First-Order Modal Logic in HOL:

Deep and Shallow Embeddings with Automated Faithfulness

Christoph Benzmüller (and Daniel Kirchner)

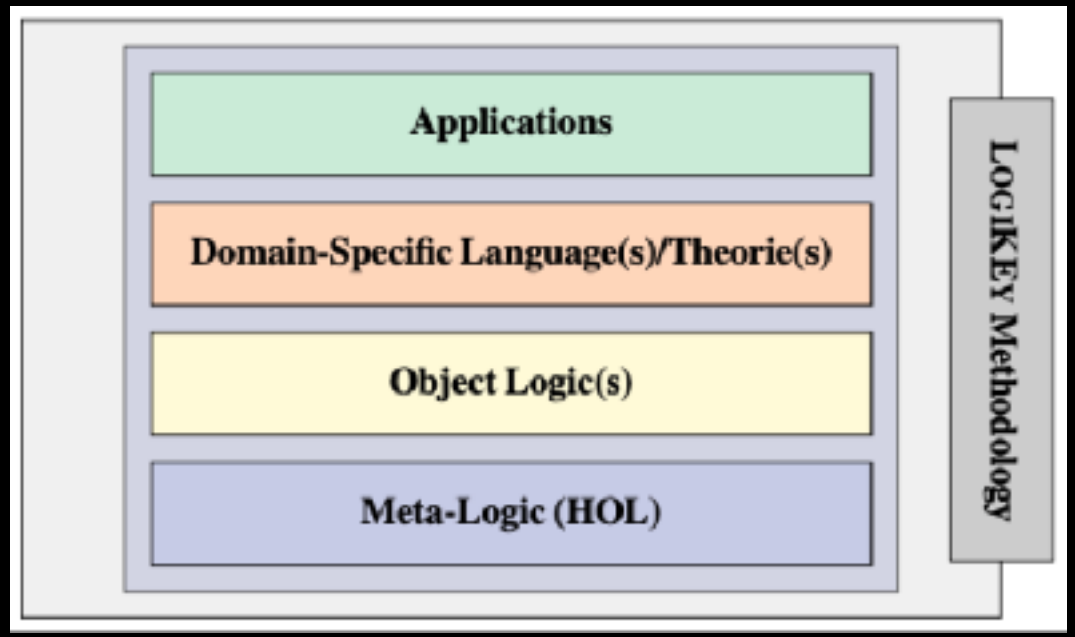
U Bamberg | FU Berlin

arXiv:2607.10880
paper + Isabelle sources

ARQNL 2026, IJCAR / FLoC, Lisbon, 24.7.2026

The bigger context: LogiKEy

- LogiKEy — Logic-pluralistic KR&R framework developed for more than a decade (going back to jww Larry Paulson; term LogiKEy going back to jww Xavier Parent and Leon van der Torre)
- So far primarily (minimal) shallow embeddings: the object logic as a natural fragment of HOL — good automation
- Recent work (CADE-30, 2025 — propositional): connecting shallow to deep embeddings, with automated faithfulness
- This talk: **quantified case** — FML with constant domains
  - Binders, substitution, and a Löwenheim–Skolem argument against uncountable domains



LogiKEy layered architecture

PML in HOL
CADE-30, 2025

→

FML in HOL
this talk

Deep vs. shallow: complementary strengths

	Deep embedding	Shallow embedding
Formulae are …	an **inductive datatype** — syntax as first-class data	**HOL terms** (e.g. via the standard translation)
Relation to HOL	**axiomatises the logic** language inside HOL (datatype + recursive truth predicate)	exhibits the logic as a **natural fragment of HOL** — (essentially) no axioms needed
Good for …	**reasoning about the language**: induction over syntax, substitution, metatheorems (e.g. Löwenheim–Skolem)	**automated reasoning in/using the language**, in particular, **schematic** problem formulations: sledgehammer proofs, nitpick countermodels
Semantics	**explicit:** defined by primitive recursion	**implicit:** inherited from HOL's own semantics

Maximal shallow = the bridge:
All semantic dependencies explicitly encoded (as in deep), yet formulae are HOL terms (as in shallow)

FML with constant-domain Kripke semantics

$$\phi, \psi \;:=\; r(x, y) \mid \neg\phi \mid \phi \wedge \psi \mid \exists x.\, \phi \mid \Diamond\phi \qquad (\vee, \supset, \forall, \Box \text{ defined as usual})$$

Model ⟨**W, R, I, D**⟩**:** worlds W, accessibility R, world-indexed interpretation I, domain D

Assignment g: V → D

$$\langle W,R,I,D\rangle, g, w \vDash r(x,y) \;\text{ iff }\; I(r)(w)(g(x), g(y))$$

$$\langle W,R,I,D\rangle, g, w \vDash \Diamond\phi \;\text{ iff }\; \exists \mathbf{v} \in \mathbf{W}.\; R(w,v) \;\wedge\; \langle W,R,I,D\rangle, g, \mathbf{v} \vDash \phi$$

$$\langle W,R,I,D\rangle, \mathbf{g}, w \vDash \exists x.\, \phi \;\text{ iff }\; \exists \mathbf{d} \in \mathbf{D}.\; \langle W,R,I,D\rangle, \mathbf{g[x{\leftarrow}d]}, w \vDash \phi$$

(¬, ∧ as expected; validity closes over all models, g into D, w ∈ W)

- Only binary relations → n-ary definable
- Constant domain → Barcan and converse Barcan valid
- Rigid designators: g world-independent
- Basic FML of Bräuner & Ghilardi; ∃/◇ primitive

Preliminaries

Types

Variable assignments

Reflexitivity, symmetry and transitivity

Notation for bounded quantifiers
$\exists w{:}W.\ \phi$

Further useful notions

```
theory MinFMLinHOL_preliminaries imports Main
begin
—‹Some global settings› declare[[syntax_ambiguity_warning=false]] nitpick_params[user_axioms,expect=genuine]
—‹Declarations shared between the deep, maximal-shallow, and minimal-shallow embeddings of FML in HOL.›
typedecl D                    —‹Domain of individuals (constant across worlds).›
typedecl w                    —‹Type of possible worlds.›
type_synonym R = nat          —‹Binary relation symbols (encoded as naturals).›
type_synonym V = nat          —‹Individual variable symbols (encoded as naturals).›
type_synonym 𝓡 = "D⇒D⇒bool"   —‹Binary relations on individuals.›
type_synonym 𝒲 = "w⇒bool"     —‹Sets of possible worlds (the universe).›
type_synonym 𝒜 = "w⇒w⇒bool"   —‹Accessibility relations between worlds.›
type_synonym ℐ = "R⇒w⇒𝓡"      —‹World-indexed interpretations of relation symbols.›
type_synonym ℰ = "V⇒D"        —‹Variable assignment functions.›
type_synonym 𝒟 = "D⇒bool"     —‹Domain restrictions (subsets of D as predicates).›
—‹Pointwise update of variable assignments.›
definition EnvMod::"ℰ⇒V⇒D⇒ℰ" ("_[_ ← _]" [110,0,0] 110) where "g[x←d] ≡ λz. if z = x then d else g z"
—‹Standard lemmas about variable-assignment update.›
lemma L1 [simp]: "x ≠ y ⟹ (g[y←d]) x = g x"                    by (simp add: EnvMod_def)
lemma L2:        "a ≠ c ⟹ g[a←d₁][c←d₂] = g[c←d₂][a←d₁]"       by (auto simp: EnvMod_def)
lemma L3 [simp]: "(g[a←d]) a = d"                               by (simp add: EnvMod_def)
lemma L4 [simp]: "g[a←d₁][a←d₂] = g[a←d₂]"                      by (auto simp: EnvMod_def)
—‹Standard predicates on accessibility relations (carried over from the PML preliminaries).›
abbreviation "reflexive  ≡ λR::𝒜. ∀x. R x x"
abbreviation "symmetric  ≡ λR::𝒜. ∀x y. R x y ⟶ R y x"
abbreviation "transitive ≡ λR::𝒜. ∀x y z. R x y ∧ R y z ⟶ R x z"
—‹Bounded quantifiers: ‹∀x:W. φ› stands for ‹∀x. W x ⟶ φ x› and ‹∃x:W. φ› stands for ‹∃x. W x ∧ φ x›.›
abbreviation (input) "BAll W φ ≡ ∀x. W x ⟶ φ x"
syntax "BAll"::"pttrn⇒logic⇒bool⇒bool" ("(∀(_:_)./ _)" [0,0,10]10)
translations "∀x:W. φ" ⇌ "CONST BAll W (λx. φ)"
abbreviation (input) "BEx W φ ≡ ∃x. W x ∧ φ x"
syntax "BEx"::"pttrn⇒logic⇒bool⇒bool" ("(∃(_:_)./ _)" [0,0,10]10)
translations "∃x:W. φ" ⇌ "CONST BEx W (λx. φ)"
—‹Set-as-predicate operations, range, and the universal domain.›
abbreviation (input) "Into"     (infix "into" 100)   where " f into D ≡ ∀x. D (f x)"
abbreviation (input) "Range"                         where "Range f ≡ λx. ∃y. x = f y"
abbreviation (input) "Onto"     (infix "onto" 100)   where "f onto D ≡ D = Range f"
abbreviation (input) "Subset"   (infix "⊆" 100)      where "A ⊆ B ≡ ∀x. A x ⟶ B x"
abbreviation (input) "Union"    (infixl "∪" 110)     where "A ∪ B ≡ λx. A x ∨ B x"
abbreviation (input) "Univ"                          where "Univ ≡ λx. True"
—‹Backward implication, useful for stating equivalences in two directions.›
abbreviation (input) Bimp (infixr "⟵" 50) where "φ ⟵ ψ ≡ ψ ⟶ φ"
end
```

