Foundations of Data and Knowledge Systems

VU 181.212, WS 2010

3. Foundations of Automated Theorem Proving

Thomas Eiter and Reinhard Pichler

Institut für Informationssysteme
Technische Universität Wien

16 November, 2010

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Roadmap

Motivation

- This part of the lecture is based on the following book: Alexander Leitsch: *The Resolution Calculus*, Texts in Theoretical Computer Science, Springer-Verlag Berlin, Heidelberg, New York, 1997.
- Several fundamental results on First-Order Predicate Logic have been stated without proof in the first part of this lecture, like the Completeness Theorem, the Compactness Theorem, and the Löwenheim-Skolem Theorem.
- We proceed in the spirit of Automated Theorem Proving and first prove Herbrand's Theorem. It is then easy to prove the other results.
- In the article of Bry et al., the argumentation is in the opposite direction: Herbrand's Theorem is obtained as an easy consequence of the Compactness Theorem which in turn follows easily from the Completeness Theorem (which is stated without proof).

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Substitutions

Definition (Substitution)

A substitution is a function σ, written in postfix notation, that maps terms to terms and is

- homomorphous, i.e., $f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma)$ for compound terms and $c\sigma = c$ for constants.
- identical almost everywhere, i.e., $\{x \mid x \text{ is a variable and } x\sigma \neq x\}$ is finite.

The domain of a substitution σ is the finite set of variables on which it is not identical. Its codomain is the set of terms to which it maps its domain.

A substitution σ is represented by the finite set $\{x_1 \mapsto x_1\sigma, \ldots, x_k \mapsto x_k\sigma\}$ where $\{x_1, \ldots, x_k\}$ is its domain and $\{x_1\sigma, \ldots, x_k\sigma\}$ is its codomain.

(Ground) Instances

Definition (Ground substitution, ground instance)

A ground substitution is a substitution whose codomain consists of ground terms only. A grounding substitution for a term t is a ground substitution σ whose domain includes all variables in t, such that $t\sigma$ is ground. A ground instance of t is an instance of t that is ground.

Definition (Instance of a formula)

Let φ be a formula and σ a ground substitution. Then $\varphi\sigma$ is the formula obtained from φ by replacing each free variable occurrence x in φ by $x\sigma$.

Definition (Ground instance of a formula)

Let φ be a formula. Let φ' be a rectified form of φ. Let φ'' be obtained from φ' by removing each occurrence of a quantifier for a variable. A ground instance of φ is a ground instance of φ''.

Unification

Definition (Unification)

Two terms s and t are unifiable, if there exists a substitution σ with $s\sigma = t\sigma$. In this case σ is called a unifier of s and t.
A most general unifier or mgu of s and t is a unifier σ, s.t. for any other unifier σ' of s and t, there exists a substitution ϑ with $\sigma' = \sigma\vartheta$.
If σ is a most general unifier of s and t, then the term $s\sigma$ is called a most general common instance of s and t.

Remarks

- For any two terms s and t, if they are unifiable, then there exists an mgu of s and t. In this case, the most general common instance is unique up to variable renaming.
- Testing if s and t are unifiable and, if so, computing an mgu can be done efficiently. However, care is required concerning the used data structures (e.g., dag representation rather than string representation of terms).

Example

$s = h(x_1, x_2, \ldots, x_n)$
$t = h(f(x_0, x_0), f(x_1, x_1), \ldots, f(x_{n-1}, x_{n-1}))$
Unification of s and t yields the mgu
$\{x_1 \mapsto f(x_0, x_0), x_2 \mapsto f(f(x_0, x_0), f(x_0, x_0)), \ldots\}$

Remarks

- The mgu maps each x_i to a complete binary tree of height i.
- The size of the mgu (represented as a string or a tree) is exponential in the size of the input due to copying (or duplication) of (sub)terms.
- Better alternative. Represent terms as directed acyclic graphs.
- Intuition. Build up a substitution by collecting a list of bindings without duplicating terms, i.e. $\{x_1 \mapsto f(x_0, x_0), x_2 \mapsto f(x_1, x_1), \ldots\}$

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Overview of the Transformation

Transformation

Every formula φ can be transformed into a formula ψ, s.t.

- ψ is a set (or conjunction) of closed, universally quantified clauses
- φ and ψ are sat-equivalent, i.e., φ is satisfiable iff ψ is satisfiable.

