Rewrite Techniques in Theorem Proving
Part I: Orderings

Leo Bachmair

Harald Ganzinger

CADE'94

Overview

We will show how (well-founded) orderings can be used to

- refine inference rules
- construct distinguished models for certain clause sets
- prove the refutational completeness of inference systems

Clauses

A clause is a finite multiset of literals, written either as a disjunction

$$\neg A_1 \vee \cdots \vee \neg A_m \vee B_1 \vee \cdots \vee B_n$$

or as a sequent

$$A_1, \ldots, A_m \rightarrow B_1, \ldots, B_n$$

Herbrand interpretations

An (Herbrand) interpretation is a set I of ground (variable-free) atomic formulas.

> If $A \in I$, then A is true (and $\neg A$ false) in I;
>
> if $A \notin I$, then A is false (and $\neg A$ true) in I.

A ground clause is true if at least one of its literals is true. A non-ground clause is true if all its ground instances are true.

Refutational theorem proving

A set of clauses is satisfiable if there exists an interpretation in which all its clauses are true; and unsatisfiable otherwise.

Refutational theorem provers attempt to determine whether a formula

F logically follows from a set N

by checking whether the set of clauses corresponding to

$N \cup \{\neg F\}$ is unsatisfiable.

Well-founded orderings

An ordering $\succ$ is said to be wellfounded if there is no infinite sequence

$$x_1 \succ x_2 \succ x_3 \succ \cdots$$

An ordering $\succ$ can be extended to multisets as follows:

$$M \succ N \text{ if and only if}$$

$M \neq N$ and, whenever $M(x) < N(x)$, then $N(y) < M(y)$, for some y with $y \succ x$.

EXAMPLE.

$$\{5\} \succ \{1, 1, 2, 4, 4, 4\}$$

Example

Consider the following clauses (listed in increasing order by size):

$$P(0)$$

$$P(0) \vee P(s0)$$

$$P(s0) \vee P(ss0)$$

$$\ldots$$

$$P(s^{2n-1}0) \vee P(s^{2n}0)$$

$$P(s^{2n}0) \vee P(s^{2n+1}0)$$

$$\ldots$$

Example (cont.)

We use the wellfounded ordering to construct a model.

Clause set	Interpretation
$\boxed{P(0)}$	$P(0)$
$P(0) \vee P(s0)$	
$P(s0) \vee \boxed{P(ss0)}$	$P(ss0)$
$\ldots$	$\ldots$
$P(s^{2n-1}0) \vee \boxed{P(s^{2n}0)}$	$P(s^{2n}0)$
$P(s^{2n}0) \vee P(s^{2n+1}0)$	
$\ldots$	$\ldots$

Model construction

Given a set of ground clauses and a total wellfounded ordering $\succ$, define for each clause C a Herbrand interpretation I_C and an auxiliary set E_C.

More specifically,

$$I_C = \bigcup_{C \succ D} E_D$$

and

$$E_C = \{A\} \quad \text{if} \begin{cases} \text{(i) } C \text{ is false in } I_C \\ \text{(ii) } C = \Gamma \rightarrow \Delta, A \\ \text{(iii) } A \succ \Gamma, \Delta \end{cases}$$

$$E_C = \emptyset \qquad \text{otherwise.}$$

If $E_C \neq \emptyset$ the clause C is called productive. Finally,

$$I = \bigcup_C E_C$$

Model construction (cont.)

Productive clauses are evidently true in I, but non-productive clauses may be false.

EXAMPLE. Suppose $\neg B \succ B \succ A$ and consider the clauses

$$\begin{aligned} A &\rightarrow \\ &\rightarrow A, B \\ B &\rightarrow \end{aligned}$$

The second clause is the only productive clause; it produces B. Hence

$$I = \{B\}$$

and the third clause is false.

Minimal counter-examples

A minimal counter-example is a smallest clause false in I (e.g. the third clause in the example above).

NOTE.

> A minimal counter-example is non-productive; hence must violate one of the conditions imposed on productive clauses.
>
> We shall distinguish two cases, depending on whether the second or the third condition is violated.

Ground resolution

Taking the above example, we see that

$$B \rightarrow$$

is the minimal counter-example and

$$\rightarrow A, B$$

produces B.

