure 10 four different instances of derivations in which several undesirable side effects concerning the *c*-dependence structure occur. We put in evidence with dashed lines *c*-dependent attributes in different mccs.

We comment now the four cases.

Case 1: The preexisting *c*-dependency structure in H, due to the labelled surface γ to be expanded, is modified in H'_1; in fact, for instance, A and F are *c*-dependent in H and *c*-independent in H'_1.

Case 2: In H'_2, on the other side, new *c*-dependencies have been introduced by λ among previously *c*-independent concepts (e. g. D and Y).

Case 3: Similarly, in H'_3 new *c*-dependencies are introduced between previously *c*-independent concepts (D and H) by a part of β° ($\underset{E}{\bigcirc} \rightarrow \underset{Y}{\bigcirc}$) *c*-independent from λ in the rewriting rule.

Case 4: Finally, in H'_4 a new *c*-dependency (e. g. N and B) is introduced between concepts that are on the other side considered *c*-independent in the rewriting rule.

From the above examples it is clear that several different types of anomalies, with regard to the previously stated discipline, can occur. We are now interested to characterize (in terms of properties of φ_α and φ_β) the design properties that express the absence of anomalies.

First of all, we are interested to characterize (in terms of properties of φ_α and φ_β) the class of rewriting rules that preserve "*c*-dependency structure".

A first property that must be preserved in the derivations is *monotonicity of c-dependencies*: i. e. when a *c*-dependence exists between nodes in H, such a dependence must be preserved in H'.

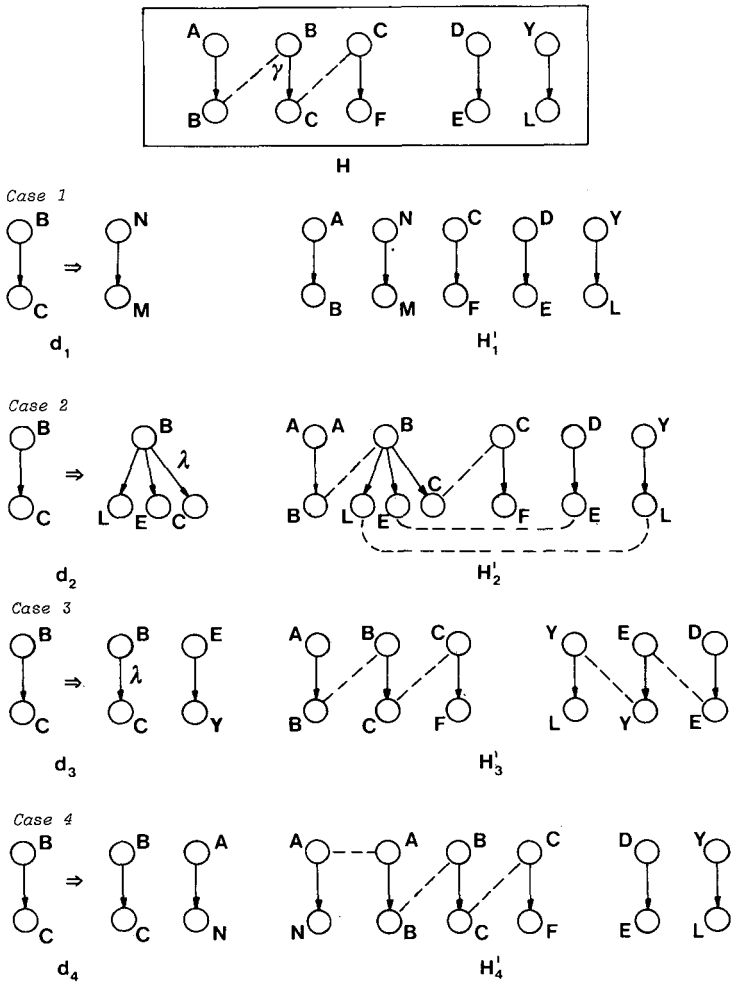


Figure 11

Let $H \in \mathscr{H}$ and $\delta \subseteq H$. In the following (see *fig.* 12):

(*a*) H_δ is the maximal subhypergraph of H such that $C(H_\delta) \equiv M(C(H), \delta)$;

(*b*) given a derivation $H \overset{d, \gamma}{\rightarrow} H'$ we denote λ', $\beta^{\circ\prime}$ and $(H\text{-}H_\gamma)'$ the sub hypergraphs of H' isomorphic to λ, β° and $H\text{-}H_\gamma$.

