

# Decision Procedures in Verification

First-Order Logic (3)

28.11.2022

Viorica Sofronie-Stokkermans

e-mail: [sofronie@uni-koblenz.de](mailto:sofronie@uni-koblenz.de)

# Until now:

---

**Syntax** (one-sorted signatures vs. many-sorted signatures)

## **Semantics**

Structures (also many-sorted)

Models, Validity, and Satisfiability

Entailment and Equivalence

**Theories (Syntactic vs. Semantics view)**

**Normal Forms**

**Herbrand Interpretations**

## 2.6 Herbrand Interpretations

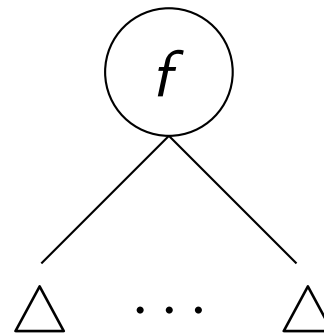
---

From now on we shall consider PL without equality.  $\Omega$  shall contain at least one constant symbol.

A **Herbrand interpretation** (over  $\Sigma$ ) is a  $\Sigma$ -algebra  $\mathcal{A}$  such that

- $U_{\mathcal{A}} = T_{\Sigma}$  (= the set of ground terms over  $\Sigma$ )
- $f_{\mathcal{A}} : (s_1, \dots, s_n) \mapsto f(s_1, \dots, s_n)$ ,  $f/n \in \Omega$

$$f_{\mathcal{A}}(\triangle, \dots, \triangle) =$$



# Herbrand Interpretations

---

In other words, *values are fixed* to be ground terms and *functions are fixed* to be the **term constructors**. Only predicate symbols  $p/m \in \Pi$  may be freely interpreted as relations  $p_{\mathcal{A}} \subseteq T_{\Sigma}^m$ .

## Proposition 2.12

Every set of ground atoms  $I$  uniquely determines a Herbrand interpretation  $\mathcal{A}$  via

$$(s_1, \dots, s_n) \in p_{\mathcal{A}} \quad :\Leftrightarrow \quad p(s_1, \dots, s_n) \in I$$

Thus we shall identify Herbrand interpretations (over  $\Sigma$ ) with sets of  $\Sigma$ -ground atoms.

# Existence of Herbrand Models

---

A Herbrand interpretation  $I$  is called a **Herbrand model** of  $F$ , if  $I \models F$ .

## Theorem 2.13

Let  $N$  be a set of  $\Sigma$ -clauses.

$$\begin{aligned} N \text{ satisfiable} &\Leftrightarrow N \text{ has a Herbrand model (over } \Sigma) \\ &\Leftrightarrow G_{\Sigma}(N) \text{ has a Herbrand model (over } \Sigma) \end{aligned}$$

where  $G_{\Sigma}(N) = \{C\sigma \text{ ground clause} \mid C \in N, \sigma : X \rightarrow T_{\Sigma}\}$  is the set of **ground instances** of  $N$ .

(Proof – completeness proof of resolution for first-order logic.)

# Consequences of Herbrans's theorem

---

## Decidability results.

Satisfiability of formulae in the Bernays-Schönfinkel Class

$$\exists^* \forall^*$$

is decidable.

# Consequences of Herbrans's theorem

---

## Decidability results.

Satisfiability of formulae in the Bernays-Schönfinkel Class

$$\exists^* \forall^*$$

is decidable.

$\Sigma = (\Omega, \Pi)$ ,  $\Omega$  is a finite set of constants

The Bernays-Schönfinkel class consists only of sentences of the form

$$\exists x_1 \dots \exists x_n \forall y_1 \dots \forall y_m F(x_1, \dots, x_n, y_1, \dots, y_m)$$

# Consequences of Herbrans's theorem

---

## Decidability results.

Satisfiability of formulae in the Bernays-Schönfinkel Class

$$\exists^* \forall^*$$

is decidable.

$\Sigma = (\Omega, \Pi)$ ,  $\Omega$  is a finite set of constants

The Bernays-Schönfinkel class consists only of sentences of the form

$$\exists x_1 \dots \exists x_n \forall y_1 \dots \forall y_m F(x_1, \dots, x_n, y_1, \dots, y_m)$$

**Idea:** Skolemization introduces only constants

$\mapsto$  no function symbols with arity  $\geq 1$

The Herbrand Universe is finite  $\mapsto$  decidability



# Tractable fragments of FOL

---

We showed that satisfiability of any finite set of ground Horn clauses can be checked in PTIME (linear time)

