

Kuchárka normalizačných algoritmov

Ján Štúr FMFI-UK

Motto: Podstata normalizácie je minimalizácia
ľavých strán a maximalizácia pravých strán.

Výpočet uzáveru množiny atribútov – maximalizácia pravej strany

Vstup: množina atribútov X a množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$.

Výstup: uzáver X^* množiny X .

Algoritmus: $Y = X$;
repeat
 for $i = 1$ **to** n **do**
 if $L_i \in Y$ **then** $Y = Y \cup R_i$;
 until niečo pribudlo;
 $X^* = Y$;

Tento algoritmus funguje, ale nie je optimálny. Dôvodom je, že každú závislosť môžeme úspešne použiť najviac raz. V optimálnom algoritme by sa závislosti úspešne použité nesmeli znova skúšať.

Testovania vyplývania funkčných závislostí

Vstup: Závislosť $f_0 \equiv L_0 \rightarrow R_0$ a množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$.

Výstup: Pravdivostná hodnota výroku f_0 vyplýva z $\mathcal{F}$.

Algoritmus: Vypočítaj L_0^* vzhľadom k $\mathcal{F}$ použitím algoritmu uzáver množiny atribútov.
Výstup je **true**, ak $R_0 \subseteq L_0^*$, inak výstup je **false**.

Minimalizácia ľavej strany

Vstup: Množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$ a funkčná závislosť $L_0 \rightarrow R_0$ z $\mathcal{F}$,
kde $L_0 \equiv A_1 \dots A_r$.

Výstup: Funkčná závislosť $L'_0 \rightarrow R_0$ taká, že L'_0 neobsahuje žiaden zbytočný atribút.

Algoritmus: $Y = L_0$;
for $k = 1$ **to** r **do** **if** $R_0 \subseteq (Y - \{A_k\})^*$ **then** $Y = Y - \{A_k\}$;
/* Uzáver počítame vzhľadom k $(\mathcal{F} - \{L_0 \rightarrow R_0\}) \cup \{Y \rightarrow R_0\}$ */
 $L'_0 = Y$;

Minimálne pokrytie množiny funkčných závislostí

Vstup: Množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$.

Výstup: Množina funkčných závislostí $\mathcal{G}$ v kánonickom tvare taká, že $\mathcal{G}^* = \mathcal{F}^*$ a z $\mathcal{G}$ sa nedá žiadna závislosť vynechať bez narušenia uvedenej rovnosti.

Metóda: 1. Rozlož každú závislosť $L_i \rightarrow R_i$ z $\mathcal{F}$ na $|R_i|$ závislostí tak, aby na pravej strane bol len jeden atribút.
2. Minimalizuj ľavé strany všetkých „kanonických“ závislostí.
3. Pre takto vzniklé závislosti postupne pre každú závislosť zisti, či vyplýva z aktuálnej množiny závislostí. Ak áno vyškrtni uvedenú závislosť.

Poradie krokov 2 a 3 nemožno zameniť !

Môže existovať viac minimálnych pokrytí, ktoré z nich najdeme závisí na počiatočnom usporiadaní množiny závislostí. Algoritmus výpočtu jedného minimálneho pokrytia je polynomiálny.

Testovanie bezstrátovosti spojenia

Vstup: $\mathbf{S} = \{\mathcal{S}_k\}_{k=1}^p$ dekompozícia schémy $\mathcal{S} = \{A_j\}_{j=1}^m$ a množina funkčných závislostí $\mathcal{F}$.

Výstup: Pravdivostná hodnota, či $\mathbf{S}$ sa spája bezstrátovo.

