Typed Ordered Resolution

Adnan Mohammed Ahmed Balazs Toth

September 3, 2025

Abstract

Ordered Resolution is a proof calculus for reasoning about first-order logic that is implemented in many automatic theorem provers. It works by saturating the given set of clauses and is refutationally complete, meaning that if the set is inconsistent, the saturation will contain a contradiction. In this formalization, we restructured the completeness proof to cleanly separate the ground (i.e., variable-free) and nonground aspects. We also added a type system to the calculus. We relied on the library for first-order clauses and on the saturation framework.

Contents

theory *Ground-Ordered-Resolution*
  imports
    First-Order-Clause.Selection-Function
    First-Order-Clause.Ground-Order
    First-Order-Clause.Literal-Functor
begin

1 Resolution Calculus

locale *ground-ordered-resolution-calculus* =

ground-order **where** $less_t = less_t$ +
selection-function select
for
$less_t$ :: $'t \Rightarrow 't \Rightarrow bool$ **and**
select :: $'t$ *clause* $\Rightarrow$ $'t$ *clause*
begin

1.1 Resolution Calculus

inductive *resolution* :: $'t$ *clause* $\Rightarrow$ $'t$ *clause* $\Rightarrow$ $'t$ *clause* $\Rightarrow$ *bool*
where
resolutionI:
$C = \textit{add-mset}\ L_C\ C' \Longrightarrow$
$D = \textit{add-mset}\ L_D\ D' \Longrightarrow$
$L_C = (\textit{Neg}\ t) \Longrightarrow$
$L_D = (\textit{Pos}\ t) \Longrightarrow$
$D \prec_c C \Longrightarrow$
$\textit{select}\ C = \{\#\} \wedge \textit{is-maximal}\ L_C\ C \vee \textit{is-maximal}\ L_C\ (\textit{select}\ C) \Longrightarrow$
$\textit{select}\ D = \{\#\} \Longrightarrow$
$\textit{is-strictly-maximal}\ L_D\ D \Longrightarrow$
$R = (C' + D') \Longrightarrow$
$\textit{resolution}\ D\ C\ R$

inductive *factoring* ::
$'t$ *clause* $\Rightarrow$ $'t$ *clause* $\Rightarrow$ *bool*
where
factoringI:
$C = \textit{add-mset}\ L_1\ (\textit{add-mset}\ L_1\ C') \Longrightarrow$
$L_1 = (\textit{Pos}\ t) \Longrightarrow$
$\textit{select}\ C = \{\#\} \Longrightarrow$
$\textit{is-maximal}\ L_1\ C \Longrightarrow$
$D = \textit{add-mset}\ L_1\ C' \Longrightarrow$
$\textit{factoring}\ C\ D$

1.2 Ground Layer

abbreviation *resolution-inferences* **where**
$\textit{resolution-inferences} \equiv \{\textit{Infer}\ [D,\ C]\ R \mid D\ C\ R.\ \textit{resolution}\ D\ C\ R\}$

abbreviation *factoring-inferences* **where**
$\textit{factoring-inferences} \equiv \{\textit{Infer}\ [C]\ D \mid C\ D.\ \textit{factoring}\ C\ D\}$

definition *G-Inf* :: $'t$ *clause inference set* **where**
$\textit{G-Inf} =$
$\{\textit{Infer}\ [D,\ C]\ R \mid D\ C\ R.\ \textit{resolution}\ D\ C\ R\} \cup$
$\{\textit{Infer}\ [P]\ C \mid P\ C.\ \textit{factoring}\ P\ C\}$

abbreviation *G-Bot* :: $'t$ *clause set* **where**
$\textit{G-Bot} \equiv \{\{\#\}\}$

definition *G-entails* :: $'t$ *clause set* $\Rightarrow$ $'t$ *clause set* $\Rightarrow$ *bool* **where**
G-entails $N_1\ N_2 \longleftrightarrow (\forall\ I.\ I \Vdash s\ N_1 \longrightarrow I \Vdash s\ N_2)$

1.3 Smaller Conclussions

lemma *ground-resolution-smaller-conclusion*:
assumes
step: *resolution D C R*
shows $R \prec_c C$
⟨*proof*⟩

lemma *ground-factoring-smaller-conclusion*:
assumes *step*: *factoring C D*
shows $D \prec_c C$
⟨*proof*⟩

end

1.4 Redundancy Criterion

sublocale *ground-ordered-resolution-calculus* ⊆ *consequence-relation* **where**
Bot = *G-Bot* **and**
entails = *G-entails*
⟨*proof*⟩

sublocale *ground-ordered-resolution-calculus* ⊆ *calculus-with-finitary-standard-redundancy*
where
Inf = *G-Inf* **and**
Bot = *G-Bot* **and**
entails = *G-entails* **and**
less = $(\prec_c)$
defines *GRed-I* = *Red-I* **and** *GRed-F* = *Red-F*
⟨*proof*⟩

end
theory *Relation-Extra*
imports *Main*
begin

lemma *partition-set-around-element*:
assumes *tot*: *totalp-on N R* **and** *x-in*: $x \in N$
shows $N = \{y \in N.\ R\ y\ x\} \cup \{x\} \cup \{y \in N.\ R\ x\ y\}$
⟨*proof*⟩

end
theory *Ground-Ordered-Resolution-Completeness*
imports
Ground-Ordered-Resolution
Relation-Extra
First-Order-Clause.HOL-Extra

begin

1.5 Mode Construction

context *ground-ordered-resolution-calculus*
begin

context
  fixes N :: $'t$ *clause set*
begin

function *epsilon* :: $'t$ *clause* $\Rightarrow$ $'t$ *set* **where**
  epsilon $C = \{A \mid A\ C'.$
    $C \in N \wedge$
    $C =$ *add-mset* $(Pos\ A)\ C' \wedge$
    select $C = \{\#\} \wedge$
    is-strictly-maximal $(Pos\ A)\ C \wedge$
    $\neg (\bigcup D \in \{D \in N.\ D \prec_c C\}.\ epsilon\ D) \models C\}$
  $\langle proof \rangle$

