COMPUTING LEAST GENERALIZATION BY ANTI-COMBINATION

MIKIO YOSHIDA

BBR INC., JAPAN

CHIAKI SAKAMA

WAKAYAMA UNIVERSITY, JAPAN

ILP 2014, Nancy, France, September 2014

SLD-RESOLUTION

Program

append([], x, x) <-
append([x|y], z, [x|w]) <- append(y,z,w)

Resolution

G_0: <- append([1,2], [3], s)

append([x|y], z, [x|w]) <- append(y,z,w)

mgu: σ_1={1/x, [2]/y, [3]/z, [1|w]/s}

G_1: <- append([2], [3], w)

append([x'|y'], z', [x'|w']) <- append(y',z',w')

mgu: σ_2={2/x', []/y', [3]/z', [2|w']/w}

G_2: <- append([], [3], w')

append([], x", x") <-

mgu: σ_3={[3]/x", [3]/w'}

G_3: □

Answer:

$\sigma_1\sigma_2\sigma_3$ | G_0 ={[1,2,3]/s}

Composition of mgu
(+ Restricting it to the variables of G_0)

COMBINATION

G_0: <- append([1,2], [3], s) append([x|y], z, [x|w]) <- append(y, z, w)

mgu: σ_1={1/x, [2]/y, [3]/z, [1|w]/s}

G_1: <- append(y, z, w) append([x'|y'], z', [x'|w']) <- append(y', z', w')

mgu: σ_2={ [x'|y']/y, z'/z, [x'|w']/w}

G_2: <- append(y', z', w') append([], x", x") <-

mgu: σ_3={[]/y', x''/z', x''/w'}

Answer: $(\sigma_1 + \sigma_2 + \sigma_3) \mid G_0$ ={[1,2,3]/s}

Combination of mgu
(+ Restricting it to the variables of G_0)

PROPERTIES OF COMBINATION

- Given a set of idempotent substitutions $\Theta=\{ \theta_1,\ldots,\theta_n \}$ (where $\theta_i\theta_i=\theta_i$), the **combination** $\theta_1+\ldots+\theta_n$ is defined as the **glb** of $(\Theta,\leqq)$ where $\theta'\leqq\theta$ if $\theta'=\theta\lambda$ for some substitution λ (**Eder 1985**).

- When $\theta_i = \{ t^i{}_1/ x^i{}_1,\ldots, t^i{}_{ki}/ x^i{}_{ki} \}$ $(1\leqq i\leqq n$, t_i : term, x_i : variable), $\theta_1+\ldots+\theta_n$ is computed as the mgu of two terms (**Chang&Lee 1973**):
  $E1= f(t^1{}_1,\ldots, t^1{}_{k1} ,\ldots, t^n{}_1,\ldots, t^n{}_{kn})$
  $E2= f(x^1{}_1,\ldots, x^1{}_{k1} ,\ldots, x^n{}_1,\ldots, x^n{}_{kn})$
  where f is a function symbol.

- Mgus used in combination are computed **independently** one another, and resolution is done by manipulation of substitutions without instantiating any rule (**Yamasaki et al., TCS 1986**).

OUR GOAL

- We apply the combination operation for computing **least generalization** (***lg***) of a set of terms.
- We show that ***lg*** of a set of terms is computed by an inverse substitution of combination, which we call **anti-combination**.
- The method enables us to compute ***lg*** **in parallel**.

LEAST GENERALIZATION OF TERMS

- Given a set of terms $\{ t_1,\ldots, t_n \}$,
  its **least generalization** (***lg***) is computed as

  $$\boldsymbol{lg}(t_1, \boldsymbol{lg}(t_2,\ldots, \boldsymbol{lg}(t_{n-1}, t_n) \ldots))$$

  where $\boldsymbol{lg}(t_i, t_j)$ is an ***lg*** of t_i and t_j, that is obtained by the **anti-unification** algorithm.
  (**Plotkin**, **Reynolds 1970**)

- The method is iterative and sequential.

