Decision Procedures in Verification

First-Order Logic (3)

26.11.2018

Viorica Sofronie-Stokkermans

e-mail: 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)

Algorithmic Problems

- Decidability/Undecidability
- Methods: Resolution
  - Normal Forms and Skolemization

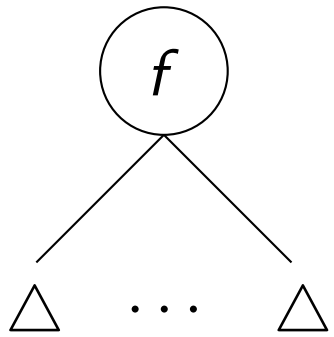
2.6 Herbrand Interpretations

From now an we shall consider PL without equality. Ω shall contains at least one constant symbol.

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

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

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



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 $p_{\mathcal{A}} \subseteq \mathrm{T}_{\Sigma}^{m}$.

Proposition 2.12

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

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

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

Herbrand Interpretations

Example: $\Sigma_{Pres} = (\{0/0, s/1, +/2\},\ \{</2, \leq/2\})$

$\mathbb{N}$ as Herbrand interpretation over Σ_{Pres}:

$$
\begin{aligned}
I = \{ \quad & 0 \leq 0,\ 0 \leq s(0),\ 0 \leq s(s(0)),\ \ldots, \\
& 0 + 0 \leq 0,\ 0 + 0 \leq s(0),\ \ldots, \\
& \ldots,\ (s(0) + 0) + s(0) \leq s(0) + (s(0) + s(0)) \\
& \ldots \\
& s(0) + 0 < s(0) + 0 + 0 + s(0) \\
& \ldots\}
\end{aligned}
$$

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 Σ-clauses.

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

where $G_\Sigma(N) = \{C\sigma \text{ ground clause} \mid C \in N,\ \sigma : X \to \mathrm{T}_\Sigma\}$ is the set of ground instances of N.

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

Example of a G_Σ

For Σ_{Pres} one obtains for

$$C = (x < y) \vee (y \leq s(x))$$

the following ground instances:

$(0 < 0) \vee (0 \leq s(0))$
$(s(0) < 0) \vee (0 \leq s(s(0)))$
...
$(s(0) + s(0) < s(0) + 0) \vee (s(0) + 0 \leq s(s(0) + s(0)))$
...

Consequences of Herbrans's theorem

Decidability results.

- Formulae without function symbols and without equality

  The Bernays-Schönfinkel Class $\exists^*\forall^*$

The Bernays-Schönfinkel Class

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

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

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

The Bernays-Schönfinkel Class

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

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

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

Idea: CNF translation:

$$\begin{aligned}
&\exists \overline{x}_1 \forall \overline{y}_1 F_1 \wedge \ldots \exists \overline{x}_n \forall \overline{y}_n F_n \\
&\qquad \Rightarrow_P \exists \overline{x}_1 \ldots \exists \overline{x}_n \forall \overline{y}_1 \ldots \forall \overline{y}_n F(\overline{x}_1, \ldots, \overline{x}_n, \overline{y}_1, \ldots, \overline{y}_n) \\
&\qquad \Rightarrow_S \forall \overline{y}_1 \ldots \forall \overline{y}_m F(\overline{c}_1, \ldots, \overline{c}_n, \overline{y}_1, \ldots, \overline{y}_n) \\
&\qquad \Rightarrow_K \forall \overline{y}_1 \ldots \forall \overline{y}_m \bigwedge \bigvee L_i((\overline{c}_1, \ldots, \overline{c}_n, \overline{y}_1, \ldots, \overline{y}_n)
\end{aligned}$$

$\overline{c}_1, \ldots, \overline{c}_n$ are tuples of Skolem constants

The Bernays-Schönfinkel Class

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

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

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

Idea: CNF translation:

$$\exists \overline{x}_1 \forall \overline{y}_1 F_1 \wedge \ldots \exists \overline{x}_n \forall \overline{y}_n F_n$$
$$\Rightarrow_K^* \forall \overline{y}_1 \ldots \forall \overline{y}_m \bigwedge \bigvee L_i((\overline{c}_1, \ldots, \overline{c}_n, \overline{y}_1, \ldots, \overline{y}_n)$$

