

# Formalisation of Ground Resolution and CDCL in Isabelle/HOL

Mathias Fleury and Jasmin Blanchette

April 22, 2016



# Contents

<b>1</b>	<b>Resolution-based techniques</b>	<b>5</b>
1.1	Resolution . . . . .	5
1.1.1	Simplification Rules . . . . .	5
1.1.2	Unconstrained Resolution . . . . .	6
1.1.3	Inference Rule . . . . .	7
1.1.4	Lemma about the simplified state . . . . .	12
1.1.5	Resolution and Invariants . . . . .	13
1.2	Superposition . . . . .	21
1.2.1	We can now define the rules of the calculus . . . . .	26
<b>theory</b>	<i>Prop-Resolution</i>	
<b>imports</b>	<i>Partial-Clausal-Logic List-More Wellfounded-More</i>	
<b>begin</b>		



# Chapter 1

## Resolution-based techniques

This chapter contains the formalisation of resolution and superposition.

### 1.1 Resolution

#### 1.1.1 Simplification Rules

**inductive** *simplify* :: 'v clauses  $\Rightarrow$  'v clauses  $\Rightarrow$  bool **for** *N* :: 'v clause set **where**

*tautology-deletion*:

$A + \{\#Pos\ P\# \} + \{\#Neg\ P\# \} \in N \implies simplify\ N\ (N - \{A + \{\#Pos\ P\# \} + \{\#Neg\ P\# \}\})$

*condensation*:

$A + \{\#L\# \} + \{\#L\# \} \in N \implies simplify\ N\ (N - \{A + \{\#L\# \} + \{\#L\# \}\} \cup \{A + \{\#L\# \}\})$

*subsumption*:

$A \in N \implies A \subset\# B \implies B \in N \implies simplify\ N\ (N - \{B\})$

**lemma** *simplify-preserves-un-sat'*:

**fixes** *N N'* :: 'v clauses

**assumes** *simplify N N'*

**and** *total-over-m I N*

**shows**  $I \models_s N' \longrightarrow I \models_s N$

*<proof>*

**lemma** *simplify-preserves-un-sat*:

**fixes** *N N'* :: 'v clauses

**assumes** *simplify N N'*

**and** *total-over-m I N*

**shows**  $I \models_s N \longrightarrow I \models_s N'$

*<proof>*

**lemma** *simplify-preserves-un-sat''*:

**fixes** *N N'* :: 'v clauses

**assumes** *simplify N N'*

**and** *total-over-m I N'*

**shows**  $I \models_s N \longrightarrow I \models_s N'$

*<proof>*

**lemma** *simplify-preserves-un-sat-eq*:

**fixes** *N N'* :: 'v clauses

**assumes** *simplify N N'*

**and** *total-over-m I N*

**shows**  $I \models_s N \longleftrightarrow I \models_s N'$

$\langle \text{proof} \rangle$

**lemma** *simplify-preserves-finite*:

**assumes** *simplify*  $\psi \ \psi'$

**shows** *finite*  $\psi \longleftrightarrow \text{finite } \psi'$

$\langle \text{proof} \rangle$

**lemma** *rtranclp-simplify-preserves-finite*:

**assumes** *rtranclp simplify*  $\psi \ \psi'$

**shows** *finite*  $\psi \longleftrightarrow \text{finite } \psi'$

$\langle \text{proof} \rangle$

**lemma** *simplify-atms-of-ms*:

**assumes** *simplify*  $\psi \ \psi'$

**shows** *atms-of-ms*  $\psi' \subseteq \text{atms-of-ms } \psi$

$\langle \text{proof} \rangle$

**lemma** *rtranclp-simplify-atms-of-ms*:

**assumes** *rtranclp simplify*  $\psi \ \psi'$

**shows** *atms-of-ms*  $\psi' \subseteq \text{atms-of-ms } \psi$

$\langle \text{proof} \rangle$

**lemma** *factoring-imp-simplify*:

**assumes**  $\{\#L\# \} + \{\#L\# \} + C \in N$

**shows**  $\exists N'. \text{ simplify } N \ N'$

$\langle \text{proof} \rangle$

### 1.1.2 Unconstrained Resolution

**type-synonym** *'v uncon-state* = *'v clauses*

**inductive** *uncon-res* :: *'v uncon-state*  $\Rightarrow$  *'v uncon-state*  $\Rightarrow$  *bool* **where**

*resolution*:

$\{\#Pos \ p\# \} + C \in N \implies \{\#Neg \ p\# \} + D \in N \implies (\{\#Pos \ p\# \} + C, \{\#Neg \ p\# \} + D) \notin$   
*already-used*

$\implies \text{uncon-res } (N) (N \cup \{C + D\}) \mid$

*factoring*:  $\{\#L\# \} + \{\#L\# \} + C \in N \implies \text{uncon-res } N (N \cup \{C + \{\#L\# \}\})$

**lemma** *uncon-res-increasing*:

**assumes** *uncon-res*  $S \ S'$  **and**  $\psi \in S$

**shows**  $\psi \in S'$

$\langle \text{proof} \rangle$

**lemma** *rtranclp-uncon-inference-increasing*:

**assumes** *rtranclp uncon-res*  $S \ S'$  **and**  $\psi \in S$

**shows**  $\psi \in S'$

$\langle \text{proof} \rangle$

### Subsumption

**definition** *subsumes* :: *'a literal multiset*  $\Rightarrow$  *'a literal multiset*  $\Rightarrow$  *bool* **where**

*subsumes*  $\chi \ \chi' \longleftrightarrow$

$(\forall I. \text{total-over-m } I \ \{\chi'\} \longrightarrow \text{total-over-m } I \ \{\chi\})$

$\wedge (\forall I. \text{total-over-m } I \ \{\chi\} \longrightarrow I \models \chi \longrightarrow I \models \chi')$

**lemma** *subsumes-refl[simp]*:

*subsumes*  $\chi \ \chi$

$\langle \text{proof} \rangle$

**lemma** *subsumes-subsumption*:

**assumes** *subsumes*  $D \chi$   
**and**  $C \subset\# D$  **and**  $\neg \text{tautology } \chi$   
**shows** *subsumes*  $C \chi$   $\langle \text{proof} \rangle$

**lemma** *subsumes-tautology*:

**assumes** *subsumes*  $(C + \{\# \text{Pos } P\# \} + \{\# \text{Neg } P\# \}) \chi$   
**shows** *tautology*  $\chi$   
 $\langle \text{proof} \rangle$

### 1.1.3 Inference Rule

**type-synonym** *'v state* = *'v clauses*  $\times$  (*'v clause*  $\times$  *'v clause*) *set*

**inductive** *inference-clause* :: *'v state*  $\Rightarrow$  *'v clause*  $\times$  (*'v clause*  $\times$  *'v clause*) *set*  $\Rightarrow$  *bool*

(**infix**  $\Rightarrow_{\text{Res}}$  100) **where**

*resolution*:

$\{\# \text{Pos } p\# \} + C \in N \implies \{\# \text{Neg } p\# \} + D \in N \implies (\{\# \text{Pos } p\# \} + C, \{\# \text{Neg } p\# \} + D) \notin$   
*already-used*

$\implies \text{inference-clause } (N, \text{already-used}) (C + D, \text{already-used} \cup \{(\{\# \text{Pos } p\# \} + C, \{\# \text{Neg } p\# \} + D)\}) \mid$

*factoring*:  $\{\# L\# \} + \{\# L\# \} + C \in N \implies \text{inference-clause } (N, \text{already-used}) (C + \{\# L\# \}, \text{already-used})$

**inductive** *inference* :: *'v state*  $\Rightarrow$  *'v state*  $\Rightarrow$  *bool* **where**

*inference-step*: *inference-clause*  $S$  (*clause*, *already-used*)

$\implies \text{inference } S (\text{fst } S \cup \{\text{clause}\}, \text{already-used})$

**abbreviation** *already-used-inv*

:: *'a literal multiset set*  $\times$  (*'a literal multiset*  $\times$  *'a literal multiset*) *set*  $\Rightarrow$  *bool* **where**

*already-used-inv state*  $\equiv$

$(\forall (A, B) \in \text{snd state}. \exists p. \text{Pos } p \in\# A \wedge \text{Neg } p \in\# B \wedge$   
 $((\exists \chi \in \text{fst state}. \text{subsumes } \chi ((A - \{\# \text{Pos } p\# \}) + (B - \{\# \text{Neg } p\# \})))$   
 $\vee \text{tautology } ((A - \{\# \text{Pos } p\# \}) + (B - \{\# \text{Neg } p\# \}))))$

