

Basic First-Order Model Theory – A Translation from HOL Light

Sophie Touret and Lawrence Paulson

March 29, 2025

Abstract

This AFP entry presents a proof of compactness of first-order logic, the Löwenheim-Skolem theorem and the Uniformity lemma. It is a translation of a HOL Light formalization work by John Harrison [1]. Whenever possible, existing Isabelle/HOL theories have been used instead of direct translation.

Contents

theory *FOL-Syntax*

imports

Main

Propositional-Proof-Systems.Compactness

First-Order-Terms.Term

First-Order-Terms.Subterm-and-Context

begin

no-notation *Not* (\neg)

no-notation *And* (**infix** \wedge 68)

no-notation *Or* (**infix** \vee 68)

lemma *count-terms:*

OFCLASS('f::countable, 'v::countable) term, countable-class)

<proof>

instance *term :: (countable, countable) countable*

<proof>

type-synonym $nterm = \langle (nat, nat) term \rangle$

lemma $count\text{-}nterms: OFCLASS(nterm, countable\text{-}class)$
 $\langle proof \rangle$

instance $formula :: (countable) countable$
 $\langle proof \rangle$

term $test (0, [Var 0])$

abbreviation $functions\text{-}term :: \langle nterm \Rightarrow (nat \times nat) set \rangle$ **where**
 $\langle functions\text{-}term t \equiv funas\text{-}term t \rangle$

datatype $form =$
 $Bot \langle \perp \rangle$
 $| Atom (pred: nat) (args: \langle nterm list \rangle)$
 $| Implies form form \langle \mathbf{infixl} \langle \longrightarrow \rangle 85 \rangle$
 $| Forall nat form \langle \langle \forall \ . \ \rightarrow \ [0, 70] 70 \rangle \rangle$

fun $functions\text{-}form :: \langle form \Rightarrow (nat \times nat) set \rangle$ **where**
 $\langle functions\text{-}form \perp = \{ \} \rangle$
 $| \langle functions\text{-}form (Atom p ts) = (\bigcup t \in set\ ts. functions\text{-}term t) \rangle$
 $| \langle functions\text{-}form (\varphi \longrightarrow \psi) = functions\text{-}form \varphi \cup functions\text{-}form \psi \rangle$
 $| \langle functions\text{-}form (\forall x. \varphi) = functions\text{-}form \varphi \rangle$

fun $predicates\text{-}form :: \langle form \Rightarrow (nat \times nat) set \rangle$ **where**
 $\langle predicates\text{-}form \perp = \{ \} \rangle$
 $| \langle predicates\text{-}form (Atom p ts) = \{ (p, length\ ts) \} \rangle$
 $| \langle predicates\text{-}form (\varphi \longrightarrow \psi) = predicates\text{-}form \varphi \cup predicates\text{-}form \psi \rangle$
 $| \langle predicates\text{-}form (\forall x. \varphi) = predicates\text{-}form \varphi \rangle$

definition $functions\text{-}forms :: \langle form\ set \Rightarrow (nat \times nat) set \rangle$ **where**
 $\langle functions\text{-}forms fms \equiv \bigcup f \in fms. functions\text{-}form f \rangle$

definition $predicates :: \langle form\ set \Rightarrow (nat \times nat) set \rangle$ **where**
 $\langle predicates fms \equiv \bigcup f \in fms. predicates\text{-}form f \rangle$

definition $language :: \langle form\ set \Rightarrow ((nat \times nat) set \times (nat \times nat) set) \rangle$ **where**
 $\langle language fms = (functions\text{-}forms fms, predicates fms) \rangle$

lemma $lang\text{-}singleton: \langle language \{p\} = (functions\text{-}form p, predicates\text{-}form p) \rangle$
 $\langle proof \rangle$

abbreviation $Not :: \langle form \Rightarrow form \rangle \langle \langle \neg \rightarrow \ [90] 90 \rangle \rangle$ **where**
 $\langle \neg \varphi \equiv \varphi \longrightarrow \perp \rangle$

abbreviation $Top :: \langle form \rangle \langle \langle \top \rangle \rangle$ **where**
 $\langle \top \equiv \neg \perp \rangle$

abbreviation *Or* :: $\langle \text{form} \Rightarrow \text{form} \Rightarrow \text{form} \rangle$

(**infixl** $\langle \vee \rangle$ 84) **where**

$\langle \varphi \vee \psi \equiv (\varphi \longrightarrow \psi) \longrightarrow \psi \rangle$

abbreviation *And* :: $\langle \text{form} \Rightarrow \text{form} \Rightarrow \text{form} \rangle$

(**infixl** $\langle \wedge \rangle$ 84) **where**

$\langle \varphi \wedge \psi \equiv \neg (\neg \varphi \vee \neg \psi) \rangle$

abbreviation *Equiv* :: $\langle \text{form} \Rightarrow \text{form} \Rightarrow \text{form} \rangle$

(**infix** $\langle \longleftrightarrow \rangle$ 70) **where**

$\langle \varphi \longleftrightarrow \psi \equiv (\varphi \longrightarrow \psi \wedge \psi \longrightarrow \varphi) \rangle$

abbreviation *Exists* :: $\langle \text{nat} \Rightarrow \text{form} \Rightarrow \text{form} \rangle$

($\langle \exists \text{ .} \rightarrow [0, 70] \text{ } 70 \rangle$) **where**

$\langle \exists x. \varphi \equiv \neg (\forall x. \neg \varphi) \rangle$

lemma *ex-all-distinct*: $\langle \forall x. \varphi \neq \exists y. \psi \rangle$

$\langle \text{proof} \rangle$

abbreviation *FVT* :: $\langle \text{nterm} \Rightarrow \text{nat set} \rangle$ **where**

$\langle \text{FVT} \equiv \text{vars-term} \rangle$

fun *FV* :: $\langle \text{form} \Rightarrow \text{nat set} \rangle$ **where**

$\langle \text{FV} \perp = \{\} \rangle$

| $\langle \text{FV} (\text{Atom} - \text{ts}) = (\bigcup a \in \text{set ts. FVT } a) \rangle$

| $\langle \text{FV} (\varphi \longrightarrow \psi) = \text{FV } \varphi \cup \text{FV } \psi \rangle$

| $\langle \text{FV} (\forall x. \varphi) = \text{FV } \varphi - \{x\} \rangle$

lemma *FV-all-subs*: $\langle \text{FV } \varphi \subseteq \text{FV} (\forall x. \varphi) \cup \{x\} \rangle$

$\langle \text{proof} \rangle$

lemma *FV-exists*: $\langle \text{FV} (\exists x. \varphi) = \text{FV } \varphi - \{x\} \rangle$

$\langle \text{proof} \rangle$

lemma *finite-FV*: $\langle \text{finite} (\text{FV } \varphi) \rangle$

$\langle \text{proof} \rangle$

fun *BV* :: $\langle \text{form} \Rightarrow \text{nat set} \rangle$ **where**

$\langle \text{BV} \perp = \{\} \rangle$

| $\langle \text{BV} (\text{Atom} - \text{args}') = \{\} \rangle$

| $\langle \text{BV} (\varphi \longrightarrow \psi) = \text{BV } \varphi \cup \text{BV } \psi \rangle$

| $\langle \text{BV} (\forall x. \varphi) = \text{BV } \varphi \cup \{x\} \rangle$

lemma *finite-BV*: $\langle \text{finite} (\text{BV } \varphi) \rangle$

$\langle \text{proof} \rangle$

definition *variant* :: $\langle \text{nat set} \Rightarrow \text{nat} \rangle$ **where**

$\langle \text{variant } s = \text{Max } s + 1 \rangle$

lemma *variant-finite*: $\langle \text{finite } s \Longrightarrow \neg (\text{variant } s \in s) \rangle$

$\langle \text{proof} \rangle$

lemma *variant-form*: $\langle \neg \text{variant } (FV \ \varphi) \in FV \ \varphi \rangle$

$\langle \text{proof} \rangle$

fun *formsubst* :: $\langle \text{form} \Rightarrow (\text{nat}, \text{nat}) \text{ subst} \Rightarrow \text{form} \rangle$ (**infixl** $\langle \cdot_{fm} \rangle$ 75) **where**

$\langle \perp \cdot_{fm} - = \perp \rangle$

| $\langle (\text{Atom } p \ ts) \cdot_{fm} \ \sigma = \text{Atom } p \ [t \cdot \sigma. \ t \leftarrow ts] \rangle$

| $\langle (\varphi \longrightarrow \psi) \cdot_{fm} \ \sigma = (\varphi \cdot_{fm} \ \sigma) \longrightarrow (\psi \cdot_{fm} \ \sigma) \rangle$

| $\langle (\forall x. \ \varphi) \cdot_{fm} \ \sigma =$

$\quad (\text{let } \sigma' = \sigma(x := \text{Var } x);$

$\quad \quad z = \text{if } \exists y. \ y \in FV \ (\forall x. \ \varphi) \wedge x \in FVT \ (\sigma' \ y)$

$\quad \quad \quad \text{then variant } (FV \ (\varphi \cdot_{fm} \ \sigma')) \text{ else } x \text{ in}$

$\quad \quad \forall z. \ (\varphi \cdot_{fm} \ \sigma(x := \text{Var } z))) \rangle$

fun *formsubst2* :: $\langle \text{form} \Rightarrow (\text{nat}, \text{nat}) \text{ subst} \Rightarrow \text{form} \rangle$ (**infixl** $\langle \cdot_{fm2} \rangle$ 75) **where**

$\langle \perp \cdot_{fm2} - = \perp \rangle$

| $\langle (\text{Atom } p \ ts) \cdot_{fm2} \ \sigma = \text{Atom } p \ [t \cdot \sigma. \ t \leftarrow ts] \rangle$

| $\langle (\varphi \longrightarrow \psi) \cdot_{fm2} \ \sigma = (\varphi \cdot_{fm2} \ \sigma) \longrightarrow (\psi \cdot_{fm2} \ \sigma) \rangle$

| $\langle (\forall x. \ \varphi) \cdot_{fm2} \ \sigma = (\text{let } \sigma' = \sigma(x := \text{Var } x) \text{ in}$

$\quad (\text{if } \exists y. \ y \in FV \ (\forall x. \ \varphi) \wedge x \in FVT \ (\sigma' \ y)$

$\quad \quad \text{then } (\text{let } z = \text{variant } (FV \ (\varphi \cdot_{fm2} \ \sigma')) \text{ in}$

$\quad \quad \quad \forall z. \ (\varphi \cdot_{fm2} \ \sigma(x := \text{Var } z)))$

$\quad \quad \text{else } \forall x. \ (\varphi \cdot_{fm2} \ \sigma')) \rangle$

lemma *formsubst-def-switch*: $\langle \varphi \cdot_{fm} \ \sigma = \varphi \cdot_{fm2} \ \sigma \rangle$

$\langle \text{proof} \rangle$

lemma *termsubst-valuation*: $\langle \forall x \in FVT \ t. \ \sigma \ x = \sigma' \ x \Longrightarrow t \cdot \sigma = t \cdot \sigma' \rangle$

$\langle \text{proof} \rangle$

lemma *termsetsubst-valuation*: $\langle \forall y \in T. \ \forall x \in FVT \ y. \ \sigma \ x = \sigma' \ x \Longrightarrow t \in T \Longrightarrow t \cdot \sigma = t \cdot \sigma' \rangle$

$\langle \text{proof} \rangle$

lemma *formsubst-valuation*: $\langle \forall x \in (FV \ \varphi). \ (\text{Var } x) \cdot \sigma = (\text{Var } x) \cdot \sigma' \Longrightarrow \varphi \cdot_{fm} \ \sigma = \varphi \cdot_{fm} \ \sigma' \rangle$

$\langle \text{proof} \rangle$

lemma $\langle \{x. \ \exists y. \ y \in (s \cup t) \wedge P \ x \ y\} = \{x. \ \exists y. \ y \in s \wedge P \ x \ y\} \cup \{x. \ \exists y. \ y \in t \wedge P \ x \ y\} \rangle$

$\langle \text{proof} \rangle$

lemma *formsubst-structure-bot*: $\langle \varphi \cdot_{fm} \ \sigma = \perp \longleftrightarrow \varphi = \perp \rangle$

$\langle \text{proof} \rangle$

lemma *formsubst-structure-pred*: $\langle (\exists p \text{ ts}. \varphi \cdot_{fm} \sigma = \text{Atom } p \text{ ts}) \longleftrightarrow (\exists p \text{ ts}. \varphi = \text{Atom } p \text{ ts}) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-imp*: $\langle (\exists \varphi 1 \varphi 2. \varphi \cdot_{fm} \sigma = \varphi 1 \longrightarrow \varphi 2) \longleftrightarrow (\exists \psi 1 \psi 2. \varphi = \psi 1 \longrightarrow \psi 2) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-all*: $\langle (\exists x \psi. \varphi \cdot_{fm} \sigma = (\forall x. \psi)) \longleftrightarrow (\exists x \psi. \varphi = (\forall x. \psi)) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-not*: $\langle (\exists \psi. \varphi \cdot_{fm} \sigma = \text{Not } \psi) \longleftrightarrow (\exists \psi. \varphi = \text{Not } \psi) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-not-all-imp*:
 $\langle (\exists x \psi. \varphi \cdot_{fm} \sigma = (\forall x. \psi) \longrightarrow \perp) \longleftrightarrow (\exists x \psi. \varphi = (\forall x. \psi) \longrightarrow \perp) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-all-not*:
 $\langle (\exists x \psi. \varphi \cdot_{fm} \sigma = (\forall x. \psi \longrightarrow \perp)) \longleftrightarrow (\exists x \psi. \varphi = (\forall x. \psi \longrightarrow \perp)) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure-ex*: $\langle (\exists x \psi. \varphi \cdot_{fm} \sigma = (\exists x. \psi)) \longleftrightarrow (\exists x \psi. \varphi = (\exists x. \psi)) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-structure*: $\langle (\varphi \cdot_{fm} \sigma = \perp \longleftrightarrow \varphi = \perp) \wedge$
 $((\exists p \text{ ts}. \varphi \cdot_{fm} \sigma = \text{Atom } p \text{ ts}) \longleftrightarrow (\exists p \text{ ts}. \varphi = \text{Atom } p \text{ ts})) \wedge$
 $((\exists \varphi 1 \varphi 2. \varphi \cdot_{fm} \sigma = \varphi 1 \longrightarrow \varphi 2) \longleftrightarrow (\exists \psi 1 \psi 2. \varphi = \psi 1 \longrightarrow \psi 2)) \wedge$
 $((\exists x \psi. \varphi \cdot_{fm} \sigma = (\forall x. \psi)) \longleftrightarrow (\exists x \psi. \varphi = (\forall x. \psi))) \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-fv*: $\langle FV (\varphi \cdot_{fm} \sigma) = \{x. \exists y. y \in (FV \varphi) \wedge x \in FVT ((Var y) \cdot \sigma)\} \rangle$
 $\langle \text{proof} \rangle$

lemma *subst-var [simp]*: $\langle \varphi \cdot_{fm} \text{Var } = \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *formsubst-rename*: $\langle FV (\varphi \cdot_{fm} (\text{subst } x (\text{Var } y))) - \{y\} = FV \varphi - \{x\} - \{y\} \rangle$
 $\langle \text{proof} \rangle$

lemma *termsubst-functions-term*:
 $\langle \text{functions-term } (t \cdot \sigma) = \text{functions-term } t \cup \{x. \exists y. y \in FVT t \wedge x \in \text{functions-term } ((Var y) \cdot \sigma)\} \rangle$