termination *epsilon*
$\langle proof \rangle$

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

end

lemma *epsilon-eq-empty-or-singleton*: $epsilon\ N\ C = \{\} \vee (\exists A.\ epsilon\ N\ C = \{A\})$
$\langle proof \rangle$

definition *rewrite-sys* **where**
  rewrite-sys $N\ C \equiv (\bigcup D \in \{D \in N.\ D \prec_c C\}.\ epsilon\ N\ D)$

lemma *rewrite-sys-subset-if-less-cls*: $C \prec_c D \longrightarrow$ *rewrite-sys* $N\ C \subseteq$ *rewrite-sys* $N\ D$
  $\langle proof \rangle$

lemma *mem-epsilonE*:
  assumes *rule-in*: $A \in epsilon\ N\ C$
  obtains C' **where**
    $C \in N$ **and**
    $C =$ *add-mset* $(Pos\ A)\ C'$ **and**
    select $C = \{\#\}$ **and**
    is-strictly-maximal $(Pos\ A)\ C$ **and**
    $\neg$ *rewrite-sys* $N\ C \models C$
  $\langle proof \rangle$

lemma *epsilon-unfold*: *epsilon* N C = $\{A \mid A\ C'.$
$C \in N \wedge$
$C = \textit{add-mset}\ (\textit{Pos}\ A)\ C' \wedge$
$\textit{select}\ C = \{\#\} \wedge$
$\textit{is-strictly-maximal}\ (\textit{Pos}\ A)\ C \wedge$
$\neg\ \textit{rewrite-sys}\ N\ C \models C\}$
⟨*proof*⟩

lemma *epsilon-subset-if-less-cls*: $C \prec_c D \Longrightarrow \textit{epsilon}\ N\ C \subseteq \textit{rewrite-sys}\ N\ D$
⟨*proof*⟩

lemma
assumes
$D \preceq_c C$ **and**
C-prod: $A \in \textit{epsilon}\ N\ C$ **and**
L-in: $L \in\#\ D$
shows
lesseq-trm-if-pos: $\textit{is-pos}\ L \Longrightarrow \textit{atm-of}\ L \preceq_t A$ **and**
less-trm-if-neg: $\textit{is-neg}\ L \Longrightarrow \textit{atm-of}\ L \prec_t A$
⟨*proof*⟩

lemma *less-trm-iff-less-cls-if-mem-epsilon*:
assumes *C-prod*: $A_C \in \textit{epsilon}\ N\ C$ **and** *D-prod*: $A_D \in \textit{epsilon}\ N\ D$
shows $A_C \prec_t A_D \longleftrightarrow C \prec_c D$
⟨*proof*⟩

lemma *false-cls-if-productive-epsilon*:
assumes *C-prod*: $A \in \textit{epsilon}\ N\ C$ **and** $D \in N$ **and** $C \prec_c D$
shows $\neg\ \textit{rewrite-sys}\ N\ D \models C - \{\#\textit{Pos}\ A\#\}$
⟨*proof*⟩

lemma *neg-notin-Interp-not-produce*:
$\textit{Neg}\ A \in\#\ C \Longrightarrow A \notin \textit{rewrite-sys}\ N\ D \cup \textit{epsilon}\ N\ D \Longrightarrow C \preceq_c D \Longrightarrow A \notin \textit{epsilon}\ N\ D''$
⟨*proof*⟩

lemma *lift-interp-entails*:
assumes
D-in: $D \in N$ **and**
D-entailed: $\textit{rewrite-sys}\ N\ D \models D$ **and**
C-in: $C \in N$ **and**
D-lt-C: $D \prec_c C$
shows $\textit{rewrite-sys}\ N\ C \models D$
⟨*proof*⟩

lemma *produces-imp-in-interp*:
assumes $\textit{Neg}\ A \in\#\ C$ **and** *D-prod*: $A \in \textit{epsilon}\ N\ D$
shows $A \in \textit{rewrite-sys}\ N\ C$
⟨*proof*⟩

lemma *split-Union-epsilon*:
assumes *D-in*: $D \in N$
shows $(\bigcup C \in N.\ \mathit{epsilon}\ N\ C) =$
$\mathit{rewrite\text{-}sys}\ N\ D \cup \mathit{epsilon}\ N\ D \cup (\bigcup C \in \{C \in N.\ D \prec_c C\}.\ \mathit{epsilon}\ N\ C)$
⟨*proof*⟩

lemma *split-Union-epsilon′*:
assumes *D-in*: $D \in N$
shows $(\bigcup C \in N.\ \mathit{epsilon}\ N\ C) = \mathit{rewrite\text{-}sys}\ N\ D \cup (\bigcup C \in \{C \in N.\ D \preceq_c C\}.$
$\mathit{epsilon}\ N\ C)$
⟨*proof*⟩

lemma *lift-entailment-to-Union*:
fixes $N\ D$
assumes
D-in: $D \in N$ **and**
R_D*-entails-D*: $\mathit{rewrite\text{-}sys}\ N\ D \Vdash D$
shows
$(\bigcup C \in N.\ \mathit{epsilon}\ N\ C) \Vdash D$
⟨*proof*⟩

lemma *true-cls-if-productive-epsilon*:
assumes $A \in \mathit{epsilon}\ N\ C\ C \prec_c D$
shows $\mathit{rewrite\text{-}sys}\ N\ D \Vdash C$
⟨*proof*⟩