Deep embedding

```
theory MinFMLinHOL_preliminaries imports Main
begin
—‹Some global settings› declare[[syntax_ambiguity_warning=false]] nitpick_params[user_axioms,expect=genuine]
—‹Declarations shared between the deep, maximal-shallow, and minimal-shallow embeddings of FML in HOL.›
typedecl D                                  ‹Domain of individuals (constant across worlds)›
```

```
theory MinFMLinHOL_deep imports MinFMLinHOL_preliminaries
begin
—‹Deep embedding of (minimal) FML in HOL with constant-domain relational semantics.›
datatype 𝓕 = AtmD R V V ("_ᵈ(_,_)") | NegD 𝓕 ("¬ᵈ _" [58]59) | AndD 𝓕 𝓕 (infixr "∧ᵈ" 56)
           | ExD V 𝓕 ("∃ᵈ_. _") | DiaD 𝓕 ("◇ᵈ _" [58]59)
—‹Further connectives as derived definitions.›
definition OrD  (infixr "∨ᵈ" 54)        where "φ ∨ᵈ ψ ≡ ¬ᵈ(¬ᵈφ ∧ᵈ ¬ᵈψ)"
definition ImpD (infixr "⊃ᵈ" 55)        where "φ ⊃ᵈ ψ ≡ ¬ᵈφ ∨ᵈ ψ"
definition AllD ("∀ᵈ_. _")              where "∀ᵈx. φ ≡ ¬ᵈ(∃ᵈx. ¬ᵈφ)"
definition BoxD ("□ᵈ _" [58]59)         where "□ᵈφ ≡ ¬ᵈ(◇ᵈ(¬ᵈφ))"
—‹Relative truth in a model. Existential ranges over D; box ranges over R-successors in W.›
primrec RelativeTruthD::"𝒲⇒𝒜⇒ℐ⇒𝒟⇒ℰ⇒w⇒𝓕⇒bool" ("⟨_,_,_,_⟩,_,_ ⊨ᵈ _" [10,10,10,10,10,10] 100) where
  "⟨W,R,I,D⟩,g,w ⊨ᵈ (rᵈ(x,y))  = I r w (g x) (g y)"
| "⟨W,R,I,D⟩,g,w ⊨ᵈ (¬ᵈφ)     = (¬ ⟨W,R,I,D⟩,g,w ⊨ᵈ φ)"
| "⟨W,R,I,D⟩,g,w ⊨ᵈ (φ ∧ᵈ ψ)  = (⟨W,R,I,D⟩,g,w ⊨ᵈ φ ∧ ⟨W,R,I,D⟩,g,w ⊨ᵈ ψ)"
| "⟨W,R,I,D⟩,g,w ⊨ᵈ (∃ᵈx. φ)  = (∃d:D. ⟨W,R,I,D⟩,g[x := d],w ⊨ᵈ φ)"
| "⟨W,R,I,D⟩,g,w ⊨ᵈ (◇ᵈφ)     = (∃v:W. R w v ∧ ⟨W,R,I,D⟩,g,v ⊨ᵈ φ)"
—‹Validity: true in every model, every domain-respecting assignment, and every world of the universe.›
definition ValD ("⊨ᵈ _" 9)      where "⊨ᵈ φ ≡ ∀W R I D g. g into D ⟶ (∀w:W. ⟨W,R,I,D⟩,g,w ⊨ᵈ φ)"
—‹Auxiliary "full-domain" notion of validity: restricts to assignments ranging over Univ of D.›
definition ValD' ("⊨ᵈ' _" 9)    where "⊨ᵈ' φ = ∀ R I g. ∀w. ⟨Univ,R,I,Univ⟩,g,w ⊨ᵈ φ"
—‹General validity implies full-domain validity.›
lemma Val: "⊨ᵈ φ ⟹ ⊨ᵈ' φ"  using ValD'_def ValD_def by simp
—‹Adding a nonemptiness premise on the world universe leaves validity unchanged.›
lemma ValD_nonemptyW: "(⊨ᵈ φ) = (∀W R I D g. (∃w. W w) ⟶ g into D ⟶ (∀w:W. ⟨W,R,I,D⟩,g,w ⊨ᵈ φ))"
  by (smt (verit, ccfv_threshold) ValD_def)
—‹Bag of definitions for collective unfolding.›
named_theorems DefD lemmas [DefD] = OrD_def ImpD_def AllD_def BoxD_def ValD_def ValD'_def
end
```

How ValD and ValD′ differ

⊨ᵈφ: truth in every domain D, with assignments g ranging into D → arbitrary / varying domain.

⊨ᵈ′φ: domain fixed to the full universe Univ, g unrestricted → just the single case D = Univ.

Lemma Val: ⊨ᵈφ ⟹ ⊨ᵈ′φ
(but: ⊨ᵈ′φ ⇏ ⊨ᵈφ)

Maximal shallow embedding

```
theory MinFMLinHOL_shallow imports MinFMLinHOL_preliminaries
begin
—‹Maximal (heavyweight) shallow embedding of FML in HOL; FML formulas are HOL terms of the following type σ.›
 type_synonym σ = "𝒲⇒𝒜⇒ℐ⇒𝒟⇒ℰ⇒w⇒bool"
—‹The five primitive cases.›
 definition AtmS :: "R⇒V⇒V⇒σ"  ("_ˢ'(_,_')")       where  "rˢ(x,y) ≡ λW R I D g w. I r w (g x) (g y)"
 definition NegS :: "σ⇒σ"       ("¬ˢ _" [58]59)     where  "¬ˢφ    ≡ λW R I D g w. ¬(φ W R I D g w)"
 definition AndS :: "σ⇒σ⇒σ"     (infixr "∧ˢ" 56)    where  "φ ∧ˢ ψ ≡ λW R I D g w. φ W R I D g w ∧ ψ W R I D g w"
 definition ExS  :: "V⇒σ⇒σ"     ("∃ˢ_. _" 53)       where  "∃ˢx. φ ≡ λW R I D g w. ∃d:D. φ W R I D (g[x←d]) w"
 definition DiaS :: "σ⇒σ"       ("◇ˢ _" [58]59)     where  "◇ˢφ    ≡ λW R I D g w. ∃v:W. R w v ∧ φ W R I D g v"
—‹Derived connectives.›
 definition OrS  :: "σ⇒σ⇒σ"     (infixr "∨ˢ" 54)    where  "φ ∨ˢ ψ ≡ ¬ˢ(¬ˢφ ∧ˢ ¬ˢψ)"
 definition ImpS :: "σ⇒σ⇒σ"     (infixr "⊃ˢ" 55)    where  "φ ⊃ˢ ψ ≡ ¬ˢφ ∨ˢ ψ"
 definition AllS :: "V⇒σ⇒σ"     ("∀ˢ_. _" 53)       where  "∀ˢx. φ ≡ ¬ˢ(∃ˢx. ¬ˢφ)"
 definition BoxS :: "σ⇒σ"       ("□ˢ _" [58]59)     where  "□ˢφ    ≡ ¬ˢ(◇ˢ(¬ˢφ))"
—‹Relative truth and validity (mirroring the deep embedding).›
 definition RelTruthS :: "𝒲⇒𝒜⇒ℐ⇒𝒟⇒ℰ⇒w⇒σ⇒bool" ("(_,_,_,_),_,_ ⊨ˢ _" [100,0,0,0,0,100] 100)
                                 where "(W,R,I,D),g,w ⊨ˢ φ ≡ φ W R I D g w"
 definition ValS :: "σ⇒bool" ("⊨ˢ _" 9) where  "⊨ˢ φ ≡ ∀W R I D g. g into D ⟶ (∀w:W. (W,R,I,D),g,w ⊨ˢ φ)"
—‹Bag of definitions.›
 named_theorems DefS lemmas DefS_defs [DefS] = AtmS_def NegS_def AndS_def ExS_def BoxS_def OrS_def
   ImpS_def AllS_def DiaS_def RelTruthS_def ValS_def
end
```