We are interested to investigate two different types of monotonicity.

First of all, monotonicity can be mantained by *c*-dependencies in β^0_λ. Formally: given the derivation $H \overset{d, \lambda}{\rightarrow} H'$ we say that such derivation preserves

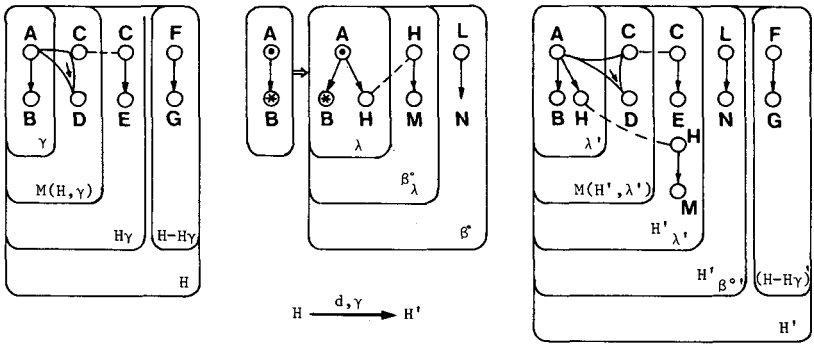


Figure 12

level 1 *monotonicity* if, given the derivation $H_\gamma \overset{d^*,\gamma}{\rightarrow} H'_\gamma$ where $d^* = \langle p^*, \varphi_\alpha, \varphi_\beta \rangle$, $p = \langle \alpha, \eta, h \rangle$ with $\eta \sqsubseteq \beta$ and $\eta^\circ = M\ (C\ (\beta^\circ), \lambda)$, $C\ (H'_\gamma)$ is connected (all its nodes are 1-dependent).

Even strongly, we may ask that the monotonicity is mantained only for 1-dependencies in λ. Formally, the above definition must hold with λ instead of η° (*level* 2 *monotonicity*).

See in figure 13 a derivation that preserves level 1 monotonicity without preserving level 2.

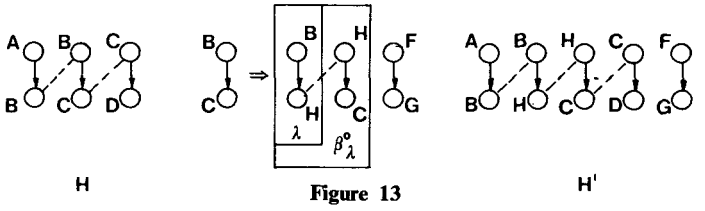


Figure 13

Case 1 on figure 11 shows a derivation in which both level 1 and level 2 monotonicity are not mantained.

We show now for level 1 monotonicity (the theory is similar for level 2) several characterizations for monotonicity, expressed in terms of renaming bounds.

THEOREM 4.1: *Level* 1 *monotonicity is achieved in a derivation* $H \overset{d,\gamma}{\rightarrow} H'$ *iff* $\varphi\ (\beta^0_\lambda)$ *contains at least one label for every mcc in* $C\ (H_\gamma - M\ (H, \gamma))$.

Proof: If the condition expressed by theorem 4.1 holds, *c*-dependencies in H_γ are all mantained by β^0_λ, and so the number of mccs in the compact of the image of H_γ (i.e. H'_γ in definition of level 1 monotonicity) cannot increase.

