3.13 A Resolution Prover

So far: static view on completeness of resolution:

Saturated sets are inconsistent if and only if they contain $\bot$.

We will now consider a dynamic view:

How can we get saturated sets in practice?

The theorems 3.41 and 3.42 are the basis for the completeness proof of our prover RP.

Rules for Simplifications and Deletion

We want to employ the following rules for simplification of prover states N:

- *Deletion of tautologies*

$$N \cup \{C \vee A \vee \neg A\} \;\Rightarrow\; N$$

- *Deletion of subsumed clauses*

$$N \cup \{C, D\} \;\Rightarrow\; N \cup \{C\}$$

  if $C\sigma \subseteq D$ (C *subsumes* D).

- *Reduction* (also called *subsumption resolution*)

$$N \cup \{C \vee L,\;\; D \vee C\sigma \vee \overline{L}\sigma\} \;\Rightarrow\; N \cup \{C \vee L, D \vee C\sigma\}$$

Resolution Prover RP

3 clause sets:

N(ew) containing new resolvents
P(rocessed) containing simplified resolvents
clauses get into *O(ld)* once their inferences have been computed

Strategy:

Inferences will only be computed when there are no possibilities for simplification

Transition Rules for RP (I)

Tautology elimination

$N \cup \{C\} \mid P \mid O \quad \Rightarrow_{RP} \quad N \mid P \mid O$

if C is a tautology

Forward subsumption

$N \cup \{C\} \mid P \mid O \quad \Rightarrow_{RP} \quad N \mid P \mid O$

if some $D \in P \cup O$ subsumes C

Backward subsumption

$N \cup \{C\} \mid P \cup \{D\} \mid O \quad \Rightarrow_{RP} \quad N \cup \{C\} \mid P \mid O$

$N \cup \{C\} \mid P \mid O \cup \{D\} \quad \Rightarrow_{RP} \quad N \cup \{C\} \mid P \mid O$

if C strictly subsumes D

Transition Rules for RP (II)

Forward reduction

$N \cup \{C \vee L\} \mid P \mid O \quad \Rightarrow_{RP} \quad N \cup \{C\} \mid P \mid O$

if there exists $D \vee L' \in P \cup O$
such that $\overline{L} = L'\sigma$ and $D\sigma \subseteq C$

Backward reduction

$N \mid P \cup \{C \vee L\} \mid O \quad \Rightarrow_{RP} \quad N \mid P \cup \{C\} \mid O$

$N \mid P \mid O \cup \{C \vee L\} \quad \Rightarrow_{RP} \quad N \mid P \cup \{C\} \mid O$

if there exists $D \vee L' \in N$
such that $\overline{L} = L'\sigma$ and $D\sigma \subseteq C$

Transition Rules for RP (III)

Clause processing

$N \cup \{C\} \mid P \mid O \quad \Rightarrow_{RP} \quad N \mid P \cup \{C\} \mid O$

Inference computation

$\emptyset \mid P \cup \{C\} \mid O \quad \Rightarrow_{RP} \quad N \mid P \mid O \cup \{C\}$,

where N is the set of conclusions
of $Res_S^{\succ}$-inferences from clauses
in $O \cup \{C\}$ where C is one of the
premises

Soundness and Completeness

Theorem 3.43

$$N \models \bot \quad \Leftrightarrow \quad N \mid \emptyset \mid \emptyset \quad \overset{*}{\Rightarrow}_{RP} \quad N' \cup \{\bot\} \mid \_ \mid \_$$

Proof in L. Bachmair, H. Ganzinger: Resolution Theorem Proving appeared in the Handbook of Automated Reasoning, 2001

Fairness

Problem:

If N is inconsistent, then $N \mid \emptyset \mid \emptyset \overset{*}{\Rightarrow}_{RP} N' \cup \{\bot\} \mid \_ \mid \_$.

Does this imply that *every* derivation starting from an inconsistent set N eventually produces $\bot$?

No: a clause could be kept in P without ever being used for an inference.

