Mathematical Logic 2016

Lecture 15: Unification

Instructor: Ashutosh Gupta

TIFR, India

Compile date: 2016-10-01

Where are we and where are we going?

We have seen

- syntax and semantics of FOL
- sound and complete proof methods for FOL : tableaux and resolution

We will see computational aspects of FOL proving

- Unification

Skipped proof systems

We will skip FOL proof systems

- Hilbert
- Natural deduction

Example : γ-RULE is hard to apply

Not only γ-RULE has inherent non-determinism but also there are magic terms that close the proofs immediately.

Example 15.1

Consider

$\Sigma = \{\forall x_1, x_2, x_3, x_4.\ f(x_1, x_3, x_2) \not\approx f(g(x_2), j(x_4), h(x_3, a))\}$

Let us construct a resolution proof for the above Σ

1. $\{\forall x_1, x_2, x_3, x_4.\ f(x_1, x_3, x_2) \not\approx f(g(x_2), j(x_4), h(x_3, a))\}$
2. $\{\forall x_2, x_3, x_4.\ f(g(h(j(c), a)), x_3, x_2) \not\approx f(g(x_2), j(x_4), h(x_3, a))\}$
3. $\{\forall x_3, x_4.\ f(g(h(j(c), a)), x_3, h(j(c), a)) \not\approx f(g(h(j(c), a)), j(x_4), h(x_3, a))\}$
4. $\{\forall x_4.\ f(g(h(j(c), a)), j(c), h(j(c), a)) \not\approx f(g(h(j(c), a)), j(x_4), h(j(c), a))\}$
5. $\{f(g(h(j(c), a)), j(c), h(j(c), a)) \not\approx f(g(h(j(c), a)), j(c), h(j(c), a))\}$
6. $\{f(g(h(j(c), a)), j(c), h(j(c), a)) \approx f(g(h(j(c), a)), j(c), h(j(c), a))\}$
7. $\{\}$

We need a mechanism to auto detect substitutions such that terms with variables become equal

Topic 15.1

Unification

Unifier

Definition 15.1

For terms t and u, a substitution σ is a unifier of t and u if $t\sigma = u\sigma$.
We say t and u are unifiable if there is a unifier σ of t and u.

Example 15.2

Find a unifier σ of the following terms

- $x_4\sigma = f(x_1)\sigma$ — $\sigma = \{x_1 \mapsto c, x_4 \mapsto f(c)\}$
- $x_4\sigma = f(x_1)\sigma$ — $\sigma = \{x_1 \mapsto x_2, x_4 \mapsto f(x_2)\}$
- $g(x_1)\sigma = f(x_1)\sigma$ — *not unifiable*
- $x_1\sigma = f(x_1)\sigma$ — *not unifiable*

More general substitution

Definition 15.2

Let σ_1 and σ_2 be substitutions.
σ_1 is more general than σ_2 if there is a substitution τ such that $\sigma_2 = \sigma_1\tau$.

Example 15.3

- $\sigma_1 = \{x \mapsto f(y,z)\}$ *is more general than* $\sigma_2 = \{x \mapsto f(c, g(z))\}$ *because* $\sigma_2 = \sigma_1\{y \mapsto c, z \mapsto g(z)\}$
- $\sigma_1 = \{x \mapsto f(y,z)\}$ *is more general than* $\sigma_2 = \{x \mapsto f(z, z)\}$ *because* $\sigma_2 = \sigma_1\{y \mapsto z\}$

Exercise 15.1

If σ_1 is more general than σ_2 and σ_2 is more general than σ_3. Then, σ_1 is more general than σ_3.

Most general unifier (mgu)

Definition 15.3

Let t and u be terms with variables, and σ be a unifier of t and u.
σ is most general unifier(mgu) of u and t if it is more general than any other unifier.

Example 15.4

Consider terms $f(x, g(y))$ and $f(g(z), u)$
Consider the following three unifiers

1. $\sigma = \{x \mapsto g(z), u \mapsto g(y), z \mapsto z, y \mapsto y\}$
2. $\sigma = \{x \mapsto g(c), u \mapsto g(d), z \mapsto c, y \mapsto d\}$
3. $\sigma = \{x \mapsto g(z), u \mapsto g(z), z \mapsto z, y \mapsto z\}$

The first is more general unifier than the bottom two.
The second and third are incomparable(why?)