# Variable-free Horn clauses

---

## Data structures

Atoms  $P_1, \dots, P_n \mapsto \{1, \dots, n\}$

neg-occ-list( $A$ ): list of all clauses in which  $A$  occurs negatively

pos-occ-list( $A$ ): list of all clauses in which  $A$  occurs positively

|         |       |       |     |       |                    |
|---------|-------|-------|-----|-------|--------------------|
| Clause: | $P_1$ | $P_2$ | ... | $P_n$ | counter            |
|         | neg   | neg   |     | pos   | ↑                  |
|         |       | ↑     |     |       | number of literals |

**first-active-literal (fal)**: first literal not marked as deleted.

atom status: pos (deduced as positive unit clause)

neg (deduced as negative unit clause)

nounit (otherwise)

# Variable-free Horn clauses

---

Example on the blackboard:

$$N = \{ \quad P_4 \rightarrow P_6$$

$$\quad P_2 \rightarrow P_1$$

$$P_3, P_4 \rightarrow P_7$$

$$P_2, P_5 \rightarrow P_7$$

$$\quad \rightarrow P_4$$

$$P_2 \rightarrow$$

$$P_1 \rightarrow P_6$$

# Variable-free Horn clauses

---

**Input:** Set  $N$  of Horn formulae

**Step 1.** Collect unit clauses; check if complementary pairs exist

**forall**  $C \in N$  **do**

**if** is-unit( $C$ ) **then begin**

const. time

$L :=$  first-active-literal( $C$ )

const. time

**if** state(atom( $L$ )) = nunit **then** state(atom( $L$ )) = sign( $L$ ) const. time

        push(atom( $L$ ), stack)

**else if** state(atom( $L$ ))  $\neq$  sign( $L$ ) **then return false**

# Variable-free Horn clauses

---

## 2. Process the unit clauses in the stack

```
while stack  $\neq \emptyset$  do  
  begin A := top(stack); pop(stack)  
    if state(A) = pos then delete-literal-list := neg-oc-list(A)           O(# neg-oc-list)  
      else delete-literal-list := pos-oc-list(A)           O(# pos-oc-list)  
    endif  
    for all C in delete-literal-list do  
      if state(A) = pos then delete-literal(A,C)           const. time + nfal - ofal  
      if state(A) = neg then delete-literal( $\neg$  A,C)       const. time + nfal - ofal  
      if unit(C) then L1 := first-active-literal(C)         const. time  
        if state(atom(L1)) = nunit then state(atom(L1)) = sign(L1),  
          L1  $\rightarrow$  stack  
        elseif state(atom(L1))  $\neq$  sign(L1) then return false  
      endif  
    endif  
  end
```

# Tractable fragments of FOL

---

We showed that satisfiability of any finite set of ground Horn clauses can be checked in PTIME (linear time)

- Similar fragment of the Bernays-Schönfinkel class?

# Motivation: Deductive Databases

---

## Deductive database

Inference rules:

Facts:

Query:

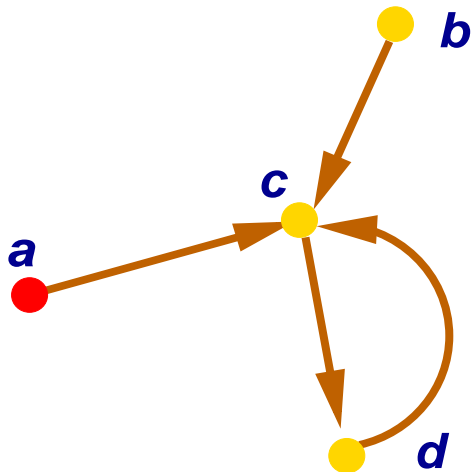
# Motivation: Deductive Databases

---

## Deductive database

Example: reachability in graphs

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| Inference rules: | $\frac{S(x)}{R(x)} \quad \frac{R(x) \quad E(x, y)}{R(y)}$ |
| Facts:           | $S(a), E(a, c), E(c, d), E(d, c), E(b, c)$                |
| Query:           | $R(d)$                                                    |



$S(a), E(a, c), E(c, d), E(d, c), E(b, c)$

**Note:**  $S, E$  stored relations (Extensional DB)

$R$  defined relation (Intensional DB)



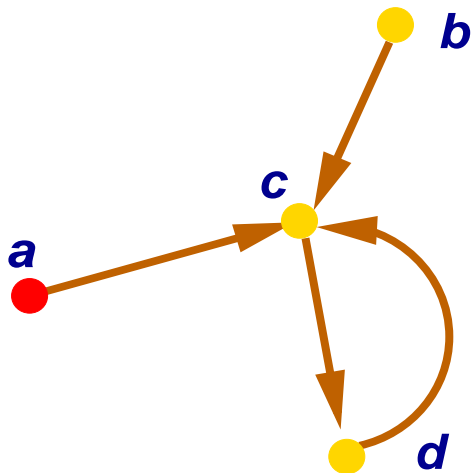
# Motivation: Deductive Databases

---

## Deductive database

Example: reachability in graphs

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| Inference rules: | $\frac{S(x)}{R(x)} \quad \frac{R(x) \quad E(x, y)}{R(y)}$ |
| Facts:           | $S(a), E(a, c), E(c, d), E(d, c), E(b, c)$                |
| Query:           | $R(d)$                                                    |



