Now its time to prove completeness for standard first-order tableau. The basic idea is, similar to superposition in the propositional and also first-order case, Section 3.12. If a tableau cannot be closed there exists a model for some branch.

Theorem 3.6.9 (Standard First-Order Tableau is Complete)**.** If ϕ is valid then the tableau calculus computes $\{((\neg\phi), J)\} \Rightarrow^*_{\mathrm{FT}} N$ and N is closed.

Proof. Proof by contradiction. Assume N is not closed. Therefore, it must contain an open branch. By Lemma 3.6.8 this branch constitutes a Hintikka set. By Lemma 3.6.7 the branch constitutes a model for $\neg\phi$, hence ϕ cannot be valid. □

One of the disadvantages of the standard tableau calculus is the guessing of ground terms in γ-Extensions. To get rid of this, the idea is to simply keep the universally quantified variable. Then branches are no longer closed by syntactically complementary formulas, but by complementary formulas modulo "appropriate instantiation" of the universally quantified variables. This requires a procedure that computes, in the simplest case, for two literals a substitution that makes them complementary, i.e., the respective atoms equal. Searching for a substitution making two terms, atoms (formulas) equal is called *unification*, see Section 3.7.

Lemma 3.6.10 (Compactness of First-Order Logic)**.** Let N be a, possibly countably infinite, set of first-order logic ground clauses. Then N is unsatisfiable iff there is a finite subset $N' \subseteq N$ such that N' is unsatisfiable.

Proof. If N is unsatisfiable, saturation via the tableau calculus generates a closed tableau. So there is an i such that $N \Rightarrow^i_{\mathrm{TAB}} N'$ and N' is closed. Every closed branch is the result of finitely many tableau rule applications on finitely many clauses $\{C_1, \ldots, C_n\} \subseteq N$. Let M be the union of all these finite clause sets, so $M \subseteq N$. Tableau is sound, so M is a finite, unsatisfiable subset of N. □

3.7 Unification

Definition 3.7.1 (Unifier)**.** Two terms s and t of the same sort are said to be *unifiable* if there exists a well-sorted substitution σ so that $s\sigma = t\sigma$, the substitution σ is then called a well-sorted *unifier* of s and t. The unifier σ is called a *most general unifier*, written $\sigma = mgu(s, t)$, if any other well-sorted unifier τ of s and t it can be represented as $\tau = \sigma\tau'$, for some well-sorted substitution τ'. A well-sorted subsitution σ is called a *matcher* from s to t, if $s\sigma = t$.

Obviously, two terms of different sort cannot be made equal by well-sorted instantiation. Since well-sortedness is preserved by all rules of the unification

calculus, we assume from now an that all equations, terms, and substitutions are well-sorted.

The first calculus is the naive standard unification calculus that is typically found in the (old) literature on automated reasoning [40]. A state of the naive standard unification calculus is a set of equations E or $\bot$, where $\bot$ denotes that no unifier exists. The set E is also called a *unification problem*. The start state for checking whether two terms s, t, $\mathrm{sort}(s) = \mathrm{sort}(t)$, (or two non-equational atoms A, B) are unifiable is the set $E = \{s = t\}$ ($E = \{A = B\}$). A variable x is *solved* in E if $E = \{x = t\} \uplus E'$, $x \notin \mathrm{vars}(t)$ and $x \notin \mathrm{vars}(E)$.

A variable $x \in \mathrm{vars}(E)$ is called *solved* in E if $E = E' \uplus \{x = t\}$ and $x \notin \mathrm{vars}(t)$ and $x \notin \mathrm{vars}(E')$.

Tautology $E \uplus \{t = t\} \Rightarrow_{\mathrm{SU}} E$

Decomposition $E \uplus \{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)\} \Rightarrow_{\mathrm{SU}} E \cup \{s_1 = t_1, \ldots, s_n = t_n\}$

Clash $E \uplus \{f(s_1, \ldots, s_n) = g(s_1, \ldots, s_m)\} \Rightarrow_{\mathrm{SU}} \bot$
if $f \neq g$

Substitution $E \uplus \{x = t\} \Rightarrow_{\mathrm{SU}} E\{x \mapsto t\} \cup \{x = t\}$
if $x \in vars(E)$ and $x \notin vars(t)$

Occurs Check $E \uplus \{x = t\} \Rightarrow_{\mathrm{SU}} \bot$
if $x \neq t$ and $x \in vars(t)$