**lemma** *inference-clause-preserves-already-used-inv*:

**assumes** *inference-clause*  $S S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $(\text{fst } S \cup \{\text{fst } S'\}, \text{snd } S')$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-already-used-inv*:

**assumes** *inference*  $S S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle \text{proof} \rangle$

**lemma** *rtranclp-inference-preserves-already-used-inv*:

**assumes** *rtranclp inference*  $S S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle \text{proof} \rangle$

**lemma** *subsumes-condensation*:

**assumes** *subsumes*  $(C + \{\#L\# \} + \{\#L\# \}) D$   
**shows** *subsumes*  $(C + \{\#L\# \}) D$   
 $\langle \text{proof} \rangle$

**lemma** *simplify-preserves-already-used-inv*:  
**assumes** *simplify*  $N N'$   
**and** *already-used-inv*  $(N, \text{already-used})$   
**shows** *already-used-inv*  $(N', \text{already-used})$   
 $\langle \text{proof} \rangle$

**lemma**  
*factoring-satisfiable*:  $I \models \{\#L\# \} + \{\#L\# \} + C \longleftrightarrow I \models \{\#L\# \} + C$  **and**  
*resolution-satisfiable*:  
*consistent-interp*  $I \implies I \models \{\#Pos\ p\# \} + C \implies I \models \{\#Neg\ p\# \} + D \implies I \models C + D$  **and**  
*factoring-same-vars*:  $\text{atms-of } (\{\#L\# \} + \{\#L\# \} + C) = \text{atms-of } (\{\#L\# \} + C)$   
 $\langle \text{proof} \rangle$

**lemma** *inference-increasing*:  
**assumes** *inference*  $S S'$  **and**  $\psi \in \text{fst } S$   
**shows**  $\psi \in \text{fst } S'$   
 $\langle \text{proof} \rangle$

**lemma** *rtranclp-inference-increasing*:  
**assumes** *rtranclp inference*  $S S'$  **and**  $\psi \in \text{fst } S$   
**shows**  $\psi \in \text{fst } S'$   
 $\langle \text{proof} \rangle$

**lemma** *inference-clause-already-used-increasing*:  
**assumes** *inference-clause*  $S S'$   
**shows**  $\text{snd } S \subseteq \text{snd } S'$   
 $\langle \text{proof} \rangle$

**lemma** *inference-already-used-increasing*:  
**assumes** *inference*  $S S'$   
**shows**  $\text{snd } S \subseteq \text{snd } S'$   
 $\langle \text{proof} \rangle$

**lemma** *inference-clause-preserves-un-sat*:  
**fixes**  $N N' :: 'v \text{ clauses}$   
**assumes** *inference-clause*  $T T'$   
**and** *total-over-m*  $I (\text{fst } T)$   
**and** *consistent*: *consistent-interp*  $I$   
**shows**  $I \models_s \text{fst } T \longleftrightarrow I \models_s \text{fst } T \cup \{\text{fst } T'\}$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-un-sat*:  
**fixes**  $N N' :: 'v \text{ clauses}$   
**assumes** *inference*  $T T'$   
**and** *total-over-m*  $I (\text{fst } T)$   
**and** *consistent*: *consistent-interp*  $I$   
**shows**  $I \models_s \text{fst } T \longleftrightarrow I \models_s \text{fst } T'$   
 $\langle \text{proof} \rangle$



**lemma** *inference-clause-preserves-atms-of-ms*:  
**assumes** *inference-clause*  $S\ S'$   
**shows**  $\text{atms-of-ms } (\text{fst } (\text{fst } S \cup \{\text{fst } S'\}, \text{snd } S')) \subseteq \text{atms-of-ms } (\text{fst } S)$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-atms-of-ms*:  
**fixes**  $N\ N' :: 'v \text{ clauses}$   
**assumes** *inference*  $T\ T'$   
**shows**  $\text{atms-of-ms } (\text{fst } T') \subseteq \text{atms-of-ms } (\text{fst } T)$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-total*:  
**fixes**  $N\ N' :: 'v \text{ clauses}$   
**assumes** *inference*  $(N, \text{already-used})\ (N', \text{already-used}')$   
**shows**  $\text{total-over-m } I\ N \implies \text{total-over-m } I\ N'$   
 $\langle \text{proof} \rangle$

**lemma** *rtranclp-inference-preserves-total*:  
**assumes** *rtranclp inference*  $T\ T'$   
**shows**  $\text{total-over-m } I\ (\text{fst } T) \implies \text{total-over-m } I\ (\text{fst } T')$   
 $\langle \text{proof} \rangle$

**lemma** *rtranclp-inference-preserves-un-sat*:  
**assumes** *rtranclp inference*  $N\ N'$   
**and**  $\text{total-over-m } I\ (\text{fst } N)$   
**and** *consistent: consistent-interp*  $I$   
**shows**  $I \models_s \text{fst } N \longleftrightarrow I \models_s \text{fst } N'$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-finite*:  
**assumes** *inference*  $\psi\ \psi'$  **and** *finite*  $(\text{fst } \psi)$   
**shows** *finite*  $(\text{fst } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *inference-clause-preserves-finite-snd*:  
**assumes** *inference-clause*  $\psi\ \psi'$  **and** *finite*  $(\text{snd } \psi)$   
**shows** *finite*  $(\text{snd } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *inference-preserves-finite-snd*:  
**assumes** *inference*  $\psi\ \psi'$  **and** *finite*  $(\text{snd } \psi)$   
**shows** *finite*  $(\text{snd } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *rtranclp-inference-preserves-finite*:  
**assumes** *rtranclp inference*  $\psi\ \psi'$  **and** *finite*  $(\text{fst } \psi)$   
**shows** *finite*  $(\text{fst } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *consistent-interp-insert*:  
**assumes** *consistent-interp*  $I$   
**and**  $\text{atm-of } P \notin \text{atm-of } I$

**shows** *consistent-interp* (insert *P I*)  
 ⟨*proof*⟩

**lemma** *simplify-clause-preserves-sat*:  
**assumes** *simp: simplify*  $\psi \ \psi'$   
**and** *satisfiable*  $\psi'$   
**shows** *satisfiable*  $\psi$   
 ⟨*proof*⟩

**lemma** *simplify-preserves-unsat*:  
**assumes** *inference*  $\psi \ \psi'$   
**shows** *satisfiable* (fst  $\psi'$ )  $\longrightarrow$  *satisfiable* (fst  $\psi$ )  
 ⟨*proof*⟩

**lemma** *inference-preserves-unsat*:  
**assumes** *inference\*\**  $S \ S'$   
**shows** *satisfiable* (fst  $S'$ )  $\longrightarrow$  *satisfiable* (fst  $S$ )  
 ⟨*proof*⟩

**datatype** *'v sem-tree* = *Node* *'v 'v sem-tree 'v sem-tree* | *Leaf*

**fun** *sem-tree-size* :: *'v sem-tree*  $\Rightarrow$  *nat* **where**  
*sem-tree-size* *Leaf* = 0 |  
*sem-tree-size* (*Node* - *ag ad*) = 1 + *sem-tree-size* *ag* + *sem-tree-size* *ad*

**lemma** *sem-tree-size*[*case-names bigger*]:  
 ( $\bigwedge xs:: 'v \text{ sem-tree. } (\bigwedge ys:: 'v \text{ sem-tree. } \text{sem-tree-size } ys < \text{sem-tree-size } xs \implies P \ ys) \implies P \ xs$ )  
 $\implies P \ xs$   
 ⟨*proof*⟩

**fun** *partial-interps* :: *'v sem-tree*  $\Rightarrow$  *'v interp*  $\Rightarrow$  *'v clauses*  $\Rightarrow$  *bool* **where**  
*partial-interps* *Leaf I*  $\psi$  = ( $\exists \chi. \neg I \models \chi \wedge \chi \in \psi \wedge \text{total-over-m } I \ \{\chi\}$ ) |  
*partial-interps* (*Node v ag ad*) *I*  $\psi \longleftrightarrow$   
 (*partial-interps* *ag* ( $I \cup \{\text{Pos } v\}$ )  $\psi \wedge$  *partial-interps* *ad* ( $I \cup \{\text{Neg } v\}$ )  $\psi$ )

**lemma** *simplify-preserve-partial-leaf*:  
*simplify*  $N \ N' \implies$  *partial-interps* *Leaf I N*  $\implies$  *partial-interps* *Leaf I N'*  
 ⟨*proof*⟩

**lemma** *simplify-preserve-partial-tree*:  
**assumes** *simplify*  $N \ N'$   
**and** *partial-interps*  $t \ I \ N$   
**shows** *partial-interps*  $t \ I \ N'$   
 ⟨*proof*⟩

