TECHNISCHE UNIVERSITÄT DRESDEN

Computational Logic Group

Hannes Strass

(based on slides by Michael Thielscher)

Faculty of Computer Science, Institute of Artificial Intelligence, Computational Logic Group

Unification

Lecture 2, 16th Oct 2023 // Foundations of Logic Programming, WS 2023/24

Previously . . .

Prolog Programs

- Prolog programs consist of **facts** and **rules**.
- We use Prolog by asking **queries** to programs.
- Answers to queries can be Boolean (yes/no) . . .
- . . . or given by variable assignments.
- Prolog programs are **declarative** (to a certain extent).

```
direct(frankfurt,san_francisco).
direct(frankfurt,chicago).
direct(san_francisco,honolulu).
direct(honolulu,maui).
```

```
connection(X, Y) :- direct(X, Y).
connection(X, Y) :- direct(X, Z),
                    connection(Z, Y).
```

```
| ?- connection(frankfurt, maui).
yes
```

```
| ?- connection(frankfurt, X).
X = san_francisco
```

Overview

Ranked Alphabets and Terms

Substitutions

Unifiers and Most General Unifiers

Martelli-Montanari Algorithm

The Need to Perform Unification

```
p(f(X),g(f(c),X)).

| ?- p(U,g(V,f(W))).

U = f(f(W)),
V = f(c)

| ?- p(U,g(c,f(W))).

no

| ?- p(U,g(V,U)).

U = f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f(f...
```

Ranked Alphabets and Terms

Ranked Alphabets and Term Universes

- A **variable** is a first-order predicate logic variable
- A **ranked alphabet** is a finite set Σ of symbols; to every symbol a natural number $n \geq 0$ (its **arity** or **rank**) is assigned
  ($\Sigma^{(n)}$ denotes the subset of Σ with symbols of arity n)
- Parentheses, commas
- For V a set of variables, F a ranked alphabet of **function symbols**:

  The **term universe** $TU_{F,V}$ (over F and V) is the smallest set with
  1. $V \subseteq TU_{F,V}$;
  2. if $f \in F^{(0)}$, then $f \in TU_{F,V}$;
  3. if $f \in F^{(n)}$ with $n \geq 1$ and $t_1, \ldots, t_n \in TU_{F,V}$, then $f(t_1, \ldots, t_n) \in TU_{F,V}$.

  The elements of $TU_{F,V}$ are called **terms**.

Ground Terms and Herbrand Universes

- $\mathit{Var}(t) \;:=\;$ set of variables in t
- t **ground** term $\;:\Longleftrightarrow\;$ $\mathit{Var}(t) = \emptyset$
- F ranked alphabet of function symbols:
  Herbrand universe HU_F (over F) $\;:\Longleftrightarrow\;$ $TU_{F,\emptyset}$
- s **sub-term** of t $\;:\Longleftrightarrow\;$ term s is sub-string of t (equivalently: sub-tree)

Substitutions

Substitutions

Definition

Let V be a set of variables, $X \subseteq V$ be finite, and F be a ranked alphabet.
A **substitution** is a function $\theta : X \to TU_{F,V}$ with $x \neq \theta(x)$ for every $x \in X$.

We use the notation $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ to express that

1. $X = \{x_1, \ldots, x_n\}$, and
2. $\theta(x_i) = t_i$ for every $x_i \in X$.

- **empty** substitution ε $:\Longleftrightarrow$ $n = 0$
- θ **ground** substitution $:\Longleftrightarrow$ $t_1, \ldots, t_n$ ground terms
- θ **pure variable** substitution $:\Longleftrightarrow$ $t_1, \ldots, t_n$ variables
- θ **renaming** $:\Longleftrightarrow$ $\{t_1, \ldots, t_n\} = \{x_1, \ldots, x_n\}$

Substitutions (2)

Consider a substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$.

