QÜESTIIÓ, Vol 17, 1 pp. 75-101, 1993

MÉTODOS PARA LA COMPROBACIÓN DE LA INTEGRIDAD EN BASE DE DATOS DEDUCTIVAS

LAURA MOTA HERRANZ y MATILDE CELMA GIMÉNEZ*

Universidad Politécnica de Valencia

La comprobación de la integridad es un problema clásico en bases de datos; los primeros métodos fueron propuestos para simplificar la comprobación de restricciones estáticas en bases de datos relacionales extendiéndose posteriormente a las bases de datos deductivas. Estos métodos se basan en la idea común de evaluar instancias de las restricciones, obtenidas a partir de actualizaciones inducidas por la transacción, y se diferencian entre sí en la estrategia seguida para la instanciación y evaluación de las restricciones.

En este trabajo se presenta una clasificación de los métodos más importantes propuestos en la literatura haciendo un análisis de los mismos.

Methods for Integrity Checking in Deductive Databases.

Key words: Bases de Datos Deductivas, Restricciones de Integridad.

---

*Dept. Sistemas Informáticos y Computación. Universidad Politécnica de Valencia. email mati@dsic.upv.es.

-Article rebut el juny de 1991.

-Acceptat el desembre de 1992.

1. INTRODUCCIÓN

Las Bases de Datos Deductivas (BDD) extienden la capacidad expresiva de las Bases de Datos Relacionales (BDR) incorporando reglas que permiten derivar información a partir de la explícitamente almacenada.

El esquema de una BDD consiste en un conjunto de esquemas de relación de la forma: $R(A_1: D_1, \ldots, A_n: D_n)$ donde R es el nombre de la relación, $A_1, \ldots, A_n$ son identificadores de atributos y $D_1, \ldots, D_n$ los dominios asociados; y un conjunto de Restricciones de Integridad (RI). En este esquema se distinguen dos tipos de relaciones: básicas cuyas tuplas se almacenan explícitamente y derivadas definidas a partir de reglas deductivas. Una BDR es un caso particular de BDD sin relaciones derivadas. Un estado de BDD consiste en un conjunto de hechos (tuplas de las relaciones básicas) más un conjunto de reglas deductivas.

Desde la perspectiva de la lógica, una BDD puede formalizarse de la forma siguiente (Reiter (1984), Lloyd (1987b)):

- el esquema de la BDD se representa por un par (R, RI) donde:

  R es un lenguaje relacional definido a partir de los esquemas de relación y RI es el conjunto de restricciones de integridad (fórmulas cerradas de R)

- un estado D se representa por una teoría de primer orden definida en R:

$$D = \{A: A \text{ es un átomo base (hecho)}\}$$

$$\cup$$

$$\{A \leftarrow W: A \text{ es un átomo y } W \text{ una fórmula bien formada (fbf)}\}.$$

Como es sabido, las restricciones de integridad son propiedades que la base de datos debe satisfacer en cualquier instante. La evolución en el tiempo de una BDD puede describirse por una secuencia de estados donde dado un estado, D, su sucesor, D', se obtiene aplicando a D una transacción T (conjunto de inserciones y/o borrados de hechos y/o reglas). Dependiendo del número de estados implicados en la propiedad existen dos tipos de restricciones de integridad: restricciones estáticas que dependen sólo del estado actual de la base de datos y restricciones dinámicas que dependen de dos o más estados. En el presente trabajo sólo se tratarán las restricciones estáticas.

La comprobación de la integridad es un problema clásico en bases de datos; los primeros métodos fueron propuestos para la comprobación de restricciones estáticas en bases de datos relacionales extendiéndose posteriormente a las bases de datos deductivas. La forma más sencilla de comprobar las restricciones estáticas es evaluar cada una de ellas después de la transacción, sin embargo,

esta aproximación puede ser muy costosa en bases de datos voluminosas, ya que no explota el hecho de que la base de datos era íntegra (satisfacía todas las restricciones) antes de la transacción. Basándose en esta hipótesis, los métodos propuestos **simplifican la comprobación de la integridad,** evitando comprobar instancias de las restricciones que se satisfacían antes de la transacción y que además no se ven afectadas de ésta. Para que esta simplificación sea correcta las fórmulas que representan las restricciones de integridad deben ser independientes del dominio. Intuitivamente, una fórmula es independiente del dominio si su evaluación sólo depende de las extensiones de los predicados que aparecen en ella (Topor(1987)).

La estructura del trabajo es la siguiente: en el apartado 2 se presenta el método para la comprobación simplificada de la integridad en BDR de Nicolas que contiene las ideas básicas utilizadas en los métodos para BDD que se presentan en el apartado 3; en el apartado 4 se hace un análisis de estos métodos.

2. COMPROBACIÓN DE INTEGRIDAD EN BDR: MÉTODO DE NICOLAS (NICOLAS (1982))

El método de Nicolas consiste en dada una base de datos íntegra y una transacción T (conjunto de inserciones o borradas de tuplas) obtener *instancias simplificadas de las restricciones de integridad relevantes para la transacción* que será suficiente comprobar en D' para asegurar su integridad.

Restricciones de integridad relevantes para T

Si W es una restricción de integridad de rango restringido (independiente del dominio) podemos afirmar que:

> "***W* será relevante respecto a la inserción (borrado) de la tupla** $(e_1, e_2, \ldots, e_n)$ en R si y sólo si $R(e_1, e_2, \ldots, e_n)$ es unificable con un átomo que ocurre negativamente (resp. positivamente) en W"

Instancias de las restricciones de integridad

Sea:

- W una restricción de integridad relevante respecto a la inserción (resp. borrado) de $(e_1, e_2, \ldots, e_n)$

- ϕ el unificador más general (mgu) que unifica $R(e_1, e_2, \ldots, e_n)$ con un átomo que ocurre negativamente (resp. positivamente) en W
- θ la restricción de ϕ a aquellas variables cuantificadas universalmente no precedidas de un cuantificador existencial

entonces, definimos una **instancia de W** generada por la inserción (resp. borrado) de $(e_1, e_2, \ldots, e_n)$ como la fórmula $W\phi$.

Simplificación de la comprobación de la integridad

Si θ es el conjunto de sustituciones obtenidas de la forma anterior considerando todas las operaciones de la transacción T y W_s es la fórmula resultante de aplicar a $W\theta_1 \wedge W\theta_2 \wedge \cdots \wedge W\theta_n$ $(\forall\theta_i \in \Theta)$ las siguientes reglas de simplificación:

- sustituir cada ocurrencia de una tupla insertada (resp. borrada) por el valor cierto (resp. falso)
- aplicar reglas de absorción

entonces se cumple: ***D' satisface W si y sólo si D' satisface W_s.***

Si Θ es el conjunto vacío, la restricción W no se ve afectada por la transacción es decir, ésta sigue satisfaciéndose en D'.

