Fondamenti di Teoria delle Basi di Dati

Riccardo Torlone

Parte 7: Datalog

Calcolo e algebra relazionale: limiti

- Calcolo e algebra sono sostanzialmente equivalenti: l'insieme di interrogazioni con essi esprimibili è quindi significativo; il concetto è robusto
- Ci sono però interrogazioni interessanti non esprimibili:
  - calcolo di valori derivati: possiamo solo estrarre valori, non calcolarne di nuovi; calcoli di interesse:
    - a livello di tupla o di singolo valore (conversioni somme, differenze, etc.)
    - su insiemi di tuple (somme, medie, etc.)
  - interrogazioni inerentemente ricorsive, come la chiusura transitiva

Chiusura transitiva

Supervisione(Impiegato, Capo)

- Per ogni impiegato, trovare tutti i superiori (cioè il capo, il capo del capo, e cosi' via)

Impiegato	Capo
Rossi	Lupi
Neri	Bruni
Lupi	Falchi

Impiegato	Superiore
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Rossi	Falchi

Chiusura transitiva, come si fa?

- Nell'esempio, basterebbe il join della relazione con se stessa, previa opportuna ridenominazione
- Ma:

Impiegato	Capo
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Falchi	Leoni

Impiegato	Superiore
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Rossi	Falchi
Lupi	Leoni
Rossi	Leoni

Chiusura transitiva, impossibile!

- Non esiste in algebra e calcolo relazionale la possibilità di esprimere l'interrogazione che, per ogni relazione binaria, ne calcoli la chiusura transitiva
- Per ciascuna relazione, è possibile calcolare la chiusura transitiva, ma con un'espressione ogni volta diversa:
  - quanti join servono?
  - non c'è limite!

Datalog

- Un linguaggio per basi di dati basato sulla logica derivato dal Prolog
- Si basa su una approccio *proof theoretic*
  - Ma si può dare anche una semantica model theoretic
- Utilizza predicati di due tipi:
  - estensionali: relazioni della base di dati
  - intensionali: corrispondono alle viste
- Il linguaggio è basato su regole utilizzate per "definire" i predicati intensionali

Datalog, sintassi

- Regole:

  testa ← *corpo*

  - *testa* è un predicato atomico (intensionale)
  - *corpo* è una lista (congiunzione) di predicati atomici

- Le interrogazioni sono specificate per mezzo di predicati atomici (convenzionalmente preceduti da "?")

Esempio 1

- Trovare matricola, nome ed età di tutti gli impiegati

$$\pi_{\text{Matricola, Nome, Età}}(\text{Impiegati})$$

{ Matricola: m, Nome: n, Età: e |
Impiegati(Matricola: m, Nome: n, Età: e, Stipendio: s)}

InfoPubbliche(Matricola: m, Nome: n, Età: e)
← Impiegati(Matricola: m, Nome: n, Età: e, Stipendio: s)

? InfoPubbliche(Matricola: m, Nome: n, Età: e)

Esempio 2

- Trovare le matricole dei capi degli impiegati che guadagnano più di 40 mila

{ Capo: c | Supervisione(Capo:c,Impiegato:m) $\wedge$
Impiegati(Matricola: m, Nome: n, Età: e, Stipendio: s) $\wedge$ s > 40 }

ImpRicchi(Matricola:m, Nome:n, Età:e, Stipendio:s) $\leftarrow$
Impiegati(Matricola:m, Nome:n, Età:e, Stipendio:s), s > 40

CapiDeiRicchi (Capo:c) $\leftarrow$
ImpRicchi(Matricola:m, Nome:n, Età:e, Stipendio:s),
Supervisione (Capo:c, Impiegato:m)

? CapiDeiRicchi (Capo:c)

Esempio 5

- Trovare matricola e nome dei capi i cui impiegati guadagnano tutti più di 40 mila.
  - serve la negazione

CapiDiNonRicchi (Capo:c) ←
Supervisione (Capo:c,Impiegato:m),
Impiegati (Matricola: m, Nome: n, Età: e, Stipendio: s) ,
$s \leq 40$

CapiSoloDiRicchi (Matricola: c, Nome: n) ←
Impiegati (Matricola: c, Nome: n, Età: e, Stipendio: s) ,
Supervisione (Capo:c,Impiegato:m),
¬ CapiDiNonRicchi (Capo:c)

? CapiSoloDiRicchi (Matricola: c, Nome: n)