lemma *model-preconstruction*:
fixes
$N :: {}'t\ \mathit{clause\ set}$ **and**
$C :: {}'t\ \mathit{clause}$
defines
$\mathit{entails} \equiv \lambda E\ C.\ E \Vdash C$
assumes $\mathit{saturated}\ N$ **and** $\{\#\} \notin N$ **and** *C-in*: $C \in N$
shows
$\mathit{epsilon}\ N\ C = \{\} \longleftrightarrow \mathit{entails}\ (\mathit{rewrite\text{-}sys}\ N\ C)\ C$
$(\bigcup D \in N.\ \mathit{epsilon}\ N\ D) \Vdash C$
$D \in N \Longrightarrow C \prec_c D \Longrightarrow \mathit{entails}\ (\mathit{rewrite\text{-}sys}\ N\ D)\ C$
⟨*proof*⟩

lemma *model-construction*:
fixes
$N :: {}'t\ \mathit{clause\ set}$ **and**
$C :: {}'t\ \mathit{clause}$
defines $\mathit{entails} \equiv \lambda E\ C.\ E \Vdash C$
assumes $\mathit{saturated}\ N$ **and** $\{\#\} \notin N$ **and** *C-in*: $C \in N$
shows $\mathit{entails}\ (\bigcup D \in N.\ \mathit{epsilon}\ N\ D)\ C$
⟨*proof*⟩

1.6 Static Refutational Completeness

lemma *statically-complete*:
  fixes N :: $'t$ *clause set*
  assumes *saturated* N **and** *G-entails* N $\{\{\#\}\}$
  shows $\{\#\} \in N$
  ⟨*proof*⟩

sublocale *statically-complete-calculus* **where**
  Bot = *G-Bot* **and**
  Inf = *G-Inf* **and**
  entails = *G-entails* **and**
  Red-I = *Red-I* **and**
  Red-F = *Red-F*
  ⟨*proof*⟩

end

end
theory *Ground-Ordered-Resolution-Soundness*
  imports *Ground-Ordered-Resolution*
begin

lemma (**in** *ground-ordered-resolution-calculus*) *soundness-ground-resolution*:
  assumes
    step: *resolution* D C R
  shows *G-entails* $\{D, C\}$ $\{R\}$
  ⟨*proof*⟩

lemma (**in** *ground-ordered-resolution-calculus*) *soundness-ground-factoring*:
  assumes *step*: *factoring* C D
  shows *G-entails* $\{C\}$ $\{D\}$
  ⟨*proof*⟩

sublocale *ground-ordered-resolution-calculus* ⊆ *sound-inference-system* **where**
  Inf = *G-Inf* **and**
  Bot = *G-Bot* **and**
  entails = *G-entails*
⟨*proof*⟩

end
theory *Ordered-Resolution*
  imports
    First-Order-Clause.Nonground-Order
    First-Order-Clause.Nonground-Selection-Function
    First-Order-Clause.Nonground-Typing
    First-Order-Clause.Typed-Tiebreakers
    Saturation-Framework.Lifting-to-Non-Ground-Calculi

    Ground-Ordered-Resolution

begin

locale *ordered-resolution-calculus* =
witnessed-nonground-typing **where**
welltyped = *welltyped* **and** *term-to-ground* = *term-to-ground* :: $'t \Rightarrow 't_G$ +
nonground-order **where** $less_t = less_t$ +
nonground-selection-function **where**
select = *select* **and** *atom-subst* = $(\cdot t)$ **and** *atom-vars* = *term.vars* **and**
atom-from-ground = *term.from-ground* **and** *atom-to-ground* = *term.to-ground* +
tiebreakers tiebreakers
for
select :: *'t select* **and**
$less_t$ :: $'t \Rightarrow 't \Rightarrow bool$ **and**
tiebreakers :: $('t_G, 't)$ *tiebreakers* **and**
welltyped :: $('v$ :: *infinite*, $'ty)$ *var-types* $\Rightarrow 't \Rightarrow 'ty \Rightarrow bool$
begin

inductive *factoring* :: $('t, 'v, 'ty)$ *typed-clause* $\Rightarrow$ $('t, 'v, 'ty)$ *typed-clause* $\Rightarrow bool$
where
factoringI:
C = *add-mset* L_1 (*add-mset* L_2 C') $\Longrightarrow$
$L_1 = (Pos\ t_1) \Longrightarrow$
$L_2 = (Pos\ t_2) \Longrightarrow$
D = (*add-mset* L_1 C') $\cdot\ \mu \Longrightarrow$
factoring $(\mathcal{V}, C)$ $(\mathcal{V}, D)$
if
select $C = \{\#\}$
is-maximal $(L_1 \cdot l\ \mu)$ $(C \cdot \mu)$
type-preserving-on (*clause.vars* C) $\mathcal{V}$ μ
term.is-imgu μ $\{\{t_1, t_2\}\}$

inductive *resolution* ::
$('t, 'v, 'ty)$ *typed-clause* $\Rightarrow$ $('t, 'v, 'ty)$ *typed-clause* $\Rightarrow$ $('t, 'v, 'ty)$ *typed-clause* $\Rightarrow$ *bool*
where
resolutionI:
C = *add-mset* L_1 $C' \Longrightarrow$
D = *add-mset* L_2 $D' \Longrightarrow$
$L_1 = (Neg\ t_1) \Longrightarrow$
$L_2 = (Pos\ t_2) \Longrightarrow$
$R = (C' \cdot \varrho_1 + D' \cdot \varrho_2) \cdot \mu \Longrightarrow$
resolution $(\mathcal{V}_2, D)$ $(\mathcal{V}_1, C)$ $(\mathcal{V}_3, R)$
if
infinite-variables-per-type $\mathcal{V}_1$
infinite-variables-per-type $\mathcal{V}_2$
term.is-renaming ϱ_1
term.is-renaming ϱ_2
clause.vars $(C \cdot \varrho_1) \cap$ *clause.vars* $(D \cdot \varrho_2) = \{\}$