Si una sustitución $\theta \in \Theta$ coincide con la sustitución identidad, la restricción W no se puede simplificar y debe ser evaluada en D' en su forma original.

El método de Nicolas que se ha presentado utiliza comp(D) como teoría de primer orden que representa el estado actual D, donde comp(D) es la compleción de D definida en Clark (1978). En esta semántica, si (R, RI) es el esquema de una BDR, se dice que el estado D satisface la restricción W ($W \in$ RI) si y sólo si W es consecuencia lógica de comp(D) (Reiter (1984)).

Ejemplo:

Sea:

$D = \{p(1,1), p(2,2), q(1,1,1), q(1,2,2), q(2,1,1)\}$ un estado íntegro de base de datos,

$\mathrm{RI} = \{W = \forall x \forall y (p(x,y) \rightarrow \exists z\, q(z,x,y))\}$ el conjunto de restricciones de integridad y

$T =$ {insertar $p(3,3)$, borrar $q(1,1,1)$} una transacción.

El método detecta que W es relevante respecto a ambas operaciones:

insertar $p(3,3)$: $\theta_1 = \{x/3, y/3\}$
$W\theta_1 = p(3,3) \rightarrow \exists z\, q(z,3,3)$

borrar $q(1,1,1)$: $\theta_2 = \{x/1, y/1\}$
$W\theta_2 = p(1,1) \rightarrow \exists z\, q(z,1,1)$

Simplificando las fórmulas anteriores obtenemos:

$$W_S = \exists z\, q(z,3,3) \wedge p(1,1) \rightarrow \exists z\, q(z,1,1)$$

$$D' \text{ viola } W_S \Longrightarrow D' \text{ viola } W$$

3. COMPROBACIÓN DE INTEGRIDAD EN BASES DE DATOS DEDUCTIVAS

3.1. Introducción

Siguiendo las ideas de Nicolas para el caso relacional, los métodos para la comprobación simplificada de la integridad propuestos para BDD se basan en la idea común de evaluar instancias de las restricciones, obtenidas a partir de actualizaciones inducidas por la transacción, y se diferencian entre sí en la estrategia seguida para la instanciación y evaluación de las restricciones. La existencia de reglas deductivas introduce un nuevo problema ya que las actualizaciones inducidas por la transacción no son sólo las explícitamente requeridas por ésta, sino también las inducidas por estas reglas.

Independientemente de la estrategia seguida, los métodos propuestos simplifican la comprobación de la integridad en dos fases: en una primer fase, de generación, se obtiene instancias simplificadas de las restricciones de integridad a partir de las actualizaciones de la transacción; en una segunda fase, de evaluación, estas instancias se evalúan en el nuevo estado según el concepto de satisfacción asumido. De acuerdo a esto, los métodos objeto de estudio pueden clasificarse en dos grupos:

a) Métodos con fase de generación potencial (sin acceso a los hechos).

En estos métodos en la fase de generación se obtiene, a partir de las operaciones de la transacción y de las reglas deductivas, un conjunto de actualizaciones

potenciales con las cuales se instancian y simplifican las restricciones de integridad siguiendo las ideas de Nicolas. (Las actualizaciones potenciales son átomos parcialmente instanciados tales que cualquier actualización real es una instancia de uno de ellos).

En este grupo se incluyen, entre otros, los métodos presentados por Lloyd (1987a), Bry (1988) y Asirelli (1988).

Ejemplo:

Sea:

- D la siguiente Base de Datos Deductiva:

$$\begin{array}{lll} p(x,y) & \leftarrow & t(x), r(y) \\ r(a) & & \\ t(a) & & \\ s(a,a) & & \end{array}$$

- RI = $\{\forall x \forall y \ (s(x,y) \rightarrow p(x,y))\}$ una restricción de integridad
- $T = \{\text{borrar } (t(a))\}$ la transacción

En este ejemplo, el conjunto de actualizaciones potenciales es:

$$\{\neg t(a), \neg p(a,y)\}.$$

Siguiendo las ideas de Nicolas se obtiene la instancia de la restricción:

$$(\forall y(s(a,y) \rightarrow p(a,y))$$

que evaluada en el nuevo estado nos permite concluir que la transacción viola la restricción de integridad.

b) Métodos sin fase de generación potencial (con acceso a los hechos).

En estos métodos, en la fase de generación se accede a la base de datos explícita.

En este grupo se incluyen, entre otros, los métodos presentados en Decker (1986), Sadri (1987), Olive (1991) y Das (1989).

Algunos de estos métodos representan las restricciones de integridad en forma negada, para ello cada restricción W_i es sustituida por la fórmula $\leftarrow inc_i$, donde el predicado de inconsistencia inc_i se define por la regla $inc_i \leftarrow \neg W_i$ que debe ser

incluida en la base de datos. La comprobación de la integridad en los métodos que utilizan esta representación se reduce a determinar las inserciones de átomos de inconsistencia generadas por la transacción.

Ejemplo:

Sea

- D la siguiente Base de Datos Deductiva:

$$\begin{array}{lll} p(x,y) & \leftarrow & t(x), r(y) \\ r(a) & & \\ t(a) & & \\ s(a,a) & & \\ inc & \leftarrow & s(x,y),\ \neg p(x,y) \end{array}$$

- RI = $\{\forall x \forall y (s(x,y) \rightarrow p(x,y))\}$ una restricción de integridad cuya forma negada es: $\leftarrow inc$ donde el predicado inc viene definido por la última regla deductiva.
- $T = \{$ borrar $(t(a))\}$ la transacción.

Para determinar si se ha insertado algún átomo de inconsistencia se pueden derivar actualizaciones inducidas a partir de la operación de la transacción:

$$\begin{array}{ll} \neg t(a) & \\ \quad | & p(x) \leftarrow r(x), t(x) \\ \quad | & \alpha = x/a \\ \neg p(a) & \\ \quad | & inc \leftarrow s(x,y),\ \neg p(x) \\ \quad | & \alpha = x/a \\ \quad | & \beta = y/b \\ \quad | & \\ inc & \end{array}$$

En este ejemplo, la transacción viola la integridad de la Base de Datos al provocar la inserción de un átomo de inconsistencia.

A continuación se presentan los métodos más importantes de cada grupo indicando para cada uno de ellos:

- concepto de satisfacción
- representación y propiedades de las restricciones

- tipo y propiedades de la base de datos
- características del método.

