Lezione16

Table of contents

Unificazione di Termini

Sostituzione

Partendo da un insieme di atomi noti A e un insieme di variabili note V, definiamo una *sostituzione* come un insieme finito eventualmente vuoto, formato da coppie dove a sinistra mettiamo un termine e a destra mettiamo una variabile

$$\{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$$

dove le x sono variabili e t dei termini.
Inoltre richiediamo:

- che qualsiasi sia n, t_i è diverso da x_i per evitare di generare tante varianti equivalenti della stessa sostituzione;
- che tutte le x siano diverse, siccome sostituzioni ben fatte sono proprio questo.

A sostituzione in mano, preso un termine t e una sostituzione θ, se scriviamo $t\theta$ stiamo scrivendo un nuovo termine che si ottiene sostituendo simultaneamente tutte le variabili nella parte destra degli elementi di sostituzione θ, con i rispettivi termini.
Stiamo sostituendo tutte le variabili nelle parentesi, con i loro valori associati, in una volta sola.

Differenza rispetto i linguaggi

Nei linguaggi di programmazione, siamo abituati a trovare le variabili a sinistra dell'operando di assegnamento, come `x=1`: qui le due si scambiano di posto.
Inoltre da notare che scrivere $t\theta$ non vuole altro che dire "applicare θ al termine t".

Sostituzione

Consideriamo una sostituzione $\theta = \{f(z,z)/x, c/z\}$ con atomi e variabili rispettivamente $A = \{c, f\}, V = \{x, z\}$, prendiamo un termine formato da

$$p(f(x,y), x, g(z))\theta$$

e lo sostituiamo, per ciascuna delle sue parti, con il corrispettivo valore.
La nuova espressione sostituita sarà

$$p(f(f(z,z),y), f(z,z), g(c)))$$

Sostituzione composta

Date 2 sostituzioni $\theta = \{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$ e $\sigma = \{u_1/y_1, u_2/y_2, \dots, u_m/y_m\}$, la *sostituzione composta* $\theta \circ \sigma$ si ottiene da

$$\{t_1\sigma/x_1, t_2\sigma/x_2, \dots, t_n\sigma/x_n, u_1/y_1, u_2/y_2 \dots, u_m/y_m\}$$

eliminando gli elementi del tipo

- $t_j\sigma/x_j$ se $t_j\sigma = x_j$, siccome una sostituzione ben fatta non deve contenere termini che rimangono identici anche dopo la sostituzione;
- u_j/y_j se $y_j \in \{x_1, x_2, \dots, x_n\}$, siccome alcune y sono uguali a delle x.

Sostituzione composta

Componiamo $\theta = \{f(y)/x, z/y\}$ e $\sigma = \{a/x, b/y, y/z\}$ con atomi e vincoli rispettivamente $A = \{a, b, f\}, V = \{x, y, z\}$. La sostituzione equivale a

$$\theta \circ \sigma = \{f(b)/x, y/z\} \subset \{f(b)/x, y/y, a/x, b/y, y/z\}$$

dove le proprietà di eliminazione sono state applicate al set originario esteso, per renderlo ridotto.

Proprietà fondamentale della composizione di sostituzioni

Prese 2 sostituzioni θ e σ, con termine t qualsiasi, costruendo la composizione composta $\theta \circ \sigma$ e applicando la composizione composta $\theta\sigma$ a t, quello che otteniamo è sempre una sostituzione θ applicata a t che restituisce termine a cui applichiamo σ, o scritto concisamente

$$t(\theta \circ \sigma) = (t\theta)\sigma$$

questa proprietà è la fondamentale della composizione di sostituzioni.

Proprietà fondamentale della composizione di sostituzioni

Se $\theta = \{f(y)/x, z/y\}$ e $\sigma = \{a/x, b/y, y/z\}$ e $t = \{h(x, g(y), z\}$, con atomi e termini $A = \{a, b, f, g, h\}, V = \{x, y, z\}$, allora

$$t\theta = h(f(y), g(z), z) \qquad (t\theta)\sigma = h(f(b), g(y), y)$$

dove infatti $t(\theta \circ \sigma) = h(f(b), g(y), y)$.

Termini unificabili

Prendendo k termini e una sostituzione θ, il nostro obbiettivo è cercare di capire se esiste una sostituzione σ tale per cui

$$t_1\theta = t_2\theta = \dots = t_k\theta$$

ovvero utilizziamo le sostituzioni, per cercare di risolvere un problema abbastanza generale, che è quello di capire se esiste una sostituzione che rende tutti i termini uguali. Se la sostituzione esiste, allora i termini t_k vengono detti tra di loro **unificabili**: possono essere uniti con un unico termine.

Di questi *unificatori* possono esisterne diversi per dei termini, siccome unificatori utilizzano più variabili del necessario; di questi magari c'interessa soltanto quella più corta, per arrivare allo scopo prima. Esistono anche problemi di unificazione che non hanno soluzione.

