Prolog

Wykład p.t.

Prolog

Podstawienia, unifikacja, algorytm unifikacji

Antoni Ligęza

`ligeza@agh.edu.pl`
`http://galaxy.uci.agh.edu.pl/~ligeza`

Wykorzystano materiały:
`http://www.swi-prolog.org/`

Substitutions

Substitution is an operation allowing to replace some variables occurring in a formula with terms. The goal of applying a substitution is to make a certain formula more specific so that it matches another formula. Typically, substitutions are applied in resolution theorem proving for unification of formulae. A substitution is defined as follows:

Definition 1 *A substitution σ is any finite mapping of variables into terms of the form*

$$\sigma{:}V \rightarrow TER.$$

Extending Substitutions over Terms and Formulae

Since substitutions are applied to more complex expressions, it is necessary to extend the definition of substitutions on terms and formulae. This is done in a straightforward way as follows:

Definition 2 *Any substitution σ ($\sigma\colon V \rightarrow TER$) is extended to operate on terms and formulae so that a finite mapping of the form*

$$\sigma\colon TER \cup FOR \rightarrow TER \cup FOR$$

satisfying the following conditions is induced:

- $\sigma(c) = c$ *for any* $c \in C$*;*
- $\sigma(X) \in TER$*, and* $\sigma(X) \neq X$ *for a certain finite number of variables only;*
- *if* $f(t_1, t_2, \ldots, t_n) \in TER$*, then*

  $$\sigma(f(t_1, t_2, \ldots, t_n)) = f(\sigma(t_1), \sigma(t_2), \ldots, \sigma(t_n));$$

- *if* $p(t_1, t_2, \ldots, t_n) \in ATOM$*, then*

  $$\sigma(p(t_1, t_2, \ldots, t_n)) = p(\sigma(t_1), \sigma(t_2), \ldots, \sigma(t_n));$$

- $\sigma(\Phi \diamond \Psi) = \sigma(\Phi) \diamond \sigma(\Psi)$ *for any two formulae* $\Phi, \Psi \in FOR$ *and for* $\diamond \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow\}$*;*

Instances and Ground Terms

A substitution σ is any finite mapping of variables into terms extended over terms and formulae in the above way. Any formula $\sigma(\Phi)$ resulting from application of substitution σ to the variables of Φ will be denoted as $\Phi\sigma$ and it will be called a *substitution instance* or simply an *instance* of Φ. If no variables occur in $\Phi\sigma$ (or any other formula or term), it will be called a *ground instance* (a *ground formula* or a *ground term*, respectively).

Note, that according to the above definition, substitutions in fact operate only on free variables (they change only free variables, i.e. the ones that are not quantified). For example, in resolution theorem proving all the quantifiers (formally) are removed, and the resulting formulae are quantifier-free; thus all the variables can be regarded as free variables, at least with regard to substitutions application.

Notation and Operation of Substitutions

Since substitutions operate in fact on a finite number of variables only, they can be conveniently denoted as sets of ordered pairs of variables and the terms to be substituted for them. Hence, any substitution σ can be presented as

$$\sigma = \{X_1/t_1, X_2/t_2, \ldots, X_n/t_n\},$$

where t_i is a term to be substituted for variable X_i , $i = 1, 2, \ldots, n$. If Φ is a formula (or term) and σ is a substitution, then $\Phi\sigma$ is the formula (or term) resulting from simultaneous replacing the variables of Φ with the appropriate terms of σ.

Composition of Substitutions

Since substitutions are mappings, a *composition* of substitutions is well defined. Note that, having two substitutions, say σ and θ, the composed substitution $\sigma\theta$ can be obtained from σ by *simultaneous* application of θ to all the terms of σ, deletion of any pairs of the form X/t where $t = X$ (identity substitutions), and enclosing all the pairs X/t of θ, such that σ does not substitute for (operate on) X [**?**, **?**].

Let $\sigma = \{X_1/t_1, X_2/t_2, \ldots, X_n/t_n\}$ and let $\theta = \{Y_1/s_1, Y_2/s_2, \ldots, Y_m/s_m\}$. The composition of the above substitutions is obtained from the set

$$\{X_1/t_1\theta, X_2/t_2\theta, \ldots, X_n/t_n\theta, Y_1/s_1, Y_2/s_2, \ldots, Y_m/s_m\}$$

by:

- removing all the pairs $X_i/t_i\theta$ where $X_i = t_i\theta$, and
- removing all the pairs Y_j/s_j where $Y_j \in \{X_1, X_2, \ldots, X_n\}$.

