Completeness of Superposition

Andrei Voronkov

The University of Manchester
EasyChair

Proof of Completeness

A trace of a clause C: a set of clauses $\{C_1, \ldots, C_n\} \subseteq S_\omega$ such that

1. $C \succeq C_i$ for all $i = 1, \ldots, n$;
2. $C_1, \ldots, C_n \models C$.

Lemma 1. Every clause C occurring in any S_i has a trace.

Lemma 2. The limit S_ω is saturated up to redundancy.

Lemma 3. The limit S_ω is logically equivalent to the initial set S_0.

Lemma 4. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Interestingly, only the last lemma uses the rules of $\mathbb{S}\mathrm{up}_{\succ,\sigma}$ and the fact that σ is well-behaving.

Rewrite Rule Systems

- A rewrite rule is an expression $l \to r$.
- A rewrite rule system R is a set of rewrite rules.
- We say that a rewrite rule $l \to r$ rewrites $s[l]$ into $s[r]$.
- We write $s \to_R t$ if some rewrite rule in R rewrites s into t. The relation $\to_R^*$ is the reflexive and transitive closure of $\to_R$. In other words, we have $s \to_R^* t$, if there exists a sequence of terms $t_0, \ldots, t_n$ such that, $n \geq 0$, $s = t_0$, $t_n = t$ and we have

$$t_0 \to_R t_1 \to_R \cdots \to_R t_n.$$

  In this case we also say that s rewrites into t in n steps using R.

Rewrite Rule Systems

- t is irreducible w.r.t. R is no rule in R rewrites t.
- R is called terminating if there is no infinite sequence

$$t_0 \rightarrow_R t_1 \rightarrow_R t_2 \rightarrow_R \cdots$$

- t is called a normal form of s w.r.t. R if $s \rightarrow_R^* t$ and t is irreducible w.r.t. R.
- If R is terminating, then every term has a normal form
- R is called convergent if it is terminating and every term has a unique normal form.
- R is called non-overlapping if for every two different rules $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ in R, l_1 is not a subterm of l_2.
- If R is terminating and non-overlapping, then it is convergent

Congruence

- A congruence relation is a relation satisfying equality axioms (reflexivity, symmetry, transitivity and congruence);
- Any convergent system R defines a congruence relation, denoted by $=_R$ as follows: $s =_R t$ iff s and t have the same normal form.

Model Construction

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Model Construction

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\Box \in S$.

Let set S be saturated up to redundancy and $\Box \notin S$. We will build a congruence relation satisfying all clauses in S.

Model Construction

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Let set S be saturated up to redundancy and $\square \notin S$. We will build a congruence relation satisfying all clauses in S.

We will build this relation as $=_R$ for some convergent rewrite system R, which will be built step by step by induction on $\succ$. R can be growing during the model construction. At each step, we denote by I_R the interpretation in which the equality is defined as $=_R$.

Model Construction

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Let set S be saturated up to redundancy and $\square \notin S$. We will build a congruence relation satisfying all clauses in S.

We will build this relation as $=_R$ for some convergent rewrite system R, which will be built step by step by induction on $\succ$. R can be growing during the model construction. At each step, we denote by I_R the interpretation in which the equality is defined as $=_R$.

[Step l], where l is a ground term. We assume that R was built for all terms $t \prec l$. Then, if

1. l is irreducible w.r.t. R;
2. S contains a clause $\underline{l \simeq r} \vee C$ such that (i) $I_R \not\models C$ (ii) $l \succ r$; (iii) l is greater than any term in C.

Then take any such clause and add $l \rightarrow r$ to R

Model Construction

We claim:

1. R is convergent;
2. I_R satisfies all clauses in S.

(Proof on the board)

Proof

Note that:

- R is non-overlapping by construction;
- R is terminating (because $\succ$ is monotonic and well-founded).

These two properties imply that R is convergent.

Proof

Some general properties of the model construction:

- If l is irreducible after step l, then it will be irreducible after
- The normal form of a term l does not change after step l
- If C is a clause in which l is the greatest term, then the truth value of C in I_R does not change after step l

(*) Lemma. If $C \in S_i$ is false in I, then there is a persistent clause $C' \in S_\omega$ such that $C \succeq C'$ and C' is false in I too.

