

# Inference by Resolution

**Readings:** Chapter 9 of Russell & Norvig

## Automating Inference in FOL

- Consider the inference system  $\mathcal{R}_1$  for First-Order Logic.
- How do we prove that sentence  $\varphi$  is a consequence of a set  $\Gamma$  of sentence?
- We try to build a proof of  $\varphi$  from  $\Gamma$  by using the inference rules of  $\mathcal{R}_1$ .
- How can implement a program that builds the proof automatically?  
(if there is one)
- Turn the derivation problem into a search problem.

## Inference as Search

Sets of sentences  $\longrightarrow$  States  
Inference rules  $\longrightarrow$  Operators

### Operators

- **Preconditions:**  $Op_r$  applies to a state if the corresponding inference rule  $r$  applies to some of the sentences in the state.
- **Effects:** Applying an operator  $Op_r$  to a state with sentences  $\Gamma$  produces a new state with sentences  $\Gamma \cup \{\alpha\}$  where  $\alpha$  is possible conclusion of  $r$ .

Example:

$$Op_{MP}(\{Happy(Joe) \Rightarrow Sings(Joe), Happy(Joe), \dots\}) = \\ \{Happy(Joe) \Rightarrow Sings(Joe), Happy(Joe), \dots\} \cup \{Sings(Joe)\}$$

## Inference as Search

### Problem

Prove that  $\varphi$  follows from  $\{\alpha_1, \dots, \alpha_n\}$

### Initial State

$\{\alpha_1, \dots, \alpha_n\}$

### Goal State

Any state containing  $\varphi$ .

### Implementation

- Use one of the search strategies seen earlier to search the state space until a goal state is found.
- The sequence of operators along the path from the initial state to the goal state is a proof of  $\varphi$  from  $\{\alpha_1, \dots, \alpha_n\}$ .

## Inference as Search

- The inference system  $\mathcal{R}_1$  is not appropriate for automated inference by search.
- The corresponding state space is often huge!
- There are too many rules and each of them can be applied in too many ways.
- To reduce the branching factor will consider an inference system for FOL with just one rule: *the resolution rule*.

## Resolution: Main Idea

Let  $A$  be an atomic sentence and  $\alpha, \beta$  disjunctions of literals<sup>1</sup>.

### The Resolution Rule

$$\frac{\alpha \vee A \quad \neg A \vee \beta}{\alpha \vee \beta}$$

*Example:*

$$\frac{\neg Pet(x) \vee Cat(x) \vee Bird(x) \quad \neg Bird(x) \vee Parrot(x) \vee Canary(x)}{\neg Pet(x) \vee Cat(x) \vee Parrot(x) \vee Canary(x)}$$

---

<sup>1</sup>A **literal** is either an atomic sentence (a predicate) or the negation of one.

## Resolution: A More General Formulation

Assume that all our sentences look like the following where all the  $A$ 's and  $B$ 's are **atomic** sentences.

$$\neg A_1 \vee \dots \vee \neg A_m \vee B_1 \vee \dots \vee B_n$$

Then, all we need is a rule like this:

$$\frac{\neg A_1 \vee \dots \vee \neg A_m \vee \boxed{A} \vee B_1 \vee \dots \vee B_n \quad \neg C_1 \vee \dots \vee \neg C_p \vee \boxed{\neg A} \vee D_1 \vee \dots \vee D_q}{\neg A_1 \vee \dots \vee \neg A_m \vee \neg C_1 \vee \dots \vee \neg C_p \vee B_1 \vee \dots \vee B_m \vee D_1 \vee \dots \vee D_q}$$

Simplified forms of the *same* rule:

$$\frac{A \vee B_1 \vee \dots \vee B_n \quad \neg A}{B_1 \vee \dots \vee B_n} \quad \frac{A \quad \neg A \vee D_1 \vee \dots \vee D_q}{D_1 \vee \dots \vee D_q} \quad \frac{A \quad \neg A}{\mathbf{False}}$$

## Resolution in FOL

Can we apply the previous rule to the following sentences?

$$\neg Pet(Joe) \vee Cat(Joe) \vee Bird(Joe) \quad \neg Bird(x) \vee Parrot(x) \vee Canary(x)$$

No, because  $Bird(Joe)$  and  $Bird(x)$  are not identical sentences.