$\overline{c}_1, \ldots, \overline{c}_n$ are tuples of Skolem constants

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 $P_1, \ldots, P_n \quad \mapsto \quad \{1, \ldots, n\}$

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	$\ldots$	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

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)) = nounit **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)) = nounit **then** state(atom(L1)) = sign(L1),
L1 $\rightarrow$ stack

elseif state(atom(L1)) $\neq$ sign(L1) **then** return false

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

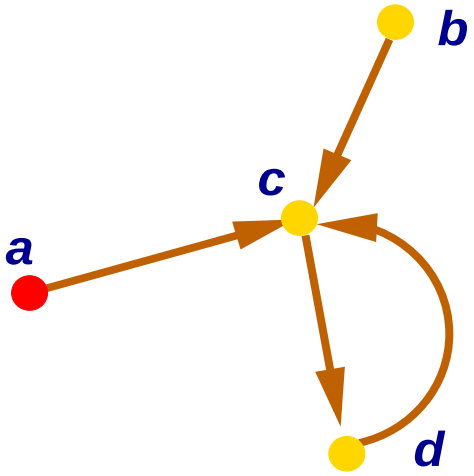
Deductive database

Inference rules:
Facts:
Query:

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

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



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

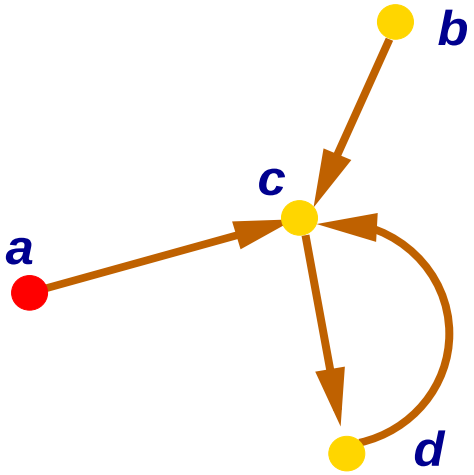
Note: S, E stored relations (Extensional DB)

R defined relation (Intensional DB)

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

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



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

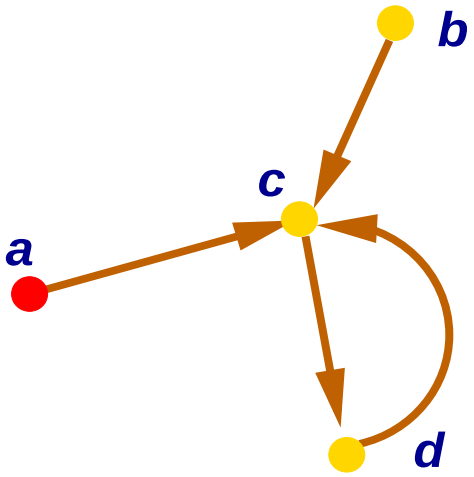
Note: S, E stored relations (Extensional DB)

R defined relation (Intensional DB)

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

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



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

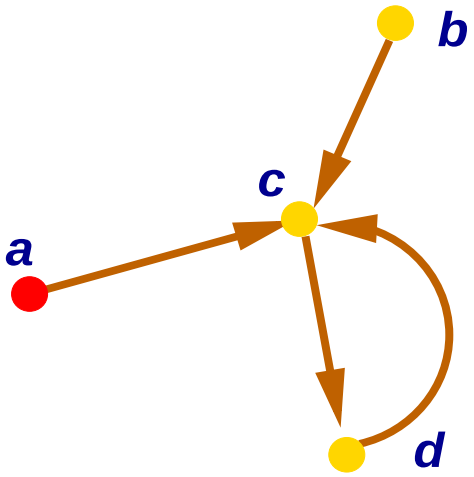
Note: S, E stored relations (Extensional DB)

R defined relation (Intensional DB)

