Decision Procedures in Verification

First-Order Logic (3)

21.11.2013

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

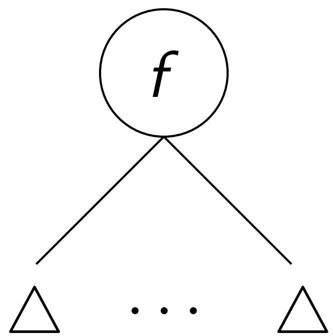
2.6 Herbrand Interpretations

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

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.

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

where $G_\Sigma(N) = \{C\sigma \text{ ground clause} \mid C \in N,\ \sigma : X \to \mathsf{T}_\Sigma\}$ 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:

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

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

$\ldots$

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

$\ldots$

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	$\uparrow$
		$\uparrow$			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 ≠ ∅ 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(¬ 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 → stack
                      elseif state(atom(L1)) ≠ 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

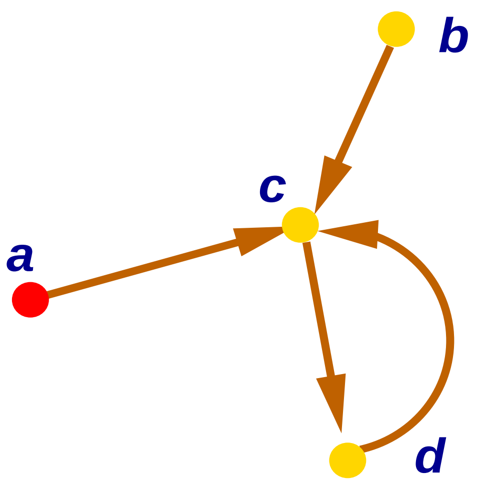
Deductive database

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:	$R(d)$



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

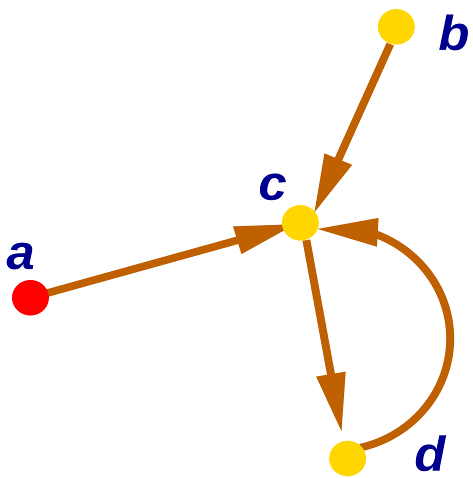
Note: S, E stored relations (Extensional DB)

R defined relation (Intensional DB)

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

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



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

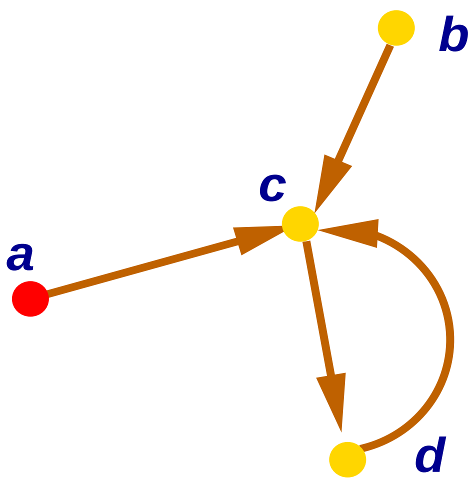
Note: S, E stored relations (Extensional DB)

R defined relation (Intensional DB)

Motivation: Deductive Databases

Deductive database Example: reachability in graphs

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



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

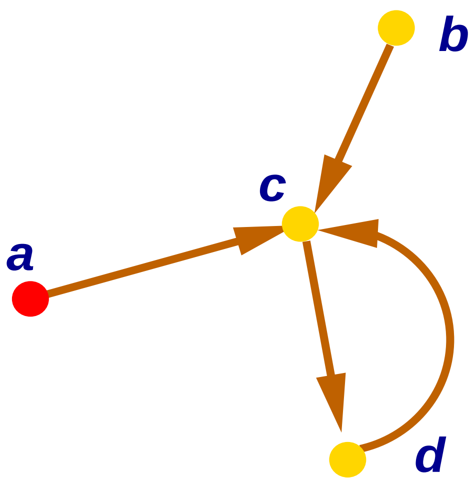
Note: S, E stored relations (Extensional DB)

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

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

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) $H \wedge \bigwedge A_i \wedge \neg A \models \perp$

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:

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

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.	$\perp$	(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)

  $$S : C \quad \mapsto \quad \text{set of occurrences of } \textit{negative} \text{ literals in } C$$

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

Soundness: sufficient to show that

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

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

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:

Construction of Candidate Models Formally

Let $N, \succ$ be given.

