Logic and calculus: Resolution for propositional logic

Logic and Calculus

Logic: formal language for expressions

- *syntax*: "spelling rules" for expressions
- *semantics*: meaning of expressions (logical consequence $\models$)
- *calculus*: set of given formulae and *syntactic* rules for manipulation of formulae to perform proofs.
  - *derivation*: $\phi \vdash \rho$ (ϕ, ρ formulae)

Calculus

- *axioms*: given formulae, elementary tautologies and contradictions which cannot be derived within the calculus
- *inference rules*: allow to derive new formulae from given formulae
- *derivation* $\phi \vdash \psi$: a sequence of inference rule applications starting with formula ϕ and ending in formula ψ

$\models$ and $\vdash$ should coincide

- *Soundness*: $\phi \vdash \rho$ implies $\phi \models \rho$
- *Completeness*: $\phi \models \rho$ implies $\phi \vdash \rho$

Inference Rules

- An inference rule has the form

$$\boxed{\frac{F_1, \ldots, F_n}{F}}$$

  - where the fomulae $F_1, \ldots, F_n$ are the *premises*
  - and the formula F is the conclusion.
  - Given a set of formulae X, if the premises are given (i.e. contained in X) then the conclusion is added to X.
  - *A derivation step* $F_1, \ldots, F_n \vdash F$ is the application of an inference rule.
  - *A derivation* is sequence of derivation steps with conclusion taken as premises for next step
- Rule application usually is nondeterministic.
- Set of derivations can be represented by *derivation trees*

Example: propositional logic

- *Syntax*: propositional symbols (a.k.a. propositional variables) and connectives
- *Semantics* $\models$: Truth tables and assignment of truth values to propositional symbols
- *Calculus* $\vdash$: Natural deduction

    Inference rules: Introduction and elimination rules for connectives

    Derivations: trees obtained by using rules and axioms.

$\models$ (Soundness and) Completeness theorem for propositional logic:

- $\phi \vdash \rho$ iff $\phi \models \rho$

Resolution

Robinson 1965

- inference rule that can be easily implemented.
- used as execution mechanism of (constraint) logic programming
- uses clausal normal form (and unification for FOL)

Resolution calculus for propositional logic

The idea is to work by contradiction, i.e. to use *refutation*: Theory united with negated consequence must be unsatisfiable.

We want to prove that $\phi \models \psi$. We have already seen the refutation theorem:

$$\phi \models \psi \text{ iff } \not\models \phi \wedge \neg\psi$$

Then we can proceed as follows

- Step 1. Transform all the fomulae in (equivalente formulae which are in) conjunctive normal form (i. e. in clauses).
- Step 2. Apply the resolution inference rule.
- Step 3. Stop when we derive the empty clause (i.e. the elementary contradiction i.e. *false*).

Resolution inference rule for PL

Axiom

empty clause (i.e. the elementary contradiction)

Resolution

$$\frac{R \vee A \qquad R' \vee \neg A}{R \vee R'}$$

In the above rule R and R' are disjunction of literals, A is an atomic formula. $R \vee R'$ is called resolvent. The empty clause is usually represented by $\Box$.

Theorem: Resolution is sound and complete for propositional logic.

An example

Suppose we want to prove that B is a logical consequence of $F = \{A \rightarrow B, A\}$, i.e. that $F \models B$ holds. We proceed as follows:

1. We first transform formulae in F into clausal form: $F' = \{\neg A \vee B, A\}$.
2. Then we add the negation of the conclusion that we want to prove: $F'' = \{\neg A \vee B, A, \neg B\}$.
3. We apply resolution

$$\dfrac{\dfrac{\neg A \vee B \qquad A}{\neg B \qquad B}}{\Box}$$

4. We have obtained the empty clause, which means that F'' is inconsistent and therefore that $F \models B$.

Note that this process is much more easy to automatize and to implement than natural deduction: less rules, less non-determinism, moreover (as we will see) the seacrg can be driven by the formula we want to prove.