UN ALGORITM BRUT DE CĂUTARE A CHEILOR NESINTETIZATE

Vitalie COTELEA
Catedra Cibernetică şi Informatică Economică, ASEM

This paper focuses on a problem that occurs in the process of database design, that is keys generation. The problem of keys determination is reduced to a problem of searching the unsynthesized keys. Here are defined the conditions of existence of unsynthesized keys and based on them a polynomial algorithm for their generation is presented.

Introducere

Prezentul articol este consacrat unei probleme ce apare în procesul de proiectare a bazei de date – generarea cheilor, şi unei probleme omoloage – determinarea atributelor primare.

Aceste probleme au fost examinate de mai mulţi specialişti în domeniu. C. Delobel şi R. Casey [1], şi R. Fadoum şi J. Forsithe [2] au propus doi algoritmi diferiţi de calculare a cheilor posibile în bazele de date relaţionale. Analiza algoritmilor propuşi de ei nu a fost adusă. Problema generării cheilor posibile a fost cercetată şi de M. c. Fernandes [3]. Rezolvarea problemei, folosind funcţiile booleene, a fost propusă de E. A. Nekliudova şi M. S. Ţalenko [4]. Dar, în [5] este descris un algoritm de determinare a cheilor, a cărui complexitatea creşte polinomial în raport cu cardinalităţile mulţimii de dependenţe funcţionale, mulţimii de atribute şi mulţimii de chei posibile.

Însă, aşa cum este arătat în [6], schema relaţională poate avea $max(\mid F \mid, \sqrt{\mid R_i \mid})!$ chei posibile, unde $\mid F \mid$ reprezintă numărul de dependenţe funcţionale, iar $\mid R_i \mid$ numărul de atribute ce constituie schema. Prin urmare, această problemă face parte din clasa problemelor NP-complete [7], şi pentru soluţionarea acesteia, se vede că nu poate fi găsit un algoritm de natură polinomială.

Practica proiectării bazelor de date impune însă cerinţele sale. În primul rând, unii algoritmi de proiectare a schemelor apelează la problema de generare a cheilor. De exemplu, algoritmul expus în [1], la primul pas al acestuia, se calculează toate cheile schemei universale date. Algoritmul de sinteză propus în [4], de asemenea, apelează la problema de construire a cheilor posibile şi de determinare a mulţimii de atribute primare. În al doilea rând, definiţiile formelor normale includ noţiunile de cheie-candidat, de atribute primar şi neprimar. De aceea, automatizarea procesului de analiză a schemelor şi de determinare a gradului de normalizare necesită soluţionarea problemelor de generare a cheilor şi de calculare a atributelor primare [8].

Cu toate acestea, pot fi trasate căile de îmbunătăţire a algoritmilor de soluţionare a acestor probleme, pentru a putea fi aplicate în calitate de instrumente de proiectare a schemelor.

Deoarece cheile se formează din cheile deja existente cu ajutorul dependenţelor funcţionale, pot fi formulate următoarele întrebări:

1. Pentru construirea cheilor noi sunt necesare toate dependenţele?
2. Toate cheile sunt necesare pentru generarea celorlalte chei?
3. Poate fi determinată o ordine de aplicare a dependenţelor funcţionale în construirea cheilor?
4. Toate cheile sunt necesare pentru construirea mulţimii de atribute primare?

Întrebările 1, 2 şi 4 trebuie verificate la fiecare pas de generare a următoarei chei.

Soluţionarea acestor sarcini va permite esenţial să se reducă dimensiunile problemei de generare a cheilor. În prezentul articol sunt aduse unele fundamente teoretice ce servesc drept bază pentru un algoritm brut care, ulterior, va putea fi ameliorat realizând sarcinile propuse.

Noţiuni preliminare

Fie $F = F_1 \cup ... \cup F_n$ o mulţime minimală şi redusă de dependenţe funcţionale, divizată în clase de echivalenţă. Fie se selectează clasa de echivalenţă F_i. Atunci, mulţimea $PS(F_i)$ reprezintă mulţimea de chei ale schemei $Sch_i(R_i, F)$, numite sintetizate. Dar schema $Sch_i(R_i, F)$ poate avea şi chei nesintetizate.

De exemplu, fiind dată mulţimea de dependenţe funcţionale $F = \{AB \to C,\ C \to B\}$, aceasta se împarte în două clase de echivalenţă $F_1 = \{C \to B\}$ şi $F_2 = \{AB \to C\}$, iar schema bazei de date va arăta după cum urmează: $SchBD = \{Sch_1(BC, F),\ Sch_2(ABC, F)\}$. Mulţimea AB de atribute este o cheie sintetizată a schemei Sch_2, iar mulţimea AC este cheie nesintetizată a acestei scheme.