However, the following inference makes sense

$$\frac{\neg Pet(Joe) \vee Cat(Joe) \vee Bird(Joe) \quad \neg Bird(x) \vee Parrot(x) \vee Canary(x)}{\neg Pet(Joe) \vee Cat(Joe) \vee Parrot(Joe) \vee Canary(Joe)}$$

The reason is that  $Bird(Joe)$  and  $Bird(x)$  are **unifiable**, that is, can be made identical by replacing variables by appropriate terms.

## Unification

Two FOL terms  $t_1, t_2$  **unify** if there is a substitution  $\theta$  of their variables into terms such that

$$\theta(t_1) \text{ is identical to } \theta(t_2),$$

where  $\theta(t_1)$  is the term obtained by *applying* the substitution  $\theta$  to  $t_1$  (similarly for  $t_2$ ).

### *Examples*

$$\begin{array}{lll}
 F(x) \text{ and } F(A) & \text{unify with } \theta := \{x/A\} \\
 G(x, B) \text{ and } G(A, y) & \text{unify with } \theta := \{x/A, y/B\} \\
 z \text{ and } G(A, y) & \text{unify with } \theta := \{z/G(A, y)\} \\
 F(G(H(y)), H(A)) \text{ and } F(G(x), x) & \text{unify with } \theta := \{x/H(A), y/A\}
 \end{array}$$

## Unification

- a variable unifies with any term in which it does not occur.
- a constant symbol unifies only with itself
- two compound terms (or two atomic sentences) unify only if
  - their top symbol is the same,
  - that symbol is applied to the same number of arguments
  - the arguments unify pairwise.

### *Examples*

1.  $x$  and  $A$  unify but  $A$  and  $B$  do not
2.  $x$  and  $F(A, H(x))$  do not unify ( $x$  occurs in  $F(A, H(x))$ )
3.  $F(H(x))$  and  $G(H(x))$  do not unify (different top symbols)
4.  $F(x)$  and  $F(x, y)$  do not unify (different number of arguments)
5.  $F(t_1, t_2, \dots, t_n)$  and  $F(s_1, s_2, \dots, s_n)$  unify if  $t_1$  and  $s_1$  unify,  $t_2$  and  $s_2$  unify,  $\dots$ ,  $t_n$  and  $s_n$  unify.

## Unifiers

- $\theta$  is a **unifier** of  $t_1$  and  $t_2$  if  $\theta(t_1)$  is identical to  $\theta(t_2)$ .
- Some terms may have more than one unifier:

$$G(x, H(z)) \text{ and } G(A, y) \text{ unify with } \theta_1 := \{x/A, y/H(A), z/A\}$$
$$\text{or with } \theta_2 := \{x/A, y/H(z)\}$$

- But some unifiers are more general than others.  
( $\theta_2$  above is more general than  $\theta_1$ )
- When unifying two terms we are interested in their **most general unifier (mgu)**.

## The Unification Procedure

- Input:** a set  $V$  of the form  $\{t =? t'\}$
- Output:** either FAIL or a set of the form  $\{v_1 =? t_1, \dots, v_m =? t_m\}$
- Procedure:** Repeatedly apply (in any order) the transformations rules **Delete**, **Eliminate**, **Decompose**, **Orient**, **Occurs** and **Mismatch** to  $V$  until no more transformations apply.

If the procedure returns  $\{v_1 =? t_1, \dots, v_m =? t_m\}$  on input  $\{t =? t'\}$ , then  $\{v_1/t_1, \dots, v_m/t_m\}$  is a mgu of  $t$  and  $t'$ .