⟨proof⟩

lemma *formsubst-functions-form*:

⟨functions-form $(\varphi \cdot_{fm} \sigma) = \text{functions-form } \varphi \cup \{x. \exists y. y \in FV \varphi \wedge x \in \text{functions-term } ((Var y) \cdot \sigma)\}$ ⟩
⟨proof⟩

lemma *formsubst-predicates*: ⟨predicates-form $(\varphi \cdot_{fm} \sigma) = \text{predicates-form } \varphi$ ⟩
⟨proof⟩

lemma *formsubst-language-rewrite*: ⟨language $\{\varphi \cdot_{fm} \text{subst } x (Var y)\} = \text{language } \{\varphi\}$ ⟩
⟨proof⟩

end

theory *FOL-Semantics*

imports *FOL-Syntax*

begin

locale *struct* =

fixes

$M :: \langle 'm \text{ set} \rangle$ **and**
 $FN :: \langle \text{nat} \Rightarrow 'm \text{ list} \Rightarrow 'm \rangle$ **and**
 $REL :: \langle \text{nat} \Rightarrow 'm \text{ list set} \rangle$

assumes

$M\text{-nonempty}: \langle M \neq \{\} \rangle$

typedef $'m \text{ intrp} =$

⟨ $\{ (M :: 'm \text{ set}, FN :: \text{nat} \Rightarrow 'm \text{ list} \Rightarrow 'm, REL :: \text{nat} \Rightarrow 'm \text{ list set}). \text{struct } M \}$ ⟩
⟨proof⟩

declare *Abs-intrp-inverse* [simp] *Rep-intrp-inverse* [simp]

setup-lifting *type-definition-intrp*

lift-definition *dom* :: $\langle 'm \text{ intrp} \Rightarrow 'm \text{ set} \rangle$ **is** *fst* ⟨proof⟩

lift-definition *intrp-fn* :: $\langle 'm \text{ intrp} \Rightarrow (\text{nat} \Rightarrow 'm \text{ list} \Rightarrow 'm) \rangle$ **is** $\langle \text{fst} \circ \text{snd} \rangle$ ⟨proof⟩

lift-definition *intrp-rel* :: $\langle 'm \text{ intrp} \Rightarrow (\text{nat} \Rightarrow 'm \text{ list set}) \rangle$ **is** $\langle \text{snd} \circ \text{snd} \rangle$ ⟨proof⟩

lemma *intrp-is-struct* [iff]: ⟨struct $(\text{dom } \mathcal{M})$ ⟩
⟨proof⟩

lemma *dom-Abs-is-fst* [simp]: ⟨struct $M \implies \text{dom } (\text{Abs-intrp } (M, FN, REL)) = M$ ⟩
⟨proof⟩

lemma *intrp-fn-Abs-is-fst-snd* [simp]: ⟨struct $M \implies \text{intrp-fn } (\text{Abs-intrp } (M, FN,$

$REL)) = FN$
 $\langle proof \rangle$

lemma *intrap-rel-Abs-is-snd-snd* [simp]:
 $\langle struct\ M \implies intrp-rel\ (Abs-intrp\ (M, FN, REL)) = REL \rangle$
 $\langle proof \rangle$

definition *is-valuation* :: $\langle 'm\ intrp \Rightarrow (nat \Rightarrow 'm) \Rightarrow bool \rangle$ **where**
 $\langle is-valuation\ \mathcal{M}\ \beta \longleftrightarrow (\forall\ v.\ \beta\ v \in dom\ \mathcal{M}) \rangle$

lemma *valuation-valmod*: $\langle \llbracket is-valuation\ \mathcal{M}\ \beta; a \in dom\ \mathcal{M} \rrbracket \implies is-valuation\ \mathcal{M}\ (\beta(x := a)) \rangle$
 $\langle proof \rangle$

fun *eval*
 $:: \langle nterm \Rightarrow 'm\ intrp \Rightarrow (nat \Rightarrow 'm) \Rightarrow 'm \rangle$
 $\langle \langle \llbracket \cdot \rrbracket \cdot \cdot \rangle [50, 0, 0] 70 \rangle$ **where**
 $\langle \llbracket Var\ v \rrbracket \cdot \cdot \rangle^{\beta} = \beta\ v \rangle$
 $\mid \langle \llbracket Fun\ f\ ts \rrbracket \cdot \cdot \rangle^{\mathcal{M}, \beta} = intrp-fn\ \mathcal{M}\ f\ [\llbracket t \rrbracket \cdot \cdot]^{\mathcal{M}, \beta} \cdot t \leftarrow ts \rangle$

definition *list-all* :: $\langle 'a \Rightarrow bool \Rightarrow 'a\ list \Rightarrow bool \rangle$ **where**
 $\langle simp \rangle: \langle list-all\ P\ ls \longleftrightarrow (fold\ (\lambda l\ b.\ b \wedge P\ l)\ ls\ True) \rangle$

lemma *term-subst-eval*: $\langle intrp-fn\ M = Fun \implies t \cdot v = eval\ t\ M\ v \rangle$
 $\langle proof \rangle$

lemma *term-eval-triv*[simp]: $\langle intrp-fn\ M = Fun \implies eval\ t\ M\ Var = t \rangle$
 $\langle proof \rangle$

lemma *fold-bool-prop*: $\langle (fold\ (\lambda l\ b.\ b \wedge P\ l)\ ls\ b) = (b \wedge (\forall l \in set\ ls.\ P\ l)) \rangle$
 $\langle proof \rangle$

lemma *list-all-set*: $\langle list-all\ P\ ls = (\forall l \in set\ ls.\ P\ l) \rangle$
 $\langle proof \rangle$

hide-const *lang*

definition *is-interpretation* **where**
 $\langle is-interpretation\ lang\ \mathcal{M} \longleftrightarrow$
 $(\forall f\ l.\ (f, length(l)) \in fst\ lang \wedge set\ l \subseteq dom\ \mathcal{M} \longrightarrow intrp-fn\ \mathcal{M}\ f\ l \in dom\ \mathcal{M}) \rangle$

lemma *interpretation-sublanguage*: $\langle funs2 \subseteq funs1 \implies is-interpretation\ (funs1, pred1)\ \mathcal{M} \implies is-interpretation\ (funs2, preds2)\ \mathcal{M} \rangle$
 $\langle proof \rangle$

lemma *interpretation-eval*:

assumes \mathcal{M} : $\langle \text{is-interpretation (functions-term } t, \text{any) } \mathcal{M} \rangle$ **and** val : $\langle \text{is-valuation } \mathcal{M} \ \beta \rangle$
shows $\langle \llbracket t \rrbracket^{\mathcal{M}, \beta} \in \text{dom } \mathcal{M} \rangle$
 $\langle \text{proof} \rangle$

fun *holds*

$\langle \text{'m intrp} \Rightarrow (\text{nat} \Rightarrow \text{'m}) \Rightarrow \text{form} \Rightarrow \text{bool} \rangle$ $\langle \langle -, \models \rightarrow [30, 30, 80] 80 \rangle$ **where**
 $\langle \mathcal{M}, \beta \models \perp \longleftrightarrow \text{False} \rangle$
 $\langle \mathcal{M}, \beta \models \text{Atom } p \ ts \longleftrightarrow \llbracket t \rrbracket^{\mathcal{M}, \beta}. t \leftarrow ts \in \text{intrp-rel } \mathcal{M} \ p \rangle$
 $\langle \mathcal{M}, \beta \models \varphi \longrightarrow \psi \longleftrightarrow ((\mathcal{M}, \beta \models \varphi) \longrightarrow (\mathcal{M}, \beta \models \psi)) \rangle$
 $\langle \mathcal{M}, \beta \models (\forall x. \varphi) \longleftrightarrow (\forall a \in \text{dom } \mathcal{M}. \mathcal{M}, \beta(x := a) \models \varphi) \rangle$

lemma *holds-exists*: $\langle \mathcal{M}, \beta \models (\exists x. \varphi) \longleftrightarrow (\exists a \in \text{dom } \mathcal{M}. \mathcal{M}, \beta(x := a) \models \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *holds-indep- β -if*:

$\langle \forall v \in FV \ \varphi. \ \beta_1 \ v = \beta_2 \ v \implies \mathcal{M}, \beta_1 \models \varphi \longleftrightarrow \mathcal{M}, \beta_2 \models \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *holds-indep-intrp-if*:

fixes
 $\varphi :: \text{form}$ **and**
 $\mathcal{M} \ \mathcal{M}' :: \langle \text{'m intrp} \rangle$
assumes
 dom-eq : $\langle \text{dom } \mathcal{M} = \text{dom } \mathcal{M}' \rangle$ **and**
 rel-eq : $\langle \forall p. \text{intrp-rel } \mathcal{M} \ p = \text{intrp-rel } \mathcal{M}' \ p \rangle$ **and**
 fn-eq : $\langle \forall f \ ts. (f, \text{length } ts) \in \text{functions-form } \varphi \longrightarrow \text{intrp-fn } \mathcal{M} \ f \ ts = \text{intrp-fn } \mathcal{M}' \ f \ ts \rangle$
shows
 $\langle \forall \beta. \ \mathcal{M}, \beta \models \varphi \longleftrightarrow \mathcal{M}', \beta \models \varphi \rangle$
 $\langle \text{proof} \rangle$

the above in a more idiomatic form (it is a congruence rule)

corollary *holds-cong*:

assumes
 $\langle \text{dom } \mathcal{M} = \text{dom } \mathcal{M}' \rangle$
 $\langle \bigwedge p. \text{intrp-rel } \mathcal{M} \ p = \text{intrp-rel } \mathcal{M}' \ p \rangle$
 $\langle \bigwedge f \ ts. (f, \text{length } ts) \in \text{functions-form } \varphi \implies \text{intrp-fn } \mathcal{M} \ f \ ts = \text{intrp-fn } \mathcal{M}' \ f \ ts \rangle$
shows $\langle \mathcal{M}, \beta \models \varphi \longleftrightarrow \mathcal{M}', \beta \models \varphi \rangle$
 $\langle \text{proof} \rangle$

abbreviation (*input*) $\langle \text{termsubst } \mathcal{M} \ \beta \ \sigma \ v \equiv \llbracket \sigma \ v \rrbracket^{\mathcal{M}, \beta} \rangle$

lemma *subst-lemma-terms*: $\langle \llbracket t \cdot \sigma \rrbracket^{\mathcal{M}, \beta} = \llbracket t \rrbracket^{\mathcal{M}, \text{termsubst } \mathcal{M} \ \beta \ \sigma} \rangle$
 $\langle \text{proof} \rangle$

lemma *eval-indep- β -if*:
assumes $\langle \forall v \in FVT \ t. \ \beta \ v = \beta' \ v \rangle$
shows $\langle \llbracket t \rrbracket^{\mathcal{M}, \beta} = \llbracket t \rrbracket^{\mathcal{M}, \beta'} \rangle$
 $\langle \text{proof} \rangle$

lemma *concat-map*: $\langle [f \ t. \ t \leftarrow [g \ t. \ t \leftarrow ts]] = [f \ (g \ t). \ t \leftarrow ts] \rangle \langle \text{proof} \rangle$

lemma *swap-subst-eval*: $\langle \mathcal{M}, \beta \models (\varphi \cdot_{fm} \sigma) \iff \mathcal{M}, (\lambda v. \text{termsubst } \mathcal{M} \ \beta \ \sigma \ v) \models \varphi \rangle$
 $\langle \text{proof} \rangle$

definition *satisfies* :: $\langle 'm \ \text{intrp} \Rightarrow \text{form set} \Rightarrow \text{bool} \rangle$ **where**
 $\langle \text{satisfies } \mathcal{M} \ S \equiv (\forall \beta \ \varphi. \ \text{is-valuation } \mathcal{M} \ \beta \longrightarrow \varphi \in S \longrightarrow \mathcal{M}, \beta \models \varphi) \rangle$

lemma *satisfies-iff-satisfies-sing*: $\langle \text{satisfies } \mathcal{M} \ S \iff (\forall \varphi \in S. \ \text{satisfies } \mathcal{M} \ \{\varphi\}) \rangle$
 $\langle \text{proof} \rangle$

end

theory *Ground-FOL-Compactness*

imports

FOL-Semantics

begin

fun *qfree* :: $\langle \text{form} \Rightarrow \text{bool} \rangle$ **where**
 $\langle \text{qfree } \perp = \text{True} \rangle$
 $\mid \langle \text{qfree } (\text{Atom } p \ ts) = \text{True} \rangle$
 $\mid \langle \text{qfree } (\varphi \longrightarrow \psi) = (\text{qfree } \varphi \wedge \text{qfree } \psi) \rangle$
 $\mid \langle \text{qfree } (\forall x. \ \varphi) = \text{False} \rangle$

lemma *qfree-iff-BV-empty*: $\text{qfree } \varphi \iff BV \ \varphi = \{\}$
 $\langle \text{proof} \rangle$

lemma *qfree-no-quantif*: $\langle \text{qfree } r \implies \neg(\exists x \ p. \ r = \forall x. \ p) \wedge \neg(\exists x \ p. \ r = \exists x. \ p) \rangle$
 $\langle \text{proof} \rangle$

lemma *qfree-formsubst*: $\langle \text{qfree } \varphi \equiv \text{qfree } (\varphi \cdot_{fm} \sigma) \rangle$
 $\langle \text{proof} \rangle$

fun *form-to-formula* :: $\text{form} \Rightarrow (\text{nat} \times \text{nterm list}) \ \text{formula}$ **where**

