Automated Deduction

Laura Kovács

TU Wien

Outline

Lifting (Robinson, 1965)(Bachmair & Ganzinger, 1990)

Lifting Lemma for BR in $\mathbb{BR}$:

Let C and D clauses with no shared variables. If:

$$\frac{\begin{array}{cc} C & D \\ \downarrow\sigma_1 & \downarrow\sigma_2 \\ C\sigma_1 & D\sigma_2 \end{array}}{C'} \ (\textit{ground}\ \text{BR})$$

then there exists a substitution σ sucht that:

$$\begin{array}{c} \dfrac{C \quad D}{C''} \ (\textit{non} - \textit{ground}\ \text{BR}) \\ \downarrow\sigma \\ C' = C''\sigma \end{array}$$

Similar lifting lemmas each inferences of $\mathbb{BR}$ and $\mathbb{S}\mathrm{up}$.

What should we lift?

- Ordering $\succ$;
- Selection function σ;
- Calculus $\mathbb{S}\mathrm{up}_{\succ,\sigma}$ (thanks to lifting lemmas).

Most importantly, for the lifting to work we should be able to solve equations $s = t$ between terms and between atoms. This can be done using most general unifiers.

Unifier

Unifier of expressions s_1 and s_2: a substitution θ such that $s_1\theta = s_2\theta$. In other words, a unifier is a solution to an "equation" $s_1 = s_2$. In a similar way we can define solutions to systems of equations $s_1 = s_1', \ldots, s_n = s_n'$. We call such solutions simultaneous unifiers of $s_1, \ldots, s_n$ and $s_1', \ldots, s_n'$.

(Most General) Unifiers

A solution θ to a set of equations E is said to be a most general solution if for every other solution σ there exists a substitution τ such that $\theta\tau = \sigma$. In a similar way can define a most general unifier.

Consider terms $f(x_1, g(x_1), x_2)$ and $f(y_1, y_2, y_2)$.
(Some of) their unifiers are
$\theta_1 = \{y_1 \mapsto x_1, y_2 \mapsto g(x_1), x_2 \mapsto g(x_1)\}$ and
$\theta_2 = \{y_1 \mapsto a, y_2 \mapsto g(a), x_2 \mapsto g(a), x_1 \mapsto a\}$:

$f(x_1, g(x_1), x_2)\theta_1 = f(x_1, g(x_1), g(x_1))$;
$f(y_1, y_2, y_2)\theta_1 = f(x_1, g(x_1), g(x_1))$;
$f(x_1, g(x_1), x_2)\theta_2 = f(a, g(a), g(a))$;
$f(y_1, y_2, y_2)\theta_2 = f(a, g(a), g(a))$.

But only θ_1 is most general.

Unification

Let E be a set of equations. An isolated equation in E is any equation $x = t$ in E such that x has exactly one occurrence in E.

input:
A finite set of equations E
(s, t denote terms, c, d constants, f, g function symbols, x variable)
output:
A solution to E or failure.
begin
while there exists a non-isolated equation $(s = t) \in E$
do
case (s, t) **of**
$(t, t) \Rightarrow$ Remove this equation from E
$(x, t) \Rightarrow$
if x occurs in t
then halt with failure
else replace x by t in all other equations of E
$(t, x) \Rightarrow$ replace this equation by $x = t$
and do the same as in the case (x, t)
$(c, d) \Rightarrow$ halt with failure
$(c, f(t_1, \ldots, t_n)) \Rightarrow$ halt with failure
$(f(t_1, \ldots, t_n), c) \Rightarrow$ halt with failure
$(f(s_1, \ldots, s_m), g(t_1, \ldots, t_n)) \Rightarrow$ halt with failure
$(f(s_1, \ldots, s_n), f(t_1, \ldots, t_n)) \Rightarrow$ replace this equation by the set
$s_1 = t_1, \ldots, s_n = t_n$
end
od
Now E has the form $\{x_1 = r_1, \ldots, x_l = r_l\}$ and every equation in it is isolated
return the substitution $\{x_1 \mapsto r_1, \ldots, x_l \mapsto r_l\}$
end

Examples

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

Properties

Theorem Suppose we run the unification algorithm on $s = t$. Then

- If s and t are unifiable, then the algorithms terminates and outputs a most general unifier of s and t.
- If s and t are not unifiable, then the algorithms terminates with failure.

Notation (slightly ambiguous):

- $mgu(s, t)$ for a most general unifier;
- $mgs(E)$ for a most general solution.