Astfel, problema generării cheilor aici se tratează ca problemă a căutării cheilor nesintetizate.

Pentru expunerea materiei, este nevoie de câteva rezultate preliminare.

Lema 1 ([9], lema 5.7). Fie $F = F_1 \cup ... \cup F_n$ o mulţime neredundantă de dependenţe funcţionale împărţită în clase de echivalenţă. Să presupunem că $X \in PS(F_i)$ şi orice Y, unde $X \leftrightarrow Y$ în raport cu F. Atunci, există o parte stângă Z în $PS(F_i)$, astfel că $Y \to Z \in (F - F_i)^+$.

Lema 2 ([10], lema 3.1). Dacă $X \subseteq Y$ şi secvenţele $< X_0, X_1, ..., X_n >$, $< Y_0, Y_1, ..., Y_m >$ sunt derivaţii maximale ale mulţimilor X şi Y, corespunzător, în raport cu F, atunci pentru orice X_i există o mulţime Y_j, încât $X_i \subseteq Y_j$ şi $j \leq i$.

Lema 3 ([10], lema 3.2). Dacă $< X_0, X_1, ..., X_n >$ constituie derivarea maximală a mulţimii X în raport cu mulţimea de dependenţe funcţionale F, atunci $X \to X_i \in F^+$, $i = \overline{0n}$.

Consecinţa 1 ([10], consecinţa 3.1). $X \to Y \in F^+$ atunci şi numai atunci, când există derivarea dependenţei $X \to Y$ în raport cu F.

Observaţia 1. Dacă $X \to Y \in F^+$ şi dependenţa $V \to W$ din F este folosită în construirea derivării dependenţei $X \to Y$ în raport cu F, atunci $X \to V \in F^+$

Condiţii de existenţă a cheilor nesintetizate

În continuare, vor fi examinate pentru o schemă dată condiţiile de existenţă a cheilor nesintetizate.

Poate fi formulată următoarea teoremă, care arată legătura dintre chei şi dependenţe într-o schemă $Sch_i(R_i, F)$.

Teorema 1. Dacă Y este o cheie a schemei $Sch_i(R_i, F)$, atunci pentru orice atribut A din Y există în mulţimea F o dependenţă funcţională $V \to W$, astfel că $A \in V$.

Demonstraţie. Dacă $Y \in PS(F_i)$, adică dacă Y este o cheie sintetizată, teorema e demonstrată. Fie că Y este o cheie nesintetizată a schemei $Sch_i(R_i, F)$ şi fie că există un atribut A în Y, astfel că $A \notin V$ pentru orice dependenţă funcţională $V \to W \in F$. Conform lemei 1, în $PS(F_i)$ se va găsi o cheie sintetizată X, pentru care are loc $Y \to X \in (F - F_i)^+$. Atunci, în virtutea lemei 2, există derivarea $H =< Y_0, ..., Y_n >$ pentru dependenţa $Y \to X$ în raport cu $F - F_i$. Dar deoarece $A \notin X$ şi părţile stângi ale tuturor dependenţelor utilizate în construirea derivării H, de asemenea, nu conţin atributul A, atunci, folosind lema 3, are loc $(Y_0 - \{A\}) \to Y \in (F - F_i)^+$, pentru $i = \overline{1, n}$, şi, prin urmare, $(Y_0 - \{A\}) \to X \in (F - F_i)^+$. Dar acest fapt contrazice supoziţia că mulţimea de atribute Y este cheie.

Consecinţa 2. Dacă $X \to Y \in F^+$ şi X este determinantă pentru Y în raport cu F, atunci, pentru orice atribut A din $X - Y$ există în mulţimea F o dependenţă $V \to W$, folosită în derivarea dependenţei $X \to Y$ în raport cu F, astfel că $A \in V$.

Consecinţa 3. Fie $Sch_i(Ri, F)$ o schemă relaţională şi fie $A \in R_i$. Dacă $A \notin V$ pentru orice dependenţă $V \to W$ din F, atunci atributul A este neprimar în $Sch_i(Ri, F)$.

Următoarea teoremă serveşte drept bază pentru algoritmul de generare a cheilor şi constituie o extindere a lemei 4 din [5] pentru o schemă a bazei de date. În [5] lema respectivă cuprinde o singură schemă relaţională.