Minimal shallow embedding as a locale

```
theory MinFMLinHOL_shallow_minimal_locale imports MinFMLinHOL_preliminaries
begin
—‹Minimal (lightweight) shallow embedding of FML in HOL, packaged as a locale.›
 locale MinS = fixes RR :: 𝒜 and II :: ℐ and gg :: ℰ and WW::𝒲
 begin
—‹Formulas as HOL terms of type σ = w⇒bool: only the dependency on the world is explicit.›
  type_synonym σ = "w⇒bool"
—‹Five primitive cases.  ExM, AllM are HOL binders over V = nat; atoms consult II via gg.›
  definition AtmM :: "R⇒V⇒V⇒σ"  ("_ᵐ'(_,_')")      where  "rᵐ(x,y) ≡ λw. II r w (gg x) (gg y)"
  definition NegM :: "σ⇒σ"      ("¬ᵐ _" [58]59)     where  "¬ᵐφ ≡ λw. ¬(φ w)"
  definition AndM :: "σ⇒σ⇒σ"    (infixr "∧ᵐ" 56)    where  "φ ∧ᵐ ψ ≡ λw. φ w ∧ ψ w"
  definition ExM  :: "(V⇒σ)⇒σ"  (binder "∃ᵐ" 53)    where  "∃ᵐd. Φ d ≡ λw. ∃d. Φ d w"
  definition DiaM :: "υ⇒υ"      ("◇ᵐ _" [58]59)     where  "◇ᵐφ ≡ λw. ∃v:WW. RR w v ∧ φ v"
—‹Derived connectives.›
  definition OrM  :: "σ⇒σ⇒σ"    (infixr "∨ᵐ" 54)    where  "φ ∨ᵐ ψ ≡ ¬ᵐ(¬ᵐφ ∧ᵐ ¬ᵐψ)"
  definition ImpM :: "σ⇒σ⇒σ"    (infixr "⊃ᵐ" 55)    where  "φ ⊃ᵐ ψ ≡ ¬ᵐφ ∨ᵐ ψ"
  definition AllM :: "(V⇒σ)⇒σ"  (binder "∀ᵐ" 53)    where  "∀ᵐd. Φ d ≡ λw. ∀d. Φ d w"
  definition BoxM :: "σ⇒σ"      ("□ᵐ _" [58]59)     where  "□ᵐφ ≡ ¬ᵐ(◇ᵐ(¬ᵐφ))"
—‹Relative truth and validity, with worlds restricted to the universe WW.›
  definition RelativeTruthM :: "w⇒σ⇒bool" ("_ ⊨ᵐ _") where  "w ⊨ᵐ φ ≡ φ w"
  definition ValM :: "σ⇒bool"   ("⊨ᵐ _" 9)          where  "⊨ᵐ φ ≡ ∀w:WW. w ⊨ᵐ φ"
—‹Bag of definitions.›
  named_theorems DefM lemmas DefM_defs [DefM] = AtmM_def NegM_def AndM_def ExM_def BoxM_def OrM_def
    ImpM_def AllM_def DiaM_def RelativeTruthM_def ValM_def
 end
end
```

Three interlinked embeddings: deep & max/min shallow

Deep $(\cdot)^d$

datatype $\mathcal{F}$ of formulae

truth by primitive recursion

→ metalogical reasoning

Maximal shallow $(\cdot)^s$

$\sigma = \mathcal{W}\Rightarrow\mathcal{A}\Rightarrow\mathcal{I}\Rightarrow\mathcal{D}\Rightarrow\mathcal{E}\Rightarrow w\Rightarrow bool$

every dependency a λ-abstraction

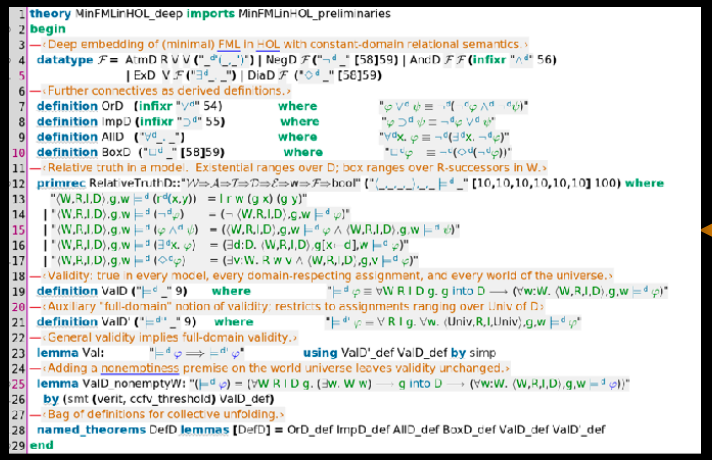
→ standard translation, heavyweight

Minimal shallow $(\cdot)^m$

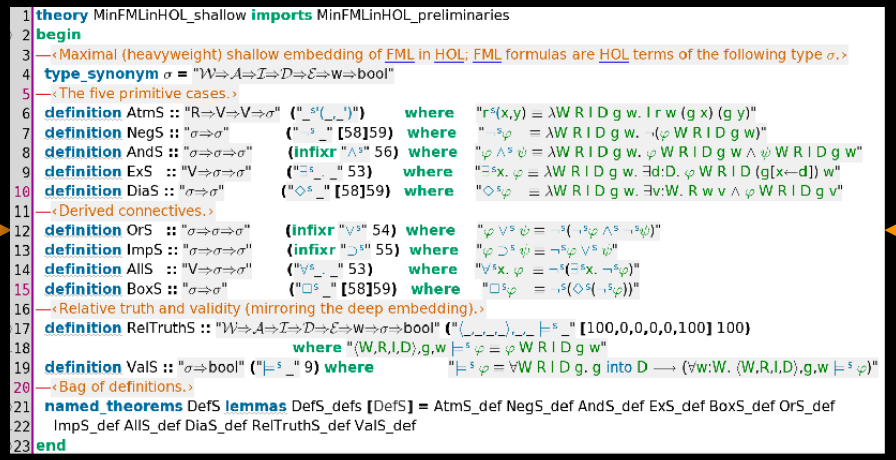
locale MinS (WW, RR, II, gg)