$$
\begin{aligned}
Dom(\theta) &:= \{x_1, \ldots, x_n\} \\
Range(\theta) &:= \{t_1, \ldots, t_n\} \\
Ran(\theta) &:= Var(Range(\theta)) \\
Var(\theta) &:= Dom(\theta) \cup Ran(\theta) \\
\theta|_Y &:= \{\, y/t \mid y/t \in \theta \text{ and } y \in Y\} && \text{for every } Y \subseteq \{x_1, \ldots, x_n\}
\end{aligned}
$$

Applying Substitutions

Definition

Let t be a term and θ be a substitution.
The **application of θ to t** is the term $t\theta$ obtained as follows:

1. If $t = x$ is a variable, then $t\theta = x\theta := \begin{cases} \theta(x) & \text{if } x \in Dom(\theta), \\ x & \text{otherwise.} \end{cases}$
2. If $t = c \in \Sigma^{(0)}$ is a constant symbol, then $t\theta = c\theta := c$.
3. If $t = f(t_1, \ldots, t_n)$ for an $f \in \Sigma^{(n)}$, then $t\theta = f(t_1, \ldots, t_n)\theta := f(t_1\theta, \ldots, t_n\theta)$.

- t is an **instance** of s $:\Longleftrightarrow$ there is a substitution θ with $s\theta = t$
- s is **more general than** t $:\Longleftrightarrow$ t is an instance of s
- t is a **variant** of s $:\Longleftrightarrow$ there is a renaming θ with $s\theta = t$

Lemma 2.5

Term t is a variant of term s iff t is an instance of s and s is an instance of t.

Composition

Definition

Let θ and η be substitutions. The **composition** $\theta\eta$ is defined by setting

$$(\theta\eta)(x) := (x\theta)\eta$$

for each variable x.

Intuition: First apply θ, then apply η.

Lemma

Let $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, $\eta = \{y_1/s_1, \ldots, y_m/s_m\}$.
Then $\theta\eta$ can be constructed from the sequence

$$x_1/t_1\eta, \ldots, x_n/t_n\eta,\ y_1/s_1, \ldots, y_m/s_m$$

1. by removing all bindings $x_i/t_i\eta$ where $x_i = t_i\eta$
   and all bindings y_j/s_j where $y_j \in \{x_1, \ldots, x_n\}$, and
2. then forming a substitution from the resulting sequence.

Comparing Substitutions

Definition

Let θ and τ be substitutions.
θ is **more general than** τ $:\Longleftrightarrow$ $\tau = \theta\eta$ for some substitution η.

Examples

- $\theta = \{x/y\}$ is more general than $\tau = \{x/a, y/a\}$ (with $\eta = \{y/a\}$)
- $\theta = \{x/y\}$ is not more general than $\tau = \{x/a\}$
  (since for every η with $\tau = \theta\eta$:
  $x/a \in \{x/y\}\eta \implies y/a \in \eta \implies y \in Dom(\theta\eta) = Dom(\tau)$)
- θ is more general than θ for every θ, via $\theta = \theta\varepsilon$
- $\theta = \{x/y\}$ is more general than $\tau = \{y/x\}$ (with $\eta = \tau$),
  and τ is more general than θ (with $\eta = \theta$), but $\theta \neq \tau$.

Unifiers and Most General Unifiers

Unifiers

Definition

Let s and t be terms.

- Substitution θ is a **unifier** of terms s and t $\quad :\Longleftrightarrow \quad s\theta = t\theta$.
- Terms s and t are **unifiable** $\quad :\Longleftrightarrow \quad$ a unifier of s and t exists.
- Substitution θ is the **most general unifier** (**mgu**) of s and t $\quad :\Longleftrightarrow$
  θ is a unifier of s and t that is more general than all unifiers of s and t.

Definition

Let $s_1, \ldots, s_n, t_1, \ldots, t_n$ be terms, let $s_i \doteq t_i$ denote the (ordered) pair (s_i, t_i) and let $E = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$.

- Substitution θ is a **unifier** of E $\quad :\Longleftrightarrow \quad s_i\theta = t_i\theta$ for every $i \in [1, n]$.
- θ is the **most general unifier** (**mgu**) of E $\quad :\Longleftrightarrow$
  θ is a unifier of E that is more general than all unifiers of E.