$\langle \text{form-to-formula } \perp = \perp \rangle$
 $\mid \langle \text{form-to-formula } (\text{Atom } p \ ts) = \text{formula.Atom } (p, ts) \rangle$
 $\mid \langle \text{form-to-formula } (\varphi \longrightarrow \psi) = \text{Imp } (\text{form-to-formula } \varphi) (\text{form-to-formula } \psi) \rangle$
 $\mid \langle \text{form-to-formula } (\forall x. \varphi) = \perp \rangle$

fun *pholds* :: $\langle (\text{form} \Rightarrow \text{bool}) \Rightarrow \text{form} \Rightarrow \text{bool} \rangle \langle \cdot \models_p \cdot \rightarrow [30, 80] 80 \rangle$ **where**
 $\langle I \models_p \perp \longleftrightarrow \text{False} \rangle$
 $\mid \langle I \models_p \text{Atom } p \ ts \longleftrightarrow I (\text{Atom } p \ ts) \rangle$
 $\mid \langle I \models_p \varphi \longrightarrow \psi \longleftrightarrow ((I \models_p \varphi) \longrightarrow (I \models_p \psi)) \rangle$
 $\mid \langle I \models_p (\forall x. \varphi) \longleftrightarrow I (\forall x. \varphi) \rangle$

definition *psatisfiable* :: $\text{form set} \Rightarrow \text{bool}$ **where**
 $\langle \text{psatisfiable } S \equiv \exists I. \forall \varphi \in S. I \models_p \varphi \rangle$

abbreviation *psatisfies where* $\langle \text{psatisfies } I \ \Phi \equiv \forall \varphi \in \Phi. \text{pholds } I \ \varphi \rangle$

definition *val-to-prop-val* :: $(\text{form} \Rightarrow \text{bool}) \Rightarrow ((\text{nat} \times \text{nterm list}) \Rightarrow \text{bool})$ **where**
 $\langle \text{val-to-prop-val } I = (\lambda x. I (\text{Atom } (\text{fst } x) (\text{snd } x))) \rangle$

lemma *pholds-Not*: $\langle I \models_p \text{Not } \varphi \longleftrightarrow \neg (I \models_p \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *pentails-equiv*: $\langle \text{qfree } \varphi \Longrightarrow (I \models_p \varphi \equiv (\text{val-to-prop-val } I) \models (\text{form-to-formula } \varphi)) \rangle$
 $\langle \text{proof} \rangle$

lemma *pentails-equiv-set*:
assumes *all-qfree*: $\langle \forall \varphi \in S. \text{qfree } \varphi \rangle$
shows $\langle \text{psatisfiable } S \equiv \text{sat } (\text{form-to-formula } 'S) \rangle$
 $\langle \text{proof} \rangle$

definition *finsat* :: $\text{form set} \Rightarrow \text{bool}$ **where**
 $\langle \text{finsat } S \equiv \forall T \subseteq S. \text{finite } T \longrightarrow \text{psatisfiable } T \rangle$

lemma *finsat-fin-sat-eq*:
assumes *all-qfree*: $\langle \forall \varphi \in S. \text{qfree } \varphi \rangle$
shows $\langle \text{finsat } S \longleftrightarrow \text{fin-sat } (\text{form-to-formula } 'S) \rangle$
 $\langle \text{proof} \rangle$

lemma *psatisfiable-mono*: $\langle \text{psatisfiable } S \Longrightarrow T \subseteq S \Longrightarrow \text{psatisfiable } T \rangle$
 $\langle \text{proof} \rangle$

lemma *finsat-mono*: $\langle \text{finsat } S \Longrightarrow T \subseteq S \Longrightarrow \text{finsat } T \rangle$
 $\langle \text{proof} \rangle$

lemma *finsat-satisfiable*: $\langle \text{psatisfiable } S \Longrightarrow \text{finsat } S \rangle$
 $\langle \text{proof} \rangle$

lemma *prop-compactness*: $\langle (\forall \varphi \in S. \text{qfree } \varphi) \Longrightarrow \text{finsat } S = \text{psatisfiable } S \rangle$

$\langle \text{proof} \rangle$

as above, more in the style of HOL Light

lemma *compact-prop*:

assumes $\langle \bigwedge B. \llbracket \text{finite } B; B \subseteq A \rrbracket \implies \exists I. \text{psatisfies } I B \rangle$ **and** $\langle \bigwedge \varphi. \varphi \in A \longrightarrow \text{qfree } \varphi \rangle$

shows $\langle \exists I. \text{psatisfies } I A \rangle$

$\langle \text{proof} \rangle$

Three results required for the FOL uniformity theorem

lemma *compact-prop-alt*:

assumes $\langle \bigwedge I. \exists \varphi \in A. I \models_p \varphi \rangle$ $\langle \bigwedge \varphi. \varphi \in A \longrightarrow \text{qfree } \varphi \rangle$

obtains B **where** $\langle \text{finite } B \rangle$ $\langle B \subseteq A \rangle$ $\langle \bigwedge I. \exists \varphi \in B. I \models_p \varphi \rangle$

$\langle \text{proof} \rangle$

lemma *finite-disj-lemma*:

assumes $\langle \text{finite } A \rangle$

shows $\langle \exists \Phi. \text{set } \Phi \subseteq A \wedge (\forall I. I \models_p \text{foldr } (\vee) \Phi \perp \longleftrightarrow (\exists \varphi \in A. I \models_p \varphi)) \rangle$

$\langle \text{proof} \rangle$

lemma *compact-disj*:

assumes $\langle \bigwedge I. \exists \varphi \in A. I \models_p \varphi \rangle$ $\langle \bigwedge \varphi. \varphi \in A \longrightarrow \text{qfree } \varphi \rangle$

obtains Φ **where** $\langle \text{set } \Phi \subseteq A \rangle$ $\langle \bigwedge I. I \models_p \text{foldr } (\vee) \Phi \perp \rangle$

$\langle \text{proof} \rangle$

end

theory *Prenex-Normal-Form*

imports

Ground-FOL-Compactness

begin

inductive *is-prenex* :: *form* \Rightarrow *bool* **where**

$\langle \text{qfree } \varphi \implies \text{is-prenex } \varphi \rangle$

| $\langle \text{is-prenex } \varphi \implies \text{is-prenex } (\forall x. \varphi) \rangle$

| $\langle \text{is-prenex } \varphi \implies \text{is-prenex } (\exists x. \varphi) \rangle$

inductive-simps *is-prenex-simps* [*simp*]:

is-prenex *Bot*

is-prenex (*Atom* *p* *ts*)

is-prenex ($\varphi \longrightarrow \psi$)

is-prenex ($\forall x. \varphi$)

lemma *prenex-formsubst1*: $\langle \text{is-prenex } \varphi \implies \text{is-prenex } (\varphi \cdot_{fm} \sigma) \rangle$

$\langle \text{proof} \rangle$

lemma *prenex-formsubst2*: $\langle \text{is-prenex } (\varphi \cdot_{fm} \sigma) \implies \text{is-prenex } \varphi \rangle$

⟨proof⟩

lemma *prenex-formsubst*: ⟨*is-prenex* ($\varphi \cdot_{fm} \sigma$) \equiv *is-prenex* φ ⟩
⟨proof⟩

lemma *prenex-imp*: ⟨*is-prenex* ($\varphi \longrightarrow \psi$) \implies
qfree ($\varphi \longrightarrow \psi$) \vee ($\psi = \perp \wedge (\exists x \varphi'. \textit{is-prenex } \varphi' \wedge \varphi = (\forall x. \varphi' \longrightarrow \perp))$)⟩
⟨proof⟩

inductive *universal* :: *form* \Rightarrow *bool* **where**
⟨*qfree* $\varphi \implies$ *universal* φ ⟩
| ⟨*universal* $\varphi \implies$ *universal* ($\forall x. \varphi$)⟩

inductive-simps *universal-simps* [*simp*]:
universal *Bot*
universal (*Atom* p ts)
universal ($\varphi \longrightarrow \psi$)
universal ($\forall x. \varphi$)

fun *size* :: *form* \Rightarrow *nat* **where**
⟨*size* $\perp = 1$ ⟩
| ⟨*size* (*Atom* p ts) = 1⟩
| ⟨*size* ($\varphi \longrightarrow \psi$) = *size* φ + *size* ψ ⟩
| ⟨*size* ($\forall x. \varphi$) = 1 + *size* φ ⟩

lemma *wf-size*: ⟨*wfP* ($\lambda\varphi \psi. \textit{size } \varphi < \textit{size } \psi$)⟩
⟨proof⟩

lemma *size-indep-subst*: ⟨*size* ($\varphi \cdot_{fm} \sigma$) = *size* φ ⟩
⟨proof⟩

lemma *prenex-distinct*: ⟨ $(\forall x. \varphi) \neq (\exists y. \psi)$ ⟩
⟨proof⟩

lemma *uniq-all-x*: *Uniq* ($\lambda x. \exists p. r = \forall x. p$)
⟨proof⟩

lemma *uniq-all-p*: ⟨*Uniq* ($(\lambda p. r = \forall (THE\ x. \exists p. r = \forall x. p). p)$)⟩
⟨proof⟩

lemma *uniq-ex-x*: *Uniq* ($\lambda x. \exists p. r = \exists x. p$)
⟨proof⟩

lemma *uniq-ex-p*: ⟨*Uniq* ($(\lambda p. r = \exists (THE\ x. \exists p. r = \exists x. p). p)$)⟩
⟨proof⟩

definition *ppat* :: (*nat* \Rightarrow *form* \Rightarrow *form*) \Rightarrow (*nat* \Rightarrow *form* \Rightarrow *form*) \Rightarrow (*form* \Rightarrow
form) \Rightarrow *form* \Rightarrow *form* **where**

$\langle \text{ppat } A \ B \ C \ r = (\text{if } (\exists x \ p. \ r = \forall x. \ p) \ \text{then}$
 $\quad A \ (\text{THE } x. \ \exists p. \ r = \forall x. \ p) \ (\text{THE } p. \ r = \forall (\text{THE } x. \ \exists p. \ r = \forall x. \ p). \ p)$
 $\text{else } (\text{if } \exists x \ p. \ r = \exists x. \ p \ \text{then}$
 $\quad B \ (\text{THE } x. \ \exists p. \ r = \exists x. \ p) \ (\text{THE } p. \ r = \exists (\text{THE } x. \ \exists p. \ r = \exists x. \ p). \ p)$
 $\text{else } C \ r) \rangle$

lemma *ppat-simpA*: $\langle \forall x \ p. \ \text{ppat } A \ B \ C \ (\forall x. \ p) = A \ x \ p \rangle$
 $\langle \text{proof} \rangle$

lemma *ppat-simpB*: $\langle \forall x \ p. \ \text{ppat } A \ B \ C \ (\exists x. \ p) = B \ x \ p \rangle$
 $\langle \text{proof} \rangle$

lemma *ppat-last*: $\langle (\forall r. \ \neg(\exists x \ p. \ r = \forall x. \ p) \wedge \neg(\exists x \ p. \ r = \exists x. \ p)) \implies \text{ppat } A \ B \ C \ r = C \ r \rangle$
 $\langle \text{proof} \rangle$

lemma *ppat-last-qfree*: $\langle \text{qfree } r \implies \text{ppat } A \ B \ C \ r = C \ r \rangle$
 $\langle \text{proof} \rangle$

lemma *ppat-to-ex-qfree*:
 $\langle (\exists f. \ (\forall x \ p \ q. \ f \ p \ (\forall x. \ q) = ((A :: \text{form} \Rightarrow \text{nat} \Rightarrow \text{form} \Rightarrow \text{form}) \ p) \ x \ q) \wedge$
 $\quad (\forall x \ p \ q. \ f \ p \ (\exists x. \ q) = (B \ p) \ x \ q) \wedge$
 $\quad (\forall p \ q. \ \text{qfree } q \longrightarrow f \ p \ q = (C \ p) \ q)) \rangle$
 $\langle \text{proof} \rangle$

lemma *size-rec*:
 $\langle \forall f \ g \ x. \ (\forall (z :: \text{form}). \ \text{size } z < \text{size } x \longrightarrow (f \ z = g \ z)) \longrightarrow (H \ f \ x = H \ g \ x) \implies$
 $\quad (\exists f. \ \forall x. \ f \ x = H \ f \ x) \rangle$
 $\langle \text{proof} \rangle$

abbreviation *prenex-right-forall* :: $(\text{form} \Rightarrow \text{form} \Rightarrow \text{form}) \Rightarrow \text{form} \Rightarrow \text{nat} \Rightarrow \text{form} \Rightarrow \text{form}$ **where**
 $\langle \text{prenex-right-forall} \equiv$
 $\quad (\lambda p \ \varphi \ x \ \psi. \ (\text{let } y = \text{variant}(FV \ \varphi \cup FV \ (\forall x. \ \psi)) \ \text{in } (\forall y. \ p \ \varphi \ (\psi \cdot_{fm} (\text{subst } x \ (\text{Var } y)))))) \rangle$

abbreviation *prenex-right-exists* :: $(\text{form} \Rightarrow \text{form} \Rightarrow \text{form}) \Rightarrow \text{form} \Rightarrow \text{nat} \Rightarrow \text{form} \Rightarrow \text{form}$ **where**
 $\langle \text{prenex-right-exists} \equiv$
 $\quad (\lambda p \ \varphi \ x \ \psi. \ (\text{let } y = \text{variant}(FV \ \varphi \cup FV \ (\exists x. \ \psi)) \ \text{in } (\exists y. \ p \ \varphi \ (\psi \cdot_{fm} (\text{subst } x \ (\text{Var } y)))))) \rangle$

lemma *prenex-right-ex*:
 $\langle \exists \text{prenex-right}. \ (\forall \varphi \ x \ \psi. \ \text{prenex-right } \varphi \ (\forall x. \ \psi) = \text{prenex-right-forall } \text{prenex-right } \varphi \ x \ \psi)$
 $\quad \wedge \ (\forall \varphi \ x \ \psi. \ \text{prenex-right } \varphi \ (\exists x. \ \psi) = \text{prenex-right-exists } \text{prenex-right } \varphi \ x \ \psi)$