type-preserving-on (*clause.vars* ($C \cdot \varrho_1$) $\cup$ *clause.vars* ($D \cdot \varrho_2$)) $\mathcal{V}_3$ μ
term.is-imgu μ $\{\{t_1 \cdot t\ \varrho_1,\ t_2 \cdot t\ \varrho_2\}\}$
$\neg$ ($C \cdot \varrho_1 \odot \mu \preceq_c D \cdot \varrho_2 \odot \mu$)
select $C = \{\#\} \Longrightarrow$ *is-maximal* ($L_1 \cdot l\ \varrho_1 \odot \mu$) ($C \cdot \varrho_1 \odot \mu$)
select $C \neq \{\#\} \Longrightarrow$ *is-maximal* ($L_1 \cdot l\ \varrho_1 \odot \mu$) ((*select* C) $\cdot\ \varrho_1 \odot \mu$)
select $D = \{\#\}$
is-strictly-maximal ($L_2 \cdot l\ \varrho_2 \odot \mu$) ($D \cdot \varrho_2 \odot \mu$)
$\forall\, x \in$ *clause.vars* C. $\mathcal{V}_1\ x = \mathcal{V}_3$ (*term.rename* $\varrho_1\ x$)
$\forall\, x \in$ *clause.vars* D. $\mathcal{V}_2\ x = \mathcal{V}_3$ (*term.rename* $\varrho_2\ x$)
type-preserving-on (*clause.vars* C) $\mathcal{V}_1$ ϱ_1
type-preserving-on (*clause.vars* D) $\mathcal{V}_2$ ϱ_2

abbreviation *factoring-inferences* **where**
factoring-inferences $\equiv$ { *Infer* [C] D | C D. *factoring* C D}

abbreviation *resolution-inferences* **where**
resolution-inferences $\equiv$ { *Infer* [D, C] R | D C R. *resolution* D C R}

definition *inferences* :: ('*t*, '*v*, '*ty*) *typed-clause inference set* **where**
inferences $\equiv$ *resolution-inferences* $\cup$ *factoring-inferences*

abbreviation *bottom*$_F$ :: ('*t*, '*v*, '*ty*) *typed-clause set* ($\bot_F$) **where**
bottom$_F$ $\equiv$ $\{(\mathcal{V}, \{\#\})$ | $\mathcal{V}$. *infinite-variables-per-type* $\mathcal{V}$ }

end

end
theory *Grounded-Ordered-Resolution*
imports
Ordered-Resolution
Ground-Ordered-Resolution

First-Order-Clause.Grounded-Selection-Function
First-Order-Clause.Nonground-Inference
Saturation-Framework.Lifting-to-Non-Ground-Calculi

Polynomial-Factorization.Missing-List
begin

locale *grounded-ordered-resolution-calculus* =
ordered-resolution-calculus **where**
select = *select* **and** *welltyped* = *welltyped* **and** *term-from-ground* = *term-from-ground*
:: $'t_G \Rightarrow 't$ +
grounded-selection-function **where**
select = *select* **and**
atom-subst = ($\cdot t$) **and**
atom-vars = *term.vars* **and**
atom-from-ground = *term.from-ground* **and**
atom-to-ground = *term.to-ground* **and**

$is\text{-}ground\text{-}instance = is\text{-}ground\text{-}instance$
for
$select :: 't\ select$ **and**
$welltyped :: ('v :: infinite, 'ty)\ var\text{-}types \Rightarrow 't \Rightarrow 'ty \Rightarrow bool$
begin

sublocale *ground*: *ground-ordered-resolution-calculus* **where**
$less_t = (\prec_{tG})$ **and** $select = select_G$
rewrites
$multiset\text{-}extension.multiset\text{-}extension\ (\prec_{tG})\ ground.literal\text{-}to\text{-}mset = (\prec_{lG})$ **and**
$multiset\text{-}extension.multiset\text{-}extension\ (\prec_{lG})\ (\lambda x.\ x) = (\prec_{cG})$ **and**
$\bigwedge l_G\ C_G.\ ground.is\text{-}maximal\ l_G\ C_G \longleftrightarrow ground\text{-}is\text{-}maximal\ l_G\ C_G$ **and**
$\bigwedge l_G\ C_G.\ ground.is\text{-}strictly\text{-}maximal\ l_G\ C_G \longleftrightarrow ground\text{-}is\text{-}strictly\text{-}maximal\ l_G\ C_G$
$\langle proof \rangle$

abbreviation *is-inference-ground-instance-one-premise* **where**
$is\text{-}inference\text{-}ground\text{-}instance\text{-}one\text{-}premise\ D\ C\ \iota_G\ \gamma \equiv$
$case\ (D,\ C)\ of\ ((\mathcal{V}',\ D),\ (\mathcal{V},\ C)) \Rightarrow$
$inference.is\text{-}ground\ (Infer\ [D]\ C \cdot\iota\ \gamma) \wedge$
$\iota_G = inference.to\text{-}ground\ (Infer\ [D]\ C \cdot\iota\ \gamma) \wedge$
$type\text{-}preserving\text{-}on\ (clause.vars\ C)\ \mathcal{V}\ \gamma \wedge$
$\mathcal{V} = \mathcal{V}' \wedge$
$infinite\text{-}variables\text{-}per\text{-}type\ \mathcal{V}$