We need in addition a *fairness condition:*

If an inference is possible forever (that is, none of its premises is ever deleted), then it must be computed eventually.

One possible way to guarantee fairness: Implement P as a queue (there are other techniques to guarantee fairness).

With this additional requirement, we get a stronger result: If N is inconsistent, then every *fair* derivation will eventually produce $\bot$.

Hyperresolution

There are *many* variants of resolution. (We refer to [Bachmair, Ganzinger: Resolution Theorem Proving] for further reading.)

One well-known example is hyperresolution (Robinson 1965):

Assume that several negative literals are selected in a clause C. If we perform an inference with C, then one of the selected literals is eliminated.

Suppose that the remaining selected literals of C are again selected in the conclusion. Then we must eliminate the remaining selected literals one by one by further resolution steps.

Hyperresolution replaces these successive steps by a single inference. As for $Res^{\succ}_{S}$, the calculus is parameterized by an atom ordering $\succ$ and a selection function S.

$$\frac{D_1 \vee B_1 \quad \dots \quad D_n \vee B_n \qquad C \vee \neg A_1 \vee \dots \vee \neg A_n}{(D_1 \vee \dots \vee D_n \vee C)\sigma}$$

with $\sigma = \text{mgu}(A_1 \doteq B_1, \dots, A_n \doteq B_n)$, if

(i) $B_i\sigma$ strictly maximal in $D_i\sigma$, $1 \leq i \leq n$;

(ii) nothing is selected in D_i;

(iii) the indicated occurrences of the $\neg A_i$ are exactly the ones selected by S, or else nothing is selected in the right premise and $n = 1$ and $\neg A_1\sigma$ is maximal in $C\sigma$.

Similarly to resolution, hyperresolution has to be complemented by a factorization inference.

As we have seen, hyperresolution can be simulated by iterated binary resolution.

However this yields intermediate clauses which HR might not derive, and many of them might not be extendable into a full HR inference.

3.14 Summary: Resolution Theorem Proving

- Resolution is a machine calculus.
- Subtle interleaving of enumerating instances and proving inconsistency through the use of unification.
- Parameters: atom ordering $\succ$ and selection function S. On the non-ground level, ordering constraints can (only) be solved approximatively.
- Completeness proof by constructing candidate interpretations from productive clauses $C \vee A$, $A \succ C$; inferences with those reduce counterexamples.
- *Local* restrictions of inferences via $\succ$ and S
  $\Rightarrow$ fewer proof variants.
- *Global* restrictions of the search space via elimination of redundancy
  $\Rightarrow$ computing with "smaller" clause sets;
  $\Rightarrow$ termination on many decidable fragments.
- However: not good enough for dealing with orderings, equality and more specific algebraic theories (lattices, abelian groups, rings, fields)
  $\Rightarrow$ further specialization of inference systems required.

3.15 Semantic Tableaux

Literature:

M. Fitting: First-Order Logic and Automated Theorem Proving, Springer-Verlag, New York, 1996, chapters 3, 6, 7.

R. M. Smullyan: First-Order Logic, Dover Publ., New York, 1968, revised 1995.

Like resolution, semantic tableaux were developed in the sixties, independently by Zbigniew Lis and Raymond Smullyan on the basis of work by Gentzen in the 30s and of Beth in the 50s.

Idea

Idea (for the propositional case):

A set $\{F \wedge G\} \cup N$ of formulas has a model if and only if $\{F \wedge G,\ F,\ G\} \cup N$ has a model.

A set $\{F \vee G\} \cup N$ of formulas has a model if and only if $\{F \vee G,\ F\} \cup N$ or $\{F \vee G,\ G\} \cup N$ has a model.

(and similarly for other connectives).

To avoid duplication, represent sets as paths of a tree.

Continue splitting until two complementary formulas are found $\Rightarrow$ inconsistency detected.