**Notation:**  $v$  denotes a variable,  $t$  denotes a term (possibly a variable).

## The Transformation Rules

Decompose.  $U \cup \{f(t_1, \dots, t_n) =? f(s_1, \dots, s_n)\} \longrightarrow U \cup \{t_1 =? s_1, \dots, t_n =? s_n\}$

Orient.  $U \cup \{t =? v\} \longrightarrow U \cup \{v =? t\}$

Delete.  $U \cup \{v =? v\} \longrightarrow U$

Eliminate.  $U \cup \{v =? t\}, v \in \mathcal{Vars}(U) \setminus \mathcal{Vars}(t) \longrightarrow U[v/t] \cup \{v =? t\}$

Mismatch.  $U \cup \{f(t_1, \dots, t_m) =? g(s_1, \dots, s_n)\},$   
 $f, g$  distinct or  $m \neq n \longrightarrow FAIL$

Occurs.  $U \cup \{v =? t\}, v \neq t$  but  $v \in \mathcal{Vars}(t) \longrightarrow FAIL$

**Notation:**  $\mathcal{Vars}(U), \mathcal{Vars}(t)$  denote the set of variables in  $U, t$ ;  
 $v$  denotes a variable,  $s, t$  denote terms (possibly variables);  
 $f, g$  denote function/constant symbols.

## The Unification Procedure: Examples

**Unify:**  $F(G(H(y)), H(A))$  and  $F(G(x), x)$

$$\{ \underline{F(G(H(y)), H(A))} =? F(G(x), x) \} \xrightarrow{\text{Decompose}}$$

$$\{ \underline{G(H(y))} =? G(x), H(A) =? x \} \xrightarrow{\text{Decompose}}$$

$$\{ \underline{H(y)} =? x, H(A) =? x \} \xrightarrow{\text{Orient}}$$

$$\{ \underline{x =? H(y)}, H(A) =? x \} \xrightarrow{\text{Eliminate } x}$$

$$\{ x =? H(y), \underline{H(A) =? H(y)} \} \xrightarrow{\text{Decompose}}$$

$$\{ x =? H(y), \underline{A =? y} \} \xrightarrow{\text{Orient}}$$

$$\{ x =? H(y), \underline{y =? A} \} \xrightarrow{\text{Eliminate } y}$$

$$\{ x =? H(A), y =? A \}$$

$$mgu = \{ x/H(A), y/A \}$$

## The Unification Procedure: Examples

**Unify:**  $F(A, H(y), z)$  and  $F(A, x, z)$

$$\{ \underline{F(A, H(y), z)} =? F(A, x, z) \} \xrightarrow{\text{Decompose}}$$

$$\{ \underline{A =? A}, H(y) =? x, z =? z \} \xrightarrow{\text{Decompose}}$$

$$\{ \underline{H(y) =? x}, z =? z \} \xrightarrow{\text{Orient}}$$

$$\{ x =? H(y), \underline{z =? z} \} \xrightarrow{\text{Delete}}$$

$$\{ x =? H(y) \}$$

$$mgu = \{ x/H(y) \}$$

## The Unification Procedure: Examples

**Unify:**  $F(H(y), z)$  and  $F(G(x), z)$

$$\begin{array}{l} \{F(H(y), z) =? F(G(x), z)\} \xrightarrow{\text{Decompose}} \\ \{H(y) =? G(x), z =? z\} \xrightarrow{\text{Mismatch}} \\ \text{FAIL} \end{array}$$

do not unify

**Unify:**  $F(y)$  and  $F(H(x, y))$

$$\begin{array}{l} \{F(y) =? F(H(x, y))\} \xrightarrow{\text{Decompose}} \\ \{y =? H(x, y)\} \xrightarrow{\text{Occurs}} \\ \text{FAIL} \end{array}$$

do not unify

## Clauses

- A **literal** is an atomic sentence or a negated atomic sentence.

