View-based query answering

Maurizio Lenzerini

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

Anno Accademico 2004/2005

Prerequisites

Calcolo relazionale (FOL queries)

Calcolo relazionale: le basi

Idea base: si usa il linguaggio della logica del primo ordine per esprimere le tuple che si vogliono ottenere come risultato.

Sia U un insieme infinito numerabile di attributi, che possono assumere valori di un insieme qualunque D (dominio per le base di dati), e sia COST un insieme di simboli di costanti che contiene un simbolo di costante per ogni valore di D.

Sia S uno schema di base di dati. Diciamo che una base di dati B su S e D è una istanza di S che contiene come dati le costanti di COST (che, come abbiamo visto prima, sono interpretabili come valori di D).

Sia L_S il linguaggio della logica del primo ordine i cui simboli di costante sono quelli in COST, ed i cui simboli di predicato sono quelli di S più tutti quelli che possono comparire in una condizione booleana dell'algebra relazionale (ad esempio: <,>,…), ed i cui simboli di funzione sono assenti.

Calcolo relazionale: sintassi

Una espressione del calcolo relazionale (sul dominio) ha la forma:

$$\{ A_1 : x_1,\ldots,A_k : x_k \mid F \}$$

dove:

- $k > 0$
- $A_1,\ldots,A_k$ sono attributi,
- $x_1,\ldots,x_k$ sono variabili,
- F è il corpo dell'espressione, cioè una formula del linguaggio L_S del primo ordine in cui le uniche variabili libere sono $x_1,\ldots,x_k$
- $A_1 : x_1,\ldots, A_k : x_k$ viene chiamata la target list della espressione, e si può anche scrivere semplicemente come $x_1,\ldots,x_k$ (intendendo che ogni valore nella terget list è identificato dalla posizione, e non dall'attributo)

Se R è una relazione con schema $R(A_1,\ldots,A_h)$, in F, un atomo con simbolo di predicato R ha la forma

$$R(x_1,\ldots,x_h)$$

che si può anche scrivere come $R(A_1 : x_1,\ldots,A_h : x_h)$.

Calcolo relazionale: semantica

Le espressioni del calcolo relazionale si valutano su particolari interpretazioni. Nel seguito, faremo sempre riferimento ad uno schema S di basi di dati.

Una interpretazione corretta per il calcolo relazionale è una coppia <D,B>, dove D è un dominio, e B è una base di dati su S e D.

Il valore della espressione del calcolo relazionale

$$\{ A_1 : x_1, \ldots, A_k : x_k \mid F \}$$

in una interpretazione M = <D,B> è l'insieme delle tuple $<c_1,\ldots,c_k>$ di costanti in COST tale che

$$<M,v> \models F$$

dove v è l'assegnazione che associa ad ogni variabile x_i nella target list la corrispondente costante c_i nella tupla $<c_1,\ldots,c_k>$.

Calcolo relazionale: semantica

Quando è evidente qual è il dominio D su cui è definita la base di dati B, la specifica di D si può omettere e scrivere direttamente <B,v> |= F invece di < <D,B>, v > |= F.

Il risultato della valutazione di una espressione { $A_1 : x_1,\ldots, A_k : x_k$ | F } rispetto ad una base di dati B su S e D è la relazione R tale che

- lo schema di R è l'insieme degli attributi $A_1,\ldots,A_k$ che compaiono nella target list
- l'istanza di R è l'insieme delle tuple che costituiscono il valore dell'espressione nell'interpretazione <D,B>.

Teorema Per ogni espressione del calcolo relazionale, c'è una espressione equivalente in cui la formula F in essa contenuta coinvolge solo negazione, un connettivo tra disgiunzione e congiunzione, ed un quantiifcatore tra esistenziale e universale.

Conjunctive query

Una particolare sottoclasse del calcolo relazionale è costituita dalla query congiuntive (conjunctive query).

Una conjunctive query è una espressione del calcolo relazionale (sul dominio) che ha la forma:

$$\{ x_1,\ldots,x_k \mid \exists\, y_1,\ldots,y_h \ (a_1 \wedge \ldots \wedge a_m) \}$$

Dove:

- K > 0, e h >= 0
- $a_1,\ldots,a_m$ sono atomi il cui predicato è un simbolo di relazione
- $x_1,\ldots,x_k,y_1,\ldots,y_h$ sono le uniche variabili che compaiono in $a_1,\ldots,a_m$

Complessità del calcolo relazionale

Complessità delle conjunctive query:

- La complessità rispetto ai dati delle conjunctive query è polinomiale.
- Per quanto riguarda la complessità combinata, il problema di valutare se una tupla è contenuta nel risultato della valutazione di una conjunctive query è NP-completo.

Complessità del calcolo relazionale:

- La complessità rispetto ai dati del calcolo relazionale è polinomiale.
- Per quanto riguarda la complessità combinata, il problema di valutare se una tupla è contenuta nel risultato della valutazione di una espressione del calcolo relazionale è Pspazio-completo.

