CNF Transformation

Universal closure of a formula A is a formula

$$(\forall x_1) \ldots (\forall x_n) A,$$

denoted by $\forall A$, where $x_1, \ldots, x_n$ are all free variables of A.

CNF transformation transforms a closed formula F into a set of clauses $C_1, \ldots, C_n$ such that F is satisfiable if and only if so is the set of formulas $\forall C_1, \ldots, \forall C_n$.

Example

Suppose we want to prove (establish validity of)

$$(\exists y)(\forall x)p(x,y) \rightarrow (\forall x)(\exists y)p(x,y).$$

It is valid if and only if its negation

$$\neg((\exists y)(\forall x)p(x,y) \rightarrow (\forall x)(\exists y)p(x,y))$$

is unsatisfiable.

The transformation of this formula to CNF gives us two clauses:

$$p(x,a)$$
$$\neg p(b,y).$$

Example

How can we check unsatisfiability of

$$(\forall x)p(x, a)$$
$$(\forall y)\neg p(b, y)?$$

- Since we have $(\forall x)p(x, a)$, we also have $p(b, a)$;
- Since we have $(\forall y)\neg p(b, y)$, we also have $\neg p(b, a)$;
- $p(b, a)$ and $p(b, a)$ are unsatisfiable (e.g., by resolution).

Ideas

Note that we established unsatisfiability by

- ▷ Substituting terms for variables, e.g. b for x in $p(x, a)$;
- ▷ Using propositional resolution.

Are these two ingredients sufficient to have a complete procedure?

- A substitution θ is a mapping from variables to terms such that the set $\{x \mid \theta(x) \neq x\}$ is finite.
- This set is called the domain of θ.
- Notation: $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$, where $x_1, \ldots, x_n$ are pairwise different variables, denotes the substitution θ such that

$$\theta(x) = \begin{cases} t_i & \text{if } x = x_i; \\ x & \text{if } x \notin \{x_1, \ldots, x_n\}. \end{cases}$$

- Application of this substitution to an expression E: simultaneous replacement of x_i by t_i.
- Application of a substitution θ to E is denoted by $E\theta$.
- Since substitutions are functions, we can define their composition (writen $\sigma\tau$ instead of $\tau \circ \sigma$). Note that we have $E(\sigma\tau) = (E\sigma)\tau$.

Exercise

Suppose we have two substitutions

$$\{x_1 \mapsto s_1, \ldots, x_m \mapsto s_m\} \text{ and}$$

$$\{y_1 \mapsto t_1, \ldots, y_n \mapsto t_n\}.$$

How can we write their composition using the same notation?

Instances, Ground

An instance of an expression (that is term, atom, literal, or clause) E is obtained by applying a substitution to E. Examples:

- some instances of the term $f(x, a, g(x))$ are:
  $f(x, a, g(x))$,
  $f(y, a, g(y))$,
  $f(a, a, g(a))$,
  $f(g(b), a, g(g(b)))$;
- but the term $f(b, a, g(c))$ is not an instance of this term.

Ground instance: instance with no variables.

For a set of clauses S denote by S^* the set of ground instances of clauses in S.

Theorem Let S be a set of clauses. The following conditions are equivalent.

1. S is unsatisfiable;
2. S^* is unsatisfiable;

Note that by compactness the last condition is equivalent to

3. there exists a finite unsatisfiable set of ground instances of clauses in S.

The theorem reduces the problem of checking inconsistency of sets of arbitrary clauses to checking inconsistency of sets of ground clauses . . .

the only problem is that S^* can be infinite even if S is finite.

Lifting

Lifting is a technique for proving completeness theorems in the following way:

1. Prove completeness of the system for a set of ground clauses;
2. Lift the proof to the non-ground case.

Lifting, Example

Consider two (non-ground) clauses $p(x, a) \vee q_1(x)$ and $\neg p(y, z) \vee q_2(y, z)$. If the signature contains function symbols, then both clauses have infinite sets of instances:

$$\{p(r, a) \vee q_1(r) \quad | \quad r \text{ is ground}\}$$
$$\{\neg p(s, t) \vee q_2(s, t) \quad | \quad s, t \text{ are ground}\}$$

We can resolve such instances if and only if $r = s$ and $t = a$. Then we can apply the following inference

$$\frac{p(s, a) \vee q_1(s) \quad \neg p(s, a) \vee q_2(s, a)}{q_1(s) \vee q_2(s, a)} \text{ (BR)}$$

But there is an infinite number of such inferences.

Lifting, Idea

The idea is to represent an infinite number of ground inferences of the form

$$\frac{p(s,a) \vee q_1(s) \quad \neg p(s,a) \vee q_2(s,a)}{q_1(s) \vee q_2(s,a)} \text{ (BR)}$$

by a single non-ground inference

$$\frac{p(x,a) \vee q_1(x) \quad \neg p(y,z) \vee q_2(y,z)}{q_1(y) \vee q_2(y,a)} \text{ (BR)}$$

Is this always possible?

Yes!

$$\frac{p(x,a) \vee q_1(x) \quad \neg p(y,z) \vee q_2(y,z)}{q_1(y) \vee q_2(y,a)} \text{ (BR)}$$

Note that the substitution $\{x \mapsto y, z \mapsto a\}$ is a solution of the "equation" $p(x,a) = p(y,z)$.