**lemma** *inference-preserve-partial-tree*:  
**assumes** *inference*  $S \ S'$   
**and** *partial-interps*  $t \ I \ (\text{fst } S)$   
**shows** *partial-interps*  $t \ I \ (\text{fst } S')$   
 ⟨*proof*⟩

**lemma** *rtranclp-inference-preserve-partial-tree*:

**assumes** *rtranclp inference*  $N\ N'$   
**and** *partial-interps*  $t\ I\ (\text{fst } N)$   
**shows** *partial-interps*  $t\ I\ (\text{fst } N')$   
 $\langle \text{proof} \rangle$

**function** *build-sem-tree* ::  $'v :: \text{linorder set} \Rightarrow 'v \text{ clauses} \Rightarrow 'v \text{ sem-tree}$  **where**

*build-sem-tree*  $\text{atms } \psi =$   
 (if  $\text{atms} = \{\}$   $\vee \neg \text{finite atms}$   
 then *Leaf*  
 else *Node* (*Min atms*) (*build-sem-tree* (*Set.remove* (*Min atms*)  $\text{atms}$ )  $\psi$ )  
 (*build-sem-tree* (*Set.remove* (*Min atms*)  $\text{atms}$ )  $\psi$ ))

$\langle \text{proof} \rangle$

**termination**

$\langle \text{proof} \rangle$

**declare** *build-sem-tree.induct*[*case-names tree*]

**lemma** *unsatisfiable-empty[simp]*:

$\neg \text{unsatisfiable } \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *partial-interps-build-sem-tree-atms-general*:

**fixes**  $\psi :: 'v :: \text{linorder clauses}$  **and**  $p :: 'v \text{ literal list}$   
**assumes** *unsat: unsatisfiable*  $\psi$  **and** *finite*  $\psi$  **and** *consistent-interp*  $I$   
**and** *finite atms*  
**and**  $\text{atms-of-ms } \psi = \text{atms} \cup \text{atms-of-s } I$  **and**  $\text{atms} \cap \text{atms-of-s } I = \{\}$   
**shows** *partial-interps* (*build-sem-tree*  $\text{atms } \psi$ )  $I\ \psi$   
 $\langle \text{proof} \rangle$

**lemma** *partial-interps-build-sem-tree-atms*:

**fixes**  $\psi :: 'v :: \text{linorder clauses}$  **and**  $p :: 'v \text{ literal list}$   
**assumes** *unsat: unsatisfiable*  $\psi$  **and** *finite: finite*  $\psi$   
**shows** *partial-interps* (*build-sem-tree* ( $\text{atms-of-ms } \psi$ )  $\psi$ )  $\{\} \psi$   
 $\langle \text{proof} \rangle$

**lemma** *can-decrease-count*:

**fixes**  $\psi'' :: 'v \text{ clauses} \times ('v \text{ clause} \times 'v \text{ clause} \times 'v) \text{ set}$   
**assumes** *count*  $\chi\ L = n$   
**and**  $L \in \# \chi$  **and**  $\chi \in \text{fst } \psi$   
**shows**  $\exists \psi' \chi'. \text{inference}^{**} \psi\ \psi' \wedge \chi' \in \text{fst } \psi' \wedge (\forall L. L \in \# \chi \longleftrightarrow L \in \# \chi')$   
 $\wedge \text{count } \chi' L = 1$   
 $\wedge (\forall \varphi. \varphi \in \text{fst } \psi \longrightarrow \varphi \in \text{fst } \psi')$   
 $\wedge (I \models \chi \longleftrightarrow I \models \chi')$   
 $\wedge (\forall I'. \text{total-over-m } I' \{\chi\} \longrightarrow \text{total-over-m } I' \{\chi'\})$   
 $\langle \text{proof} \rangle$

**lemma** *can-decrease-tree-size*:

**fixes**  $\psi :: 'v \text{ state}$  **and**  $\text{tree} :: 'v \text{ sem-tree}$   
**assumes** *finite* ( $\text{fst } \psi$ ) **and** *already-used-inv*  $\psi$   
**and** *partial-interps*  $\text{tree } I\ (\text{fst } \psi)$   
**shows**  $\exists (\text{tree}' :: 'v \text{ sem-tree}) \psi'. \text{inference}^{**} \psi\ \psi' \wedge \text{partial-interps } \text{tree}'\ I\ (\text{fst } \psi')$   
 $\wedge (\text{sem-tree-size } \text{tree}' < \text{sem-tree-size } \text{tree} \vee \text{sem-tree-size } \text{tree} = 0)$   
 $\langle \text{proof} \rangle$

**lemma** *inference-completeness-inv*:  
**fixes**  $\psi :: 'v :: \text{linorder state}$   
**assumes**  
    $\text{unsat}: \neg \text{satisfiable } (\text{fst } \psi)$  **and**  
    $\text{finite}: \text{finite } (\text{fst } \psi)$  **and**  
    $\text{a-u-v}: \text{already-used-inv } \psi$   
**shows**  $\exists \psi'. (\text{inference}^{**} \psi \psi' \wedge \{\#\} \in \text{fst } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *inference-completeness*:  
**fixes**  $\psi :: 'v :: \text{linorder state}$   
**assumes**  $\text{unsat}: \neg \text{satisfiable } (\text{fst } \psi)$   
**and**  $\text{finite}: \text{finite } (\text{fst } \psi)$   
**and**  $\text{snd } \psi = \{\}$   
**shows**  $\exists \psi'. (\text{rtranclp inference } \psi \psi' \wedge \{\#\} \in \text{fst } \psi')$   
 $\langle \text{proof} \rangle$

**lemma** *inference-soundness*:  
**fixes**  $\psi :: 'v :: \text{linorder state}$   
**assumes**  $\text{rtranclp inference } \psi \psi' \text{ and } \{\#\} \in \text{fst } \psi'$   
**shows**  $\text{unsatisfiable } (\text{fst } \psi)$   
 $\langle \text{proof} \rangle$

**lemma** *inference-soundness-and-completeness*:  
**fixes**  $\psi :: 'v :: \text{linorder state}$   
**assumes**  $\text{finite}: \text{finite } (\text{fst } \psi)$   
**and**  $\text{snd } \psi = \{\}$   
**shows**  $(\exists \psi'. (\text{inference}^{**} \psi \psi' \wedge \{\#\} \in \text{fst } \psi')) \longleftrightarrow \text{unsatisfiable } (\text{fst } \psi)$   
 $\langle \text{proof} \rangle$

#### 1.1.4 Lemma about the simplified state

**abbreviation**  $\text{simplified } \psi \equiv (\text{no-step simplify } \psi)$

**lemma** *simplified-count*:  
**assumes**  $\text{simp}: \text{simplified } \psi$  **and**  $\chi: \chi \in \psi$   
**shows**  $\text{count } \chi L \leq 1$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-no-both*:  
**assumes**  $\text{simp}: \text{simplified } \psi$  **and**  $\chi: \chi \in \psi$   
**shows**  $\neg (L \in \# \chi \wedge \neg L \in \# \chi)$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-not-tautology*:  
**assumes**  $\text{simplified } \{\psi\}$   
**shows**  $\sim \text{tautology } \psi$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-remove*:  
**assumes**  $\text{simplified } \{\psi\}$   
**shows**  $\text{simplified } \{\psi - \{\#l\#\}\}$   
 $\langle \text{proof} \rangle$

**lemma** *in-simplified-simplified*:

**assumes** *simp*: *simplified*  $\psi$  **and** *incl*:  $\psi' \subseteq \psi$   
**shows** *simplified*  $\psi'$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-in*:  
**assumes** *simplified*  $\psi$   
**and**  $N \in \psi$   
**shows** *simplified*  $\{N\}$   
 $\langle \text{proof} \rangle$

**lemma** *subsumes-imp-formula*:  
**assumes**  $\psi \leq \# \varphi$   
**shows**  $\{\psi\} \models_p \varphi$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-imp-distinct-mset-tauto*:  
**assumes** *simp*: *simplified*  $\psi'$   
**shows** *distinct-mset-set*  $\psi'$  **and**  $\forall \chi \in \psi'. \neg \text{tautology } \chi$   
 $\langle \text{proof} \rangle$

**lemma** *simplified-no-more-full1-simplified*:  
**assumes** *simplified*  $\psi$   
**shows**  $\neg \text{full1 simplify } \psi \psi'$   
 $\langle \text{proof} \rangle$

### 1.1.5 Resolution and Invariants

**inductive** *resolution* :: '*v state*  $\Rightarrow$  '*v state*  $\Rightarrow$  *bool* **where**  
*full1-simp*: *full1 simplify*  $N N' \Rightarrow \text{resolution } (N, \text{already-used}) (N', \text{already-used})$  |  
*inferring*: *inference*  $(N, \text{already-used}) (N', \text{already-used}') \Rightarrow \text{simplified } N$   
 $\Rightarrow \text{full simplify } N' N'' \Rightarrow \text{resolution } (N, \text{already-used}) (N'', \text{already-used}')$