The transformation proceeds in two steps:

1. Transformation into prenex form whose matrix is in CNF.
2. Skolemization: eliminate all existential quantifiers

Theorem

Every formula is equivalent to a formula in prenex form whose matrix is in conjunctive normal form (CNF).

Model-preserving Transformations

Proposition (Model-preserving transformations)

Let φ, φ', ψ, ψ', χ be formulas. The following equivalences hold:

- $\varphi \models\!\dashv \varphi'$ *if* φ' *is a rectified form of* φ
- $\varphi \models\!\dashv \varphi'$ *if* $\psi \models\!\dashv \psi'$ *and* φ' *is obtained from* φ *by replacing an occurrence of the subformula* ψ *by* ψ'
- $((\varphi \vee \psi) \wedge \chi) \models\!\dashv ((\varphi \wedge \chi) \vee (\psi \wedge \chi))$, $((\varphi \wedge \psi) \vee \chi) \models\!\dashv ((\varphi \vee \chi) \wedge (\psi \vee \chi))$
- $(\bot \vee \varphi) \models\!\dashv \varphi$, $(\top \wedge \varphi) \models\!\dashv \varphi$, $(\varphi \vee \neg\varphi) \models\!\dashv \top$, $(\varphi \wedge \neg\varphi) \models\!\dashv \bot$
- $(\varphi \vee \varphi) \models\!\dashv \varphi$, $(\varphi \wedge \varphi) \models\!\dashv \varphi$, $\neg\neg\varphi \models\!\dashv \varphi$
- $\neg(\varphi \vee \psi) \models\!\dashv (\neg\varphi \wedge \neg\psi)$, $\neg(\varphi \wedge \psi) \models\!\dashv (\neg\varphi \vee \neg\psi)$
- $(\varphi \Rightarrow \psi) \models\!\dashv (\neg\varphi \vee \psi)$
- $\forall x \forall y \varphi \models\!\dashv \forall y \forall x \varphi$, $\exists x \exists y \varphi \models\!\dashv \exists y \exists x \varphi$, $\neg\forall x \varphi \models\!\dashv \exists x \neg\varphi$, $\neg\exists x \varphi \models\!\dashv \forall x \neg\varphi$
- $\exists x(\varphi \wedge \psi) \models\!\dashv (\varphi \wedge \exists x \psi)$ *and* $\forall x(\varphi \wedge \psi) \models\!\dashv (\varphi \wedge \forall x \psi)$ *if* x *is not free in* φ

Skolemization

Definition (Skolemization step)

Let φ be a closed formula in prenex form and let $\exists x$ be the outermost existential quantifier in the quantifier prefix. Moreover, suppose that $\exists x$ is in the scope of the universal quantifiers $\forall y_1 \ldots \forall y_k$ with $k \geq 0$.

Let f be a k-ary function symbol that does not occur in φ. Let φ_s be φ, s.t. $\exists x$ is dropped and every occurrence of x in the matrix of φ is replaced by $f(y_1, \ldots, y_k)$. Then the transformation from φ to φ_s is a Skolemization step with Skolem function symbol f and Skolem term $f(y_1, \ldots, y_k)$.

Proposition

If a Skolemization step transforms φ *to* φ_s*, then* $\varphi_s \models \varphi$*, and for each interpretation* $\mathcal{I}$ *with* $\mathcal{I} \models \varphi$ *there exists an interpretation* $\mathcal{I}'$ *with* $\mathcal{I}' \models \varphi$ *and* $\mathcal{I}' \models \varphi_s$*. Moreover,* $\mathcal{I}'$ *coincides with* $\mathcal{I}$ *except possibly* $f^{\mathcal{I}'} \neq f^{\mathcal{I}}$.

Clause Form

Notation

- A clause is either the empty clause (denoted by $\Box$) or a disjunction of literals. The variables in a clause are thought of as universally quantified.
- A universally quantified formula in CNF can therefore be represented as a clause set. The latter representation is more common in automated deduction (and will be used in the remainder of this part of the lecture).