- Order N increasing w.r.t. the extension of $\succ$ to clauses.
- Define sets I_C and Δ_C for all ground clauses C over the given signature inductively over $\succ$:

$$I_C := \bigcup_{C \succ D} \Delta_D$$

$$\Delta_C := \begin{cases} \{A\}, & \text{if } C \in N,\ C = C' \vee A,\ A \succ C',\ I_C \not\models C \\ & \text{and nothing is selected in } C \\ \emptyset, & \text{otherwise} \end{cases}$$

We say that C produces A, if $\Delta_C = \{A\}$.

The candidate model for N (wrt. $\succ$) is given as $I_N^{\succ} := \bigcup_C \Delta_C$.

(We write I_N for $I_N^{\succ}$ if $\succ$ is irrelevant or known from the context.)

Completeness

Theorem. Let $\succ$ be a clause ordering, let N be saturated wrt. $Res_S^\succ$, and suppose that $\bot \notin N$. Then $I_N^\succ \models N$.

Proof: Suppose $\bot \notin N$, but $I_N^\succ \not\models N$. Let $C \in N$ minimal (in $\succ$) such that $I_N^\succ \not\models C$. Since C is false in I_N, C is not productive. As $C \neq \bot$ there exists a maximal atom A in C.

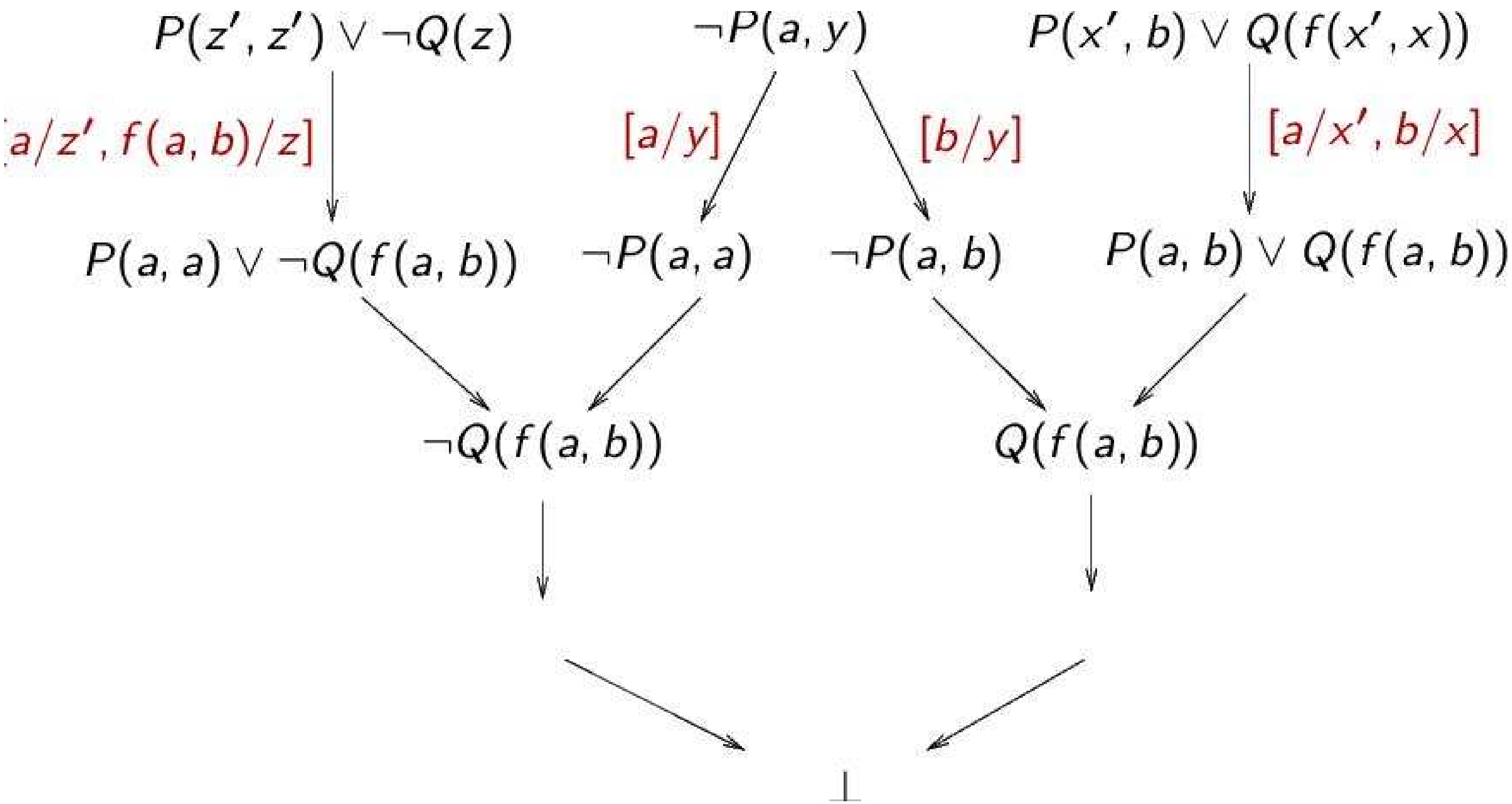
1. $C = \neg A \vee C'$ (maximal atom occurs negatively) $\Rightarrow I_N \models A$, $I_N \not\models C'$
   Then some $D = D' \vee A \in N$ produces A. As $\frac{D' \vee A \qquad \neg A \vee C'}{D' \vee C'}$, we infer that $D' \vee C' \in N$, and $C \succ D' \vee C'$ and $I_N \not\models D' \vee C'$
   $\Rightarrow$ contradicts minimality of C.