$\wedge (\forall \varphi \psi. \text{qfree } \psi \longrightarrow \text{prenex-right } \varphi \psi = (\varphi \longrightarrow \psi))\rangle$
 $\langle \text{proof} \rangle$

consts *prenex-right* :: *form* \Rightarrow *form* \Rightarrow *form*

specification (*prenex-right*) \langle

$(\forall \varphi x \psi. \text{prenex-right } \varphi (\forall x. \psi) = \text{prenex-right-forall } \text{prenex-right } \varphi x \psi) \wedge$
 $(\forall \varphi x \psi. \text{prenex-right } \varphi (\exists x. \psi) = \text{prenex-right-exists } \text{prenex-right } \varphi x \psi) \wedge$
 $(\forall \varphi \psi. \text{qfree } \psi \longrightarrow \text{prenex-right } \varphi \psi = (\varphi \longrightarrow \psi))\rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-right-qfree-case*: $\langle \text{qfree } \psi \Longrightarrow \text{prenex-right } \varphi \psi = (\varphi \longrightarrow \psi) \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-right-all-case*: $\langle \text{prenex-right } \varphi (\forall x. \psi) = \text{prenex-right-forall } \text{prenex-right } \varphi x \psi \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-right-exist-case*: $\langle \text{prenex-right } \varphi (\exists x. \psi) = \text{prenex-right-exists } \text{prenex-right } \varphi x \psi \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-right-exists-shape-case*:

$\langle \exists x2 \sigma. \text{prenex-right } \varphi (\exists x. \psi) = \exists x2. \text{prenex-right } \varphi (\psi \cdot_{fm} \sigma) \rangle$
 $\langle \text{proof} \rangle$

abbreviation *prenex-left-forall* :: (*form* \Rightarrow *form* \Rightarrow *form*) \Rightarrow *form* \Rightarrow *nat* \Rightarrow *form*
 \Rightarrow *form* **where**

$\langle \text{prenex-left-forall} \equiv$
 $(\lambda p \varphi x \psi. (\text{let } y = \text{variant}(FV (\forall x. \varphi) \cup FV \psi) \text{ in } (\exists y. p (\varphi \cdot_{fm} (\text{subst } x$
 $(\text{Var } y))) \psi))) \rangle$

abbreviation *prenex-left-exists* :: (*form* \Rightarrow *form* \Rightarrow *form*) \Rightarrow *form* \Rightarrow *nat* \Rightarrow *form*
 \Rightarrow *form* **where**

$\langle \text{prenex-left-exists} \equiv$
 $(\lambda p \varphi x \psi. (\text{let } y = \text{variant}(FV (\exists x. \varphi) \cup FV \psi) \text{ in } (\forall y. p (\varphi \cdot_{fm} (\text{subst } x$
 $(\text{Var } y))) \psi))) \rangle$

lemma *prenex-left-ex*:

$\langle \exists \text{prenex-left}. (\forall \varphi x \psi. \text{prenex-left } (\forall x. \varphi) \psi = \text{prenex-left-forall } \text{prenex-left } \varphi x \psi)$
 $\wedge (\forall \varphi x \psi. \text{prenex-left } (\exists x. \varphi) \psi = \text{prenex-left-exists } \text{prenex-left } \varphi x \psi)$
 $\wedge (\forall \varphi \psi. \text{qfree } \varphi \longrightarrow \text{prenex-left } \varphi \psi = \text{prenex-right } \varphi \psi) \rangle$
 $\langle \text{proof} \rangle$

definition *prenex-left where* $\langle \text{prenex-left} = (\text{SOME } \text{prenex-left}.$

$(\forall \varphi x \psi. \text{prenex-left } (\forall x. \varphi) \psi = \text{prenex-left-forall } \text{prenex-left } \varphi x \psi) \wedge$
 $(\forall \varphi x \psi. \text{prenex-left } (\exists x. \varphi) \psi = \text{prenex-left-exists } \text{prenex-left } \varphi x \psi) \wedge$

$\langle \forall \varphi \psi. \text{qfree } \varphi \longrightarrow \text{prenex-left } \varphi \psi = \text{prenex-right } \varphi \psi \rangle$

lemma *prenex-left-forall-case*: $\langle \text{prenex-left } (\forall x. \varphi) \psi = \text{prenex-left-forall } \text{prenex-left } \varphi \ x \ \psi \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-left-qfree-case*: $\langle \text{qfree } \varphi \implies \text{prenex-left } \varphi \psi = \text{prenex-right } \varphi \psi \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-left-exists-case*: $\langle \text{prenex-left } (\exists x. \varphi) \psi = \text{prenex-left-exists } \text{prenex-left } \varphi \ x \ \psi \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-left-exists-shape-case*:
 $\langle \exists x2 \ \sigma. \text{prenex-left } (\exists x. \varphi) \psi = \forall x2. \text{prenex-left } (\varphi \cdot_{fm} \sigma) \psi \rangle$
 $\langle \text{proof} \rangle$

fun *prenex where*
 $\langle \text{prenex } \perp = \perp \rangle$
 $| \langle \text{prenex } (\text{Atom } p \ ts) = \text{Atom } p \ ts \rangle$
 $| \langle \text{prenex } (\varphi \longrightarrow \psi) = \text{prenex-left } (\text{prenex } \varphi) (\text{prenex } \psi) \rangle$
 $| \langle \text{prenex } (\forall x. \varphi) = \forall x. (\text{prenex } \varphi) \rangle$

lemma *holds-indep-forall*:
assumes *y-notin*: $\langle y \notin FV (\forall x. \varphi) \rangle$
shows $\langle (I, \beta \models (\forall x. \varphi) \longleftrightarrow I, \beta \models (\forall y. \varphi \cdot_{fm} (\text{subst } x (\text{Var } y)))) \rangle$
 $\langle \text{proof} \rangle$

lemma *forall-imp-commute*:
assumes *y-notin*: $\langle y \notin FV \varphi \rangle$
shows $\langle ((I :: 'a \text{ intrp}), \beta \models (\varphi \longrightarrow (\forall y. \psi)) \longleftrightarrow I, \beta \models (\forall y. \varphi \longrightarrow \psi)) \rangle$
 $\langle \text{proof} \rangle$

lemma *forall-imp-exists*:
assumes *y-notin*: $\langle y \notin FV \psi \rangle$
shows $\langle ((I :: 'a \text{ intrp}), \beta \models ((\forall y. \varphi) \longrightarrow \psi) \longleftrightarrow I, \beta \models (\exists y. (\varphi \longrightarrow \psi))) \rangle$
 $\langle \text{proof} \rangle$

lemma *exists-imp-forall*:
assumes *y-notin*: $\langle y \notin FV \psi \rangle$
shows $\langle (I, \beta \models ((\exists y. \varphi) \longrightarrow \psi) \longleftrightarrow I, \beta \models (\forall y. (\varphi \longrightarrow \psi))) \rangle$
 $\langle \text{proof} \rangle$

lemma *exists-imp-commute*:
assumes *y-notin*: $\langle y \notin FV \varphi \rangle$
shows $\langle ((I :: 'a \text{ intrp}), \beta \models (\varphi \longrightarrow (\exists y. \psi)) \longleftrightarrow I, \beta \models (\exists y. \varphi \longrightarrow \psi)) \rangle$
 $\langle \text{proof} \rangle$

lemma holds-indep-exists:

$\langle y \notin FV (\exists x. \varphi) \implies (I, \beta \models (\exists x. \varphi)) \iff I, \beta \models (\exists y. \varphi \cdot_{fm} (subst\ x\ (Var\ y)))) \rangle$
 $\langle proof \rangle$

lemma prenex-right-forall-is:

assumes $\langle dom\ I \neq \{\} \rangle$
shows $\langle ((I, \beta \models \varphi \longrightarrow (\forall x. \psi)) \iff (I, \beta \models (\forall (variant\ (FV\ \varphi \cup FV\ (\forall x. \psi)))))) \rangle$
 $\langle (\varphi \longrightarrow (\psi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ \varphi \cup FV\ (\forall x. \psi)))))))) \rangle$
(is ?lhs = ?rhs)
 $\langle proof \rangle$

lemma prenex-right-exists-is:

assumes $\langle dom\ I \neq \{\} \rangle$
shows $\langle ((I, \beta \models \varphi \longrightarrow (\exists x. \psi)) \iff (I, \beta \models (\exists (variant\ (FV\ \varphi \cup FV\ (\exists x. \psi)))))) \rangle$
 $\langle (\varphi \longrightarrow (\psi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ \varphi \cup FV\ (\exists x. \psi)))))))) \rangle$
(is ?lhs = ?rhs)
 $\langle proof \rangle$

lemma prenex-left-forall-is:

assumes $\langle dom\ I \neq \{\} \rangle$
shows $\langle (I, \beta \models ((\forall x. \varphi) \longrightarrow \psi)) \equiv (I, \beta \models (\exists (variant\ (FV\ (\forall x. \varphi) \cup FV\ \psi))) \longrightarrow \psi) \rangle$
 $\langle ((\varphi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ (\forall x. \varphi) \cup FV\ \psi)))) \longrightarrow \psi)) \rangle$
 $\langle proof \rangle$

lemma prenex-left-exists-is:

assumes $\langle dom\ I \neq \{\} \rangle$
shows $\langle (I, \beta \models ((\exists x. \varphi) \longrightarrow \psi)) \equiv (I, \beta \models (\forall (variant\ (FV\ (\exists x. \varphi) \cup FV\ \psi))) \longrightarrow \psi) \rangle$
 $\langle ((\varphi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ (\exists x. \varphi) \cup FV\ \psi)))) \longrightarrow \psi)) \rangle$
 $\langle proof \rangle$

lemma prenex-right-forall-FV: $\langle FV (\varphi \longrightarrow (\forall x. \psi)) =$

$FV (\forall (variant\ (FV\ \varphi \cup FV\ (\forall x. \psi)))) \cdot (\varphi \longrightarrow (\psi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ \varphi \cup FV\ (\forall x. \psi)))))) \rangle$
 $\langle proof \rangle$

lemma prenex-right-exists-FV: $\langle FV (\varphi \longrightarrow (\exists x. \psi)) =$

$FV (\forall (variant\ (FV\ \varphi \cup FV\ (\exists x. \psi)))) \cdot (\varphi \longrightarrow (\psi \cdot_{fm} (subst\ x\ (Var\ (variant\ (FV\ \varphi \cup FV\ (\exists x. \psi)))))) \rangle$
 $\langle proof \rangle$

lemma prenex-left-forall-FV: $\langle FV ((\forall x. \varphi) \longrightarrow \psi) =$

$FV (\exists (\text{variant } (FV (\forall x. \varphi) \cup FV \psi)). ((\varphi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV (\forall x. \varphi) \cup FV \psi)))) \longrightarrow \psi)))$
 $\langle \text{proof} \rangle$

lemma *prenex-left-exists-FV*: $\langle FV ((\exists x. \varphi) \longrightarrow \psi) =$
 $FV (\forall (\text{variant } (FV (\exists x. \varphi) \cup FV \psi)). ((\varphi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV (\exists x. \varphi) \cup FV \psi)))) \longrightarrow \psi)))$
 $\langle \text{proof} \rangle$

lemma *prenex-right-forall-language*: $\langle \text{language } \{\varphi \longrightarrow (\forall x. \psi)\} =$
 $\text{language } \{\forall (\text{variant } (FV \varphi \cup FV (\forall x. \psi))). (\varphi \longrightarrow (\psi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV \varphi \cup FV (\forall x. \psi))))))\}$
 $\langle \text{proof} \rangle$

lemma *prenex-right-exists-language*: $\langle \text{language } \{\varphi \longrightarrow (\exists x. \psi)\} =$
 $\text{language } \{\exists (\text{variant } (FV \varphi \cup FV (\exists x. \psi))). (\varphi \longrightarrow (\psi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV \varphi \cup FV (\exists x. \psi))))))\}$
 $\langle \text{proof} \rangle$

lemma *prenex-left-forall-language*: $\langle \text{language } \{(\forall x. \varphi) \longrightarrow \psi\} =$
 $\text{language } \{\exists (\text{variant } (FV (\forall x. \varphi) \cup FV \psi)). ((\varphi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV (\forall x. \varphi) \cup FV \psi)))) \longrightarrow \psi)\}$
 $\langle \text{proof} \rangle$

lemma *prenex-left-exists-language*: $\langle \text{language } \{(\exists x. \varphi) \longrightarrow \psi\} =$
 $\text{language } \{\forall (\text{variant } (FV (\exists x. \varphi) \cup FV \psi)). ((\varphi \cdot_{fm} (\text{subst } x (\text{Var } (\text{variant } (FV (\exists x. \varphi) \cup FV \psi)))) \longrightarrow \psi)\}$
 $\langle \text{proof} \rangle$

lemma *prenex-props-forall*: $\langle P \wedge FV \varphi = FV \psi \wedge \text{language } \{\varphi\} = \text{language } \{\psi\}$
 \wedge
 $(\forall (I :: 'a \text{ intrp}) \beta. \text{dom } I \neq \{\} \longrightarrow (I, \beta \models \varphi \longleftrightarrow I, \beta \models \psi)) \implies$
 $P \wedge FV (\forall x. \varphi) = FV (\forall x. \psi) \wedge \text{language } \{(\forall x. \varphi)\} = \text{language } \{(\forall x. \psi)\} \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. \text{dom } I \neq \{\} \longrightarrow (I, \beta \models (\forall x. \varphi) \longleftrightarrow I, \beta \models (\forall x. \psi)))$
 \rangle
 $\langle \text{proof} \rangle$

lemma *prenex-props-exists*: $\langle P \wedge FV \varphi = FV \psi \wedge \text{language } \{\varphi\} = \text{language } \{\psi\}$
 \wedge
 $(\forall (I :: 'a \text{ intrp}) \beta. \text{dom } I \neq \{\} \longrightarrow (I, \beta \models \varphi \longleftrightarrow I, \beta \models \psi)) \implies$
 $P \wedge FV (\exists x. \varphi) = FV (\exists x. \psi) \wedge \text{language } \{(\exists x. \varphi)\} = \text{language } \{(\exists x. \psi)\} \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. \text{dom } I \neq \{\} \longrightarrow (I, \beta \models (\exists x. \varphi) \longleftrightarrow I, \beta \models (\exists x. \psi)))$
 \rangle
 $\langle \text{proof} \rangle$

lemma *prenex-right-props-imp0*:
assumes $\langle \text{qfree } \varphi \rangle$

shows $\langle is\text{-prenex } \psi \implies is\text{-prenex } (prenex\text{-right } \varphi \psi) \rangle$
 $\langle proof \rangle$

lemma *prenex-right-props-imp*:

assumes $\langle qfree \varphi \rangle$
shows $\langle is\text{-prenex } \psi \implies$
 $is\text{-prenex } (prenex\text{-right } \varphi \psi) \wedge$
 $FV (prenex\text{-right } \varphi \psi) = FV (\varphi \longrightarrow \psi) \wedge$
 $language \{prenex\text{-right } \varphi \psi\} = language \{(\varphi \longrightarrow \psi)\} \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. dom I \neq \{\} \longrightarrow ((I, \beta \models (prenex\text{-right } \varphi \psi)) \longleftrightarrow (I, \beta$
 $\models (\varphi \longrightarrow \psi)))) \rangle$
(is $\langle is\text{-prenex } \psi \implies ?P \psi \rangle$
 $\langle proof \rangle$

lemma *prenex-right-props*:

$\langle qfree \varphi \wedge is\text{-prenex } \psi \implies$
 $is\text{-prenex } (prenex\text{-right } \varphi \psi) \wedge$
 $FV (prenex\text{-right } \varphi \psi) = FV (\varphi \longrightarrow \psi) \wedge$
 $language \{prenex\text{-right } \varphi \psi\} = language \{(\varphi \longrightarrow \psi)\} \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. dom I \neq \{\} \longrightarrow ((I, \beta \models (prenex\text{-right } \varphi \psi)) \longleftrightarrow (I, \beta \models (\varphi$
 $\longrightarrow \psi)))) \rangle$
 $\langle proof \rangle$

lemma *prenex-left-props-imp0*:

assumes $\langle is\text{-prenex } \psi \rangle$
shows $\langle is\text{-prenex } \varphi \implies is\text{-prenex } (prenex\text{-left } \varphi \psi) \rangle$
 $\langle proof \rangle$

lemma *prenex-left-props-imp*:

assumes $\langle is\text{-prenex } \psi \rangle$
shows $\langle is\text{-prenex } \varphi \implies$
 $is\text{-prenex } (prenex\text{-left } \varphi \psi) \wedge$
 $FV (prenex\text{-left } \varphi \psi) = FV (\varphi \longrightarrow \psi) \wedge$
 $(language \{(prenex\text{-left } \varphi \psi)\} = language \{(\varphi \longrightarrow \psi)\}) \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. dom I \neq \{\} \longrightarrow (I, \beta \models prenex\text{-left } \varphi \psi \longleftrightarrow I, \beta \models \varphi$
 $\longrightarrow \psi)) \rangle$
(is $\langle is\text{-prenex } \varphi \implies ?P \varphi \rangle$
 $\langle proof \rangle$

lemma *prenex-left-props*:

$\langle is\text{-prenex } \varphi \wedge is\text{-prenex } \psi \implies$
 $is\text{-prenex } (prenex\text{-left } \varphi \psi) \wedge$
 $FV (prenex\text{-left } \varphi \psi) = FV (\varphi \longrightarrow \psi) \wedge$
 $(language \{(prenex\text{-left } \varphi \psi)\} = language \{(\varphi \longrightarrow \psi)\}) \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. dom I \neq \{\} \longrightarrow (I, \beta \models prenex\text{-left } \varphi \psi \longleftrightarrow I, \beta \models \varphi$
 $\longrightarrow \psi)) \rangle$
 $\langle proof \rangle$

theorem *prenex-props*: $\langle \text{is-prenex } (\text{prenex } \varphi) \wedge (FV (\text{prenex } \varphi) = FV \varphi) \wedge$
 $(\text{language } \{\text{prenex } \varphi\} = \text{language } \{\varphi\}) \wedge$
 $(\forall (I :: 'a \text{ intrp}) \beta. \text{dom } I \neq \{\} \implies (I, \beta \models (\text{prenex } \varphi)) \longleftrightarrow (I, \beta \models \varphi)) \rangle$
 $\langle \text{proof} \rangle$

corollary *is-prenex-prenex* [*simp*]: $\langle \text{is-prenex } (\text{prenex } \varphi) \rangle$
and *FV-prenex* [*simp*]: $\langle FV (\text{prenex } \varphi) = FV \varphi \rangle$
and *language-prenex* [*simp*]: $\langle \text{language } \{\text{prenex } \varphi\} = \text{language } \{\varphi\} \rangle$
 $\langle \text{proof} \rangle$

corollary *prenex-holds* [*simp*]: $\langle \text{dom } I \neq \{\} \implies (I, \beta \models (\text{prenex } \varphi)) \longleftrightarrow (I, \beta \models \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *prenex-satisfies* [*simp*]:
assumes $\text{dom } M \neq \{\}$
shows $\text{satisfies } M \{\text{prenex } \varphi\} \longleftrightarrow \text{satisfies } M \{\varphi\}$
 $\langle \text{proof} \rangle$

end

theory *Bumping*
imports
FOL-Semantics
HOL-Library.Nat-Bijection
begin

abbreviation *numpair where*
 $\langle \text{numpair } m \ n \equiv \text{prod-encode } (m, n) \rangle$

abbreviation *numfst where*
 $\langle \text{numfst } k \equiv \text{fst } (\text{prod-decode } k) \rangle$

abbreviation *numsnd where*
 $\langle \text{numsnd } k \equiv \text{snd } (\text{prod-decode } k) \rangle$

definition *bump-intrp* :: $'m \text{ intrp} \Rightarrow 'm \text{ intrp}$ **where**
 $\langle \text{bump-intrp } \mathcal{M} = \text{Abs-intrp } ((\text{dom } \mathcal{M}), (\lambda k \text{ zs. } (\text{intrp-fn } \mathcal{M}) (\text{numsnd } k) \text{ zs}),$
 $(\text{intrp-rel } \mathcal{M})) \rangle$

lemma *bump-dom* [*simp*]: $\langle \text{dom } (\text{bump-intrp } \mathcal{M}) = \text{dom } \mathcal{M} \rangle$
 $\langle \text{proof} \rangle$

lemma *bump-intrp-fn* [*simp*]: $\langle \text{intrp-fn } (\text{bump-intrp } \mathcal{M}) (\text{numpair } 0 \ f) \ ts = \text{intrp-fn } \mathcal{M} \ f \ ts \rangle$
 $\langle \text{proof} \rangle$

lemma *bump-intrp-rel [simp]*: $\langle \text{intrp-rel } (\text{bump-intrp } \mathcal{M}) \ n = \text{intrp-rel } \mathcal{M} \ n \rangle$
 $\langle \text{proof} \rangle$

fun *bump-nterm* :: *nterm* \Rightarrow *nterm* **where**
 $\langle \text{bump-nterm } (\text{Var } x) = \text{Var } x \rangle$
 $| \langle \text{bump-nterm } (\text{Fun } f \ ts) = \text{Fun } (\text{numpair } 0 \ f) \ (\text{map } \text{bump-nterm } \ ts) \rangle$

fun *bump-form* :: *form* \Rightarrow *form* **where**
 $\langle \text{bump-form } \perp = \perp \rangle$
 $| \langle \text{bump-form } (\text{Atom } p \ ts) = \text{Atom } p \ (\text{map } \text{bump-nterm } \ ts) \rangle$
 $| \langle \text{bump-form } (\varphi \longrightarrow \psi) = (\text{bump-form } \varphi) \longrightarrow (\text{bump-form } \psi) \rangle$
 $| \langle \text{bump-form } (\forall \ x. \ \varphi) = \forall \ x. \ (\text{bump-form } \varphi) \rangle$

lemma *bumpnterm*: $\langle \llbracket t \rrbracket^{\mathcal{M}, \beta} = \llbracket \text{bump-nterm } t \rrbracket^{\text{bump-intrp } \mathcal{M}, \beta} \rangle$
 $\langle \text{proof} \rangle$

lemma *bump-intrp-rel-holds*: $\langle (\text{map } (\lambda t. \llbracket t \rrbracket^{\mathcal{M}, \beta}) \ ts \in \text{intrp-rel } \mathcal{M} \ n) =$
 $(\text{map } ((\lambda t. \llbracket t \rrbracket^{\text{bump-intrp } \mathcal{M}, \beta}) \circ \text{bump-nterm}) \ ts \in \text{intrp-rel } (\text{bump-intrp } \mathcal{M}) \ n) \rangle$
 $\langle \text{proof} \rangle$

lemma *bumpform*: $\langle \mathcal{M}, \beta \models \varphi = (\text{bump-intrp } \mathcal{M}), \beta \models (\text{bump-form } \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *functions-form-bumpform*: $\langle (f, m) \in \text{functions-form } (\text{bump-form } \varphi) \implies$
 $\exists k. (f = \text{numpair } 0 \ k) \wedge (k, m) \in \text{functions-form } \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *bumpform-interpretation*: $\langle \text{is-interpretation } (\text{language } \{\varphi\}) \ \mathcal{M} \implies$
 $\text{is-interpretation } (\text{language } \{(\text{bump-form } \varphi)\}) \ (\text{bump-intrp } \mathcal{M}) \rangle$
 $\langle \text{proof} \rangle$

fun *unbump-nterm* :: *nterm* \Rightarrow *nterm* **where**
 $\langle \text{unbump-nterm } (\text{Var } x) = \text{Var } x \rangle$
 $| \langle \text{unbump-nterm } (\text{Fun } f \ ts) = \text{Fun } (\text{numsnd } f) \ (\text{map } \text{unbump-nterm } \ ts) \rangle$

fun *unbump-form* :: *form* \Rightarrow *form* **where**
 $\langle \text{unbump-form } \perp = \perp \rangle$
 $| \langle \text{unbump-form } (\text{Atom } p \ ts) = \text{Atom } p \ (\text{map } \text{unbump-nterm } \ ts) \rangle$
 $| \langle \text{unbump-form } (\varphi \longrightarrow \psi) = (\text{unbump-form } \varphi) \longrightarrow (\text{unbump-form } \psi) \rangle$
 $| \langle \text{unbump-form } (\forall \ x. \ \varphi) = \forall \ x. \ (\text{unbump-form } \varphi) \rangle$

lemma *unbumpnterm [simp]*: $\langle \text{unbump-nterm } (\text{bump-nterm } t) = t \rangle$
 $\langle \text{proof} \rangle$

lemma *unbumpform* [*simp*]: $\langle \text{unbump-form } (\text{bump-form } \varphi) = \varphi \rangle$
 $\langle \text{proof} \rangle$

definition *unbump-intrp* :: *'m intrp* \Rightarrow *'m intrp* **where**
 $\langle \text{unbump-intrp } \mathcal{M} = \text{Abs-intrp } (\text{dom } \mathcal{M}, (\lambda k \text{ zs. } (\text{intrp-fn } \mathcal{M}) (\text{numpair } 0 \ k) \ \text{zs}),$
 $(\text{intrp-rel } \mathcal{M})) \rangle$

lemma *unbump-term-intrp*: $\langle \llbracket \text{bump-nterm } t \rrbracket^{\mathcal{M}, \beta} = \llbracket t \rrbracket^{\text{unbump-intrp } \mathcal{M}, \beta} \rangle$
 $\langle \text{proof} \rangle$

lemma *unbump-holds*: $\langle (\mathcal{M}, \beta \models \text{bump-form } \varphi) = (\text{unbump-intrp } \mathcal{M}, \beta \models \varphi) \rangle$
 $\langle \text{proof} \rangle$

abbreviation *numlist* **where**
 $\langle \text{numlist } ns \equiv \text{list-encode } ns \rangle$

fun *num-of-term* :: *nterm* \Rightarrow *nat* **where**
 $\langle \text{num-of-term } (\text{Var } x) = \text{numpair } 0 \ x \rangle$
 $| \langle \text{num-of-term } (\text{Fun } f \ ts) = \text{numpair } 1 \ (\text{numpair } f \ (\text{numlist } (\text{map } \text{num-of-term } \ ts))) \rangle$

lemma *term-induct2*:
 $(\bigwedge x \ y. P (\text{Var } x) (\text{Var } y))$
 $\implies (\bigwedge x \ g \ us. P (\text{Var } x) (\text{Fun } g \ us))$
 $\implies (\bigwedge f \ ts \ y. P (\text{Fun } f \ ts) (\text{Var } y))$
 $\implies (\bigwedge f \ ts \ g \ us. (\bigwedge p \ q. p \in \text{set } ts \implies q \in \text{set } us \implies P \ p \ q) \implies P (\text{Fun } f \ ts)$
 $(\text{Fun } g \ us))$
 $\implies P \ t1 \ t2$
 $\langle \text{proof} \rangle$

lemma *num-of-term-inj*: $\langle \text{num-of-term } s = \text{num-of-term } t \iff s = t \rangle$
 $\langle \text{proof} \rangle$

fun *num-of-form* :: *form* \Rightarrow *nat* **where**
 $\langle \text{num-of-form } \perp = \text{numpair } 0 \ 0 \rangle$
 $| \langle \text{num-of-form } (\text{Atom } p \ ts) = \text{numpair } 1 \ (\text{numpair } p \ (\text{numlist } (\text{map } \text{num-of-term } \ ts))) \rangle$
 $| \langle \text{num-of-form } (\varphi \longrightarrow \psi) = \text{numpair } 2 \ (\text{numpair } (\text{num-of-form } \varphi) (\text{num-of-form } \psi)) \rangle$
 $| \langle \text{num-of-form } (\forall x. \varphi) = \text{numpair } 3 \ (\text{numpair } x \ (\text{num-of-form } \varphi)) \rangle$

lemma *numlist-num-of-term*: $\langle \text{numlist } (\text{map } \text{num-of-term } \ ts) = (\text{numlist } (\text{map } \text{num-of-term } \ us)) \equiv \text{ts} = \text{us} \rangle$