Example

Let $F_0 = \neg[(\forall x \exists y P(x, g(y, f(x))) \wedge \exists z Q(z)) \vee \exists y \forall x R(x, y))]$.
The transformation into clause form can be done as follows:

$F_1 = \neg[(\forall x \exists y P(x, g(y, f(x))) \wedge \exists z Q(z)) \vee \exists v \forall u R(u, v))]$ (rectification)
$F_2 = (\exists x \forall y \neg P(x, g(y, f(x))) \vee \forall z \neg Q(z)) \wedge \forall v \exists u \neg R(u, v)$ (shift $\neg$)
$F_3 = \exists x \forall y \forall z \forall v \exists u (\neg P(x, g(y, f(x))) \vee \neg Q(z)) \wedge \neg R(u, v)$ (prenex form)
$F_4 = \forall y \forall z \forall v (\neg P(a, g(y, f(a))) \vee \neg Q(z)) \wedge \neg R(h(y, z, v), v)$ (Skolemization)

$\mathcal{C} = \{C_1, C_2\}$ with $C_1 = \neg P(a, g(y, f(a))) \vee \neg Q(z)$, $C_2 = \neg R(h(y, z, v), v))$.

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Herbrand Interpretations

Motivation

- An interpretation according to Tarski's model theory may use any nonempty set as its domain. This makes it apparently incomputable.
- Herbrand interpretations have as domain the so-called Herbrand universe, the set of all ground terms constructible with the signature considered.
- Some important properties of Herbrand interpretations:
  - Ground terms are interpreted "by themselves". An Herbrand interpretation for a closed formula is therefore fully characterized by the set of ground atoms that are true in it.
  - In an Herbrand interpretation, quantification reduces to ground instantiation (i.e., semantics can be expressed in terms of syntax).
  - Result due to Jacques Herbrand: If a *universal formula* is true in any interpretation, then this formula is also true in an *Herbrand interpretation*.

Herbrand Interpretations

Definition (Herbrand universe and base)

Let $\mathcal{L}$ be a signature for first-order predicate logic. The Herbrand universe $HU_{\mathcal{L}}$ is the set of all ground $\mathcal{L}$-terms. The Herbrand base $HB_{\mathcal{L}}$ is the set of all ground $\mathcal{L}$-atoms.

(We assume that $\mathcal{L}$ specifies at least one constant.)

Definition (Herbrand interpretation)

An interpretation $\mathcal{I}$ is an Herbrand interpretation if $dom(\mathcal{I}) = HU$ and $f^{\mathcal{I}}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$ for each n-ary function symbol f and all $t_1, \ldots, t_n \in HU$.

Semantics vs. Syntax

Theorem

Let $\mathcal{I}$ be an Herbrand interpretation and φ a formula:

- $\mathcal{I} \models \forall x \varphi$ *iff* $\mathcal{I} \models \varphi\{x \mapsto t\}$ *for each* $t \in HU$.
- $\mathcal{I} \models \exists x \varphi$ *iff* $\mathcal{I} \models \varphi\{x \mapsto t\}$ *for at least one* $t \in HU$.

i.e., the effect of modifying the interpretation's variable assignment can be achieved by applying the ground substitution $\{x \mapsto t\}$ *to* φ.

Corollary

Let S be a set of universal closed formulas and S_{ground} the set of all ground instances of members of S.

For each Herbrand interpretation $\mathcal{I}$ we have: $\mathcal{I} \models S$ iff $\mathcal{I} \models S_{ground}$.

Representation of Herbrand Interpretations

Definition (Herbrand interpretation given by a set of ground atoms)

Let V be some fixed variable assignment in HU. Let $B \subseteq HB$ be a set of ground atoms. Then $HI(B)$ is the Herbrand interpretation with variable assignment V and $p^{HI(B)} = \{(t_1, \ldots, t_n) \mid p(t_1, \ldots, t_n) \in B\}$ for each n-ary relation symbol p.

Definition (Herbrand interpretation induced by an interpretation)

Let $\mathcal{I}$ be an arbitrary interpretation. The Herbrand interpretation induced by $\mathcal{I}$, denoted $HI(\mathcal{I})$, is $HI(\{A \in HB \mid \mathcal{I} \models A\})$.

Theorem (Herbrand model induced by a model)

Let φ be a universal closed formula. Each model of φ induces an Herbrand model of φ, that is, for each interpretation $\mathcal{I}$, if $\mathcal{I} \models \varphi$ then $HI(\mathcal{I}) \models \varphi$.