A Tableau for $\{P \wedge \neg(Q \vee \neg R),\ \neg Q \vee \neg R\}$

1. $P \wedge \neg(Q \vee \neg R)$
2. $\neg Q \vee \neg R$

3. $\neg Q$	4. $\neg R$
5. P	10. P
6. $\neg(Q \vee \neg R)$	11. $\neg(Q \vee \neg R)$
7. $\neg Q$	
8. $\neg\neg R$	
9. R	

This tableau is not "maximal", however the first "path" is. This path is not "closed", hence the set $\{1, 2\}$ is satisfiable. (These notions will all be defined below.)

Properties

Properties of tableau calculi:

analytic: inferences according to the logical content of the symbols.

goal oriented: inferences operate directly on the goal to be proved (unlike, e. g., ordered resolution).

global: some inferences affect the entire proof state (set of formulas), as we will see later.

Propositional Expansion Rules

Expansion rules are applied to the formulas in a tableau and expand the tableau at a leaf. We append the conclusions of a rule (horizontally or vertically) at a *leaf*, whenever the premise of the expansion rule matches a formula appearing *anywhere* on the path from the root to that leaf.

Negation Elimination

$$\frac{\neg\neg F}{F} \qquad \frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top}$$

α-Expansion

(for formulas that are essentially conjunctions: append subformulas α_1 and α_2 one on top of the other)

$$\frac{\alpha}{\begin{array}{l}\alpha_1\\ \alpha_2\end{array}}$$

β-Expansion

(for formulas that are essentially disjunctions:
append β_1 and β_2 horizontally, i. e., branch into β_1 and β_2)

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Classification of Formulas

conjunctive			disjunctive		
α	α_1	α_2	β	β_1	β_2
$X \wedge Y$	X	Y	$\neg(X \wedge Y)$	$\neg X$	$\neg Y$
$\neg(X \vee Y)$	$\neg X$	$\neg Y$	$X \vee Y$	X	Y
$\neg(X \rightarrow Y)$	X	$\neg Y$	$X \rightarrow Y$	$\neg X$	Y

We assume that the binary connective $\leftrightarrow$ has been eliminated in advance.

Tableaux: Notions

A *semantic tableau* is a marked (by formulas), finite, unordered tree and inductively defined as follows: Let $\{F_1, \ldots, F_n\}$ be a set of formulas.

(i) The tree consisting of a single path

$$\begin{array}{c} F_1 \\ \vdots \\ F_n \end{array}$$

is a tableau for $\{F_1, \ldots, F_n\}$. (We do not draw edges if nodes have only one successor.)

(ii) If T is a tableau for $\{F_1, \ldots, F_n\}$ and if T' results from T by applying an expansion rule then T' is also a tableau for $\{F_1, \ldots, F_n\}$.

A *path* (from the root to a leaf) in a tableau is called *closed*, if it either contains $\bot$, or else it contains both some formula F and its negation $\neg F$. Otherwise the path is called *open*.

A tableau is called *closed*, if all paths are closed.

A *tableau proof* for F is a closed tableau for $\{\neg F\}$.

A path P in a tableau is called *maximal*, if for each non-atomic formula F on P there exists a node in P at which the expansion rule for F has been applied.

In that case, if F is a formula on P, P also contains:

(i) F_1 and F_2, if F is a α-formula,

(ii) F_1 or F_2, if F is a β-formula, and

(iii) F', if F is a negation formula, and F' the conclusion of the corresponding elimination rule.

A tableau is called *maximal*, if each path is closed or maximal.

A tableau is called *strict*, if for each formula the corresponding expansion rule has been applied at most once on each path containing that formula.

A tableau is called *clausal*, if each of its formulas is a clause.

A Sample Proof

One starts out from the negation of the formula to be proved.

$$
\begin{array}{lcr}
1. & \neg[(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \vee S) \rightarrow ((Q \rightarrow R) \vee S))] & \\
2. & (P \rightarrow (Q \rightarrow R)) & [1_1] \\
3. & \neg((P \vee S) \rightarrow ((Q \rightarrow R) \vee S)) & [1_2] \\
4. & P \vee S & [3_1] \\
5. & \neg((Q \rightarrow R) \vee S)) & [3_2] \\
6. & \neg(Q \rightarrow R) & [5_1] \\
7. & \neg S & [5_2]
\end{array}
$$

8. $\neg P$ $[2_1]$ 9. $Q \rightarrow R$ $[2_2]$

10. P $[4_1]$ 11. S $[4_2]$

There are three paths, each of them closed.