3.2. Concepto de satisfacción

En base de datos deductivas existen tres definiciones del concepto de satisfacción. Sea D un estado de base de datos y W una restricción de integridad:

a) punto de vista de la *demostración:*

$$\boldsymbol{D \text{ satisface } W \text{ si y sólo si } \mathrm{comp}(D) \models W}$$

b) punto de vista de la *consistencia:*

$$\boldsymbol{D \text{ satisface } W \text{ si y sólo si } \mathrm{comp}(D) \cup W \text{ es consistente}}$$

c) punto de vista *canónico:*

D satisface W si y sólo si W es cierta en el modelo estándar de D

donde comp(D) es la complección de D definida en Clark (1978) y el modelo estándar de D es el modelo minimal definido en Apt(1988).

3.3. Métodos con fase de generación potencial

3.3.1. *Método de Lloyd, Sonemberg y Topor (Lloyd (1987a))*

El método de Lloyd *et al.* utiliza como concepto de satisfacción, el punto de vista de la demostración: D satisface W si y sólo si $\mathrm{comp}(D) \models W$.

Este método trabaja con dos conjuntos de actualizaciones potenciales (inserciones y borrados) que pueden ser calculados sin acceder a la base de datos explícita.

Actualizaciones potenciales

Sea T una transacción y D y D' estados consecutivos relacionados por T tales que $D \subseteq D'$ entonces, se define $\mathbf{pos}_{D,D'}$ (conjunto de inserciones potenciales)

y $\mathbf{neg}_{D,D'}$ (conjunto de borrados potenciales) inductivamente de la forma siguiente:

$\text{pos}^0_{D,D'} = \{A: A \leftarrow W \in D' \backslash D\}$ (inserciones potenciales explícitas)

$\text{pos}^{n+1}_{D,D'} = \{A\theta: A \leftarrow W \in D, B$ ocurre positivamente en $W, C \in \text{pos}^n_{D,D'}$ y $\theta = \text{mgu}(B, C)\}$

$\cup$

$\{A\theta: A \leftarrow W \in D, B$ ocurre negativamente en $W, C \in \text{neg}^n_{D,D'}$ y $\theta = \text{mgu}(B, C)\}$

(inserciones potenciales inducidas)

$\text{neg}^0_{D,D'} = \{\ \}$ (borrados potenciales explícitos)

$\text{neg}^{n+1}_{D,D'} = \{A\theta: \leftarrow W \in D, B$ ocurre positivamente en $W, C \in \text{neg}^n_{D,D'}$ y $\theta = \text{mgu}(B, C)\}$

$\cup$

$\{A\theta: A \leftarrow W \in D, B$ ocurre negativamente en $W, C \in \text{pos}^n_{D,D'}$ y $\theta = \text{mgu}(B, C)\}$

(borrados potenciales inducidos)

$\text{pos}_{D,D'} = \bigcup_{n>0} \text{pos}^n_{D,D'}$

$\text{neg}_{D,D'} = \bigcup_{n>0} \text{neg}^n_{D,D'}$

Según la anterior definición, la obtención de los conjuntos $\text{pos}_{D,D'}$ y $\text{neg}_{D,D'}$ podría significar el cálculo de infinitos conjuntos $\text{pos}^n_{D,D'}$ y $\text{neg}^n_{D,D'}$. En la práctica, el cálculo puede realizarse en un número finito de pasos si se utiliza alguna regla de parada del tipo siguiente: en lugar de calcular los conjuntos $\text{pos}^n_{D,D'}$ y $\text{neg}^n_{D,D'}$ calcular los conjuntos P^n y N^n definidos de la siguiente forma:

$P^n(N^n) = \{A: A \in \text{pos}^n_{D,D'}(\text{neg}^n_{D,D'})$ y no existe $A' \in \text{pos}^k_{D,D'}(\text{neg}^k_{D,D'})(0 \leq k \leq n)$ tal que A es una instancia de $A'\}$

La computación finaliza cuando para un valor de n, los conjuntos P^n y N^n son vacíos.

El conjunto $\text{pos}_{D,D'}$ (resp. $\text{neg}_{D,D'}$) se caracteriza porque cualquier inserción real (resp. borrado real) es una instancia de alguno de sus elementos.

TEOREMA DE SIMPLIFICACIÓN

Sea:

- (R, RI) el esquema de una base de datos deductiva, donde:

  R es un lenguaje relacional tipado y
  $\text{RI} = \{W: \forall x_1 x_2 \ldots x_n W'\}$ el conjunto de restricciones de integridad en forma prenexa normal.

- T una transacción que no modifica el lenguaje R y que no es contradictoria (no exige la inserción y el borrado del mismo hecho o regla):

$$D - T_{\text{del}} \rightarrow D'' - T_{\text{ins}} \rightarrow D'$$

T_{del}: borrados de T

T_{ins}: inserciones de T.

- D y D' son estratificadas.

- $\Theta = \{\theta : \theta$ es la restricción a $x_1, x_2, \ldots, x_n$ de un mgu de un átomo que ocurre negativamente en W y un átomo de $\text{pos}_{D'',D'}$ o de un átomo que ocurre positivamente en W y un átomo de $\text{neg}_{D'',D'}\}$.

- $\Psi = \{\varphi : \varphi$ es la restricción a $x_1, x_2, \ldots, x_n$ de un mgu de un átomo que ocurre positivamente en W y un átomo de $\text{pos}_{D'',D}$ o de un átomo que ocurre negativamente en W y un átomo de $\text{neg}_{D'',D}\}$.

Se cumple:

a) D' satisface W si y sólo si D' satisface $\forall(W'\phi)$ para todo $\phi \in \Psi \cup \Theta$.

b) Si $D' \cup \{\leftarrow \forall(W'\phi)\}$ tiene una refutación-SLDNF para todo $\phi \in \Psi \cup \Theta$ entonces D' satisface W.

c) Si $D' \cup \{\leftarrow \forall(W'\phi)\}$ tiene un árbol-SLDNF fallado finitamente para algún $\phi \in \Psi \cup \Theta$ entonces D' viola W.

Si $\Psi \cup \Theta$ es el conjunto vacío W no se ve afectada por la transacción.

Si $\epsilon \in \Psi \cup \Theta$ W no admite simplificación.

3.3.2. *Método de Bry, Decker y Manthey (Bry 1988)*

El método de Bry *et al.* utiliza como concepto de satisfacción, el punto de vista canónico: D satisface W si y sólo si W es cierto en el modelo estándar de D.

Este método considera sólo operaciones simples de hechos: inserciones de hechos no explícitos en D y borrados de hechos explícitos en D. En Bry (1987) esta propuesta se extiende considerando transacciones generales.