What should we lift?

- Selection function σ.
- Calculus $\mathbb{BR}_\sigma$.
- Ordering $\succ$, if we use an ordered resolution.

Most importantly, for the lifting to work we should be able to solve equations $s \simeq t$ between terms and between atoms.

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.

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

`input`:

A finite set of equations E

`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

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

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

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

Occurs check

- The check "x occurs in t" is called an occurs check.
- In Prolog, the predicate = implements unification without occurs check.
- There is also a predicate (and a command) for unification with occurs check.

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):

- $\mathrm{mgu}(s, t)$ 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?

Properties

Theorem Let C be a clause and E a set of equations. Then

$$\{D \in C^* \mid \exists\theta(C\theta = D \text{ and } \theta \text{ is a solution to } E)\} = (Cmgs(E))^*.$$

Binary Resolution System, Non-Ground Case

Binary resolution is the following inference rule:

$$\frac{\underline{A} \vee C \quad \underline{\neg B} \vee D}{(C \vee D)\mathrm{mgu}(A, B)}\ (\mathsf{BR}),$$

Factoring is the following inference rule:

$$\frac{\underline{A} \vee \underline{B} \vee C}{(A \vee C)\mathrm{mgu}(A, B)}\ (\mathsf{Fact}),$$

Completeness

$\mathbb{BR}$ is sound and complete, that is, if a set of clauses is unsatisfiable, then one can derive an empty clause from this set.

Soundness is evident since the conclusion of any inference rule is a logical consequence of its premises.

Completeness can be proved using completeness of propositional resolution and lifting.

Ordered resolution?

Binary resolution with arbitrary selection is incomplete.

To define ordered resolution one has to define ordering for non-ground clauses in a way so that they also work for their ground instances.

A problem

Is the following set of clauses unsatisfiable

$$p(x, a)$$
$$\neg p(b, x)?$$

A problem

Is the following set of clauses unsatisfiable

$$p(x, a)$$
$$\neg p(b, x)?$$

Yes, since clauses denote their universal closures:

$$(\forall x) p(x, a)$$
$$(\forall x) \neg p(b, x).$$

A problem

Is the following set of clauses unsatisfiable

$$p(x, a)$$
$$\neg p(b, x)?$$

Yes, since clauses denote their universal closures:

$$(\forall x) p(x, a)$$
$$(\forall x) \neg p(b, x).$$

But no rule of the resolution system is applicable to these clauses.

Renaming away

The domain of a substitution θ is the set of variables $\{x \mid \theta(x) \neq x\}$ is finite.

The range of θ is the set of terms $\{x\theta \mid x\theta \neq x\}$.

A substitution θ is called renaming if (three equivalent characterisations)

- the domain of θ coincides with its range.
- θ has an inverse σ (that is, $\theta \circ \sigma = \sigma \circ \theta = \{\}$).
- there exists an n such that $\theta^n = \{\}$.

A variant of a term (atom, literal, clause) t is any term obtained from t by appying a renaming.

Hidden rule: renaming away

Renaming E_1 away from E_2: replace E_1 by its variant E_1' so that E_1' and E_2 have no common variables.

Before applying resolution to two clauses C_1 and C_2 we should always rename C_1 away from C_2.

Renaming is sometimes called standardising apart (especially in the logic programming literature).

Example

(1)	$\neg p(x) \vee \neg q(y)$	input	
(2)	$\neg p(x) \vee q(y)$	input	
(3)	$p(x) \vee \neg q(y)$	input	
(4)	$p(x) \vee q(y)$	input	
(5)	$\neg p(x) \vee \neg p(y)$	BR	(1,2)
(6)	$\neg p(x)$	Fact	(5)
(7)	$p(x) \vee p(y)$	BR	(3,4)
(8)	$p(x)$	Fact	(7)
(9)	$\square$	BR	(6,8)

Term algebra $\mathrm{TA}(\Sigma)$ of signature Σ:

- ▷ Domain: the set of all ground terms of Σ.
- ▷ Interpretation of any function symbol f or constant c is defined as follows::

$$f_{\mathrm{TA}(\Sigma)}(t_1, \ldots, t_n) \stackrel{\mathrm{def}}{=} f(t_1, \ldots, t_n);$$
$$c_{\mathrm{TA}(\Sigma)} \stackrel{\mathrm{def}}{=} c.$$

Theorem In a term algebra every ground term is interpreted by itself:

$$t_{\mathrm{TA}(\Sigma)} = t.$$

Herbrand interpretation

- ▷ Herbrand interpretation of a signature Σ: any structure M of Σ extending $\mathrm{TA}(\Sigma)$.
- ▷ Herbrand model of a set of clauses in a signature Σ: any model of this set that is a Herbrand interpretation of the signature Σ.

Theorem Every satisfiable set of clauses of a signature Σ has a Herbrand model.

Herbrand interpretation

Given a Herbrand interpretation M, consider the following set of ground atoms:

$$\{A \mid A \text{ is a ground atom and } M \models A\}.$$

Two Herbrand interpretations are equal if and only if their sets of true ground atoms are equal.

Therefore, we can define a Herbrand interpretation of a signature Σ as a set of ground atoms in this signature.