Properties of Propositional Tableaux

We assume that T is a tableau for $\{F_1, \dots, F_n\}$.

Theorem 3.44 *$\{F_1, \dots, F_n\}$ satisfiable $\Leftrightarrow$ some path (i. e., the set of its formulas) in T is satisfiable.*

Proof. By induction over the structure of T. ☐

Corollary 3.45 *T closed $\Rightarrow$ $\{F_1, \dots, F_n\}$ unsatisfiable*

Theorem 3.46 *Let T be a strict propositional tableau. Then T is finite.*

Proof. New formulas resulting from expansion are either $\bot$, $\top$ or subformulas of the expanded formula. By strictness, on each path a formula can be expanded at most once. Therefore, each path is finite, and a finitely branching tree with finite paths is finite by Lemma 1.8. ☐

Conclusion: Strict and maximal tableaux can be effectively constructed.

Refutational Completeness

Theorem 3.47 *Let P be a maximal, open path in a tableau. Then set of formulas on P is satisfiable.*

Proof. (The theorem holds for arbitrary tableaux, but in this proof we consider only the case of a clausal tableau. The full proof can be found, e.g., in Fitting 1996.)

Let N be the set of formulas on P. As P is open, $\perp$ is not in N. Let $C \vee A$ and $D \vee \neg A$ be two resolvable clauses in N. One of the two subclauses C or D, C say, is not empty, as otherwise P would be closed. Since P is maximal, in P the β-rule was applied on $C \vee A$. Therefore, P (and N) contains a proper subclause of $C \vee A$, and hence $C \vee A$ is redundant w. r. t. N. By the same reasoning, if N contains a clause that can be factored, that clause must be redundant w. r. t. N. In other words, N is saturated up to redundancy w. r. t. Res(olution). Now apply Theorem 3.19 to prove satisfiability of N. □

Theorem 3.48 $\{F_1, \ldots, F_n\}$ *satisfiable* $\Leftrightarrow$ *there exists no closed strict tableau for* $\{F_1, \ldots, F_n\}$.

Proof. One direction is clear by Theorem 3.44. For the reverse direction, let T be a strict, maximal tableau for $\{F_1, \ldots, F_n\}$ and let P be an open path in T. By the previous theorem, the set of formulas on P, and hence by Theorem 3.44 the set $\{F_1, \ldots, F_n\}$, is satisfiable. □

Consequences

The validity of a propositional formula F can be established by constructing a strict, maximal tableau for $\{\neg F\}$:

- T closed $\Leftrightarrow$ F valid.
- It suffices to test complementarity of paths w. r. t. atomic formulas (cf. reasoning in the proof of Theorem 3.47).
- Which of the potentially many strict, maximal tableaux one computes does not matter. In other words, tableau expansion rules can be applied don't-care non-deterministically ("*proof confluence*").
- The expansion strategy, however, can have a dramatic impact on tableau size.
- Since it is sufficient to saturate paths w. r. t. ordered resolution (up to redundancy), tableau expansion rules can be even more restricted, in particular by certain ordering constraints.

Semantic Tableaux for First-Order Logic

Additional classification of quantified formulas:

universal		existential	
γ	$\gamma(t)$	δ	$\delta(t)$
$\forall xF$	$F[t/x]$	$\exists xF$	$F[t/x]$
$\neg\exists xF$	$\neg F[t/x]$	$\neg\forall xF$	$\neg F[t/x]$

Moreover we assume that the set of variables X is partitioned into 2 disjoint infinite subsets X_g and X_f, so that bound [free] variables variables can be chosen from X_g $[X_f]$. (This avoids the variable capturing problem.)