<proof>

lemma *num-of-form-inj*: $\langle \text{num-of-form } \varphi = \text{num-of-form } \psi \longleftrightarrow \varphi = \psi \rangle$
<proof>

consts *term-of-num* :: $\text{nat} \Rightarrow \text{nterm}$

specification (*term-of-num*) $\langle \forall n. \text{term-of-num } n = (\text{THE } t. \text{num-of-term } t = n) \rangle$
<proof>

lemma *term-of-num-of-term [simp]*: $\langle \text{term-of-num}(\text{num-of-term } t) = t \rangle$
<proof>

consts *form-of-num* :: $\text{nat} \Rightarrow \text{form}$

specification (*form-of-num*) $\langle \forall n. \text{form-of-num } n = (\text{THE } \varphi. \text{num-of-form } \varphi = n) \rangle$
<proof>

lemma *form-of-num-of-form [simp]*: $\langle \text{form-of-num } (\text{num-of-form } \varphi) = \varphi \rangle$
<proof>

end

theory *Skolem-Normal-Form*

imports

Prenex-Normal-Form

Bumping

begin

lemma *witness-imp-holds-exists*:

assumes $\langle \text{is-interpretation } (\text{functions-term } t, \text{preds}) (I :: (\text{nat}, \text{nat}) \text{ term intrp}) \rangle$

and

$\langle \text{is-valuation } I \beta \rangle$ **and**

$\langle I, \beta \models (\varphi \cdot_{fm} (\text{subst } x t)) \rangle$

shows $\langle I, \beta \models (\exists x. \varphi) \rangle$

<proof>

definition *skolem1* :: $\text{nat} \Rightarrow \text{nat} \Rightarrow \text{form} \Rightarrow \text{form}$ **where**

$\langle \text{skolem1 } f x \varphi = \varphi \cdot_{fm} (\text{subst } x (\text{Fun } f (\text{map } \text{Var } (\text{sorted-list-of-set } (\text{FV } (\exists x. \varphi)))))) \rangle$

lemma *fvt-var-x-simp*:

$\langle \text{FVT } (\text{Var } x \cdot \text{subst } x (\text{Fun } f (\text{map } \text{Var } (\text{sorted-list-of-set } (\text{FV } \varphi - \{x\})))) = \text{FV } \varphi - \{x\} \rangle$

<proof>

lemma *holds-indep-intrp-if2*:

fixes $I I' :: 'a \text{ intrp}$

shows

$\langle \llbracket I, \beta \models \varphi; \text{dom } I = \text{dom } I'; \bigwedge p. \text{intrp-rel } I p = \text{intrp-rel } I' p; \bigwedge f \text{ zs}. (f, \text{length } \text{zs}) \in \text{functions-form } \varphi \implies \text{intrp-fn } I f \text{ zs} = \text{intrp-fn } I' f \text{ zs} \rrbracket \implies I', \beta \models \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *fun-upds-prop*: $\langle \text{length } \text{xs} = \text{length } \text{zs} \implies \forall z \in \text{set } \text{zs}. P z \implies \forall v. P (g v) \implies \forall v. P ((\text{foldr } (\lambda kv f. \text{fun-upd } f (fst kv) (snd kv)) (\text{zip } \text{xs } \text{zs}) g) v) \rangle$
 $\langle \text{proof} \rangle$

lemma $\langle \{z. \exists y. y \in FV \varphi \wedge z \in \text{functions-term } (\text{Var } y \cdot \text{subst } x t)\} = \text{functions-term } t \vee \{z. \exists y. y \in FV \varphi \wedge z \in \text{functions-term } (\text{Var } y \cdot \text{subst } x t)\} = \{\} \rangle$
 $\langle \text{proof} \rangle$

lemma *func-form-subst*: $\langle x \in FV \varphi \implies (f, \text{length } \text{ts}) \in \text{functions-form } (\varphi \cdot_{fm} \text{subst } x (\text{Fun } f \text{ ts})) \rangle$
 $\langle \text{proof} \rangle$

lemma *holds-formsubst*:

$M, \beta \models (p \cdot_{fm} i) \longleftrightarrow M, (\lambda t. \llbracket t \rrbracket^{M, \beta}) \circ i \models p$
 $\langle \text{proof} \rangle$

lemma *holds-formsubst1*:

$M, \beta \models (p \cdot_{fm} \text{Var}(x:=t)) \longleftrightarrow M, \beta(x := \llbracket t \rrbracket^{M, \beta}) \models p$
 $\langle \text{proof} \rangle$

lemma *holds-formsubst2*:

$M, \beta \models (p \cdot_{fm} \text{subst } x t) \longleftrightarrow M, \beta(x := \llbracket t \rrbracket^{M, \beta}) \models p$
 $\langle \text{proof} \rangle$

lemma *size-nonzero [simp]*: $\text{size } fm > 0$

$\langle \text{proof} \rangle$

lemma

assumes *prenex-ex-phi*: $\langle \text{is-prenex } (\exists x. \varphi) \rangle$

and *notin-ff*: $\langle \neg (f, \text{card } (FV (\exists x. \varphi))) \in \text{functions-form } (\exists x. \varphi) \rangle$

shows *holds-skolem1a*: $\text{is-prenex } (\text{skolem1 } f x \varphi)$ (**is** ?A)

and *holds-skolem1b*: $FV (\text{skolem1 } f x \varphi) = FV (\exists x. \varphi)$ (**is** ?B)

and *holds-skolem1c*:

$\text{Prenex-Normal-Form.size } (\text{skolem1 } f x \varphi) < \text{Prenex-Normal-Form.size } (\exists x. \varphi)$

φ) (is ?C)
and holds-skolem1d: predicates-form (skolem1 f x φ) = predicates-form ($\exists x.$
 φ) (is ?D)
and holds-skolem1e: functions-form ($\exists x.$ φ) \subseteq functions-form (skolem1 f x
 φ) (is ?E)
and holds-skolem1f: functions-form (skolem1 f x φ)
 \subseteq (f, card (FV ($\exists x.$ φ))) \triangleright functions-form ($\exists x.$ φ) (is ?F)
 <proof>

definition define-fn $\equiv \lambda FN f n h. \lambda g zs. \text{if } g=f \wedge \text{length } zs = n \text{ then } h \text{ } zs \text{ else } FN$
 $g \text{ } zs$

lemma holds-skolem1g:

assumes prenex-ex-phi: $\langle \text{is-prenex } (\exists x. \varphi) \rangle$
and notin-ff: $\langle \neg (f, \text{card } (FV (\exists x. \varphi))) \in \text{functions-form } (\exists x. \varphi) \rangle$
fixes I :: 'a intrp
assumes interp-I: is-interpretation (language { φ }) I
and nempty-I: dom I \neq {}
and valid: $\bigwedge \beta. \text{is-valuation } I \beta \implies I, \beta \models (\exists x. \varphi)$
obtains M **where** dom M = dom I
 intrp-rel M = intrp-rel I
 $\bigwedge g zs. g \neq f \vee \text{length } zs \neq \text{card } (FV (\exists x. \varphi)) \implies \text{intrp-fn } M g zs$
 $= \text{intrp-fn } I g zs$
 is-interpretation (language {skolem1 f x φ }) M
 $\bigwedge \beta. \text{is-valuation } M \beta \implies M, \beta \models \text{skolem1 f x } \varphi$
 <proof>

lemma holds-skolem1h:

assumes prenex-ex-phi: $\langle \text{is-prenex } (\exists x. \varphi) \rangle$ **and** $\langle \neg (f, \text{card } (FV (\exists x. \varphi))) \in$
 functions-form ($\exists x. \varphi$) \rangle
assumes is-intrp: is-interpretation (language {skolem1 f x φ }) N
and nempty-N: dom N \neq {}
and is-val: is-valuation N β
and skol-holds: N, $\beta \models \text{skolem1 f x } \varphi$
shows N, $\beta \models (\exists x. \varphi)$
 <proof>

lemma holds-skolem1i:

assumes $\langle \text{is-prenex } (\exists x. \varphi) \rangle$ **and** $\langle \neg (f, \text{card } (FV (\exists x. \varphi))) \in \text{functions-form}$
 ($\exists x. \varphi$) \rangle
shows $\langle \text{is-prenex } (\text{skolem1 f x } \varphi) \wedge$
 $FV (\text{skolem1 f x } \varphi) = FV (\exists x. \varphi) \wedge$
 $\text{size } (\text{skolem1 f x } \varphi) < \text{size } (\exists x. \varphi) \wedge$
 $\text{predicates-form } (\text{skolem1 f x } \varphi) = \text{predicates-form } (\exists x. \varphi) \wedge$
 $\text{functions-form } (\exists x. \varphi) \subseteq \text{functions-form } (\text{skolem1 f x } \varphi) \wedge$
 $\text{functions-form } (\text{skolem1 f x } \varphi) \subseteq \text{insert } (f, \text{card } (FV (\exists x. \varphi))) (\text{functions-form}$
 ($\exists x. \varphi$)) \wedge
 $(\forall (I :: 'a \text{ intrp}). \text{is-interpretation } (\text{language } \{\varphi\}) I \wedge$
 $\neg (\text{dom } I = \{\}) \wedge$

$(\forall \beta. \text{is-valuation } I \beta \longrightarrow (I, \beta \models (\exists x. \varphi))) \longrightarrow$
 $(\exists (M :: 'a \text{ intrp}). \text{dom } M = \text{dom } I \wedge$
 $\text{intrp-rel } M = \text{intrp-rel } I \wedge$
 $(\forall g \text{ zs}. \neg g=f \vee \neg(\text{length } \text{zs} = \text{card } (FV (\exists x. \varphi)))) \longrightarrow \text{intrp-fn } M g \text{ zs} =$
 $\text{intrp-fn } I g \text{ zs}) \wedge$
 $\text{is-interpretation } (\text{language } \{\text{skolem1 } f x \varphi\}) M \wedge$
 $(\forall \beta. \text{is-valuation } M \beta \longrightarrow (M, \beta \models (\text{skolem1 } f x \varphi)))) \wedge$
 $(\forall (N :: 'a \text{ intrp}). \text{is-interpretation } (\text{language } \{\text{skolem1 } f x \varphi\}) N \wedge$
 $\neg (\text{dom } N = \{\}) \longrightarrow$
 $(\forall \beta. \text{is-valuation } N \beta \wedge (N, \beta \models (\text{skolem1 } f x \varphi)) \longrightarrow (N, \beta \models (\exists x. \varphi))))$
 \rangle
 $\langle \text{proof} \rangle$

lemma *skolems-ex*: $\langle \exists \text{skolems}. \forall \varphi. \text{skolems } \varphi = (\lambda k. \text{ppat } (\lambda x \psi. \forall x. (\text{skolems } \psi k)))$
 $(\lambda x \psi. \text{skolems } (\text{skolem1 } (\text{numpair } J k) x \psi) (\text{Suc } k)) (\lambda \psi. \psi) \varphi \rangle$
 $\langle \text{proof} \rangle$

consts *skolems* :: $\text{nat} \Rightarrow \text{form} \Rightarrow \text{nat} \Rightarrow \text{form}$

specification (*skolems*)

skolems-eq: $\langle \bigwedge J \psi k. \text{skolems } J \psi k$
 $= \text{ppat } (\lambda x \varphi'. \forall x. (\text{skolems } J \varphi' k)) (\lambda x \varphi'. \text{skolems } J (\text{skolem1}$
 $(\text{numpair } J k) x \varphi') (\text{Suc } k)) (\lambda \varphi. \varphi) \psi \rangle$
 $\langle \text{proof} \rangle$

bounding the possible Skolem functions in a given formula

definition *skolems-bounded* $\equiv \lambda p J k. \forall l m. (\text{numpair } J l, m) \in \text{functions-form } p$
 $\longrightarrow l < k$

lemma *skolems-bounded-mono*: $\llbracket \text{skolems-bounded } p J k'; k' \leq k \rrbracket \Longrightarrow \text{skolems-bounded } p J k$
 $\langle \text{proof} \rangle$

lemma *skolems-bounded-prenex*: $\text{skolems-bounded } \varphi K k \Longrightarrow \text{skolems-bounded } (\text{prenex } \varphi) K k$
 $\langle \text{proof} \rangle$

Basic properties proved by induction on the number of Skolemisation steps. Harrison's gigantic conjunction broken up for more manageable proofs, at the cost of some repetition

first, the simplest properties

lemma *holds-skolems-induction-A*:

assumes $\text{size } p = n$ **and** *is-prenex* p **and** *skolems-bounded* $p J k$

shows $\text{universal}(\text{skolems } J p k) \wedge$

$FV(\text{skolems } J p k) = FV p \wedge$

$\text{predicates-form } (\text{skolems } J p k) = \text{predicates-form } p \wedge$

$\text{functions-form } p \subseteq \text{functions-form } (\text{skolems } J p k) \wedge$

$$\text{functions-form (skolems } J p k) \subseteq \{(numpair J l, m) \mid l m. k \leq l\} \cup$$

$$\text{functions-form } p$$
 <proof>

the final conjunct of the HOL Light version

lemma *holds-skolems-induction-B:*

fixes $N :: 'a \text{ intrp}$

assumes $size\ p = n$ **and** *is-prenex* p **and** *skolems-bounded* $p\ J\ k$

and *is-interpretation* (language $\{skolems\ J\ p\ k\}$) $N\ dom\ N \neq \{\}$

and *is-valuation* $N\ \beta\ N, \beta \models skolems\ J\ p\ k$

shows $N, \beta \models p$

<proof>

the penultimate conjunct of the HOL Light version

lemma *holds-skolems-induction-C:*

fixes $M :: 'a \text{ intrp}$

assumes $size\ p = n$ **and** *is-prenex* p **and** *skolems-bounded* $p\ J\ k$

and *is-interpretation* (language $\{p\}$) $M\ dom\ M \neq \{\}$ *satisfies* $M\ \{p\}$

shows $\exists M'. dom\ M' = dom\ M \wedge intrp\text{-rel}\ M' = intrp\text{-rel}\ M \wedge$

$(\forall g\ zs. intrp\text{-fn}\ M'\ g\ zs \neq intrp\text{-fn}\ M\ g\ zs$

$\longrightarrow (\exists l. k \leq l \wedge g = numpair\ J\ l)) \wedge$

is-interpretation (language $\{skolems\ J\ p\ k\}$) $M' \wedge$

satisfies $M'\ \{skolems\ J\ p\ k\}$

<proof>

corollary *holds-skolems-prenex-A:*

assumes *is-prenex* φ *skolems-bounded* $\varphi\ K\ 0$

shows $universal(skolems\ K\ \varphi\ 0) \wedge (FV\ (skolems\ K\ \varphi\ 0) = FV\ \varphi) \wedge$