Sea L un literal y U un literal base que representa una operación (un literal positivo representa la inserción de un hecho y un literal negativo el borrado) entonces, se definen los siguientes conceptos:

Actualizaciones potenciales

Un átomo A (resp. $\neg A$) no necesariamente base *depende directamente de* L si y sólo si:

- existe una regla deductiva, $A' \leftarrow B$ tal que B contiene un literal L' unificable con L (resp. con el complementario de L)
- $A = A'\theta$, donde $\theta = \text{mgu}(L, L')$ (resp. mgu (complementario de L, L')).

A *depende* de L si y sólo si A depende directamente de L o de un literal que depende de L. Todo literal que depende de U es una *actualización potencial* inducida por U.

El concepto de actualización potencial definido de esta forma coincide con el de Lloyd, aunque Lloyd diferencia dos conjuntos de átomos *pos* (inserciones potenciales) y *neg* (borrados potenciales) en lugar del conjunto único de literales (actualizaciones potenciales) del método de Bry.

Actualizaciones reales

Un átomo base A (resp. $\neg A$) *es inducido directamente por* L sobre D' si y sólo si:

- existe una regla deductiva, $A' \leftarrow B$ tal que B contiene un literal L' unificable con L (resp. con el complementario de L) con mgu θ.
- $A = (A'\theta)\varphi$, donde φ es una respuesta obtenida al evaluar $(B/L')\theta$ en $D'(B/L'$ representa B sin L' o cierto si $B = L')$
- A (resp. $\neg A$) se evalúa a falso (resp. cierto) en D (resp. en D')

Un literal es *inducido* por L sobre D' si y sólo si es directamente inducido por L sobre D' o por un literal inducido por L sobre D'. Todo literal inducido por U sobre D' es una *actualización real* inducida por U.

Al igual que en el método de Lloyd, toda actualización real es una instancia de una actualización potencial.

TEOREMA DE SIMPLIFICACIÓN

Sea:

- $(R$, RI) el esquema de una base de datos deductiva:

  R es un lenguaje relacional y

RI el conjunto de restricciones de integridad, fórmulas de cuantificadores restringidos (independiente del dominio), en forma normalizada. En una fórmula de cuantificadores restringidos las subfórmulas cuantificadas tienen la forma:

$$\exists x_1, x_2, \ldots, x_n \ (A_1 \wedge \cdots \wedge A_m \wedge Q)$$
$$\forall x_1, x_2, \ldots, x_n \ (\neg A_1 \vee \cdots \vee \neg A_m \vee Q)$$

donde cada $x_i (1 \leq 1 \leq n)$ aparece en algún $A_j (1 \leq i \leq m)$ y Q es una fórmula bien formada de cuantificadores restringidos.

La forma normalizada utilizada en el método se obtiene:

- reduciendo al máximo el ámbito de los cuantificadores
- expresando las implicaciones y equivalencias con las conectivas lógicas $\wedge, \vee, \neg$
- llevando las negaciones sobre los átomos
- distribuyendo las disyunciones respecto a las conjunciones.

- $D = \{A: A$ es un átomo base$\} \cup \{A \leftarrow W: W$ es una conjunción de literales$\}$.
- U un literal base que representa la operación.
- D, D' son estratificadas.
- *delta* es un meta-predicado, tal que para todo literal totalmente instanciado L, delta(U, L) es cierto si y sólo si L es cierto en D' y falso en D.
- *new* un meta-predicado tal que new(U, F) es cierto si y sólo si F es cierto en D'.
- RA $= \{\forall$ delta$(U, L\theta) \rightarrow$ new$(U, W_S)\}$ el conjunto de restricciones auxiliares asociadas a W que se obtiene aplicando a la restricción el método de simplificación de Nicolas a partir del conjunto de actualizaciones potenciales inducidas por U donde:
  - L es una actualización potencial inducida por U o bien por U.
  - θ es la restricción de un mgu entre un literal L' de W y el complementario de L, a las variables de L' cuantificadas universalmente no precedidas de un cuantificador existencial y a las variables de L.
  - W_S es la instancia simplificada de W obtenida a partir de $W\theta$ aplicando las siguientes reglas de simplificación
    i) eliminando los cuantificadores sobre las variables instanciadas por θ

ii) reemplazando cada literal de $W\theta$ unificable con el complementario de $L\theta$ por falso y aplicando las reglas de absorción.

Se cumple:

D' satisface W si y sólo si las restricciones auxiliares asociadas a W se satisfacen en D'.

3.4. Métodos sin fase de generación potencial

3.4.1. *Método de Decker (Decker (1986))*

El método de Decker utiliza como concepto de satisfacción el punto de vista de la demostración: D satisface W si y sólo si $\text{comp}(D) \models W$.

Este método obtiene instancias simplificadas de las restricciones de integridad a partir de las actualizaciones reales inducidas por la transacción.

Actualizaciones reales

Los conjuntos de actualizaciones reales asociados a una cláusula C de la forma $H \leftarrow B$ se definen de la forma:

$$D^C = \{H\theta : H\theta \text{ es base, } \text{comp}(D') \models B\theta \text{ y } \text{comp}(D) \not\models H\theta\}$$
$$D_C = \{H\theta : H\theta \text{ es base, } \text{comp}(D) \models B\theta \text{ y } \text{comp}(D') \not\models H\theta\}.$$

ALGORITMO DE SIMPLIFICACIÓN

Sea:

- (R, RI) el esquema de una base de datos deductiva, donde:

  R es un lenguaje relacional y

  RI el conjunto de restricciones de integridad, fórmulas de rango restringido

- $D = \{A : A \text{ es un átomo base}\} \cup \{A \leftarrow W : W \text{ es una conjunción de literales}\}$

- T una transacción formadas por operaciones de la forma:

  - insertar (C)
  - borrar(C)

    donde C es una cláusula.

- D y D' son estratificadas.
- RA = {insertar $\underline{L}$ sólo si $W^{\underline{L}}$: L es un literal positivo que ocurre negativamente en W}

  $\cup$

  {borrar $\underline{L}$ sólo si $W_{\underline{L}}$: L es un literal positivo que ocurre positivamente en W}

  el conjunto de restricciones auxiliares asociado a W, donde:

  - $\underline{L}$ se obtiene a partir de L renombrando las variables de L cuantificadas existencialmente o cuantificadas universalmente precedidas de un cuantificador existencial
  - las formas simplificadas $W^{\underline{L}}(W_{\underline{L}})$ se obtienen reemplazando cada ocurrencia positiva de $\underline{L}$ en W por el valor cierto (falso), cada ocurrencia negativa de $\underline{L}$ en W por el valor falso (cierto) y aplicando las correspondientes reglas de absorción.

