UNIVERSITÉ CÔTE D'AZUR

Logic for AI

Master 1 Informatique

Andrea G. B. Tettamanzi
Laboratoire I3S – Pôle SPARKS
andrea.tettamanzi@univ-cotedazur.fr

Unit 7

Resolution

Agenda

- Resolution for Predicate Logic

Introduction

- In previous unit, we introduced the notion of unification
- We shall now use unification to extend resolution to (first-order/Herbrand) predicate logic
- The most important part of applying the resolution principle is finding a literal in a clause that is complementary to a literal in another clause
- Since clauses may contain variables, we must *unify* those literals

Motivating Example (from Unit 6)

Clause 1: $P(x) \vee Q(x)$

Clause 2: $\neg P(f(x)) \vee R(x)$

$\Downarrow$

Clause 1': $P(f(a)) \vee Q(f(a))$

Clause 2': $\neg P(f(a)) \vee R(a)$

$\Rightarrow$ $Q(f(a)) \vee R(a)$

Clause 1*: $P(f(x)) \vee Q(f(x))$

Clause 2*: $\neg P(f(x)) \vee R(x)$

$\Rightarrow$ $Q(f(x)) \vee R(x)$

Factor

- If two or more literals (with the same sign) of a clause C have a MGU σ, then Cσ is called a **factor** of C
- If Cσ is a unit clause, it is called a **unit factor** of C

Example:

$$C = \underline{P(x) \vee P(f(y))} \vee \neg Q(x)$$

$$\sigma = \{f(y)/x\}$$

$$C\sigma = P(f(y)) \vee \neg Q(f(y))$$

Binary Resolvent

$$C_1 = \ldots \vee L_1 \vee \ldots$$

No variables in common!

$$\sigma = \mathsf{MGU}(L_1, \neg L_2) \quad \bigwedge \quad (C_1\sigma \setminus L_1\sigma) \cup (C_2\sigma \setminus L_2\sigma)$$

$$C_2 = \ldots \vee L_2 \vee \ldots$$

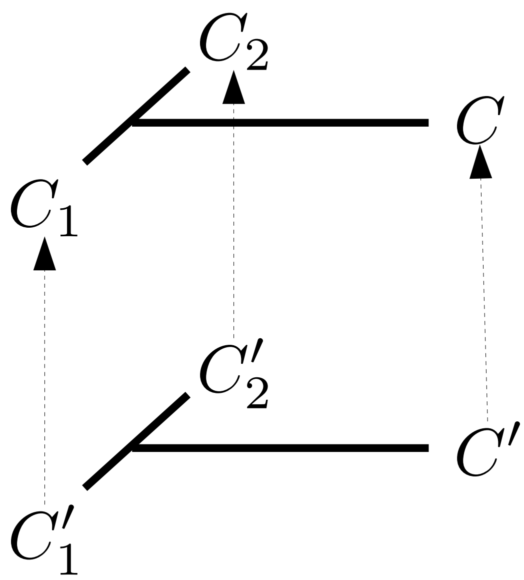
Resolvent

A resolvent of (parent) clauses C_1 and C_2 is one of the following binary resolvents:

- A binary resolvent of C_1 and C_2
- A binary resolvent of C_1 and a factor of C_2
- A binary resolvent of a factor of C_1 and C_2
- A binary resolvent of a factor of C_1 and a factor of C_2

Lifting Lemma

If C_1' and C_2' are instances of C_1 and C_2, respectively, and if C' is a resolvent of C_1' and C_2', then there exists a resolvent C of C_1 and C_2 such that C' is an instance of C.



Completeness of Resolution

A set S of clauses is unsatisfiable if and only if there is a deduction of the empty clause F from S.

Proof (sketch):

- [$\Rightarrow$]: Suppose S is unsatisfiable; let T be a complete semantic tree for S; T has a finite closed semantic tree T'. Use structural induction on T' together with the Lifting Lemma to show that there is a deduction of the empty clause from S
- [$\Leftarrow$]: Suppose there is a deduction of F. Let $R_1, \ldots, R_k$ be the resolvents in the deduction. Assume S is satisfiable. Then, there is a model M of S. If M |= C_1 and C_2, it also |= any resolvent; then M |= $R_1, \ldots, R_k$; then M |= F, which is impossible!

Thank you for your attention