*Ex:*  $P(x)$ ,  $Q(y, A)$ ,  $\neg P(x)$ ,  $\neg Q(y, A)$

- A **clause** is the universal closure of a disjunction of (zero or more) literals.

*Ex:*  $\forall x P(x)$ ,  $\forall x \neg Q(y, A)$ ,  $\forall x \forall y [\neg P(x) \vee Q(y) \vee R(y) \vee \neg S(x)]$

- *The resolution rule applies only to clauses.*

## More on Clauses

- Clauses are usually written without the universal quantifiers (which are then implicitly assumed).

*Ex:*  $P(x)$ ,  $\neg Q(y, A)$ ,  $\neg P(x) \vee Q(y) \vee R(y) \vee \neg S(x)$

- Since  $\vee$  is commutative, the order of the literals in a clause is irrelevant.

*Ex:*  $\neg P(x) \vee Q(y)$  and  $Q(y) \vee \neg P(x)$  considered the same.

- Since  $\vee$  is associative, the disjunction of two clauses is also a clause.

*Ex:*  $(\neg P(x) \vee Q(y)) \vee (R(y) \vee \neg S(x))$  same as  $\neg P(x) \vee Q(y) \vee R(y) \vee \neg S(x)$ .

- Since the **empty clause** (ie, the empty disjunction of literals) is always false, we represent it as **False**.

## The Resolution Rule for FOL

$$\frac{\varphi_1 \vee \alpha_1 \quad \neg\alpha_2 \vee \varphi_2}{\theta(\varphi_1 \vee \varphi_2)} \quad (+)$$

(+) where  $\varphi_1 \vee \alpha_1$ ,  $\neg\alpha_2 \vee \varphi_2$  are clauses with no variables in common,  $\alpha_1, \alpha_2$  are unifiable *atomic* sentences, and  $\theta$  is the mgu of  $\alpha_1, \alpha_2$ .

Simplified forms:

$$\frac{\alpha_1 \quad \neg\alpha_2 \vee \varphi}{\theta(\varphi)}$$

$$\frac{\varphi \vee \alpha_1 \quad \neg\alpha_2}{\theta(\varphi)}$$

$$\frac{\alpha_1 \quad \neg\alpha_2}{\mathbf{False}}$$

If  $\gamma$  is obtained from  $\gamma_1, \gamma_2$  by an application of the resolution rule,  $\gamma$  is a **resolvent** of  $\gamma_1$  and  $\gamma_2$ .

## Resolution: Examples

$$\frac{\neg Pet(Joe) \vee Cat(Joe) \vee Bird \downarrow(Joe) \quad Parrot(x) \vee \neg Bird \downarrow(x)}{\neg Pet(Joe) \vee Cat(Joe) \vee Parrot(Joe)} \quad (+)$$

$$(+)\ mgu(Bird(x), Bird(Joe)) = \{x/Joe\}$$

$$\frac{\neg On \downarrow(x, y) \vee Above(x, y) \quad On(B, A) \vee On \downarrow(A, B)}{Above(A, B) \vee On(B, A)} \quad (*)$$

