Automated Theorem Proving

Lecture 13: Superposition Continued

Prof. Dr. Jasmin Blanchette

based on slides by Dr. Uwe Waldmann

Winter Term 2025/26

5.4 Superposition: Refutational Completeness

For a set E of ground equations, $\mathrm{T}_\Sigma(\emptyset)/E$ is an E-interpretation (or E-algebra) with universe $\{[t] \mid t \in \mathrm{T}_\Sigma(\emptyset)\}$.

One can show (similar to the proof of Birkhoff's Theorem) that for every *ground* equation $s \approx t$ we have $\mathrm{T}_\Sigma(\emptyset)/E \models s \approx t$ if and only if $s \leftrightarrow_E^* t$.

In particular, if E is a convergent set of rewrite rules R and $s \approx t$ is a ground equation, then $\mathrm{T}_\Sigma(\emptyset)/R \models s \approx t$ if and only if $s \downarrow_R t$.
By abuse of terminology, we say that an equation or clause is valid (or true) in R if and only if it is true in $\mathrm{T}_\Sigma(\emptyset)/R$.

Superposition: Refutational Completeness

Construction of candidate interpretations
(Bachmair and Ganzinger 1990):

Let N be a set of clauses not containing $\bot$.
Using induction on the clause ordering we define sets of rewrite rules E_C and R_C for all $C \in G_\Sigma(N)$ as follows:

Assume that E_D has already been defined for all $D \in G_\Sigma(N)$ with $D \prec_C C$.
Then $R_C = \bigcup_{D \prec_C C} E_D$.

Superposition: Refutational Completeness

The set E_C contains the rewrite rule $s \to t$ if

(a) $C = C' \vee s \approx t$.

(b) $s \approx t$ is strictly maximal in C.

(c) $s \succ t$.

(d) C is false in R_C.

(e) C' is false in $R_C \cup \{s \to t\}$.

(f) s is irreducible w.r.t. R_C.

In this case, C is called productive. Otherwise $E_C = \emptyset$.

Finally, $R_\infty = \bigcup_{D \in G_\Sigma(N)} E_D$.

Superposition: Refutational Completeness

Example:

We use the lpo with the precedence $f \succ e \succ d \succ c \succ b \succ a$ (max. side of max. literals in red).

Let $N = \{d \approx c,\ b \approx a \lor e \not\approx c,\ b \not\approx b \lor f(b) \approx a,\ f(c) \approx b,\ f(b) \approx a \lor f(c) \not\approx b,\ f(b) \approx a \lor f(d) \not\approx b\}$ be a clause set saturated w.r.t. the ground superposition calculus.

The next table shows each iteration of the candidate interpretation construction for N.

Superposition: Refutational Completeness

Iter.	Clause C	R_C	E_C
0	$d \approx c$	$\emptyset$	$\{d \to c\}$
1	$b \approx a \vee e \not\approx c$	$\{d \to c\}$	$\emptyset$
2	$b \not\approx b \vee f(b) \approx a$	$\{d \to c\}$	$\{f(b) \to a\}$
3	$f(c) \approx b$	$\{d \to c,\ f(b) \to a\}$	$\{f(c) \to b\}$
4	$f(b) \approx a \vee f(c) \not\approx b$	$\{d \to c,\ f(b) \to a,\ f(c) \to b\}$	$\emptyset$
5	$f(b) \approx a \vee f(d) \not\approx b$	$\{d \to c,\ f(b) \to a,\ f(c) \to b\}$	$\emptyset$

At each iteration $i + 1$, the term rewriting system consists of the union of the rewrite rules R_C and the "epsilon" E_C of iteration i. The interpretation $R_\infty = \{d \to c,\ f(b) \to a,\ f(c) \to b\}$ after iteration 5 is a model of N.

Superposition: Refutational Completeness

Lemma 5.4.1:
If $E_C = \{s \to t\}$ and $E_D = \{u \to v\}$, then $s \succ u$ if and only if $C \succ_{\mathrm{C}} D$.