$\sigma = w\Rightarrow bool$

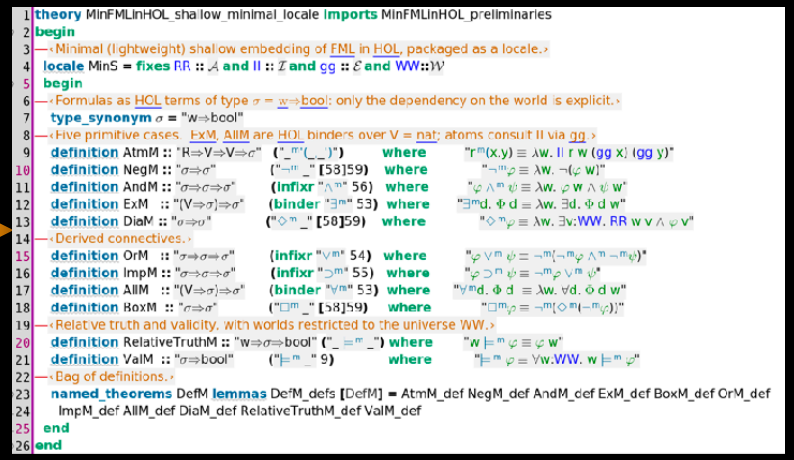
→ lightweight, best automation



```
theory MinFMLinHOL_deep imports MinFMLinHOL_preliminaries
begin
‹Deep embedding of (minimal) FML in HOL with constant-domain relational semantics.›
datatype 𝓕 = AtmD R V V ("_ᵈ(_,_)") | NegD 𝓕 ("¬ᵈ_" [58]59) | AndD 𝓕 𝓕 (infixr "∧ᵈ" 56)
           | ExD V 𝓕 ("∃ᵈ_._") | DiaD 𝓕 ("◇ᵈ_" [58]59)
‹Further connectives as derived definitions.›
definition OrD  (infixr "∨ᵈ" 54)   where "φ ∨ᵈ ψ ≡ ¬ᵈ(¬ᵈφ ∧ᵈ ¬ᵈψ)"
definition ImpD (infixr "⊃ᵈ" 55)   where "φ ⊃ᵈ ψ ≡ ¬ᵈφ ∨ᵈ ψ"
definition AllD ("∀ᵈ_._")          where "∀ᵈx. φ ≡ ¬ᵈ(∃ᵈx. ¬ᵈφ)"
definition BoxD ("□ᵈ_" [58]59)     where "□ᵈφ  ≡ ¬ᵈ(◇ᵈ(¬ᵈφ))"
‹Relative truth in a model. Existential ranges over D; box ranges over R-successors in W.›
primrec RelativeTruthD::"W⇒A⇒I⇒D⇒E⇒w⇒𝓕⇒bool" ("_,_,_,_,_,_⊨ᵈ_" [10,10,10,10,10,10] 100) where
  "(W,R,I,D),g,w ⊨ᵈ (r$(x,y))   = I r w (g x) (g y)"
| "(W,R,I,D),g,w ⊨ᵈ (¬ᵈφ)       = (¬ (W,R,I,D),g,w ⊨ᵈ φ)"
| "(W,R,I,D),g,w ⊨ᵈ (φ ∧ᵈ ψ)    = ((W,R,I,D),g,w ⊨ᵈ φ ∧ (W,R,I,D),g,w ⊨ᵈ ψ)"
| "(W,R,I,D),g,w ⊨ᵈ (∃ᵈx. φ)    = (∃d:D. (W,R,I,D),g[x↦d],w ⊨ᵈ φ)"
| "(W,R,I,D),g,w ⊨ᵈ (◇ᵈφ)       = (∃v:W. R w v ∧ (W,R,I,D),g,v ⊨ᵈ φ)"
‹Validity: true in every model, every domain-respecting assignment, and every world of the universe.›
definition ValD ("⊨ᵈ _" 9)     where "⊨ᵈ φ ≡ ∀W R I D g. g into D ⟶ (∀w:W. (W,R,I,D),g,w ⊨ᵈ φ)"
‹Auxiliary "full-domain" notion of validity: restricts to assignments ranging over Univ of D.›
definition ValD' ("⊨ᵈ' _" 9)   where "⊨ᵈ' φ = ∀W R I g. ∀w. (Univ,R,I,Univ),g,w ⊨ᵈ φ"
‹General validity implies full-domain validity.›
lemma Val:  "⊨ᵈ φ ⟹ ⊨ᵈ' φ"  using ValD'_def ValD_def by simp
‹Adding a nonemptiness premise on the world universe leaves validity unchanged.›
lemma ValD_nonemptyW: "(⊨ᵈ φ) = (∀W R I D g. (∃w. W w) ⟶ g into D ⟶ (∀w:W. (W,R,I,D),g,w ⊨ᵈ φ))"
  by (smt (verit, ccfv_threshold) ValD_def)
‹Bag of definitions for collective unfolding.›
named_theorems DefD lemmas [DefD] = OrD_def ImpD_def AllD_def BoxD_def ValD_def ValD'_def
end
```



```
theory MinFMLinHOL_shallow imports MinFMLinHOL_preliminaries
begin
‹Maximal (heavyweight) shallow embedding of FML in HOL: FML formulas are HOL terms of the following type σ.›
type_synonym σ = "𝒲⇒𝒜⇒ℐ⇒𝒟⇒ℰ⇒w⇒bool"
‹The five primitive cases.›
definition AtmS :: "R⇒V⇒V⇒σ"   ("_ˢ(_,_)")  where "rˢ(x,y) = λW R I D g w. I r w (g x) (g y)"
definition NegS :: "σ⇒σ"        ("¬ˢ_" [58]59) where "¬ˢφ   = λW R I D g w. ¬(φ W R I D g w)"
definition AndS :: "σ⇒σ⇒σ"      (infixr "∧ˢ" 56) where "φ ∧ˢ ψ = λW R I D g w. φ W R I D g w ∧ ψ W R I D g w"
definition ExS  :: "V⇒σ⇒σ"      ("∃ˢ_._" 53)   where "∃ˢx. φ = λW R I D g w. ∃d:D. φ W R I D (g[x↦d]) w"
definition DiaS :: "σ⇒σ"        ("◇ˢ_" [58]59) where "◇ˢφ   = λW R I D g w. ∃v:W. R w v ∧ φ W R I D g v"
‹Derived connectives.›
definition OrS  :: "σ⇒σ⇒σ"      (infixr "∨ˢ" 54) where "φ ∨ˢ ψ ≡ ¬ˢ(¬ˢφ ∧ˢ ¬ˢψ)"
definition ImpS :: "σ⇒σ⇒σ"      (infixr "⊃ˢ" 55) where "φ ⊃ˢ ψ ≡ ¬ˢφ ∨ˢ ψ"
definition AllS :: "V⇒σ⇒σ"      ("∀ˢ_._" 53)   where "∀ˢx. φ ≡ ¬ˢ(∃ˢx. ¬ˢφ)"
definition BoxS :: "σ⇒σ"        ("□ˢ_" [58]59) where "□ˢφ  ≡ ¬ˢ(◇ˢ(¬ˢφ))"
‹Relative truth and validity (mirroring the deep embedding).›
definition RelTruthS :: "𝒲⇒𝒜⇒ℐ⇒𝒟⇒ℰ⇒w⇒σ⇒bool" ("_,_,_,_,_,_⊨ˢ_" [100,0,0,0,0,100] 100)
                        where "(W,R,I,D),g,w ⊨ˢ φ = φ W R I D g w"
definition ValS :: "σ⇒bool" ("⊨ˢ _" 9) where "⊨ˢ φ = ∀W R I D g. g into D ⟶ (∀w:W. (W,R,I,D),g,w ⊨ˢ φ)"
‹Bag of definitions.›
named_theorems DefS lemmas DefS_defs [DefS] = AtmS_def NegS_def AndS_def ExS_def BoxS_def OrS_def
  ImpS_def AllS_def DiaS_def RelTruthS_def ValS_def
end
```



```
theory MinFMLinHOL_shallow_minimal_locale imports MinFMLinHOL_preliminaries
begin
‹Minimal (lightweight) shallow embedding of FML in HOL, packaged as a locale.›
locale MinS = fixes RR :: 𝒜 and II :: ℐ and gg :: ℰ and WW::𝒲
begin
‹Formulas as HOL terms of type σ = w⇒bool: only the dependency on the world is explicit.›
  type_synonym σ = "w⇒bool"
‹Five primitive cases. ExM, AllM are HOL binders over V = nat; atoms consult II via gg.›
  definition AtmM :: "R⇒V⇒V⇒σ"   ("_ᵐ(_,_)")  where "rᵐ(x,y) ≡ λw. I r w (gg x) (gg y)"
  definition NegM :: "σ⇒σ"        ("¬ᵐ_" [58]59) where "¬ᵐφ ≡ λw. ¬(φ w)"
  definition AndM :: "σ⇒σ⇒σ"      (infixr "∧ᵐ" 56) where "φ ∧ᵐ ψ ≡ λw. φ w ∧ ψ w"
  definition ExM  :: "(V⇒σ)⇒σ"    (binder "∃ᵐ" 53) where "∃ᵐd. Φ d ≡ λw. ∃d. Φ d w"
  definition DiaM :: "σ⇒σ"        ("◇ᵐ_" [58]59) where "◇ᵐφ ≡ λw. ∃v:WW. RR w v ∧ φ v"
‹Derived connectives.›
  definition OrM  :: "σ⇒σ⇒σ"      (infixr "∨ᵐ" 54) where "φ ∨ᵐ ψ ≡ ¬ᵐ(¬ᵐφ ∧ᵐ ¬ᵐψ)"
  definition ImpM :: "σ⇒σ⇒σ"      (infixr "⊃ᵐ" 55) where "φ ⊃ᵐ ψ ≡ ¬ᵐφ ∨ᵐ ψ"
  definition AllM :: "(V⇒σ)⇒σ"    (binder "∀ᵐ" 53) where "∀ᵐd. Φ d ≡ λw. ∀d. Φ d w"
  definition BoxM :: "σ⇒σ"        ("□ᵐ_" [58]59) where "□ᵐφ ≡ ¬ᵐ(◇ᵐ(¬ᵐφ))"
‹Relative truth and validity, with worlds restricted to the universe WW.›
  definition RelativeTruthM :: "w⇒σ⇒bool" ("_ ⊨ᵐ _") where "w ⊨ᵐ φ ≡ φ w"
  definition ValM :: "σ⇒bool"     ("⊨ᵐ _" 9)     where "⊨ᵐ φ ≡ ∀w:WW. w ⊨ᵐ φ"
‹Bag of definitions.›
  named_theorems DefM lemmas DefM_defs [DefM] = AtmM_def NegM_def AndM_def ExM_def BoxM_def OrM_def
    ImpM_def AllM_def DiaM_def RelativeTruthM_def ValM_def
end
end
```