$$(*)\ mgu(On(x, y), On(A, B)) = \{x/A, y/B\}$$

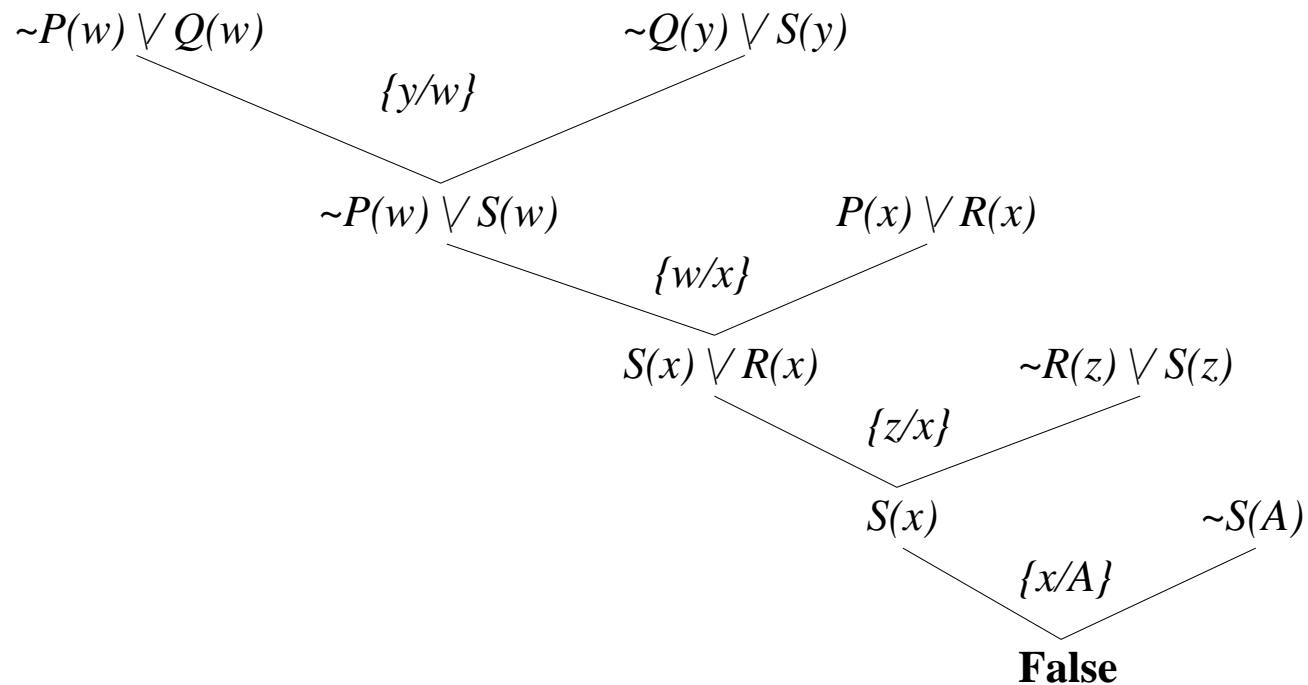
$$\frac{\neg Bird(x) \vee Feathers \downarrow(x) \quad \neg Feathers \downarrow(y) \vee Flies(y)}{\neg Bird(x) \vee Flies(x)} \quad (\dagger)$$

$$(\dagger)\ mgu(Feathers(x), Feathers(y)) = \{y/x\}$$

## Resolution Proofs by Refutation

Let  $\Gamma = \{\neg P(w) \vee Q(w), \neg Q(y) \vee S(y), P(x) \vee R(x), \neg R(z) \vee S(z)\}$ .

To prove  $S(A)$  from  $\Gamma$ , try to prove **False** from  $\Gamma \cup \{\neg S(A)\}$



## Why Resolution by Refutation Works

- Let  $\Gamma$  be a set of clauses and  $\alpha$  an atomic sentence.
- To show that  $\alpha$  follows from  $\Gamma$ , prove **False** from  $\Gamma \cup \{\neg\alpha\}$ .
- This is a correct thing to do because:

<b>False</b> provable from $\Gamma \cup \{\neg\alpha\}$ by resolution	implies
$\Gamma \cup \{\neg\alpha\} \models \mathbf{False}$	implies
$\Gamma \cup \{\neg\alpha\}$ is unsatisfiable	implies
$\Gamma \models \alpha$	

## Resolution and FOL Sentences

- The resolution rule applies only to clauses.
- What if I want to show  $\Gamma \models \alpha$  for *arbitrary* sentences of FOL?
- First convert them into a set of clauses and then use resolution.
- *Every* set of FOL sentences can be converted into **Clause Form**, that is, into an equisatisfiable set of clauses.