Assume now that level 1 monotonicity is achieved and an mcc μ in C $(H_\gamma - M\ (H, \gamma))$ exists with no label common to $\Phi(\beta^0_\lambda)$. Obviously, no node in the image of μ can be c-dependent to some node of the image of H_γ.

Q. E. D.

The following weaker property is only sufficient but more easy to check.

COROLLARY 4.1: *Level* 1 *monotonicity is achieved in* $H \overset{d,\gamma}{\rightarrow} H'$ if:

$$\Phi(\beta^0_\lambda) \supseteq \Phi(\gamma) \cap \Phi(H_\gamma - \gamma)$$

Proof: In this case labels required in the if condition of theorem 4.1 are choosen on the border between γ and H_γ.

Q. E. D.

A second property, concerning the c-dependency structure, is *conceptual context freedom* (ccf): i. e. in a derivation $H \overset{d,\gamma}{\rightarrow} H'$ the c-dependency structure of β° must not be influenced by the c-dependency structure of H, and vice versa.

Conceptual context freedom can be formally expressed by the following formulas:

Property (*a*):

$$C(H - H_\gamma) \equiv C(H') - C(H'_{\beta^{0\prime}}).$$

Property (*b*):

$$C(\beta^0 - \beta^0_\lambda) \equiv C(H') - \{C[(H - H_\gamma)'] \cup C(H'_{\lambda'})\}.$$

Property a specifies that the c-dependency structure among the concepts of H c-independent from γ must remain invariant during the derivation; the same invariance has to be mantained for the part of β° c-independent from the link λ (property *b*). Coming back to example in figure 11, property *a* is not verified in case 2 and case 3, and property *b* is not verified in case 3 and case 4.

A trivial sufficient condition for ccf is:

Fact 4.1: In the derivation $H \overset{d,\gamma}{\rightarrow} H'$ ccf is mantained if:

$$\Phi(\beta^0) \cap \Phi(H) = \emptyset.$$

The above condition can be weakened if the condition expressed in corollary 4.1 is satisfied (i. e. β^0 contains labels in the border between γ and H_γ).

THEOREM 4.2: *In the derivation* $H \overset{d,\gamma}{\rightarrow} H'$ ccf *is mantained if the following conditions hold:*

(i) $\Phi(\beta^0_\lambda) \supseteq [\Phi(\gamma) \cap \Phi(H_\gamma - \gamma)]$;

(ii) $\Phi(\beta^0) \cap [\Phi(H) - \Phi(\gamma)] = \emptyset$.

Proof: Level 1 monotonicity, implied by condition (i), avoids breaking old c-dependencies in $H - H_\gamma$ while condition (ii), avoids connecting in the derivation c-independent mccs of $H - H_\gamma$; so property (a) of ccf is verified.

Furthermore, for condition (ii) the only old labels in H reused in the derivation belong to γ, and so specify c-dependent concepts both in H and in β°. This property implies property b.

Q. E. D.

We give in the following theorem necessary and sufficient conditions for conceptual context freedom.

THEOREM 4.3: *In the derivation* $H \overset{d,\gamma}{\rightarrow} H'$ ccf *is mantained iff:*

(i) $\Phi(\beta^0_\lambda) \cap \Phi(H - H_\gamma) = \emptyset$;

(ii) $\Phi(\beta^0 - \beta^0_\lambda) \cap \{[\Phi(H) - \Phi(\gamma)] \cup \Phi(H_\gamma - \gamma)\} = \emptyset$.

Proof: if part — Property (i) prevents from creating c-dependencies among $(H - H_\gamma)'$ and the remainder of H'. Property (ii) prevents from creating c-dependencies from $\beta^0 - \beta^0_\lambda$ to both H'_λ, and $(H - H_\gamma)'$; in fact we can reuse in such SH only labels of γ nor belonging to its border with the remainder of H.