2. $C = \boxed{\neg A} \vee C'$ ($\neg A$ is selected) $\Rightarrow I_N \models A$, $I_N \not\models C'$
   The argument in 1. applies also in this case.

3. $C = C' \vee A \vee A$. Then $\frac{C' \vee A \vee A}{C' \vee A}$ yields a smaller counterexample $C' \vee A \in N$. $\Rightarrow$ contradicts minimality of C.

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:

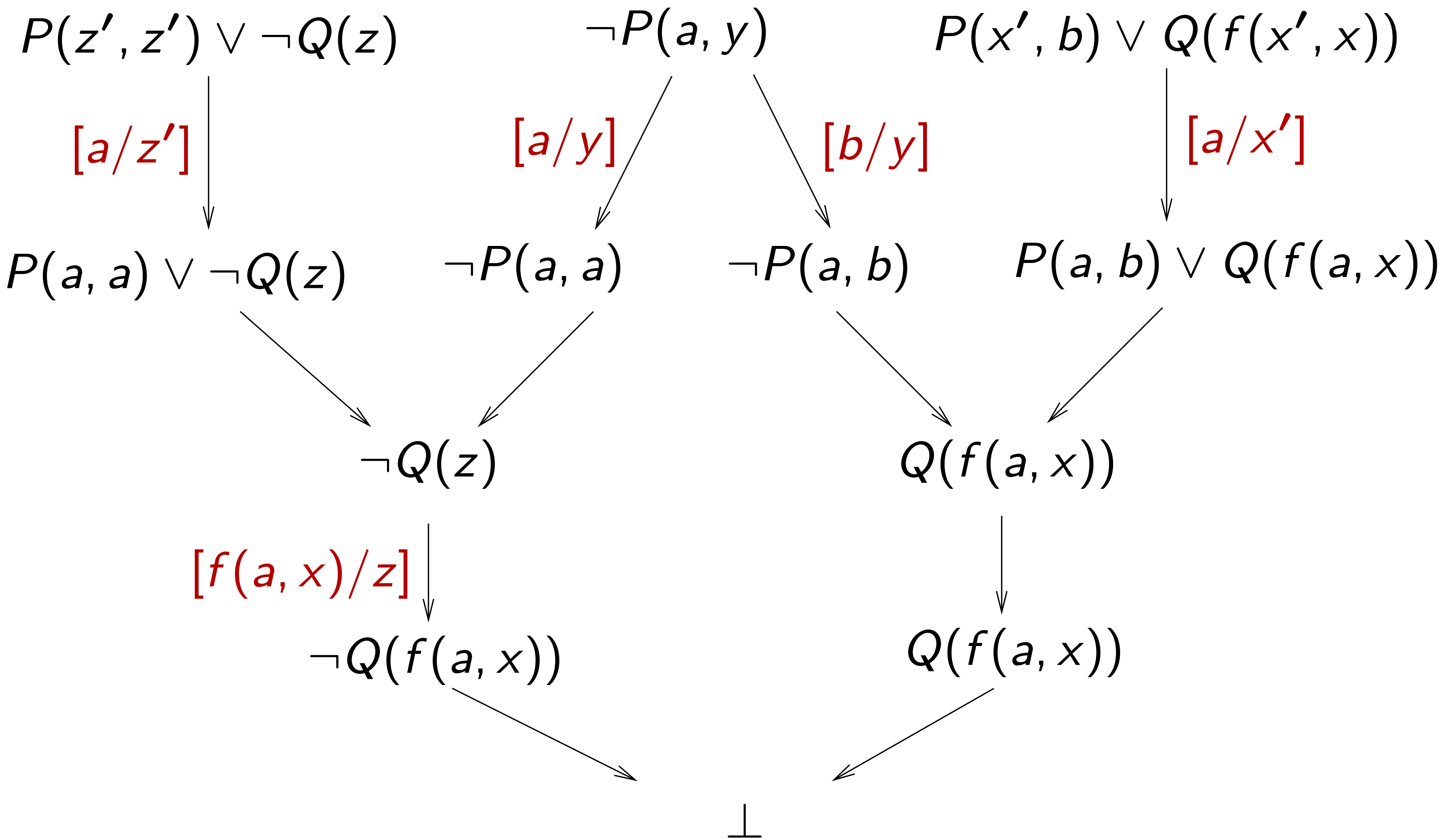
Instantiation must produce complementary literals
(so that inferences become possible).

