Unifikácia

riešenie rovníc v algebre termov

Ján Šturc

Zima, 2010

Termy a substitúcie

Definícia (term):

1. Nech $t_0, ..., t_{n-1}$ sú termy a f je n-árny funkčný symbol, potom aj $f(t_0, ..., t_{n-1})$ je term.
2. Premenná je term.

Definícia (substitúcia):

Substitúciou nazývame funkciu $\sigma: V \to T$, kde V je množina premenných a T množina termov.

Substitúcie zapisujeme: $\sigma = [\{x_i \mapsto t_i\}_{i=1}]^n$

Aplikácia substitúcie:

Substitúciu σ na term t aplikujeme tak, že výsledok je term $s = t\sigma$, ktorý vznikne nahradením premenných vyskytujúcich sa v t aj σ príslušnými termami. Nahradenie sa vykoná súčasne pre všetky premenné naraz.

Usporiadanie a ekvivalencia na množine termov

Definícia: Hovoríme, že term t je aspoň tak všeobecný ako term s, píšeme $t \sqsupseteq s$, ak existuje substitúcia σ taká, že $s=t\sigma$.

Definícia: Hovoríme, že termy t a s sú ekvivalentné, píšeme $t \cong s$, ak $t \sqsupseteq s$ a $s \sqsupseteq t$.

Lema: Dva termy sú ekvivaletné, $t \cong s$ práve vtedy, keď sa odlišujú len pomenovanín premenných.

Substitúcie

Definícia: Hovoríme, že substitúcia σ je zložením substitúcii ρ a τ, píšeme $\sigma = \rho \circ \tau$, ak pre každý term t platí $t\sigma = (t\tau)\rho$.

Hoci substitúcie sú funkcie, skladanie substitúcií nie je skladanie funcií.

Definícia: Hovoríme, že substitúcia σ je aspoň tak všeobecná ako substitúcia τ, píšeme $\sigma \sqsupseteq \tau$, ak existuje substitúcia ρ taká, že $\sigma = \rho \circ \tau$.

Definícia: Dve substitúcie σ a τ sú ekvivalentné, píšeme $\sigma \simeq \tau$, ak $\sigma \sqsupseteq \tau$ a $\tau \sqsupseteq \sigma$.

Lema: Dve permutácie premenných (nie nutne tých istých) sú ekvivalentné substitúcie.

Pozn.: Prázdna (všade nedefinovaná) substitúcia, rôzne permutácie premenných sú ekvivalentné substitúcie.

Term matching

Úloha: Daný je term t a vzor term s. Úlohou je nájsť substitúciu ρ takú, že tρ = s.

Riešenie: Začneme prázdnou (všade nedefinovanou) substitúciou ρ aplikujeme rekurzívnu procedúru:

```
procedure match(u,v): boolean
if u is a variable then { if ρ(u) is undefined then { ρ(u):= v; return(true)}
                          else if ρ(u)=v then return(true)
                          else return(false) }
  else if u=f(u_0, … u_{k-1}) and v=f(v_0, … v_{k-1}) then
        { i:= 0; while i < k and match(u_i,v_i) do i:= i+1;
          if i = k return(true) else return(false) }
  else return(false)
end;
```

Ak procedúra vráti true, v ρ bude hľadaná substitúcia. Ak vráti false, riešenie neexistuje.

Unifikácia

1. Úloha: Daná je dvojica termov t a s. Úlohou je nájsť dvojicu najvšeobecnejších substitúcií τ a σ takých, že $t\tau = s\sigma$.
2. Obvykle sa v literatúre unifikáciou nazýva riešenie úlohy: Daná je dvojica termov t a s najdite najvšeobecnejšiu substitúciu σ takú, že $t\sigma = s\sigma$. Táto úloha je klasickou matematickou úlohou. Vo volnej algebre termov riešte rovnicu: $t = s$.
3. Matematici riešia rovnice aj v algebrách charaterizovaných nejakým systémom rovnosti, vtedy hovoríme o unifikácii mod E, kde E je množina rovností.
4. Úloha 1 sa dá modifikovať nasledovne: Nájdite dvojicu najvšeobecnejších substitúcií$\iota\sigma$, takých že $t\iota\sigma = s\sigma$. Túto úlohu nazývame slabou unifikáciou.

Tvrdenia

T1: Z riešenia úlohy 4 dostaneme riešenie úlohy 1 tak, že $\tau = \iota \circ \sigma$.

T2: Najvšeobecnejšie riešenie úlohy 4 dostaneme tak, že ι je premenovanie premenných v t tak, aby boli rôzne od premenných vyskytujúcich sa v s. Označíme $t' = t\iota$.
A riešime úlohu 2: $t'\sigma = s\sigma$

T3: Riešenie úlohy 1 podľa T1 a T2 je najvšeobecnejšie riešenie úlohy 1

Algoritmy unifikácie

Existujú dva prístupy:

- Manipulácia s množinou rovníc
  - Herbrand (1930)
  - Marteli Montanari (1982)
- Manipulácia so stromami resp. dagmi termov.
  - Robinson (1965)
  - Corbin Bidoit (1983)
  - Ružička Prívara (1989)

Vybrali sme len niekoľko reprezentatov z oboch tried algoritmov. V tejto prednáške uvedieme len dva ostatné doporučujeme na samostatné štúdium.