Orient $E \uplus \{t = x\} \Rightarrow_{\mathrm{SU}} E \cup \{x = t\}$
if $t \notin \mathcal{X}$

Theorem 3.7.2 (Soundness, Completeness and Termination of $\Rightarrow_{\mathrm{SU}}$)**.** If s, t are two terms with $\mathrm{sort}(s) = \mathrm{sort}(t)$ then

1. if $\{s = t\} \Rightarrow^*_{\mathrm{SU}} E$ then any equation $(s' = t') \in E$ is well-sorted, i.e., $\mathrm{sort}(s') = \mathrm{sort}(t')$.
2. $\Rightarrow_{\mathrm{SU}}$ terminates on $\{s = t\}$.
3. if $\{s = t\} \Rightarrow^*_{\mathrm{SU}} E$ then σ is a unifier (mgu) of E iff σ is a unifier (mgu) of $\{s = t\}$.
4. if $\{s = t\} \Rightarrow^*_{\mathrm{SU}} \bot$ then s and t are not unifiable.
5. if $\{s = t\} \Rightarrow^*_{\mathrm{SU}} \{x_1 = t_1, \ldots, x_n = t_n\}$ and this is a normal form, then $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ is an mgu of s, t.

Proof. 1. by induction on the length of the derivation and a case analysis for the different rules.
2. for a state $E = \{s_1 = t_1, \ldots, s_n = t_n\}$ take the measure $\mu(E) := (n, M, k)$ where n is the number of unsolved variables, M the multiset of all term depths of the s_i, t_i and k the number of equations $t = x$ in E where t is not a variable. The state $\bot$ is mapped to $(0, \emptyset, 0)$. Then the lexicographic combination of $>$ on the naturals and its multiset extension shows that any rule application decrements the measure.
3. by induction on the length of the derivation and a case analysis for the different rules. Clearly, for any state where Clash, or Occurs Check generate $\bot$ the respective equation is not unifiable.
4. a direct consequence of 3.
5. if $E = \{x_1 = t_1, \ldots, x_n = t_n\}$ is a normal form, then for all $x_i = t_i$ we have $x_i \notin \text{vars}(t_i)$ and $x_i \notin \text{vars}(E \setminus \{x_i = t_i\})$, so $\{x_1 = t_1, \ldots, x_n = t_n\}\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\} = \{t_1 = t_1, \ldots, t_n = t_n\}$ and hence $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ is an mgu of $\{x_1 = t_1, \ldots, x_n = t_n\}$. By 3. it is also an mgu of s, t. □

Example 3.7.3 (Size of Standard Unification Problems)**.** Any normal form of the unification problem E given by

$$\{f(x_1, g(x_1, x_1), x_3, \ldots, g(x_n, x_n)) = f(g(x_0, x_0), x_2, g(x_2, x_2), \ldots, x_{n+1})\}$$

with respect to $\Rightarrow_{\text{SU}}$ is exponentially larger than E.

Note that the exponential growth cannot happen for a matching problem. In order to find a matcher σ from s to t, i.e., $s\sigma = t$ the size of the substituion σ is bound by t, thus linear in size of the input s, t. The rules of $\Rightarrow_{\text{SU}}$ without Orient and Occurs check are sufficient to compute a matcher, see Exercise **??**.

The second calculus, polynomial unification, prevents the problem of exponential growth by introducing an implicit representation for the mgu. For this calculus the size of a normal form is always polynomial in the size of the input unification problem.

Tautology $E \uplus \{t = t\} \Rightarrow_{\text{PU}} E$

Decomposition $E \uplus \{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)\} \Rightarrow_{\text{PU}} E \uplus \{s_1 = t_1, \ldots, s_n = t_n\}$

Clash $E \uplus \{f(t_1, \ldots, t_n) = g(s_1, \ldots, s_m)\} \Rightarrow_{\text{PU}} \bot$
if $f \neq g$

Occurs Check $E \uplus \{x = t\} \Rightarrow_{\text{PU}} \bot$
if $x \neq t$ and $x \in vars(t)$

Orient $E \uplus \{t = x\} \Rightarrow_{\text{PU}} E \uplus \{x = t\}$

if $t \notin \mathcal{X}$

Substitution $E \uplus \{x = y\} \Rightarrow_{\mathrm{PU}} E\{x \mapsto y\} \uplus \{x = y\}$
if $x \in vars(E)$ and $x \neq y$