abbreviation *is-inference-ground-instance-two-premises* **where**
$is\text{-}inference\text{-}ground\text{-}instance\text{-}two\text{-}premises\ D\ E\ C\ \iota_G\ \gamma\ \varrho_1\ \varrho_2 \equiv$
$case\ (D,\ E,\ C)\ of\ ((\mathcal{V}_2,\ D),\ (\mathcal{V}_1,\ E),\ (\mathcal{V}_3,\ C)) \Rightarrow$
$term.is\text{-}renaming\ \varrho_1 \wedge$
$term.is\text{-}renaming\ \varrho_2 \wedge$
$clause.vars\ (E \cdot \varrho_1) \cap clause.vars\ (D \cdot \varrho_2) = \{\} \wedge$
$inference.is\text{-}ground\ (Infer\ [D \cdot \varrho_2,\ E \cdot \varrho_1]\ C \cdot\iota\ \gamma) \wedge$
$\iota_G = inference.to\text{-}ground\ (Infer\ [D \cdot \varrho_2,\ E \cdot \varrho_1]\ C \cdot\iota\ \gamma) \wedge$
$type\text{-}preserving\text{-}on\ (clause.vars\ C)\ \mathcal{V}_3\ \gamma \wedge$
$infinite\text{-}variables\text{-}per\text{-}type\ \mathcal{V}_1 \wedge$
$infinite\text{-}variables\text{-}per\text{-}type\ \mathcal{V}_2 \wedge$
$infinite\text{-}variables\text{-}per\text{-}type\ \mathcal{V}_3$

abbreviation *is-inference-ground-instance* **where**
$is\text{-}inference\text{-}ground\text{-}instance\ \iota\ \iota_G\ \gamma \equiv$
$(case\ \iota\ of$
$Infer\ [D]\ C \Rightarrow is\text{-}inference\text{-}ground\text{-}instance\text{-}one\text{-}premise\ D\ C\ \iota_G\ \gamma$
$\mid Infer\ [D,\ E]\ C \Rightarrow \exists\, \varrho_1\ \varrho_2.\ is\text{-}inference\text{-}ground\text{-}instance\text{-}two\text{-}premises\ D\ E\ C\ \iota_G\ \gamma\ \varrho_1\ \varrho_2$
$\mid \text{-} \Rightarrow False)$
$\wedge\ \iota_G \in ground.G\text{-}Inf$

definition *inference-ground-instances* **where**
$inference\text{-}ground\text{-}instances\ \iota = \{\ \iota_G \mid \iota_G\ \gamma.\ is\text{-}inference\text{-}ground\text{-}instance\ \iota\ \iota_G\ \gamma\ \}$

lemma *is-inference-ground-instance*:
is-inference-ground-instance ι ι_G $\gamma \Longrightarrow \iota_G \in$ *inference-ground-instances* ι
⟨*proof*⟩

lemma *is-inference-ground-instance-one-premise*:
assumes *is-inference-ground-instance-one-premise* D C ι_G γ $\iota_G \in$ *ground.G-Inf*
shows $\iota_G \in$ *inference-ground-instances* (*Infer* [D] C)
⟨*proof*⟩

lemma *is-inference-ground-instance-two-premises*:
assumes *is-inference-ground-instance-two-premises* D E C ι_G γ ϱ_1 ϱ_2 $\iota_G \in$ *ground.G-Inf*
shows $\iota_G \in$ *inference-ground-instances* (*Infer* [D, E] C)
⟨*proof*⟩

lemma *ground-inference_concl-in-ground-instances*:
assumes $\iota_G \in$ *inference-ground-instances* ι
shows *concl-of* $\iota_G \in$ *uncurried-ground-instances* (*concl-of* ι)
⟨*proof*⟩

lemma *ground-inference-red-in-ground-instances-of-concl*:
assumes $\iota_G \in$ *inference-ground-instances* ι
shows $\iota_G \in$ *ground.Red-I* (*uncurried-ground-instances* (*concl-of* ι))
⟨*proof*⟩

sublocale *lifting*:
tiebreaker-lifting
$\bot_F$
inferences
ground.G-Bot
ground.G-entails
ground.G-Inf
ground.GRed-I
ground.GRed-F
uncurried-ground-instances
Some $\circ$ *inference-ground-instances*
typed-tiebreakers
⟨*proof*⟩
end

context *ordered-resolution-calculus*
begin

abbreviation *grounded-inference-ground-instances* **where**
grounded-inference-ground-instances $select_G \equiv$
grounded-ordered-resolution-calculus.inference-ground-instances
($\odot$) *Var* ($\cdot t$) *term.vars* *term.to-ground* ($\prec_t$) *term.from-ground* $select_G$ *welltyped*

sublocale
  lifting-intersection
    inferences
    $\{\{\#\}\}$
    $select_{Gs}$
    ground-ordered-resolution-calculus.G-Inf $(\prec_{tG})$
    λ-. *ground-ordered-resolution-calculus.G-entails*
    ground-ordered-resolution-calculus.GRed-I $(\prec_{tG})$
    λ-. *ground-ordered-resolution-calculus.GRed-F* $(\prec_{tG})$
    $\bot_F$
    λ-. *uncurried-ground-instances*
    $\lambda select_G$. *Some* $\circ$ *grounded-inference-ground-instances* $select_G$
    typed-tiebreakers
⟨*proof*⟩

end

end
theory *Ordered-Resolution-Soundness*
  imports *Grounded-Ordered-Resolution*
begin

1.7 Soundness

context *grounded-ordered-resolution-calculus*
begin

notation *lifting.entails-$\mathcal{G}$* (**infix** $\models_F$ *50*)

lemma *factoring-sound*:
  assumes *factoring*: *factoring C D*
  shows $\{C\} \models_F \{D\}$
  ⟨*proof*⟩

lemma *resolution-sound*:
  assumes *resolution*: *resolution D C R*
  shows $\{C, D\} \models_F \{R\}$
  ⟨*proof*⟩

end

sublocale *grounded-ordered-resolution-calculus* $\subseteq$ *sound-inference-system inferences*
$\bot_F$ $(\models_F)$
⟨*proof*⟩

sublocale *ordered-resolution-calculus* $\subseteq$ *sound-inference-system inferences* $\bot_F$ *entails-$\mathcal{G}$*
⟨*proof*⟩

end
theory *Ordered-Resolution-Completeness*
  imports
    Grounded-Ordered-Resolution
    Ground-Ordered-Resolution-Completeness
begin