Corollary 5.4.2:
The rewrite systems R_C and R_∞ are convergent (i.e., terminating and confluent).

Superposition: Refutational Completeness

Lemma 5.4.3:
If $D \preceq_{\mathrm{C}} C$ and $E_C = \{s \to t\}$, then $s \succ u$ for every term u occurring in a negative literal in D and $s \succeq u$ for every term u occurring in a positive literal in D.

Corollary 5.4.4:
If $D \in G_\Sigma(N)$ is true in R_D, then D is true in R_∞ and R_C for all $C \succ_{\mathrm{C}} D$.

Corollary 5.4.5:
If $D = D' \vee u \approx v$ is productive, then D' is false and D is true in R_∞ and R_C for all $C \succ_{\mathrm{C}} D$.

Superposition: Refutational Completeness

Lemma 5.4.6 ("Lifting Lemma"):
Let C be a clause and let θ be a substitution such that $C\theta$ is ground. Then every equality resolution or equality factoring inference from $C\theta$ is a ground instance of an inference from C.

Proof:
Omitted. $\square$

Superposition: Refutational Completeness

Lemma 5.4.7 ("Lifting Lemma"):
Let $D = D' \vee u \approx v$ and $C = C' \vee [\neg]\, s \approx t$ be two clauses (without common variables) and let θ be a substitution such that $D\theta$ and $C\theta$ are ground.

If there is a superposition inference between $D\theta$ and $C\theta$ where $u\theta$ and some subterm of $s\theta$ are overlapped, and $u\theta$ does not occur in $s\theta$ at or below a variable position of s, then the inference is a ground instance of a superposition inference from D and C.

Proof:
Omitted. $\square$

Superposition: Refutational Completeness

Theorem 5.4.8 (“Model Construction”):
Let N be a set of clauses that is saturated up to redundancy and does not contain the empty clause. Then we have for every ground clause $C\theta \in G_\Sigma(N)$:

(i) $E_{C\theta} = \emptyset$ if and only if $C\theta$ is true in $R_{C\theta}$.

(ii) If $C\theta$ is redundant w.r.t. $G_\Sigma(N)$, then it is true in $R_{C\theta}$.

(iii) $C\theta$ is true in R_∞ and in R_D for every $D \in G_\Sigma(N)$ with $D \succ_{\mathrm{c}} C\theta$.

Superposition: Refutational Completeness

A Σ-interpretation $\mathcal{A}$ is called term-generated if for every $b \in U_{\mathcal{A}}$ there is a ground term $t \in \mathrm{T}_{\Sigma}(\emptyset)$ such that $b = \mathcal{A}(\beta)(t)$.

Lemma 5.4.9:
Let N be a set of (universally quantified) Σ-clauses and let $\mathcal{A}$ be a term-generated Σ-interpretation.
Then $\mathcal{A}$ is a model of $G_{\Sigma}(N)$ if and only if it is a model of N.

Superposition: Refutational Completeness

Theorem 5.4.10 (Refutational Completeness: Static View):
Let N be a set of clauses that is saturated up to redundancy.
Then N has a model if and only if N does not contain the empty clause.

Superposition: Refutational Completeness

So far, we have considered only inference rules that add new clauses to the current set of clauses
(corresponding to the "Deduce" rule of Knuth–Bendix completion).

In other words, we have derivations of the form $N_0 \vdash N_1 \vdash N_2 \vdash \cdots$, where each N_{i+1} is obtained from N_i by adding the consequence of some inference from clauses in N_i.

Under which circumstances are we allowed to delete (or simplify) a clause during the derivation?

Superposition: Refutational Completeness

A run of the superposition calculus is a sequence
$N_0 \vdash N_1 \vdash N_2 \vdash \cdots$ such that
(i) $N_i \models N_{i+1}$, and
(ii) all clauses in $N_i \setminus N_{i+1}$ are redundant w.r.t. N_{i+1}.

In other words, during a run we may add a new clause if it follows from the old ones, and we may delete a clause if it is redundant w.r.t. the remaining ones.