Cycle $E \uplus \{x_1 = t_1, \ldots, x_n = t_n\} \Rightarrow_{\mathrm{PU}} \bot$
if there are positions p_i with $t_i|_{p_i} = x_{i+1}, t_n|_{p_n} = x_1$ and some $p_i \neq \epsilon$

Merge $E \uplus \{x = t, x = s\} \Rightarrow_{\mathrm{PU}} E \uplus \{x = t, t = s\}$
if $t, s \notin \mathcal{X}$ and $|t| \leq |s|$

Theorem 3.7.4 (Soundness, Completeness and Termination of $\Rightarrow_{\mathrm{PU}}$)**.** If s, t are two terms with $\mathrm{sort}(s) = \mathrm{sort}(t)$ then

1. if $\{s = t\} \Rightarrow^*_{\mathrm{PU}} E$ then any equation $(s' = t') \in E$ is well-sorted, i.e., $\mathrm{sort}(s') = \mathrm{sort}(t')$.
2. $\Rightarrow_{\mathrm{PU}}$ terminates on $\{s = t\}$.
3. if $\{s = t\} \Rightarrow^*_{\mathrm{PU}} E$ then σ is a unifier (mgu) of E iff σ is a unifier (mgu) of $\{s = t\}$.
4. if $\{s = t\} \Rightarrow^*_{\mathrm{PU}} \bot$ then s and t are not unifiable.

Theorem 3.7.5 (Normal Forms generated by $\Rightarrow_{\mathrm{PU}}$)**.** Let $\{s = t\} \Rightarrow^*_{\mathrm{PU}} \{x_1 = t_1, \ldots, x_n = t_n\}$ be a normal form. Then

1. $x_i \neq x_j$ for all $i \neq j$ and without loss of generality $x_i \notin \mathrm{vars}(t_{i+k})$ for all i, k, $1 \leq i < n$, $i + k \leq n$.
2. the substitution $\{x_1 \mapsto t_1\}\{x_2 \mapsto t_2\} \ldots \{x_n \mapsto t_n\}$ is an mgu of $s = t$.

Proof. 1. If $x_i = x_j$ for some $i \neq j$ then Merge is applicable. If $x_i \in \mathrm{vars}(t_i)$ for some i then Occurs Check is applicable. If the x_i cannot be ordered in the described way, then either Substitution or Cycle is applicable.
2. Since $x_i \notin \mathrm{vars}(t_{i+k})$ the composition yields the mgu. □

Lemma 3.7.6 (Size of Unifiers)**.** Let $\{s = t\}$ be a unification problem between two non-variable terms. Then

1. if s and t are linear then for any unifier σ and any term $r \in \mathrm{codom}(\sigma)$, $|r| < |s|$ and $|r| < |t|$ as well as $\mathrm{depth}(r) < \mathrm{depth}(s)$ and $\mathrm{depth}(r) < \mathrm{depth}(t)$,
2. if s is shallow and linear, then the mgu σ of s and t is also a matcher from s to t, i.e., $s\sigma = t$

Proof. Both parts follow directly from the structure of the terms s, t: if they are both linear then the substitution rule is never applied. If s is shallow and linear, it has the form $f(x_1, \ldots, x_n)$, all x_i different, then the unifier is $\sigma = \{x_i \mapsto t|_i \mid 1 \leq i \leq n\}$. □

C In addition to the consideration of repeated subformulas, discussed in Section 2.5, for first-order renaming another technique can pay off: generalization. Consider the formula $[\phi_1 \vee (Q_1(a_1) \wedge Q_2(a_1))] \wedge [\phi_2 \vee (Q_1(a_2) \wedge Q_2(a_2))] \wedge \ldots \wedge [\phi_n \vee (Q_1(a_n) \wedge Q_2(a_n)]$. SimpleRenaming on obvious renamings applied to this formula will independently rename any occurrences of a formula $(Q_1(a_i) \wedge Q_2(a_i))$. However generalization pays off here. By adding the definition $\forall x, y\,(R(x,y) \rightarrow (Q_1(x) \wedge Q_2(y)))$ and replacing the i^{th} occurrence of the conjunct by $R(x,y)\{x \mapsto a_i, y \mapsto a_i\}$ one definition for all subformula occurrences suffices.