Proof

Now we prove that I_R satisfies all clauses in S. Suppose it does not. Then there is a clause $F \in S$ such that $I_R \not\models F$. Note that F is non-empty, since S does not contain the empty clause.

Since $\succ$ is well-founded on clauses, the set of all clauses in S, which are false in I_R contains the least element. Denote this clause by F.

We will now show, by contradiction, that S contains a clause smaller than F and false in I_R. To prove this, we consider several cases, depending on which literal(s) are selected in F.

Proof Case

Case 1: F has a negative selected literal. Then F has the form $\underline{s \not\simeq t} \vee D$.
Case 1.1: s coincides with t, then F has the form $\underline{s \not\simeq s} \vee D$. Consider the equality resolution inference

$$\frac{\underline{s \not\simeq s} \vee D}{D}$$

Since F is persistent and the process is fair, this inference was applied at some step, so D belongs to some search space S_i. Note that $F \succ D$ and $D \vdash F$. Then we have $I_R \not\models D$.

By (*), there is a persistent clause D' such that $D \succeq D'$ and $I_R \not\models D'$, and we are done.

Proof Case

Case 1.2: F has the form $\underline{s \not\simeq t} \vee D$ and s does not coincide with t.
W.l.o.g. assume $s \succ t$.

Proof Case

Case 1.2: F has the form $\underline{s \not\approx t} \vee D$ and s does not coincide with t.
W.l.o.g. assume $s \succ t$. !!! $\succ$ is total on ground terms. !!!

Proof Case

Case 1.2.1: F has the form $\underline{s \not\simeq t} \vee D$, $s \succ t$ and s is reducible.

Then R contains some rule $l \to r$ and F has the form $\underline{s[l] \not\simeq t} \vee D$.

By construction, S contains a clause $\underline{l \simeq r} \vee C$, such that l is greater than any term in r, C and $I_R \not\models C$.

Consider the superposition inference

$$\frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \not\simeq t} \vee D}{s[r] \not\simeq t \vee C \vee D}$$

Similar to the previous case we can prove that the conclusion $s[r] \not\simeq t \vee C \vee D$ is smaller than F and false.

Then use (*), as before.

Proof Case

Case 1.2.2: F has the form $\underline{s \not\simeq t} \vee D$, $s \succ t$ and s is irreducible.

Since F is false in I_R, we have $I_R \models s \simeq t$. Then s and t have the same normal form, so s must be reducible: contradiction.

Proof Case

Case 2: All selected literals in F are positive. Then F has a positive selected literal that is also a maximal literal in F.

Proof Case

Case 2: All selected literals in F are positive. Then F has a positive selected literal that is also a maximal literal in F.

!!! The literal selection function is well-behaving !!!

Proof Case

Case 2.1: F has a positive selected literal that is also a maximal literal in F and the maximal term in F is reducible.

Then F has the form $\underline{s \simeq t} \vee D$, $s \succ t$ and s is reducible. The proof is the same as in cases 1.2.1.

We have that R contains some rule $l \to r$ and F has the form $\underline{s[l] \simeq t} \vee D$.

By construction, S contains a clause $\underline{l \simeq r} \vee C$, such that l is greater than any term in r, C and $I_R \not\models C$.

Consider the superposition inference

$$\frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \simeq t} \vee D}{s[r] \simeq t \vee C \vee D}$$

Again, we can prove that the conclusion $s[r] \simeq t \vee C \vee D$ is smaller than F and false.

Then use (*), as before.

Proof Case

Case 2.2: F has a positive selected literal that is also a maximal literal in F and the maximal term in F is irreducible.

Case 2.2.1: F has a positive selected literal $l \simeq r'$ that is also a maximal literal in F, $l \succ r'$, l is irreducible, and there is exactly one occurrence of l in positive literals in F.

Then F has the form $\underline{l \simeq r'} \vee C$, l is greater than any term in r, C. By construction, R contains some rewrite rule $l \to r$, so l must be reducible. Contradiction.