#### Invariants

**lemma** *resolution-finite*:  
**assumes** *resolution*  $\psi \psi'$  **and** *finite* (*fst*  $\psi$ )  
**shows** *finite* (*fst*  $\psi'$ )  
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-resolution-finite*:  
**assumes** *resolution*\*\*  $\psi \psi'$  **and** *finite* (*fst*  $\psi$ )  
**shows** *finite* (*fst*  $\psi'$ )  
 $\langle \text{proof} \rangle$

**lemma** *resolution-finite-snd*:  
**assumes** *resolution*  $\psi \psi'$  **and** *finite* (*snd*  $\psi$ )  
**shows** *finite* (*snd*  $\psi'$ )  
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-resolution-finite-snd*:  
**assumes** *resolution*\*\*  $\psi \psi'$  **and** *finite* (*snd*  $\psi$ )  
**shows** *finite* (*snd*  $\psi'$ )  
 $\langle \text{proof} \rangle$

**lemma** *resolution-always-simplified*:  
**assumes** *resolution*  $\psi \psi'$

**shows** *simplified* (fst  $\psi'$ )  
 <proof>

**lemma** *trancpl-resolution-always-simplified*:

**assumes** *trancpl resolution*  $\psi \psi'$   
**shows** *simplified* (fst  $\psi'$ )  
 <proof>

**lemma** *resolution-atms-of*:

**assumes** *resolution*  $\psi \psi'$  **and** *finite* (fst  $\psi$ )  
**shows** *atms-of-ms* (fst  $\psi'$ )  $\subseteq$  *atms-of-ms* (fst  $\psi$ )  
 <proof>

**lemma** *rtrancpl-resolution-atms-of*:

**assumes** *resolution*\*\*  $\psi \psi'$  **and** *finite* (fst  $\psi$ )  
**shows** *atms-of-ms* (fst  $\psi'$ )  $\subseteq$  *atms-of-ms* (fst  $\psi$ )  
 <proof>

**lemma** *resolution-include*:

**assumes** *res: resolution*  $\psi \psi'$  **and** *finite: finite* (fst  $\psi$ )  
**shows** *fst*  $\psi' \subseteq$  *simple-clss* (*atms-of-ms* (fst  $\psi$ ))  
 <proof>

**lemma** *rtrancpl-resolution-include*:

**assumes** *res: trancpl resolution*  $\psi \psi'$  **and** *finite: finite* (fst  $\psi$ )  
**shows** *fst*  $\psi' \subseteq$  *simple-clss* (*atms-of-ms* (fst  $\psi$ ))  
 <proof>

**abbreviation** *already-used-all-simple*

$:: ('a \text{ literal multiset} \times 'a \text{ literal multiset}) \text{ set} \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$  **where**  
*already-used-all-simple* *already-used* *vars*  $\equiv$   
 $(\forall (A, B) \in \text{already-used. simplified } \{A\} \wedge \text{simplified } \{B\} \wedge \text{atms-of } A \subseteq \text{vars} \wedge \text{atms-of } B \subseteq \text{vars})$

**lemma** *already-used-all-simple-vars-incl*:

**assumes** *vars*  $\subseteq$  *vars'*  
**shows** *already-used-all-simple* *a vars*  $\implies$  *already-used-all-simple* *a vars'*  
 <proof>

**lemma** *inference-clause-preserves-already-used-all-simple*:

**assumes** *inference-clause*  $S S'$   
**and** *already-used-all-simple* (snd  $S$ ) *vars*  
**and** *simplified* (fst  $S$ )  
**and** *atms-of-ms* (fst  $S$ )  $\subseteq$  *vars*  
**shows** *already-used-all-simple* (snd (fst  $S \cup \{\text{fst } S'\}, \text{snd } S'$ )) *vars*  
 <proof>

**lemma** *inference-preserves-already-used-all-simple*:

**assumes** *inference*  $S S'$   
**and** *already-used-all-simple* (snd  $S$ ) *vars*  
**and** *simplified* (fst  $S$ )  
**and** *atms-of-ms* (fst  $S$ )  $\subseteq$  *vars*  
**shows** *already-used-all-simple* (snd  $S'$ ) *vars*  
 <proof>

**lemma** *already-used-all-simple-inv*:

**assumes** *resolution*  $S S'$

**and** *already-used-all-simple* (*snd S*) *vars*  
**and** *atms-of-ms* (*fst S*)  $\subseteq$  *vars*  
**shows** *already-used-all-simple* (*snd S'*) *vars*  
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-already-used-all-simple-inv*:  
**assumes** *resolution\*\* S S'*  
**and** *already-used-all-simple* (*snd S*) *vars*  
**and** *atms-of-ms* (*fst S*)  $\subseteq$  *vars*  
**and** *finite* (*fst S*)  
**shows** *already-used-all-simple* (*snd S'*) *vars*  
 $\langle \text{proof} \rangle$

**lemma** *inference-clause-simplified-already-used-subset*:  
**assumes** *inference-clause S S'*  
**and** *simplified* (*fst S*)  
**shows** *snd S*  $\subset$  *snd S'*  
 $\langle \text{proof} \rangle$

**lemma** *inference-simplified-already-used-subset*:  
**assumes** *inference S S'*  
**and** *simplified* (*fst S*)  
**shows** *snd S*  $\subset$  *snd S'*  
 $\langle \text{proof} \rangle$

**lemma** *resolution-simplified-already-used-subset*:  
**assumes** *resolution S S'*  
**and** *simplified* (*fst S*)  
**shows** *snd S*  $\subset$  *snd S'*  
 $\langle \text{proof} \rangle$

**lemma** *trancp-resolution-simplified-already-used-subset*:  
**assumes** *trancp resolution S S'*  
**and** *simplified* (*fst S*)  
**shows** *snd S*  $\subset$  *snd S'*  
 $\langle \text{proof} \rangle$

**abbreviation** *already-used-top vars*  $\equiv$  *simple-clss vars*  $\times$  *simple-clss vars*

**lemma** *already-used-all-simple-in-already-used-top*:  
**assumes** *already-used-all-simple s vars* **and** *finite vars*  
**shows** *s*  $\subseteq$  *already-used-top vars*  
 $\langle \text{proof} \rangle$

**lemma** *already-used-top-finite*:  
**assumes** *finite vars*  
**shows** *finite* (*already-used-top vars*)  
 $\langle \text{proof} \rangle$

**lemma** *already-used-top-increasing*:  
**assumes** *var*  $\subseteq$  *var'* **and** *finite var'*  
**shows** *already-used-top var*  $\subseteq$  *already-used-top var'*  
 $\langle \text{proof} \rangle$

**lemma** *already-used-all-simple-finite*:  
**fixes** *s* :: ('a literal multiset  $\times$  'a literal multiset) *set* **and** *vars* :: 'a *set*

**assumes** *already-used-all-simple s vars* **and** *finite vars*  
**shows** *finite s*  
 $\langle \text{proof} \rangle$

**abbreviation** *card-simple vars  $\psi \equiv \text{card}(\text{already-used-top vars} - \psi)$*

**lemma** *resolution-card-simple-decreasing:*

**assumes** *res: resolution  $\psi \psi'$*   
**and** *a-u-s: already-used-all-simple (snd  $\psi$ ) vars*  
**and** *finite-v: finite vars*  
**and** *finite-fst: finite (fst  $\psi$ )*  
**and** *finite-snd: finite (snd  $\psi$ )*  
**and** *simp: simplified (fst  $\psi$ )*  
**and** *atms-of-ms (fst  $\psi$ )  $\subseteq$  vars*  
**shows** *card-simple vars (snd  $\psi'$ )  $<$  card-simple vars (snd  $\psi$ )*  
 $\langle \text{proof} \rangle$

**lemma** *trancp-resolution-card-simple-decreasing:*

**assumes** *trancp resolution  $\psi \psi'$  and finite-fst: finite (fst  $\psi$ )*  
**and** *already-used-all-simple (snd  $\psi$ ) vars*  
**and** *atms-of-ms (fst  $\psi$ )  $\subseteq$  vars*  
**and** *finite-v: finite vars*  
**and** *finite-snd: finite (snd  $\psi$ )*  
**and** *simplified (fst  $\psi$ )*  
**shows** *card-simple vars (snd  $\psi'$ )  $<$  card-simple vars (snd  $\psi$ )*  
 $\langle \text{proof} \rangle$