- **Parallel algorithms** for anti-unification are realized by tree-representation of terms + implementation on shared memory parallel machine.
  (**Kuper et al., LICS'88**; **Delcher&Kasif, JAR 1992**)

INVERSE SUBSTITUTION

- ***Var*** : the set of variables, ***Term***: the set of terms
- A **substitution** is a mapping σ from ***Var*** to ***Term***.
  When $\sigma(x_i)=t_i$ for $i=1,\ldots,n$, it is written as
  $\sigma = \{ t_1 / x_1 ,\ldots, t_n / x_n \}$ (where $t_i \neq x_i$).
  The set $D(\sigma)=\{x_1,\ldots,x_n\}$ is the **domain** of σ.
- Let t be a term and σ an injective substitution where t and $D(\sigma)$ have no common variable. Then an **inverse substitution** σ^{-1}:***Term***→***Var*** is defined as
  - $t\sigma^{-1} = x$ if $(t/x) \in \sigma$
  - $f(t_1,\ldots,t_n)\,\sigma^{-1} = f(t_1\sigma^{-1},\ldots, t_n\sigma^{-1})$
    if $(f(t_1,\ldots,t_n)/x) \notin \sigma$ for some $x \in$ ***Var***
  - $y\sigma^{-1} = y$ if $(y/x) \notin \sigma$ for some $x \in$ ***Var***

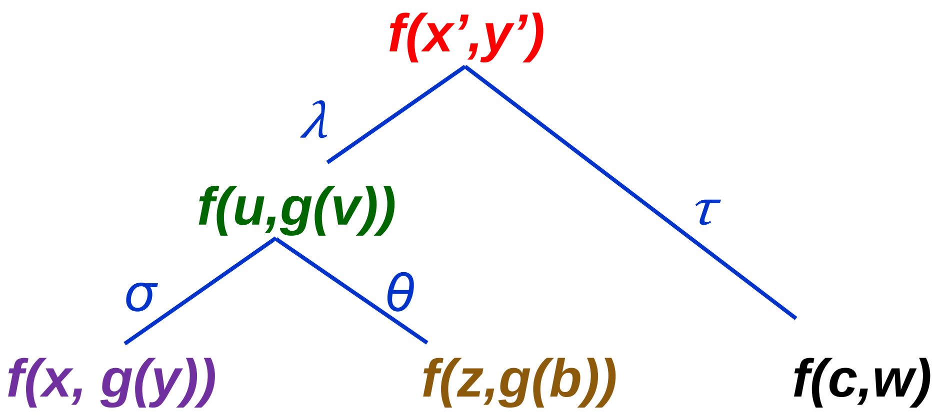
REMARK

- If t and $D(\sigma)$ have common variables, variables in t are renamed to make them different from those in $D(\sigma)$.
- If σ is not injective, a technique of (**N-Cheng&deWolf, 1997**) is applied to compute σ^{-1}. For instance, given $\sigma = \{ a/x, a/y \}$, $\sigma^{-1} = \{ (x/a,<1>), (y/a,<2>) \}$ meaning that a at position $<1>$ is mapped to x and a at position $<2>$ is mapped to y.
- Combining injective substitutions may produce a non-injective substitution. To compute its inverse substitution, incorporate information of substitutions from which each binding comes from. For instance, $\sigma_1 = \{ a/x \}$ and $\sigma_2 = \{ a/y \}$ produce $\sigma_1 + \sigma_2 = \{ a/x, a/y \}$. Then define $(\sigma_1 + \sigma_2)^{-1} = \{ (x/a,<\sigma_1>), (y/a,<\sigma_2>) \}$ which means a from σ_1 is mapped to x and a from σ_2 is mapped to y.