$S(a), E(a, c), E(a, d), E(c, d), E(b, c),$   
 $R(a)$

**Note:**  $S, E$  stored relations (Extensional DB)

$R$  defined relation (Intensional DB)

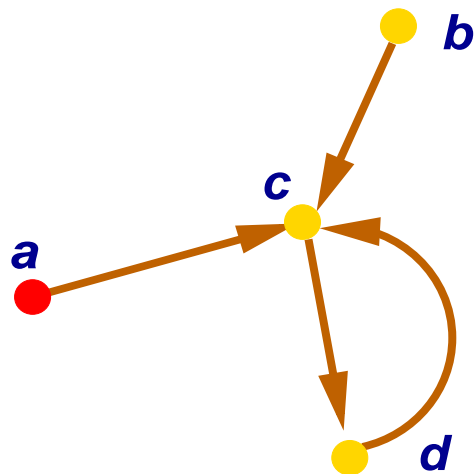
# Motivation: Deductive Databases

---

## Deductive database

Example: reachability in graphs

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| Inference rules: | $\frac{S(x)}{R(x)} \quad \frac{R(x) \quad E(x, y)}{R(y)}$ |
| Facts:           | $S(a), E(a, c), E(c, d), E(d, c), E(b, c)$                |
| Query:           | $R(d)$                                                    |



$S(a), E(a, c), E(a, d), E(c, d), E(b, c),$   
 $R(a), R(c)$

**Note:**  $S, E$  stored relations (Extensional DB)  
 $R$  defined relation (Intensional DB)

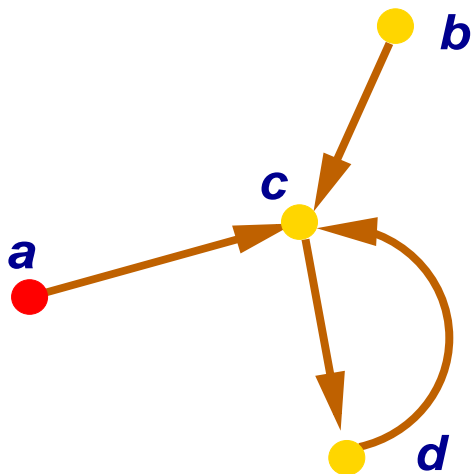
# Motivation: Deductive Databases

---

## Deductive database

Example: reachability in graphs

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| Inference rules: | $\frac{S(x)}{R(x)} \quad \frac{R(x) \quad E(x, y)}{R(y)}$ |
| Facts:           | $S(a), E(a, c), E(c, d), E(d, c), E(b, c)$                |
| Query:           | $R(d)$                                                    |



$S(a), E(a, c), E(a, d), E(c, d), E(b, c),$   
 $R(a), R(c), R(d)$

**Note:**  $S, E$  stored relations (Extensional DB)  
 $R$  defined relation (Intensional DB)

# Motivation: Deductive Databases

---

**Deductive database**  $\mapsto$  **Datalog** (Horn clauses, no function symbols)

|                  |                                                                                                                                   |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Inference rules: | $\underbrace{S(x) \rightarrow R(x) \quad R(x) \wedge E(x, y) \rightarrow R(y)}_{\text{set } \mathcal{K} \text{ of Horn clauses}}$ |
| Facts:           | $\underbrace{S(a), E(a, c), E(c, d), E(d, c), E(b, c)}_{\text{set } \mathcal{F} \text{ of ground atoms}}$                         |
| Query:           | $\underbrace{R(d)}_{\text{ground atom } G}$                                                                                       |

$$\mathcal{F} \models_{\mathcal{K}} G \quad \text{iff} \quad \mathcal{K} \cup \mathcal{F} \models G \quad \text{iff} \quad \mathcal{K} \cup \mathcal{F} \cup \neg G \models \perp$$

**Note:**  $S, E$  stored relations (Extensional DB)

$R$  defined relation (Intensional DB)

# Motivation: Deductive Databases

**Deductive database**  $\mapsto$  **Datalog** (Horn clauses, no function symbols)

|                  |                                                                                                                                   |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Inference rules: | $\underbrace{S(x) \rightarrow R(x) \quad R(x) \wedge E(x, y) \rightarrow R(y)}_{\text{set } \mathcal{K} \text{ of Horn clauses}}$ |
| Facts:           | $\underbrace{S(a), E(a, c), E(c, d), E(d, c), E(b, c)}_{\text{set } \mathcal{F} \text{ of ground atoms}}$                         |
| Query:           | $\underbrace{R(d)}_{\text{ground atom } G}$                                                                                       |

$$\frac{S(a) \quad S(x) \rightarrow R(x)}{R(a)}$$

$$\frac{R(a) \quad E(a, c) \quad R(x) \wedge E(x, y) \rightarrow R(y)}{R(c)}$$

$$R(c)$$

$$\frac{E(c, d) \quad R(x) \wedge E(x, y) \rightarrow R(y)}{R(d)}$$

$$R(d)$$

**Ex:**

# Ground entailment for function-free Horn clauses

---

## Assumption:

The signature does not contain function symbols of arity  $\geq 1$ .