Herbrandova metóda

Herbrandova metóda predpokladá, že je daná množina E rovníc medzi termami.
Nech σ je prázdna substitúcia.
Algoritmus:

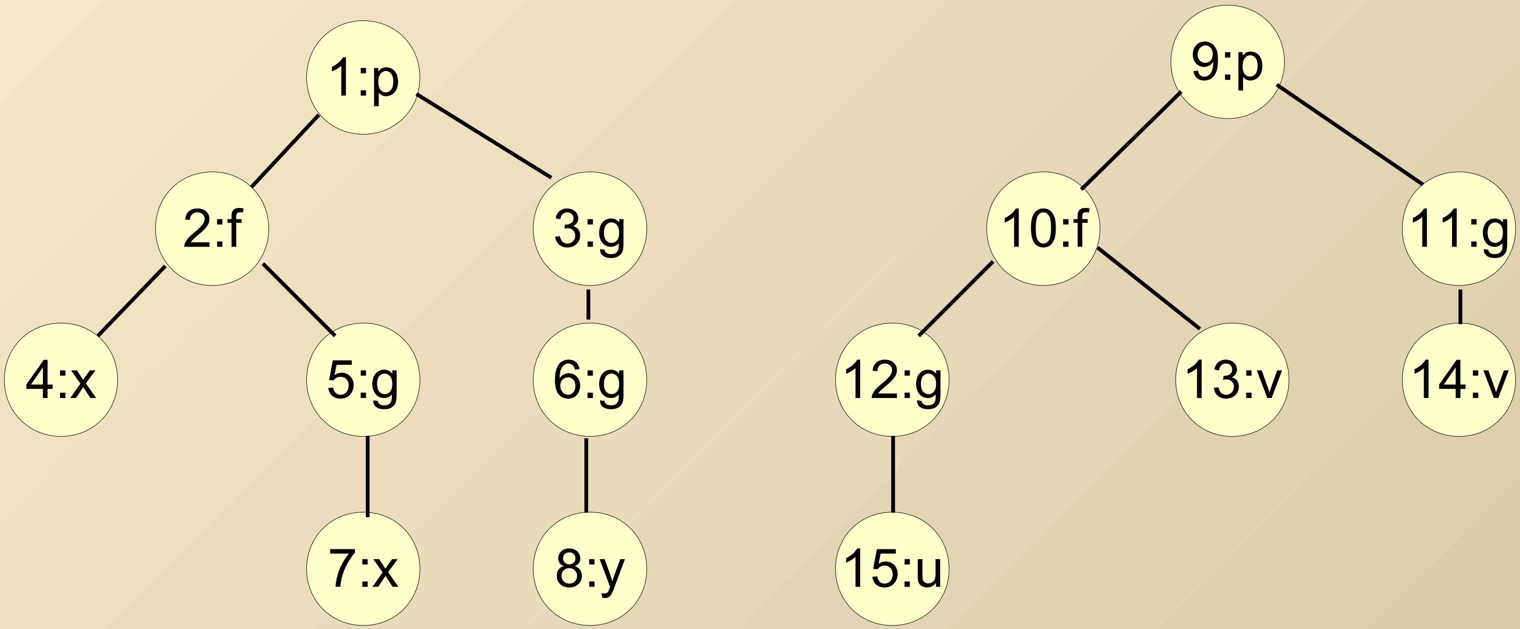
while $E \neq \emptyset$ **do**
{ select an equation $e \in E$; $E := E - \{e\}$;
  if e is $x = t$ or e is $t = x$ **then**
    { **if** x occurs in t **then** return(solution does not exist)
      else { $\sigma := \sigma \circ [x \mapsto t]$; $E := E[x \mapsto t]$ }
  else if e is of the form $f(s_0, \ldots, s_{n-1}) = f(t_0, \ldots, t_{n-1})$ **then**
      $E := E \cup \{s_0 = t_0, \ldots, s_{n-1} = t_{n-1}\}$
  else return(solution does not exist)
};

Algoritmus riešenia rovníc

Predpokladáme, že termy sú reprezentované stromami alebo dagmi.
Definujeme ekvivalenciu medzi vrcholmi nasledovne:

1. Korene oboch termov sú ekvivalentné.
2. Ak sú dva uzly ekvivalentné musia byť ekvivalentné aj dvojice ich synov v poradí.
3. Listy označené rovnakou premennou alebo tou istou konštatou sú ekvivalentné.
4. Z takto vypočítaných dvojíc ekvivalentných uzlov (hrán) vypočítame sy-metrický, reflexívny a tranzitívny uzáver. Množiny ekvivalentných uzlov.
5. K triedám ekvivalencie priradíme substitúcie:
   - Ak nejaká trieda obsahuje uzly s rôznym funkčnými symbolmi, riešenie neexistuje
   - Ak obsahuje iba premenné, vyberieme jednu, na ktorú ostatné premenujeme.
   - Ak obsahuje premenné aj uzly s funkčným symbolom, nahradíme všetky premenné podtermom prísluchajúcemu jednému uzlu s funkčným symbolom (je jedno ktorému).
6. Preveríme na zacyklenie (occur check).