Unificazione

Consideriamo l'insieme di termini $\{p(x), p(y)\}$ con atomo $A = \{p\}$ e variabili $V = \{x, y, z\}$, cercando una sostituzione che costruisca un termine applicato a $p(x)$, equivalente, applicabile a $p(y)$, che dia lo stesso risultato per entrambe.

Ne esistono almeno 3 di unificatori in questo esempio

$$\{x/y\} \qquad \{y/x\} \qquad \{z/x, z/y\}$$

per tutti e tre i casi quindi, il risultato sarà uguale tra i due p.
Il 3° unificatore ci piace meno degli altri:

- perché più lungo;
- aggiunge una variabile.

Most General Unifier (MGU)

Di nostro interesse, è quell'insieme di unificatori che vengono rappresentati da una sostituzione che prende il nome di Most General Unifier (MGU), rappresentati con $\mathtt{mgu}(S)$. Prendendo l'esempio sopra, i primi 2 unificatori sono esempio di MGU, mentre il terzo non lo è.

Dato un insieme di atomi e variabili, il nostro problema di unificazione lo possiamo scrivere come un insieme di uguaglianze che vogliamo rendere vere tutte insieme: dato un insieme di termini T costruito su insieme di atomi A e variabili V, un *problema di unificazione* (sintattico) è un insieme del tipo

$$\{l_1 \doteq r_1, l_2 \doteq r_2, \ldots, l_n \doteq r_n\}$$

dove $\doteq$ suppone un'eventuale uguaglianza sintattica se i 2 elementi lo sono effettivamente.
Se una singola sostituzione θ viene trovata, tale per cui $l_i\theta = r_i\theta$ per ogni $1 \leq i \leq n$, allora il problema viene detto risolubile e la sostituzione trovata è soluzione.

Funzioni per un problema di unificazione

Dato un problema di unificazione, quello che vorremmo costruire è un algoritmo che applicato a questo, ci generi un θ o s'interrompa senza generazione, se e soltanto se la sostituzione non può esistere. Per realizzare questo codice, vediamo prima un paio delle definizioni. Se S è un problema di unificazione, insieme di equazioni come nelle parentesi viste sopra, allora:

1. $\mathtt{vars}(S)$ è l'insieme delle variabili contenute nella parte sinistra e destra dell'equazione S, un insieme di coppie e termini;
2. se θ è una sostituzione, θS è un altro problema di unificazione che otteniamo applicando alla parte sinistra e destra di ogni equazione, la sostituzione θ; l'obbiettivo è quello di arrivare a una forma del problema originale, che ci interessa.

L'algoritmo di Gauss

Un esempio di algoritmo usatissimo nelle trasformazioni delle matrici è l'algoritmo di Gauss. A ogni passaggio dell'algoritmo, quello che altro non facciamo è creare un nuovo problema che possa essere risoluto, operando sulle righe, per poter arrivare alla soluzione finale ovvero la matrice ridotta a scala. L'algoritmo di Gauss è un esempio di θS.

Algoritmo di unificazione di Martelli e Montanari

Vedremo 2 algoritmi per risolvere problemi di sostituzione, dei quale il più efficiente/ottimo dal punto di vista computazionale, e anche il più noto nell'accademia italiana, quello di unificazione di Martelli e Montanari.

Partendo da un problema di unificazione S, il risultato che darà l'algoritmo è uno tra:

1. $\perp$ se non esiste nessuna sostituzione tale da rendere il problema risolto, ovvero non esiste sostituzione che renda tutte insieme vere, $l_i\sigma = r_i\sigma$;
2. se e soltanto se non viene prodotto il primo risultato, allora il problema di unificazione è risolubile e quello che ci viene restituito è un problema equivalente in cui nella parte sinistra dell'equazioni, ci sono soltanto variabili; la soluzione è la sostituzione costruibile mettendo prima variabile e poi termine $\{x_1 \doteq t_1, x_2 \doteq t_2, \dots, x_m \doteq t_m\}$.

Anziché costruire uno pseudo codice, si usa fare una *funzione non deterministica* andando a dire quindi, quanto vale il valore di una funzione deterministica, in questo caso **unify**, caso per caso ricordando della non mutua esclusione.

1. $\mathtt{unify}(G \cup \{t \doteq t\}) = \mathtt{unify}(G)$
   (caso `delete`)
   Il problema di unificazione è composto da un insieme di uguaglianze, unito all'uguaglianza $\{t \doteq t\}$. La **unify** in questo caso, ha come risultato l'applicazione $\mathtt{unify}(G)$, perché in fondo l'uguaglianza non serve a nulla: la togliamo.