## Skolemization

- The main hurdle in converting into clause form is eliminating *existential quantifiers*.
- After  $\exists$  quantifiers have been eliminated from a sentence, all variables are universally quantified.
- Therefore,  $\forall$  quantifiers can be simply dropped.
- The process of eliminating existential quantifiers is called **Skolemization**.

## Skolemization

### Example 1

$$\exists x \forall y \text{ Loves}(x, y)$$

is satisfiable if there is an  $x$  that makes  $\text{Loves}(x, y)$  true no matter the  $y$ .

The value of  $x$  must be the same for any  $y$ .

Replace  $x$  by a *fresh* constant, say,  $S$ .

$$\forall y \text{ Loves}(S, y)$$

$S$  is called a **Skolem constant**.

## Skolemization

### Example 2

$$\forall y \exists x \text{ Loves}(x, y)$$

is satisfiable if for each  $y$  we can find an  $x$  that makes  $\text{Loves}(x, y)$  true.

The value of  $x$  can change depending on  $y$ .

Replace  $x$  by a *fresh* function of  $y$ , say,  $\text{Lover}(y)$ .

$$\forall y \text{ Loves}(\text{Lover}(y), y)$$

*Lover* is called a **Skolem function**.

## Skolemization

### General Rule

If an  $\exists$  variable  $x$  is in the scope of  $n \geq 0$   $\forall$  quantifiers, drop  $\exists x$  and replace  $x$  by a *new* Skolem function with  $n$  arguments.

The arguments of the Skolem function are the variables of the  $\forall$  quantifiers.

*Example:*

$$\exists x \forall y \forall z \exists u P(x, y, z, u)$$

*becomes*

$$\forall y \forall z P(A, y, z, F(y, z))$$

## Conversion to Clause Form

### Procedure:

1. **Eliminate implications**
2. **Move  $\neg$  inwards**
3. **Standardize bound variables apart**
4. **Move quantifiers out**
5. **Skolemize existential variables**
6. **Eliminate universal quantifiers**
7. **Distribute  $\vee$  over  $\wedge$**
8. **Flatten conjunctions and disjunctions**
9. **Eliminate conjunctions**

$$\{\forall x (\forall y P(x, y)) \Rightarrow \neg(\forall y Q(x, y) \Rightarrow R(x, y))\}$$

## 1. Eliminate implications

$\alpha \Rightarrow \beta$	$\longrightarrow$	$\neg\alpha \vee \beta$
$\alpha \Leftrightarrow \beta$	$\longrightarrow$	$(\neg\alpha \vee \beta) \wedge (\neg\beta \vee \alpha)$

$$\{\forall x \neg(\forall y P(x, y)) \vee \neg(\forall y \neg Q(x, y) \vee R(x, y))\}$$

## 2. Move $\neg$ inwards

$\neg\neg\alpha$	$\longrightarrow$	$\alpha$
$\neg(\alpha \vee \beta)$	$\longrightarrow$	$\neg\alpha \wedge \neg\beta$
$\neg(\alpha \wedge \beta)$	$\longrightarrow$	$\neg\alpha \vee \neg\beta$
$\neg\forall v \alpha$	$\longrightarrow$	$\exists v \neg\alpha$
$\neg\exists v \alpha$	$\longrightarrow$	$\forall v \neg\alpha$

$$\{\forall x (\exists y \neg P(x, y)) \vee (\exists y Q(x, y) \wedge \neg R(x, y))\}$$

$$\{\forall x (\exists y \neg P(x, y)) \vee (\exists y Q(x, y) \wedge \neg R(x, y))\}$$

### 3. Standardize bound variables apart

no two different quantifiers with the same variable

$$\{\forall x (\exists y \neg P(x, y)) \vee (\exists z Q(x, z) \wedge \neg R(x, z))\}$$