Example 1 *Consider the following substitutions* $\sigma = \{X/g(U), Y/f(Z), V/W, Z/c\}$ *and* $\theta = \{Z/f(U), W/V, U/b\}$. *The composition of them is defined as*

$$\sigma\theta = \{X/g(b), Y/f(f(U)), Z/c, W/V, U/b\}.$$

Renaming and Inverse Substitution

Substitutions are in general mappings, but not one-to-one mappings; hence, in general an inverse substitution for a given one may not exist. However, there exists a class of substitutions, the so-called *renaming substitutions*, such that an inverse substitution always exists provided that they are one-to-one mappings.

Definition 3 *Substitution λ is a renaming substitution iff it is off the form*

$$\theta = \{X_1/Y_1, X_2/Y_2, \ldots, X_n/Y_n\} \tag{1}$$

Moreover, it is a one-to-one mapping if $Y_i \neq Y_j$ for $i \neq j$, $i, j \in \{1, 2, \ldots, n\}$.

Assume λ is a renaming, one-to-one substitution . The inverse substitution for it is given by $\lambda^{-1} = \{Y_1/X_1, Y_2/X_2, \ldots, Y_n/X_n,\}$. The composition of a renaming substitution and the inverse one leads to an *empty substitution*, traditionally denoted with ϵ; we have $\lambda\lambda^{-1} = \epsilon$.

Some Properties

Let E denote an expression (formula or term), ϵ denote an empty substitution, and let λ be a one-to-one renaming substitution; σ and θ denote any substitution. The following properties are satisfied for any substitutions:

- $E(\sigma\theta) = (E\sigma)\theta$,
- $\sigma(\theta\gamma) = (\sigma\theta)\gamma$ (associativity),
- $E\epsilon = E$,
- $\epsilon\sigma = \sigma\epsilon = \sigma$.

Note that, in general, the composition of substitutions is not commutative.

Unification

Substitutions are applied to *unify* terms and formulae. Unification is a process of determining and applying a certain substitution to a set of expressions (terms or formulae) in order to make them identical. We have the following definition of unification.

Definition 4 *Let* $E_1, E_2, \ldots, E_n \in TER \cup FOR$ *are certain expressions. We shall say that expressions* $E_1, E_2, \ldots, E_n$ *are* unifiable *if and only if there exists a substitution* σ*, such that* $\{E_1, E_2, \ldots, E_n\}\sigma = \{E_1\sigma, E_2\sigma, \ldots, E_n\sigma\}$ *is a single-element set.*

Substitution σ *satisfying the above condition is called a* unifier *(or a* unifying substitution*) for expressions* $E_1, E_2, \ldots, E_n$.

The Most General Unifier

Note that if there exists a unifying substitution for some two or more expressions (terms or formulae), then there usually exists more than one such substitution. It is useful to define the so-called *most general unifier* (*mgu*, for short), which, roughly speaking, substitutes terms for variables only if it is necessary, leaving as much place for possible further substitutions, as possible. The most general unifier is defined as follows.

Definition 5 *A substitution σ is a* most general unifier *for a certain set of expressions if and only if, for any other unifier θ of this set of expressions, there exists a substitution λ, such that $\theta = \sigma\lambda$.*

The meaning of the above definition is obvious. Substitution θ is not a most general unifier, since it is a composition of some simpler substitution σ with an auxiliary substitution λ.

In general, for arbitrary expressions there may exist an infinite number of unifying substitutions. However, it can be proved that any two most general unifiers can differ only with respect to variable names. This is stated with the following theorem.

Twierdzenie 1 *Let θ_1 and θ_2 be two most general unifiers for a certain set of expressions. Then, there exists a one-to-one renaming substitution λ such that $\theta_1 = \theta_2\lambda$ and $\theta_2 = \theta_1\lambda^{-1}$.*

Example

As an example consider atomic formulae $p(X, f(Y))$ and $p(Z, f(Z))$. The following substitutions are all most general unifiers:

- $\theta = \{X/U, Y/U, Z/U\}$,
- $\theta_1 = \{Z/X, Y/X\}$,
- $\theta_2 = \{X/Y, Z/Y\}$,
- $\theta_3 = \{X/Z, Y/Z\}$.

