Computational Logic

Herbrand's Theorem

Damiano Zanardini

UPM European Master in Computational Logic (EMCL)
School of Computer Science
Technical University of Madrid
damiano@fi.upm.es

Academic Year 2009/2010

Motivation

The theorem

Herbrand's theorem is the basis for most proof techniques in *automatic theorem proving* (ATP)

How is it useful?

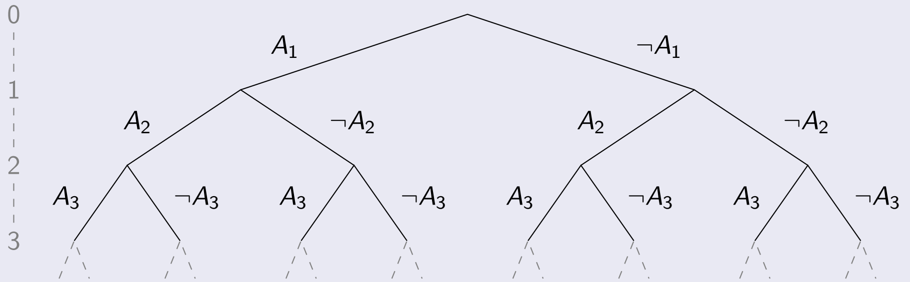
- in order to decide the (un)satisfiability of a formula F, it is enough to study its Herbrand interpretations
- it is necessary to have an *ordered* and *exhaustive* way to *produce* the Herbrand interpretations
- this can be done by means of *semantic trees*

Semantic trees (Robinson '68, Kowalski-Hayes '69)

Definition

Let $HB(F) = \{A_1, A_2, A_3, ..\}$ be the Herbrand base of a formula F in clause form: a *semantic tree* for F is a binary tree where

- every level of the tree corresponds to a ground atom of $HB(F)$
- the two links from a node at level $i-1$ to nodes at level i are labeled, resp., with A_i and $\neg A_i$



Semantic trees (Robinson '68, Kowalski-Hayes '69)

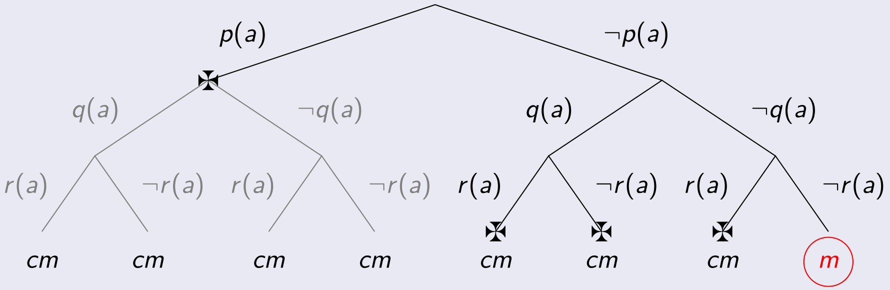
Completeness, failure nodes and closed trees

- a semantic tree is *complete* if every path from the root to a leaf contains A_i or $\neg A_i$ for all $A_i \in HB(F)$
  - a complete tree for F contains all Herbrand interpretations of F
- given a node N, $I(N)$ is the set of all literals which label the path from the root to N
  - ☞ $I(N)$ *partially* represents a Herbrand interpretation
- a node N is a *failure node* (denoted by ✠) if $I(N)$ makes some ground instance of some clause false, and $I(N')$ for any predecessor N' of N does not
  - ☞ that is, $I(N')$ does *not* falsify *any* ground instance of *any* clause
- a tree is *closed* iff all paths from the root to a leaf contain a failure node
  - a closed tree has level n if n is the maximum length of paths from the root to a failure node

Semantic trees (Robinson '68, Kowalski-Hayes '69)

Example: $F = \{\neg q(x) \vee r(x),\ p(x) \vee \neg r(x),\ \neg p(x)\}$

- $H(F) = \{a\}$ $\qquad HB(F) = \{p(a), q(a), r(a)\}$

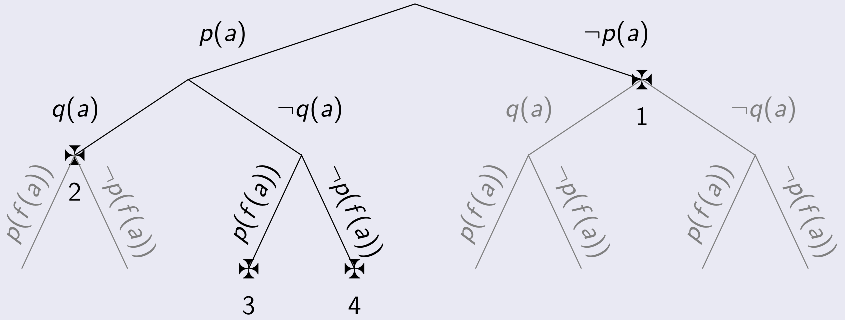


- ✠ = failure node, m = model, cm = countermodel

Semantic trees (Robinson '68, Kowalski-Hayes '69)

Example: $F = \{p(y),\ q(a) \vee \neg p(f(x)),\ \neg q(x)\}$

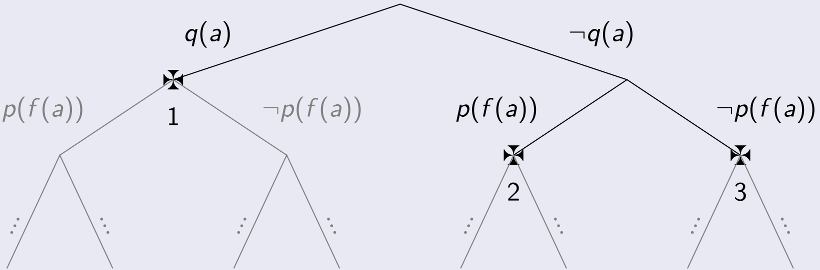
- $H(F) = \{f^n(a) \mid n \geq 0\}$
  $HB(F) = \{p(t) \mid t \in H(F)\} \cup \{q(t) \mid t \in H(F)\}$
- every Herbrand interpretation falsifies some instance of some clause, so that F is unsatisfiable