Teorema 2. Dacă K_i este o mulţime de chei posibile ale schemei $Sch_i(Ri, F)$ şi $PS(F_i) \subseteq K_i$, atunci există o cheie suplimentară pentru schema $Sch_i(Ri, F)$, care nu aparţine mulţimii de chei K_i atunci şi numai atunci, dacă pentru o cheie X, unde $X \in K_i$ în mulţimea de dependenţe $F - F_i$ există o dependenţă $V \to W$ şi o mulţime $S \subseteq R_i$, astfel că S este determinantă a lui V, în raport cu $F - F_i$ şi mulţimea $S \cup (X - S^+_{F-F_i})$ nu conţine chei din K_i.

Demonstraţie. Necesitate. Fie Y este o cheie nesintetizată pentru schema $Sch_i(Ri, F)$ şi fie $Y \notin K_i$. Conform lemei 1, în $PS(F_i)$, există o mulţime Z, astfel încât $Y \to Z \in (F - F_i)^+$. Dar atunci, în virtutea consecinţei 1, există derivarea $H =< Y_0, ..., Y_n >$ pentru dependenţa $Y \to Z$ în raport cu $F - F_i$.

Fie Y_j este ultimul termen al derivaţiei care nu conţine vreo cheie din K_i şi se presupune fără limitarea generalităţii că termenul Y_{j+1} se construieşte aplicând doar o singură dependenţă $V \to W$ din $F - F_i$, adică $Y_{j+1} = Y_j \cup W$. Deoarece Y_{j+1} conţine, deja, o cheie X din K_i, iar Y_j nu o conţine, reiese că $(X - W) \subseteq Y_j$. Întrucât $V \subseteq Y_j$, atunci $V \cup (X - W) \subseteq Y_j$. Dar $W \subseteq V^+_{F-F_i}$. Astfel, $V \cup (X - V^+_{F-F_i}) \subseteq Y_j$.

Ţinând cont de observaţia 1, are loc $Y \to V \in (F - F_i)^+$. Fie că submulţimea S a mulţimii Y este determinantă pentru V în raport cu $F - F_i$. Deoarece $S \subseteq Y_j$ şi $V^+_{F-F_i} \subseteq S^+_{F-F_i}$, atunci $S \cup (X - S^+_{F-F_i}) \subseteq Y_j$. Mulţimea Y_j se ştie că nu conţine chei din K_i, prin urmare şi mulţimea $S \cup (X - S^+_{F-F_i})$, de asemenea, nu conţine chei din K_i.

Suficienţă. Fie $X \in K_i$, $V \to W \in (F - F_i)$, $S \subseteq R_i$, S este determinantă pentru V în raport cu $F - F_i$ şi mulţimea $S \cup (X - S^+_{F-F_i})$ nu conţine chei din K_i. Deoarece mulţimea $S \cup X \cup S^+_{F-F_i}$ conţine cheia X, aceasta este supercheie. Din faptul că $S \to S^+_{F-F_i} \in (F - F_i)^+$ reiese că $(S \cup (X - S^+_{F-F_i})) \to (S \cup X \cup S^+_{F-F_i}) \in (F - F_i)^+$, adică mulţimea $S \cup (X - S^+_{F-F_i})$, de asemenea, conţine o cheie, care, conform ipotezei, nu este în K_i. Din faptul că $(S \cup (X - S^+_{F-F_i})) \subseteq R_i$, teorema e demonstrată.

Algoritmul de generare a cheilor nesintetizate

În continuare va fi descris algoritmul. Întrucât orice cheie a schemei $Sch_i(R_i, F)$ este determinantă pentru R_i în raport cu F, se va considera mai întâi algoritmul de găsire a unei determinante.

Algoritmul de calculare a determinantei

Date de intrare:

F este o mulţime de dependenţe funcţionale definite pe mulţimea de atribute R.

X este o submulţime a mulţimii R.

Date de ieşire:

K este o determinantă a mulţimii de atribute X în raport cu mulţimea de dependenţe funcţionale F.

$DETERMINANTA(F, X, K)$
$K := X;$
for each $A \in X$ *do;*
  if $(K - \{A\}) \to X \in F^+$ *then* $K := K - \{A\};$
endfor;
end $DETERMINANTA.$

Condiţia (*if* $(K - \{A\}) \to X \in F^+$) se calculează apelând la algoritmul de apartenenţă [9], care necesită timp proporţional cu $|| F ||$. Deoarece această condiţie se verifică pentru orice atribut A din X, complexitatea algoritmului $DETERMINANTA$ este $O(| R | \cdot || F ||)$.

Algoritmul de calculare a cheilor posibile

Date de intrare:

F este o mulţime de dependenţe funcţionale divizată în clase de echivalenţă $F_1, ..., F_n$.