Look at the resolution inference

$$\frac{\rightarrow A, B \qquad B \rightarrow}{\rightarrow A}$$

The conclusion is a smaller counter-example!

Thus, if a set of clauses is closed under resolution, there are no minimal counter-examples of this kind.

Positive factoring

Take the clauses

$$\begin{aligned} A &\rightarrow \\ &\rightarrow A, B, B \\ B &\rightarrow \end{aligned}$$

There is no productive clause; the second clause is the minimal counter-example.

The factoring inference

$$\frac{\rightarrow A, B, B}{\rightarrow A, B}$$

yields a smaller counter-example.

Thus, if a set of clauses is closed under positive factoring, this kind of counter-example cannot be present.

Selection

Let $\succ$ be a well-founded ordering and S be a (partial) function that selects negative literals in clauses.

Ordered Ground Resolution

$$\frac{\Gamma \rightarrow \Delta, A \qquad A, \Sigma \rightarrow \Pi}{\Gamma, \Sigma \rightarrow \Delta, \Pi}$$

if (i) A is maximal in the first premise and selected or maximal in the second premise *and* (ii) the first premise contains no selected literals.

Positive Ground Factoring

$$\frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}$$

if (i) A is maximal in the premise *and* (ii) the premise contains no selected literals.

Refutational Completeness

THEOREM.

If a set of (ground) clauses is saturated under ordered resolution with selection and under positive factoring, then it is unsatisfiable if and only if it contains the empty clause.

Remarks

1. Productive clauses are conditional rewrite rules with positive and negative conditions. They define a *perfect model* in the logic programming sense.

2. Hyperresolution can be simulated by ordered resolution with selection.

3. Tautologies and subsumed clauses can neither be minimal counter-examples nor productive. They are redundant!

Lock resolution

EXAMPLE.

$$\begin{array}{rcl} A_1 & \vee & B_5 \\ \neg A_2 & \vee & \neg B_6 \\ A_7 & \vee & \neg B_3 \\ \neg A_8 & \vee & B_4 \end{array}$$

Resolve only on literals with minimal index:

$$\begin{array}{rcl} B_5 & \vee & \neg B_6 \\ A_7 & \vee & \neg A_8 \end{array}$$

If we delete tautologies, no contradiction can be deduced!

Lock resolution as ordered resolution

First introduce new literals:

$$\begin{array}{ll} L_{1,1} \supset A & L_{1,2} \supset B \\ L_{2,1} \supset \neg A & L_{2,2} \supset \neg B \\ L_{3,1} \supset A & L_{3,2} \supset \neg B \\ L_{4,1} \supset \neg A & L_{4,2} \supset B \end{array}$$

Then saturate under ordered resolution with $A, B \succ L_{i,j}$, e.g.

$$\frac{\neg L_{1,1} \vee A \quad L_{2,1} \vee \neg A}{\neg L_{1,1} \vee \neg L_{2,1}}$$

The result is a set of connections:

$$\begin{array}{c} \neg L_{1,1} \vee \neg L_{2,1} \\ \neg L_{1,1} \vee \neg L_{4,1} \\ \dots \\ \neg L_{3,2} \vee \neg L_{4,2} \end{array}$$

Lock resolution as ordered resolution (cont.)

Then encode original clauses:

$$L_{1,1} \vee L_{1,2}$$
$$L_{2,1} \vee L_{2,2}$$
$$L_{3,1} \vee L_{3,2}$$
$$L_{4,1} \vee L_{4,2}$$

Finally apply (negative) ordered hyperresolution with an ordering that reflects the indexing in the original clauses, e.g.

$$\frac{L_{1,1} \vee L_{1,2} \quad L_{2,1} \vee L_{2,2} \quad \neg L_{1,1} \vee \neg L_{2,1}}{L_{1,2} \vee L_{2,2}}$$

NOTE.

> The inverse method can be encoded in a similar fashion.

Literature

Further details can be found in:

L. Bachmair and H. Ganzinger. Non-clausal resolution and superposition with selection and redundancy criteria. *Proc. LPAR 92, LNAI 624*. Springer-Verlag, 1992, pp. 273–284.

L. Bachmair and H. Ganzinger. Perfect model semantics for logic programs with equality, *Proc. ICLP 91*. MIT Press, 1991, pp. 645–659.