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

Inference rules:	$\dfrac{S(x)}{R(x)}$ $\qquad$ $\dfrac{R(x) \quad E(x,y)}{R(y)}$
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:	$\underbrace{S(x) \to R(x) \qquad R(x) \wedge E(x,y) \to R(y)}_{\text{set } \mathcal{K} \text{ of Horn clauses}}$
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:	$\underbrace{R(d)}_{\text{ground atom } G}$

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

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:	$\underbrace{S(x) \to R(x) \quad R(x) \wedge E(x,y) \to R(y)}_{\text{set } \mathcal{K} \text{ of Horn clauses}}$
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:	$\underbrace{R(d)}_{\text{ground atom } G}$

Ex:

$$\dfrac{\dfrac{\dfrac{S(a) \quad S(x) \to R(x)}{R(a)} \quad E(a,c) \quad R(x) \wedge E(x,y) \to R(y)}{R(c)} \quad E(c,d) \quad R(x) \wedge E(x,y) \to R(}{R(d)}$$

Ground entailment for function-free Horn clauses

Assumption:

The signature does not contain function symbols of arity ≥ 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) $H \wedge \bigwedge A_i \wedge \neg A \models \bot$

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:

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

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.	$\bot$	(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)

  $S : C \quad \mapsto \quad$ 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

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

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

Soundness: sufficient to show that

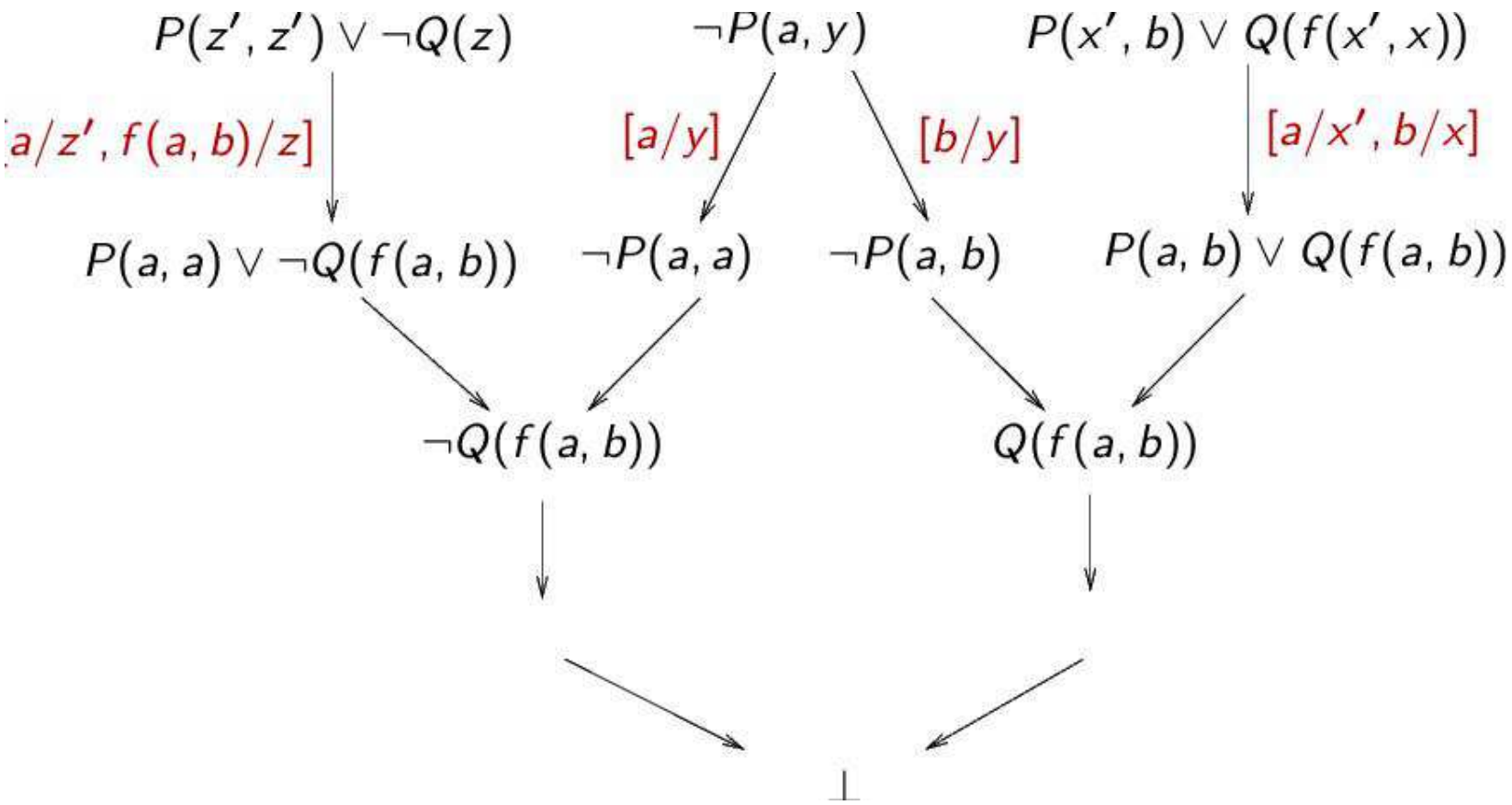
(1) $C \vee A, D \vee \neg A \models C \vee D$