Datalog

Datalog

Datalog è basato sulla logica del primo ordine senza simboli di funzione, come il calcolo relazionale. L'insieme delle costanti COST si chiama anche universo di Herbrand.

In Datalog, l'unità fondamentale è la clausola di Horn. Ricordiamo che una clausola è una disgiunzione di literal, e un literal o è un atomo, oppure è la negazione di un atomo. Tutte le variabili di una clausola si intendono quantificate universalmente. Una clausola di Horn è una clausola con al più un literal positivo.

Esempio (le maiuscole sono variabili, le minuscole sono costanti):

$C = (\neg\, p(X,a) \vee q(Y,b))$ che corrisponde a $(\forall X\, \forall Y\, (\neg\, p(X,a) \vee q(Y,b))$

Una clausola si scrive anche come l'insieme dei suoi literal. Ad esempio, C si può scrivere come $\{\neg\, p(X,a)\ ,q(Y,b)\ \}$.

Una clausola di dice positiva se non ha literal negativi. Si dice unit se ha un literal, e si dice ground se vi compaiono costanti ma non variabili.

Datalog

Ci sono tre tipi di clausole di Horn:

- Fatti: clausole con un literal positivo e senza literal negativi. Ad esempio,

  { padre(mario, aldo) }

  è un fatto, che si scrive anche semplicemente come

  padre(mario,aldo).

- Regola: clausola con un literal positivo ed almeno uno negativo. Ad esempio:

  {¬ genitore(X), padre(X), ¬ uomo(X) }

  è una regola, che si scrive anche come

  padre(X) :- genitore(X), uomo(x).

  padre(X) si dice testa della regola, e genitore(X), uomo(x) si dice corpo della regola

- Goal: clausola con un literal negativo e senza literal positivi. Ad esempio

  {¬ padre(mario,X) }

  è un goal, che si scrive anche come

  ?- padre(mario,X).

Datalog

L'insieme dei simboli di predicato è partizionato in due insiemi: EPred e IPred.

- EPred contiene i predicati che corrispondono alle relazioni della base di dati
- IPred contiene i predicati che corrispondono a relazioni che vengono definite nel programma Datalog.

La base di Herbrand HB è l'insieme di tutte le clausole ground unit positive che si possono formare con i predicati e le costanti. Denotiamo con EHB il sottoinsieme della base di Herbrand formato da atomi i cui predicati appartengono ad EPred, e con IHB il sottoinsieme della base di Herbrand formato da atomi i cui predicati appartengono ad IPred.

Programmi Datalog

In Datalog una query si esprime mediante un programma.

Un programma Datalog P è un insieme finito di clausole di Horn che

- contiene al massimo un goal
- è tale che, per ogni C in P che non è il goal, si ha che C è una clusola Datalog, ovvero:
  - o C appartiene a EHB, cioè è una clausola ground unit positiva il cui predicato appartiene a EPred,
  - oppure C è una regola che soddisfa queste condizioni:
    - il predicato che compare nella testa di C appartiene a IPred
    - tutte le variabili che compaiono nella testa compaiono anche nel corpo (safety condition)

Datalog: esempio di programma

Siano:

EPred = { par, lives }
IPred = { anc, person }

Questo è un programma Datalog:

```
anc(X,Y) :- par(X,Y).
anc(X,Y) :- par(X,Z), anc(Z,Y).
anc(adamo,X) :- person(X).
person(X) :- lives(X,Y).
```

Datalog: semantica

Sia C una clausola. Una sostituzione per C è un insieme finito della forma

$$\{ X_1 / t_1, \ldots, X_n / t_n \}$$

tale che le varie X_i sono variabili distinte di C, ogni t_i è un termine, e X_i è diverso da t_i per ogni i.

Se lambda è una sostituzione, e C è costituita dai literal L1,…,Ln, allora lambda(C) denota la clausola ottenuta da C sostituendo ad ogni Li il literal lambda(Li) ottenuto applicando a Li la sostituzione lambda. In particolare, lambda(Li) si ottiene sostituendo simultaneamente ogni variabile Xj che occorre in Li con il termine tj se e solo se Xj / tj è un elemento di lambda.

Se per una coppia di literal <L,M> esiste una sostituzione lambda tale che lambda(L)=lambda(M), allora lambda si dice unifier per L e M. Una sostituzione gamma si dice più generale di una sostituzione lambda se esiste una sostituzione theta tale che gamma · theta = lambda, dove gamma · theta indica la composizione di theta e gamma.

Il most general unifier (MGU) di L e M è un unifier lambda tale che non esiste alcun altro unifier di L e M che è più generale di lambda.

Datalog: semantica

Per stabilire la semantica di Datalog potremmo ricorrere semplicemente alla logica del primo ordine, in particolare alle nozioni di interpetazione, di modello, e di implicazione logica, in modo classico. In realtà, vogliamo interpretare ogni programma Datalog come una query, e quindi come funzione. È quindi necessario definire la semantica in modo adeguato a questo scopo.