Is mgu unique? Does mgu always exist?

Uniqueness of mgu

Definition 15.4

A substitution σ is a renaming if $\sigma : \textbf{Vars} \to \textbf{Vars}$ and σ is one-to-one

Theorem 15.1

If σ_1 and σ_2 are mgus of u and t. Then there is a renaming τ s.t. $\sigma_1\tau = \sigma_2$.

Proof.

Since σ_1 is mgu, therefore there is a substitution $\hat{\sigma_1}$ s.t. $\sigma_2 = \sigma_1\hat{\sigma_1}$.
Since σ_2 is mgu, therefore there is a substitution $\hat{\sigma_2}$ s.t. $\sigma_1 = \sigma_2\hat{\sigma_2}$.
Therefore $\sigma_1 = \sigma_1\hat{\sigma_1}\hat{\sigma_2}$.

Wlog, for each $y \in \textbf{Vars}$, if for each $x \in \textbf{Vars}$, $y \not\in FV(x\sigma_1)$ then we assume $y\hat{\sigma_1} = y$

Uniqueness of mgu (contd.)

Proof(contd.)

claim: for each $y \in \mathbf{Vars}$, $y\hat{\sigma_1} \in \mathbf{Vars}$

Consider a variable x s.t. $y \in FV(x\sigma_1)$.

Suppose $y\hat{\sigma_1} = f(..)$. $x\sigma_1\hat{\sigma_1}$ will be longer than $x\sigma_1$.

Therefore, $x\sigma_1\hat{\sigma_1}\hat{\sigma_2}$ will be longer than $x\sigma_1$. Contradiction.

Suppose $y\hat{\sigma_1} = c$. $\hat{\sigma_2}$ will not be able to rename c back to y in $x\sigma_1$.

Therefore $y\hat{\sigma_1} \in \mathbf{Vars}$ is variable.

claim: for each $y_1 \neq y_2 \in \mathbf{Vars}$, $y_1\hat{\sigma_1} \neq y_2\hat{\sigma_1}$

Assume $y_1\hat{\sigma_1} = y_2\hat{\sigma_1}$.

$\hat{\sigma_2}$ will not be able to rename the variables back to distinct variables.(why?) □

Disagreement pair

Definition 15.5

For terms t and u, d_1 and d_2 are disagreement pair if

1. *d_1 and d_2 are subterms of t and u respectively,*
2. *the path to d_1 in t is same as and the path to d_2 in u, and*
3. *roots of d_1 and d_2 are different.*

Example 15.5

Consider terms $t = f(g(c), h(x, d))$ and $u = f(g(y), d)$
(Node labels are pairs of function symbols and argument number)

$$
\begin{tikzcd}
 & f,1 \arrow[dl] \arrow[dr] & & \\
g,1 \arrow[d] & & h,2 \arrow[dl] \arrow[dr] & \\
c,1 & x,1 & & d,2
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
 & f,1 \arrow[dl] \arrow[dr] & \\
g,1 \arrow[d] & & d,2 \\
y,1 & &
\end{tikzcd}
$$

Disagreement pairs: $h(x, d)$ and d *Disagreement pairs: c and y*

Robinsion's algorithm for computing mgu

Algorithm 15.1: $\textsc{mgu}(t, u \in T_{\mathbf{S}})$

$\sigma := \lambda x.x$;
while $t\sigma \neq u\sigma$ **do**
  choose disagreement pair d_1, d_2 in $t\sigma$ and $u\sigma$;
  if *both* d_1 *and* d_2 *are non-variables* **then** **return** *FAIL* ;
  if $d_1 \in$ **Vars then**
    $x := d_1$; $s := d_2$;
  else
    $x := d_2$; $s := d_1$;
  if $x \in FV(s)$ **then** **return** *FAIL* ;
  $\sigma := \sigma[x \mapsto s]$
return σ

If $\textsc{mgu}$ is sound and always terminates then mgus for unifiable terms always exists.

Exercise 15.2

Run the above algorithm with the following inputs.

