Completeness of Superposition

Andrei Voronkov

The University of Manchester
EasyChair

Bag Extension of an Ordering

Bag = finite multiset.
Let $>$ be any ordering on a set X. The bag extension of $>$ is a binary relation $>^{bag}$, on bags over X, defined as the smallest transitive relation on bags such that

$$\{x, y_1, \ldots, y_n\} >^{bag} \{x_1, \ldots, x_m, y_1, \ldots, y_n\}$$
$$\text{if } x > x_i \text{ for all } i \in \{1 \ldots m\},$$

where $m \geq 0$.
Idea: a bag becomes smaller if we replace an element by any finite number of smaller elements.
The following results are known about the bag extensions of orderings:

1. $>^{bag}$ is an ordering;
2. If $>$ is total, then so is $>^{bag}$;
3. If $>$ is well-founded, then so is $>^{bag}$.

Clause Orderings

From now on consider clauses also as bags of literals. Note:

- we have an ordering $\succ$ for comparing literals;
- a clause is a bag of literals.

Hence

- we can compare clauses using the bag extension $\succ^{bag}$ of $\succ$.

For simpicity we denote the multiset ordering also by $\succ$.

Redundancy

A clause $C \in S$ is called redundant in S if it is a logical consequence of clauses in S strictly smaller than C.

Inference Process with Redundancy

Let $\mathbb{I}$ be an inference system. Consider an inference process with two kinds of step $S_i \Rightarrow S_{i+1}$:

1. Adding the conclusion of an $\mathbb{I}$-inference with premises in S_i.
2. Deletion of a clause redundant in S_i, that is

$$S_{i+1} = S_i - \{C\},$$

where C is redundant in S_i.

Fairness: Persistent Clauses and Limit

Consider an inference process

$$S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \dots$$

A clause C is called persistent if

$$\exists i \forall j \geq i (C \in S_j).$$

The limit S_ω of the inference process is the set of all persistent clauses:

$$S_\omega = \bigcup_{i=0,1,\dots} \bigcap_{j \geq i} S_j.$$

Fairness

The process is called $\mathbb{I}$-fair if every inference with persistent premises in S_ω has been applied, that is, if

$$\frac{C_1 \quad \ldots \quad C_n}{C}$$

is an inference in $\mathbb{I}$ and $\{C_1, \ldots, C_n\} \subseteq S_\omega$, then $C \in S_i$ for some i.

Ground Superposition Inference System $\mathbb{S}\text{up}_{\succ,\sigma}$

Superposition: (right and left)

$$\frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \simeq t} \vee D}{s[r] \simeq t \vee C \vee D}\ (\text{Sup}), \quad \frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \not\simeq t} \vee D}{s[r] \not\simeq t \vee C \vee D}\ (\text{Sup}),$$

where (i) $l \succ r$;

(iii) $s[l] \succ t$;

Equality Resolution:

$$\frac{\underline{s \not\simeq s} \vee C}{C}\ (\text{ER}),$$

Equality Factoring:

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

where (i) $s \succ t$;
(ii) $s \simeq t$ is maximal in $s \simeq t \vee s \simeq t' \vee C$

Ground Superposition Inference System $\mathbb{S}\mathrm{up}_{\succ,\sigma}$

Superposition: (right and left)

$$\frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \simeq t} \vee D}{s[r] \simeq t \vee C \vee D}\ (\mathrm{Sup}), \quad \frac{\underline{l \simeq r} \vee C \quad \underline{s[l] \not\simeq t} \vee D}{s[r] \not\simeq t \vee C \vee D}\ (\mathrm{Sup}),$$

where (i) $l \succ r$;
(ii) $l \simeq r$ is the greatest literal in $l \simeq r \vee C$
(iii) $s[l] \succ t$;
(iv) $s[l] \simeq t$ is the greatest literal in $s[l] \simeq t \vee D$ (only for superposition-right)

