Transforming First-Order Formulas to Sets of Clauses,

Hans de Nivelle, Marc Bezem

First-Order Logic

Terms are defined as follows:

- If f is a function symbol with arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Atoms are defined as follows:

- If p is a predicate symbol with arity n, and $t_1, \ldots, t_n$ are terms, $p(t_1, \ldots, t_n)$ is an atom.

Note: We have no formal distinction between variables and constants. When a 0-arity function symbol is bound, it is a variable. Otherwise, it is a constant.

First-Order Logic

The set of formulas is recursively defined as follows:

- $\perp$ and $\top$ are formulas,
- every atom is a formula,
- if F is a formula, then $\neg F$ is a formula,
- If F and G are formulas, then
  $F \vee G$, $F \wedge G$, $F \rightarrow G$, $F \leftrightarrow G$ are formulas.
- If F is a formula, and v is a variable, then $\forall v\ F$ and $\exists v\ F$ are formulas.

First-Order Logic

Definition: Let F be a first-order formula, v be a variable. We call v free in F if there is an occurrence of v in F which is not in the scope of a $\forall v$ or $\exists v$.

Definition: Two formulas F and G are called α-variants of each other, if they differ only in bound variables.

Definition: Let F be a first-order formula, v be a variable, t be a term. We define capture avoiding substitution $F[v := t]$ (somewhat informally) as follows:

- As long as there is a free occurrence of v in F, which is in the scope of a quantifier $\forall x$ or $\exists x$, s.t. x occurs in t, replace F by an α-variant that uses another variable y that does not occur in t.
- After that, simply replace all free occurrences of v by t.

Theorem Proving Procedure:

Resolution is a refutation-based procedure.

That means that resolution can only establish unsatisfiability.

Suppose we want to test whether $A_1, \ldots, A_p \vdash B$.

Then we try to refute $A_1, \ldots, A_p, \neg B$.

Before we can use resolution, we have to replace $A_1, \ldots, A_p, \neg B$ by clauses.

This is done through a series of transformations that we now will describe:

Negation Normal Form

Definition A formula is in *negation normal form* if it has no occurrences of $\rightarrow$ and $\leftrightarrow$, and $\neg$ is applied only on atoms.

Algorithm for NNF

The first parameter is the polarity, either 0 or 1. the second parameter is the formula.

- For an atom A,
  $\mathrm{NNF}(0, A) = \neg A$,
  $\mathrm{NNF}(1, A) = A$,

- $\mathrm{NNF}(0, \top) = \bot$,
  $\mathrm{NNF}(1, \top) = \top$,

- $\mathrm{NNF}(0, \bot) = \top$,
  $\mathrm{NNF}(1, \bot) = \bot$,

Algorithm for NNF (2)

- $\text{NNF}(0, P \vee Q) = \text{NNF}(0, P) \wedge \text{NNF}(0, Q)$,
  $\text{NNF}(1, P \vee Q) = \text{NNF}(1, P) \vee \text{NNF}(1, Q)$,

- $\text{NNF}(0, P \wedge Q) = \text{NNF}(0, P) \vee \text{NNF}(0, Q)$,
  $\text{NNF}(1, P \wedge Q) = \text{NNF}(1, P) \wedge \text{NNF}(1, Q)$,

- $\text{NNF}(0, P \leftrightarrow Q) =$
  $(\ \text{NNF}(1, P) \wedge \text{NNF}(0, Q)\ ) \vee (\ \text{NNF}(0, P) \wedge \text{NNF}(1, Q)\ )$,
  $\text{NNF}(1, P \leftrightarrow Q) =$
  $(\ \text{NNF}(0, P) \vee \text{NNF}(1, Q)\ ) \wedge (\ \text{NNF}(1, P) \vee \text{NNF}(0, Q)\ )$,

Algorithm for NNF (3)

- $\mathrm{NNF}(0, \exists x\ P) = \forall x\ \mathrm{NNF}(0, P)$,
  $\mathrm{NNF}(1, \exists x\ P) = \exists x\ \mathrm{NNF}(1, P)$,

- $\mathrm{NNF}(0, \forall x\ P) = \exists x\ \mathrm{NNF}(0, P)$,
  $\mathrm{NNF}(1, \forall x\ P) = \forall x\ \mathrm{NNF}(1, P)$,

Skolemization

Skolemization replaces existential quantifiers by function symbols.

If we have a formula of form $F_1 = \forall x\ F[\ \exists y\ P(x,y)\ ]$,

then it is replaced by $F_2 = \forall x\ F[\ P(x, f(x))\ ]$, where f is function symbol that occurs nowhere else.

It is easy to show that $F_2 \vdash F_1$, but it does not hold that $F_1 \vdash F_2$.

Skolemization

It can be shown that if F_1 has a model, then F_2 has a model. A (very sketchy) argument is as follows:

Let $(D, [\ ])$ be the interpretation that makes F_1 true. Here D is the domain.