Príklad 1

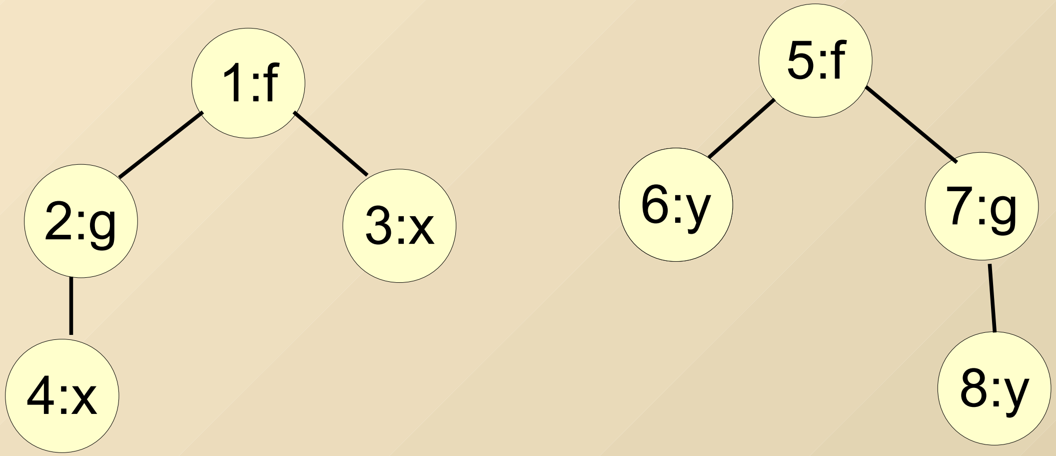


Triedy ekvivalencie:
{1,9}, {2,10}, {3,11},
{4,7,8,12}, {5,6,13,14}, {15}

Konštrukcia substitúcie (mgu):
$y \mapsto g(u)$, $x \mapsto g(u)$, $v \mapsto g(x)$.

Explicitné riešenie:
$x = g(u)$, $y = g(u)$, $v = g(g(u))$.

Príklad 2



Triedy ekvivalencie:
{1,5}, {2,6,8}, {3,4,7}

Konštrukcia mgu:
$y \mapsto g(x)$, $x \mapsto g(y)$.
Cyklus.

Riešenie neexistuje.
x = g(g(...g(u)...)), y = g(g(...g(u)...)) ?

Asymptotická zložitosť riešenia

- Traverzovaním oboch výrazov vytvoríme triedy ekvivalencie, že každý uzol je ekvivaletný iba sám so sebou. *O(n).*
- Nasleduje synchronné traverzovanie:
  - Find uzol n_1
  - Find uzol n_2
  - Union n_1 n_2
- Zložitosť union find úlohy je $O(n\alpha(n))$, kde α je inverzná Ackermanova funkcia.
- Kontrola zacyklenia je zložitosti *O(n)* napr. topologickým triedením.

Efektívna implementácia

Nie je potrebné implementovať program, ako sme ho prezentovali kvôli ľahšej analýze. Nepotrebujeme iniciálnu fázu. Môžeme začať priamo synchronným traverzovaním.

K tomu musíme viesť oddelene triedy ekvivalencie uzlov a triedy ekvivalencie premenných.

Ak operácia Find nie je úspešná, založíme novú triedu ekvivalencie. O triede ekvivalencie, okrem zoznamu uzlov vedieme aj funkčný symbol a premennú.

Ak sa funkčný symbol pri opätovnom vyhladaní nezhoduje, môžeme hneď skončiť. Ak trieda ekvivalencie obsahuje už nejakú premennú a pri opätovnom vyhladaní máme inú premennú, urobíme tieto premenné ekvivaletné.

Occur check sa modifikuje: Či niektorá premenná výslednej substitúcie nie je ekvivalentná premennej na ľavej strane.

Varovanie

Každý pokus o explicitné vyjadrenie výsledku môže viesť k exponenciálnej zložitosti výpočtu.

Príklad:
$f(x_0, x_1, x_2, \ldots, x_{n-1}) = f(g(x_1, x_1), g(x_2, x_2), \ldots, g(x_n, x_n))$

$x_0 \mapsto g(x_1, x_1)$
$x_1 \mapsto g(x_2, x_2)$
$x_2 \mapsto g(x_3, x_3)$
...
$x_{n-1} \mapsto g(x_n, x_n)$

Postupným spätným explicitným vyjadrením:
$x_{n-1} = g(x_n, x_n)$
$x_{n-2} = g(g(x_n, x_n), g(x_n, x_n))$
$x_{n-3} = g(g(g(x_n, x_n), g(x_n, x_n)), g(g(x_n, x_n), g(x_n, x_n))$
...
Pre dĺžku l_{i-1} i-1. výrazu platí: $l_{i-1} = 2 \times l_i + 4$
1, 6, 16, 36, 76, 156, 316, ...