Equality Resolution:

$$\frac{\underline{s \not\simeq s} \vee C}{C}\ (\mathrm{ER}),$$

Equality Factoring:

$$\frac{\underline{s \simeq t} \vee s \simeq t' \vee C}{s \simeq t \vee t \not\simeq t' \vee C}\ (\mathrm{EF}),$$

where (i) $s \succ t$;
(ii) $s \simeq t$ is maximal in $s \simeq t \vee s \simeq t' \vee C$

Completeness of $\mathbb{S}\mathrm{up}_{\succ,\sigma}$

Completeness Theorem. Let $\succ$ be a simplification ordering and σ a well-behaved selection function. Let also

1. S_0 be a set of clauses;
2. $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ be a fair $\mathbb{S}\mathrm{up}_{\succ,\sigma}$-inference process.

Then S_0 is unsatisfiable if and only if $\square \in S_i$ for some i.

Saturation up to Redundancy

A set S of clauses is called saturated up to redundancy if for every $\mathbb{I}$-inference

$$\frac{C_1 \quad \ldots \quad C_n}{C}$$

with premises in S, either

1. $C \in S$; or
2. C is redundant w.r.t. S, that is, $S_{\prec C} \models C$.

Proof of Completeness

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

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

Lemma. Every removed clause has a trace.
Lemma. The limit S_ω is saturated up to redundancy.
Lemma. The limit S_ω is logically equivalent to the initial set S_0.
Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

(Proof on the board)

Proof of Completeness

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

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

Lemma. Every removed clause has a trace.
Lemma. The limit S_ω is saturated up to redundancy.
Lemma. The limit S_ω is logically equivalent to the initial set S_0.
Lemma. A set S of clauses saturated up to redundancy is unsatisfiable if and only if $\square \in S$.

(Proof on the board)

Interestingly, only the last lemma uses rules of $\mathbb{S}\mathrm{up}_{\succ,\sigma}$.

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

$$t_0 \to_R t_1 \to_R t_2 \to_R \cdots$$

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

Congruence

- A congruence relation is a relation satisfying equality axioms (reflexivity, symmetry, transitivity and congruence);
- Any canonical 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 $\square \in S$.

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

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 canonical 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 canonical 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) $l \simeq r$ is the greatest literal in $l \simeq r \vee C$

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

Model Construction

We claim:

1. R is canonical;
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 canonical.

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 (this follows from the previous property)

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

Suppose that F has a negative selected literal. Then F has the form $\underline{s \not\simeq t} \vee D$.
Consider the case when 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 $D \vdash F$. The last property implies that $I_R \not\models F$.

If $D \in S$, that is, D is persistent, than we are done, since we have found a clause in S that is false in I_R and smaller than D.

If $D \notin S$, then it was removed and so has a trace $D_1, \ldots, D_n$. Since $D_1, \ldots, D_n \vdash D$ and D is false, then there exists some D_i such that D_i is false too. Note that we have $D_i \prec D \prec F$, so again we have found a clause smaller than F and false in I_R.

Proof Case

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

Consider the case when 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 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.

Again, to conclude we consider two cases: when $s[r] \not\simeq t \vee C \vee D$ is persistent and when it was removed. The proof is exactly as in the the previous case.

Proof Case

The next case is when 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 $I_R \models s = t$. But then by our construction s must be reducible because we would add some rule $s \to l$ at the step s.

Other Cases

Try yourself . . .

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

Superposition: (right and left)

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

where (i) $l \succ r$;

Binary Resolution:

$$\frac{\underline{A} \vee C \quad \underline{\neg A} \vee D}{C \vee D} \text{ (ER)},$$

Equality Factoring:

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

$\mathbb{S}\mathrm{up}_{\succ,\sigma}$ 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 $A[l] \lor D$ (only for superposition-right)

Binary Resolution:

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

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

Equality 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

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

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 $l \simeq r \vee C$,

Saturation up to Redundancy and Satisfiability Checking

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

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 characterisation is that there is no obvious way to build a model of S_0 out of a saturated set.