2 Completeness

context *grounded-ordered-resolution-calculus*
begin

2.1 Liftings

lemma *factoring-lifting*:
  fixes
    $D_G\ C_G :: 't_G\ clause$ **and**
    $D\ C :: 't\ clause$ **and**
    $\gamma :: 'v \Rightarrow 't$
  defines
    $[simp]$: $D_G \equiv clause.to\text{-}ground\ (D \cdot \gamma)$ **and**
    $[simp]$: $C_G \equiv clause.to\text{-}ground\ (C \cdot \gamma)$
  assumes
    ground-factoring: $ground.factoring\ D_G\ C_G$ **and**
    D-grounding: $clause.is\text{-}ground\ (D \cdot \gamma)$ **and**
    C-grounding: $clause.is\text{-}ground\ (C \cdot \gamma)$ **and**
    select: $clause.from\text{-}ground\ (select_G\ D_G) = (select\ D) \cdot \gamma$ **and**
    type-preserving-γ: $type\text{-}preserving\text{-}on\ (clause.vars\ D)\ \mathcal{V}\ \gamma$ **and**
    $\mathcal{V}$: $infinite\text{-}variables\text{-}per\text{-}type\ \mathcal{V}$
  obtains C'
  where
    $factoring\ (\mathcal{V},\ D)\ (\mathcal{V},\ C')$
    $Infer\ [D_G]\ C_G \in inference\text{-}ground\text{-}instances\ (Infer\ [(\mathcal{V},\ D)]\ (\mathcal{V},\ C'))$
    $C' \cdot \gamma = C \cdot \gamma$
  ⟨*proof*⟩

lemma *resolution-lifting*:
  fixes
    $C_G\ D_G\ R_G :: 't_G\ clause$ **and**
    $C\ D\ R :: 't\ clause$ **and**
    $\gamma\ \varrho_1\ \varrho_2 :: 'v \Rightarrow 't$ **and**
    $\mathcal{V}_1\ \mathcal{V}_2 :: ('v,\ 'ty)\ var\text{-}types$
  defines
    $[simp]$: $C_G \equiv clause.to\text{-}ground\ (C \cdot \varrho_1 \odot \gamma)$ **and**
    $[simp]$: $D_G \equiv clause.to\text{-}ground\ (D \cdot \varrho_2 \odot \gamma)$ **and**
    $[simp]$: $R_G \equiv clause.to\text{-}ground\ (R \cdot \gamma)$ **and**
    $[simp]$: $N_G \equiv ground\text{-}instances\ \mathcal{V}_1\ C \cup ground\text{-}instances\ \mathcal{V}_2\ D$ **and**
    $[simp]$: $\iota_G \equiv Infer\ [D_G,\ C_G]\ R_G$
  assumes

$\quad$ *ground-resolution*: *ground.resolution* D_G C_G R_G **and**
$\quad$ ϱ_1: *term.is-renaming* ϱ_1 **and**
$\quad$ ϱ_2: *term.is-renaming* ϱ_2 **and**
$\quad$ *rename-apart*: *clause.vars* $(C \cdot \varrho_1) \cap$ *clause.vars* $(D \cdot \varrho_2) = \{\}$ **and**
$\quad$ *C-grounding*: *clause.is-ground* $(C \cdot \varrho_1 \odot \gamma)$ **and**
$\quad$ *D-grounding*: *clause.is-ground* $(D \cdot \varrho_2 \odot \gamma)$ **and**
$\quad$ *R-grounding*: *clause.is-ground* $(R \cdot \gamma)$ **and**
$\quad$ *select-from-C*: *clause.from-ground* $(select_G\ C_G) = (select\ C) \cdot \varrho_1 \odot \gamma$ **and**
$\quad$ *select-from-D*: *clause.from-ground* $(select_G\ D_G) = (select\ D) \cdot \varrho_2 \odot \gamma$ **and**
$\quad$ *type-preserving-*ϱ_1*-*γ: *type-preserving-on* (*clause.vars* C) $\mathcal{V}_1$ $(\varrho_1 \odot \gamma)$ **and**
$\quad$ *type-preserving-*ϱ_2*-*γ: *type-preserving-on* (*clause.vars* D) $\mathcal{V}_2$ $(\varrho_2 \odot \gamma)$ **and**
$\quad$ *type-preserving-*ϱ_1: *type-preserving-on* (*clause.vars* C) $\mathcal{V}_1$ ϱ_1 **and**
$\quad$ *type-preserving-*ϱ_2: *type-preserving-on* (*clause.vars* D) $\mathcal{V}_2$ ϱ_2 **and**
$\quad$ $\mathcal{V}_1$: *infinite-variables-per-type* $\mathcal{V}_1$ **and**
$\quad$ $\mathcal{V}_2$: *infinite-variables-per-type* $\mathcal{V}_2$
obtains R' $\mathcal{V}_3$
where
$\quad$ *resolution* $(\mathcal{V}_2, D)$ $(\mathcal{V}_1, C)$ $(\mathcal{V}_3, R')$
$\quad$ $\iota_G \in$ *inference-ground-instances* (*Infer* $[(\mathcal{V}_2, D), (\mathcal{V}_1, C)]$ $(\mathcal{V}_3, R')$)
$\quad$ $R' \cdot \gamma = R \cdot \gamma$
$\langle proof \rangle$