Unifying Sets of Pairs of Terms

Definition

- Sets E and E' of pairs of terms are **equivalent**
  $:\Longleftrightarrow$ E and E' have the same set of unifiers.
- The set $\{x_1 \doteq t_1, \dots, x_n \doteq t_n\}$ of pairs is **solved**
  $:\Longleftrightarrow$ x_i, x_j pairwise distinct variables $(1 \leq i \neq j \leq n)$
  and no x_i occurs in t_j $(1 \leq i, j \leq n)$.

Lemma

If $E = \{x_1 \doteq t_1, \dots, x_n \doteq t_n\}$ is solved, then $\theta = \{x_1/t_1, \dots, x_n/t_n\}$ is an mgu of E.

Proof.

1. $x_i\theta = t_i = t_i\theta$
2. for every unifier η of E: $x_i\eta = t_i\eta = x_i\theta\eta$ for every $i \in [1, n]$ and $x\eta = x\theta\eta$ for every $x \notin \{x_1, \dots, x_n\}$; thus $\eta = \theta\eta$. $\square$

Quiz: Most General Unifiers

Quiz

Consider the following set of pairs:

$$E = \{\, f(a,y) \doteq x,\ g(y) \doteq g(z) \,\}$$

$\ldots$

Martelli-Montanari Algorithm

Martelli-Montanari Algorithm

Let E be a set of pairs of terms.

Martelli-Montanari Algorithm

As long as possible, nondeterministically choose a pair of a form below and perform the associated action:

(1)	$f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n)$	replace by $s_1 \doteq t_1, \ldots, s_n \doteq t_n$
(2)	$f(s_1, \ldots, s_n) \doteq g(t_1, \ldots, t_m)$ where $f \neq g$	halt with failure
(3)	$x \doteq x$	delete the pair
(4)	$t \doteq x$ where t is not a variable	replace by $x \doteq t$
(5)	$x \doteq t$ where $x \notin Var(t)$ and x occurs in some other pair	perform substitution $\{x/t\}$ on all other pairs
(6)	$x \doteq t$ where $x \in Var(t)$ and $x \neq t$	halt with failure

Terminate with success when no action can be performed.

(2) $\triangleq$ "clash"

(6) $\triangleq$ "occur check"-failure

Martelli-Montanari (Theorem)

Theorem

If the original set E has a unifier, then the algorithm successfully terminates and produces a solved set E' that is equivalent to E;
otherwise the algorithm terminates with failure.

Corollary: In case of success, E' determines an mgu of E.

Proof Steps

1. Prove that the algorithm terminates.
2. Prove that each action replaces the set of pairs by an equivalent one.
3. Prove that if the algorithm terminates successfully, then the final set of pairs is solved.
4. Prove that if the algorithm terminates with failure, then the set of pairs at the moment of failure does not have a unifier.

Relations

$$
\begin{aligned}
R \textbf{ relation} \text{ on a set } \mathcal{A} \;&:\Longleftrightarrow\; R \subseteq \mathcal{A} \times \mathcal{A} \\
R \textbf{ reflexive} \;&:\Longleftrightarrow\; (a,a) \in R \text{ for all } a \in \mathcal{A} \\
R \textbf{ irreflexive} \;&:\Longleftrightarrow\; (a,a) \notin R \text{ for all } a \in \mathcal{A} \\
R \textbf{ antisymmetric} \;&:\Longleftrightarrow\; (a,b) \in R \text{ and } (b,a) \in R \text{ implies } a = b \\
R \textbf{ transitive} \;&:\Longleftrightarrow\; (a,b) \in R \text{ and } (b,c) \in R \text{ implies } (a,c) \in R
\end{aligned}
$$