3.10 First-Order Resolution

As already mentioned, I still consider first-order logic without equality. First-order resolution on ground clauses corresponds to propositional resolution. Each ground atom becomes a propositional variable. However, since there are up to infinitely many ground instances for a first-order clause set with variables and it is not a priori known which ground instances are needed in a proof, the first-order resolution calculus operates on clauses with variables. Roughly, the relationship between ground resolution and first-order resolution corresponds to the relationship between standard tableau and free-variable tableau. However, the variables in free-variable tableau can only be instantiated once, whereas in resolution they can be instantiated arbitrarily often.

Propositional (or first-order ground) resolution is refutationally complete, without reduction rules it is not guaranteed to terminate on satisfiable sets of clauses, and inferior to the CDCL calculus. However, in contrast to the CDCL calculus, resolution can be easily extended to non-ground clauses via unification. The problem to generalize the CDCL calculus lies in the generalization of the model representation. For example, whereas in propositional logic the maximal size of a partial model (trail) is linear in the size of a clause set, this does not hold for first-order logic. There can't even be an overall finite model representation for all satisfiable first-order clause sets. I'll discuss this in more detail in Section 3.15.

Lemma 3.10.1. Let $\mathcal{A}$ be a $\Sigma - algebra$ and let ϕ be a $\Sigma - formula$ with free variables $x_1, \ldots, x_n$. Then $\mathcal{A} \models \forall x_1, \ldots, x_n \phi$ iff $\mathcal{A} \models \phi$

Lemma 3.10.2. Let ϕ be a $\Sigma - formula$ with free variables $x_1, \ldots, x_n$, let σ be a substitution and let $y_1, \ldots, y_m$ be free variables of $\phi\sigma$. Then $\mathcal{A} \models \forall x_1, \ldots, x_n \phi$ implies $\mathcal{A} \models \forall y_1, \ldots, y_m \phi\sigma$.

In particular, if $\mathcal{A}$ is a model of an (implicitly universally quantified) clause C then it is also a model of all (implicitly universally quantified) instances $C\sigma$ of C. Consequently, if it is shown that some instances of clauses in a set N are unsatisfiable then it is also shown that N itself is unsatisfiable.

General Resolution through Instantiation

The approach is to instantiate clauses appropriately. An example is shown in Figure 3.3. However, this may lead to several problems. First of all, more than one instance of a clause can participate in a proof and secondly, which is even worse, there are infinitely many possible instances. Due to the fact that instantiation must produce complementary literals so that inferences become possible, the idea is to not instantiate more than necessary to get complementary literals. An instantiation of the clause set from Figure 3.3 is again shown in Figure 3.4 with the difference that the latter instantiates only as much as necessary, inevitably reducing the number of substitutions.

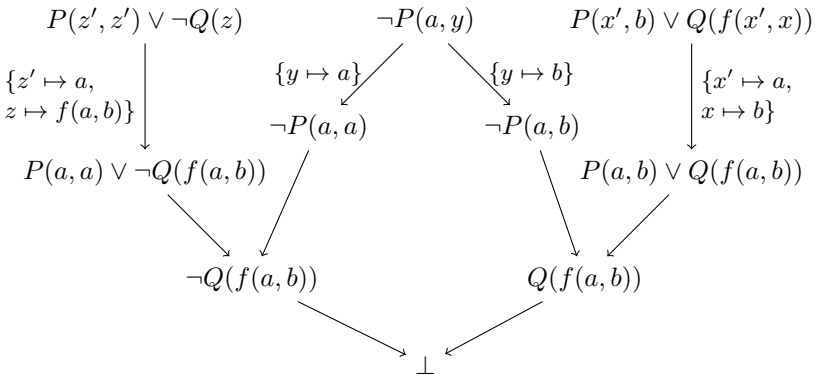

Figure 3.3: Instantiation of the clause set $N = P(z', z') \vee \neg Q(z), \neg P(a, y), P(x', b) \vee Q(f(x', x))$

Lifting Principle

In order to overcome the problem of effectively and efficiently saturating infinite sets of clauses as they arise from taking the (ground) instances of finitely many *general* clauses (with variables), the general idea is to lift the resolution principle as proposed by Robinson [77]. The lifting is as follows: For the resolution of general clauses, *equality* of ground atoms is generalized to *unifiability* of general atoms and only the *most general* (minimal) unifiers (mgu) are computed.

The advantage of the method in Robinson [77] compared with Gilmore [43] 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.