2.2 Ground instances

context
fixes ι_G N
assumes
$\quad$ *subst-stability*: *subst-stability-on* N **and**
$\quad$ ι_G*-Inf-from*: $\iota_G \in$ *ground.Inf-from-q* $select_G$ $(\bigcup($*uncurried-ground-instances* ‘ $N))$
begin

lemma *factoring-ground-instance*:
assumes *ground-factoring*: $\iota_G \in$ *ground.factoring-inferences*
obtains ι **where**
$\quad$ $\iota \in$ *Inf-from* N
$\quad$ $\iota_G \in$ *inference-ground-instances* ι
$\langle proof \rangle$

lemma *resolution-ground-instance*:
assumes *ground-resolution*: $\iota_G \in$ *ground.resolution-inferences*
obtains ι **where**
$\quad$ $\iota \in$ *Inf-from* N
$\quad$ $\iota_G \in$ *inference-ground-instances* ι
$\langle proof \rangle$

lemma *ground-instances*:
obtains ι **where**

$\iota \in$ *Inf-from N*
$\iota_G \in$ *inference-ground-instances* ι
⟨*proof*⟩

end

end

context *ordered-resolution-calculus*
begin

lemma *overapproximation*:
obtains $select_G$ **where**
ground-Inf-overapproximated $select_G$ *premises*
is-grounding $select_G$
⟨*proof*⟩

sublocale *statically-complete-calculus* $\bot_F$ *inferences entails-*$\mathcal{G}$ *Red-I-*$\mathcal{G}$ *Red-F-*$\mathcal{G}$
⟨*proof*⟩

end

end
theory *Ordered-Resolution-Welltypedness-Preservation*
imports *Grounded-Ordered-Resolution*
begin

context *ordered-resolution-calculus*
begin

lemma *factoring-preserves-typing*:
assumes *factoring*: *factoring* $(\mathcal{V}, C)$ $(\mathcal{V}, D)$
shows *clause.is-welltyped* $\mathcal{V}$ $C \longleftrightarrow$ *clause.is-welltyped* $\mathcal{V}$ D
⟨*proof*⟩

lemma *resolution-preserves-typing*:
assumes
resolution: *resolution* $(\mathcal{V}_2, D)$ $(\mathcal{V}_1, C)$ $(\mathcal{V}_3, R)$ **and**
D-is-welltyped: *clause.is-welltyped* $\mathcal{V}_2$ D **and**
C-is-welltyped: *clause.is-welltyped* $\mathcal{V}_1$ C
shows *clause.is-welltyped* $\mathcal{V}_3$ R
⟨*proof*⟩

end

end
theory *Untyped-Ordered-Resolution*
imports
First-Order-Clause.Nonground-Order

First-Order-Clause.Nonground-Selection-Function
First-Order-Clause.Tiebreakers

Fresh-Identifiers.Fresh
begin

locale *untyped-ordered-resolution-calculus* =
nonground-order **where**
$less_t$ = $less_t$ **and** *Var* = *Var* **and** *term-from-ground* = *term-from-ground* :: $'t_G$ ⇒ $'t$ +

nonground-selection-function **where**
select = *select* **and** *atom-subst* = $(\cdot t)$ **and** *atom-vars* = *term.vars* **and**
atom-from-ground = *term.from-ground* **and** *atom-to-ground* = *term.to-ground* **and** *Var* = *Var* +

tiebreakers tiebreakers +

term: *exists-imgu* **where** *vars* = *term-vars* **and** *subst* = $(\cdot t)$ **and** *id-subst* = *Var*
for
select :: $'t$ *select* **and**
$less_t$ :: $'t \Rightarrow 't \Rightarrow$ *bool* **and**
tiebreakers :: $('t_G, 't)$ *tiebreakers* **and**
Var :: $'v$ :: *infinite* ⇒ $'t$
begin

inductive *factoring* :: $'t$ *clause* ⇒ $'t$ *clause* ⇒ *bool*
where
factoringI:
C = *add-mset* L_1 (*add-mset* L_2 C') $\Longrightarrow$
$L_1 = (Pos\ t_1) \Longrightarrow$
$L_2 = (Pos\ t_2) \Longrightarrow$
D = (*add-mset* L_1 C') $\cdot\ \mu \Longrightarrow$
factoring C D
if
select $C = \{\#\}$
is-maximal $(L_1 \cdot l\ \mu)$ $(C \cdot \mu)$
term.is-imgu μ $\{\{t_1, t_2\}\}$

inductive *resolution* :: $'t$ *clause* ⇒ $'t$ *clause* ⇒ $'t$ *clause* ⇒ *bool*
where
resolutionI:
C = *add-mset* L_1 $C' \Longrightarrow$
D = *add-mset* L_2 $D' \Longrightarrow$
$L_1 = (Neg\ t_1) \Longrightarrow$
$L_2 = (Pos\ t_2) \Longrightarrow$
$R = (C' \cdot \varrho_1 + D' \cdot \varrho_2) \cdot \mu \Longrightarrow$
resolution D C R

if
term.is-renaming ϱ_1
term.is-renaming ϱ_2
clause.vars $(C \cdot \varrho_1) \cap$ *clause.vars* $(D \cdot \varrho_2) = \{\}$
term.is-imgu μ $\{\{t_1 \cdot t\ \varrho_1,\ t_2 \cdot t\ \varrho_2\}\}$
$\neg\ (C \cdot \varrho_1 \odot \mu \preceq_c D \cdot \varrho_2 \odot \mu)$
select $C = \{\#\} \Longrightarrow$ *is-maximal* $(L_1 \cdot l\ \varrho_1 \odot \mu)$ $(C \cdot \varrho_1 \odot \mu)$
select $C \neq \{\#\} \Longrightarrow$ *is-maximal* $(L_1 \cdot l\ \varrho_1 \odot \mu)$ $((select\ C) \cdot \varrho_1 \odot \mu)$
select $D = \{\#\}$
is-strictly-maximal $(L_2 \cdot l\ \varrho_2 \odot \mu)$ $(D \cdot \varrho_2 \odot \mu)$

abbreviation *factoring-inferences* **where**
factoring-inferences $\equiv$ \{ *Infer* [C] D | C D. *factoring* C D\}

abbreviation *resolution-inferences* **where**
resolution-inferences $\equiv$ \{ *Infer* [D, C] R | D C R. *resolution* D C R\}