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Motivating Example

Example

Consider the clause set $\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(a))$, $C_2 = \neg P(u, v) \vee Q(f(v))$ and $C_3 = \neg Q(z)$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, Q. We claim that $\mathcal{C}$ is unsatisfiable.

Suppose that $\mathcal{C}$ has an Herbrand interpretation $\mathcal{I}$ with $B = \{A \in HB \mid \mathcal{I} \models A\}$. Since $C_3 = \neg Q(z)$ is true in $\mathcal{I}$, we have $Q(t) \notin B$ for each ground term $t \in HB_{\mathcal{L}}$. Hence, since $C_2 = \neg P(u, v) \vee Q(f(v))$ is true in $\mathcal{I}$, we have $P(s, t) \notin B$ for any two ground terms $s, t \in HB_{\mathcal{L}}$.

But then $C_1 = P(x, f(a))$ is false in $\mathcal{I}$, which is a contradiction.

Remark

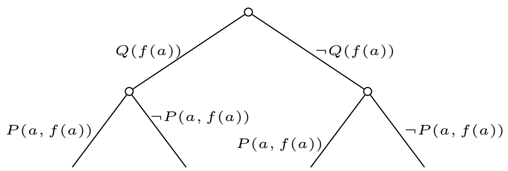
The above argument yields a semantical counterpart of a resolution-based refutation. This technique of excluding interpretations from being models can be systematized and represented in the form of so-called semantic trees.

Semantic Tree Representation

Example continued

$\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(a))$, $C_2 = \neg P(u, v) \vee Q(f(v))$, and $C_3 = \neg Q(z)$ is unsatisfiable. Even $\mathcal{C}' = \{C_1', C_2', C_3'\}$ with $C_1' = P(a, f(a))$, $C_2' = \neg P(a, f(a)) \vee Q(f(a))$, and $C_3' = \neg Q(f(a))$ is unsatisfiable.

The following semantic tree represents all possible truth values of all possible (Herbrand) interpretations of $\mathcal{C}$ restricted to the atoms $P(a, f(a))$ and $Q(f(a))$. In all cases, at least one clause is falsified (since the labels along each branch contain the dual literals of some ground instance of some clause in $\mathcal{C}$).

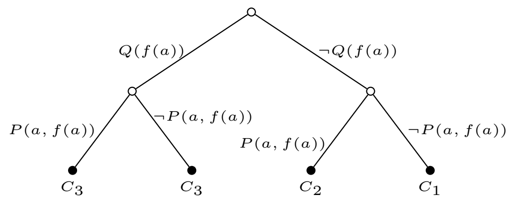


Semantic Tree Representation

Example continued

$\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(a))$, $C_2 = \neg P(u, v) \vee Q(f(v))$, and $C_3 = \neg Q(z)$ is unsatisfiable. Even $\mathcal{C}' = \{C_1', C_2', C_3'\}$ with $C_1' = P(a, f(a))$, $C_2' = \neg P(a, f(a)) \vee Q(f(a))$, and $C_3' = \neg Q(f(a))$ is unsatisfiable.

The following semantic tree represents all possible truth values of all possible (Herbrand) interpretations of $\mathcal{C}$ restricted to the atoms $P(a, f(a))$ and $Q(f(a))$. In all cases, at least one clause is falsified (since the labels along each branch contain the dual literals of some ground instance of some clause in $\mathcal{C}$).



Semantic Trees

Definition (Semantic Tree)

A semantic tree for a set of clauses $\mathcal{C}$ over signature $\mathcal{L}$ is an edge-labelled tree $\mathcal{T} = \langle NOD, E, \xi \rangle$ with nodes NOD, edges E and labelling function ξ, s.t. the following conditions are fulfilled:

1. $\xi\colon E \to HB_{\mathcal{L}} \cup \{\neg A \mid A \in HB_{\mathcal{L}}\}$.
2. $\mathcal{T}$ is a proper binary tree (i.e., every non-leaf node has exactly 2 children).
3. If e_1 and e_2 are edges starting from a common node then $\xi(e_1)$ and $\xi(e_2)$ are dual literals (i.e., A and $\neg A$ for some $A \in HB_{\mathcal{L}}$).
4. Let N be a node in $\mathcal{T}$ and π the (unique) path connecting N with the root and let $\gamma_N = \{L \mid \exists e \in E$, s.t. e is an edge on π and $\xi(e) = L\}$. Then γ_N does not contain complementary literals (i.e., γ_N is satisfiable).