## Given:

- Set  $H$  of (function-free) Horn clauses
- Ground Horn clause  $G = \bigwedge A_i \rightarrow A$ .

The following are equivalent:

- (1)  $H \models \bigwedge A_i \rightarrow A$
- (2)  $H \wedge \bigwedge A_i \models A$
- (3)  $H \wedge \bigwedge A_i \wedge \neg A \models \perp$

Decidable in PTIME in the size of  $G$  for a fixed  $H$ .

# Generalization: Local theories

---

[McAllester,Givan'92], [Basin,Ganzinger'96,01], [Ganzinger'01]

**Assumption:** the signature is allowed to contain function symbols

**Definition.**  $H$  set of Horn clauses is called **local** iff for every ground clause  $C$  the following are equivalent:

(1)  $H \models C$

(2)  $H[C] \models C$ ,

where  $H[C]$  is the family of all instances of  $H$  in which the variables are replaced by ground subterms occurring in  $H$  or  $C$ .

**Theorem.** For a fixed local theory  $H$ , testing ground entailment w.r.t.  $H$  is in PTIME.

Will be discussed in more detail in the exercises

## 2.7 General Resolution

---

Propositional resolution:

refutationally complete,

clearly inferior to the DPLL procedure  
(even with various improvements).

But: in contrast to the DPLL procedure, resolution can be easily extended to non-ground clauses.



# Propositional resolution: reminder

---

Resolution inference rule:

$$\frac{C \vee A \quad \neg A \vee D}{C \vee D}$$

Terminology:  $C \vee D$ : **resolvent**;  $A$ : **resolved atom**

(Positive) factorisation inference rule:

$$\frac{C \vee A \vee A}{C \vee A}$$

# Resolution for ground clauses

---

- Exactly the same as for propositional clauses

Ground atoms  $\mapsto$  propositional variables

## Theorem

Res is sound and refutationally complete (for all sets of ground clauses)

# Sample Refutation

---

1.  $\neg P(f(a)) \vee \neg P(f(a)) \vee Q(b)$  (given)
2.  $P(f(a)) \vee Q(b)$  (given)
3.  $\neg P(g(b, a)) \vee \neg Q(b)$  (given)
4.  $P(g(b, a))$  (given)
5.  $\neg P(f(a)) \vee Q(b) \vee Q(b)$  (Res. 2. into 1.)
6.  $\neg P(f(a)) \vee Q(b)$  (Fact. 5.)
7.  $Q(b) \vee Q(b)$  (Res. 2. into 6.)
8.  $Q(b)$  (Fact. 7.)
9.  $\neg P(g(b, a))$  (Res. 8. into 3.)
10.  $\perp$  (Res. 4. into 9.)

# Resolution for ground clauses

---

- Refinements with orderings and selection functions:

Need: - well-founded ordering on ground atomic formulae/literals  
- selection function (for negative literals)

$S : C \mapsto$  set of occurrences of *negative* literals in  $C$

Example of selection with selected literals indicated as  $\boxed{X}$ :

$$\boxed{\neg A} \vee \neg A \vee B$$

$$\boxed{\neg B_0} \vee \boxed{\neg B_1} \vee A$$

# Resolution Calculus $Res_S^\succ$

---

## Ordered resolution with selection

$$\frac{C \vee A \quad D \vee \neg A}{C \vee D}$$

if

1.  $A \succ C$ ;
2. nothing is selected in  $C$  by  $S$ ;
3.  $\neg A$  is selected in  $D \vee \neg A$ ,  
or else nothing is selected in  $D \vee \neg A$  and  $\neg A \succeq \max(D)$ .

**Note:** For positive literals,  $A \succ C$  is the same as  $A \succ \max(C)$ .

## Ordered factoring

$$\frac{C \vee A \vee A}{(C \vee A)}$$

if  $A$  is maximal in  $C$  and nothing is selected in  $C$ .

# Resolution for ground clauses

---

Let  $\succ$  be a total and well-founded ordering on ground atoms, and  $S$  a selection function.