definition *inferences* :: *'t clause inference set* **where**
inferences $\equiv$ *resolution-inferences* $\cup$ *factoring-inferences*

abbreviation *bottom* :: *'t clause set* **where**
bottom $\equiv$ $\{\{\#\}\}$

end

end
theory *Untyped-Ordered-Resolution-Inference-System*
imports
Untyped-Ordered-Resolution
First-Order-Clause.Untyped-Calculus
Grounded-Ordered-Resolution
begin

context *untyped-ordered-resolution-calculus*
begin

sublocale *typed*: *ordered-resolution-calculus* **where**
welltyped $= \lambda$- - (). *True*
$\langle proof \rangle$

declare
typed.term.welltyped-renaming [*simp del*]
typed.term.welltyped-subst-stability [*simp del*]
typed.term.welltyped-subst-stability' [*simp del*]

abbreviation *entails* **where**
entails N N' $\equiv$ *typed.entails-* $\mathcal{G}$ (*empty-typed* ‘ N) (*empty-typed* ‘ N')

sublocale *untyped-consequence-relation* **where**
typed-bottom = $\bot_F$ **and** *typed-entails* = *typed.entails-$\mathcal{G}$* **and**
bottom = *bottom* **and** *entails* = *entails*
⟨*proof*⟩

sublocale *untyped-inference-system* **where**
inferences = *inferences* **and** *typed-inferences* = *typed.inferences*
⟨*proof*⟩

end

end
theory *Untyped-Ordered-Resolution-Completeness*
imports
Untyped-Ordered-Resolution-Inference-System
Ordered-Resolution-Completeness
begin

context *untyped-ordered-resolution-calculus*
begin

abbreviation *Red-F* **where**
Red-F N ≡ *snd* ‘ *typed.Red-F-$\mathcal{G}$* (*empty-typed* ‘ *N*)

abbreviation *Red-I* **where**
Red-I N ≡ *remove-types* ‘ *typed.Red-I-$\mathcal{G}$* (*empty-typed* ‘ *N*)

sublocale *untyped-complete-calculus* **where**
typed-bottom = $\bot_F$ **and** *typed-entails* = *typed.entails-$\mathcal{G}$* **and**
typed-inferences = *typed.inferences* **and** *typed-Red-I* = *typed.Red-I-$\mathcal{G}$* **and**
typed-Red-F = *typed.Red-F-$\mathcal{G}$* **and** *bottom* = *bottom* **and** *inferences* = *inferences*
and *Red-F* = *Red-F* **and**
Red-I = *Red-I* **and** *entails* = *entails*
⟨*proof*⟩

end

end
theory *Untyped-Ordered-Resolution-Soundness*
imports
Untyped-Ordered-Resolution-Inference-System
Ordered-Resolution-Soundness
begin

context *untyped-ordered-resolution-calculus*
begin

sublocale *untyped-sound-inference-system* **where**

typed-bottom = $\bot_F$ **and** *typed-entails* = *typed.entails-*$\mathcal{G}$ **and**
typed-inferences = *typed.inferences* **and** *bottom* = *bottom* **and** *inferences* = *inferences* **and**
entails = *entails*
⟨*proof*⟩

end

end
theory *Monomorphic-Ordered-Resolution*
imports
Ordered-Resolution

First-Order-Clause.IsaFoR-Nonground-Clause
First-Order-Clause.Monomorphic-Typing
begin

locale *monomorphic-ordered-resolution-calculus* =
monomorphic-term-typing +

ordered-resolution-calculus **where**
comp-subst = $(\circ_s)$ **and** *Var* = *Var* **and** *term-subst* = $(\cdot)$ **and** *term-vars* = *term.vars* **and**
term-from-ground = *term.from-ground* **and** *term-to-ground* = *term.to-ground* **and** *welltyped* = *welltyped*

end
theory *Ordered-Resolution-Example*
imports
Monomorphic-Ordered-Resolution

First-Order-Clause.IsaFoR-KBO
begin

hide-type *Uprod-Literal-Functor.clause*

abbreviation *trivial-tiebreakers* ::
'f gterm clause ⇒ *('f,'v) term clause* ⇒ *('f,'v) term clause* ⇒ *bool* **where**
trivial-tiebreakers ≡ ⊥

abbreviation *trivial-select* :: *'a clause* ⇒ *'a clause* **where**
trivial-select - ≡ {#}

abbreviation *unit-typing* **where**
unit-typing - - ≡ *Some* ([], ())

interpretation *unit-types*: *witnessed-monomorphic-term-typing* **where** $\mathcal{F}$ = *unit-typing*
⟨*proof*⟩

interpretation *example1*: *monomorphic-ordered-resolution-calculus* **where**
select = *trivial-select* :: (('*f* :: *weighted* , '*v* :: *infinite*) *term*) *select* **and**
$less_t$ = *less-kbo* **and**
$\mathcal{F}$ = *unit-typing* **and**
tiebreakers = *trivial-tiebreakers*
⟨*proof*⟩

instantiation *nat* :: *infinite*
begin

instance
⟨*proof*⟩

end

datatype *type* = *A* | *B*

abbreviation *types* :: *nat* ⇒ *nat* ⇒ (*type list* × *type*) *option* **where**
types f n ≡
let type = *if even f then A else B*
in Some (*replicate n type*, *type*)

interpretation *example-types*: *witnessed-monomorphic-term-typing* **where** $\mathcal{F}$ = *types*
⟨*proof*⟩

interpretation *example2*: *monomorphic-ordered-resolution-calculus* **where**
select = *trivial-select* :: (*nat*, *nat*) *term select* **and**
$less_t$ = *less-kbo* **and**
$\mathcal{F}$ = *types* **and**
tiebreakers = *trivial-tiebreakers*
⟨*proof*⟩

end