2. $\mathtt{unify}(G \cup \{f(l_1, l_2, \dots, l_m) \doteq g(r_1, r_2, \dots, r_k)\}) = \perp$ se $f \neq g \lor m \neq k$
   (caso `conflict`)
   Se all'interno del problema abbiamo f applicato a dei termini, uguale a g applicato a dei termini, con $f \neq g$, allora il problema non è risolubile ($\perp$). Siccome la testa di un termine strutturato S non può mai essere una variabile, se abbiamo 2 termini strutturati che hanno la testa diversa, nessuna sostituzione potrà rendere questi uguali, perché f e g non saranno mai variabili (sostituzioni applicabili solo su variabili). Se il termine strutturato a sinistra ha 2 argomenti mentre quello a destra ne ha 4, nessuna sostituzione è in grado di ridurre questi argomenti.

3. $\mathtt{unify}(G \cup \{f(l_1, l_2, \dots, l_m) \doteq x\}) = \mathtt{unify}(G \cup \{x \doteq f(l_1, l_2, \dots, l_m)\})$ se $x \in V$
   (caso `swap`)
   Se siamo nella situazione di avere 1 degli elementi del nostro problema scritto come "termine strutturato uguale a variabile", allora il risultato della nostra **unify** l'otteniamo girando la parte destra con la sinistra, nei 2 termini. Siccome la forma risolta prevede le x a sinistra, questa operazione ci serve unicamente per raggiunge la soluzione.

4. $\mathtt{unify}(G \cup \{x \doteq t\}) = \mathtt{unify}(G\{t/x\} \cup \{x \doteq t\})$ se $x \in \mathtt{vars}(G)$ e $x \notin \mathtt{vars}(t)$
   (caso `eliminate`)
   Abbiamo un problema in cui da qualche parte vi è scritto che $\{x \doteq t\}$. Se così è, nessuno vieta di prendere la sostituzione che mappa x con t, applicandola alla parte G del nostro problema. Quello che otteniamo è un nuovo problema di unificazione in cui tutte le x sono state sostituite con t, tenendo conto dell'uguaglianza (non eliminandola, per ricordarci che abbiamo effettuato l'operazione). Questo lo possiamo fare fintanto che x non sia contenuto in t.

5. $\mathtt{unify}(G \cup \{f(l_1, l_2, \dots, l_m) \doteq f(r_1, r_2, \dots, r_m)\}) = \mathtt{unify}(G \cup \{l_1 \doteq r_1, l_2 \doteq r_2, \dots, l_m \doteq r_m\})$
   (caso `decompose`)
   2 termini strutturati hanno la stessa arità e la stessa testa, l'uguaglianza la otteniamo se sono uguali gli argomenti. Prendendo il problema originale, togliendo l'equazione su cui stiamo lavorando (perché non più presente), aggiungiamo tutte le equazioni che impongono che ogni argomento sia uguale per l, a sinistra e a destra. Stiamo imponendo che $l_m \doteq r_m$. Se riusciamo a ottenere una sostituzione che rende vero il problema ottenuto dalla riscrittura, allora è vero il problema di partenza. Preso un problema con una equazione, sostituiamo questa con altre che se soddisfacibili, soddisfano anche la precedente.

6. $\mathtt{unify}(G \cup \{x \doteq f(t_1, t_2, \dots, t_m)\}) = \perp$ se $x \in \mathtt{vars}(f(t_1, t_2, \dots, t_m))$
   (caso `check`)
   Se abbiamo un'equazione del tipo $\{x \doteq f(t_1, \dots, t_m)\}$ e ci accorgiamo che x è una variabile del termine

$f(t_1, \ldots, t_m)$, allora il risultato è $\perp$. Se siamo in una condizione del tipo $x = x + 2$, dal punto di vista sintattico quello che stiamo cercando è un termine che non esiste.

La regola #6 *occurs check* dal punto di vista computazionale, cambia completamente la classe di complessità di caso pessimo: ci basta questa per aumentare in modo significativo, quando gli alberi sono abbastanza grandi, il tempo necessario per risolvere il problema di unificazione. Siccome il caso #6 è inoltre abbastanza raro, alcune implementazioni lo tolgono, esplicitando espressamente la volontà.

Preso un problema di unificazione semplice, tipo $\{l \doteq r\}$, con l'algoritmo di Martelli e Montanari è calcolabile MGU dei due termini $\mathtt{mgu}\{l, r\}$. Inoltre, l'algoritmo deve essere in grado, per proprietà che vanno oltre all'equivalenza dei termini o *problemi di unificazione modulo teoria*, di implementare proprietà algebriche come la proprietà commutativa, qual'ora questo fosse necessario.

Per esempio, $\{f(a, b) \doteq f(x, y)\}$ accetta per la proprietà commutativa, 2 soluzioni che sono $\{a/x, b/y\}$ e $\{b/x, a/y\}$.

Questa implicazione detta che anche proprietà teoriche semplici possono trasformare problemi che ne fanno uso, in non risolubili o estremamente difficili da risolvere.

04/05/2023