Metóda:

1. Zostroj $\mathcal{G} = \{L_i \rightarrow A_i\}_{i=1}^n$ minimálne pokrytie $\mathcal{F}$.
2. **while** $\exists(u, v, i) (L_i \subseteq \mathcal{S}_u \cap \mathcal{S}_v) \wedge (A_i \in \mathcal{S}_u) \wedge (A_i \notin \mathcal{S}_v)$ **do**
 $\{\mathcal{S}_v = \mathcal{S}_v \cup \{A_i\};$
 if $\mathcal{S}_v = \mathcal{S}$ **then return(true);**
 }
return(false);

Testovanie bezstrátovosti spojenia nevyžaduje konštrukciu minimálneho pokrytia. Test v cykle while je, ale komplikovanejší o to, že treba hľadať atribút z pravej strany funkčnej závislosti.

Systematické generovanie všetkých kľúčov

Vstup: Schéma $\mathcal{S} = \{A_j\}_{j=1}^m$ a množina funkčných závislostí $\mathcal{F}$.

Výstup: Množina $\mathcal{K}$ všetkých kľúčov $\mathcal{S}$.

Metóda:

1. Vypočítaj minimálne pokrytie $\mathcal{G}$ množiny funkčných závislostí $\mathcal{F}$.
Nech $\mathcal{G} = \{L_i \rightarrow A_i\}_{i=1}^n$.
2. Nájdi jeden kľúč: minimalizáciou ľavej strany triviálnej závislosti $\mathcal{S} \rightarrow \mathcal{S}$. Nech je to K_1 . Označme $\mathcal{K} = \{K_l\}_{l=1}^k$ množinu už vygenerovaných kľúčov.
3. Polož $k = 1$ a $\mathcal{K} = \{K_1\}$ a aplikuj nasledujúci algoritmus:
 for $j = 1$ **to** k **do**
 for $i = 1$ **to** n **do**
 if $A_i \in K_j$ **then** $\{ Y = (K_j - A_i) \cup L_i;$
 if $\forall(1 \leq jj \leq k) K_{jj} \not\subseteq Y$ **then**
 $\{ \text{Minimalizuj ľavú stranu závislosti } Y \rightarrow \mathcal{S};$
 $\mathcal{K} = \mathcal{K} \cup \{Y\}; \quad k = k + 1;$
 }
 }
 }

Testovanie zachovania závislostí

Vstup: $\mathbf{S} = \{\mathcal{S}_k\}_{k=1}^p$ dekompozícia schémy $\mathcal{S} = \{A_j\}_{j=1}^m$ a množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$.

Výstup: Pravdivostná hodnota, či $\mathbf{S}$ zachová závislosti.

Algoritmus:

```
set function Closure(set X);  
    { var set Z;  
      integer i;  
       $Z = X;$   
      while niečo pribudlo do Z do  
          for  $i = 1$  to  $p$  do  $Z = Z \cup ((Z \cap \mathcal{S}_i)^* \cap \mathcal{S}_i); \quad // \text{uzáver vzhľadom k } \mathcal{F}$   
      return}(Z);  
    };  
boolean function main;  
    { var integer j;  
      boolean preserve = true;  
      for  $j = 1$  to  $n$  do  $\text{preserve} = \text{preserve} \wedge (R_j \subseteq \text{Closure}(L_j));$   
      return}(preserve);  
    }.
```

Dekompozícia do BCNF

Vstup: Schéma $\mathcal{S} = \{A_j\}_{j=1}^m$ a množina funkčných závislostí $\mathcal{F} = \{L_i \rightarrow R_i\}_{i=1}^n$.

Výstup: Dekompozícia $\mathcal{S}$ na systém schém $\mathbf{S} = \{\mathcal{S}_k\}_{k=1}^p$ takých, že pre každé k je $\mathcal{S}_k$ v BCNF.