Well-founded Order(ing)s

- $(\mathcal{A}, \sqsubseteq)$ **partial order**
  $:\Longleftrightarrow \quad \sqsubseteq$ reflexive, antisymmetric, and transitive relation on $\mathcal{A}$
- $(\mathcal{A}, \sqsubset)$ **strict partial order**
  $:\Longleftrightarrow \quad \sqsubset$ irreflexive and transitive relation on $\mathcal{A}$
- strict partial order $(\mathcal{A}, \sqsubset)$ **well-founded**
  $:\Longleftrightarrow \quad$ there is no infinite descending chain
  $$\ldots \sqsubset a_2 \sqsubset a_1 \sqsubset a_0$$
  of elements $a_0, a_1, a_2, \ldots \in \mathcal{A}$

Examples

- $(\mathbb{N}, \leq)$, $(\mathbb{Z}, \leq)$, $(\mathcal{P}(\{1, 2, 3\}), \subseteq)$ partial orders;
- $(\mathbb{N}, <)$, $(\mathbb{Z}, <)$, $(\mathcal{P}(\{1, 2, 3\}), \subsetneq)$ strict partial orders;
- $(\mathbb{N}, <)$, $(\mathcal{P}(\{1, 2, 3\}), \subsetneq)$ are well-founded,
- whereas $(\mathbb{Z}, <)$ is not.

Lexicographic Ordering

The **lexicographic ordering** $\prec_n$ ($n \geq 1$) is defined inductively on the set $\mathbf{N}^n$ of n-tuples of natural numbers:

$$
\begin{aligned}
(a_1) \prec_1 (b_1) \quad &:\Longleftrightarrow \quad a_1 < b_1 \\
&\text{and} \\
(a_1, \ldots, a_{n+1}) \prec_{n+1} (b_1, \ldots, b_{n+1}) \quad &:\Longleftrightarrow \quad (a_1, \ldots, a_n) \prec_n (b_1, \ldots, b_n) \\
&\quad\ \text{or} \ \ (a_1, \ldots, a_n) = (b_1, \ldots, b_n) \text{ and } a_{n+1} < b_{n+1}
\end{aligned}
$$

Example

For $n = 3$, we have $(3, 12, 7) \prec_3 (4, 2, 1)$ and $(8, 4, 2) \prec_3 (8, 4, 3)$.

Theorem

For every $n \in \mathbf{N}$, the pair $(\mathbf{N}^n, \prec_n)$ is a well-founded strict partial order.

Proof Step 1 (1)

Proposition

The Martelli-Montanari Algorithm terminates.

Definition

Variable x is **solved in** E

$:\Longleftrightarrow \quad x \doteq t \in E$, and this is the only occurrence of x in E.

$$\begin{array}{rcl} uns(E) & := & \text{number of variables in } E \text{ that are unsolved} \\ lfun(E) & := & \text{number of occurrences of function symbols} \\ & & \text{in the first (left) components of pairs in } E \\ card(E) & := & \text{number of pairs in } E \end{array}$$

Example

Consider $E = \{f(x) \doteq f(y), y \doteq a\}$. Then $uns(E) = 2$, $lfun(E) = 1$, $card(E) = 2$.

Proof Step 1 (2)

Proposition

The Martelli-Montanari Algorithm terminates.

Proof.

Each successful action reduces $(uns(E), lfun(E), card(E))$ wrt. $\prec_3$.
For every $u, l, c \in \mathbb{N}$ the reduction is as follows:

$$
\begin{array}{llllll}
(1) & (u, l, c) & \succ_3 & (u - k, \underline{l - 1}, c + n - 1) & \text{for some } k \in [0, n] \\
(3) & (u, l, c) & \succ_3 & (u - k, l, \underline{c - 1}) & \text{for some } k \in \{0, 1\} \\
(4) & (u, l, c) & \succ_3 & (u - k_1, \underline{l - k_2}, c) & \text{for some } k_1 \in \{0, 1\} \text{ and } k_2 \geq 1 \\
(5) & (u, l, c) & \succ_3 & (\underline{u - 1}, l + k, c) & \text{for some } k \geq 1
\end{array}
$$

Termination is now a consequence of $(\mathbb{N}^3, \prec_3)$ being well-founded. □

Proof Step 2

Proposition

Each action replaces the set of pairs by an equivalent one.