Intended meaning

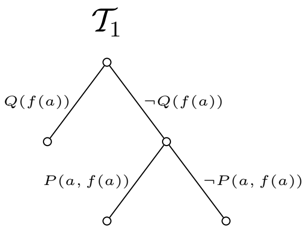
A node in a semantic tree represents a partial truth assignment for $HB_{\mathcal{L}}$, which can be extended to those truth assignments where all literals in γ_N are true.

Example

Consider a clause set $\mathcal{C}$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, Q. Then $\mathcal{T}_1$ is a semantic tree while $\mathcal{T}_2$ and $\mathcal{T}_3$ are not.

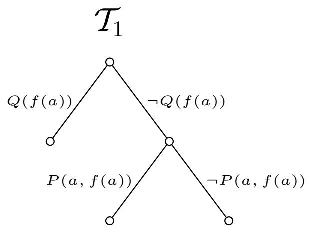
Example

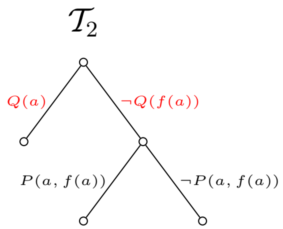
Consider a clause set $\mathcal{C}$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, Q. Then $\mathcal{T}_1$ is a semantic tree while $\mathcal{T}_2$ and $\mathcal{T}_3$ are not.



Example

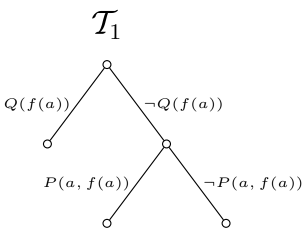
Consider a clause set $\mathcal{C}$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, Q. Then $\mathcal{T}_1$ is a semantic tree while $\mathcal{T}_2$ and $\mathcal{T}_3$ are not.

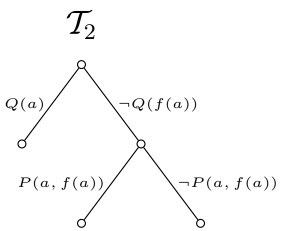


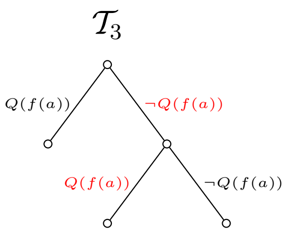


Example

Consider a clause set $\mathcal{C}$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, Q. Then $\mathcal{T}_1$ is a semantic tree while $\mathcal{T}_2$ and $\mathcal{T}_3$ are not.







Complete Semantic Trees

Definition (Branch)

Let $\mathcal{T}$ be a semantic tree. A path π of $\mathcal{T}$ is called a branch if the following properties are fulfilled:

1. π starts at the root of $\mathcal{T}$.
2. π is either infinite or it goes from the root to some leaf.

We define the (partial) interpretation $\gamma(\pi)$ as follows:

1. If π is finite, then we set $\gamma(\pi) = \gamma_N$, where N is the leaf node on π.
2. If π is infinite with nodes $(N_i)_{i \in \mathbb{N}}$, then we set $\gamma(\pi) = \bigcup_{i \in \mathbb{N}} \gamma_{N_i}$.

Definition (Complete semantic tree)

Let $\mathcal{T}$ be a semantic tree for a clause set $\mathcal{C}$ over the signature $\mathcal{L}$. Then $\mathcal{T}$ is called complete if for every branch π in $\mathcal{T}$ and every $A \in HB_{\mathcal{L}}$, either $A \in \gamma(\pi)$ or $\neg A \in \gamma(\pi)$ (i.e., every branch represents a full Herbrand interpretation).

Construction of a Complete Semantic Tree

Construction

Let $\psi\colon \mathbb{N} \to HB_{\mathcal{L}}$ be an enumeration of $HB_{\mathcal{L}}$

Set $\mathcal{T}_0 = \langle NOD_0, E_0, \xi_0\rangle$ with $NOD_0 = \{root\}$, $E_0 = \emptyset$, and $\xi_0 = \emptyset$

Let $n < |HB_{\mathcal{L}}|$, let FIN_n denote the set of all leaf nodes in NOD_n. Moreover, for every node $N \in FIN_n$, let $\alpha_1(N)$ and $\alpha_2(N)$ denote two new nodes.