- $\textsc{mgu}(f(x_1, x_3, x_2), f(g(x_2), j(x_4), h(x_3, a)))$
- $\textsc{mgu}(f(g(x), x), f(y, g(y)))$

Termination of MGU

Theorem 15.2

MGU *always terminates.*

Proof.

Total number of variables in $t\sigma$ and $u\sigma$ decreases in every iteration.(why?)

Since initially there were finite variables in t and u, MGU terminates. $\square$

Soundness of MGU

Theorem 15.3

MGU(t, u) *returns unifier* σ *iff* t *and* u *are unifiable. Furthermore,* σ *is a mgu.*

Proof.

Since MGU must terminate, if t and u are not unifiable then MGU must return FAIL.

Let us suppose t and u are unifiable and τ is a unifier of t and u.
claim: $\tau = \sigma\tau$ is the loop invariant of MGU.

base case:
Initially, σ is identity. Therefore, the invariant holds initially.

induction step:
We assume $\tau = \sigma\tau$ holds at the loop head.

Soundness of MGU(contd.)

Proof(contd.)

We show that the invariant holds after the loop body and FAIL is not returned.

claim: no FAIL at the first **if**

$t\sigma$ and $u\sigma$ are unifiable because $t\sigma\tau = t\tau = u\tau = u\sigma\tau$(why?).

One of d_1 and d_2 is a variable, otherwise $t\sigma$ and $u\sigma$ are not unifiable.

claim: no FAIL at the last **if**

Since $t\sigma\tau = u\sigma\tau$, $x\tau = s\tau$.

If x occurs in s then no unifier can make them equal(why?).

claim: $\sigma[x \mapsto s]\tau = \tau$

$x\sigma[x \mapsto s]\tau = s\tau = x\tau$.

Let $y \neq x$ then $y\sigma[x \mapsto s]\tau = y\sigma\tau = y\tau$.

Therefore, $\sigma[x \mapsto s]\tau = \tau$.

Due to the invariant $\tau = \sigma\tau$, σ is mgu at the termination. □

Multiple unification

Definition 15.6

Let $t_1, .., t_n$ be terms with variables.
A substitution σ is a unifier of $t_1, .., t_n$ if $t_1\sigma = .. = t_n\sigma$.
We say $t_1, .., t_n$ are unifiable if there is a unifier σ of them.

Exercise 15.3

Write an algorithm for computing multiple unifiers using the binary MGU.

Concurrent unification

Definition 15.7

Let $t_1, .., t_n$ and $u_1, .., u_n$ be terms with variables.
A substitution σ is a concurrent unifier of $t_1, .., t_n$ and $u_1, .., u_n$ if

$$t_1\sigma = u_1\sigma, \quad .., \quad t_n\sigma = u_n\sigma.$$

We say $t_1, .., t_n$ and $u_1, .., u_n$ are concurrently unifiable if there is a unifier σ for them.

Exercise 15.4

Write an algorithm for concurrent unifiers using the binary $\textsc{mgu}$.

Topic 15.2

Unification in proving

Unification in proving

Example 15.6

Consider again

$\Sigma = \{\forall x_1, x_2, x_3, x_4.\ f(x_1, x_3, x_2) \not\approx f(g(x_2), j(x_4), h(x_3, a))\}$

Given the above, one may ask

Are $f(x_1, x_3, x_2)$ *and* $f(g(x_2), j(x_4), h(x_3, a))$ *unifiable?*

Exercise 15.5

Run the unification algorithm on the above terms

Answer:

- $x_1 \mapsto g(h(j(x_4), a))$
- $x_2 \mapsto h(j(x_4), a)$
- $x_3 \mapsto j(x_4)$

We will integrate unification and resolution proof system.

The above instantiations are not magic anymore!

Changing γ-Rule

Unification gives the mechanism for finding the magic terms for γ-Rule.

Instead of instantiating γ for a closed term t, we may instantiate γ as $\gamma(x)$ for a fresh variable x. Later, find the appropriate closed term for x for closing the proofs.

Example 15.7

Consider once again
$\Sigma = \{\forall x_1, x_2, x_3, x_4.\ f(x_1, x_3, x_2) \not\approx f(g(x_2), j(x_4), h(x_3, a))\}$

Drop quantifier and introduce fresh variables z, u, v, and w.
$f(z, u, v) \not\approx f(g(v), j(w), h(u, a))$

Apply unification.