Proof Case

Case 2.2.2: F has a positive selected literal $l \simeq r$ that is also a maximal literal in F, $l \succ r$, l is irreducible and there is more than one occurrence of l in positive literals in F.

Then F has the form $\underline{l \simeq r} \vee l \simeq r' \vee C$.
Consider the equality factoring inference

$$\frac{\underline{l \simeq r} \vee l \simeq r' \vee C}{l \simeq r \vee r \not\simeq r' \vee C} \text{ (EF)},$$

Again, we can prove that the conclusion $l \simeq r \vee r \not\simeq r' \vee C$ is smaller than F and false.

Then use (*), as before.

$\mathbb{S}\mathrm{up}_{\succ,\sigma}$ with Predicates

Superposition: (right and left)

$$\frac{\underline{l \simeq r} \lor C \quad \underline{A[l]} \lor D}{A[r] \lor C \lor D}\ (\mathrm{Sup}), \quad \frac{\underline{l \simeq r} \lor C \quad \underline{\neg A[l]} \lor D}{\neg A[r] \lor C \lor D}\ (\mathrm{Sup}),$$

where (i) $l \succ r$;

Binary Resolution:

$$\frac{\underline{A} \lor C \quad \underline{\neg A} \lor D}{C \lor D}\ (\mathrm{BR}),$$

Factoring:

$$\frac{\underline{A} \lor \underline{A} \lor C}{A \lor C}\ (\mathrm{Fact}),$$

$\mathbb{S}\mathrm{up}_{\succ,\sigma}$ with Predicates

Superposition: (right and left)

$$\frac{\underline{l \simeq r} \lor C \quad \underline{A[l]} \lor D}{A[r] \lor C \lor D}\ (\text{Sup}), \quad \frac{\underline{l \simeq r} \lor C \quad \underline{\neg A[l]} \lor D}{\neg A[r] \lor C \lor D}\ (\text{Sup}),$$

where (i) $l \succ r$;
(ii) $l \simeq r$ is the greatest literal in $l \simeq r \lor C$
(iii) $A[l]$ is the greatest literal in $A[l] \lor D$

Binary Resolution:

$$\frac{\underline{A} \lor C \quad \underline{\neg A} \lor D}{C \lor D}\ (\text{BR}),$$

where (i) A is maximal in $A \lor C$

Factoring:

$$\frac{\underline{A} \lor \underline{A} \lor C}{A \lor C}\ (\text{Fact}),$$

where (i) A is maximal in $A \lor A \lor C$

Arbitrary Predicates: Model Construction

First, build the congruence as before. We use induction on ground atoms instead of ground terms. We will again build an interpretation I_R step by step. Initially, all ground non-equality atoms are false in I_R. Then we will make some of them true.

Arbitrary Predicates: Model Construction

First, build the congruence as before. We use induction on ground atoms instead of ground terms. We will again build an interpretation I_R step by step. Initially, all ground non-equality atoms are false in I_R. Then we will make some of them true.

Take a ground atom $P(t_1, \ldots, t_n)$, where $t_1, \ldots, t_n$ are irreducible. We assume that for all atoms smaller than $P(t_1, \ldots, t_n)$, we have already defined whether they are true. We make $P(t_1, \ldots, t_n)$ true if

- S contains a clause $\underline{P(t_1, \ldots, t_n)} \vee C$ such that (i) $I_R \not\models C$ and (ii) $P(t_1, \ldots, t_n)$ is the greatest literal in $P(t_1, \ldots, t_n) \vee C$,

Saturation up to Redundancy and Satisfiability Checking

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\Box \in S$.

Saturation up to Redundancy and Satisfiability Checking

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Therefore, if we built a set saturated up to redundancy, then the initial set S_0 is satisfiable. This is a powerful way of checking redundancy: one can even check satisfiability of formulas having only infinite models.

Saturation up to Redundancy and Satisfiability Checking

Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

Therefore, if we built a set saturated up to redundancy, then the initial set S_0 is satisfiable. This is a powerful way of checking redundancy: one can even check satisfiability of formulas having only infinite models.

The only problem with this characterization is that there is no obvious way to build a model of S_0 from a saturated set in the non-ground case.