faithfulness — machine-checked, largely automated

Further experiment stated and proved in all three variants (d / s / m) — the alignment is operational

Three interlinked embeddings: deep & max/min shallow

Deep $(\cdot)^d$

datatype $\mathcal{F}$ of formulae

truth by primitive recursion

→ metalogical reasoning

⟷

Maximal shallow $(\cdot)^s$

$\sigma = \mathcal{W} \Rightarrow \mathcal{A} \Rightarrow \mathcal{I} \Rightarrow \mathcal{D} \Rightarrow \mathcal{E} \Rightarrow w \Rightarrow bool$

every dependency a λ-abstraction

→ standard translation, heavyweight

⟷

Minimal shallow $(\cdot)^m$

locale MinS (WW, RR, II, gg)

$\sigma = w \Rightarrow bool$

→ lightweight, best automation

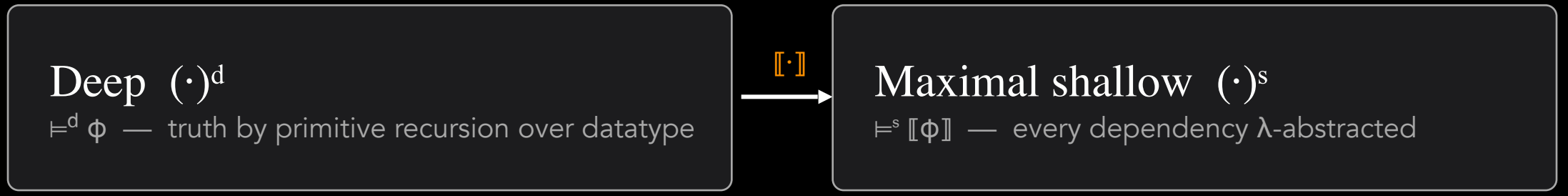
Simple and fully automated

difficult

```
theory MinFMLinHOL_faithfulness_locale imports MinFMLinHOL_deep MinFMLinHOL_shallow
 MinFMLinHOL_shallow_minimal_locale MinFMLinHOL_deep_subst_lemma MinFMLinHOL_ElementarySubstructure
begin
—‹Translation from deep to maximal shallow.›
 fun DpToShS ("⌊_⌋") where
   "⌊r^d(x,y)⌋ = r^s(x,y)" | "⌊¬^d φ⌋ = ¬^s⌊φ⌋" | "⌊φ ∧^d ψ⌋ = ⌊φ⌋ ∧^s ⌊ψ⌋" | "⌊∃^d v. φ⌋ = (∃^s v. ⌊φ⌋)" | "⌊◇^d φ⌋ = ◇^s⌊φ⌋"
 
 
 
—‹Faithfulness deep ⟷ maximal shallow. Pointwise first; global by unfolding validity.›
 theorem FaithfulSDlem: "(⟨W,R,I,D⟩,g,w ⊨^s ⌊φ⌋) ⟷ (⟨W,R,I,D⟩,g,w ⊨^d φ)"
  by (induct φ arbitrary: g w; auto simp: DefS DefD)
 theorem FaithfulSD: "(⊨^s ⌊φ⌋) ⟷ (⊨^d φ)" by (simp add: FaithfulSDlem ValD_def ValS_def)
```

Faithfulness I: deep ↔ maximal shallow, the easy case

Simple and fully automated



FaithfulSDlem : $\langle W,R,I,D\rangle, g, w \models^s [\![\phi]\!] \leftrightarrow \langle W,R,I,D\rangle, g, w \models^d \phi$ — all ϕ, g, w

FaithfulSD : $\models^s [\![\phi]\!] \leftrightarrow \models^d \phi$ — validity transfers

- One structural induction on ϕ, generalising over g and w — auto closes every case (simp: DefS, DefD)
- Both ∃-cases quantify over the full domain D — no named-variable gap, no surjectivity issue
- No substitution machinery needed — the alignment is definitional

Contrast: deep ↔ minimal shallow is the hard case — it needs **substitution machinery** and **Löwenheim–Skolem**