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.

Lifting Principle

Significance: The advantage of the method in (Robinson 65) compared with (Gilmore 60) is that unification enumerates only those instances of clauses that participate in an inference. Moreover, clauses are not right away instantiated into ground clauses. Rather they are instantiated only as far as required for an inference. Inferences with non-ground clauses in general represent infinite sets of ground inferences which are computed simultaneously in a single step.

Resolution for General Clauses

General binary resolution ***Res*****:**

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

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

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 $E = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (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

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

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.

MM: Main Properties

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.

Proof:

(1) We have to show this for each of the rules. Let's treat the case for the 4th rule here. Suppose σ is a unifier of $x \doteq t$, that is, $x\sigma = t\sigma$. Thus, $\sigma \circ [t/x] = \sigma[x \mapsto t\sigma] = \sigma[x \mapsto x\sigma] = \sigma$. Therefore, for any equation $u \doteq v$ in E: $u\sigma = v\sigma$, iff $u[t/x]\sigma = v[t/x]\sigma$. (2) and (3) follow by induction from (1) using Proposition 2.28.

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. $[x_1 \mapsto f(x_0, x_0), x_2 \mapsto f(f(x_0, x_0), f(x_0, x_0)), ...]$

$x_i \mapsto$ 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

$$\begin{array}{ccc} C & & D \\ \sigma\downarrow & & \downarrow\rho \\ C\sigma & & D\rho \\ \hline & C' & \end{array} \qquad \text{[propositional resolution]}$$

then there exists a substitution τ such that

$$\begin{array}{ccc} C & & D \\ \hline & C'' & \\ & \rho\downarrow & \\ & C' = C''\tau & \end{array} \qquad \text{[general resolution]}$$

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

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 $\mathit{Res}(N)$ nor $G_\Sigma(N)$.)

Let $C' \in \mathit{Res}(G_\Sigma(N))$, 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 τ. 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 Σ-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 Σ-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 Σ-clauses is satisfiable if and only if it has a Herbrand model over Σ.