La comprobación simplificada de la integridad se realiza practicando el siguiente algoritmo α con argumento insertar(C) (resp. borrar(C)) para cada cláusula C para la cual existe una operación insertar(C) (resp. borrar(C)) en la transacción.

ALGORITMO α

PASO 1: para cada átomo L^* de D^C (resp. D_C) y para cada restricción auxiliar **insertar L sólo si F** (resp. **borrar L sólo si F**) tal que L unifica con L^* con mgu θ evaluar $F\theta$ en D'. Si $F\theta$ se evalúa a falso parar (la transacción viola la integridad).

PASO 2: para cada átomo L^* de D^C (resp. D_C) y para cada cláusula ocurre_positivo (L, R) tal que L unifica con L^* con mgu μ, llamar al algoritmo con argumento insertar($R\mu$) (resp.borrar ($R\mu$)).

PASO 3: para cada átomo L^* de D^C (resp. D_C) y para cada cláusula ocurre_negativo (L, R) tal que L unifica con L^* con mgu φ, llamar al algoritmo con argumento borrar($R\mu$) (resp.insertar($R\varphi$)).

Cuando todas las operaciones de la transacción han sido procesados por el algoritmo sin que se detecte una violación de la integridad, se puede afirmar que la transacción no viola la integridad de D'.

3.4.2. *Método de Sadri y Kowalski (Sadri (1987))*

El método de Sadri *et al.* utiliza como concepto de satisfacción, el punto de vista de la consistencia: D satisface W si y sólo si comp(D)$\cup\{W\}$ es consistente.

El método se caracteriza por utilizar una extensión del procedimiento SLDNF permitiendo entonces aplicar resolución a partir de las actualizaciones de la transacción.

Actualizaciones de la transacción

El conjunto de actualizaciones de la transacción se define como sigue:

$\{A$: insertar_hecho$(A) \in T\}$
$\cup$
$\{A \leftarrow L_1, L_2, \ldots, L_n$: insertar_regla$(A \leftarrow L_1, L_2, \ldots, L_n) \in T\}$
$\cup$
$\{\neg A$: borrar_hecho$(A) \in T$ y $\exists$ árbol-SLDNF fallado finitamente para $D' \cup \{\leftarrow A\}\}$
$\cup$
$\{\neg A\theta$: borrar_regla$(A \leftarrow L_1, L_2, \ldots, L_n) \in T$ y θ es una respuesta computada para $D \cup \{\leftarrow L_1, L_2, \ldots, L_n\}$ y $\exists$ árbol-SLDNF fallado finitamente para $D' \cup \{\leftarrow A\}\}$.

Procedimiento SLDNF* (SLDNF extendido)

Sea:

- $S = D \cup$ RI donde:
  - D es un conjunto de cláusulas de la forma:

    A o bien
    $A \leftarrow L_1, L_2, \ldots, L_n$

  - RI un conjunto de cláusulas de la forma:

    $$\leftarrow L_1, L_2, \ldots, L_n$$

    Si L_i es positivo se denomina condición positiva, si es negativo condición negativa.

- C_0 una cláusula de S o un átomo negado $(\neg A)$ tal que $S \cup \{\leftarrow A\}$ falla finitamente utilizando el SLDNF*.

- R una regla de computación segura (nunca selecciona un literal negativo no base).

Una *derivación* vía R para $S \cup \{C_0\}$ es una secuencia posiblemente infinita $C_0, C_1, C_2 \ldots$ tal que C_i para $i > 0$ es una cláusula, y para todo $i \geq 0$ C_{i+1} se obtiene a partir de C_i de la siguiente forma:

a) Si R selecciona de C_i un literal L que no es una condición negativa de C_i, entonces C_{i+1} es el resolvente sobre L de C_i y alguna cláusula de S.

b) Si R selecciona una condición negativa, $\neg A$, de C_i. Entonces C_{i+1} es C_i eliminando el literal seleccionado, $\neg A$, si existe un árbol-SLDNF* fallado finitamente para $S \cup \{\leftarrow A\}$.

c) Si C_i es $\neg A$ y en S hay una cláusula $B \leftarrow \neg A', C$ de forma que A y A' unifican a través del mgu θ entonces C_{i+1} es $(B \leftarrow C)\theta$.

El SLDNF* es **correcto** en el sentido siguiente:
"Si existe una refutación-SLDNF* para $S \cup \{C_0\}$ entonces comp(D)$\cup$ RI es inconsistente".

TEOREMA DE SIMPLIFICACIÓN

Sea:

- (R, RI) el esquema de una base de datos relacional donde:

  R es un lenguaje relacional y
  RI = $\{\leftarrow \text{inc}_i : \text{inc}_i \leftarrow \neg W_i,\ 1 \leq i \leq n\}$ el conjunto de restricciones de integridad en forma negada (el conjunto de cláusulas resultantes de aplicar el algoritmo de Lloyd (Lloyd (1987b)) a $\{\text{inc}_i \leftarrow \neg W_i : 1 \leq i \leq n\}$ deben ser de rango restringido y forma parte de cualquier estado de la base de datos).

- D = $\{A: A$ es un átomo base$\}$ $\cup$
  $\{A \leftarrow W$: es de rango restringido y W es una conjunción de literales$\}$.

- T una transacción.

- D y D' son estratificadas.

Se cumple:

"Si existe una refutación-SLDNF* para $D'\cup$ RI $\cup\{C_0\}$ para alguna actualización C_0 de la transacción entonces D' viola RI".

3.4.3. *Método de Olivé (Olivé 1991)*

El método de Olivé utiliza como concepto de satisfacción, el punto de vista de la demostración: D satisface W si y sólo si comp(D) $\models W$.

El método se caracteriza por:

- extender el lenguaje de la base de datos con dos tipos de predicados:
  - i) predicados de *transición:* permiten simular la base de datos actualizada
  - ii) predicados de *eventos internos* (inserción y borrado): representan las actualizaciones generadas por la transacción
- ampliar la base de datos con las reglas deductivas que definen dichos predicados
- comprobación directa de las restricciones de integridad utilizando dichas reglas
- tratamiento uniforme para restricciones de integridad estáticas y de transición
- uso del procedimiento SLDNF para la comprobación de la integridad.