Then we set $\mathcal{T}_{n+1} = \langle NOD_{n+1}, E_{n+1}, \xi_{n+1}\rangle$ with

$$\begin{aligned} NOD_{n+1} &= NOD_n \cup \bigcup_{N \in FIN_n} \{\alpha_1(N), \alpha_2(N)\}, \\ E_{n+1} &= E_n \cup \bigcup_{N \in FIN_n} \{(N, \alpha_1(N)), (N, \alpha_2(N))\}, \\ \xi_{n+1} &= \xi_n \cup \bigcup_{N \in FIN_n} \{((N, \alpha_1(N)), \psi(n)), ((N, \alpha_2(N)), \neg\psi(n))\}, \end{aligned}$$

Limit Tree

Let $\alpha = |HB_{\mathcal{L}}|$. Then we define the *limit tree* $\hat{\mathcal{T}} = \langle N\hat{O}D, \hat{E}, \hat{\xi}\rangle$ with

$$N\hat{O}D = \bigcup_{i=0}^{\alpha} NOD_i \qquad \hat{E} = \bigcup_{i=0}^{\alpha} E_i \qquad \hat{\xi} = \bigcup_{i=0}^{\alpha} \xi_i$$

Motivation

- Clearly, a clause set $\mathcal{C}$ is unsatisfiable iff in a complete semantic tree every branch falsifies (at least one ground instance of some clause in) $\mathcal{C}$.
- If $HB_{\mathcal{L}}$ is infinite, we cannot construct the entire semantic tree $\hat{\mathcal{T}}$.
- However, we may stop expanding a node that falsifies some clause in $C \in \mathcal{C}$, since all branches resulting from this expansion are guaranteed to falsify this clause as well.
- This idea will be formalized in the notion of failure nodes and will be crucial for the proof of Herbrand's Theorem.

Failure Nodes and Complete Semantic Trees

Definition (Failure node)

Let $\mathcal{T} = \langle NOD, E, \xi\rangle$ be a semantic tree for clause set $\mathcal{C}$ over signature $\mathcal{L}$.

Let $N \in NOD$ and $C \in \mathcal{C}$. We say that N falsifies C if there exsits a ground instance C' of C s.t. for all literals L in C', the dual of L is contained in γ_N.

Let $N \in NOD$. We call N a failure node if N falsifies some clause $C \in \mathcal{C}$ but no ancestor node of N falsifies any clause in $\mathcal{C}$.

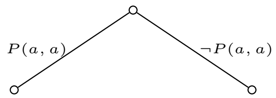
Definition (Closed Semantic Tree)

A semantic tree $\mathcal{T}$ is called closed if on every branch of $\mathcal{T}$ there is a failure node.

Example

Consider the clause set $\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(x))$, $C_2 = \neg P(a, f(y)) \vee R(y)$ and $C_3 = \neg R(z)$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, R.

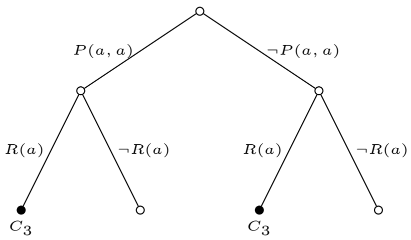
We construct the limit tree $\hat{\mathcal{T}}$ for $\mathcal{C}$ via the following enumeration ψ of $HB_{\mathcal{L}}$: $\psi(0) = P(a, a)$, $\psi(1) = R(a)$, $\psi(2) = P(a, f(a))$, etc. If we do not further expand failure nodes, then we get the sequence $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ of semantic trees:



Example

Consider the clause set $\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(x))$, $C_2 = \neg P(a, f(y)) \vee R(y)$ and $C_3 = \neg R(z)$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, R.