For every $d \in D$, we define the value $f(d)$ as follows:

- If there is an $e \in D$, s.t. $(d, e) \in [P]$, then possibly there exists more than one e, s.t. $(d, e) \in [P]$ holds. Choose (in some mysterious way) one of those e, and make it the value of $f(d)$.
- If there is no $e \in D$, s.t. $(d, e) \in [P]$ holds, then give $f(d)$ an aribrary value. (Surprise!, here we are not using AC, because we can reuse the same element all the time)

(A real proof takes 5 or more pages. If you want to avoid to use of axiom of choice, even longer)

Skolemization

We first define $\mathrm{SKOL}(F)$ as $\mathrm{SKOL}(\{\}, F)$. Then:

- For a literal L, $\quad \mathrm{SKOL}(V, L) = L$,
- $\mathrm{SKOL}(V, \top) = \top$,
- $\mathrm{SKOL}(V, \bot) = \bot$,
- $\mathrm{SKOL}(V, P \vee Q) = \mathrm{SKOL}(V, P) \vee \mathrm{SKOL}(V, Q)$,
- $\mathrm{SKOL}(V, P \wedge Q) = \mathrm{SKOL}(V, P) \wedge \mathrm{SKOL}(V, Q)$,
- $\mathrm{SKOL}(V, \forall x\ P) = \forall x\ \mathrm{SKOL}(V \cup \{x\}, P)$,
- $\mathrm{SKOL}(V, \exists y\ P)$ is defined as $\mathrm{SKOL}(V, P[y := f(x_1, \ldots, x_n)])$.

  Here $x_1, \ldots, x_n$ is an enumeration of the variables that are free in P and which occur in V.

  f is a function symbol of arity n that occurs nowhere else.

Factoring of Disjunctions

We first introduce some notation:

Definition:

$$\bigwedge\{\ \} = \perp,$$
$$\bigwedge\{F\} = F,$$
$$\bigwedge\{F_1, \ldots, F_n\} = F_1 \wedge \cdots \wedge F_n, \text{ if } n > 2.$$

Definition: Let S_1 and S_2 be a sequences of formulas: The product $S_1 \times S_2$ is defined as:

$$\bigwedge_{s_1 \in S_1 \ s_2 \in S_2} \big(s_1 \vee s_2\big)$$

Factoring of Conjunctions

- For a literal L, $\mathrm{FACT}(L) = (L)$,
- $\mathrm{FACT}(\top) = (\ )$,
- $\mathrm{FACT}(\bot) = (\bot)$,
- $\mathrm{FACT}(P \wedge Q) = \mathrm{FACT}(P) \cdot \mathrm{FACT}(Q)$,
  (the dot $\cdot$ means sequence concatenation)
- $\mathrm{FACT}(P \vee Q) = \Pi\ (\ \mathrm{FACT}(P),\ \mathrm{FACT}(Q)\ )$,
- $\mathrm{FACT}(\forall x\ P)$ is defined as follows: Write
  $\mathrm{FACT}(P) = (F_1, \ldots, F_n)$. Then

$$\mathrm{FACT}(\forall x\ P) = (\forall x\ F_1, \ldots, \forall x\ F_n).$$

Final Step:

At this point, we have a sequence of formulas, which is in NNF, contains no $\top$, no $\wedge$, and no $\exists$. Each formula F in the sequence is replaced by $\mathrm{CLS}(F)$, which is defined as follows:

- For a literal L, $\quad \mathrm{CLS}(L) = [L]$,
- $\mathrm{CLS}(\bot) = [\,]$.
- $\mathrm{CLS}(P \vee Q) = \mathrm{CLS}(P) \cup \mathrm{CLS}(Q)$,
- $\mathrm{CLS}(\forall x\ P) = \mathrm{CLS}(\ P[x := X]\ )$, where X is an implicit variable that occurs nowhere else.

Subformula Replacement

Subformula replacement is based on a function $R(V, F)$ that decides whether the formula F really needs to be replaced. In case it replaces F, it needs the variables V in order to determine the quantified variables that F depends on.

- $\mathrm{REPL}(V, \bot) = \bot, \quad \mathrm{REPL}(V, \top) = \top,$
- For a literal L, $\mathrm{REPL}(V, L) = L,$
- For a formula of form $\neg F$,
  $\mathrm{REPL}(V, \neg P) = R(V, \neg\, \mathrm{REPL}(V, P)),$
- For a formula of form $P \times Q$,
  $\mathrm{REPL}(V, P \times Q) = R(V, \mathrm{REPL}(V, P) \times \mathrm{REPL}(V, Q)).$
  Here $\times$ is one of $\wedge,\ \vee,\ \rightarrow,\ \leftrightarrow$.
- For a formula of form $\forall\exists v\ P$,
  $\mathrm{REPL}(\forall\exists v\ P) = R(V, \forall\exists v\ \mathrm{REPL}(V \cup \{v\}, P)).$

The function $R(V, F)$ decides whether F needs to be replaced. It has access to a global variable Δ, into which the definitions are inserted.

Later, Δ has to be added to the set of formulas.

If F is a formula that is not replaced, then $R(V, F) = F$ and Δ is not modified.

Otherwise, $\mathrm{REPL}(V, F) = p(x_1, \ldots, x_n)$. Here $x_1, \ldots, x_n$ is an enumeration of the variables that are free in F and which occur in V.

p is an atom symbol of arity n that occurs nowhere else.

To Δ, the formula

$$\forall x_1 \cdots x_n \; p(x_1, \ldots, x_n) \leftrightarrow F$$

is added.