Proof.

This is obviously true for actions (1), (3), and (4).
Regarding action (5), consider $E \cup \{x \doteq t\}$ and $E\{x/t\} \cup \{x \doteq t\}$. Then:

$$\begin{array}{ll} & \theta \text{ is a unifier of } E \cup \{x \doteq t\} \\ \text{iff} & \theta \text{ is a unifier of } E \quad \text{and} \quad x\theta = t\theta \\ \text{iff} & \theta \text{ is a unifier of } E\{x/t\} \quad \text{and} \quad x\theta = t\theta \\ \text{iff} & \theta \text{ is a unifier of } E\{x/t\} \cup \{x \doteq t\} \end{array}$$

$\square$

Proof Step 3

Proposition

If the algorithm successfully terminates, then the final set of pairs is solved.

Proof.

- If the algorithm successfully terminates, then the actions (1), (2), and (4) do not apply, so each pair in E is of the form $x \doteq t$ with x being a variable.
- Moreover, actions (3), (5), and (6) do not apply, so the variables in the first components of all pairs in E are pairwise disjoint and do not occur in the second component of a pair in E. $\square$

Proof Step 4

Proposition

If the algorithm terminates with failure, then the set of pairs at the moment of failure does not have a unifier.

Proof.

- If the failure results from action (2), then some

$$f(s_1, \ldots, s_n) \doteq g(t_1, \ldots, t_m)$$

  occurs in E (where $f \neq g$), and for no substitution θ we have $f(s_1, \ldots, s_n)\theta = g(t_1, \ldots, t_m)\theta$.
- If the failure results by action (6), then some $x \doteq t$ (where x is a proper subterm of t) occurs in E, and for no substitution θ we have $x\theta = t\theta$. □

Unifiers may be Exponential

$$E_1 = \{f(x_1) \doteq f(g(x_0, x_0))\}$$

$$\theta_1 = \{x_1/g(x_0, x_0)\}$$

$$E_2 = \{f(x_1, x_2) \doteq f(g(x_0, x_0), g(x_1, x_1))\}$$

$$\theta_2 = \theta_1 \cup \{x_2/g(g(x_0, x_0), g(x_0, x_0))\}$$

$$E_3 = \{f(x_1, x_2, x_3) \doteq f(g(x_0, x_0), g(x_1, x_1), g(x_2, x_2))\}$$

$$\theta_3 = \theta_2 \cup \{x_3/g(g(g(x_0, x_0), g(x_0, x_0)), g(g(x_0, x_0), g(x_0, x_0)))\}$$

$$\vdots$$

MM Algorithm without Occur Check

- In most PROLOG systems the occur check does not apply, for the sake of efficiency.
- As for the Martelli-Montanari algorithm this amounts to omitting the occur check in action (5) and to drop action (6).
- Then the algorithm terminates with success, e.g., for $\{x \doteq f(x)\}$, despite x and $f(x)$ not being unifiable.
- Moreover, the algorithm may not terminate at all:

$$
\begin{aligned}
& \{x \doteq f(x),\ y \doteq g(x)\} \\
\overset{(5)}{\rightsquigarrow}\ & \{x \doteq f(x),\ y \doteq g(f(x))\} \\
\overset{(5)}{\rightsquigarrow}\ & \{x \doteq f(x),\ y \doteq g(f(f(x)))\} \\
& \quad \vdots
\end{aligned}
$$

Conclusion

Summary

- A **substitution** replaces variables by terms, and is applied to terms.
- A **unifier** is a substitution that equates two terms when applied to them.
- The **Martelli-Montanari Algorithm** decides if a set of pairs of terms has a unifier and even outputs a (most general) unifier if one exists.
- The algorithm is **correct** (i.e., sound and complete) and **terminates**.

Suggested action points:

- Try out the Martelli-Montanari Algorithm on a few examples by hand.
- Verify your results using a Prolog system (try to turn the occur check on).
- Come up with examples how the different values for parameters k, k_1, and k_2 in proof step 1 could be realised.