Proof:

The "$\Leftarrow$" part is trivial. For the "$\Rightarrow$" part let $N \not\models \bot$.

$$
\begin{aligned}
N \not\models \bot &\Rightarrow \bot \notin \mathit{Res}^*(N) && \text{(resolution is sound)} \\
&\Rightarrow \bot \notin G_\Sigma(\mathit{Res}^*(N)) \\
&\Rightarrow I_{G_\Sigma(\mathit{Res}^*(N))} \models G_\Sigma(\mathit{Res}^*(N)) && \text{(Thm. 2.23; Cor. 2.32)} \\
&\Rightarrow I_{G_\Sigma(\mathit{Res}^*(N))} \models \mathit{Res}^*(N) && \text{(Lemma 2.34)} \\
&\Rightarrow I_{G_\Sigma(\mathit{Res}^*(N))} \models N && (N \subseteq \mathit{Res}^*(N))
\end{aligned}
$$

The Theorem of Löwenheim-Skolem

Theorem 2.36 (Löwenheim–Skolem):
Let Σ be a countable signature and let S be a set of closed Σ-formulas. Then S is satisfiable iff S has a model over a countable universe.

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

$$N \models \bot \Leftrightarrow \bot \in N.$$

Proof:

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

$$
\begin{aligned}
N \models \bot &\Leftrightarrow G_\Sigma(N) \models \bot && \text{(Lemma 2.33/2.34; Theorem 2.35)}\\
&\Leftrightarrow \bot \in G_\Sigma(N) && \text{(propositional resolution sound and complete)}\\
&\Leftrightarrow \bot \in N
\end{aligned}
$$

Compactness of Predicate Logic

Theorem 2.38 (Compactness Theorem for First-Order Logic):
Let Φ be a set of first-order formulas.
Φ is unsatisfiable $\Leftrightarrow$ some finite subset $\Psi \subseteq \Phi$ is unsatisfiable.

Proof:
The "$\Leftarrow$" part is trivial. For the "$\Rightarrow$" part let Φ be unsatisfiable and let N be the set of clauses obtained by Skolemization and CNF transformation of the formulas in Φ. Clearly $Res^*(N)$ is unsatisfiable. By Theorem 2.37, $\bot \in Res^*(N)$, and therefore $\bot \in Res^n(N)$ for some $n \in \mathbb{N}$. Consequently, $\bot$ has a finite resolution proof B of depth $\leq n$. Choose Ψ as the subset of formulas in Φ such that the corresponding clauses contain the assumptions (leaves) of B.

2.12 Ordered Resolution with Selection

Motivation: Search space for *Res* *very* large.

Ideas for improvement:

1. In the completeness proof (Model Existence Theorem 2.23) one only needs to resolve and factor maximal atoms
   $\Rightarrow$ if the calculus is restricted to inferences involving maximal atoms, the proof remains correct
   $\Rightarrow$ *order restrictions*

2. In the proof, it does not really matter with which negative literal an inference is performed
   $\Rightarrow$ choose a negative literal don't-care-nondeterministically
   $\Rightarrow$ *selection*

Selection Functions

A selection function is a mapping

$$S : C \quad \mapsto \quad \text{set of occurrences of } \textit{negative} \text{ literals in } C$$

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

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

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

Resolution Calculus $Res_S^{\succ}$

In the completeness proof, we talk about (strictly) maximal literals of *ground* clauses.

In the non-ground calculus, we have to consider those literals that correspond to (strictly) maximal literals of ground instances:

Let $\succ$ be a total and well-founded ordering on ground atoms.
A literal L is called [strictly] maximal in a clause C if and only if there exists a ground substitution σ such that for all L' in C: $L\sigma \succeq L'\sigma$ $[L\sigma \succ L'\sigma]$.

Resolution Calculus $Res_S^{\succ}$

Let $\succ$ be an atom ordering and S a selection function.

$$\frac{C \vee A \qquad \neg B \vee D}{(C \vee D)\sigma}$$ [ordered resolution with selection]

if $\sigma = \mathrm{mgu}(A, B)$ and

(i) $A\sigma$ strictly maximal wrt. $C\sigma$;

(ii) nothing is selected in C by S;

(iii) either $\neg B$ is selected,
or else nothing is selected in $\neg B \vee D$ and $\neg B\sigma$ is maximal in $D\sigma$.

Resolution Calculus $Res_S^{\succ}$

$$\frac{C \vee A \vee B}{(C \vee A)\sigma} \qquad \text{[ordered factoring]}$$

if $\sigma = \text{mgu}(A, B)$ and $A\sigma$ is maximal in $C\sigma$ and nothing is selected in C.