The *first-order resolution calculus* consists of the inference rules *Resolution* and *Factoring* and generalizes the propositional resolution calculus (Section 2.6). Variables in clauses are implicitly universally quantified, so they can be instantiated in an arbitrary way. For the application of any inference or reduction rule,

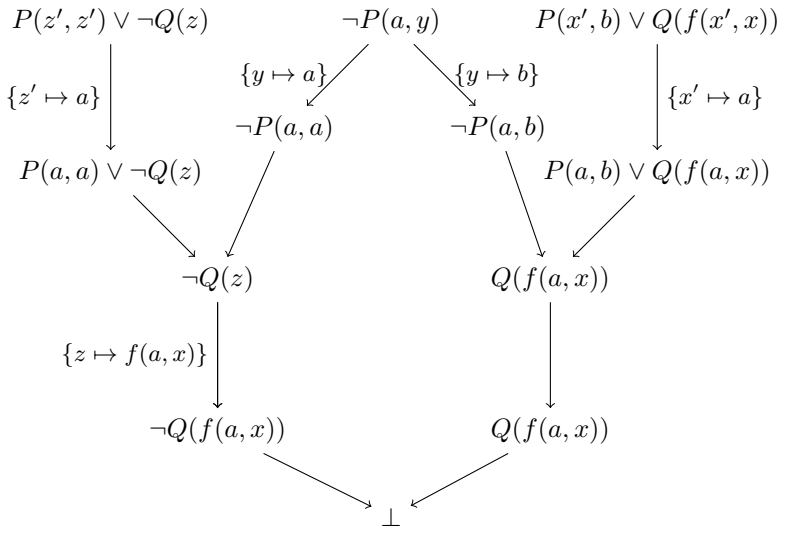

Figure 3.4: Instantiation of the clause set $N = P(z', z') \vee \neg Q(z), \neg P(a, y), P(x', b) \vee Q(f(x', x))$ with a reduced number of instantiations.

I can therefore assume that the involved clauses don't share any variables, i.e., variables are a priori renamed. Furthermore, clauses are assumed to be unique with respect to renaming in a set.

Resolution $(N \uplus \{D \vee A, \neg B \vee C\}) \Rightarrow_{\text{RES}} (N \cup \{D \vee A, \neg B \vee C\} \cup \{(D \vee C)\sigma\})$
if $\sigma = mgu(A, B)$ for atoms A, B

Factoring $(N \uplus \{C \vee L \vee K\}) \Rightarrow_{\text{RES}} (N \cup \{C \vee L \vee K\} \cup \{(C \vee L)\sigma\})$
if $\sigma = mgu(L, K)$ for literals L, K

The reduction rules are

Subsumption $(N \uplus \{C_1, C_2\}) \Rightarrow_{\text{RES}} (N \cup \{C_1\})$
provided $C_1\sigma \subset C_2$ for some matcher σ

Tautology Deletion $(N \uplus \{C \vee A \vee \neg A\}) \Rightarrow_{\text{RES}} (N)$

Condensation $(N \uplus \{C\}) \Rightarrow_{\text{RES}} (N \cup \{C'\})$
where C' is the result of removing duplicate literals from $C\sigma$ for some matcher σ and C' subsumes C

Subsumption Resolution $(N \uplus \{C_1 \vee L, C_2 \vee K\}) \Rightarrow_{\mathrm{RES}} (N \cup \{C_1 \vee L, C_2\})$
where $L\sigma = \mathrm{comp}(K)$ and $C_1\sigma \subseteq C_2$

Lemma 3.10.3 (Lifting Lemma)**.** Let C and D be variable-disjoint clauses. There is one version for Resolution and one for Factoring.

(i) if $(N \uplus \{D\delta \vee A\delta, \neg B\gamma \vee C\gamma\}) \Rightarrow_{\mathrm{RES}} (N \cup \{D\delta \vee A\delta, \neg B\gamma \vee C\gamma\} \cup \{(D\delta \vee C\gamma)\sigma\})$ where $\sigma = mgu(A\delta, B\gamma)$ then $(N \uplus \{D \vee A, \neg B \vee C\}) \Rightarrow_{\mathrm{RES}} (N \cup \{D \vee A, \neg B \vee C\} \cup \{(D \vee C)\sigma'\})$ where $\sigma' = mgu(A, B)$ and $(D \vee C)\sigma'\delta\gamma\sigma = (D\delta \vee C\gamma)\sigma$.