Additional Expansion Rules

γ-expansion

$$\frac{\gamma}{\gamma(x)} \quad \text{where } x \text{ is a variable in } X_f$$

δ-expansion

$$\frac{\delta}{\delta(f(x_1,\ldots,x_n))}$$

where f is a *new* Skolem function, and the x_i are the free variables in δ

Skolemization becomes part of the calculus and needs not necessarily be applied in a preprocessing step. Of course, one could do Skolemization beforehand, and then the δ-rule would not be needed.

Note that the rules are parametric, instantiated by the choices for x and f, respectively. Strictness here means that only one instance of the rule is applied on each path to any formula on the path.

In this form the rules go back to Hähnle and Schmitt: The liberalized δ-rule in free variable semantic tableaux, J. Automated Reasoning 13,2, 1994, 211–221.

Free-Variable Tableaux

Let $\{F_1, \ldots, F_n\}$ be a set of *closed formulas*.

(i) The tree consisting of a single path

$$\begin{array}{c} F_1 \\ \vdots \\ F_n \end{array}$$

is a tableau for $\{F_1, \ldots, F_n\}$.

(ii) If T is a tableau for $\{F_1, \ldots, F_n\}$ and if T' results by applying an expansion rule to T, then T' is also a tableau for $\{F_1, \ldots, F_n\}$.

(iii) If T is a tableau for $\{F_1, \ldots, F_n\}$ and if σ is a substitution, then $T\sigma$ is also a tableau for $\{F_1, \ldots, F_n\}$.

The *substitution rule* (iii) may, potentially, modify all the formulas of a tableau. This feature is what is makes the tableau method a *global proof method*. (Resolution, by comparison, is a local method.)

If one took (iii) literally, by repeated application of γ-rule one could enumerate all substitution instances of the universally quantified formulas. That would be a major drawback compared with resolution. Fortunately, we can improve on this.

Example

1.	$\neg[\exists w \forall x\ p(x, w, f(x, w)) \rightarrow \exists w \forall x \exists y\ p(x, w, y)]$	
2.	$\exists w \forall x\ p(x, w, f(x, w))$	1_1 $[\alpha]$
3.	$\neg \exists w \forall x \exists y\ p(x, w, y)$	1_2 $[\alpha]$
4.	$\forall x\ p(x, c, f(x, c))$	$2(c)$ $[\delta]$
5.	$\neg \forall x \exists y\ p(x, v_1, y)$	$3(v_1)$ $[\gamma]$
6.	$\neg \exists y\ p(b(v_1), v_1, y)$	$5(b(v_1))$ $[\delta]$
7.	$p(v_2, c, f(v_2, c))$	$4(v_2)$ $[\gamma]$
8.	$\neg p(b(v_1), v_1, v_3)$	$6(v_3)$ $[\gamma]$

7. and 8. are complementary (modulo unification):

$$v_2 \doteq b(v_1),\ c \doteq v_1,\ f(v_2, c) \doteq v_3$$

is solvable with an mgu $\sigma = [c/v_1, b(c)/v_2, f(b(c), c)/v_3]$, and hence, $T\sigma$ is a closed (linear) tableau for the formula in 1.

AMGU-Tableaux

Idea: Restrict the substitution rule to unifiers of complementary formulas.

We speak of an *AMGU-Tableau*, whenever the substitution rule is only applied for substitutions σ for which there is a path in T containing two *literals* $\neg A$ and B such that $\sigma = \mathrm{mgu}(A, B)$.

Correctness

Given an signature Σ, by Σ^{sko} we denote the result of adding infinitely many new Skolem function symbols which we may use in the δ-rule.

Let $\mathcal{A}$ be a Σ^{sko}-interpretation, T a tableau, and β a variable assignment over $\mathcal{A}$.

T is called $(\mathcal{A}, \beta)$-*valid*, if there is a path P_β in T such that $\mathcal{A}, \beta \models F$, for each formula F on P_β.

T is called *satisfiable* if there exists a structure $\mathcal{A}$ such that for each assignment β the tableau T is $(\mathcal{A}, \beta)$-valid. (This implies that we may choose P_β depending on β.)