Exercise

Consider a trivial system of equations $\{\}$ or $\{a = a\}$.
What is the set of solutions to it?
What is the set of most general solutions to it?

Revisit: What should we lift?

- Ordering $\succ$;
- Selection function σ;
- Calculus $\mathbb{S}\mathrm{up}_{\succ,\sigma}$ (thanks to lifting lemmas).

Most importantly, for the lifting to work we use most general unifiers.

Knuth-Bendix Ordering (KBO), Ground Case (Recap)

Let us fix

- Signature Σ, it induces the term algebra $TA(\Sigma)$.
- Total ordering $\gg$ on Σ, called precedence relation;
- Weight function $w : \Sigma \to \mathbb{N}$.

Weight of a ground term t is

$$|g(t_1, \dots, t_n)| = w(g) + \sum_{i=1}^{n} |t_i|.$$

$g(t_1, \dots, t_n) \succ_{KB} h(s_1, \dots, s_m)$ if

1. $|g(t_1, \dots, t_n)| > |h(s_1, \dots, s_m)|$ (by weight) or
2. $|g(t_1, \dots, t_n)| = |h(s_1, \dots, s_m)|$ and one of the following holds:

   2.1 $g \gg h$ (by precedence) or

   2.2 $g = h$ and for some $1 \leq i \leq n$ we have $t_1 = s_1, \dots, t_{i-1} = s_{i-1}$ and $t_i \succ_{KB} s_i$ (lexicographically, i.e. left-to-right).

Note: Weight functions w are **not arbitrary functions** – need to be “compatible” with $\gg$.

Why? Compare for example a and $f(a)$ with arbitrary $\gg$ and w.

Weight Functions, Ground Case

A weight function $w : \Sigma \to \mathbb{N}$ is any function satisfying:

- $w(a) > 0$ for any constant $a \in \Sigma$;
- if $w(f) = 0$ for a unary function $f \in \Sigma$, then $f \gg g$ for all functions $g \in \Sigma$ with $f \neq g$.
  That is, f is the greatest element of Σ wrt $\gg$.

As a consequence, there is at most one unary function f with $w(f) = 0$.

Weight Functions, Non-Ground Case

A weight function $w : \Sigma \cup Vars \to \mathbb{N}$, with $Vars$ denoting the set of variables, is any function satisfying:

- $w(x) = v_0$ for all variables $x \in Vars$, where $v_0 > 0$;
- $w(a) \geq v_0$ for any constant $a \in \Sigma$;
- if $w(f) = 0$ for a unary function $f \in \Sigma$, then $f \gg g$ for all functions $g \in \Sigma$ with $f \neq g$.
  That is, f is the greatest element of Σ wrt $\gg$.

As a consequence, there is at most one unary function f with $w(f) = 0$.

Notation: Given a term s and variable x, we write $\#(x, s)$ to denote the number of occurences of x in s.

Knuth-Bendix Ordering (KBO), Non-Ground Case

Let us fix

- Signature Σ, it induces the term algebra $TA(\Sigma)$.
- Total ordering $\gg$ on Σ, called precedence relation;
- Weight function $w : \Sigma \cup Vars \rightarrow \mathbb{N}$.

Weight of a term t is

$$|g(t_1, \ldots, t_n)| = w(g) + \sum_{i=1}^{n} |t_i|.$$

$s \succ_{KB} t$ if

1. $\#(x, s) \geq \#(x, t)$ for all variables x and $|s| > |t|$ (by weight) or
2. $\#(x, s) \geq \#(x, t)$ for all variables x and $|s| = |t|$ and one of the following holds:
   - 2.1 $t = x$, $s = f^n(x)$ for some $n \geq 1$, or
   - 2.2 $s = g(t_1, \ldots, t_n)$, $t = h(s_1, \ldots, s_m)$ and $g \gg h$ (by precedence) or
   - 2.3 $s = g(t_1, \ldots, t_n)$, $t = g(s_1, \ldots, s_n)$ and for some $1 \leq i \leq n$ we have $t_1 = s_1, \ldots, t_{i-1} = s_{i-1}$ and $t_i \succ_{KB} s_i$ (lexicographically, i.e. left-to-right).

Selection Functions, Lifting

If for some grounding substitution θ, $L\theta$ is selected in $L\theta \vee C\theta$, then L is selected in $L \vee C$.

If the ground selection function is well-behaved, then its corresponding non-ground selection function lifted as above is also well-behaved.