**Theorem.**  $\text{Res}_S^\succ$  is sound and refutationally complete for all sets of ground clauses.

**Soundness:** sufficient to show that

$$(1) C \vee A, D \vee \neg A \models C \vee D$$

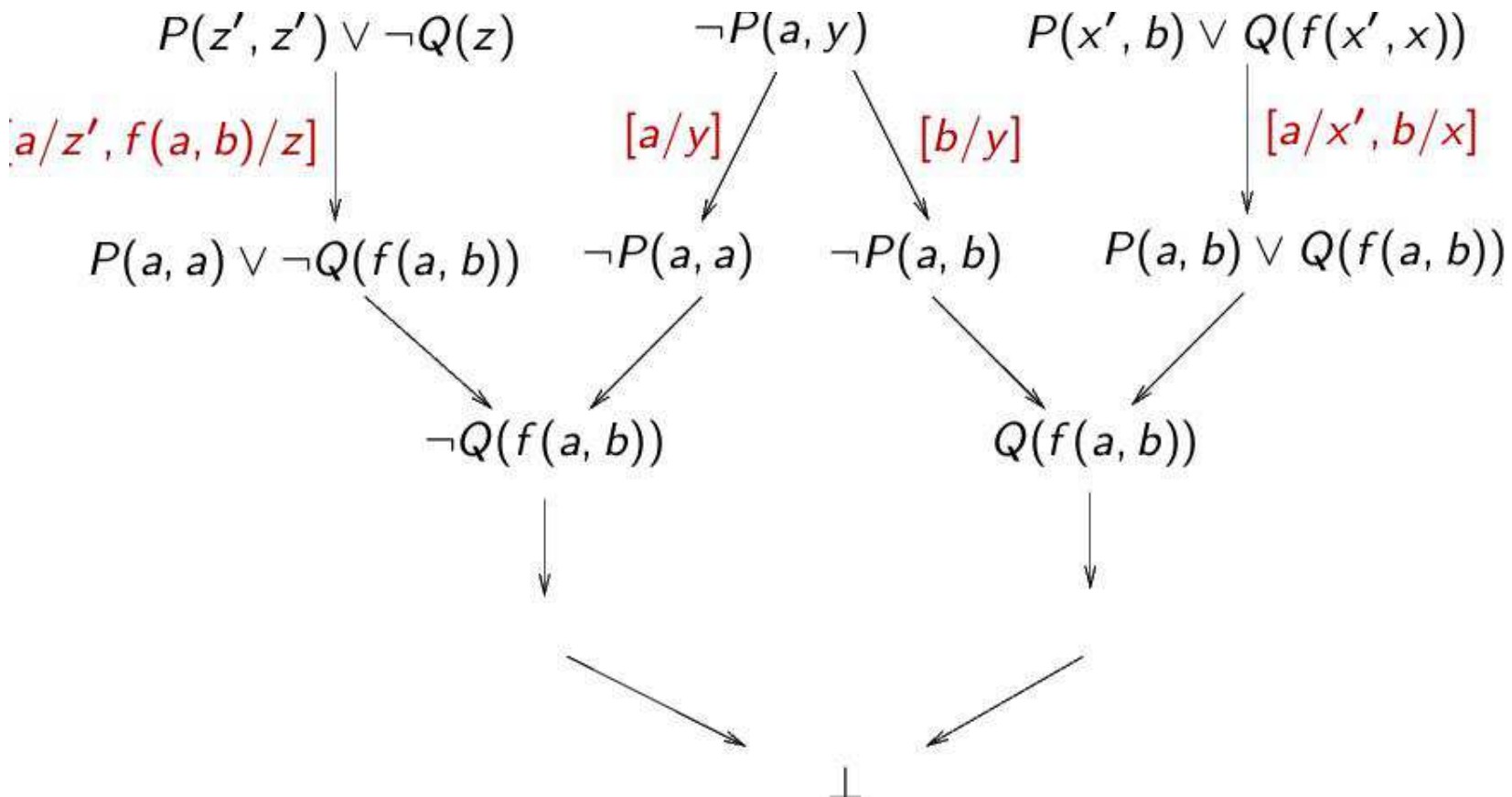
$$(2) C \vee A \vee A \models C \vee A$$

**Completeness:** Let  $\succ$  be a clause ordering, let  $N$  be saturated wrt.  $\text{Res}_S^\succ$ , and suppose that  $\perp \notin N$ . Then  $I_N^\succ \models N$ , where  $I_N^\succ$  is incrementally constructed as follows:

# General Resolution through Instantiation

---

Idea: instantiate clauses appropriately:



# General Resolution through Instantiation

---

Problems:

More than one instance of a clause can participate in a proof.

Even worse: There are infinitely many possible instances.

Observation:

Instantiation must produce complementary literals  
(so that inferences become possible).