**lemma** *trancp-resolution-card-simple-decreasing-2:*

**assumes** *trancp resolution  $\psi \psi'$*   
**and** *finite-fst: finite (fst  $\psi$ )*  
**and** *empty-snd: snd  $\psi = \{\}$*   
**and** *simplified (fst  $\psi$ )*  
**shows** *card-simple (atms-of-ms (fst  $\psi$ )) (snd  $\psi'$ )  $<$  card-simple (atms-of-ms (fst  $\psi$ )) (snd  $\psi$ )*  
 $\langle \text{proof} \rangle$

## well-foundness if the relation

**lemma** *wf-simplified-resolution:*

**assumes** *f-vars: finite vars*  
**shows** *wf  $\{(y:: 'v:: \text{linorder state}, x). (\text{atms-of-ms (fst } x) \subseteq \text{vars} \wedge \text{simplified (fst } x) \wedge \text{finite (snd } x) \wedge \text{finite (fst } x) \wedge \text{already-used-all-simple (snd } x) \text{ vars}) \wedge \text{resolution } x y\}$*   
 $\langle \text{proof} \rangle$

**lemma** *wf-simplified-resolution':*

**assumes** *f-vars: finite vars*  
**shows** *wf  $\{(y:: 'v:: \text{linorder state}, x). (\text{atms-of-ms (fst } x) \subseteq \text{vars} \wedge \neg \text{simplified (fst } x) \wedge \text{finite (snd } x) \wedge \text{finite (fst } x) \wedge \text{already-used-all-simple (snd } x) \text{ vars}) \wedge \text{resolution } x y\}$*   
 $\langle \text{proof} \rangle$

**lemma** *wf-resolution:*

**assumes** *f-vars: finite vars*  
**shows** *wf  $(\{(y:: 'v:: \text{linorder state}, x). (\text{atms-of-ms (fst } x) \subseteq \text{vars} \wedge \text{simplified (fst } x) \wedge \text{finite (snd } x) \wedge \text{finite (fst } x) \wedge \text{already-used-all-simple (snd } x) \text{ vars}) \wedge \text{resolution } x y\} \cup \{(y, x). (\text{atms-of-ms (fst } x) \subseteq \text{vars} \wedge \neg \text{simplified (fst } x) \wedge \text{finite (snd } x) \wedge \text{finite (fst } x) \wedge \text{already-used-all-simple (snd } x) \text{ vars}) \wedge \text{resolution } x y\})$*



$\wedge \text{already-used-all-simple } (\text{snd } x) \text{ vars}) \wedge \text{resolution } x \ y\} \text{ (is wf } (?R \cup ?S))$   
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-simplify-already-used-inv*:  
**assumes** *simplify\*\**  $S \ S'$   
**and** *already-used-inv*  $(S, N)$   
**shows** *already-used-inv*  $(S', N)$   
 $\langle \text{proof} \rangle$

**lemma** *full1-simplify-already-used-inv*:  
**assumes** *full1 simplify*  $S \ S'$   
**and** *already-used-inv*  $(S, N)$   
**shows** *already-used-inv*  $(S', N)$   
 $\langle \text{proof} \rangle$

**lemma** *full-simplify-already-used-inv*:  
**assumes** *full simplify*  $S \ S'$   
**and** *already-used-inv*  $(S, N)$   
**shows** *already-used-inv*  $(S', N)$   
 $\langle \text{proof} \rangle$

**lemma** *resolution-already-used-inv*:  
**assumes** *resolution*  $S \ S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-resolution-already-used-inv*:  
**assumes** *resolution\*\**  $S \ S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle \text{proof} \rangle$

**lemma** *rtancp-simplify-preserves-unsat*:  
**assumes** *simplify\*\**  $\psi \ \psi'$   
**shows** *satisfiable*  $\psi' \longrightarrow \text{satisfiable } \psi$   
 $\langle \text{proof} \rangle$

**lemma** *full1-simplify-preserves-unsat*:  
**assumes** *full1 simplify*  $\psi \ \psi'$   
**shows** *satisfiable*  $\psi' \longrightarrow \text{satisfiable } \psi$   
 $\langle \text{proof} \rangle$

**lemma** *full-simplify-preserves-unsat*:  
**assumes** *full simplify*  $\psi \ \psi'$   
**shows** *satisfiable*  $\psi' \longrightarrow \text{satisfiable } \psi$   
 $\langle \text{proof} \rangle$

**lemma** *resolution-preserves-unsat*:  
**assumes** *resolution*  $\psi \ \psi'$   
**shows** *satisfiable*  $(\text{fst } \psi') \longrightarrow \text{satisfiable } (\text{fst } \psi)$   
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-resolution-preserves-unsat*:  
**assumes** *resolution\*\**  $\psi \ \psi'$   
**shows** *satisfiable*  $(\text{fst } \psi') \longrightarrow \text{satisfiable } (\text{fst } \psi)$   
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-simplify-preserve-partial-tree*:  
**assumes** *simplify\*\* N N'*  
**and** *partial-interps t I N*  
**shows** *partial-interps t I N'*  
 $\langle \text{proof} \rangle$

**lemma** *full1-simplify-preserve-partial-tree*:  
**assumes** *full1 simplify N N'*  
**and** *partial-interps t I N*  
**shows** *partial-interps t I N'*  
 $\langle \text{proof} \rangle$

**lemma** *full-simplify-preserve-partial-tree*:  
**assumes** *full simplify N N'*  
**and** *partial-interps t I N*  
**shows** *partial-interps t I N'*  
 $\langle \text{proof} \rangle$

**lemma** *resolution-preserve-partial-tree*:  
**assumes** *resolution S S'*  
**and** *partial-interps t I (fst S)*  
**shows** *partial-interps t I (fst S')*  
 $\langle \text{proof} \rangle$

**lemma** *rtrancp-resolution-preserve-partial-tree*:  
**assumes** *resolution\*\* S S'*  
**and** *partial-interps t I (fst S)*  
**shows** *partial-interps t I (fst S')*  
 $\langle \text{proof} \rangle$   
**thm** *nat-less-induct nat.induct*

**lemma** *nat-ge-induct[case-names 0 Suc]*:  
**assumes** *P 0*  
**and**  $\bigwedge n. (\bigwedge m. m < \text{Suc } n \implies P m) \implies P (\text{Suc } n)$   
**shows** *P n*  
 $\langle \text{proof} \rangle$

**lemma** *wf-always-more-step-False*:  
**assumes** *wf R*  
**shows**  $(\forall x. \exists z. (z, x) \in R) \implies \text{False}$   
 $\langle \text{proof} \rangle$

**lemma** *finite-finite-mset-element-of-mset[simp]*:  
**assumes** *finite N*  
**shows** *finite  $\{f \varphi L \mid \varphi L. \varphi \in N \wedge L \in \# \varphi \wedge P \varphi L\}$*   
 $\langle \text{proof} \rangle$

**definition** *sum-count-ge-2* :: *'a multiset set  $\Rightarrow$  nat ( $\Xi$ ) where*  
*sum-count-ge-2  $\equiv$  folding.F ( $\lambda \varphi. \text{op} + (\text{msetsum } \{\# \text{count } \varphi L \mid L \in \# \varphi. 2 \leq \text{count } \varphi L\}\})) 0$*

**interpretation** *sum-count-ge-2*:  
*folding ( $\lambda \varphi. \text{op} + (\text{msetsum } \{\# \text{count } \varphi L \mid L \in \# \varphi. 2 \leq \text{count } \varphi L\}\})) 0$*   
**rewrites**

*folding.F* ( $\lambda\varphi. op + (msetsum \{ \#count \varphi L \mid L \in \# \varphi. 2 \leq count \varphi L \# \})$ )  $0 = sum-count-ge-2$   
 <proof>

**lemma** *finite-incl-le-setsum*:

*finite* ( $B :: 'a \text{ multiset set}$ )  $\implies A \subseteq B \implies \Xi A \leq \Xi B$   
 <proof>

**lemma** *simplify-finite-measure-decrease*:

*simplify*  $N N' \implies \text{finite } N \implies card N' + \Xi N' < card N + \Xi N$   
 <proof>

**lemma** *simplify-terminates*:

*wf*  $\{(N', N). \text{finite } N \wedge \text{simplify } N N'\}$   
 <proof>

**lemma** *wf-terminates*:

**assumes** *wf*  $r$   
**shows**  $\exists N'. (N', N) \in r^* \wedge (\forall N''. (N'', N') \notin r)$   
 <proof>

**lemma** *rtranclp-simplify-terminates*:

**assumes** *fin*: *finite*  $N$   
**shows**  $\exists N'. \text{simplify}^{**} N N' \wedge \text{simplified } N'$   
 <proof>