For a run, $N_\infty = \bigcup_{i \geq 0} \bigcap_{j \geq i} N_j$.
The set N_∞ of all persistent clauses is called the limit of the run.

Superposition: Refutational Completeness

Lemma 5.4.11:

If $N \subseteq N'$, then $Red(N) \subseteq Red(N')$.

Proof:

Obvious. $\square$

Superposition: Refutational Completeness

Lemma 5.4.12:
If $N' \subseteq Red(N)$, then $Red(N) \subseteq Red(N \setminus N')$.

Proof:
Omitted. $\square$

Superposition: Refutational Completeness

Lemma 5.4.13:

Let $N_0 \vdash N_1 \vdash N_2 \vdash \cdots$ be a run.

Then $Red(N_i) \subseteq Red(\bigcup_{j \geq 0} N_j)$ and $Red(N_i) \subseteq Red(N_\infty)$ for every i.

Proof:

Omitted. $\square$

Superposition: Refutational Completeness

Corollary 5.4.14:

$N_i \subseteq N_\infty \cup \mathit{Red}(N_\infty)$ for every i.

Proof:

If $C \in N_i \setminus N_\infty$, then there is a $k \geq i$ such that $C \in N_k \setminus N_{k+1}$.

Therefore C must be redundant w.r.t. N_{k+1}.

Consequently, C is redundant w.r.t. N_∞. $\square$

Superposition: Refutational Completeness

A run is called fair if the conclusion of every inference from clauses in $N_\infty \setminus Red(N_\infty)$ is contained in some $N_i \cup Red(N_i)$.

Lemma 5.4.15:
If a run is fair, then its limit is saturated up to redundancy.

Proof:
If the run is fair, then the conclusion of every inference from nonredundant clauses in N_∞ is contained in some $N_i \cup Red(N_i)$, and therefore contained in $N_\infty \cup Red(N_\infty)$.
Hence N_∞ is saturated up to redundancy. $\square$

Superposition: Refutational Completeness

Theorem 5.4.16 (Refutational Completeness: Dynamic View):

Let $N_0 \vdash N_1 \vdash N_2 \vdash \cdots$ be a fair run, let N_∞ be its limit.

Then N_0 has a model if and only if $\bot \notin N_\infty$.

5.5 Improvements and Refinements

The superposition calculus as described so far can be improved and refined in several ways.

Concrete Redundancy and Simplification Criteria

Redundancy is undecidable.

Even decidable approximations are often expensive
(experimental evaluations are needed to see what pays off in practice).

Often a clause can be *made* redundant by adding another clause that is entailed by the existing ones.

This process is called simplification.

Concrete Redundancy and Simplification Criteria

Examples:

Subsumption:
If N contains clauses D and $C = C' \vee D\sigma$, where C' is nonempty, then D subsumes C and C is redundant.

Example:
$f(x) \approx g(x)$ subsumes $f(y) \approx a \vee f(h(y)) \approx g(h(y))$.

Concrete Redundancy and Simplification Criteria

Examples:

Trivial literal elimination:

Duplicated literals and trivially false literals can be deleted:

A clause $C' \vee L \vee L$ can be simplified to $C' \vee L$;

a clause $C' \vee s \not\approx s$ can be simplified to C'.

Concrete Redundancy and Simplification Criteria

Examples:

Condensation:
If we obtain a clause D from C by applying a substitution, followed by deletion of duplicated literals, and if D subsumes C, then C can be simplified to D.

Example:
By applying $\{y \to g(x)\}$ to $C \;=\; f(g(x)) \approx a \vee f(y) \approx a$ and deleting the duplicated literal, we obtain $f(g(x)) \approx a$, which subsumes C.

Concrete Redundancy and Simplification Criteria

Examples:

Semantic tautology deletion:
Every clause that is a tautology is redundant. Note that in the nonequational case, a clause is a tautology if and only if it contains two complementary literals, whereas in the equational case we need a congruence closure algorithm to detect that a clause like $x \not\approx y \vee f(x) \approx f(y)$ is tautological.