Semantic trees (Robinson '68, Kowalski-Hayes '69)

Example: $F = \{p(y),\ q(a) \vee \neg p(f(x)),\ \neg q(x)\}$

- $H(F) = \{f^n(a) \mid n \geq 0\}$
  $HB(F) = \{p(t) \mid t \in H(F)\} \cup \{q(t) \mid t \in H(F)\}$
- every Herbrand interpretation falsifies some instance of some clause, so that F is unsatisfiable



Semantic trees (Robinson '68, Kowalski-Hayes '69)

Note on cardinalities

We want to use semantic trees in order to enumerate Herbrand interpretations

- yet, how many interpretations can we have?
- how it is possible to *enumerate* them?

Herbrand's theorem

Lemma (König's Lemma)

In an infinite tree with finite branching (i.e., such that every node has a finite number of children), there must exist an infinite path from the root

Proof.

(typical result in tree theory)

Herbrand's theorem

Theorem

$\mathcal{C}$ is unsatisfiable iff its complete semantic tree is closed

Proof.

- $\mathcal{C}$ is unsatisfiable

$\leftrightarrow$ all Herbrand interpretations make $\mathcal{C}$ false

$\leftrightarrow$ all paths from the root contain a failure node

$\leftrightarrow$ the tree is closed

Herbrand's theorem

Lemma

A complete semantic tree is closed iff a finite tree is obtained by pruning all successors of failure nodes

Proof ($\longrightarrow$).

❶ the complete semantic tree is closed

❷ suppose the pruned tree were not finite

❸ then, by König's lemma, there exists an infinite path

❹ such infinite path would not have any failure nodes

❺ the tree would not be closed: contradiction between ❶ and ❷

❻ the pruned tree is finite

Proof ($\longleftarrow$).

(easy)

Herbrand's theorem

Theorem (Herbrand's theorem (Ph.D. Thesis, 1929))

A set of clauses $\mathcal{C}$ is unsatisfiable iff there exists a finite set of ground instances of $\mathcal{C}$ clauses which is unsatisfiable

Proof ($\longrightarrow$).

1. $\mathcal{C}$ is unsatisfiable
2. there exists a finite semantic tree for $\mathcal{C}$ whose every leaf is a failure node (by ❶ and the above results)
3. every path falsifies at least one ground instance (by ❷)
4. since the tree is finite, collecting one (falsified) instance for every failure node gives a finite set S
5. all Herbrand interpretations falsify some instances in S
6. such finite set S of instances is unsatisfiable (by ❺)

(why Herbrand interpretations of $\mathcal{C}$ are enough to prove $UNSAT(S)$?)

Herbrand's theorem

Theorem (Herbrand's theorem (Ph.D. Thesis, 1929))

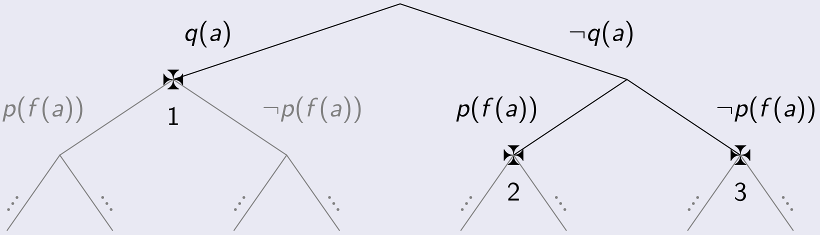
A set of clauses $\mathcal{C}$ is unsatisfiable iff there exists a finite set of ground instances of $\mathcal{C}$ clauses which is unsatisfiable

Proof ($\longleftarrow$).

1. there exists an unsatisfiable finite set S of ground instances of $\mathcal{C}$ clauses
2. suppose $\mathcal{C}$ be satisfiable: then, some Herbrand interpretation would verify every instance of every clause
3. in particular, such interpretation would verify all instances in S
4. S would be satisfiable (by ❸): contradiction between ❶ and ❷
5. $\mathcal{C}$ is unsatisfiable (by ❹)

Herbrand's theorem

Example: $\mathcal{C} = \{p(y),\ q(a) \vee \neg p(f(x)),\ \neg q(x)\}$



- in 1, the instance $\neg q(a)$ of $\neg q(x)$ is falsified
- in 2, the instance $q(a) \vee \neg p(f(a))$ of $q(a) \vee \neg p(f(x))$ is falsified
- in 3, the instance $p(f(a))$ of $p(y)$ is falsified

$\longrightarrow$ this set of ground instances is unsatisfiable

$\longrightarrow$ Herbrand's theorem guarantees that $\mathcal{C}$ is unsatisfiable

Herbrand's theorem

The theorem suggests a method

Given a set $\mathcal{C}$ of clauses, generate its ground instances incrementally, and put them in a set until the whole set becomes unsatisfiable:

$$
\begin{array}{l}
B = \emptyset; \\
\text{while } (B \text{ is satisfiable}) \\
\quad b = \text{new-instance}(\mathcal{C}); \\
\quad B = B \cup \{b\};
\end{array}
$$

Implementations of Herbrand's theorem

It is necessary to choose a *strategy* for generating instances

- method of Gilmore (1960)
- method of Davis-Putnam (1960)
- resolution method by Robinson (1965)