**lemma** *finite-simplified-full1-simp*:

**assumes** *finite*  $N$   
**shows**  $\text{simplified } N \vee (\exists N'. \text{full1 simplify } N N')$   
 <proof>

**lemma** *finite-simplified-full-simp*:

**assumes** *finite*  $N$   
**shows**  $\exists N'. \text{full simplify } N N'$   
 <proof>

**lemma** *can-decrease-tree-size-resolution*:

**fixes**  $\psi :: 'v \text{ state}$  **and**  $\text{tree} :: 'v \text{ sem-tree}$   
**assumes** *finite* (*fst*  $\psi$ ) **and** *already-used-inv*  $\psi$   
**and** *partial-interps*  $\text{tree } I$  (*fst*  $\psi$ )  
**and** *simplified* (*fst*  $\psi$ )  
**shows**  $\exists (\text{tree}' :: 'v \text{ sem-tree}) \psi'. \text{resolution}^{**} \psi \psi' \wedge \text{partial-interps } \text{tree}' I$  (*fst*  $\psi'$ )  
 $\wedge (\text{sem-tree-size } \text{tree}' < \text{sem-tree-size } \text{tree} \vee \text{sem-tree-size } \text{tree} = 0)$   
 <proof>

**lemma** *resolution-completeness-inv*:

**fixes**  $\psi :: 'v :: \text{linorder state}$   
**assumes**  
*unsat*:  $\neg \text{satisfiable}$  (*fst*  $\psi$ ) **and**  
*finite*: *finite* (*fst*  $\psi$ ) **and**  
*a-u-v*: *already-used-inv*  $\psi$   
**shows**  $\exists \psi'. (\text{resolution}^{**} \psi \psi' \wedge \{\#\} \in \text{fst } \psi')$   
 <proof>

**lemma** *resolution-preserves-already-used-inv*:

**assumes** *resolution*  $S\ S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle proof \rangle$

**lemma** *rtrancp-resolution-preserves-already-used-inv*:  
**assumes** *resolution*\*\*  $S\ S'$   
**and** *already-used-inv*  $S$   
**shows** *already-used-inv*  $S'$   
 $\langle proof \rangle$

**lemma** *resolution-completeness*:  
**fixes**  $\psi :: 'v :: linorder\ state$   
**assumes** *unsat*:  $\neg \text{satisfiable}\ (fst\ \psi)$   
**and** *finite*: *finite*  $(fst\ \psi)$   
**and** *snd*  $\psi = \{\}$   
**shows**  $\exists \psi'. (\text{resolution}^{**}\ \psi\ \psi' \wedge \{\#\} \in fst\ \psi')$   
 $\langle proof \rangle$

**lemma** *rtrancp-preserves-sat*:  
**assumes** *simplify*\*\*  $S\ S'$   
**and** *satisfiable*  $S$   
**shows** *satisfiable*  $S'$   
 $\langle proof \rangle$

**lemma** *resolution-preserves-sat*:  
**assumes** *resolution*  $S\ S'$   
**and** *satisfiable*  $(fst\ S)$   
**shows** *satisfiable*  $(fst\ S')$   
 $\langle proof \rangle$

**lemma** *rtrancp-resolution-preserves-sat*:  
**assumes** *resolution*\*\*  $S\ S'$   
**and** *satisfiable*  $(fst\ S)$   
**shows** *satisfiable*  $(fst\ S')$   
 $\langle proof \rangle$

**lemma** *resolution-soundness*:  
**fixes**  $\psi :: 'v :: linorder\ state$   
**assumes** *resolution*\*\*  $\psi\ \psi'$  **and**  $\{\#\} \in fst\ \psi'$   
**shows** *unsatisfiable*  $(fst\ \psi)$   
 $\langle proof \rangle$

**lemma** *resolution-soundness-and-completeness*:  
**fixes**  $\psi :: 'v :: linorder\ state$   
**assumes** *finite*: *finite*  $(fst\ \psi)$   
**and** *snd*: *snd*  $\psi = \{\}$   
**shows**  $(\exists \psi'. (\text{resolution}^{**}\ \psi\ \psi' \wedge \{\#\} \in fst\ \psi')) \longleftrightarrow \text{unsatisfiable}\ (fst\ \psi)$   
 $\langle proof \rangle$

**lemma** *simplified-falsity*:  
**assumes** *simp*: *simplified*  $\psi$   
**and**  $\{\#\} \in \psi$   
**shows**  $\psi = \{\{\#\}\}$   
 $\langle proof \rangle$

```

lemma simplify-falsity-in-preserved:
  assumes simplify  $\chi s \ \chi s'$ 
  and  $\{\#\} \in \chi s$ 
  shows  $\{\#\} \in \chi s'$ 
   $\langle proof \rangle$ 

lemma rtrancp-simplify-falsity-in-preserved:
  assumes simplify**  $\chi s \ \chi s'$ 
  and  $\{\#\} \in \chi s$ 
  shows  $\{\#\} \in \chi s'$ 
   $\langle proof \rangle$ 

lemma resolution-falsity-get-falsity-alone:
  assumes finite (fst  $\psi$ )
  shows  $(\exists \psi'. (resolution^{**} \ \psi \ \psi' \wedge \{\#\} \in fst \ \psi')) \longleftrightarrow (\exists a-u-v. resolution^{**} \ \psi \ (\{\{\#\}\}, a-u-v))$ 
  (is  $?A \longleftrightarrow ?B$ )
   $\langle proof \rangle$ 

lemma resolution-soundness-and-completeness':
  fixes  $\psi :: 'v :: linorder \ state$ 
  assumes
    finite: finite (fst  $\psi$ ) and
    snd: snd  $\psi = \{\}$ 
  shows  $(\exists a-u-v. (resolution^{**} \ \psi \ (\{\{\#\}\}, a-u-v))) \longleftrightarrow unsatisfiable \ (fst \ \psi)$ 
   $\langle proof \rangle$ 

end
theory Prop-Superposition
imports Partial-Clausal-Logic ../lib/Herbrand-Interpretation
begin

```

## 1.2 Superposition

```

no-notation Herbrand-Interpretation.true-cls (infix  $\models$  50)
notation Herbrand-Interpretation.true-cls (infix  $\models_h$  50)

no-notation Herbrand-Interpretation.true-clss (infix  $\models_s$  50)
notation Herbrand-Interpretation.true-clss (infix  $\models_{hs}$  50)

lemma herbrand-interp-iff-partial-interp-cls:
   $S \models_h C \longleftrightarrow \{Pos \ P|P. P \in S\} \cup \{Neg \ P|P. P \notin S\} \models C$ 
   $\langle proof \rangle$ 

lemma herbrand-consistent-interp:
  consistent-interp  $(\{Pos \ P|P. P \in S\} \cup \{Neg \ P|P. P \notin S\})$ 
   $\langle proof \rangle$ 

lemma herbrand-total-over-set:
  total-over-set  $(\{Pos \ P|P. P \in S\} \cup \{Neg \ P|P. P \notin S\}) \ T$ 
   $\langle proof \rangle$ 

lemma herbrand-total-over-m:
  total-over-m  $(\{Pos \ P|P. P \in S\} \cup \{Neg \ P|P. P \notin S\}) \ T$ 
   $\langle proof \rangle$ 

```

**lemma** *herbrand-interp-iff-partial-interp-clss*:

$S \models_{hs} C \longleftrightarrow \{Pos\ P | P. P \in S\} \cup \{Neg\ P | P. P \notin S\} \models_s C$   
 $\langle proof \rangle$

**definition** *clss-lt* :: 'a::wellorder clauses  $\Rightarrow$  'a clause  $\Rightarrow$  'a clauses **where**

*clss-lt*  $N\ C = \{D \in N. D \# \subset \# C\}$

**notation** (*latex output*)

*clss-lt* ( $-\hat{<}^{bsup} - \hat{<}^{esup}$ )

**locale** *selection* =

**fixes**  $S :: 'a\ clause \Rightarrow 'a\ clause$

**assumes**

$S\text{-selects-subseteq}: \bigwedge C. S\ C \leq \# C$  **and**

$S\text{-selects-neg-lits}: \bigwedge C\ L. L \in \# S\ C \implies is\text{-neg}\ L$

**locale** *ground-resolution-with-selection* =

*selection*  $S$  **for**  $S :: ('a :: wellorder)\ clause \Rightarrow 'a\ clause$

**begin**

**context**

**fixes**  $N :: 'a\ clause\ set$

**begin**

We do not create an equivalent of  $\delta$ , but we directly defined  $N_C$  by inlining the definition.