Saturation of Sets of General Clauses

Definition 3.10.4 (Resolution Saturation)**.** A set of clauses N is *saturated* up to redundancy if for all $C \in \mathrm{Res}(N)$ it holds $C \in N$ or C is subsumed by a clause from N.

Corollary 3.10.5. Let N be a set of general clauses saturated under Res, i.e., $\mathrm{Res}(N) \subseteq N$. Then also $G_\Sigma(N)$ is saturated, that is, $\mathrm{Res}(G_\Sigma(N)) \subseteq G_\Sigma(N)$.

Proof. W.l.o.g. assume that clauses in N are pairwise variable-disjoint. (Otherwise they have to be made disjoint and this renaming process changes neither $Res(N)$ nor $G_\Sigma(N)$.) Let $C' \in Res(G_\Sigma(N))$, meaning (i) there exist resolvable ground instances $D\sigma$ and $C\rho$ 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, D and C are resolvable with a resolvent C'' with $C''\tau = C'$, for a suitable substitution τ. As $C'' \in N$ by assumption, $C' \in G_\Sigma(N)$ is obtained.

Case (ii): Similar. □

Herbrand's Theorem

Lemma 3.10.6. 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 3.10.7. Let N be a set of Σ-clauses, let $\mathcal{A}$ be a *Herbrand* interpretation. Then $\mathcal{A} \models G_\Sigma(N)$ implies $\mathcal{A} \models N$.

Theorem 3.10.8 (Herbrand)**.** A set N of Σ-clauses is satisfiable if and only if it has a Herbrand model over Σ.

Proof. ($\Leftarrow$) Assume N has a Herbrand model I over Σ, i.e., $I \models N$. Then N is satisfiable.

($\Rightarrow$) Let $N \not\models \bot$.

$N \not\models \bot$	$\Rightarrow$	$\bot \notin Res^*(N)$	(resolution is sound)
	$\Rightarrow$	$\bot \notin G_\Sigma(Res^*(N))$	
	$\Rightarrow$	$I_{G_\Sigma(Res^*(N))} \models G_\Sigma(Res^*(N))$	(Theorem ; Corollary 3.10.5)
	$\Rightarrow$	$I_{G_\Sigma(Res^*(N))} \models Res^*(N)$	(Lemma 3.10.7)
	$\Rightarrow$	$I_{G_\Sigma(Res^*(N))} \models N$	($N \subseteq Res^*(N)$)

□

The Theorem of Löwenheim-Skolem

Theorem 3.10.9 (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 3.5.5. □

Refutational Completeness of General Resolution

Theorem 3.10.10 (Soundness and Completeness of Resolution)**.** The resolution calculus is sound and complete:

$$N \text{ is unsatisfiable iff } N \Rightarrow^*_{\mathrm{RES}} N' \text{ and } \bot \in N' \text{ for some } N'$$

Theorem 3.10.11 (Soundness and Completeness of Resolution)**.** Let N be a set of first-clauses where $Res(N) \subseteq N$. Then

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

Proof. Let $Res(N) \subseteq N$. By Corollary 3.10.5: $Res(G_\Sigma(N)) \subseteq G_\Sigma(N)$

$$\begin{array}{lll} N \models \bot & \Leftrightarrow G_\Sigma(N) \models \bot & \text{(Lemma 3.10.6/3.10.7; Theorem 3.5.5)} \\ & \Leftrightarrow \bot \in G_\Sigma(N) & \text{(propositional resolution sound and complete)} \\ & \Leftrightarrow \bot \in N & \end{array}$$

□

Compactness of First-Order Logic

Theorem 3.10.12 (Compactness Theorem for First-Order Logic)**.** Let S be a set of first-order formulas. S is unsatisfiable if and only if some finite subset $S' \subseteq S$ is unsatisfiable.

Proof. ($\Leftarrow$) Assume that S' is unsatisfiable. Since $S' \subseteq S$, S is also unsatisfiable.

($\Rightarrow$) Let S be unsatisfiable and let N be the set of clauses obtained by Skolemization and CNF transformation of the formulas in S. Clearly $Res^*(N)$ is unsatisfiable. By Theorem 3.10.11, $\bot \in Res^*(N)$, 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 S' as the subset of formulas in S so that the corresponding clauses contain the assumptions (leaves) of B. □