Concrete Redundancy and Simplification Criteria

Examples:

Rewriting:
If N contains a unit clause $D = s \approx t$ and a clause $C[s\sigma]$, such that $s\sigma \succ t\sigma$ and $C \succ_{\mathrm{C}} D\sigma$, then C can be simplified to $C[t\sigma]$.

Example:
If $D = f(x, x) \approx g(x)$ and $C = h(f(g(y), g(y))) \approx h(y)$, and $\succ$ is an lpo with the precedence $h \succ f \succ g$, then C can be simplified to $h(g(g(y))) \approx h(y)$.

Selection Functions

Like the ordered resolution calculus, superposition can be used with a selection function that overrides the ordering restrictions for negative literals.

A selection function is a mapping

$$S : C \quad \mapsto \quad \text{set of occurrences of } \textit{negative} \text{ literals in } C$$

We indicate selected literals by a box:

$$\boxed{\neg f(x) \approx a} \vee g(x, y) \approx g(x, z)$$

Selection Functions

The second ordering condition for inferences is replaced by

– Either the last literal in each premise is selected or there is no selected literal in the premise and the literal is maximal in the premise (strictly maximal for positive literals in superposition inferences).

In particular, clauses with selected literals can only be used in equality resolution inferences and as the second premise in negative superposition inferences.

Selection Functions

Refutational completeness is proved essentially as before:

We assume that each ground clause in $G_\Sigma(N)$ inherits the selection of one of the clauses in N of which it is a ground instance (there may be several ones).

In the proof of the model construction theorem, we replace case 3 by "$C\theta$ contains a selected or maximal negative literal" and case 4 by "$C\theta$ contains neither a selected nor a maximal negative literal."

In addition, for the induction proof of this theorem we need one more property, namely:
(iv) If $C\theta$ has selected literals then $E_{C\theta} = \emptyset$.

Redundant Inferences

So far, we have defined saturation in terms of redundant clauses:

> N is saturated up to redundancy if the conclusion of every inference from clauses in $N \setminus \mathit{Red}(N)$ is contained in $N \cup \mathit{Red}(N)$.

This definition ensures that in the proof of the model construction theorem, the conclusion $C_0\theta$ of a ground inference follows from clauses in $G_\Sigma(N)$ that are smaller than or equal to itself,
hence they are smaller than the premise $C\theta$ of the inference,
hence they are true in $R_{C\theta}$ by induction.

Redundant Inferences

However, a closer inspection of the proof shows that it is actually sufficient that the clauses from which $C_0\theta$ follows are smaller than $C\theta$—it is *not* necessary that they are smaller than $C_0\theta$ itself.
This motivates the following definition of redundant *inferences*:

A ground inference with conclusion C_0 and right (or only) premise C is called redundant w.r.t. a set of ground clauses N if one of its premises is redundant w.r.t. N, or if C_0 follows from clauses in N that are smaller than C.

An inference is redundant w.r.t. a set of clauses N if all its ground instances are redundant w.r.t. $G_\Sigma(N)$.

Redundant Inferences

Recall that a clause can be redundant w.r.t. N without being contained in N.
Analogously, an inference can be redundant w.r.t. N without being an inference from clauses in N.

The set of all inferences that are redundant w.r.t. N is denoted by $\mathit{RedInf}(N)$.

Redundant Inferences

Saturation is then redefined in the following way:

N is saturated up to redundancy if every inference from clauses in N is redundant w.r.t. N.

Using this definition, the model construction theorem can be proved essentially as before.

Redundant Inferences

The connection between redundant inferences and clauses is given by the following lemmas. They are proved in the same way as the corresponding lemmas for redundant clauses:

Lemma 5.5.1:
If $N \subseteq N'$, then $\mathit{RedInf}(N) \subseteq \mathit{RedInf}(N')$.

Lemma 5.5.2:
If $N' \subseteq \mathit{Red}(N)$, then $\mathit{RedInf}(N) \subseteq \mathit{RedInf}(N \setminus N')$.