We construct the limit tree $\hat{\mathcal{T}}$ for $\mathcal{C}$ via the following enumeration ψ of $HB_{\mathcal{L}}$: $\psi(0) = P(a,a)$, $\psi(1) = R(a)$, $\psi(2) = P(a, f(a))$, etc. If we do not further expand failure nodes, then we get the sequence $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ of semantic trees:

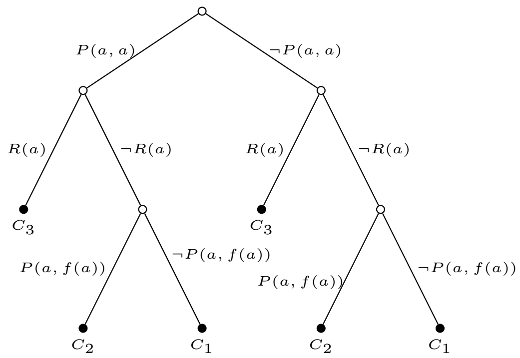


Foundations of DKS | 3. Foundations of ATP | 3.4 Semantic Trees

Example

Consider the clause set $\mathcal{C} = \{C_1, C_2, C_3\}$ with $C_1 = P(x, f(x))$, $C_2 = \neg P(a, f(y)) \vee R(y)$ and $C_3 = \neg R(z)$ over the signature $\mathcal{L}$ with function symbols a, f and predicate symbols P, R.

We construct the limit tree $\hat{\mathcal{T}}$ for $\mathcal{C}$ via the following enumeration ψ of $HB_{\mathcal{L}}$: $\psi(0) = P(a,a)$, $\psi(1) = R(a)$, $\psi(2) = P(a, f(a))$, etc. If we do not further expand failure nodes, then we get the sequence $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ of semantic trees:



Detecting Unsatisfiability via Semantic Trees

Theorem

Let $\hat{\mathcal{T}}$ be the limit tree for some enumeration of $HB_{\mathcal{L}}$ and let $\mathcal{C}$ be a clause set over signature $\mathcal{L}$.

The clause set $\mathcal{C}$ is unsatisfiable iff $\hat{\mathcal{T}}$ is closed.

Proof idea

"⇒" Suppose that $\mathcal{C}$ is unsatisfiable. Let π be a branch in $\hat{\mathcal{T}}$. Then $\mathcal{C}$ evaluates to false in the interpretation corresponding to $\gamma(\pi)$. Hence, there exists a ground instance C' of some clause $C \in \mathcal{C}$, s.t. C' is false in $\gamma(\pi)$. But then there exists a node $N \in NOD$, s.t. the dual of all literals in C' occurs in γ_N. Thus N falsifies C and therefore N or some ancestor of N is a failure node.

"⇐" Suppose that $\hat{\mathcal{T}}$ is closed. Let $\mathcal{I}$ be an arbitrary Herbrand model. Then $\mathcal{I}$ is the extension of the partial model represented by some branch π of $\hat{\mathcal{T}}$ (this holds for any semantic tree, not only closed ones). Since π ends in a failure node N (i.e., some clause $C \in \mathcal{C}$ is falsified by N), C is also false in $\mathcal{I}$. □

Herbrand's Theorem

Lemma

Let $\mathcal{T} = \langle NOD, E, \xi \rangle$ be a closed semantic tree for clause set $\mathcal{C}$ over signature $\mathcal{L}$. Let $Clos(\mathcal{T})$ denote the tree constructed from $\mathcal{T}$ by omitting all paths starting at failure nodes. Then $Clos(\mathcal{T})$ is finite.

Proof

$Clos(\mathcal{T})$ is a binary tree. Suppose that $Clos(\mathcal{T})$ is infinite. Then, by König's Lemma, $Clos(\mathcal{T})$ must possess an infinite path (since the degree of all nodes in $Clos(\mathcal{T})$ is ≤ 3) and thus an infinite branch. But an infinite branch has no failure node. Hence, $\mathcal{T}$ is not closed, which is a contradiction. □

Theorem (Herbrand's Theorem)

Let $\mathcal{C}$ be a set of clauses over some signature $\mathcal{L}$ and let $\mathcal{C}_{ground}$ denote the set of all ground instances of $\mathcal{C}$.

$\mathcal{C}$ is unsatisfiable iff there exists a finite unsatisfiable set $\mathcal{C}'$ with $\mathcal{C}' \subseteq \mathcal{C}_{ground}$.