R_i este mulţimea de atribute implicate în dependenţele clasei de echivalenţă F_i.

Date de ieşire:

K este mulţimea de chei ale schemei $Sch_i(R_i, F)$.

Structuri de date adiţionale:

$DETs$ este mulţimea de determinante ale mulţimii R_i în raport cu mulţimea de dependenţe funcţionale F.

$DETsN$ este o submulţime a mulţimii $DETs$ folosită pentru construirea determinantelor noi.

$CHEI1(F, R_i, K)$

01. $K, DETs, DETsN := PS(F_i);$
02. *while* $DETsN \neq \emptyset$ *do;*
03. *select next* X *from* $DETsN;$
04. $DETsN := DETsN - \{X\};$
05. *for each* $V \rightarrow W \in (F - F_i)$ *do;*
06. $S := V \cup (X - V^+_{F-F_i}) \subseteq R_i;\ flag := true;$
07. *for each* $Z \in DETs$ *do;*
08. *if* $Z \subseteq S$ *then* $flag := false;$
09. *endfor;*
10. *if flag then do;*
11. *call* $DETERMINANTA(F, S, Y);$
12. $DETsN := DETsN \cup \{Y\};$
13. $DETs := DETs \cup \{Y\};$
14. *if* $Y \subseteq R_i$ *then* $K := K \cup \{Y\};$
15. *endif;*
16. *endwhile;*
17. *endfor;*
end $CHEI1.$

Uşor se observă că, ţinând cont de teorema 2, algoritmul $CHEI1$ calculează toate cheile nesintetizate ale schemei $Sch_i(R_i, F)$.

Pentru calcularea complexităţii temporale a algoritmului, se consideră mai întâi bucla 6-14.

Operaţia de atribuire din linia 6 se execută în $|| F ||$ paşi, deoarece închiderea unei mulţimi de atribute în raport cu o mulţime de dependenţe F necesită $O(|| F ||)$ timp. Calculele din linia 8 se efectuează în timp $O(| R |)$, unde R constituie mulţimea de atribute pe care sunt definite dependenţele din F. Bucla 7-9 se repetă de $| DETs |$ ori. Prin urmare, complexitatea acesteia este $O(| DETs | \cdot | R |)$. Linia 11 apelează la algoritmul $DETERMINANTA$, deci are nevoie de $O(| R | \cdot || F ||)$ timp. Liniile 12 şi 13 consumă un timp constant, iar linia 14 - $O(| R |)$. Astfel, bucla 6-14 necesită $O(| R | \cdot (| DETs | + || F ||))$ timp. Deoarece liniile 6-14 se calculează pentru fiecare dependenţă, iar liniile 3-14 pentru orice determinantă, întreg algoritmul necesită $O(| DETs | \cdot | F | \cdot | R | \cdot (| DETs | + || F ||))$ timp.

Referinţe:

1. Delobel C., Casey R. Decomposition of a data base and the theory of boolean switching functions // IBM Jour. Of Res. And Develop. - 1973.- Vol.17. - No.5. - p.374-386.

2. Fadoum R., Forsithe J. Finding candidate keys for relational data base.- In: Proc. 1^{st} ACM SIGMOD Conf., Cityplace // San Jose, 1975, p.203-210.

3. Fernandes M.C. Determining the normalization level of a relation on the basis of Armstrong's axiom// Computer an Artificial Intelligence.- 1984.- Vol.3.- No.6.- p.495-504.

4. Nekliudova E.A., Ţalenko M.Ş Sintez loghicescoi shemi releaţionnoi bazi dannih // Programmirovanie. - 1979. - No.6. - c.58-68.

5. Lucchesi C.L., Osborn S.L. Candidate keys for relations // Jour. Of Comput. And Syst. Sci. - 1978. - No.17.- p.270-279.

6. Zu C.T., Johnson D.T. On the complexity of finding the set of candidate keys for a given set of functional dependencies // Information Processing Letters. - 1976. - Vol.5. - No.4. - p. 100-101.

7. Beeri, C.; Bernstein, P.A. Computational Problems Related to the design of Normal Form Relations Schemes // ACM Trans. Database Syst. - 1979. - Vol.4. - No.1. - p.30-59.

8. Fernandes M.C. Determining the normalization level of a relation on the basis of Armstrong's axiom // Computer an Artificial Intelligence. - 1984. - Vol.3.- No.6. - p.495-504.

9. Maier D. The theory of relational database. - Computer Science Press, 1983.

10. Cotelea V. Baze de date relaţionale: proiectare logică. - Chişinău: ASEM, 1997.

Prezentat la 31.03.2009