ANTI-COMBINATION

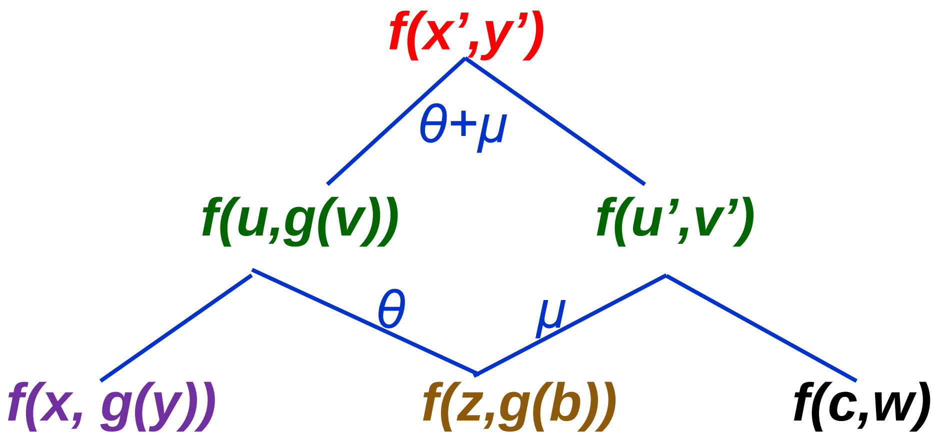
- Let $\sigma = \theta_1 + \ldots + \theta_n$ be a combination of $\{ \theta_1, \ldots, \theta_n \}$. Then the inverse substitution σ^{-1} is called an **anti-combination** of $\theta_1, \ldots, \theta_n$.
- For two terms t_1 *and* t_2 , let $s=\mathbf{lg}(\{t_1, t_2\})$. Then there is an idempotent substitution θ_i such that $t_i = s\theta_i$ (i=1,2). In this case, we write $\mathbf{slg}(t_i, \{t_1, t_2\}) = \theta_i$.
- Let $E=\{t_1,t_2,t_3\}$ be a set of terms, $\theta=\mathbf{slg}(t_2,\{t_1, t_2\})$ and $\mu=\mathbf{slg}(t_2,\{t_2, t_3\})$. Then $\mathbf{lg}(E)= t_2\, \sigma^{-1}$ *where* $\sigma \sim (\theta+\mu)$. ($\sigma \sim \theta$ *iff* $\sigma \leqq \theta$ and $\theta \leqq \sigma$; equivalent modulo renaming)
- The above result is extended to a set of terms containing *n*-terms ($n \geqq 3$).

EXAMPLE: ANTI-UNIFICATION



- Let **E**={ ***f(x,g(y)), f(z,g(b)), f(c,w)*** }.
- Then ***lg***({***f(x,g(y)), f(z,g(b))*** })=***f(u,g(v))*** with *σ*={***x/u***, ***y/v*** } and *θ*={***z/u***, ***b/v*** } where ***f(u,g(v))****σ*=***f(x,g(y))*** and ***f(u,g(v))****θ*=***f(z,g(b))***.
- Next ***lg***({***f(u,g(v)), f(c,w)***})=***f(x',y')*** with *λ*={***u/x', g(v)/y'*** } and *τ* ={***c/x', w/y'*** } where ***f(x',y')****λ*=***f(u,g(v))*** and ***f(x',y')****τ*=***f(c,w)***.
- *Then **lg**(**E**)=**f(x',y')** where **f(x',y')**λσ=**f(x,g(y))** with λσ={**x/x', g(y)/y'**} and **f(x',y')**λθ=**f(z,g(b))** with λθ={**z/x', g(b)/y'**}.*

EXAMPLE: ANTI-COMBINATION



- θ = ***slg***(***f(z,g(b))*** ,{***f(x,g(y)), f(z,g(b))***}) = {***z/u***, ***b/v*** }   and
  μ = ***slg***(***f(z,g(b))*** ,{***f(z,g(b)), f(c,w)***}) = { ***z/u'***, ***g(b)/v'*** }
- Then $\theta+\mu$ = { ***z/u***, ***b/v***, ***z/u'***, ***g(b)/v'*** } and
  $(\theta+\mu)^{-1}$ = { (***u/z***, <θ>), (***v/b***, <θ>), (***u'/z***, <μ>), (***v'/g(b)***, <μ>) }
- Applying it to ***f(z,g(b))***, ***lg***(***E***)=***f(u,v')*** (∼ ***f(x',y')***).