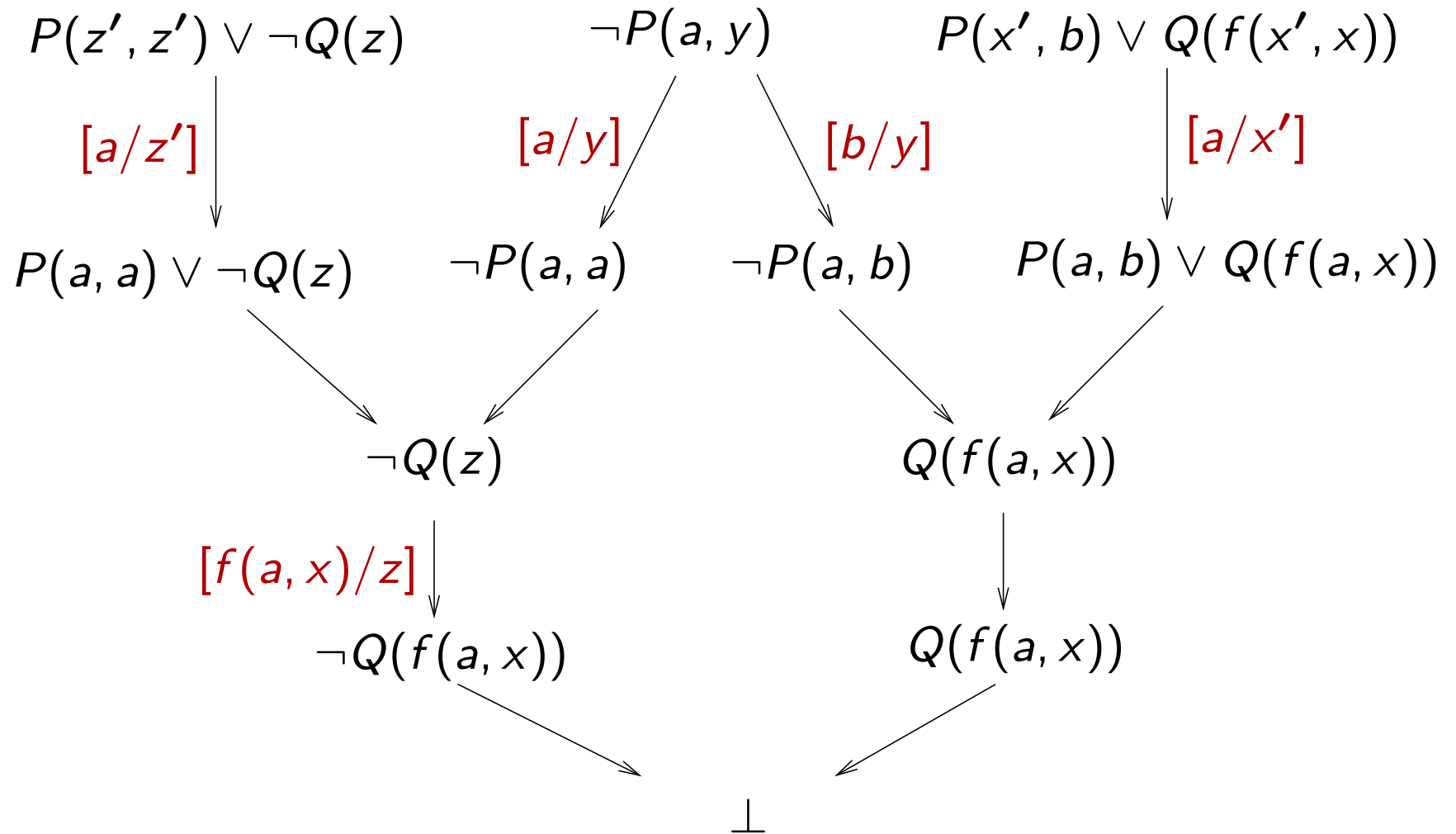
Idea:

Do not instantiate more than necessary to get complementary literals.



# General Resolution through Instantiation

Idea: do not instantiate more than necessary:



# Lifting Principle

---

**Problem:** Make saturation of infinite sets of clauses as they arise from taking the (ground) instances of finitely many **general** clauses (with variables) effective and efficient.

**Idea (Robinson 65):**

- Resolution for general clauses:
- *Equality* of ground atoms is generalized to *unifiability* of general atoms;
- Only compute *most general* (minimal) unifiers.

# Resolution for General Clauses

---

**General binary resolution** *Res*:

$$\frac{C \vee A \quad D \vee \neg B}{(C \vee D)\sigma} \quad \text{if } \sigma = \text{mgu}(A, B) \quad [\text{resolution}]$$

$$\frac{C \vee A \vee B}{(C \vee A)\sigma} \quad \text{if } \sigma = \text{mgu}(A, B) \quad [\text{factorization}]$$

For inferences with more than one premise, we assume that the variables in the premises are (bijectively) renamed such that they become different to any variable in the other premises.