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

Completeness: Let $\succ$ be a clause ordering, let N be saturated wrt. $Res_S^{\succ}$, and suppose that $\bot \notin N$. 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:

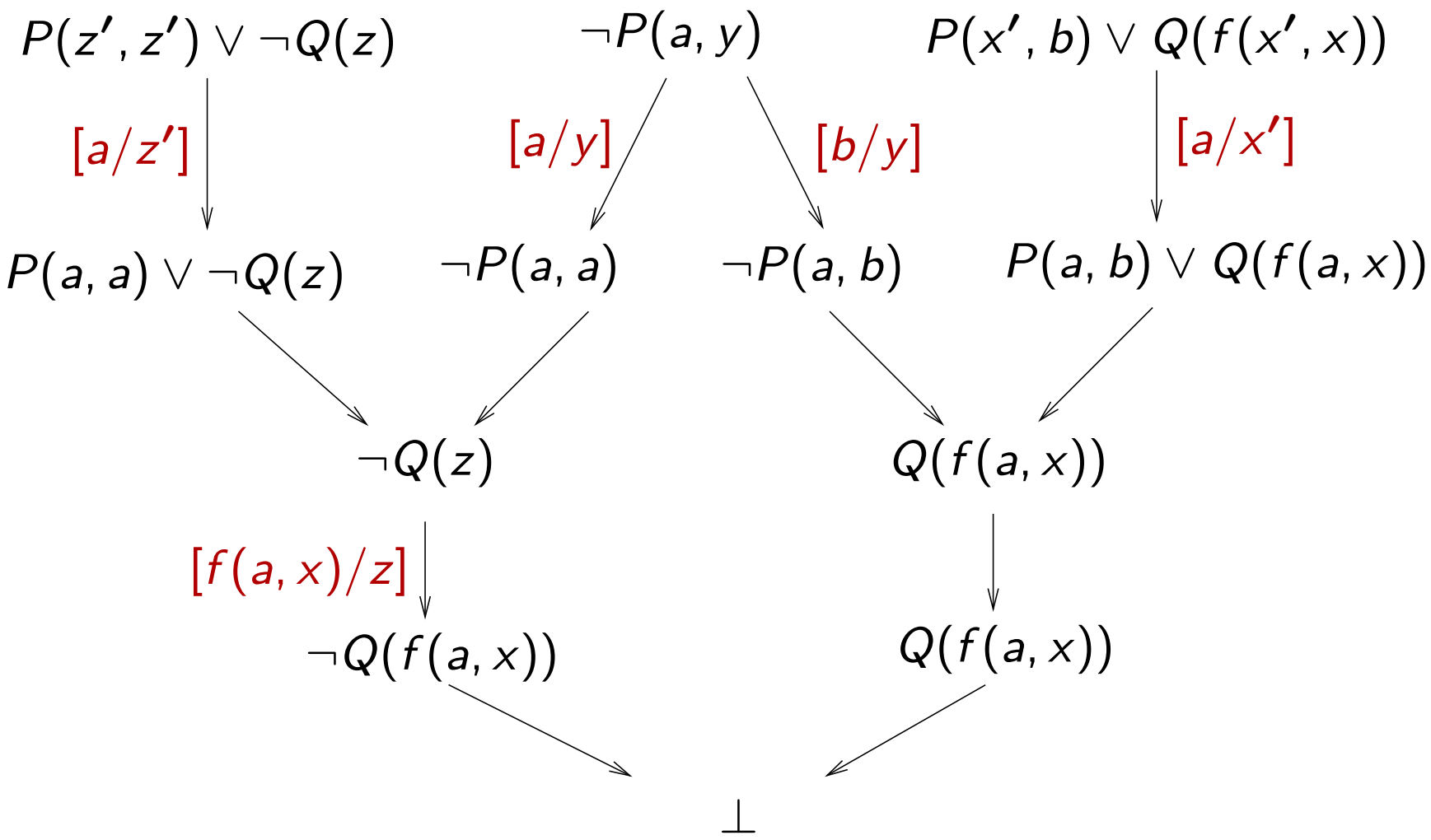
Instantiation must produce complementary literals
(so that inferences become possible).