Only in part: If in $H \overset{d,\gamma}{\rightarrow} H'$ the ccf holds, from property (a) of the definition we can derive:

a')
$$\Phi(H - H_\gamma) \cap \Phi(H'_{\beta^{0\prime}}) = \emptyset,$$

that implies condition (i) of the theorem.

Furthermore, property (a) implies also:

a'')
$$\Phi(\beta^{0\prime} - \beta^{0\prime}_\lambda) \cap \Phi(H - H_\gamma) = \emptyset,$$

and property b) implies:

b') $$\Phi(\beta^0 - \beta^0_\lambda) \cap \{[\Phi(H_\gamma) - \Phi(\gamma)] \cup \Phi(H_\gamma - \gamma)\} = \emptyset.$$

Finally, a'') and b') imply (ii).

Q. E. D.

Finally, stronger conditions are necessary if we wish to guarantee the monotonicity of *functional interrelational dependencies* in the derivation steps. See for instance the derivation of figure 14.

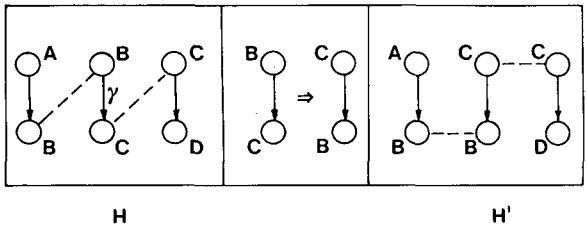


Figure 14

Dependency $\overset{A}{\bigcirc} \rightarrow \overset{D}{\bigcirc}$ does not belong to H^+ but can be derived in $C(H)$ (and therefore is an interrelational functional dependency) but cannot be derived in $C(H')$. We say that *monotonicity of functional (intra- and inter-relational) dependencies* holds in the derivation $H \overset{d,\gamma}{\rightarrow} H'$, if every surface derivable in $C(H)$, not including nodes of γ, is also derivable in $C(H')$.

THEOREM 4.4: *In a derivation* $H \overset{d,\gamma}{\rightarrow} H'$, *where*

$$d = \langle \langle \alpha, \beta, h \rangle, \varphi_\alpha, \varphi_\beta \rangle,$$

the monotonicity of functional dependencies is mantained if:

(*a*) *a surface s exists in* λ^+ *such that the embedding function maps source nodes of* λ *in source nodes of s and target nodes of* γ *in target nodes of s;*

(*b*) *for every node n in* α, $\varphi_\alpha(n) = \varphi_\beta(h(n))$.

Proof: If condition *a*) is satisfied, then all interrelational dependencies of H are also mantained in H'; in fact surface $\langle s, \varphi_\beta \rangle$ can be used instead of γ in all inference rules applied to H'. For interrelational functional dependencies, condition *b*) allows using the same surface s to derive in $C(H')$ all the dependencies of $C(H)$.

Q. E. D.

The last conditions that we want to examine in this section concern another important goal in relational data base design: "minimality" of the data base schema. A very weak definition of minimality is the following one: a schema is *minimal* if no other schema exists using exactly the same global set of functional dependencies with a lower number of relational schemata.

In our formalism such a notion of minimality corresponds to compactness.

THEOREM 4.5: *Let* $H \overset{d,\gamma}{\rightarrow} H'$ *be a derivation, where* H *is minimal,* H' *is minimal iff:*

$$\beta^0 \text{ is minimal and } \Phi(\beta^0) \cap [\Phi(H) - \Phi(\gamma)] = \emptyset.$$

Proof: If β^0 is minimal then all its nodes have different labels, so in the derivation step, if a label of γ is reused in β^0 it can be used only once. No other label in it is allowed by hypothesis; then H' is minimal.

If H' is minimal, any of its subset of tables has to be minimal; than β^0 is minimal. H is minimal, so in the derivation only labels in $\Phi(H)$ that occur in $\Phi(\gamma)$ can be used in λ and in $\beta^0 - \lambda$ can be used only labels of $\Phi(\gamma)$ that have not been used in λ.