Soundness and Refutational Completeness

Theorem 2.39:

Let $\succ$ be an atom ordering and S a selection function such that $Res_S^{\succ}(N) \subseteq N$. Then

$$N \models \bot \Leftrightarrow \bot \in N$$

Proof:

The "$\Leftarrow$" part is trivial. For the "$\Rightarrow$" part consider first the propositional level: Construct a candidate model I_N as for unrestricted resolution, except that clauses C in N that have selected literals are not productive, even when they are false in I_C and when their maximal atom occurs only once and positively.

The result for general clauses follows using Corollary 2.40.

Craig Interpolation

A theoretical application of ordered resolution is Craig- Interpolation:

Theorem 2.42 (Craig 57)

Let F and G be two propositional formulas such that $F \models G$.

Then there exists a formula H (called the interpolant for $F \models G$), such that H contains only propostional variables occurring both in F and in G, and such that $F \models H$ and $H \models G$.

Craig Interpolation

Proof:

Translate F and $\neg G$ into CNF.

Let N and M, resp., denote the resulting clause set.

Choose an atom ordering $\succ$ for which the propositional variables that occur in F but not in G are maximal.

Saturate N into N^* wrt. $\text{Res}_S^{\succ}$ with an empty selection function S.

Then saturate $N^* \cup M$ wrt. $\text{Res}\succ_S$ to derive $\bot$.

As N^* is already saturated, due to the ordering restrictions only inferences need to be considered where premises, if they are from N^*, only contain symbols that also occur in G.

The conjunction of these premises is an interpolant H.

The theorem also holds for first-order formulas. For universal formulas the above proof can be easily extended. In the general case, a proof based on resolution technology is more complicated because of Skolemization.

Redundancy

So far: local restrictions of the resolution inference rules using orderings and selection functions.

Is it also possible to delete clauses altogether?
Under which circumstances are clauses unnecessary?
(Conjecture: e. g., if they are tautologies or if they are subsumed by other clauses.)

Intuition: If a clause is guaranteed to be neither a minimal counterexample nor productive, then we do not need it.

A Formal Notion of Redundancy

Let N be a set of ground clauses and C a ground clause (not necessarily in N). C is called redundant w. r. t. N, if there exist $C_1, \dots, C_n \in N$, $n \geq 0$, such that $C_i \prec C$ and $C_1, \dots, C_n \models C$.

Redundancy for general clauses:
C is called redundant w. r. t. N, if all ground instances $C\sigma$ of C are redundant w. r. t. $G_\Sigma(N)$.

Intuition: Redundant clauses are neither minimal counterexamples nor productive.

Note: The same ordering $\succ$ is used for ordering restrictions and for redundancy (and for the completeness proof).

Examples of Redundancy

Proposition 2.40:

- C tautology (i.e., $\models C$) $\Rightarrow$ C redundant w. r. t. any set N.
- $C\sigma \subset D$ $\Rightarrow$ D redundant w. r. t. $N \cup \{C\}$
- $C\sigma \subseteq D$ $\Rightarrow$ $D \vee \overline{L}\sigma$ redundant w. r. t. $N \cup \{C \vee L,\ D\}$

(Under certain conditions one may also use non-strict subsumption, but this requires a slightly more complicated definition of redundancy.)

Saturation up to Redundancy

N is called saturated up to redundancy (wrt. $Res_S^{\succ}$)

$$:\Leftrightarrow Res_S^{\succ}(N \setminus Red(N)) \subseteq N \cup Red(N)$$

Theorem 2.41:

Let N be saturated up to redundancy. Then

$$N \models \bot \Leftrightarrow \bot \in N$$

Saturation up to Redundancy

Proof (Sketch):

(i) Ground case:

- consider the construction of the candidate model $I_N^{\succ}$ for $Res_S^{\succ}$
- redundant clauses are not productive
- redundant clauses in N are not minimal counterexamples for $I_N^{\succ}$

The premises of "essential" inferences are either minimal counterexamples or productive.

(ii) Lifting: no additional problems over the proof of Theorem 2.39.

Monotonicity Properties of Redundancy

Theorem 2.42:

(i) $N \subseteq M \Rightarrow Red(N) \subseteq Red(M)$

(ii) $M \subseteq Red(N) \Rightarrow Red(N) \subseteq Red(N \setminus M)$

Proof: Exercise.

We conclude that redundancy is preserved when, during a theorem proving process, one adds (derives) new clauses or deletes redundant clauses.