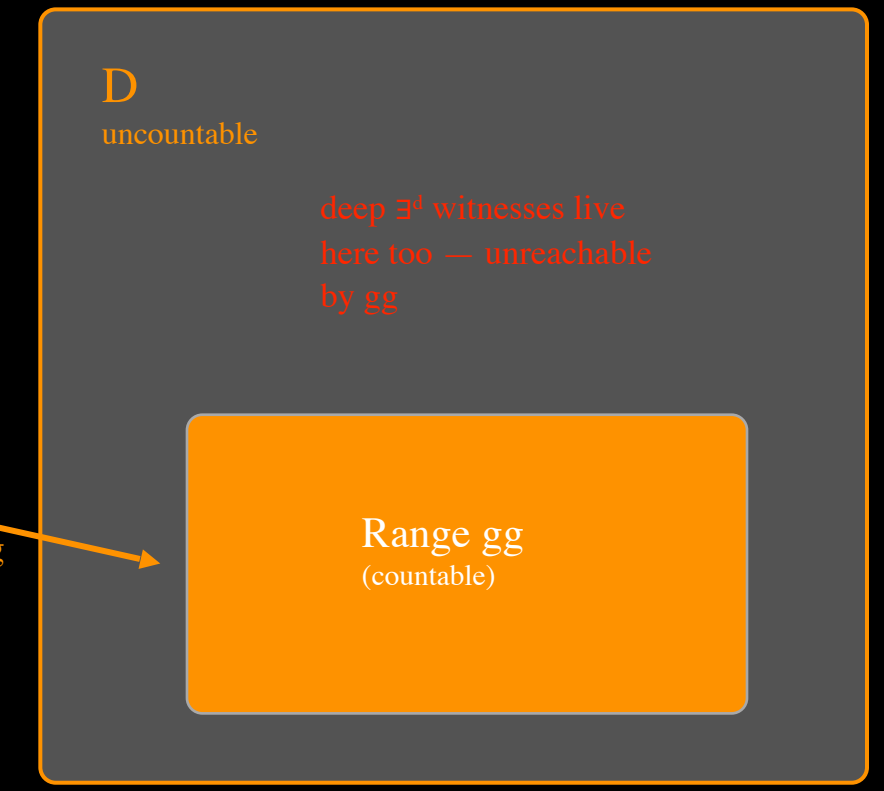
Needed: substitution machinery

- Needed for the ∃-case of the **deep ↔ minimal-shallow** faithfulness proof — not needed in propositional setting
- The standard first-order toolbox, mechanised once for the deep embedding:
  - free / bound / fresh variables; capture-avoiding substitution [x←z]ϕ; substitutability; SubstitutionLemma; alphabetic renaming; safe substitution [x←ᵣz]ϕ
  - size-based induction principles SInduct, QInduct for the trickier inductions

```
theory MinFMLinHOL_deep_subst_lemma imports MinFMLinHOL_deep
begin
‹Free and bound variable occurrences. The Box case is transparent.›
primrec is_free (infix "free'_in" 900) where
  "x free_in (r^d u v) = (x = u ∨ x = v)"
| "x free_in (¬^d φ)   = (x free_in φ)"
| "x free_in (φ ∧^d ψ) = (x free_in φ ∨ x free_in ψ)"
| "x free_in (∃^d y. φ) = (x free_in φ ∧ x ≠ y)"
| "x free_in (□^d φ)   = (x free_in φ)"
abbreviation is_not_free (infix "not_free'_in" 900) where "x not_free_in φ ≡ ¬(x free_in φ)"
fun is_bound (infix "bound'_in" 900) where
  "x bound_in (r^d u v) = False"
| "x bound_in (¬^d φ)   = (x bound_in φ)"
| "x bound_in (φ ∧^d ψ) = (x bound_in φ ∨ x bound_in ψ)"
| "x bound_in (∃^d y. φ) = (x = y ∨ x bound_in φ)"
| "x bound_in (□^d φ)   = (x bound_in φ)"
abbreviation is_not_bound (infix "not_bound'_in" 900) where "x not_bound_in φ ≡ ¬(x bound_in φ)"
abbreviation not_in       (infix "not'_in"       900) where "x not_in φ ≡ x not_free_in φ ∧ x not_bound_in φ"
‹A fresh variable: strictly larger than every variable occurring in φ.›
primrec fresh ("fresh'(_')") where
  "fresh (r^d u v) = max (u+1) (v+1)"
| "fresh (¬^d φ)   = fresh φ"
| "fresh (φ ∧^d ψ) = max (fresh φ) (fresh ψ)"
| "fresh (∃^d x. φ) = max (x+1) (fresh φ)"
| "fresh (□^d φ)   = fresh φ"
lemma L5: "x bound_in φ ⟹ x < (fresh φ)" by (induct φ) auto
lemma L6: "x free_in φ ⟹ x < (fresh φ)" by (induct φ) auto
lemma L7: "(fresh φ) not_in φ"          using L5 L6 by blast
lemma L8: "max (fresh φ) (fresh ψ) not_free_in φ" by (metis L6 L7 max.absorb1 max_def)
lemma L9: "max (fresh φ) (fresh ψ) not_bound_in φ" by (metis L5 L7 max.absorb3 max_def)
‹Irrelevance lemma: updating a non-free variable does not affect truth.›
lemma L12: "y not_free_in φ ⟹ (⟨W,R,I,D⟩,g[y:=c],w ⊨^c φ) = (⟨W,R,I,D⟩,g,w ⊨^d φ)"
  by (induct φ arbitrary: g w; simp; metis L4 L7)
‹Variable-for-variable substitution. Box descends transparently.›
primrec Subst ("[_←_]_(_)") where
  "[x←z](r^d u v) = r^d (if x = u then z else u) (if x = v then z else v)"
| "[x←z](¬^d φ)   = ¬^d [x←z]φ"
| "[x←z](φ ∧^d ψ) = ([x←z]φ ∧^d [x←z]ψ)"
| "[x←z](∃^d y. φ) = (if x = y then (∃^d y. φ) else (∃^d y. [x←z]φ))"
| "[x←z](□^d φ)   = □^d [x←z]φ"
lemma L13 [simp]: "size [x←z]φ = size φ" by (induct φ; auto)
lemma L14 [simp]: "[x←x]φ = φ"          by (induct φ; auto)
lemma L16 [simp]: assumes "a ≠ x" shows "a not_free_in ([a←x]φ)" using assms by (induct φ) auto
‹Size-based induction principle (analogous to QBF's SInduct, with a Box clause).›
lemma SInduct:
  assumes "∧r u v. P (r^d u v)"
    and "∧φ.   (∧ψ. size ψ < size φ          ⟹ P ψ) ⟹ P (¬^d φ)"
    and "∧φ ψ. (∧χ. size χ < size φ + size ψ ⟹ P χ) ⟹ P (φ ∧^d ψ)"
    and "∧y φ. (∧ψ. size ψ ≤ size φ          ⟹ P ψ) ⟹ P (∃^d y. φ)"
    and "∧φ.   (∧ψ. size ψ ≤ size φ          ⟹ P ψ) ⟹ P (□^d φ)"
  shows "P φ"
  using assms proof (induct "size φ" arbitrary: φ rule: less_induct) case less thus ?case by (induct φ) auto qed
```

The obstacle: minimal ∃ reaches only Range gg

- gg : V → D is frozen as a locale parameter; the object ∃ is HOL's own binder — every witness is some gg v
- Plain induction gives FaithfulMDlem: deep ⇔ minimal at Range gg only
- Range gg is countable (V = nat), D may be uncountable — full-domain faithfulness does not follow



Worked example — ∃x. r(x, y)

deep: true at w iff ∃ d ∈ D. (II r w) d (gg y) — the witness d roams all of D

minimal: ⦇∃x. r(x,y)⦈ = λw. ∃ v::V. (II r w) (gg v) (gg y)

i.e. true at w iff ∃ a ∈ Range gg. (II r w) a (gg y) — a roams only Range gg

The gap is cardinality, not encoding — classical model theory to the rescue: shrink D instead of stretching gg

The fix: shrink the domain, don’t grow the syntax

Idea

Replace D by a countable elementary N with the same truths — a nat-indexed gg can be onto N.

Downward Löwenheim–Skolem

Every model ⟨ W, R, I, D ⟩ has a countable elementary sub-model ⟨ W′, R, I, N⟩ — truth preserved both ways, for all ϕ.

(elementarity thereby preserves truth for nested ∃ / ◇ formulas — exactly the formulas that caused the problem)

Faithful at the full domain D — LS is the semantic license for minimalization while keeping V = nat

Downward Löwenheim–Skolem, mechanised

Every constant-domain Kripke model $\langle W,R,I,D\rangle$ with countable seeds $D_0 \sqsubseteq D$, $W_0 \sqsubseteq W$ has a countable elementary sub-model $\langle \mathcal{W}', R, I, \mathcal{N}\rangle$, where $D_0 \sqsubseteq \mathcal{N} \sqsubseteq D$ and $W_0 \sqsubseteq \mathcal{W}' \sqsubseteq W$, so that deep truth is preserved

```
255 ‹The downward Loewenheim-Skolem theorem for FML.›
256 theorem DownwardLowenheimSkclem:
257   assumes "D₀ ⊆ D" and "W₀ ⊆ W" and "countable (Collect D₀)" and "countable (Collect W₀)"
258   shows "∃N W'. D₀ ⊆ N ∧ N ⊆ D ∧ countable (Collect N) ∧ W₀ ⊆ W' ∧ W' ⊆ W ∧ countable (Collect W')
259            ∧ (∀g φ w. W' w ⟶ g into N ⟶ (⟨W,R,I,D⟩,g,w ⊨ᶜ φ) = (⟨W',R,I,N⟩,g,w ⊨ᵈ φ))"
260   apply (safe intro!: exI[where x="𝒩 D₀ W₀ W R I D"] exI[where x="𝒲 D₀ W₀ W R I D"])
261   apply (metis 𝒩_def D₀_in_𝒩)
262   apply (metis assms(1) 𝒩_subset)
263   apply (metis assms(3) assms(4) countable_𝒩)
264   apply (simp add: W₀_in_𝒲)
265   apply (metis assms(2) 𝒲_subset)
266   apply (metis assms(4) assms(3) countable_𝒲)
267   using 𝒩_valid assms(1,2) apply blast
268   by (metis assms(2) assms(1) 𝒩_valid)
```

- Entire file (~350 lines): Hilbert-choice witnesses for ∃ and ◇ → simultaneous stage-wise saturation of $\mathcal{N}$, $\mathcal{W}$
- First Löwenheim–Skolem mechanisation in the Kripke setting (to our knowledge) — reusable for other embeddings

