13

Resolution

Claudia Chirita

School of Informatics, University of Edinburgh

Based on slides by: Jacques Fleuriot, Michael Rovatsos, Michael Herrmann, Vaishak Belle

13.a

The resolution inference rule

RESOLUTION

A method for telling whether a propositional formula is satisfiable and for proving that a first-order formula is unsatisfiable.

Yields a **complete** inference algorithm.

If a set of clauses is unsatisfiable, then the resolution closure of those clauses contains the empty clause (propositional logic).

GROUND BINARY RESOLUTION

$$\frac{C \vee P \qquad D \vee \neg P}{C \vee D}$$

SOUNDNESS

$C \vee P$ iff $\neg C \rightarrow P$

$D \vee \neg P$ iff $P \rightarrow D$

Therefore, $\neg C \rightarrow D$,
which is equivalent to $C \vee D$.

If both C and D are empty, then resolution deduces the **empty clause**, i.e. *false*.

NON-GROUND BINARY RESOLUTION

$$\frac{C \vee P \qquad D \vee \neg P'}{(C \vee D)\,\theta}$$

where θ is the mgu of P and P'.

The two clauses are assumed to be **standardized apart** so that they share no variables.

SOUNDNESS

Apply θ to premises, then appeal to ground binary resolution.

$$\frac{C\theta \vee P\theta \qquad D\theta \vee \neg P\theta}{C\theta \vee D\theta}$$

BINARY RESOLUTION · EXAMPLE

$$\frac{\neg \mathsf{HasHunny}(x) \lor \mathsf{Happy}(x) \qquad \mathsf{HasHunny}(\mathsf{Pooh})}{\mathsf{Happy}(\mathsf{Pooh})}$$

with $\theta = \{x/\mathsf{Pooh}\}$

FACTORING

$$\frac{C \vee P_1 \vee \ldots \vee P_m}{(C \vee P_i)\,\theta}$$

where θ is the mgu of the P_i.

SOUNDNESS

By universal instantiation and deletion of duplicates.

FULL RESOLUTION

$$\frac{C \vee P_1 \vee \ldots \vee P_m \qquad D \vee \neg P'_1 \vee \ldots \vee \neg P'_n}{(C \vee D)\,\theta}$$

where θ is the mgu of all P_i and P'_j.

SOUNDNESS

By combination of factoring and binary resolution.

To prove α, apply resolution steps to CNF$(\mathrm{KB} \wedge \neg\alpha)$.

Complete for FOL, if using **full resolution** or
binary resolution + factoring.

13.b

Resolution algorithm

RESOLUTION ALGORITHM

IDEA

To prove α, apply resolution steps to $\text{CNF}(\text{KB} \wedge \neg\alpha)$.

Complete for FOL, if using **full resolution** or
binary resolution + factoring.

CNF CONVERSION · EXAMPLE

Everyone who loves all animals is loved by someone.

$\forall x.[\forall y.\mathsf{Animal}(y) \to \mathsf{Loves}(x, y)] \to [\exists y.\mathsf{Loves}(y, x)]$

1. Eliminate all implications and biconditionals.

$\forall x.\neg[\forall y.\neg\mathsf{Animal}(y) \lor \mathsf{Loves}(x, y)] \lor [\exists y.\mathsf{Loves}(y, x)]$

2. Move $\neg$ inwards, using $\neg\forall x.\varphi \equiv \exists x.\neg\varphi$, $\neg\exists x.\varphi \equiv \forall x.\neg\varphi$.

$\forall x.[\exists y.\neg(\neg\mathsf{Animal}(y) \lor \mathsf{Loves}(x, y))] \lor [\exists y.\mathsf{Loves}(y, x)]$

$\forall x.[\exists y.\neg\neg\mathsf{Animal}(y) \land \neg\mathsf{Loves}(x, y)] \lor [\exists y.\mathsf{Loves}(y, x)]$

$\forall x.[\exists y.\mathsf{Animal}(y) \land \neg\mathsf{Loves}(x, y)] \lor [\exists y.\mathsf{Loves}(y, x)]$

3. **Standardize variables** – each quantifier should use a different one.

$\forall x.[\exists y.\text{Animal}(y) \land \neg\text{Loves}(x, y)] \lor [\exists z.\text{Loves}(z, x)]$

4. **Skolemize** – a more general form of existential instantiation.
Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables.[1]

$\forall x.[\text{Animal}(F(x)) \land \neg\text{Loves}(x, F(x))] \lor \text{Loves}(G(x), x)$

[1] No enclosing universal quantifier? Just replace with Skolem constant.

