ATRIBUTE NEESENŢIALE ŞI RECUPERABILE ÎN SCHEMELE BAZELOR DE DATE

V. Cotelea, dr. în informatică, conf. univ.
Academia de Studii Economice din Moldova

În articol, se introduc noţiunile a două tipuri de atribute: neesenţiale şi recuperabile, se prezintă unele proprietăţi ale acestora care pot servi la analiza schemelor bazelor de date. Algoritmii de determinare a atributelor neesenţiale şi recuperabile sunt de natură polinomială.

Schema bazei de date $Sch=(R,F)$ poate fi redată printr-o mulţime de dependenţe funcţionale F definite pe o mulţime de atribute R. Mulţimea de dependenţe funcţionale F poate fi simplificată. În primul rând, poate fi găsită o acoperire neredundantă a acesteia, într-al doilea rând, poate fi redusă, împărţită în clase de echivalenţă şi chiar minimizată. [1]

Două mulţimi de dependenţe funcţionale F şi G (se notează $F \equiv G$) sunt echivalente, dacă şi numai dacă $F^+=G^+$, adică, dacă închiderile acestor mulţimi sunt aceleaşi.

Fie că prin $|F|$ şi prin $\|F\|$ sunt notate cardinalitatea mulţimii F şi numărul de atribute antrenate de F (inclusiv cele repetate), respectiv.

Mulţimea de dependenţe funcţionale F se numeşte neredundantă, dacă $\nexists G$, astfel că $G \subset F$ şi $G \equiv F$.

Mulţimea de dependenţe funcţionale F se numeşte minimală, dacă $\nexists G$, astfel că $G \equiv F$ şi $|G|<|F|$.

Fie F o mulţime de dependenţe funcţionale asupra schemei R şi $X \to Y \in F$. Atributul A este redundant în dependenţa $X \to Y$ în raport cu F, dacă

$A \in X$, $F-\{X \to Y\} \cup \{(X-\{A\}) \to Y\} \equiv F$ sau

$A \in Y$, $F-\{X \to Y\} \cup \{X \to (Y-\{A\})\} \equiv F$.

Mulţimea F se numeşte redusă în stânga (dreapta), dacă orice dependenţă din F nu are atribute redundante în partea stângă (dreaptă). Mulţimea de dependenţe redusă în stânga şi în dreapta se numeşte redusă.

Fie $X \to Y \in F$. Se defineşte în calitate de clasă de echivalenţă de dependenţe funcţionale mulţimea de dependenţe, din care face parte şi $X \to Y$, notată $E_F(X)$:

$$E_F(X)=\{V \to W \mid V \to W \in F \wedge X \leftrightarrow V\}$$

Astfel, $E_F(X)$ este mulţimea de dependenţe din F cu părţile stângi echivalente cu X în raport cu F.

În continuare, se presupune că mulţimea F de dependenţe funcţionale este redusă şi împărţită în clase de echivalenţă: $F=F_1 \cup \ldots \cup F_n$

Se va nota cu R_i mulţimea de atribute antrenate de dependenţele clasei F_i, $i=\overline{1,n}$, şi se va nota cu $F-F_i(C)=(F-F_i) \cup \{X \to (Y-C) \mid X \to Y \in F_i\}$

Definiţia 1. Fie clasa F_i, unde $|F_i|>1$ şi $X \to YC \in F_i$. Atributul C este neesenţial în schema $Sch_i=(R_i,F)$, dacă, pentru orice două părţi stângi V şi Z, unde $V,Z \in PS(F_i)$, are loc $V \to Z \in (F-F_i(C))^+$.

Nu este greu de observat că atributul neesenţial C se găseşte în partea dreaptă doar a unei singure dependenţe din clasa de echivalenţă F_i. În caz contrar, mulţimea F nu ar fi o mulţime redusă de dependenţe funcţionale.

Trebuie menţionat că, dacă atributele C_1 şi C_2 sunt neesenţiale în schema Sch_i, nu este neapărat neesenţială uniunea acestora, $C_1 \cup C_2$.

Exemplul 1. Fie dată mulţimea $F=F_1 \cup F_2 \cup F_3$, unde $F_1=\{C_1 \to D\}$, $F_2=\{C_2 \to D\}$, $F_3=\{AD \to B,\ AB \to C_1C_2\}$. Atunci şi atributul C_1, şi atributul C_2 sunt neesenţiale în schema $Sch_3=(\{A,B,C_1,C_2,D\},\ F)$, dar nu este neesenţială uniunea lor, deoarece $AB \to AD \notin (F-F_3(C_1C_2))^+$.

Aici şi în continuare, în schema Sch_i, se scrie F, deoarece introducerea mulţimii F_i nu ar fi corectă, fiindcă mulţimea F_i^+ este o submulţime a mulţimii de dependenţe funcţionale definite pe mulţimea de atribute R_i. În acest context şi ţinând cont de ipoteza schemei universale [2], în Sch_i este prezentată mulţimea F, dar se subînţelege mulţimea de dependenţe din F^+, satisfăcute de relaţiile definite pe mulţimea R_i de atribute.

