Inférence de dépendances fonctionnelles

Marc Plantevit
marc.plantevit@liris.cnrs.fr

Lyon 1

liris.cnrs.fr/marc.plantevit/doku/doku.php?id=lif10

Inférence de dépendances

Notion d'inférence

- Concept fondamental pour la théorie des bases de données.
- C'est la déduction des contraintes qui sont sémantiquement impliquées par d'autres.
- On va donner un algorithme (un calcul symbolique) qui permet de décider *effectivement* de cette implication logique.
- L'algorithme est un processus qui manipule des symboles, il s'appuie uniquement sur la *syntaxe* des dépendances. On vérifiera que cet algorithme est :
  - correct : toutes les déductions syntaxiques sont bien sémantiquement valides ;
  - complet : toutes les déductions sémantiquement valides peuvent bien être syntaxiquement déduites.

Outline

1. Inférence des dépendances fonctionnelles

Implication logique des dépendances fonctionnelles

Définition

Soit F un ensemble de DF sur un schéma de relation R et f une DF sur R. On surcharge $\models$ pour un ensemble de DFs :

$$r \models F \text{ ssi } \forall f \in F. r \models f$$

F implique logiquement (sémantiquement) f, noté

$$F \models f \text{ ssi } \forall r. r \models F \Rightarrow r \models f$$

Exemple

Soit le schéma $Etudiant(Num, Nom, Ville, Region, CP)$ et l'ensemble de dépendances $F = \{f_1 = Num \rightarrow Nom, Ville, Region\ f_2 = Ville, Region \rightarrow CP\}$. On peut déduire les DFs suivantes :

- $f_1' = Num \rightarrow Nom, Ville$
- $f_2' = Num \rightarrow CP$

Implication logique des dépendances fonctionnelles

En utilisant la sémantique des dépendances fonctionnelles, prouver qu'effectivement $F \models f_1'$ et $F \models f_2'$

- Difficile de prouver que chaque intuition est correcte.
- Comment s'assurer que l'énumération $\{f \mid F \models f\}$ est exhaustive ?
- On va donner un algorithme qui le fait.

Règles et système d'inférences

- Une règle d'inférence est une expression de la forme

$$\frac{f_0 \qquad \dots \qquad f_n}{f}$$

- Un système d'inférence est un ensemble de règles d'inférence.
- Une preuve de f à partir de F notée $F \vdash f$ est une *séquence* $(f_0, \dots, f_n)$ de DFs telle que $f_n = f$ et $\forall i \in [0..n]$, :
  - soit $f_i \in F$;
  - soit f_i est la conséquence d'une règle dont toutes les prémisses $f_0 \dots f_p$ apparaissent avant f_i dans la séquence.

Liens avec LIF11

Ce sont bien les même notions que celles des systèmes de déduction $\mathcal{G}$ et $\mathcal{LK}$. Pour les preuves formelles, on pourra utiliser le même formalise des arbres de preuves.

Système d'Armstrong

Système d'inférence d'Armstrong

Soit F un ensemble de DF sur un schéma de relation R. Les règles d'inférence suivantes sont appelées *système d'Armstrong*

- Réflexivité

$$\frac{Y \subseteq X}{X \rightarrow Y}$$

- Augmentation

$$\frac{X \rightarrow Y}{WX \rightarrow WY}$$

- Transitivité

$$\frac{X \rightarrow Y \qquad Y \rightarrow Z}{X \rightarrow Z}$$

Propriétés du système d'Armstrong

Correction et complétude

- Il est correct : $F \vdash f \Rightarrow F \models f$
- Il est complet : $F \models f \Rightarrow F \vdash f$

$$F \models \alpha \Leftrightarrow F \vdash \alpha$$

Exemple

Considérons l'ensemble $\Sigma = \{A \to C, B \to C, CD \to E\}$ de DFs sur $\{A, B, C, D, E\}$. Montrons que $\Sigma \vdash AD \to E$

$$\dfrac{\dfrac{\dfrac{A \to B \qquad B \to C}{A \to C}}{AD \to CD} \qquad CD \to E}{AD \to E}$$

Par complétude, on a montré que $\Sigma \models AD \to E$.