### 4. Move quantifiers out

$(Qx \alpha) \wedge \beta$	$\longrightarrow$	$Qx (\alpha \wedge \beta)$
$(Qx \alpha) \vee \beta$	$\longrightarrow$	$Qx (\alpha \vee \beta)$
$\alpha \wedge (Qx \beta)$	$\longrightarrow$	$Qx (\alpha \wedge \beta)$
$\alpha \vee (Qx \beta)$	$\longrightarrow$	$Qx (\alpha \vee \beta)$

$$\{\forall x \exists y \exists z \neg P(x, y) \vee (Q(x, z) \wedge \neg R(x, z))\}$$

$$\{\forall x \exists y \exists z \neg P(x, y) \vee (Q(x, z) \wedge \neg R(x, z))\}$$

## 5. Skolemize existential variables

$$\exists v \alpha \longrightarrow \alpha[v/\pi(v_1, \dots, v_n)]$$

with  $\pi$  *new* and  $v_1, \dots, v_n$  universally quantified outside  $\exists v \alpha$

$$\{\forall x \neg P(x, F_1(x)) \vee (Q(x, F_2(x)) \wedge \neg R(x, F_2(x)))\}$$

## 6. Eliminate universal quantifiers

$$\forall v \alpha \longrightarrow \alpha$$

$$\{\neg P(x, F_1(x)) \vee (Q(x, F_2(x)) \wedge \neg R(x, F_2(x)))\}$$

$$\{\neg P(x, F_1(x)) \vee (Q(x, F_2(x)) \wedge \neg R(x, F_2(x)))\}$$

## 7. Distribute $\vee$ over $\wedge$

$$\begin{array}{l} \alpha \vee (\beta \wedge \gamma) \longrightarrow (\alpha \vee \beta) \wedge (\alpha \vee \gamma) \\ (\beta \wedge \gamma) \vee \alpha \longrightarrow (\beta \vee \alpha) \wedge (\gamma \vee \alpha) \end{array}$$

$$\{(\neg P(x, F_1(x)) \vee Q(x, F_2(x))) \wedge (\neg P(x, F_1(x)) \vee \neg R(x, F_2(x)))\}$$

## 8. Flatten conjunctions and disjunctions

$$\begin{array}{l} (\alpha \wedge (\beta \wedge \gamma)) \longrightarrow (\alpha \wedge \beta \wedge \gamma) \\ (\alpha \vee (\beta \vee \gamma)) \longrightarrow (\alpha \vee \beta \vee \gamma) \\ ((\alpha \wedge \beta) \wedge \gamma) \longrightarrow (\alpha \wedge \beta \wedge \gamma) \\ ((\alpha \vee \beta) \vee \gamma) \longrightarrow (\alpha \vee \beta \vee \gamma) \end{array}$$

$$\{(\neg P(x, F_1(x)) \vee Q(x, F_2(x))) \wedge (\neg P(x, F_1(x)) \vee \neg R(x, F_2(x)))\}$$

After step 8 each sentence in the set is in **Conjunctive Normal Form (CNF)**:  $(\alpha_1 \vee \dots \vee \alpha_m) \wedge (\beta_1 \vee \dots \vee \beta_n) \wedge \dots$

$$\{(\neg P(x, F_1(x)) \vee Q(x, F_2(x))) \wedge (\neg P(x, F_1(x)) \vee \neg R(x, F_2(x)))\}$$

## 9. Eliminate conjunctions

$$\boxed{\{\alpha \wedge \beta\} \longrightarrow \{\alpha, \beta\}}$$

$$\{\neg P(x, F_1(x)) \vee Q(x, F_2(x)), \neg P(x, F_1(x)) \vee \neg R(x, F_2(x))\}$$

Done! The final set is satisfiable iff the initial set is satisfiable.

## To Summarize

- With resolution we give proofs by **contradiction**.