We do not formalize this. Which names one uses for variables is otherwise irrelevant.

# Unification

---

Let  $E = \{s_1 \doteq t_1, \dots, s_n \doteq t_n\}$  ( $s_i, t_i$  terms or atoms) a multi-set of **equality problems**. A substitution  $\sigma$  is called a **unifier** of  $E$  if  $s_i\sigma = t_i\sigma$  for all  $1 \leq i \leq n$ .

If a unifier of  $E$  exists, then  $E$  is called **unifiable**.

# Unification after Martelli/Montanari

---

- (1)  $t \doteq t, E \Rightarrow_{MM} E$
- (2)  $f(s_1, \dots, s_n) \doteq f(t_1, \dots, t_n), E \Rightarrow_{MM} s_1 \doteq t_1, \dots, s_n \doteq t_n, E$
- (3)  $f(\dots) \doteq g(\dots), E \Rightarrow_{MM} \perp$
- (4)  $x \doteq t, E \Rightarrow_{MM} x \doteq t, E[t/x]$   
if  $x \in \text{var}(E), x \notin \text{var}(t)$
- (5)  $x \doteq t, E \Rightarrow_{MM} \perp$   
if  $x \neq t, x \in \text{var}(t)$
- (6)  $t \doteq x, E \Rightarrow_{MM} x \doteq t, E$   
if  $t \notin X$

# Examples

---

## Example 1:

$$\{x \doteq f(a), g(x, x) \doteq g(x, y)\} \Rightarrow 4$$

$$\{x \doteq f(a), g(f(a), f(a)) \doteq g(f(a), y)\} \Rightarrow 2$$

$$\{x \doteq f(a), f(a) \doteq f(a), f(a) \doteq y\} \Rightarrow 1$$

$$\{x \doteq f(a), f(a) \doteq y\} \Rightarrow 6$$

$$\{x \doteq f(a), y \doteq f(a)\}$$

## Example 2:

$$\{x \doteq f(a), g(x, x) \doteq h(x, y)\} \Rightarrow 3 \perp$$

## Example 3:

$$\{f(x, x) \doteq f(y, g(y))\} \Rightarrow 2$$

$$\{x \doteq y, x \doteq g(y)\} \Rightarrow 4$$

$$\{x \doteq y, y \doteq g(y)\} \Rightarrow 5 \perp$$

# MM: Main Properties

---

If  $E = x_1 \doteq u_1, \dots, x_k \doteq u_k$ , with  $x_i$  pairwise distinct,  $x_i \notin \text{var}(u_j)$ , then  $E$  is called an (equational problem in) **solved form** representing the solution  $\sigma_E = [u_1/x_1, \dots, u_k/x_k]$ .

## Proposition 2.28:

If  $E$  is a solved form then  $\sigma_E$  is an mgu of  $E$ .

## Theorem 2.29:

1. If  $E \Rightarrow_{MM} E'$  then  $\sigma$  is a unifier of  $E$  iff  $\sigma$  is a unifier of  $E'$
2. If  $E \Rightarrow_{MM}^* \perp$  then  $E$  is not unifiable.
3. If  $E \Rightarrow_{MM}^* E'$  with  $E'$  in solved form, then  $\sigma_{E'}$  is an mgu of  $E$ .

# Main Unification Theorem

---

## Theorem 2.30:

$E$  is unifiable if and only if there is a most general unifier  $\sigma$  of  $E$ , such that  $\sigma$  is idempotent and  $dom(\sigma) \cup codom(\sigma) \subseteq var(E)$ .

**Proof:** See e.g. Baader & Nipkow: Term rewriting and all that.

**Problem:** *exponential growth* of terms possible

### Example:

$$E = \{x_1 \approx f(x_0, x_0), x_2 \approx f(x_1, x_1), \dots, x_n \approx f(x_{n-1}, x_{n-1})\}$$

$$\text{m.g.u. } [x_1 \mapsto f(x_0, x_0), x_2 \mapsto f(f(x_0, x_0), f(x_0, x_0)), \dots]$$

$$x_i \mapsto \text{complete binart tree of heigth } i$$

**Solution:** Use acyclic term graphs; union/find algorithms



# Lifting Lemma

---

## Lemma 2.31

Let  $C$  and  $D$  be variable-disjoint clauses. If

$$\frac{\begin{array}{ccc} C & & D \\ \sigma \downarrow & & \rho \downarrow \\ C\sigma & & D\rho \end{array}}{C'}$$

[propositional resolution]

then there exists a substitution  $\tau$  such that

$$\frac{C \quad D}{C''}$$
$$\rho \downarrow$$
$$C' = C''\tau$$

[general resolution]

# Lifting Lemma

---

An analogous lifting lemma holds for factorization.