Definizione La semantica di un programma P senza goal è data dalla funzione

$$M_P : 2^{EHB} \rightarrow 2^{IHB}$$

(dove 2^S denota l'insieme di tutti i sottoinsieme di S) tale che per ogni sottoinsieme V di EHB:

$$M_P(V) = \{ G \mid G \in IHB \wedge (P \cup V) \models G \}$$

Definizione La semantica di un programma P con goal H è data dalla funzione

$$M_{P,H} : 2^{EHB} \rightarrow 2^{HB}$$

tale che per ogni sottoinsieme V di EHB, si ha che

$$M_{P,H}(V) = \{ G \text{ istanza di } H \mid G \in HB \wedge (P \cup V) \models G \}$$

Datalog: interpretazioni di Herbrand

Nella definizione della semantica appena vista si fa riferimento alla condizione

$$(P \cup V) \models G$$

Herbrand e Skolem hanno indipendentemente dimostrato che, sulla base del fatto che P è un insieme di clausole, per verificare la condizione suddetta non è necessario considerare tutte le possibili interpretazioni di $P \cup V$, ma è sufficiente limitare l'attenzione alle interpretazioni di Herbrand.

Una interpretazione di Herbrand è una interpretazione che assegna

- ad ogni costante se stessa
- ad ogni atomo ground appartenente ad HB un valore di verità.

Una interpretazione di Herbrand si può vedere come un insieme di atomi ground, quelli ai quali si assegna il valore "vero". Un modello di Herbrand di un insieme di clausole è una interpretazione di Herbrand che soddisfa tutte le clausole dell'insieme.

Teorema (Herbrand e Skolem) Se P e V sono insiemi di clausole, allora $(P \cup V) \models G$ se e solo se ogni modello di Herbrand di $(P \cup V)$ è anche un modello di G.

Datalog: l'insieme cons

Il fatto che sia sufficiente limitare l'attenzione alle interpretazioni di Herbrand non assicura ancora che abbiamo a disposizione un metodo per calcolare la risposta ad una query Datalog.

Fortunatamente, vedremo che ci viene in aiuto un teorema fondamentale, per il quale abbiamo bisogno di questa nozione: se S è un insieme di clausole Datalog, indichiamo con cons(S) l'insieme di tutti i fatti ground che sono conseguenze logiche di S:

$$cons(S) = \{ F \mid F \in HB \wedge S \models F \}$$

Ovviamente, per un programma Datalog P, ed un insieme di fatti V, si ha che

$$M_P(V) = cons(P \cup V) \cap IHB$$

Quindi, se sappiamo come calcolare cons(S), per un insieme finito S, allora sappiamo "calcolare" la semantica di un programma Datalog.

Datalog: modello minimo

Sia S un insieme di clausole datalog (cioè regole e fatti).

Teorema Se S è un insieme di clausole Datalog (cioè regole e fatti), allora cons(S) è uguale alla intersezione di tutti i modelli di Herbrand di S, ed è esso stesso un modello di Herbrand.

cons(S) si chiama modello minimo di Herbrand di S (o semplicemente modello minimo di S).

Resta a questo punto di vedere come si calcola il modello minimo di un programma Datalog. Ci sono diversi algoritmi per questo calcolo, e noi illustreremo quello più semplice.

Datalog: algoritmo di valutazione

```
sottoinsiemeDiHB Compute(program P)    // P programma Datalog senza goal
{  old := ∅; new := P;
   while new <> old {
        old := new; passo := ∅;
        for each regola R della forma L0 :- L1,…,Ln in P
            for each tupla <F1,…,Fn> di fatti ground di old {
                for i := 0 to n Ki := Li;
                for i := 1 to n {
                    lambda := MGU(Ki,Fi);
                    if lambda = NIL
                    then { K0 = NIL; exit-for }
                    else for j := 0 to n Kj := lambda(Kj)
                }
                if K0 <> NIL then passo := passo ∪ {K0};
            } // passo contiene i fatti dedotti con un passo di regole a partire da old
       new := new ∪ passo;
   }
   return tutti i fatti in new;
}
```

Teorema Se P è un programma Datalog senza goal, allora Compute(P) termina restituendo il modello minimo di P.

Complessità di Datalog

Complessità di Datalog:

- La complessità rispetto ai dati di Datalog è polinomiale.
- Per quanto riguarda la complessità combinata, il problema di valutare se una tupla è contenuta nel risultato della valutazione di un programma Datalog è EXPTIME-completo.

Espressività di Datalog

Teorema Esiste un programma Datalog che, per ogni base di dati **B**, calcola la chiusura transitiva di r, dove r è una relazione binaria in **B**.

Sia RA_m l'algebra relazionale senza l'operatore di differenza.

Teorema Datalog è più espressivo di Ra_m (quindi delle conjunctive query).

La dimostrazione procede mostrando che:

- ogni espressione di Ra_m può essere trasformata in modo equivalente in Datalog,
- la chiusura transitiva di una relazione binaria si può esprimere in Datalog, ma non in algebra relazionale.

Teorema Esiste una query che si può esprimere nell'algebra relazionale ma non in Datalog. Quindi Datalog non è più espressivo dell'algebra relazionale e non è più espressivo del calcolo relazionale con dichiarazioni di range.