Proof Idea

Only the "⇒"-direction is non-trivial:
By the previous lemma, $Clos(\mathcal{T})$ is finite. Moreover, every failure node falsifies at most finitely many ground clauses. We define $\mathcal{C}'$ as the set of all ground instances of clauses in $\mathcal{C}$ which are falsified by a leaf node in $Clos(\mathcal{T})$.
Clearly, $\mathcal{C}'$ is finite. Moreover, it is easy to show that $\mathcal{C}'$ is unsatisfiable. □

Outline

3. Foundations of Automated Theorem Proving
3.1 Substitutions and Unification
3.2 Transformation into Clause Form
3.3 Herbrand Interpretations
3.4 Semantic Trees and Herbrand's Theorem
3.5 Proof of Several Fundamental Theorems

Proof of Several Fundamental Theorems

Motivation

- In the previous part of the lecture, we have stated without proof several fundamental theorems of First-Order Predicate Logic:
  - Completeness Theorem
  - Compactness Theorem
  - Löwenheim-Skolem Theorem
- These results are easy to prove with Herbrand's Theorem at hand.
- Implicitly, we have just seen a proof of the Completeness Theorem:
  1. Take an arbitrary closed formula F whose unsatisfiability shall be tested.
  2. Transform F into a sat-equivalent clause set $\mathcal{C}$.
  3. Take any enumeration of the Herbrand Base HB and construct the sequence $Clos(\mathcal{T}_1)$, $Clos(\mathcal{T}_2)$, $Clos(\mathcal{T}_3)$, ... of semantic trees.
  4. If $\mathcal{C}$ is unsatisfiable, our algorithm will eventually halt with the finite semantic tree $Clos(\hat{\mathcal{T}})$.

Completeness Theorem

Theorem (Gödel, completeness theorem)

There exist calculi for first-order predicate logic such that $S \vdash \varphi$ iff $S \models \varphi$ for any set S of closed formulas and any closed formula φ.

Proof via Herbrand's Theorem

We know that entailment, validity, and unsatisfiability can be translated into one another. The first automated theorem prover (Gilmore, 1960) tested the unsatisfiability by a direct application of Herbrand's Theorem:

W.l.o.g., we restrict ourselves to clause sets. Let $\mathcal{C}$ be a clause set over signature $\mathcal{L}$. Clearly, the Herbrand base $HB_{\mathcal{L}}$ is computably enumerable as a sequence $(H_n)_{n\in\mathbb{N}}$, s.t. at each step, one ground atom is added.

Let $\mathcal{C}'_n := \{C\sigma \mid C \in \mathcal{C}$, all atoms in $C\sigma$ are contained in $H_n\}$.

Clearly, $\mathcal{C}'_n$ is finite for each n. We test the satisfiability of every set $\mathcal{C}'_n$. Since $\mathcal{C}'_n$ is ground, this comes down to a propositional sat-test. By Herbrand's Theorem, we shall eventually find some unsatisfiable set $\mathcal{C}'_m$ if $\mathcal{C}$ is unsatisfiable. □

Compactness Theorem

Theorem (Gödel-Malcev, finiteness or compactness theorem)

Let S be an infinite set of closed formulas. If every finite subset of S is satisfiable, then S is satisfiable.

Proof

W.l.o.g., we restrict ourselves to clause sets. Let $\mathcal{C}$ be an infinite clause set. Suppose that $\mathcal{C}$ is unsatisfiable. By Herbrand's Theorem, there exists a finite subset $\mathcal{C}' \subseteq \mathcal{C}_{ground}$, s.t. $\mathcal{C}'$ is unsatisfiable.

We construct a finite subset $\hat{\mathcal{C}} \subseteq \mathcal{C}$ as follows: For each $C' \in \mathcal{C}'$, select one $C \in \mathcal{C}$, s.t. C' is a ground instance of C. Let $\hat{\mathcal{C}}$ consist of these selected clauses from $\mathcal{C}$. Clearly, $\hat{\mathcal{C}} \subseteq \mathcal{C}$ is finite and unsatisfiable. □

Löwenheim-Skolem Theorem

Theorem (Löwenheim-Skolem)

Every satisfiable enumerable set of closed formulas has a model with a finite or infinite enumerable domain.

Proof

Every enumerable set S of closed formulas is sat-equivalent to an enumerable set S_c of clauses over some enumerable signature $\mathcal{L}$, s.t. $S_c \models S$.

A set of clauses is satisfiable, iff it has a Herbrand model $\mathcal{I}$. Depending on $\mathcal{L}$, the domain of $\mathcal{I}$ is either finite or infinite enumerable.

In summary, if S is satisfiable, then S_c is also satisfiable and has a model $\mathcal{I}$ whose domain is either finite or infinite enumerable. By $S_c \models S$, we have that $\mathcal{I}$ is also a model of S. □