# Saturation of Sets of General Clauses

---

## Corollary 2.32:

Let  $N$  be a set of general clauses saturated under  $Res$ , i.e.,  $Res(N) \subseteq N$ .  
Then also  $G_{\Sigma}(N)$  is saturated, that is,

$$Res(G_{\Sigma}(N)) \subseteq G_{\Sigma}(N).$$

# Saturation of Sets of General Clauses

---

## Corollary 2.32:

Let  $N$  be a set of general clauses saturated under  $Res$ , i.e.,  $Res(N) \subseteq N$ . Then also  $G_{\Sigma}(N)$  is saturated, that is,

$$Res(G_{\Sigma}(N)) \subseteq G_{\Sigma}(N).$$

Proof:

W.l.o.g. we may assume that clauses in  $N$  are pairwise variable-disjoint. (Otherwise make them disjoint, and this renaming process changes neither  $Res(N)$  nor  $G_{\Sigma}(N)$ .)

Let  $C' \in Res(G_{\Sigma}(N))$ , meaning (i) there exist resolvable ground instances  $C\sigma$  and  $D\rho$  of  $N$  with resolvent  $C'$ , or else (ii)  $C'$  is a factor of a ground instance  $C\sigma$  of  $C$ .

Case (i): By the Lifting Lemma,  $C$  and  $D$  are resolvable with a resolvent  $C''$  with  $C''\tau = C'$ , for a suitable substitution  $\tau$ . As  $C'' \in N$  by assumption, we obtain that  $C' \in G_{\Sigma}(N)$ .

Case (ii): Similar.

# Herbrand's Theorem

---

## Lemma 2.33:

Let  $N$  be a set of  $\Sigma$ -clauses, let  $\mathcal{A}$  be an interpretation.

Then  $\mathcal{A} \models N$  implies  $\mathcal{A} \models G_\Sigma(N)$ .

## Lemma 2.34:

Let  $N$  be a set of  $\Sigma$ -clauses, let  $\mathcal{A}$  be a *Herbrand* interpretation.

Then  $\mathcal{A} \models G_\Sigma(N)$  implies  $\mathcal{A} \models N$ .

# Herbrand's Theorem

---

## Theorem 2.35 (Herbrand):

A set  $N$  of  $\Sigma$ -clauses is satisfiable if and only if it has a Herbrand model over  $\Sigma$ .

Proof:

The “ $\Leftarrow$ ” part is trivial. For the “ $\Rightarrow$ ” part let  $N \not\models \perp$ .

$$N \not\models \perp \Rightarrow \perp \notin Res^*(N) \quad (\text{resolution is sound})$$

$$\Rightarrow \perp \notin G_\Sigma(Res^*(N))$$

$$\Rightarrow I_{G_\Sigma(Res^*(N))} \models G_\Sigma(Res^*(N)) \quad (\text{Thm. 2.23; Cor. 2.32})$$

$$\Rightarrow I_{G_\Sigma(Res^*(N))} \models Res^*(N) \quad (\text{Lemma 2.34})$$

$$\Rightarrow I_{G_\Sigma(Res^*(N))} \models N \quad (N \subseteq Res^*(N))$$

# The Theorem of Löwenheim-Skolem

---

## Theorem 2.36 (Löwenheim–Skolem):

Let  $\Sigma$  be a countable signature and let  $S$  be a set of closed  $\Sigma$ -formulas. Then  $S$  is satisfiable iff  $S$  has a model over a countable universe.

Proof:

If both  $X$  and  $\Sigma$  are countable, then  $S$  can be at most countably infinite. Now generate, maintaining satisfiability, a set  $N$  of clauses from  $S$ . This extends  $\Sigma$  by at most countably many new Skolem functions to  $\Sigma'$ . As  $\Sigma'$  is countable, so is  $T_{\Sigma'}$ , the universe of Herbrand-interpretations over  $\Sigma'$ . Now apply Theorem 2.35.

# Refutational Completeness of General Resolution

---

## Theorem 2.37:

Let  $N$  be a set of general clauses where  $Res(N) \subseteq N$ . Then

$$N \models \perp \Leftrightarrow \perp \in N.$$

Proof:

Let  $Res(N) \subseteq N$ . By Corollary 2.32:  $Res(G_\Sigma(N)) \subseteq G_\Sigma(N)$

$$N \models \perp \Leftrightarrow G_\Sigma(N) \models \perp \quad (\text{Lemma 2.33/2.34; Theorem 2.35})$$

$$\Leftrightarrow \perp \in G_\Sigma(N) \quad (\text{propositional resolution sound and complete})$$

$$\Leftrightarrow \perp \in N$$