Predicados de eventos internos: actualizaciones generadas por T

Para cada predicado P, se introduce un predicado de transición P', que representa el predicado P en la base de datos actualizada D' y dos predicados de eventos internos:

ιP: predicado de evento interno de inserción
δP: predicado de evento interno de borrado

estos predicados representan las actualizaciones (inserciones y borrados) explícitas o inducidas generadas por una transacción T sobre el predicado P. De la definición de actualización real inducida por una transacción podemos afirmar:

$$\forall x_1, x_2, \ldots, x_n(\iota P(x_1, x_2, \ldots, x_n) \leftrightarrow P'(x_1, x_2, \ldots, x_n) \wedge \neg P(x_1, x_2, \ldots, x_n))$$
$$\forall x_1, x_2, \ldots, x_n(\delta P(x_1, x_2, \ldots, x_n) \leftrightarrow P(x_1, x_2, \ldots, x_n) \wedge \neg P'(x_1, x_2, \ldots, x_n))$$

Si P es un predicado base: ιP, δP serán predicados base cuyos hechos serán determinados directamente por las actualizaciones explícitas de la transacción T sobre P.

Si P es un predicado derivado: $\iota P, \delta P$ serán predicados derivados definidos por reglas deductivas que nos permitirán obtener las actualizaciones inducidas por la transacción T sobre P.

Reglas de eventos internos

Para poder determinar las reglas deductivas que definen los predicados ιP y δP para predicados derivados de la base de datos vamos a determinar previamente las reglas que definen el predicado P'.

Como P' representa al predicado P en la base de datos actualizada, se cumplirá:

$$\forall x_1, x_2, \ldots, x_n (P'(x_1, x_2, \ldots, x_n) \leftrightarrow (P(x_1, x_2, \ldots, x_n) \wedge \\ \wedge \neg\delta P(x_1, x_2, \ldots, x_n)) \vee \iota P(x_1, x_2, \ldots, x_n))$$

$$\forall x_1, x_2, \ldots, x_n (\neg P'(x_1, x_2, \ldots, x_n) \leftrightarrow (\neg P(x_1, x_2, \ldots, x_n) \wedge \\ \wedge \neg\iota P(x_1, x_2, \ldots, x_n)) \vee \delta P(x_1, x_2, \ldots, x_n))$$

Si el predicado derivado P viene definido por m reglas deductivas en D:

$$P \leftarrow P_i \quad 1 \leq i \leq m \quad \text{siendo} \quad P_i \leftrightarrow L_1 \wedge L_2 \wedge \cdots \wedge L_n$$

entonces P' estará definido por m reglas deductivas de la forma:

$$P' \leftarrow P'_i \quad 1 \leq i \leq m \quad \text{siendo} \quad P'_i \leftrightarrow L'_1 \wedge L_2 \wedge \cdots \wedge L'_n$$

reemplazando cada L'_j de la siguiente forma:

si $L_j = Q_j(x_1, x_2, \ldots, x_n)$ (literal positivo) entonces:

$$L'_j = (Q_j(x_1, x_2, \ldots, x_n) \wedge \neg\delta Q_j(x_1, x_2, \ldots, x_n)) \vee \iota Q_j(x_1, x_2, \ldots, x_n)$$

si $L_j = \neg Q_j(x_1, x_2, \ldots, x_n)$ (literal negativo) entonces:

$$L'_j = (\neg Q_j(x_1, x_2, \ldots, x_n) \wedge \neg\iota Q_j(x_1, x_2, \ldots, x_n)) \vee \delta Q_j(x_1, x_2, \ldots, x_n)$$

con lo que se obtienen m reglas (cláusulas no normales) que definen P' en términos (conjunción de disyunciones) de predicados de base de datos y predicados de eventos internos. Para obtener un conjunto equivalente de cláusulas normales basta distribuir las conjunciones sobre las disyunciones.

Una vez obtenidas las reglas que definen P', las reglas que definen ιP y δP son:

$$\iota P(x_1, x_2, \ldots, x_n) \leftarrow P'(x_1, x_2, \ldots, x_n) \wedge \neg P(x_1, x_2, \ldots, x_n)$$
$$\delta P(x_1, x_2, \ldots, x_n) \leftarrow P'(x_1, x_2, \ldots, x_n) \wedge \neg P'(x_1, x_2, \ldots, x_n).$$

Las reglas de eventos internos pueden simplificarse después de aplicarles algunas transformaciones como se indica en Olivé (1991).

TEOREMA DE SIMPLIFICACIÓN

Sea:

- (R, RI) el esquema de una base de datos deductiva donde:

  R es un lenguaje relacional y
  $\mathrm{RI} = \{\leftarrow \mathrm{inc}_i : \mathrm{inc}_i \neg W_i,\ 1 \leq i \leq n\}$ el conjunto de restricciones de integridad en forma negada (el conjunto de cláusulas resultantes de aplicar el algoritmo de Lloyd a $\{\mathrm{inc}_i \leftarrow \neg W_i : 1 \leq i \leq n\}$ deben ser de rango restringido y forma parte de cualquier estado de la base de datos).

- $D = \{A : A$ es un átomo base$\}$ $\cup$
  $\{A \leftarrow W$: es de rango restringido y W es una conjunción de literales

- T una transacción.

- D y D' son estratificadas.

Si $A(D)$ es la base de datos resultante de añadir a D las reglas de transición y de eventos internos para cada predicado derivado y predicado de inconsistencia de D entonces se cumple:

a) D' viola la restricción W_i si existe una refutación-SLDNF para $A(D) \cup T \cup \{\leftarrow \iota \mathrm{inc}_i\}$

b) D' satisface la restricción W_i si existe un árbol-SLDNF fallado finitamente para $A(D) \cup T \cup \{\leftarrow \iota \mathrm{inc}_i\}$.

3.4.4. *Método de Das y Williams (Das 1989)*

El método de Das *et al.* utiliza como concepto de satisfacción, el punto de vista de la demostración: D satisface W si y sólo si $\mathrm{comp}(D) \models W$.

En este método, la comprobación de la integridad consiste en buscar un "camino" desde una "actualización" de la transacción T hasta un átomo de inconsistencia.

Actualizaciones de T

Si A es un átomo base y $H \leftarrow B$ es una regla deductiva el conjunto de actualizaciones de T se define:

$$\{A\text{: insertar\_hecho}(A) \in T\} \cup \{\neg A\text{: borrar\_hecho}(A) \in T\}$$
$$\cup$$
$$\{H\theta\text{: insertar\_regla}(H \leftarrow B) \in T \text{ y } \theta \text{ es una respuesta-SLDNF computada para } D' \cup \{\rightarrow B\}\}$$
$$\cup$$
$$\{\neg H\text{: borrar\_regla}(H \leftarrow B) \in T\}$$

Camino

Si D es un estado de la base de datos, un camino se define como:

$$L_0 - R_1 \longrightarrow L_1 - R_2 \longrightarrow \cdots R_n \longrightarrow L_n$$

donde L_0 es el origen del camino, L_n es el destino, n es su longitud y $R_1, R_2, \ldots, R_n$ son cláusulas de D utilizadas para construir el camino de L_0 a L_n. Si L_0 es positivo es base y debe existir una refutación-SLDNF para $D \cup \{\leftarrow L_0\}$.