**function**

*production* :: 'a clause  $\Rightarrow$  'a interp

**where**

*production*  $C =$

$\{A. C \in N \wedge C \neq \{\#\} \wedge \text{Max}(\text{set-mset}\ C) = \text{Pos}\ A \wedge \text{count}\ C\ (\text{Pos}\ A) \leq 1$   
 $\wedge \neg (\bigcup D \in \{D. D \# \subset \# C\}. \text{production}\ D) \models_h C \wedge S\ C = \{\#\}\}$

$\langle proof \rangle$

**termination**  $\langle proof \rangle$

**declare** *production.simps*[*simp del*]

**definition** *interp* :: 'a clause  $\Rightarrow$  'a interp **where**

*interp*  $C = (\bigcup D \in \{D. D \# \subset \# C\}. \text{production}\ D)$

**lemma** *production-unfold*:

*production*  $C = \{A. C \in N \wedge C \neq \{\#\} \wedge \text{Max}(\text{set-mset}\ C) = \text{Pos}\ A \wedge \text{count}\ C\ (\text{Pos}\ A) \leq 1 \wedge \neg$   
 $\text{interp}\ C \models_h C \wedge S\ C = \{\#\}\}$

$\langle proof \rangle$

**abbreviation** *productive*  $A \equiv (\text{production}\ A \neq \{\})$

**abbreviation** *produces* :: 'a clause  $\Rightarrow$  'a  $\Rightarrow$  bool **where**

*produces*  $C\ A \equiv \text{production}\ C = \{A\}$

**lemma** *producesD*:

*produces*  $C\ A \implies C \in N \wedge C \neq \{\#\} \wedge \text{Pos}\ A = \text{Max}(\text{set-mset}\ C) \wedge \text{count}\ C\ (\text{Pos}\ A) \leq 1 \wedge \neg$   
 $\text{interp}\ C \models_h C \wedge S\ C = \{\#\}$

$\langle proof \rangle$

**lemma** *produces*  $C\ A \implies \text{Pos}\ A \in \# C$

$\langle \text{proof} \rangle$

**lemma** *interp'-def-in-set:*

$\text{interp } C = (\bigcup D \in \{D \in N. D \# \subset \# C\}. \text{production } D)$

$\langle \text{proof} \rangle$

**lemma** *production-iff-produces:*

$\text{produces } D A \longleftrightarrow A \in \text{production } D$

$\langle \text{proof} \rangle$

**definition** *Interp :: 'a clause  $\Rightarrow$  'a interp where*

$\text{Interp } C = \text{interp } C \cup \text{production } C$

**lemma**

**assumes** *produces C P*

**shows** *Interp C  $\models_h$  C*

$\langle \text{proof} \rangle$

**definition** *INTERP :: 'a interp where*

$\text{INTERP} = (\bigcup D \in N. \text{production } D)$

**lemma** *interp-subseteq-Interp[simp]:*  $\text{interp } C \subseteq \text{Interp } C$

$\langle \text{proof} \rangle$

**lemma** *Interp-as-UNION:*  $\text{Interp } C = (\bigcup D \in \{D. D \# \subseteq \# C\}. \text{production } D)$

$\langle \text{proof} \rangle$

**lemma** *productive-not-empty:*  $\text{productive } C \Longrightarrow C \neq \{\#\}$

$\langle \text{proof} \rangle$

**lemma** *productive-imp-produces-Max-literal:*  $\text{productive } C \Longrightarrow \text{produces } C (\text{atm-of } (\text{Max } (\text{set-mset } C)))$

$\langle \text{proof} \rangle$

**lemma** *productive-imp-produces-Max-atom:*  $\text{productive } C \Longrightarrow \text{produces } C (\text{Max } (\text{atms-of } C))$

$\langle \text{proof} \rangle$

**lemma** *produces-imp-Max-literal:*  $\text{produces } C A \Longrightarrow A = \text{atm-of } (\text{Max } (\text{set-mset } C))$

$\langle \text{proof} \rangle$

**lemma** *produces-imp-Max-atom:*  $\text{produces } C A \Longrightarrow A = \text{Max } (\text{atms-of } C)$

$\langle \text{proof} \rangle$

**lemma** *produces-imp-Pos-in-lits:*  $\text{produces } C A \Longrightarrow \text{Pos } A \in \# C$

$\langle \text{proof} \rangle$

**lemma** *productive-in-N:*  $\text{productive } C \Longrightarrow C \in N$

$\langle \text{proof} \rangle$

**lemma** *produces-imp-atms-leq:*  $\text{produces } C A \Longrightarrow B \in \text{atms-of } C \Longrightarrow B \leq A$

$\langle \text{proof} \rangle$

**lemma** *produces-imp-neg-notin-lits:*  $\text{produces } C A \Longrightarrow \text{Neg } A \notin \# C$

$\langle \text{proof} \rangle$

**lemma** *less-eq-imp-interp-subseteq-interp:*  $C \# \subseteq \# D \Longrightarrow \text{interp } C \subseteq \text{interp } D$

$\langle \text{proof} \rangle$

**lemma** *less-eq-imp-interp-subseteq-Interp*:  $C \# \subseteq \# D \implies \text{interp } C \subseteq \text{Interp } D$

$\langle \text{proof} \rangle$

**lemma** *less-imp-production-subseteq-interp*:  $C \# \subset \# D \implies \text{production } C \subseteq \text{interp } D$

$\langle \text{proof} \rangle$

**lemma** *less-eq-imp-production-subseteq-Interp*:  $C \# \subseteq \# D \implies \text{production } C \subseteq \text{Interp } D$

$\langle \text{proof} \rangle$

**lemma** *less-imp-Interp-subseteq-interp*:  $C \# \subset \# D \implies \text{Interp } C \subseteq \text{interp } D$

$\langle \text{proof} \rangle$

**lemma** *less-eq-imp-Interp-subseteq-Interp*:  $C \# \subseteq \# D \implies \text{Interp } C \subseteq \text{Interp } D$

$\langle \text{proof} \rangle$

**lemma** *false-Interp-to-true-interp-imp-less-multiset*:  $A \notin \text{Interp } C \implies A \in \text{interp } D \implies C \# \subset \# D$

$\langle \text{proof} \rangle$

**lemma** *false-interp-to-true-interp-imp-less-multiset*:  $A \notin \text{interp } C \implies A \in \text{interp } D \implies C \# \subset \# D$

$\langle \text{proof} \rangle$

**lemma** *false-Interp-to-true-Interp-imp-less-multiset*:  $A \notin \text{Interp } C \implies A \in \text{Interp } D \implies C \# \subset \# D$

$\langle \text{proof} \rangle$

**lemma** *false-interp-to-true-Interp-imp-le-multiset*:  $A \notin \text{interp } C \implies A \in \text{Interp } D \implies C \# \subseteq \# D$

$\langle \text{proof} \rangle$

**lemma** *interp-subseteq-INTERP*:  $\text{interp } C \subseteq \text{INTERP}$

$\langle \text{proof} \rangle$

**lemma** *production-subseteq-INTERP*:  $\text{production } C \subseteq \text{INTERP}$

$\langle \text{proof} \rangle$

**lemma** *Interp-subseteq-INTERP*:  $\text{Interp } C \subseteq \text{INTERP}$

$\langle \text{proof} \rangle$

This lemma corresponds to theorem 2.7.6 page 67 of Weidenbach's book.

**lemma** *produces-imp-in-interp*:

**assumes** *a-in-c*:  $\text{Neg } A \in \# C$  **and** *d: produces D A*

**shows**  $A \in \text{interp } C$

$\langle \text{proof} \rangle$

**lemma** *neg-notin-Interp-not-produce*:  $\text{Neg } A \in \# C \implies A \notin \text{Interp } D \implies C \# \subseteq \# D \implies \neg \text{produces } D'' A$

$\langle \text{proof} \rangle$

**lemma** *in-production-imp-produces*:  $A \in \text{production } C \implies \text{produces } C A$

$\langle \text{proof} \rangle$

**lemma** *not-produces-imp-notin-production*:  $\neg \text{produces } C A \implies A \notin \text{production } C$

$\langle \text{proof} \rangle$

**lemma** *not-produces-imp-notin-interp*:  $(\bigwedge D. \neg \text{produces } D A) \implies A \notin \text{interp } C$

$\langle \text{proof} \rangle$



The results below corresponds to Lemma 3.4.