Q. E. D.

The conditions of theorem 4.5 are not necessary for the minimality of an SH obtained as final result of a derivation chain. In fact, it is possible by suitable renamings to delete labels that prevent minimality.

If we denote *label perserving* a derivation chain such that no label can be deleted in any derivation step $H_i \overset{d_i,\gamma_i}{\leftarrow} H_{i+1}$ [i. e. $\Phi(H_{i+1}) \supseteq \Phi(H_i)$], it is clear that:

Fact 4.2: Let $H_0 \overset{d_0,\gamma_0}{\leftarrow} H_1 \overset{d_1,\gamma_1}{\leftarrow} \ldots \overset{d_{n-1},\gamma_{n-1}}{\leftarrow} H_n$ be a label preserving derivation chain; H_n is minimal only if H_0 is minimal and every derivation step preserves minimality.

5. RENAMING BOUNDS, INDEPENDENT DECOMPOSITIONS AND NORMALIZATION PROPERTIES

In this section we turn our attention to independent decomposition properties during top-down design, and look for grammars that have the property of generating only SHs that are independent decompositions of a particular class of SHs. In fact in its full generality the problem is complex and cannot be solved in the framework of context free grammars.

We say that $H \in \mathcal{H}$ is a *tree SH* (TH) if:

(*a*) every surface $\langle K, \bar{K} \rangle$ of H has $K \cap \bar{K} = \emptyset$;

(*b*) there is a tree ordering $\leqq$ among all pairs of surfaces s', s'' of H $s' \leqq s''$, defined in the following way:

if $[s' = \langle K', \bar{K}' \rangle$ *and* $s'' = \langle K'', \bar{K}'' \rangle$ *are such that* $\Phi(s') \cap \Phi(s'') \neq \emptyset]$

then $\Phi(K') \subseteq \Phi(K'')$ *and* $\Phi(K') \cap (K'' \cup \bar{K}'') = \emptyset$;

else s''' *exists such that* $s' \leqq s''' \leqq s''$.

In figure 15 is shown a TH.

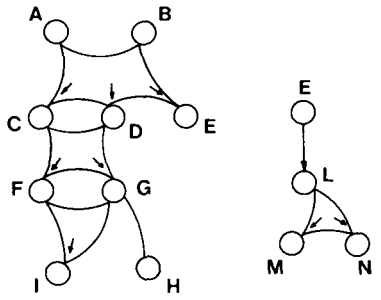


Figure 15

First of all we may observe that the following fact clearly holds.

Fact 5.1: All THs are non redundant coverings, and are in BCNF if and only if their mccs have an atomic non redundant cover.

We want now to show that the class of SHs in BCNF is "adequate to represent" the class of THs.

THEOREM 5.1: *Let H be a* TH *a* TH *H′ in* BCNF *exists such that H′ is an independent decomposition of H.*

Proof: H' may be obtained, for instance, with a chain of decompositions $H_i \to H_{i+1}$ such that $H_{i+1} = \text{Proj}\,(H_i; N_{i1}) \cup \text{Proj}\,(H_i; N_{i2})$ where N_{i2} is the set of nodes of a "leaf" surface of H_i (in the partial order $\leqq$) and N_{i1} the set of nodes of the complement set of surfaces; if the last SH H_n of the chain has all atomic mccs, it is an independent decomposition of H, since the corresponding properties are trivially verified, and it is in BCNF, since everyone of its mccs is atomic.

Q. E. D.

We are now interested to find grammars that produce all THs in BNCF (we call TBCNF such a class) and use node renaming bounds as constraint to the generative power of the grammar. We express such renaming bounds by using a graphical representation, enclosing all the nodes with the same label into dashed lines. See, for instance *fig.* 16.

In this case, in every application of a rewriting rule corresponding to this pattern production, the same label must be used for nodes {1, 3} and {2, 4, 5}.