$predicates\text{-form}\ (skolems\ K\ \varphi\ 0) = predicates\text{-form}\ \varphi \wedge$

$functions\text{-form}\ \varphi \subseteq functions\text{-form}\ (skolems\ K\ \varphi\ 0) \wedge$

$functions\text{-form}\ (skolems\ K\ \varphi\ 0) \subseteq \{(numpair\ K\ l, m) \mid l\ m. True\} \cup$

$(functions\text{-form}\ \varphi)$

<proof>

corollary *holds-skolems-prenex-B:*

assumes *is-prenex* φ *skolems-bounded* $\varphi\ K\ 0$

and *is-interpretation* (language $\{skolems\ K\ \varphi\ 0\}$) $M\ dom\ M \neq \{\}$

and *is-valuation* $M\ \beta\ M, \beta \models skolems\ K\ \varphi\ 0$

shows $M, \beta \models \varphi$

<proof>

corollary *holds-skolems-prenex-C:*

assumes *is-prenex* φ *skolems-bounded* $\varphi\ K\ 0$

and *is-interpretation* (language $\{\varphi\}$) $M\ dom\ M \neq \{\}$ *satisfies* $M\ \{\varphi\}$

shows $\exists M'. dom\ M' = dom\ M \wedge intrp\text{-rel}\ M' = intrp\text{-rel}\ M \wedge$

$(\forall g\ zs. intrp\text{-fn}\ M'\ g\ zs \neq intrp\text{-fn}\ M\ g\ zs \longrightarrow (\exists l. g = numpair\ K\ l))$

\wedge

is-interpretation (language $\{skolems\ K\ \varphi\ 0\}$) $M' \wedge$

satisfies $M' \{ \text{skolems } K \ \varphi \ 0 \}$
 ⟨proof⟩

definition skopre where

⟨skopre $k \ \varphi = \text{skolems } k \ (\text{prenex } \varphi) \ 0 \rangle$

corollary skopre-model-A:

assumes skolems-bounded $\varphi \ K \ 0$

shows universal(skopre $K \ \varphi \) $\wedge \ (FV \ (\text{skopre } K \ \varphi) = FV \ \varphi) \ \wedge$$

predicates-form (skopre $K \ \varphi) = \text{predicates-form } \varphi \ \wedge$

functions-form $\varphi \subseteq \text{functions-form } (\text{skopre } K \ \varphi) \ \wedge$

functions-form (skopre $K \ \varphi) \subseteq \{ (\text{numpair } K \ l, m) \mid l \ m. \ \text{True} \} \cup (\text{functions-form}$

$\varphi)$

⟨proof⟩

corollary skopre-model-B:

assumes skolems-bounded $\varphi \ K \ 0$

and is-interpretation (language $\{ \text{skopre } K \ \varphi \}) \ M \ \text{dom } M \neq \{ \}$

and is-valuation $M \ \beta \ M, \beta \models \text{skopre } K \ \varphi$

shows $M, \beta \models \varphi$

⟨proof⟩

corollary skopre-model-C:

assumes skolems-bounded $\varphi \ K \ 0$

and is-interpretation (language $\{ \varphi \}) \ M \ \text{dom } M \neq \{ \}$ satisfies $M \ \{ \varphi \}$

shows $\exists M'. \ \text{dom } M' = \text{dom } M \ \wedge \ \text{intrp-rel } M' = \text{intrp-rel } M \ \wedge$

$(\forall g \ \text{zs}. \ \text{intrp-fn } M' \ g \ \text{zs} \neq \text{intrp-fn } M \ g \ \text{zs} \longrightarrow (\exists l. \ g = \text{numpair } K$

$l)) \ \wedge$

is-interpretation (language $\{ \text{skopre } K \ \varphi \}) \ M' \ \wedge$

satisfies $M' \ \{ \text{skopre } K \ \varphi \}$

⟨proof⟩

definition skolemize where

⟨skolemize $\varphi = \text{skopre } (\text{num-of-form } (\text{bump-form } \varphi) + 1) \ (\text{bump-form } \varphi) \rangle$

lemma no-skolems-bump-nterm:

shows $i > 0 \implies (\text{numpair } i \ l, \ m) \notin \text{functions-term } (\text{bump-nterm } t)$

⟨proof⟩

lemma no-skolems-bump-form: $i > 0 \implies \text{skolems-bounded } (\text{bump-form } \varphi) \ i \ 0$

⟨proof⟩

lemma universal-skolemize [iff]: universal (skolemize φ)

and FV -skolemize [simp]: $FV \ (\text{skolemize } \varphi) = FV \ (\text{bump-form } \varphi)$

and predicates-form-skolemize [simp]: predicates-form (skolemize φ) = predi-

ates-form (*bump-form* φ)
 ⟨*proof*⟩

lemma *functions-bump-form*: *functions-form* (*bump-form* φ) \subseteq *functions-form* (*skolemize* φ)
 ⟨*proof*⟩

lemma *functions-skolemize*:

functions-form (*skolemize* φ) \subseteq $\{(numpair\ (num-of-form\ (bump-form\ \varphi)+1)\ l,\ m) \mid k\ l\ m.\ True\} \cup$ *functions-form* (*bump-form* φ)
 ⟨*proof*⟩

lemma *skolemize-imp-holds-bump-form*:

assumes *is-interpretation* (*language* {*skolemize* φ }) $N\ dom\ N \neq \{\}$
and *is-valuation* $N\ \beta\ N,\beta \models skolemize\ \varphi$
shows $N,\beta \models bump-form\ \varphi$
 ⟨*proof*⟩

lemma *is-interpretation-skolemize*:

assumes *is-interpretation* (*language* {*bump-form* φ }) $M\ dom\ M \neq \{\}$ *satisfies* $M\ \{bump-form\ \varphi\}$
obtains M' **where** $dom\ M' = dom\ M\ intrp-rel\ M' = intrp-rel\ M$
 $\bigwedge g\ zs.\ intrp-fn\ M'\ g\ zs \neq intrp-fn\ M\ g\ zs \implies \exists l.\ g = numpair\ (num-of-form\ (bump-form\ \varphi) + 1)\ l$
is-interpretation (*language* {*skolemize* φ }) M' *satisfies* $M'\ \{skolemize\ \varphi\}$
 ⟨*proof*⟩

lemma *functions-form-skolemize*:

assumes $\langle f,m \rangle \in functions-form\ (skolemize\ \varphi)$
obtains k **where** $\langle f = numpair\ 0\ k \rangle \langle k,m \rangle \in functions-form\ \varphi \mid l$ **where** $\langle f = numpair\ (num-of-form\ (bump-form\ \varphi) + 1)\ l \rangle$
 ⟨*proof*⟩

definition *skomod1* **where**

$\langle skomod1\ \varphi\ M \equiv$
 if *satisfies* $M\ \{\varphi\}$
 then (*SOME* $M'.$ $dom\ M' = dom\ (bump-intrp\ M) \wedge$
 $intrp-rel\ M' = intrp-rel\ (bump-intrp\ M) \wedge$
 $(\forall g\ zs.\ intrp-fn\ M'\ g\ zs \neq intrp-fn\ (bump-intrp\ M)\ g\ zs \implies$
 $(\exists l.\ g = numpair\ (num-of-form\ (bump-form\ \varphi) + 1)\ l)) \wedge$
 is-interpretation (*language* {*skolemize* φ }) $M' \wedge$ *satisfies* M'
 {skolemize φ })
 else (*Abs-intrp* ($dom\ M,$ $(\lambda g\ zs.\ (SOME\ a.\ a \in dom\ M)),\ intrp-rel\ M))$)
 \rangle

lemma *skomod1-works*:

assumes M : $\langle \text{is-interpretation (language } \{\varphi\}) M \rangle \langle \text{dom } M \neq \{\} \rangle$
shows $\langle \text{dom (skomod1 } \varphi M) = \text{dom (bump-intrp } M) \wedge$
 $\text{intrp-rel (skomod1 } \varphi M) = \text{intrp-rel (bump-intrp } M) \wedge$
 $\text{is-interpretation (language } \{\text{skolemize } \varphi\}) (\text{skomod1 } \varphi M) \wedge$
 $(\text{satisfies } M \{\varphi\} \longrightarrow$
 $(\forall g \text{ zs. intrp-fn (skomod1 } \varphi M) g \text{ zs} \neq \text{intrp-fn (bump-intrp } M) g \text{ zs} \longrightarrow$
 $(\exists l. g = \text{numpair (num-of-form (bump-form } \varphi) + 1) l)) \wedge$
 $\text{satisfies (skomod1 } \varphi M) \{\text{skolemize } \varphi\}) \rangle$
 $\langle \text{proof} \rangle$

definition $\text{skomod-FN} \equiv \lambda M g \text{ zs. if numfst } g = 0 \text{ then intrp-fn } M (\text{numsnd } g) \text{ zs}$
 $\text{else intrp-fn (skomod1 (unbump-form (form-of-num$
 $(\text{numfst } g - 1))) M) g \text{ zs}$

definition skomod where
 $\langle \text{skomod } M \equiv \text{Abs-intrp (dom } M, \text{skomod-FN } M, \text{intrp-rel } M) \rangle$

lemma $\text{skomod-interpretation}$:
assumes $\langle \text{is-interpretation (language } \{\varphi\}) M \rangle \langle \text{dom } M \neq \{\} \rangle$
shows $\langle \text{is-interpretation (language } \{\text{skolemize } \varphi\}) (\text{skomod } M) \rangle$
 $\langle \text{proof} \rangle$

proposition skomod-works :
assumes $\langle \text{is-interpretation (language } \{\varphi\}) M \rangle \langle \text{dom } M \neq \{\} \rangle$
shows $\langle \text{satisfies } M \{\varphi\} \longleftrightarrow \text{satisfies (skomod } M) \{\text{skolemize } \varphi\} \rangle$
 $\langle \text{proof} \rangle$

proposition $\text{skolemize-satisfiable}$:
 $\langle (\exists M :: 'a \text{ intrp. dom } M \neq \{\} \wedge \text{is-interpretation (language } S) M \wedge$
 $\text{satisfies } M S) \longleftrightarrow$
 $(\exists M :: 'a \text{ intrp. dom } M \neq \{\} \wedge \text{is-interpretation (language (skolemize ' } S)) M$
 $\wedge \text{satisfies } M (\text{skolemize ' } S)) \rangle$ (**is** ?lhs = ?rhs)
 $\langle \text{proof} \rangle$

fun $\text{specialize} :: \text{form} \Rightarrow \text{form where}$
 $\langle \text{specialize } \perp = \perp \rangle$
 $| \langle \text{specialize (Atom } p \text{ ts)} = \text{Atom } p \text{ ts} \rangle$
 $| \langle \text{specialize } (\varphi \longrightarrow \psi) = \varphi \longrightarrow \psi \rangle$
 $| \langle \text{specialize } (\forall x. \varphi) = \text{specialize } \varphi \rangle$

lemma $\text{specialize-satisfies}$:
fixes $M :: 'a \text{ intrp}$

assumes $\langle \text{dom } M \neq \{\} \rangle$
shows $\langle \text{satisfies } M \text{ (specialize ' } S) \longleftrightarrow \text{satisfies } M S \rangle$
 $\langle \text{proof} \rangle$

lemma *specialize-qfree*: $\langle \text{universal } \varphi \implies \text{qfree (specialize } \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *functions-form-specialize* [simp]: $\text{functions-form}(\text{specialize } \varphi) = \text{functions-form } \varphi$
 $\langle \text{proof} \rangle$

lemma *predicates-form-form-specialize* [simp]: $\text{predicates-form}(\text{specialize } \varphi) = \text{predicates-form } \varphi$
 $\langle \text{proof} \rangle$

lemma *specialize-language*: $\langle \text{language (specialize ' } S) = \text{language } S \rangle$
 $\langle \text{proof} \rangle$

definition *skolem* :: $\text{form} \Rightarrow \text{form}$ **where**
 $\langle \text{skolem } \varphi = \text{specialize}(\text{skolemize } \varphi) \rangle$

lemma *skolem-qfree*: $\langle \text{qfree (skolem } \varphi) \rangle$
 $\langle \text{proof} \rangle$

theorem *skolem-satisfiable*:
 $\langle (\exists M :: 'a \text{ intrp. } \text{dom } M \neq \{\} \wedge \text{is-interpretation (language } S) M \wedge \text{satisfies } M S) \longleftrightarrow (\exists M :: 'a \text{ intrp. } \text{dom } M \neq \{\} \wedge \text{is-interpretation (language (skolem ' } S)) M \wedge \text{satisfies } M \text{ (skolem ' } S)) \rangle$
 $\langle \text{proof} \rangle$

end

theory *Canonical-Models*
imports *Skolem-Normal-Form*
begin

inductive-set *terms-set* :: $\langle (\text{nat} \times \text{nat}) \text{ set} \Rightarrow \text{nterm set} \rangle$ **for** *fns* :: $\langle (\text{nat} \times \text{nat}) \text{ set} \rangle$ **where**
 $\text{vars}: \langle (\text{Var } v) \in \text{terms-set fns} \rangle$
 $| \text{fn}: \langle (\text{Fun } f \text{ ts}) \in \text{terms-set fns} \rangle$
if $\langle (f, \text{length ts}) \in \text{fns} \rangle \langle \wedge t. t \in \text{set ts} \implies t \in \text{terms-set fns} \rangle$

lemma *struct-terms-set* [iff]: $\langle \text{struct (terms-set } A) \rangle$

⟨proof⟩

lemma *stupid-canondom*: $\langle t \in \text{terms-set } (\text{fst } \mathcal{L}) \implies \text{functions-term } t \subseteq (\text{fst } \mathcal{L}) \rangle$
⟨proof⟩

lemma *finite-subset-instance*: $\langle \text{finite } t' \implies t' \subseteq \{\varphi \cdot_{fm} \sigma \mid \sigma \varphi. P \sigma \wedge \varphi \in s\} \implies$
 $(\exists t. \text{finite } t \wedge t \subseteq s \wedge t' \subseteq \{\varphi \cdot_{fm} \sigma \mid \sigma \varphi. P \sigma \wedge \varphi \in t\}) \rangle$
⟨proof⟩

lemma *finite-subset-skolem*: $\langle \text{finite } u \implies u \subseteq \{\text{skolem } \varphi \mid \varphi. \varphi \in s\} \implies$
 $(\exists t. \text{finite } t \wedge t \subseteq s \wedge u = \{\text{skolem } \varphi \mid \varphi. \varphi \in t\}) \rangle$
⟨proof⟩

lemma *valuation-exists*: $\langle \neg (\text{dom } M = \{\}) \implies \exists \beta. \text{is-valuation } M \beta \rangle$
⟨proof⟩

lemma *holds-itlist-exists*:

$\langle (M, \beta \models (\text{foldr } (\lambda x p. \exists x. p) xs \varphi)) \iff$
 $(\exists as. \text{length } as = \text{length } xs \wedge \text{set } as \subseteq \text{dom } M \wedge$
 $(M, (\text{foldr } (\lambda u \beta. \beta(\text{fst } u := \text{snd } u)) (\text{rev } (\text{zip } xs as)) \beta) \models \varphi) \rangle$
⟨proof⟩

definition *canonical* :: $\langle (\text{nat} \times \text{nat}) \text{ set} \times (\text{nat} \times \text{nat}) \text{ set} \Rightarrow \text{nterm intrp} \Rightarrow \text{bool} \rangle$
where
 $\langle \text{canonical } \mathcal{L} \mathcal{M} \equiv (\text{dom } \mathcal{M} = \text{terms-set } (\text{fst } \mathcal{L}) \wedge (\forall f. \text{intrp-fn } \mathcal{M} f = \text{Fun } f)) \rangle$

definition *pintrp-of-intrp* :: $\langle 'm \text{ intrp} \Rightarrow (\text{nat} \Rightarrow 'm) \Rightarrow (\text{form} \Rightarrow \text{bool}) \rangle$ **where**
 $\langle \text{pintrp-of-intrp } \mathcal{M} \beta = (\lambda \varphi. \mathcal{M}, \beta \models \varphi) \rangle$

definition

canon-of-prop :: $\langle ((\text{nat} \times \text{nat}) \text{ set} \times (\text{nat} \times \text{nat}) \text{ set}) \Rightarrow (\text{form} \Rightarrow \text{bool}) \Rightarrow \text{nterm intrp} \rangle$ **where**
 $\langle \text{canon-of-prop } \mathcal{L} I \equiv \text{Abs-intrp } (\text{terms-set } (\text{fst } \mathcal{L}), \text{Fun}, (\lambda p. \{ts. I (\text{Atom } p ts)\})) \rangle$

lemma *dom-canon-of-prop* [simp]: $\langle \text{dom } (\text{canon-of-prop } \mathcal{L} I) = \text{terms-set } (\text{fst } \mathcal{L}) \rangle$
⟨proof⟩

lemma *intrp-fn-canon-of-prop* [simp]: $\langle \text{intrp-fn } (\text{canon-of-prop } \mathcal{L} I) = \text{Fun} \rangle$
⟨proof⟩

lemma *intrap-rel-canon-of-prop* [simp]: $\langle \text{intrap-rel } (\text{canon-of-prop } \mathcal{L} I) = (\lambda p. \{ts. I \text{ (Atom } p \text{ ts)}\}) \rangle$
 $\langle \text{proof} \rangle$

lemma *pholds-pintrap-of-intrap*:
 $\langle \text{qfree } \varphi \implies (\text{pintrap-of-intrap } \mathcal{M} \beta) \models_p \varphi \longleftrightarrow \mathcal{M}, \beta \models \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *intrap-of-canon-of-prop* [simp]:
 $\langle \text{pintrap-of-intrap } (\text{canon-of-prop } \mathcal{L} I) \text{ Var } (\text{Atom } p \text{ ts}) = I \text{ (Atom } p \text{ ts)} \rangle$
 $\langle \text{proof} \rangle$

lemma *holds-canon-of-prop*:
assumes $\langle \text{qfree } \varphi \rangle$ **shows** $\langle (\text{canon-of-prop } \mathcal{L} I), \text{Var} \models \varphi \longleftrightarrow I \models_p \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *holds-canon-of-prop-general*:
assumes $\langle \text{qfree } \varphi \rangle$ **shows** $\langle (\text{canon-of-prop } \mathcal{L} I), \beta \models \varphi \longleftrightarrow I \models_p (\varphi \cdot \text{fm } \beta) \rangle$
 $\langle \text{proof} \rangle$

lemma *canonical-canon-of-prop*: $\langle \text{canonical } \mathcal{L} (\text{canon-of-prop } \mathcal{L} I) \rangle$
 $\langle \text{proof} \rangle$

lemma *interpretation-canon-of-prop*: $\langle \text{is-interpretation } \mathcal{L} (\text{canon-of-prop } \mathcal{L} I) \rangle$
 $\langle \text{proof} \rangle$

lemma *prop-valid-imp-fol-valid*: $\langle \text{qfree } \varphi \wedge (\forall I. I \models_p \varphi) \implies (\forall \mathcal{M} \beta. \mathcal{M}, \beta \models \varphi) \rangle$
 $\langle \text{proof} \rangle$

lemma *fol-valid-imp-prop-valid*: $\langle \text{qfree } \varphi \wedge (\forall \mathcal{M} \beta. \text{canonical } (\text{language } \{\varphi\}) \mathcal{M} \longrightarrow \mathcal{M}, \beta \models \varphi) \implies \forall I. I \models_p \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma *satisfies-psatisfies*: $\langle \llbracket \varphi \in \Phi; \Phi \subseteq \{\varphi. \text{qfree } \varphi\}; \text{satisfies } \mathcal{M} \Phi; \text{is-valuation } \mathcal{M} \beta \rrbracket \implies \text{psatisfies } (\text{pintrap-of-intrap } \mathcal{M} \beta) \Phi \rangle$
 $\langle \text{proof} \rangle$

lemma *psatisfies-instances*:

assumes qf : $\langle \Phi \subseteq \{\varphi. qfree \varphi\} \rangle$

and ps : $\langle psatisfies I \{\varphi \cdot_{fm} \beta \mid \varphi \beta. (\forall x. \beta x \in terms-set (fst \mathcal{L})) \wedge \varphi \in \Phi\} \rangle$

shows $\langle satisfies (canon-of-prop \mathcal{L} I) \Phi \rangle$

$\langle proof \rangle$

lemma *satisfies-instances*:

assumes $\langle is-interpretation (language \Xi) \mathcal{M} \rangle$

shows $\langle satisfies \mathcal{M} \{\varphi \cdot_{fm} \sigma \mid \varphi \sigma. \varphi \in \Phi \wedge (\forall x. \sigma x \in terms-set (fst (language \Xi)))\} \longleftrightarrow$

$satisfies \mathcal{M} \Phi \rangle$

$\langle proof \rangle$

lemma *compact-canon-qfree*:

assumes qf : $\langle \Phi \subseteq \{\varphi. qfree \varphi\} \rangle$

and int : $\langle \bigwedge \Psi. \llbracket finite \Psi; \Psi \subseteq \Phi \rrbracket$

$\implies \exists \mathcal{M}::'a \ intrp. is-interpretation (language \Xi) \mathcal{M} \wedge dom \mathcal{M} \neq \{\} \wedge$

$satisfies \mathcal{M} \Psi \rangle$

obtains \mathcal{C} **where** $\langle is-interpretation (language \Xi) \mathcal{C} \rangle \langle canonical (language \Xi) \mathcal{C} \rangle$

$\langle satisfies \mathcal{C} \Phi \rangle$

$\langle proof \rangle$

lemma *interpretation-restrictlanguage*:

$\langle \Psi \subseteq \Phi \implies is-interpretation (language \Phi) \mathcal{M} \implies is-interpretation (language \Psi) \mathcal{M} \rangle$

$\langle proof \rangle$

lemma *interpretation-extendlanguage*:

fixes $\mathcal{M}::'a \ intrp \rangle$

assumes int : $\langle is-interpretation (language \Psi) \mathcal{M} \rangle$ **and** $\langle dom \mathcal{M} \neq \{\} \rangle$

and $\langle satisfies \mathcal{M} \Psi \rangle$

obtains \mathcal{N} **where** $\langle dom \mathcal{N} = dom \mathcal{M} \rangle \langle intrp-rel \mathcal{N} = intrp-rel \mathcal{M} \rangle$

$\langle is-interpretation (language \Phi) \mathcal{N} \rangle \langle satisfies \mathcal{N} \Psi \rangle$

$\langle proof \rangle$

theorem *compact-ls*:

assumes $\langle \bigwedge \Psi. \llbracket finite \Psi; \Psi \subseteq \Phi \rrbracket$

$\implies \exists \mathcal{M}::'a \ intrp. is-interpretation (language \Phi) \mathcal{M} \wedge dom \mathcal{M} \neq \{\} \wedge$

$satisfies \mathcal{M} \Psi \rangle$

obtains $\mathcal{C}::\langle nterm \ intrp \rangle$ **where** $\langle is-interpretation (language \Phi) \mathcal{C} \rangle \langle dom \mathcal{C} \neq \{\} \rangle \langle satisfies \mathcal{C} \Phi \rangle$

$\langle proof \rangle$

lemma canon:

assumes $\langle \text{is-interpretation (language } \Phi) \mathcal{M} \rangle \langle \text{dom } \mathcal{M} \neq \{\} \rangle \langle \text{satisfies } \mathcal{M} \Phi \rangle$
obtains $\mathcal{C} :: \langle \text{nterm intrp} \rangle$ **where** $\langle \text{is-interpretation (language } \Phi) \mathcal{C} \rangle \langle \text{dom } \mathcal{C} \neq \{\} \rangle \langle \text{satisfies } \mathcal{C} \Phi \rangle$
 $\langle \text{proof} \rangle$

definition lowmod $:: \langle \text{nterm intrp} \Rightarrow \text{nat intrp} \rangle$ **where**

$\langle \text{lowmod } \mathcal{M} \equiv \text{Abs-intrp (num-of-term } \langle \text{dom } \mathcal{M} \rangle,$
 $(\lambda g \text{ ns. num-of-term (intrp-fn } \mathcal{M} \text{ } g \text{ (map term-of-num ns))),}$
 $(\lambda p. \{ \text{ns. (map term-of-num ns) } \in \text{intrp-rel } \mathcal{M} \text{ } p \}) \rangle$

lemma dom-lowmod [simp]: $\langle \text{dom (lowmod } \mathcal{M}) = \text{num-of-term } \langle \text{dom } \mathcal{M} \rangle \rangle$
 $\langle \text{proof} \rangle$

lemma intrp-fn-lowmod [simp]: $\langle \text{intrp-fn (lowmod } \mathcal{M}) \text{ } f \text{ ns} = \text{num-of-term (intrp-fn } \mathcal{M} \text{ } f \text{ (map term-of-num ns))} \rangle$
 $\langle \text{proof} \rangle$

lemma intrp-rel-lowmod [simp]: $\langle \text{intrp-rel (lowmod } \mathcal{M}) \text{ } p = \{ \text{ns. (map term-of-num ns) } \in \text{intrp-rel } \mathcal{M} \text{ } p \} \rangle$
 $\langle \text{proof} \rangle$

lemma is-valuation-lowmod: $\langle \text{is-valuation (lowmod } \mathcal{C}) \text{ (num-of-term } \circ \beta) \longleftrightarrow \text{is-valuation } \mathcal{C} \text{ } \beta \rangle$
 $\langle \text{proof} \rangle$

lemma lowmod-dom-empty: $\langle \text{dom (lowmod } \mathcal{M}) = \{\} \longleftrightarrow \text{dom } \mathcal{M} = \{\} \rangle$
 $\langle \text{proof} \rangle$

lemma lowmod-termval:

assumes $\langle \text{is-valuation (lowmod } \mathcal{M}) \beta \rangle$
shows $\langle \text{eval } t \text{ (lowmod } \mathcal{M}) \beta = \text{num-of-term (eval } t \text{ } \mathcal{M} \text{ (term-of-num } \circ \beta)) \rangle$
 $\langle \text{proof} \rangle$

lemma lowmod-holds:

assumes $\langle \text{is-valuation (lowmod } \mathcal{M}) \beta \rangle$
shows $\langle (\text{lowmod } \mathcal{M}), \beta \models \varphi \longleftrightarrow \mathcal{M}, (\text{term-of-num } \circ \beta) \models \varphi \rangle$
 $\langle \text{proof} \rangle$

lemma lowmod-intrp: $\langle \text{is-interpretation } \mathcal{L} \text{ (lowmod } \mathcal{M}) = \text{is-interpretation } \mathcal{L} \text{ } \mathcal{M} \rangle$
 $\langle \text{proof} \rangle$

lemma *loewenheim-skolem*:

assumes $\langle \text{is-interpretation (language } \Phi) \mathcal{M} \rangle \langle \text{dom } \mathcal{M} \neq \{\} \rangle$
assumes $\langle \bigwedge \varphi. \varphi \in \Phi \implies \text{qfree } \varphi \rangle \langle \text{satisfies } \mathcal{M} \Phi \rangle$
obtains $\mathcal{N} :: \langle \text{nat intrp} \rangle$ **where** $\langle \text{is-interpretation (language } \Phi) \mathcal{N} \rangle \langle \text{dom } \mathcal{N} \neq \{\} \rangle \langle \text{satisfies } \mathcal{N} \Phi \rangle$
 $\langle \text{proof} \rangle$

theorem *uniformity*:

assumes $\langle \text{qfree } \varphi \rangle$
 $\langle \bigwedge \mathcal{C} :: \text{nterm intrp. } \bigwedge \beta. [\text{dom } \mathcal{C} \neq \{\}; \text{ is-valuation } \mathcal{C} \beta] \implies \mathcal{C}, \beta \models \text{foldr Exists } xs \varphi \rangle$
obtains σs **where** $\langle \bigwedge \sigma x. \sigma \in \text{set } \sigma s \implies \sigma x \in \text{terms-set (fst (language } \{\varphi\}) \rangle$
 $\langle \bigwedge I. I \models_p (\text{foldr } (\lambda \varphi \psi. \varphi \vee \psi) (\text{map } (\lambda \sigma. \varphi \cdot_{fm} \sigma) \sigma s) \perp) \rangle$
 $\langle \text{proof} \rangle$

end

References

- [1] J. Harrison. Formalizing basic first order model theory. In *TPHOLs*, volume 1479 of *Lecture Notes in Computer Science*, pages 153–170. Springer, 1998.