L_{i+1} se define a partir de L_i de la forma:

1. Si:
   - L_i es positivo
   - L_i unifica con un literal positivo del cuerpo de la cláusula R_i: $H \leftarrow B$ con mgu α
   - G' es el resolvente de $\leftarrow B$ y L_i
   - θ es una respuesta-SLDNF computada para $D \cup \{\leftarrow G'\}$

   entonces L_{i+1} es $H\alpha\theta$

2. Si:
   - L_i es positivo
   - L_i unifica con el complementario de un literal negativo del cuerpo de la cláusula R_i: $H \leftarrow B$ con mgu α
   - $\neg H\alpha$ no es una instancia de algún $L_j (0 \leq j \leq i)$

   entonces L_{i+1} es $\neg H\alpha$

3. Si:
   - L_i es negativo
   - L_i unifica con un literal negativo del cuerpo de la cláusula R_i: $H \leftarrow B$ con mgu α

- θ es una respuesta-SLDNF computada para $D \cup \{G\}$ donde $G = \leftarrow B\alpha$

entonces L_{i+1} es $H\alpha\theta$

4. Si:

   - L_i es negativo
   - L_i unifica con el complementario de un literal L que ocurre en el cuerpo de una cláusula $R_i: H \leftarrow B$ con mgu α
   - $\neg H\alpha$ no es una instancia de algún $L_j (0 \leq j \leq i)$

   entonces L_{i+1} es $\neg H\alpha$

Un camino que finaliza en algún átomo de inconsistencia se denomina **camino de éxito** en caso contrario se denomina **camino de fallo**.

TEOREMA DE SIMPLIFICACIÓN

Sea:

- (R, RI) el esquema de una base de datos deductiva, donde:

  R es un lenguaje relacional y

  $\text{RI} = \{\leftarrow \text{inc}_i : \text{inc}_i \neg W_i,\ 1 \leq i \leq n\}$ el conjunto de restricciones de integridad en forma negada (el conjunto de cláusulas resultantes de aplicar el algoritmo de Lloyd a $\{\text{inc}_i \leftarrow \neg W_i : 1 \leq i \leq n\}$ deben ser de rango restringido y forma parte de cualquier estado de la base de datos).

- $D = \{A : A$ es un átomo base $\cup$
  $\{A \leftarrow W$: es de rango restringido y W es una conjunción de literales$\}$.

- T una transacción.

- D y D' son estratificadas.

Se cumple:

a) si existe un camino de éxito en D' con origen alguna de las actualizaciones de T entonces D' viola RI.

b) si no existe un camino de éxito en D' para ninguna de las actualizaciones de T como origen entonces D' satisface RI.

4. ANÁLISIS DE LOS MÉTODOS

En este apartado se persiguen dos objetivos:

- resumir los requisitos exigidos por los distintos métodos referentes al tipo y propiedades sintácticas de la base de datos así como a la representación y propiedades sintácticas de las restricciones.
- analizar la estrategia de cada método.

Tipo de base de datos

El método de Lloyd trabaja con bases de datos "generales" es decir, con cláusulas de la forma: $A \leftarrow W$, donde A es un átomo y W una fórmula bien formada cualquiera.

Los restantes métodos se definen para bases de datos "normales" con reglas de la forma: $A \leftarrow L_1, L_2, \ldots, L_n$, donde A es un átomo y L_i un literal. Este requisito no quita generalidad a un método ya que cualquier base de datos general puede transformarse en una base de datos normal siguiendo el algoritmo de Lloyd sin embargo, en este caso habrá que tener en cuenta que al aplicar el algoritmo se pueden perder ciertas propiedades sintácticas de la base de datos (independencia del dominio).

Propiedades sintácticas de la base de datos y la restricción

i) Todos los métodos exigen que la base de datos D sea estratificada, asegurando de esta forma que comp(D) es consistente y que D tiene un único modelo minimal (el modelo estándar).

ii) Los métodos que utilizan el SLDNF como mecanismo procedural para la comprobación de la integridad, exigen a la base de datos y a la restricción propiedades sintácticas que aseguran que en las computaciones SLDNF realizadas no aparece el problema del "tropiezo" (en algún punto de la derivación se obtiene un objetivo que sólo contiene literales negativos no base), Lloyd (1987b).

En el método de Lloyd, el uso de un lenguaje tipado asegura que la forma normal sin tipos de una base de datos y un requerimiento es permitida (Lloyd (1987b)) eliminando por tanto el problema del "tropiezo". En los restantes métodos se exige a cada cláusula normal de la base de datos la propiedad de rango restringido (o alguna propiedad sintáctica que implique ésta). Si las restricciones se representan en forma negada se les exige la

propiedad de rango restringido y si se representa en forma general se les exige alguna propiedad sintáctica (cuantificadores restringidos) que asegure que las cláusulas resultantes al aplicar el algoritmo de Lloyd son de rango restringido.

iii) La simplificación de la comprobación de la integridad basada en consideraciones sintácticas: "instanciación de las restricciones a partir de las actualizaciones inducidas por la transacción a través de las reglas deductivas", sólo es correcta cuando:

- siendo el concepto de satisfacción el de la demostración, la base de datos junto a la restricción cumplen la propiedad de independencia del dominio. Esta propiedad permite asegurar que el conjunto de respuestas correctas para comp(D)$\cup\{W\}$ es independiente del lenguaje.
- siendo el concepto de satisfacción el canónico, el modelo estándar de comp(D) es independiente del lenguaje.

Estrategia de los métodos

El análisis de las características de los métodos nos permite realizar las siguientes consideraciones:

i) la existencia de una fase de generación potencial es interesante porque permite, sin acceder a los hechos eliminar del proceso de comprobación:

- restricciones no relevantes para la transacción
- actualizaciones no relevantes para la integridad.

ii) la instanciación de las restricciones a partir de actualizaciones potenciales, genera un conjunto de restricciones menos instanciadas que en el caso de trabajar con actualizaciones reales, y por lo tanto en general más costosas de comprobar.

iii) aunque en la fase de evaluación, las restricciones simplificadas obtenidas en la fase de generación potencial se evalúen sólo para aquellas instancias correspondientes a instancias de las actualizaciones potenciales que coinciden con una actualización real (Bry (1988)), el proceso puede ser muy costoso ya que los métodos no hacen uso de la información disponible en la fase generación referente a caminos de derivación para la obtención de estas actualizaciones reales.

iv) los métodos con fase de generación potencial pueden optimizarse intercalando la fase de generación y la fase de evaluación.

v) en los métodos sin fase de generación potencial, la selección como origen de la derivación de actualizaciones que inducen a su vez actualizaciones no relevantes para la integridad puede elevar el coste de la comprobación.