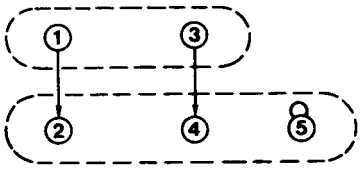


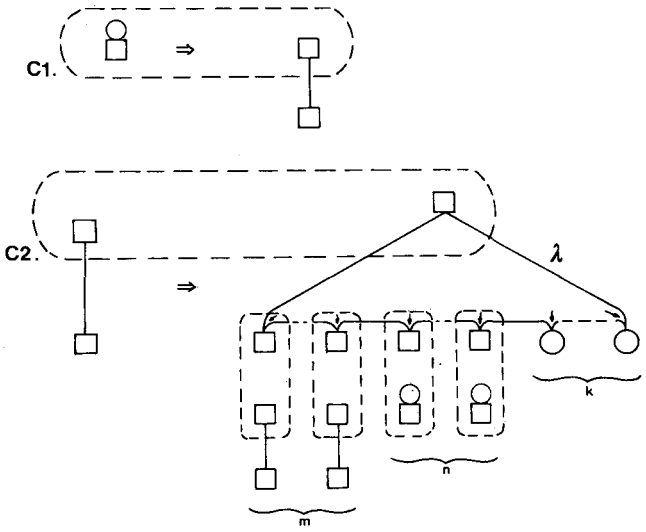
Figure 16

A grammar is said IT-canonical if:

1. the axiom is ○□.

where symbol □ denotes a nonempty set of nodes with the same role (source, target) in all surfaces in which they appear, and symbol ∩ denotes a surface with only source nodes.

2. Its rewriting rules belong to the following classes:



in which:

2.1. $m+n+k \geqq 1$;

2.2. dashed lines enclose equipotent sets of nodes;

2.3. renaming bounds impose an identity between the corresponding *sets* of associated labels.

3. During derivations, conceptual context freedom is mantained.

THEOREM 5.2: *Let G be an IT-canonical grammar, then* $\mathscr{L}(G) \subset TBCNF$. *Given any* $H \in TBCNF$, *there exists an IT-cononical grammar such that* $H \in \mathscr{L}(G)$.

Proof of the first part: Starting SHs are obviously elements of TBCNF. Suppose $H \in$ TBCNF is the resulting SH of a given derivation chain. Let now $H \xrightarrow{d,\gamma} H'$, where $d = \langle p, \varphi_\alpha, \varphi_\beta \rangle$, be a derivation step.

First case: $d \in C_1$: If H is the starting SH in G then obviously $H' \in$ TBCNF. If γ has been generated by a previous application of a rewriting rule $d' \in C_2$, then H' differs from H for the substitution of an atomic SH with a new surface whose source nodes have the same set of labels (see *fig.* 17).

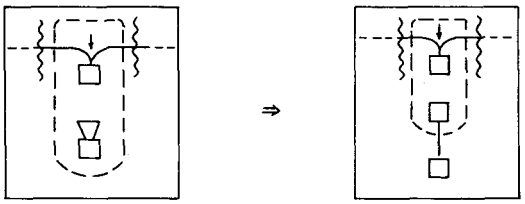


Figure 17

Since the derivation step respects conceptual context freedom, then the only common labels of the new surface with respect to the old SH are associated to the nodes inside the dashed lines. So $H \in$ TBCNF.

Second case: $d \in C_2$: In this case $\beta^\circ \in$ TBCNF and the source nodes of λ inherit all the labels of the source nodes of γ. Furthermore, because of conceptual context freedom, no old label can be duplicated. So the structure of tree SH is mantained, and so the new SH $\in$ TBCNF.

Proof of the second part: We show a recursive algorithm to generate any $H \in$ TBCNF.