Definiţia 2. Atributul A, unde $A \in R_i$, este *recuperabil* în $Sch_i = (R_i, F)$, dacă $(R_i - A) \to A \in (F - F_i)^+$.

Din păcate, ca şi în cazul atributelor neesenţiale, uniunea atributelor *recuperabile* nu întotdeauna este *recuperabilă*.-

Din definiţia atributului neesenţial, reiese o proprietate remarcabilă a acestuia: în limitele clasei de echivalenţă, i se permite navigarea liberă pe părţile drepte ale dependenţelor. Cu toate acestea, închiderea mulţimii de atribute rămâne intactă. Însă, în noua mulţime, dependenţele pot deveni nereduse.

Exemplul 2. Fie $F = F_1 \cup F_2$, unde $F_1 = \{C \to B\}$ şi $F_2 = \{AD \to B,\ \ AB \to DC\}$. Atributul C este neesenţial în schema $Sch_2 = (\{A,B,C,D\},\ \ F)$, deoarece $AB \leftrightarrow AD$ în raport cu $F - F_i(C)$. Mulţimea F este echivalentă cu G, unde $G = G_1 \cup G_2$ şi $G_1 = F_1$, iar $G_2 = \{AD \to BC,\ \ AB \to D\}$. Însă dependenţa $AD \to BC$ nu este redusă, deoarece substituirea acesteia cu $AD \to C$ nu afectează mulţimea G^+. Este uşor de verificat faptul că atributul eliminat, B, este *recuperabil* în Sch_2.

Prin urmare, deplasarea atributelor neesenţiale poate fi folosită pentru obţinerea unei mulţimi echivalente de dependenţe, dar cu mai puţine simboluri atributive.

Algoritmul de calculare a atributelor neesenţiale se bazează pe următoarea teoremă:

Teoremă. Fie $F = F_1 \cup ... \cup F_n$ o mulţime minimală, redusă de dependenţe funcţionale şi fie $f : X \to YC \in F_i$. Atributul C este neesenţial în Sch_i, dacă şi numai dacă pentru orice parte stângă $Z \in PS(F_i)$ are loc $X \to Z \in (F - F_i(C))^+$.

Demonstraţie. Necesitatea. Veracitatea urmează direct din definiţia atributului neesenţial.

Suficienţa. Pentru afirmaţia inversă este de ajuns să se arate că pentru orice parte stângă $Z \in PS(F_i)$ are loc $Z \to X \in (F - f)^+$. Într-adevăr, din faptul că $Z, X \in PS(F_i)$, reiese că $Z \to X \in F^+$. Dar pentru ca f să fie utilizată neredundant în derivarea dependenţei $Z \to X$ din $F - f$ e necesar să aibă loc $Z \to X \in (F - f)^+$.

Algoritmul de calculare a atributelor neesenţiale

ATR_NEES(F *,* $ClasEchivDep$ *)*
  $MatrAtrNees := 0$ *;*
  for each $f : X \to Y \in F$ *do;*
    $i := ClasEchivDep(f)$ *;*
    for each $C \in Y$ *do;*
      $m := 0$ *;* $j := 0$ *;* $X_j := X$ *;*
      repeat
        $j := j + 1$ *;* $X_j := X_{j-1}$ *;*
        for each $V \to W \in (F - F_i(C))$ *do;*
          if $V \subseteq X_j$ *then* $X_j := X_j \cup W$ *;*
          if $V \in PS(F_i)$ *then* $m := m + 1$ *;*
        endfor;
      until $X_j = X_{j-1}$*;*
      if $|F_i| = m$ *then* $MatrAtrNees(i, C) := 1$ *;*
    endfor;
  endfor;
end ATR_NEES.

În algoritmul descris, $ClasEchivDep$ este un tablou care indică din ce clasă de echivalenţă face parte fiecare dependenţă din F, iar $MatrAtrNees$ este un tablou bidimensional, în care, pentru fiecare clasă, se fixează atributele neesenţiale. Închiderea mulţimii de atribute X în raport cu mulţimea de dependenţe $F - F_i(C)$ se calculează (bucla *repeat*) în timp $O(\|F\|)$ [3]. În aceeaşi buclă, se stabileşte dacă sunt satisfăcute condiţiile teoremei. Prin urmare, procedura *ATR_NEES* necesită timp $O(\|F\|^2)$.

Nu este greu să se observe că algoritmul de calculare a atributelor recuperabile are o complexitate temporală similară procedurii de determinare a atributelor neesenţiale.

Bibliografie

1. Maier, D. *The theory of relational database. Computer Science Press, 637 p., 1983.*
2. Kent, W. *The universal relation revised. ACM Trans. Database Syst., V.8, N 4, p.644…648, 1983*
3. Beeri, C.; Bernstein, P.A. *Computational Problems Related to the design of Normal Form Relations Schemes, ACM Trans. Database Syst., V.4, N 1, p.30…59, March 1979.*

Recomandat spre publicare: 21.01.2009