Esempio 6

- Per ogni impiegato, trovare tutti i superiori.
  - Serve la ricorsione

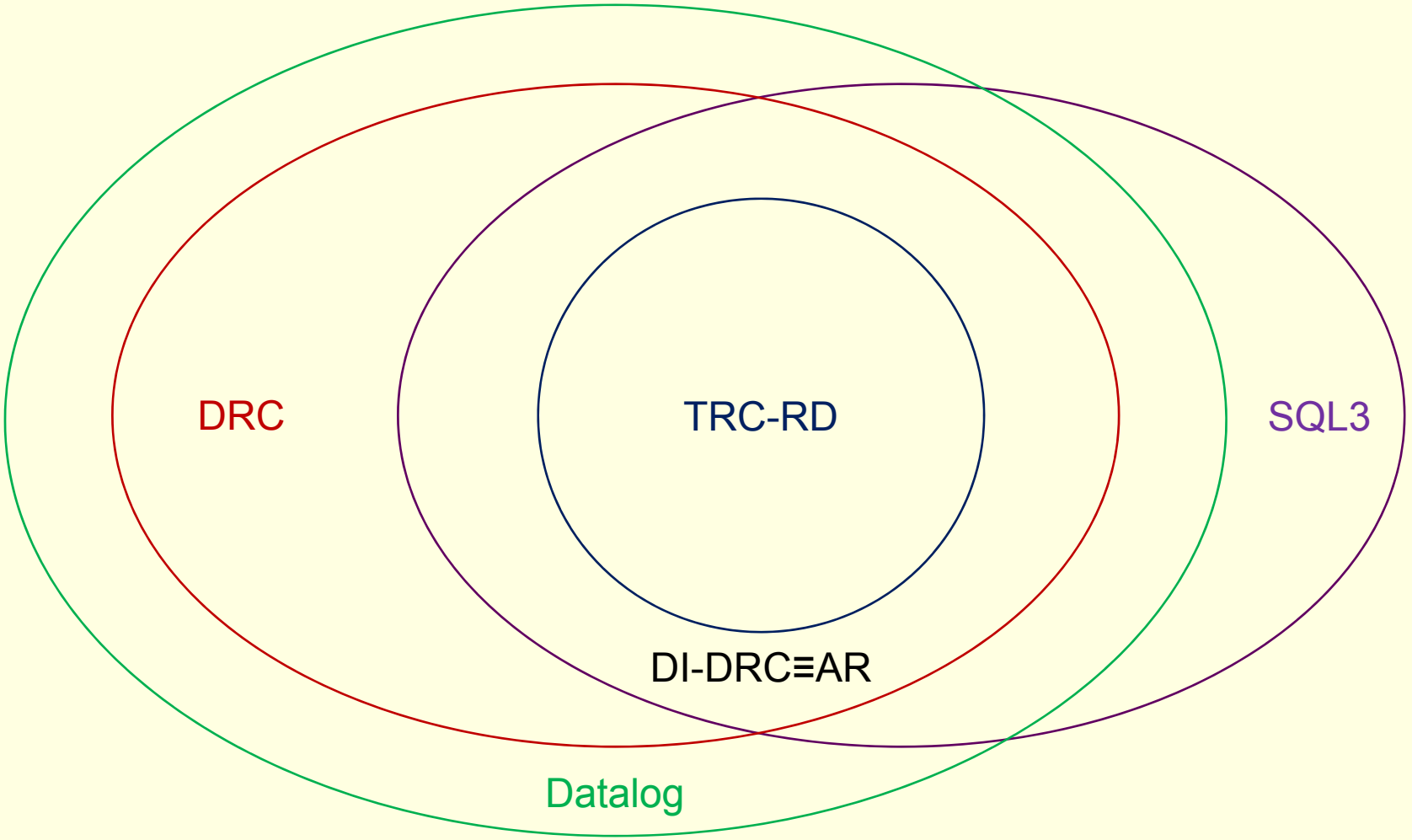
Superiore (Impiegato: i, SuperCapo: c) ←
    Supervisione (Impiegato: i, Capo: c)

Superiore (Impiegato: i, SuperCapo: c) ←
    Supervisione (Impiegato: i, Capo: c'),
    Superiore (Impiegato: c', SuperCapo: c)

Confronto con gli altri linguaggi

- La definizione della semantica delle regole ricorsive è delicata (in particolare con la negazione)
- Potere espressivo:
  - Datalog non ricorsivo con negazione è equivalente al calcolo e all'algebra
  - Datalog ricorsivo senza negazione e calcolo sono incomparabili
  - Datalog ricorsivo con negazione è più espressivo di calcolo e algebra

Relazione tra linguaggi



Esiste un concetto di completezza assoluto?

- Che potenza espressiva deve avere un linguaggio di interrogazione di basi di dati?
  - Efficacia
  - Efficienza
- Qual è la potenza espressiva “massima”?
  - Intuitivamente: da una base di dati non è possibile inventare nuovi dati ma solo estrarre (tutta) l’informazione presente nella base di dati stessa.
    - Bancilhon & Paredaens
    - Chandra and Harel