All of the above unifiers are equivalent — each of them can be obtained from another one by applying a renaming substitution. For example, $\theta = \theta_1\lambda$ for $\lambda = \{X/U\}$; on the other hand obviously $\theta_1 = \theta\lambda^{-1}$.

Unification Algorithm – An Idea

It can be proved that if the analyzed expressions are terms or formulae, then there exists an algorithm for efficient generating the most general unifier, provided that there exists one; in the other case the algorithm terminates after finite number of steps . Thus, the unification problem is decidable.

The basic idea of the unification algorithm can be explained as a subsequent search through the structure of the expressions to be unified for inconsistent relative components and replacing one of them, hopefully being a variable, with the other.

In order to find inconsistent components it is useful to define the so-called disagreement set. Let $W \subseteq TER \cup FOR$ be a set of expressions to be unified. A *disagreement set* $D(W)$ for a nonempty set W is the set of terms obtained through parallel search of all the expressions of W (from left to right), which are different with respect to the first symbol. Hence, the set $D(W)$ specifies all the inconsistent relative elements met first during the search.

Unification Algorithm

Unification Algorithm

1. Set $i = 0$, $W_i = W$, $\theta_i = \epsilon$.
2. If W_i is a singleton, then stop; θ_i is the most general unifier for W.
3. Find $D(W_i)$.
4. If there are a variable $X \in D(W_i)$ and a term $t \in D(W_i)$, such that X does not occur in t, then proceed; otherwise stop — W is not unifiable.
5. Set $\theta_{i+1} = \theta\{X/t\}$, $W_{i+1} = W_i\{X/t\}$.
6. Set $i = i + 1$ and go to 2.

Example

Example 2 *Consider two atomic formulae* $p(X, f(X, Y), g(f(Y, X)))$ *and* $p(c, Z, g(Z))$. *The following steps illustrate the application of the unification algorithm to these atomic formulae.*

1. $i = 0$, $W_0 = \{p(X, f(X, Y), g(f(Y, X))), p(c, Z, g(Z))\}$, $\theta_0 = \{\}$.
2. $D(W_0) = \{X, c\}$.
3. $\theta_1 = \{X/c\}$, $W_1 = \{p(c, f(c, Y), g(f(Y, c))), p(c, Z, g(Z))\}$.
4. $D(W_1) = \{f(c, Y), Z\}$.
5. $\theta_2 = \{X/c\}\{Z/f(c, Y)\} = \{X/c, Z/f(c, Y)\}$,
   $W_2 = \{p(c, f(c, Y), g(f(Y, c))), p(c, f(c, Y), g(f(c, Y)))\}$.
6. $D(W_2) = \{Y, c\}$.
7. $\theta_3 = \{X/c, Z/f(c, Y)\}\{Y/c\} = \{X/c, Z/f(c, c), Y/c\}$,
   $W_3 = \{p(c, f(c, c), g(f(c, c))), p(c, f(c, c), g(f(c, c)))\}$.
8. *Stop; the most general unifier is* $\theta_3 = \{X/c, Z/f(c, c), Y/c\}$.

Properties of Unification Algorithm – Theorem

The unification algorithm has some important properties given by Theorem 2.

Twierdzenie 2 *If W is a finite set of unifiable expressions, then the algorithm always terminates at step 2 and it produces the most general unifier for W. Moreover, if the expressions of W are not unifiable, then the algorithm terminates at step 4.*

Unification Algorithm in Prolog

From: R. Bartak:

```
http://kti.mff.cuni.cz/~bartak/prolog/data_struct.
html

unify(A,B):-
   atomic(A),atomic(B),A=B.
unify(A,B):-
   var(A),A=B.            % without occurs check
unify(A,B):-
   nonvar(A),var(B),A=B.  % without occurs check
unify(A,B):-
   compound(A),compound(B),
   A=..[F|ArgsA],B=..[F|ArgsB],
   unify_args(ArgsA,ArgsB).

unify_args([A|TA],[B|TB]):-
   unify(A,B),
   unify_args(TA,TB).
unify_args([],[]).
```