δ complication

Instantiation with free variables complicates the matter for the δ rule.

Since we have introduced a fresh variable instead of a term in γ rule, δ does not know which fresh symbol to choose.

Example 15.8

Consider the following satisfiable formula.

$\forall x.\exists y.\ x \not\approx y.$

$\exists y.\ x \not\approx y$ *(just drop $\forall$ and keep the variable name same)*

$x \not\approx c$ *(after applying delta rule)*

$c \not\approx c$ *(apply substitution $x \mapsto c$)*

$\perp$ ✗ *(but the formula is sat)*

Therefore we need dependent parameters, which are called skolem functions.

Skolem functions

We will need a supply of fresh symbols that may have non-zero arity.

Example 15.9

Consider the following satisfiable formula.

$\forall x.\exists y.\ x \not\approx y$

$\exists y.\ x \not\approx y$

$x \not\approx f(x)$

Introduce term that is dependent on x.
No unification can unify x and $f(x)$

Let us define an extension of signature that ensures a supply of new function symbols.

Definition 15.8

Let $\mathbf{S} = (\mathbf{F}, \mathbf{R})$ *be a signature. Let* **sko** *be a infinite countable set of function symbols disjoint from* $\mathbf{S}$. *Let* $\mathbf{S}^{\mathbf{sko}} = (\mathbf{F} \cup \mathbf{sko}, \mathbf{R})$.

We will present a proof system with unification and Skolem functions in the next lecture.

Topic 15.3

Algorithms for unification

Robinson is exponential

Robinson algorithm has worst case exponential run time.

Example 15.10

Consider unification of the following terms
$f(x_1, g(x_1, x_1), x_1, ...)$
$f(g(y_1, y_1), y_2, g(y_2, y_2), ...)$

The mgu:

- $x_1 \mapsto g(y_1, y_1)$
- $y_2 \mapsto g(g(y_1, y_1), g(y_1, y_1))$
- *.... (size of term keeps doubling)*

After discovery of a substitution $x \mapsto s$, Robinson checks if $x \in FV(s)$. Therefore, Robinson has worst case exponential time.

Martelli-Montanari algorithm

This algorithm is lazy in terms of applying occurs check

Algorithm 15.2: $\textsc{MM-Mgu}(t, u \in T_{\mathbf{S}})$

$\sigma := \lambda x.x$; $M = \{t \approx u\}$;
while *change in* M *or* σ **do**
  if $f(t_1, ...t_n) \approx f(u_1, ...u_n) \in M$ **then**
    $M := M \cup \{t_1 \approx u_1, ...t_n \approx u_n\} - \{f(t_1, ...t_n) \approx f(u_1, ...u_n)\}$;
  if $f(t_1, ...t_n) \approx g(u_1, ...u_n) \in M$ **then return** *FAIL* ;
  if $x \approx x \in M$ **then** $M := M - \{x \approx x\}$;
  if $x \approx t' \in M$ *or* $t' \approx x \in M$ **then**
    if $x \in FV(t')$ **then return** *FAIL* ;
    $\sigma := \sigma[x \mapsto t']$; $M := M\sigma$
return σ

https://pdfs.semanticscholar.org/3cc3/338b59855659ca77fb5392e2864239c0aa75.pdf

Topic 15.4

Problems

MGU

Exercise 15.6

Find mgu of the following terms

1. $f(g(x_1), h(x_2), x_4)$ *and* $f(g(k(x_2, x_3)), x_3, h(x_1))$
2. $f(x, y, z)$ *and* $f(y, z, x)$

Exercise 15.7

Let σ_1 *and* σ_2 *be the MGUs in the above unifications. Give unifiers* σ'_1 *and* σ'_2 *for the problems respectively such that they are not MGUs. Also give* τ_1 *and* τ_2 *such that*

1. $\sigma'_1 = \sigma_1\tau_1$
2. $\sigma'_2 = \sigma_2\tau_2$

Maximum and minimal mgus

Exercise 15.8

a. Give two maximum general substitutions *and two minimal general* substitutions*.*

b. Show that maximum general substitutions *are equivalent under renaming.*

End of Lecture 15