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*:

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

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

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 $E = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (s_i, t_i terms or atoms) a multi-set of equality problems. A substitution σ 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

$$
\begin{array}{lrcl}
(1) & t \doteq t, E & \Rightarrow_{MM} & E \\
(2) & f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n), E & \Rightarrow_{MM} & s_1 \doteq t_1, \ldots, s_n \doteq t_n, E \\
(3) & f(\ldots) \doteq g(\ldots), E & \Rightarrow_{MM} & \bot \\
(4) & x \doteq t, E & \Rightarrow_{MM} & x \doteq t, E[t/x] \\
 & & & \quad \text{if } x \in var(E), x \notin var(t) \\
(5) & x \doteq t, E & \Rightarrow_{MM} & \bot \\
 & & & \quad \text{if } x \neq t, x \in var(t) \\
(6) & t \doteq x, E & \Rightarrow_{MM} & x \doteq t, E \\
 & & & \quad \text{if } t \notin X
\end{array}
$$

Examples

Example 1:

$$\{x \doteq f(a), g(x,x) \doteq g(x,y)\} \quad \Rightarrow_4$$
$$\{x \doteq f(a), g(f(a),f(a)) \doteq g(f(a),y)\} \quad \Rightarrow_2$$
$$\{x \doteq f(a), f(a) \doteq f(a), f(a) \doteq y\} \quad \Rightarrow_1$$
$$\{x \doteq f(a), f(a) \doteq y\} \quad \Rightarrow_6$$
$$\{x \doteq f(a), y \doteq f(a)\}$$

Example 2:

$$\{x \doteq f(a), g(x,x) \doteq h(x,y)\} \quad \Rightarrow_3 \bot$$

Example 3:

$$\{f(x,x) \doteq f(y,g(y))\} \quad \Rightarrow_2$$
$$\{x \doteq y, x \doteq g(y)\} \quad \Rightarrow_4$$
$$\{x \doteq y, y \doteq g(y)\} \quad \Rightarrow_5 \bot$$

MM: Main Properties

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

Proposition 2.28:

If E is a solved form then σ_E is am mgu of E.

Theorem 2.29:

1. If $E \Rightarrow_{MM} E'$ then σ is a unifier of E iff σ is a unifier of E'
2. If $E \Rightarrow^*_{MM} \bot$ 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 σ of E, such that σ 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:

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

m.g.u. $[x_1 \mapsto f(x_0, x_0), x_2 \mapsto f(f(x_0, x_0), f(x_0, x_0)), \ldots]$

$x_i \mapsto$ complete binart tree of heigth i

Solution: Use acyclic term graphs; union/find algorithms