Autres règles

- Décomposition

$$\frac{X \rightarrow YZ}{X \rightarrow Y}$$

- Composition

$$\frac{X \rightarrow Y \qquad X \rightarrow Z}{X \rightarrow YZ}$$

- Pseudo-transitivité

$$\frac{X \rightarrow Y \qquad WY \rightarrow Z}{WX \rightarrow Z}$$

Prouver que ces règles sont valides et qu'on peut les ajouter au système d'Armstrong

Exercice

Soit l'ensemble F de DF suivant sur le schéma $R = ABCDEFG$:

- $A \rightarrow B$
- $A \rightarrow C$
- $A \rightarrow D$
- $CD \rightarrow E$
- $BE \rightarrow F$
- $ABE \rightarrow G$
- $EG \rightarrow ABD$
- $FG \rightarrow AE$
- Démontrer que DF $F \models A \rightarrow F$.
- Démontrer que DF $F \models A \rightarrow G$.

Problème d'inférence de DF

Le système d'Armstrong permet de systématiser la recherche de toutes les DF impliquées par d'autres DF.

Problème d'inférence des DF

Soit F un ensemble de DF, et f une DF, a-t-on $F \models f$?

- La résolution du problème d'inférence est "facile" pour les DF,
- linéaire en la taille de F et de f.
- Le concept fondamental utilisé dans cet algorithme est celui de fermeture.

Fermeture d'un ensemble de DFs

Soit F un ensemble de DF, on note F^+ la fermeture de F, l'ensemble de toutes les Dfs logiquement impliquées par F :

$$F^+ = \{f \mid F \models f\}$$

Fermeture d'un ensemble d'attributs

Fermeture d'un ensemble d'attributs

X est un ensemble d'attribut et F un ensemble de DF, on note X^+ la fermeture de X par rapport à F l'ensemble de tous les attributs qu'on peut "déduire" de X par des dépendances fonctionnelles :

$$X^+ = \{A \mid F \models X \rightarrow A\}$$

D'après la définition de la fermeture de F, on a de façon équivalente :

$$X^+ = \{A \mid X \rightarrow A \in F^+\}$$

Il faut aussi tenir compte de *toutes* les DFs qui qui sont dérivables à partir de F.

Définition alternative de la notion de clé

- une clé de R est un ensemble $X \subseteq R$ tels que $R : X \to R$
- X est clé d'un schéma R *ssi* $X^+ = R$

Lemme

Soient F un ensemble de DF et $X \to Y$ une DF :

$$F \models X \to Y \text{ ssi } Y \subseteq X^+$$

- Ainsi, pour tester si on a $F \models X \to Y$, on calcule X^+ et on vérifie si $Y \subseteq X^+$
- On va utiliser un algorithme pour calculer simplement X^+
- On obtient ainsi un algorithme pour décide de l'implication logique des DFs.

Algorithme : fermeture d'un ensemble d'attributs

Data: F un ensemble de DF, X un ensemble d'attributs.
Result: X^+, la fermeture de X par F.
$unused := F$
$closure := X$
repeat
    $closure' := closure$
    if $W \to Z \in unused$ *and* $W \subseteq closure$ **then**
        $unused := unused - \{W \to Z\}$
        $closure := closure \cup Z$
    end
until $closure' = closure$;
retourner *closure*

- L'algorithme permet de vérifier si un ensemble de DF implique logiquement une dépendance d'après le lemme vu avant.
- Pour tester l'implication d'un *ensemble de dépendances*, il suffit de tester l'implication de *chaque* dépendance.

Combien de fois (au plus) teste-t-on $W \subseteq closure$ en fonction de $|F| = n$?

Un algorithme linéaire

Amélioration pour obtenir un temps linéaire en la taille de $|F|$ *faire en sorte de ne se servir d'une DF qu'une seule fois* quand c'est nécessaire :

- Pour chaque $X \to Y \in F$ non utilisée, il faut stocker le *nombre* d'attributs de X non encore dans *closure*.
- Pour le faire efficacement, il faut maintenir à jour une liste pour chaque attribut A des DFs de F non utilisées pour lesquelles A apparaît en partie gauche.

Algorithme : fermeture linéaire

```
for W → Z ∈ F do
    count[W → Z] := |W|
    for A ∈ W do
        list[A] := list[A] ∪ W → Z
    end
end
closure := X, update := X
while update ≠ ∅ do
    update := update \ {A}
    for W → Z ∈ list[A] do
        count[W → Z] := count[W → Z] − 1
        if count[W → Z] = 0 then
            update := update ∪ (Z \ closure)
            closure := closure ∪ Z
        end
    end
end
return closure
```

Exercice

Avec l'ensemble de de DFs donné précédemment :

- Démontrer que $F \models A \rightarrow F$ (avec fermeture).
- Démontrer que $F \models A \rightarrow G$ (avec fermeture).
- Démontrez que BEF n'est pas une clé de R
- Démontrez que $CDE \rightarrow A$ n'est pas impliquée par F

Fin du troisième cours.