First of all, apply $d \in C_1$ to generate the root of the tree. Suppose now we were able to generate all the surfaces at distance $i \leqq n$ from the root of H. Every surface s' at distance $n+1$ can be generated by applying once a rewriting rule $d \in C_2$ to *the* unique surface s'' at distance n from the root such that $s' \leqq s''$ (Notice that in such canonical generation we always apply rewriting rules with $m+n=1$).

Q. E. D.

REFERENCES

1. W. W. ARMSTRONG, *Dependency Structures of Data Base Relationships*, Proc. IFIP 74, 1974, pp. 580-583.

2. G. Ausiello, A. D'Atri and D. Sacca', *Transitive Closure and Other Graph Algorithms for the Synthesis and Manipulation of Data Base Schemas*, Graph theoretic Concepts in Computer Science, L.N.C.S., Vol. 100, 1981, pp. 212-223.
3. C. Batini and A. D'Atri, *Rewriting Systems as a Tool for Relational Data Base Design*, Graph Grammars and their applications to Computer Science and Biology, L.N.C.S., Vol. 73, 1978, pp. 139-154.
4. C. Batini and A. D'Atri, *Schema Hypergraphs: A Formalism to Investigate Logical Data Base Design*, Graph Theoretic concepts in Computer Science, L.N.C.S., Vol. 100, 1981, pp. 177-194.
5. C. Beeri, P. A. Bernstein and N. Goodman, *A Sophisticate Introduction to Data Base Normalization Theory*, Proc. 4th Int. Conf. on Very Large Data Bases, 1978, pp. 113-124.
6. P. A. Bernstein, *Synthesizing Third Normal Form Relations from Functional Dependencies*, A.C.M. Trans. on Data Base Systems, Vol. 4, 1976, pp. 277-298.
7. E. F. Codd, *A Relational Model for Large Shared Data Banks*, Com.A.C.M., Vol. 13, 1970, pp. 377-387.
8. E. F. Codd, *Further Normalization of the Data Base Relational Model*, in Data Base Systems, R. Rustin, Ed. Prentice Hall, 1972, pp. 33-64.
9. C. J. Date, *Introduction to Data Base Systems*, Addison Wesley, 1977.
10. V. De Antonellis, F. De Cindio, G. Degli Antoni and G. Mauri, *Use of Bipartite Graphs as a Notation for Data Base*, Information Systems, Vol. 4, 1979, pp. 137-141.
11. P. Degano, A. Lomanto and F. Sirovich, *On Finding the Optimal Access Path to Resolve a Relational Data Base Query*, MFSC'80, L.N.C.S., Vol. 88, 1980, pp. 219-230.
12. H. Ehrigh and H. J. Kreowsky, *Algebraic Theory of Graph Grammars Applied to Consistency and Synchronization in Data Bases*, Proc. Workshop WG 79 on Graphtheoretic Concepts in Computer Science, 1979.
13. R. Fagin, *Multivalued Dependencies and a New Normal Form for Relational Data Bases*, A.C.M. Trans. on Data Base Systems, 3, 1977, pp. 262-278.
14. R. Fagin, *The Decomposition Versus the Synthetic Approach to Relational Data Base Design*, I.B.M. Journal Res. Dev., 1977.
15. A. L. Furtado, *Transformation of Data Base Structures*, Graph Grammars and their Application to Computer Science and Biology, L.N.C.S., Vol. 73, 1978, pp. 224-236.
16. K. K. Nambiar, *Some Analytic Tools for the Design of Relational Data Bases*, Proc. 6th Conf. on Very Large Data Bases, 1980, pp. 417-428.
17. V. W. Lum *et al.*, 1978 *New Orleans Data Base Design Workshop Report*, I.B.M. Report RJ 2554, 1979.
18. J. Rissanen, *Independent Components of Relations*, A.C.M. Trans. on Data Base Systems 2, Vol. 4, 1977, pp. 317-325.
19. J. F. Sowa, *Definitional Mechanism of Conceptual Graphs*, Graph Grammars and their applications to Computer Science and Biology, L.N.C.S., Vol. 73, 1978, pp. 426-439.