4. Drop universal quantifiers.

$$[\text{Animal}(F(x)) \land \neg\text{Loves}(x, F(x))] \lor \text{Loves}(G(x), x)$$

5. Distribute $\lor$ **over** $\land$.

$$[\text{Animal}(F(x)) \lor \text{Loves}(G(x), x)] \land [\neg\text{Loves}(x, F(x)) \lor \text{Loves}(G(x), x)]$$

RESOLUTION ALGORITHM

```
function PL-RESOLUTION(KB, α) returns true or false
   inputs: KB, the knowledge base, a sentence in propositional logic
           α, the query, a sentence in propositional logic

   clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
   new ← { }
   loop do
      for each pair of clauses C_i, C_j in clauses do
         resolvents ← PL-RESOLVE(C_i, C_j)
         if resolvents contains the empty clause then return true
         new ← new ∪ resolvents
      if new ⊆ clauses then return false
      clauses ← clauses ∪ new
```

returns the set of all possible clauses obtained by resolving its two inputs

EXAMPLE · WINNIE-THE-POOH CNF

$\neg \text{VeryFondOfFood}(x) \lor \neg \text{Treat}(y) \lor \neg \text{Friend}(z)$
$\lor \neg \text{Gives}(x, y, z) \lor \text{Generous}(x)$

$\text{Hunny}(\text{J})$

$\text{Owns}(\text{Eeyore}, \text{J})$

$\neg \text{Hunny}(x) \lor \neg \text{Owns}(\text{Eeyore}, x) \lor \text{Gives}(\text{Pooh}, x, \text{Eeyore})$

$\neg \text{Hunny}(x) \lor \text{Treat}(x)$

$\neg \text{Resident}(x, \text{HundredAcreWood}) \lor \text{Friend}(x)$

$\text{Resident}(\text{Eeyore}, \text{HundredAcreWood})$

$\text{VeryFondOfFood}(\text{Pooh})$

EXAMPLE · RESOLUTION PROOF

¬VeryFondOfFood(x) ∨ ¬Treat(y) ∨ ¬Friend(z) ∨ ¬Gives(x,y,z) ∨ Generous(x)

¬Generous(Pooh)

VeryFondOfFood(Pooh)

¬VeryFondOfFood(Pooh) ∨ ¬Treat(y) ∨ ¬Friend(z) ∨ ¬Gives(Pooh,y,z)

¬Hunny(x) ∨ Treat(x)

¬Treat(y) ∨ ¬Friend(z) ∨ ¬Gives(Pooh,y,z)

Hunny(J)

¬Hunny(y) ∨ ¬Friend(z) ∨ ¬Gives(Pooh,y,z)

¬Hunny(x) ∨ ¬Owns(Eeyore,x) ∨ Gives(Pooh,x,Eeyore)

¬Friend(z) ∨ ¬Gives(Pooh,J,z)

Hunny(J)

¬Hunny(J) ∨ ¬Owns(Eeyore,J) ∨ ¬Friend(Eeyore)

Owns(Eeyore,J)

¬Owns(Eeyore,J) ∨ ¬Friend(Eeyore)

¬Resident(x,HundredAcreWood) ∨ Friend(x)

¬Friend(Eeyore)

Resident(Eeyore,HundredAcreWood)

¬Resident(Eeyore,HundredAcreWood)

□

13.c

Relationship with Modus Ponens

GMP BASED INFERENCE

Backward Chaining

if *Goal* is known (goal directed)
can query for data

Forward Chaining

if specific *Goal* is not known, but the system needs to react
to new facts (data driven)
can make suggestions

What do users expect from the system?

Which direction has the larger branching factor?

LIMITATIONS

...due to restriction to definite clauses

In order to apply GMP

premises of rules contain only non-negated symbols
the conclusion of any rule is a non-negated symbol
facts are non-negated atomic sentences

Possible solution introduce more variables, e.g. $\mathrm{Q_P} := \neg\mathrm{P}$

What about... “If we cannot prove P, then $\mathrm{Q_P}$ is true”?
(works only if there is a rule for each variable)

MODUS PONENS & GROUND BINARY RESOLUTION

Ground binary resolution

$$\frac{C \lor P \qquad D \lor \neg P}{C \lor D}$$

Suppose $C = \textit{false}$.

$$\frac{P \qquad \neg P \lor D}{D}$$

i.e. P and $P \rightarrow D$ entails D.

Modus ponens is a special case of binary resolution.

GMP & FULL RESOLUTION

GMP with $p_i'\theta = p_i\theta$

$$\frac{p_1', p_2', \dots, p_n' \quad (p_1 \wedge p_2 \wedge \dots \wedge p_n \to q)}{q\theta}$$

$$\frac{p_1', p_2', \dots, p_n' \quad (q \vee \neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_n)}{q\theta}$$

Full resolution with θ **mgu of all** P_i **and** P_j'

$$\frac{C \vee P_1 \vee \dots \vee P_m \qquad D \vee \neg P_1' \vee \dots \vee \neg P_n'}{(C \vee D)\,\theta}$$