Faithfulness, automated

- Translations $[\![\cdot]\!]$ and $(\!|\cdot|\!)$
- deep ↔ maximal shallow
- deep ↔ minimal shallow: relative to Range gg

```
theory MinFMLinHOL_faithfulness_locale imports MinFMLinHOL_deep MinFMLinHOL_shallow
 MinFMLinHOL_shallow_minimal_locale MinFMLinHOL_deep_subst_lemma MinFMLinHOL_ElementarySubstructure
begin
—‹Translation from deep to maximal shallow.›
 fun DpToShS ("[_]") where
   "[r^d(x,y)] = r^s(x,y)" | "[¬^d φ] = ¬^s[φ]" | "[φ ∧^d ψ] = [φ] ∧^s [ψ]" | "[∃^d v. φ] = (∃^s v. [φ])" | "[◇^d φ] = ◇^s[φ]"
—‹Translation from deep to minimal shallow (within the locale MinS.›
 fun (in MinS) DpToShM ("(_)") where
   "(r^d(x,y)) = r^m(x,y)" | "(¬^d φ) = ¬^m(φ)" | "(φ ∧^d ψ) = (φ) ∧^m (ψ)" | "(∃^d v. φ) = (∃^m d. (φ))" | "(◇^d φ) = ◇^m(φ)"
—‹Faithfulness deep ⟷ maximal shallow. Pointwise first; global by unfolding validity.›
 theorem FaithfulSDlem: "(⟨W,R,I,D⟩,g,w ⊨^s [φ]) ⟷ (⟨W,R,I,D⟩,g,w ⊨^d φ)"
  by (induct φ arbitrary: g w; auto simp: DefS DefD)
 theorem FaithfulSD: "(⊨^s [φ]) ⟷ (⊨^d φ)" by (simp add: FaithfulSDlem ValD_def ValS_def)
—‹Faithfulness deep ⟷ minimal shallow, parametrised by the locale.›
 context MinS
  begin
   theorem FaithfulMDlem: "(w ⊨^m (φ)) ⟷ (⟨WW,RR,II,Range gg⟩,gg,w ⊨^d φ)"
    by (induct φ arbitrary: w rule: Qinduct; simp add: DefD DefM) blast
   theorem FaithfulMD: "(⊨^m (φ)) ⟷ (∀w:WW. ⟨WW,RR,II,Range gg⟩,gg,w ⊨^d φ)"
    using FaithfulMDlem by (simp add: ValM_def)
   theorem FaithfulMSlem: "(w ⊨^m (φ)) ⟷ (⟨WW,RR,II,Range gg⟩,gg,w ⊨^s [φ])"
    using FaithfulSDlem FaithfulMDlem by auto
   theorem FaithfulMS: "(⊨^m (φ)) ⟷ (∀w:WW. ⟨WW,RR,II,Range gg⟩,gg,w ⊨^s [φ])"
    using FaithfulMSlem by (simp add: ValM_def RelativeTruthM_def)
  end
—‹Stronger version for elementary substructures.›
```

```
—‹Global faithfulness: quantifying over all locale interpretations of MinS recovers exactly deep validity.›
theorem FaithfulMS_all: "(∀WW RR II gg. MinS.ValM WW (MinS.DpToShM RR II gg WW φ)) = (⊨^d φ)"
proof(unfold ValD_def; safe)
 {
  fix W :: 𝒲 and R :: 𝒜 and I D and g::ℰ and x
  assume "g into D"
  moreover assume "W x"
  ultimately interpret MinS_specific R I W D g φ x  using MinS_specific_def by simp
  assume "∀WW RR II gg. MinS.ValM WW (MinS.DpToShM RR II gg WW φ)"
  hence "⟨WW,RR,I,Range gg⟩,gg,x ⊨^d φ" using FaithfulMD w_in_WW by blast
  thus "⟨W,R,I,D⟩,g,x ⊨^d φ" using valid_specific by simp
 }
qed(metis (mono_tags, lifting) MinS.FaithfulMD)
```

Global faithfulness

Quantifying over all locale interpretations of MinS recovers exactly deep validity — every locale-level proof transfers to the deep embedding in one line.

Experiments & Automation+: Prove once, transfer for free

- K, necessitation, Barcan & converse Barcan: valid under constant domain — uniform one-line tactics, in all three embeddings
- nitpick disproves invalid schemata with small countermodels: modal collapse φ ⊃ □φ (2 worlds), □φ ⊃ φ without reflexivity (T correspondence)

```
32 —‹Existential variants of Barcan: under constant domain, ◇∃ and ∃◇ coincide.›
33 lemma Barcan_Dia_d:      "⊨d (◇d (∃dx. Pd(x,x))) ⊃d (∃dx. ◇d Pd(x,x))" unfolding DefD by auto
34 lemma Barcan_Dia_s:      "⊨s (◇s (∃sx. Ps(x,x))) ⊃s (∃sx. ◇s Ps(x,x))" unfolding DefS by auto
35 lemma Barcan_Dia_m:      "⊨m (◇m (∃mx. Pm(x,x))) ⊃m (∃mx. ◇m Pm(x,x))" unfolding DefM by auto
36 lemma Barcan_Dia_d_conv: "⊨d (∃dx. ◇d Pd(x,x)) ⊃d ◇d (∃dx. Pd(x,x))" unfolding DefD by auto
37 lemma Barcan_Dia_s_conv: "⊨s (∃sx. ◇s Ps(x,x)) ⊃s ◇s (∃sx. Ps(x,x))" unfolding DefS by auto
38 lemma Barcan_Dia_m_conv: "⊨m (∃mx. ◇m Pm(x,x)) ⊃m ◇m (∃mx. Pm(x,x))" unfolding DefM by auto
```

```
43 —‹Modal collapse φ ⊃ □φ fails. (Note: needs full domain D for ⊨d'.)›
44 lemma "⊨d' Pd(x,x) ⊃d □d Pd(x,x)" unfolding DefD apply simp nitpick oops
45 lemma "⊨m  Pm(x,x) ⊃m □m Pm(x,x)" unfolding DefM nitpick oops
46 —‹T-axiom □φ ⊃ φ fails without reflexivity. (Note: needs full domain D.)›
47 lemma "⊨d' □d Pd(x,x) ⊃d Pd(x,x)" unfolding DefD apply simp nitpick oops
48 lemma "⊨m  □m Pm(x,x) ⊃m Pm(x,x)" unfolding DefM nitpick oops
```

Transfer in one line

A theorem proved in every interpretation of the minimal-shallow locale is exported to deep validity by FaithfulMS_all — one `using MinS_to_Deep` discharges the deep goal, no re-proof.

```
1  theory MinFMLinHOL_experiments_locale imports MinFMLinHOL_faithfulness_lccale
2  begin
3  —‹Demonstration: minimal-shallow proofs that hold in every interpretation of the
4    MinS locale transfer directly to validity in the deep embedding.›
5  —‹Derived rule: from validity in every minimal interpretation to deep validity.›
6  lemmas MinS_to_Deep = FaithfulMS_all[THEN iffD1, rule_format]
7  —‹Additional simplification rules for derived connectives under DpToShM.›
8  lemma (in MinS) OrM_simp[DefM]:  "(φ ∨d ψ) = (φ) ∨m (ψ)"  by (simp add: OrD_def OrM_def)
9  lemma (in MinS) ImpM_simp[DefM]: "(φ ⊃d ψ) = (φ) ⊃m (ψ)" by (simp add: ImpD_def DefM)
10 lemma (in MinS) BoxM_simp[DefM]: "(□d φ) = □m (φ)" by (simp add: BoxD_def DefM)
11 —‹Example: the K axiom proved using only minimal shallow infrastructure.›
12 lemma (in MinS) K_MinS: "⊨m (□d(Pd(x,y) ⊃d Pd(y,x)) ⊃d (□d Pd(x,y) ⊃d □d Pd(y,x)))" by (auto simp: DefM)
13 —‹Transfer to the deep embedding requires no further work.›
14 lemma "⊨d □d(Pd(x,y) ⊃d Pd(y,x)) ⊃d (□d Pd(x,y) ⊃d □d Pd(y,x))" using MinS_to_Deep[OF MinS.K_MinS].
15 end
```