**Nitpicking:** If  $D = D'$  and  $D$  is productive,  $I^D \subseteq I_{D'}$  does not hold.

**lemma** *true-Interp-imp-general:*

**assumes**

*c-le-d:*  $C \# \subseteq \# D$  **and**

*d-lt-d':*  $D \# \subset \# D'$  **and**

*c-at-d:*  $\text{Interp } D \models_h C$  **and**

*subs:*  $\text{interp } D' \subseteq (\bigcup C \in CC. \text{production } C)$

**shows**  $(\bigcup C \in CC. \text{production } C) \models_h C$

*<proof>*

**lemma** *true-Interp-imp-interp:*  $C \# \subseteq \# D \implies D \# \subset \# D' \implies \text{Interp } D \models_h C \implies \text{interp } D' \models_h C$

*<proof>*

**lemma** *true-Interp-imp-Interp:*  $C \# \subseteq \# D \implies D \# \subset \# D' \implies \text{Interp } D \models_h C \implies \text{Interp } D' \models_h C$

*<proof>*

**lemma** *true-Interp-imp-INTERP:*  $C \# \subseteq \# D \implies \text{Interp } D \models_h C \implies \text{INTERP} \models_h C$

*<proof>*

**lemma** *true-interp-imp-general:*

**assumes**

*c-le-d:*  $C \# \subseteq \# D$  **and**

*d-lt-d':*  $D \# \subset \# D'$  **and**

*c-at-d:*  $\text{interp } D \models_h C$  **and**

*subs:*  $\text{interp } D' \subseteq (\bigcup C \in CC. \text{production } C)$

**shows**  $(\bigcup C \in CC. \text{production } C) \models_h C$

*<proof>*

This lemma corresponds to theorem 2.7.6 page 67 of Weidenbach's book. Here the strict maximality is important

**lemma** *true-interp-imp-interp:*  $C \# \subseteq \# D \implies D \# \subset \# D' \implies \text{interp } D \models_h C \implies \text{interp } D' \models_h C$

*<proof>*

**lemma** *true-interp-imp-Interp:*  $C \# \subseteq \# D \implies D \# \subset \# D' \implies \text{interp } D \models_h C \implies \text{Interp } D' \models_h C$

*<proof>*

**lemma** *true-interp-imp-INTERP:*  $C \# \subseteq \# D \implies \text{interp } D \models_h C \implies \text{INTERP} \models_h C$

*<proof>*

**lemma** *productive-imp-false-interp:*  $\text{productive } C \implies \neg \text{interp } C \models_h C$

*<proof>*

This lemma corresponds to theorem 2.7.6 page 67 of Weidenbach's book. Here the strict maximality is important

**lemma** *cls-gt-double-pos-no-production:*

**assumes**  $D: \{\#Pos P, Pos P\} \# \subset \# C$

**shows**  $\neg \text{produces } C P$

*<proof>*

This lemma corresponds to theorem 2.7.6 page 67 of Weidenbach's book.

**lemma**

**assumes**  $D: C + \{\#Neg P\} \# \subset \# D$

**shows**  $\text{production } D \neq \{P\}$

$\langle proof \rangle$

**lemma** *in-interp-is-produced*:

**assumes**  $P \in INTERP$

**shows**  $\exists D. D + \{\#Pos\ P\# \} \in N \wedge produces\ (D + \{\#Pos\ P\# \})\ P$

$\langle proof \rangle$

**end**

**end**

**abbreviation**  $MMax\ M \equiv Max\ (set-mset\ M)$

### 1.2.1 We can now define the rules of the calculus

**inductive** *superposition-rules* :: *'a clause*  $\Rightarrow$  *'a clause*  $\Rightarrow$  *'a clause*  $\Rightarrow$  *bool* **where**

*factoring*: *superposition-rules*  $(C + \{\#Pos\ P\# \} + \{\#Pos\ P\# \})\ B\ (C + \{\#Pos\ P\# \})\ |$

*superposition-l*: *superposition-rules*  $(C_1 + \{\#Pos\ P\# \})\ (C_2 + \{\#Neg\ P\# \})\ (C_1 + C_2)$

**inductive** *superposition* :: *'a clauses*  $\Rightarrow$  *'a clauses*  $\Rightarrow$  *bool* **where**

*superposition*:  $A \in N \Longrightarrow B \in N \Longrightarrow superposition-rules\ A\ B\ C$

$\Longrightarrow superposition\ N\ (N \cup \{C\})$

**definition** *abstract-red* :: *'a::wellorder clause*  $\Rightarrow$  *'a clauses*  $\Rightarrow$  *bool* **where**

*abstract-red*  $C\ N = (clss-lt\ N\ C \models_p C)$

**lemma** *less-multiset[iff]*:  $M < N \longleftrightarrow M \# \subset \# N$

$\langle proof \rangle$

**lemma** *less-eq-multiset[iff]*:  $M \leq N \longleftrightarrow M \# \subseteq \# N$

$\langle proof \rangle$

**lemma** *herbrand-true-clss-true-clss-clss-herbrand-true-clss*:

**assumes**

$AB: A \models_{hs} B$  **and**

$BC: B \models_p C$

**shows**  $A \models_h C$

$\langle proof \rangle$

**lemma** *abstract-red-subset-mset-abstract-red*:

**assumes**

*abstr*: *abstract-red*  $C\ N$  **and**

*c-lt-d*:  $C \subseteq \# D$

**shows** *abstract-red*  $D\ N$

$\langle proof \rangle$

**lemma** *true-clss-clss-extended*:

**assumes**

$A \models_p B$  **and**

*tot*: *total-over-m*  $I\ A$  **and**

*cons*: *consistent-interp*  $I$  **and**

$I-A: I \models_s A$

**shows**  $I \models B$

$\langle proof \rangle$

**lemma**

**assumes**  
 $CP: \neg \text{class-}lt\ N (\{ \#C\# \} + \{ \#E\# \}) \models_p \{ \#C\# \} + \{ \#Neg\ P\# \}$  **and**  
 $\text{class-}lt\ N (\{ \#C\# \} + \{ \#E\# \}) \models_p \{ \#E\# \} + \{ \#Pos\ P\# \} \vee \text{class-}lt\ N (\{ \#C\# \} + \{ \#E\# \}) \models_p$   
 $\{ \#C\# \} + \{ \#Neg\ P\# \}$   
**shows**  $\text{class-}lt\ N (\{ \#C\# \} + \{ \#E\# \}) \models_p \{ \#E\# \} + \{ \#Pos\ P\# \}$

$\langle proof \rangle$

**locale** *ground-ordered-resolution-with-redundancy* =  
*ground-resolution-with-selection* +  
**fixes** *redundant* :: 'a::wellorder clause  $\Rightarrow$  'a clauses  $\Rightarrow$  bool  
**assumes**  
*redundant-iff-abstract*:  $\text{redundant}\ A\ N \longleftrightarrow \text{abstract-red}\ A\ N$

**begin**

**definition** *saturated* :: 'a clauses  $\Rightarrow$  bool **where**

$\text{saturated}\ N \longleftrightarrow (\forall A\ B\ C. A \in N \longrightarrow B \in N \longrightarrow \neg \text{redundant}\ A\ N \longrightarrow \neg \text{redundant}\ B\ N$   
 $\longrightarrow \text{superposition-rules}\ A\ B\ C \longrightarrow \text{redundant}\ C\ N \vee C \in N)$

**lemma**

**assumes**  
*saturated*:  $\text{saturated}\ N$  **and**  
*finite*:  $\text{finite}\ N$  **and**  
*empty*:  $\{ \# \} \notin N$   
**shows**  $\text{INTERP}\ N \models_{hs} N$

$\langle proof \rangle$

**end**

**lemma** *tautology-is-redundant*:

**assumes** *tautology*  $C$   
**shows**  $\text{abstract-red}\ C\ N$   
 $\langle proof \rangle$

**lemma** *subsumed-is-redundant*:

**assumes**  $AB: A \subset \# B$   
**and**  $AN: A \in N$   
**shows**  $\text{abstract-red}\ B\ N$   
 $\langle proof \rangle$

**inductive** *redundant* :: 'a clause  $\Rightarrow$  'a clauses  $\Rightarrow$  bool **where**

*subsumption*:  $A \in N \Longrightarrow A \subset \# B \Longrightarrow \text{redundant}\ B\ N$

**lemma** *redundant-is-redundancy-criterion*:

**fixes**  $A :: 'a :: \text{wellorder clause}$  **and**  $N :: 'a :: \text{wellorder clauses}$   
**assumes**  $\text{redundant}\ A\ N$   
**shows**  $\text{abstract-red}\ A\ N$   
 $\langle proof \rangle$

**lemma** *redundant-mono*:

$\text{redundant}\ A\ N \Longrightarrow A \subseteq \# B \Longrightarrow \text{redundant}\ B\ N$   
 $\langle proof \rangle$

**locale** *truc* =

*selection*  $S$  **for**  $S :: \text{nat clause} \Rightarrow \text{nat clause}$

**begin**

end

end