Algoritmus: **boolean function** Decompose(**var** Z , **var** $\mathcal{S}_k$);

```

{ var  $Y$ ;
   $Y = Z$ ;
  if  $Y$  neobsahuje  $A, B$  také, že  $A \in \{Y - \{A, B\}\}^*$  then {  $\mathcal{S}_k = Y$ ; return(false);}
  /* Uzáver sa počíta vzhľadom k  $\mathcal{F}$ . */
  else { while  $Y$  obsahuje  $A, B$  také, že  $A \in \{Y - \{A, B\}\}^*$  do  $Y = Y - B$ ;
     $Z = Z - A$ ;
     $\mathcal{S}_k = Y$ ;
    return(false);
  }
};

void function main();
{ var  $Z = \mathcal{S}$ ;
   $k = 0$ ;
  bcnf: boolean;
  repeat  $k = k + 1$ ;
    bcnf = Decompose( $Z, \mathcal{S}_k$ );
  until bcnf;
}.
```

Syntéza 3NF nelámajúcej závislosti

Vstup: Schéma $\mathcal{S} = \{A_j\}_{j=1}^m$ a množina funkčných závislostí $\mathcal{F}$.

Výstup: Systém schém $\mathbf{S} = \{\mathcal{S}_k\}_{k=1}^p$ takých, že pre každé k je $\mathcal{S}_k$ v 3NF a $\mathbf{S}$ zachováva závislosti.

Metóda: 1. Zostroj $\mathcal{G} = \{L_i \rightarrow A_i\}_{i=1}^n$ minimálne pokrytie $\mathcal{F}$. Pre každú závislosť $f_i \equiv L_i \rightarrow A_i$ z $\mathcal{G}$ označíme $\mathcal{S}_i = L_i \cup \{A_i\}$ množinu všetkých atribútov, ktoré sa v nej vyskytujú.

Teraz môžeme pokračovať dvomi spôsobmi:

2. Vynecháme schémy, ktoré sú podmnožinami iných schém.

for $j = 1$ **to** n **do**

for $i = j + 1$ **to** n **do**

if $\mathcal{S}_i \subseteq \mathcal{S}_j$ **then** Označ f_i **else if** $\mathcal{S}_j \subset \mathcal{S}_i$ **then** Označ f_j ;

3. Zlúčime neoznačené závislosti s rovnakými ľavými stranami a im zodpovedajúce schémy zjednotíme. Vzniklé schémy sú už schémy vytvárajúce 3NF.

4. Ak atribúty ani jednej zo schém z predošlého kroku netvorí nadkľúč shémy $\mathcal{S}$, pridáme jeden kľúč K pre schému $\mathcal{S}$.

Druhá alternatíva je trochu zložitejšia vedie však k 3NF s menším počtom tabuliek.

3'. Rozdelíme závislosti z $\mathcal{G}$ do skupín tak, že do jednej skupiny sa dostanú závislosti s ekvivalentnými ľavými stranami: $L_i \leftrightarrow L_j$ práve vtedy, keď $(L_i \rightarrow L_j) \wedge (L_j \rightarrow L_i)$. Označíme $\mathcal{H}$ množinu závislostí tvrdiacich ekvivalencie ľavých strán.

4'. Pre každú skupinu vyberieme jedného reprezentanta (napr. ľavú stranu minimálnej dĺžky prvú v poradí). V $\mathcal{G}$ vo všetkých ľavých stranách nahradíme výskyty iných ľavých strán z tejto týmto reprezentantom. Dostaneme množinu závislostí $\mathcal{G}'$.

5'. Minimalizujeme $\mathcal{G}'$ vyškrtnutím zbytočných závislostí. Závislosť f z $\mathcal{G}'$ odstránime ak sa dá odvodiť zo zjednotenia $\mathcal{H}$ a zvyšných závislostí $\mathcal{G}'$.

schému $\mathcal{S}_k$ ako zjednotenie atribútov ľavých a pravých strán závislostí z tejto skupiny.

5. Pridaj kľúč K ako ďalšiu schému.

6. Keď nejaká schéma $\mathcal{S}_l \subseteq \mathcal{S}_k$ vynechaj $\mathcal{S}_l$.