Theorem 3.49 *Let T be a tableau for $\{F_1, \ldots, F_n\}$, where the F_i are closed Σ-formulas. Then $\{F_1, \ldots, F_n\}$ is satisfiable $\Leftrightarrow$ T is satisfiable.*

Proof. Proof of "$\Rightarrow$" by induction over the depth of T. For δ one needs to reuse the ideas for proving that Skolemization preserves [un-]satisfiability. $\square$

Incompleteness of Strictness

Strictness for γ is incomplete:

$$
\begin{array}{lcr}
1. & \neg[\forall x\ p(x) \rightarrow (p(c) \wedge p(b))] & \\
2. & \forall x\ p(x) & 1_1 \\
3. & \neg(p(c) \wedge p(b)) & 1_2 \\
4. & p(v_1) & 2(v_1)
\end{array}
$$

$$
5.\ \ \neg p(c)\ \ 3_1 \qquad\qquad 6.\ \ \neg p(b)\ \ 3_2
$$

If we placed a strictness requirement also on applications of γ, the tableau would only be expandable by the substitution rule. However, there is no substitution (for v_1) that can close both paths simultaneously.

Multiple Application of γ Solves the Problem

$$
\begin{array}{lcl}
1. & \neg[\forall x\ p(x) \rightarrow (p(c) \wedge p(b))] & \\
2. & \forall x\ p(x) & 1_1 \\
3. & \neg(p(c) \wedge p(b)) & 1_2 \\
4. & p(v_1) & 2_{v_1}
\end{array}
$$

$$
\begin{array}{lll}
5. & \neg p(c) & 3_1
\end{array}
\qquad\qquad
\begin{array}{lll}
6. & \neg p(b) & 3_2 \\
7. & p(v_2) & 2_{v_2}
\end{array}
$$

The point is that different applications of γ to $\forall x\ p(x)$ may employ different free variables for x.

Now, by two applications of the AMGU-rule, we obtain the substitution $[c/v_1, b/v_2]$ closing the tableau.

Therefore *strictness for* γ should from now on mean that each *instance* of γ (depending on the choice of the free variable) is applied at most once to each γ-formula on any path.

Refutational Completeness

Theorem 3.50 *$\{F_1, \ldots, F_n\}$ satisfiable $\Leftrightarrow$ there exists no closed, strict AMGU-Tableau for $\{F_1, \ldots, F_n\}$.*

For the proof one defines a fair tableau expansion process converging against an infinite tableau where on each path each γ-formula is expanded into all its variants (modulo the choice of the free variable).

One may then again show that each path in that tableau is saturated (up to redundancy) by resolution. This requires to apply the lifting lemma for resolution in order to show completeness of the AMGU-restriction.

How Often Do we Have to Apply γ?

Theorem 3.51 *There is no recursive function $f : F_\Sigma \times F_\Sigma \to \mathbb{N}$ such that, if the closed formula F is unsatisfiable, then there exists a closed tableau for F where to all formulas $\forall xG$ appearing in T the γ-rule is applied at most $f(F, \forall xG)$ times on each path containing $\forall xG$.*

Otherwise unsatisfiability or, respectively, validity for first-order logic would be decidable. In fact, one would be able to enumerate in finite time all tableaux bounded in depth as indicated by f. In other words, free-variable tableaux are not recursively bounded in their depth.

Again $\forall$ is treated like an infinite conjunction. By repeatedly applying γ, together with the substitution rule, one can enumerate all instances $F[t/x]$ vertically, that is, conjunctively, in each path containing $\forall xF$.

Semantic Tableaux vs. Resolution

- Tableaux: global, goal-oriented, "backward".
- Resolution: local, "forward".
- Goal-orientation is a clear advantage if only a small subset of a large set of formulas is necessary for a proof. (Note that resolution provers saturate also those parts of the clause set that are irrelevant for proving the goal.)
- Resolution can be combined with more powerful redundancy elimination methods; because of its global nature this is more difficult for the tableau method.
- Resolution can be refined to work well with equality; for tableaux this seems to be impossible.
- On the other hand tableau calculi can be easily extended to other logics; in particular tableau provers are very successful in modal and description logics.