Non-Ground Superposition, Lifting

Superposition:

$$\frac{\underline{l = r} \vee C \quad \underline{s[l'] = t} \vee D}{(s[r] = t \vee C \vee D)\theta} \text{ (Sup)}, \qquad \frac{\underline{l = r} \vee C \quad \underline{s[l'] \neq t} \vee D}{(s[r] \neq t \vee C \vee D)\theta} \text{ (Sup)},$$

where

1. θ is an mgu of l and l';
2. l' is not a variable;
3. $r\theta \not\succeq l\theta$;
4. $t\theta \not\succeq s[l']\theta$.

Observations:

- ordering is partial, hence conditions like $r\theta \not\succeq l\theta$;
- these conditions must be checked a posteriori, that is, after the rule has been applied.

Note, however, that $l \succ r$ implies $l\theta \succ r\theta$, so checking orderings a priory helps.

More rules

Equality Resolution:

$$\frac{\underline{s \neq s'} \vee C}{C\theta} \text{ (ER)},$$

where θ is an mgu of s and s'.

Equality Factoring:

$$\frac{\underline{l = r} \vee l' = r' \vee C}{(l = r \vee r \neq r' \vee C)\theta} \text{ (EF)},$$

where θ is an mgu of l and l', $r\theta \not\succeq l\theta$, $r'\theta \not\succeq l\theta$, and $r'\theta \not\succeq r\theta$.

Non-Ground Binary Resolution

- Binary resolution,

$$\frac{\underline{P} \vee C_1 \quad \underline{\neg P'} \vee C_2}{(C_1 \vee C_2)\theta} \text{ (BR).}$$

  where θ is the mgu of P and P'.

- Positive factoring,

$$\frac{\underline{P} \vee \underline{P'} \vee C}{(P \vee C)\theta} \text{ (Fact).}$$

  where θ is the mgu of P and P'.

- Negative factoring,

$$\frac{\underline{\neg P} \vee \underline{\neg P'} \vee C}{(\neg P \vee C)\theta} \text{ (Fact).}$$

  where θ is the mgu of P and P'.

Outline

Checking Redundancy

Suppose that the current search space S contains no redundant clauses. How can a redundant clause appear in the inference process?
Only when a new clause (a child of the selected clause and possibly other clauses) is added.
Classification of redundancy checks:

- The child is redundant;
- The child makes one of the clauses in the search space redundant.

Subsumption, Non-Ground Case

A clause C subsumes any clause D if $C\theta \subseteq D$ for some substitution θ.

Subsumption and redundancy: If a clause set S contains two different clauses C and D and C subsumes D, then D is redundant in S (and can be removed).

Demodulation, Non-Ground Case

$$\frac{l = r \quad \cancel{L[l'] \vee D}}{L[r\theta] \vee D} \text{ (Dem)},$$

where $l\theta = l'$, $l\theta \succ r\theta$, and $(L[l'] \vee D)\theta \succ (l\theta \succ r\theta)$.

Easier to understand:

$$\frac{l = r \quad \cancel{L[l\theta] \vee D}}{L[r\theta] \vee D} \text{ (Dem)},$$

where $l\theta \succ r\theta$, and $(L[l'] \vee D)\theta \succ (l\theta \succ r\theta)$.

Generating and Simplifying Inferences

An inference

$$\frac{C_1 \quad \ldots \quad C_n}{C}\ .$$

is called simplifying if at least one premise C_i becomes redundant after the addition of the conclusion C to the search space. We then say that C_i is simplified into C.
A non-simplifying inference is called generating.

Note. The property of being simplifying is undecidable. So is the property of being redundant. So in practice we employ sufficient conditions for simplifying inferences and for redundancy.

Redundancy Checking

Redundancy-checking occurs upon addition of a new child C. It works as follows

- Retention test: check if C is redundant.
- Forward simplification: check if C can be simplified using a simplifying inference.
- Backward simplification: check if C simplifies or makes redundant an old clause.

Examples

Retention test:

- tautology-check;
- subsumption.

Simplification:

- demodulation (forward and backward);
- subsumption resolution (forward and backward):

$$\frac{A \vee C \quad \cancel{\neg B \vee D}}{D} \text{ (Subs)}, \qquad \text{or} \qquad \frac{\neg A \vee C \quad \cancel{B \vee D}}{D} \text{ (Subs)},$$

such that for some substitution θ we have $A\theta \vee C\theta \subseteq B \vee D$.

Some redundancy criteria are expensive

- Tautology-checking is based on congruence closure.
- Subsumption and subsumption resolution are NP-complete.

Outline

Unification and Lifting

Saturation and Superposition in Practice

Summary of First-Order Theorem proving

Revisiting the example about a commutative group