To show  $\Gamma \models \varphi$

- let  $\Gamma_1 := \Gamma \cup \{\neg\varphi\}$ ;
  - convert  $\Gamma_1$  into an equisatisfiable set  $\Gamma_2$  of clauses;
  - try to derive **False** from  $\Gamma_2$  by resolution.
- If you can derive **False**, then  $\Gamma_1 := \Gamma \cup \{\neg\varphi\}$  is unsatisfiable.
  - Conclude that  $\Gamma \models \varphi$ .

## Resolution by Refutation

- A proof by contradiction is also called a **refutation**.
- The resolution rule with factoring<sup>2</sup> is a **refutation-complete** inference system:

if a set of clauses is unsatisfiable,  
then **False** is provable from it by resolution.

---

<sup>2</sup>Factoring is a technicality, see the textbook.

## Example

Prove that  $Siblings(Joe, Jill)$  follows from

$$\Gamma = \{ \forall x \forall y (\exists z Mom(x, z) \wedge Mom(y, z)) \Rightarrow Siblings(x, y), \\ Mom(Joe, Liv), Mom(Jill, Liv) \}$$

- Add negation of goal sentence to premises.

$$\Gamma' = \{ \forall x \forall y (\exists z Mom(x, z) \wedge Mom(y, z)) \Rightarrow Siblings(x, y), \\ Mom(Joe, Liv), Mom(Jill, Liv), \neg Siblings(Joe, Jill) \}$$

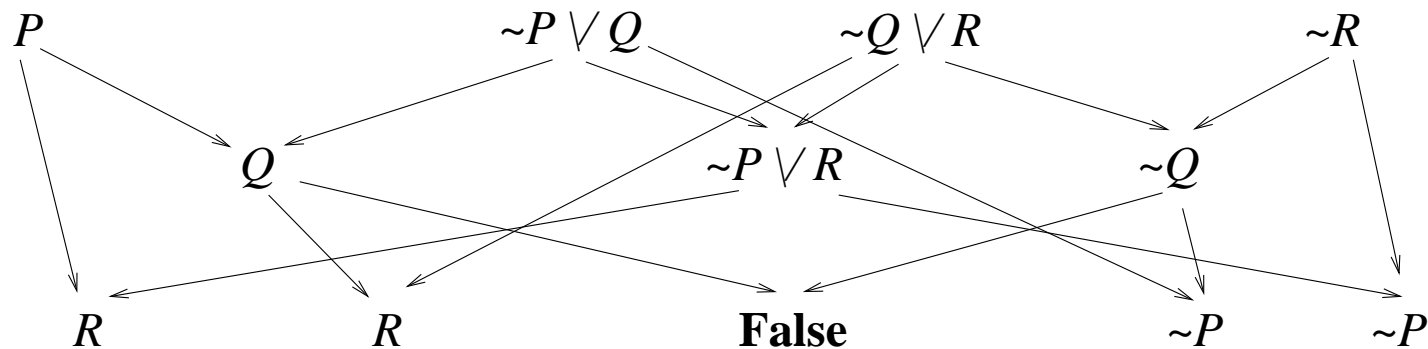
- Convert to clause form.

$$\Gamma'' = \{ \neg Mom(x, z) \vee \neg Mom(y, z) \vee Siblings(x, y), \\ Mom(Joe, Liv), Mom(Jill, Liv), \neg Siblings(Joe, Jill) \}$$

- Try to derive **False** from  $\Gamma''$  (exercise).
- Once **False** is derived, conclude that  $\Gamma \models Siblings(Joe, Jill)$ .

## Still Too Many Choices!

- A resolution-based inference system has just one rule to apply to build a proof.
- However, at any step there may be several possible ways to apply the resolution rule.



- Several **resolution strategies** have been developed in order to reduce the search space.

## Some Resolution Strategies

- **Unit preference**
- **Set of support**
- **Input resolution**
- **Linear resolution**
- **Subsumption**

**Note:** the first 3 strategies above are incomplete.