5. BIBLIOGRAFÍA

[1] **Apt, K., Blait, H.A.** & **Walker, A.** (1988). "Towards a Theory of Declarative Knowledge." *Foundations of Deductive Databases and Logic Programming.* Morgan Kaufman.

[2] **Asirelli, P., Inverardi, P.** & **Mustaro, A.** (1988). "Improving Integrity Constraint Checking in Deductive Databases". *Proc. 2nd International Conference on Database Theory.*

[3] **Bry, F.** & **Decker, H.** (1987). "Préserver l'Intégrité d'une Base de Données Déductive: une Méthode et son Implémentation". *4emes Journées de Bases de Donées Avancées.* Bénodet (France).

[4] **Bry, F., Decker, H.** & **Manthey, R.** (1988). "A Uniform Approach to Constraint Satisfaction and Constraint Satisfiability in Deductive Databases". *Proc. 1st Conf. Extending Database Technology.*

[5] **Clark. P** (1978). "Negation as Failure". *Logic and Databases.* Plenum.

[6] **Das, S.K.** & **Williams, M.H.** (1989). "A Path Finding Method for Constraint Checking in Deductive Databases". *Data & Knowledge Engineering,* **4**. North Holland.

[7] **Decker, H.** (1986). "Integrity Enforcement in Deductive Databases". *Proc. 1st Int. Conf. on Expert Database Systems.*

[8] **Lloyd, J.W., Soneberg, E.A.** & **Topor, R.W.** (1987a). "Integrity Constraint Checking in Stratified Databases". *Journal of Logic Programming,* **4**, 331–343.

[9] **Lloyd, J.W.** (1987b). "Foundations of Logic Programming." Springer-Verlag.

[10] **Nicolas, J.M.** (1982). "Logic for Improving Integrity Checking in Relational Databases". *Acta Informatica,* **18**.

[11] **Olivé, A.** (1991). "Integrity Checking in Deductive Databases". *Proc. del 17th International Conference on Very-Large Databases.*

[12] **Reiter, R.** (1984). "Towards a logical reconstruction of Relational Database Theory." *On Conceptual Modelling.* Springer-Verlag.

[13] **Sadri, F.** & **Kowalski, R.** (1987). "A Theorem-proving Approach to Database Integrity". *Proc. del Workshop on Foundations of Deductive Databases and Logic Programming.*

[14] **Topor, R.W.** (1987). "Domain Independent Formulas and Databases". *Theoretical Computer Science,* **52**.

ENGLISH SUMMARY:

METHODS FOR INTEGRITY CHECKING IN DEDUCTIVE DATABASES

Laura Mota Herranz and Matilde Celma Giménez

Deductive Databases (DDB) are an extension of Relational Databases (RDB) since they include rules that allow us to derive new information from the information explicitly stored.

A *DDB scheme* consists of:

- a set of relation schemes of the form $R(A_1: D_1, \ldots, A_n: D_n)$ where R is the name of the relation, $A_1, \ldots, A_n$ are attribute identifiers and $D_1, \ldots, D_n$ are the names of the domains associated with the attribute and
- a set of Integrity Constraints (IC).

A *database state* is a set of facts (tuples of basic relations) plus a set of deductive rules.

An *integrity constraint* is a statement that a database must satisfy at any time in order to faithfully describe the real world represented by the database. The evolution through time of a database can be described by a sequence of states where transitions from one state to the next are accomplished by database transactions (set of insertions and/or deletions of facts and/or rules). According to this evolution scheme, static and dynamic constraints can be distinguished; the former restrict the validity of each state on its own, while the latter relate the validity of a sequence of consecutive states. In the paper, we only deal with static constraints.

In previous database scheme, we can distinguish two types of relations: *basic* relations whose tuples are explicity stored and *derived* ones defined by deductive rules. A RDB is a special case of DDB without derived relations.

From the point of view of logic, a DDB can be formalized as follows (Reiter (1984), Lloyd (1987b)):

- the DDB scheme is represented by a pair (R, IC) where:

  R is a relational language defined from the relation schemes

  IC is the set of integrity constraints (closed well formed formulas (wff) of R)

- a database state D is represented by a first order theory defined over R:

$$D = \{A: A \text{ is a ground atom (fact)}\}$$

$$\cup$$

$$\{A \leftarrow W: A \text{ is an atom and } W \text{ is a wff}\}.$$

To ensure that each integrity constraint is satisfied in the new state, all of them must be checked after every transaction. However, this checking can be very costly if they are evaluated as queries, particularly in large databases. In spite of these difficulties, it is possible to reduce the amount of computation if advantage is taken of the fact that, before the transaction was made, the database was known to satisfy its integrity constraints and hence, any violation of them is due to an update induced by the transaction. Thus, every practical integrity checking method simplifies this process avoiding checking some instances of the constraints that were satisfied before the transaction (in the old state) and have not been affected by this one. This simplification will be only correct if the wff that represents the integrity constraints are domain-independent. Intuitively speaking, a wff is domain-independent if its evaluation only depends on the extensions of the predicated that appear in it (Topor (1987)).

In this paper entitled **Methods for Integrity Checking in Deductive Databases** we present the Nicolas Method for simplifying integrity checking in relational databases (Nicolas (1982)), following this proposal, many methods for simplified integrity checking in deductive databases have been proposed in the last ten years; all of them are based on the idea of evaluating simplified instances of the integrity constraints generated by the updates induced by the transaction. They differ mainly in the way these induced updates are determined and in the strategy used to instantiate and simplify the constraints. From all the methods for simplifying integrity checking in deductive databases we present the methods of Decker (Decker (1986)), Lloyd & Sonnenberg & Topor (Lloyd (1987a)), Sadri

& Kowalski (Sadri (1987)), Bry & Decker & Manthey (Bry (1988)), Das & Williams (Das (1989)) and Olivé (Olivé (1991)).

Independently of the particular strategy followed by the methods, we state that all of them simplify the integrity in two phases: in a first phase, that we will call the *generation phase,* simplified instances of the constraints are obtained using the updates induced by the transaction, in a second phase, the *evaluation phase,* these instances are evaluated in the new state. These two phases can appear separates or interleaves in the different methods.

For each one of all these methods we present:

- point of view of constraints satisfaction assumed by the method
- representation and properties of the constraints
- type and properties of the database
- features of the method