Logic Exercises

Technical University of Munich
Chair for Logic and Verification

Prof. Tobias Nipkow
Kevin Kappelmann

SS 2021 | Exercise Sheet 8 | 04.06.2021

Exercise 8.1. **[Simultaneous substitution]**
Recall that $[t_1/x_1, \ldots, t_n/x_n]$ is the *simultaneous* substitution of $x_1, \ldots, x_n$ by $t_1, \ldots, t_n$.

1. Can we always express $[t_1/x_1, \ldots, t_n/x_n]$ as a series of one-variable substitutions?
2. Can we always summarise a series of one-variable substitutions to a single simultaneous substitution?

Exercise 8.2. **[Occurs check]**
What happens if one omits the occurs check in the unification algorithm? Find an example where the unification algorithm without occurs check diverges or returns the wrong result.

Exercise 8.3. **[Unifiable terms]**
Specify the most general unifiers for the following sets of terms, if one exists:

$$L_1 = \{f(x, y), f(h(a), x)\}$$
$$L_2 = \{f(x, y), f(h(x), x)\}$$
$$L_3 = \{f(x, b), f(h(y), z)\}$$
$$L_4 = \{f(x, x), f(h(y), y)\}$$

Exercise 8.4. **[Formulas without negation]**
Prove that every predicate logic formula that only contains $\wedge, \vee, \forall, \exists, \longrightarrow$ and atomic formulas is satisfiable. Is such a formula also valid?

Homework 8.1. **[Most general unifier]** (+)

Consider the unification problem $x \stackrel{?}{=} f(y)$. Without running the unification algorithm, prove that

1. $\sigma_1 = \{x \mapsto f(y)\}$ is a most general unifier.
2. $\sigma_2 = \{x \mapsto f(z), y \mapsto z\}$ is unifier, but not a most general unifier.

Homework 8.2. **[Unification]** (+)

Use the algorithm presented in the lecture to compute a most general unifier for the following set of formulas: $\{P(g(x), f(a)), P(y, x), P(g(f(z)), f(z))\}$

Homework 8.3. **[Untangling simultaneous substitution]** (++)

Recall Exercise 8.1. Demonstrate how to "untangle" a simultaneous substitution that has been obtained by consolidating one-variable substitutions back into one-variable substitutions.

Homework 8.4. **[Anti-Unification]** (+++)

A term t is a *generalisation* of a list of terms S if for each $s \in S$ there is a substitution σ_s such that $t\sigma_s = s$. A term t is a *most specific generalisation* (msg) of S if for any generalisation t' of S, there is a substitution $\sigma_{t'}$ such that $t'\sigma_{t'} = t$.

Give a recursive procedure that computes the msg of a finite list S. Apply your algorithm to the list $S \coloneqq \big[f(g(x), x, d, x), f(x, g(x), d, g(x)), f(h(c), h(c), d, h(c))\big]$ and prove that the returned msg is indeed an msg of S.

Hint: design an algorithm that operates recursively on the structure of terms.

Bonus: Prove that your algorithm always returns the msg.

> Nature will always maintain her rights and prevail in the end over any abstract reasoning whatsoever.
>
> — David Hume