Conclusion — what's new here

- The deep-and-shallow methodology scales from PML to FML — binding and modality together
- Locale-packaged minimal shallow embedding + global faithfulness theorem over all interpretations
- Downward Löwenheim–Skolem for (constant-domain) Kripke semantics, surjectivity obstacle solved
- For LogiKEy: deep—maximal shallow—minimal shallow … now also for quantified modal logics

Future work: **varying domains; multimodal & non-normal modalities; other classical and non-classical logics**
(well, we just did MSO in HOL using similar techniques)

MinFMLinHOL — paper, appendix, full Isabelle/HOL development (11 theory files + ROOT)
arXiv:2607.10880 · loads cleanly in current Isabelle/HOL

A6 · Related work in detail

- Benzmüller & Paulson: minimal shallow embedding of quantified/higher-order modal logic in HOL (computational metaphysics) — no parallel deep embedding; faithfulness not mechanised inside HOL
- Benzmüller & Reiche (public announcement logic): shows the need for explicit evaluation-domain dependencies in a heavyweight shallow embedding — our maximal embedding generalises that pattern
- Kirst & Shillito 2025 (Coq): completeness for first-order bi-intuitionistic logic with constant domains — a syntactic metatheorem for an intuitionistic logic; ours is a semantic deep–shallow bridge with automated proof/disproof for a classical modal one
- From 2022: Löwenheim–Skolem for FOL in Isabelle — not for constant-domain Kripke semantics, nor deployed to bridge embeddings
- Other deep embeddings of FML target metalogical results (completeness, cut-elimination) rather than mutual faithfulness with shallow embeddings
- Locales are standard for algebraic structures (Ballarin) — using them to package shallow embeddings of object logics, with a global faithfulness theorem, appears to be new

Appendix

Backup slides for questions

A1 Shared preliminaries · locale vs. constants

A2 Substitution machinery (sources)

A3 Löwenheim–Skolem: the construction in detail

A4 FaithfulMS_all: statement and proof (sources)

A5 Experiments: Barcan, converse Barcan, nitpick countermodels

A6 Related work in detail

A1 · Shared preliminaries · locale vs. constants

```
theory MinFMLinHOL_preliminaries imports Main
begin
—‹Some global settings› declare[[syntax_ambiguity_warning=false]] nitpick_params[user_axioms,expect=genuine]
—‹Declarations shared between the deep, maximal-shallow, and minimal-shallow embeddings of FML in HOL.›
  typedecl D                    —‹Domain of individuals (constant across worlds).›
  typedecl w                    —‹Type of possible worlds.›
  type_synonym R = nat          —‹Binary relation symbols (encoded as naturals).›
  type_synonym V = nat          —‹Individual variable symbols (encoded as naturals).›
  type_synonym 𝓡 = "D⇒D⇒bool"   —‹Binary relations on individuals.›
  type_synonym 𝒲 = "w⇒bool"     —‹Sets of possible worlds (the universe).›
  type_synonym 𝒜 = "w⇒w⇒bool"   —‹Accessibility relations between worlds.›
  type_synonym ℐ = "R⇒w⇒𝓡"      —‹World-indexed interpretations of relation symbols.›
  type_synonym ℰ = "V⇒D"        —‹Variable assignment functions.›
  type_synonym 𝒟 = "D⇒bool"     —‹Domain restrictions (subsets of D as predicates).›
—‹Pointwise update of variable assignments.›
  definition EnvMod::"ℰ⇒V⇒D⇒ℰ" ("_[_←_]" [110,0,0] 110) where   "g[x←d] ≡ λz. if z = x then d else g z"
—‹Standard lemmas about variable-assignment update.›
  lemma L1 [simp]: "x ≠ y ⟹ (g[y←d]) x = g x"                  by (simp add: EnvMod_def)
  lemma L2:        "a ≠ c ⟹ g[a←d₁][c←d₂] = g[c←d₂][a←d₁]"    by (auto simp: EnvMod_def)
  lemma L3 [simp]: "(g[a←d]) a = d"                            by (simp add: EnvMod_def)
  lemma L4 [simp]: "g[a←d₁][a←d₂] = g[a←d₂]"                   by (auto simp: EnvMod_def)
—‹Standard predicates on accessibility relations (carried over from the PML preliminaries).›
  abbreviation "reflexive   ≡ λR::𝒜. ∀x. R x x"
  abbreviation "symmetric   ≡ λR::𝒜. ∀x y. R x y ⟶ R y x"
  abbreviation "transitive  ≡ λR::𝒜. ∀x y z. R x y ∧ R y z ⟶ R x z"
—‹Bounded quantifiers: ‹∀x:W. φ› stands for ‹∀x. W x ⟶ φ x› and ‹∃x:W. φ› stands for ‹∃x. W x ∧ φ x›.›
  abbreviation (input) "BAll W φ ≡ ∀x. W x ⟶ φ x"
  syntax "BAll"::"pttrn⇒logic⇒bool⇒bool" ("(∀(_:_)./ _)" [0,0,10]10)
  translations "∀x:W. φ" ⇌ "CONST BAll W (λx. φ)"
  abbreviation (input) "BEx W φ ≡ ∃x. W x ∧ φ x"
  syntax "BEx"::"pttrn⇒logic⇒bool⇒bool" ("(∃(_:_)./ _)" [0,0,10]10)
  translations "∃x:W. φ" ⇌ "CONST BEx W (λx. φ)"
—‹Set-as-predicate operations, range, and the universal domain.›
  abbreviation (input) "Into"     (infix "into" 100)   where  "f into D ≡ ∀x. D (f x)"
  abbreviation (input) "Range"                         where  "Range f ≡ λx. ∃y. x = f y"
  abbreviation (input) "Onto"     (infix "onto" 100)   where  "f onto D ≡ D = Range f"
  abbreviation (input) "Subset"   (infix "⊆" 100)      where  "A ⊆ B ≡ ∀x. A x ⟶ B x"
  abbreviation (input) "Union"    (infixl "∪" 110)     where  "A ∪ B ≡ λx. A x ∨ B x"
  abbreviation (input) "Univ"                          where  "Univ ≡ λx. True"
—‹Backward implication, useful for stating equivalences in two directions.›
  abbreviation (input) Bimp (infixr "⟵" 50) where "φ ⟵ ψ ≡ ψ ⟶ φ"
end
```

MinFMLinHOL_preliminaries.thy

```
theory MinFMLinHOL_shallow_minimal imports MinFMLinHOL_faithfulness_locale
begin
—‹Global interpretation of the minimal shallow embedding.›
  consts RR::𝒜 II::ℐ gg::𝒟 WW::𝒲
  global_interpretation MinS RR II gg WW.
—‹Re-introduce the locale-internal notation at the global level.›
  notation AtmM ("_'(_')") and NegM ("¬_" [58]59) and AndM (infixr "∧" 56) and ExM (binder "∃" 53)
    and BoxM ("□_" [58]59) and OrM (infixr "∨" 54) and ImpM (infixr "⊃" 55) and AllM (binder "∀" 53)
    and DiaM ("◇_" [58]59) and RelativeTruthM ("_ ⊨ _") and ValM ("⊨ _" 9) and DpToShM ("⌊_⌋")
end
```

MinFMLinHOL_shallow_minimal.thy — global_interpretation

- Preliminaries shared by all embeddings: types D, w, R; V := nat; assignment update g[x←d] with lemmata L1–L4; reflexive/ symmetric/transitive; bounded ∀/∃; ⊑, ⊔, Range, Univ, into, onto
- Locale MinS is the single source of theorems; global_interpretation with uninterpreted constants RR, II, gg, WW propagates them — no proof duplication
- Why both? nitpick handles uninterpreted constants more reliably than locale parameters; the locale enables coexisting interpretations and the global theorem

A2 · Substitution machinery (sources)

```
theory MinFMLinHOL_deep_subst_lemma imports MinFMLinHOL_deep
begin
‹Free and bound variable occurrences. The Box case is transparent.›
primrec is_free (infix "free'_in" 900) where
  "x free_in (r⁰(u,v)) = (x = u ∨ x = v)"
| "x free_in (¬⁰φ)     = (x free_in φ)"
| "x free_in (φ ∧⁰ ψ) = (x free_in φ ∨ x free_in ψ)"
| "x free_in (∃⁰y. φ) = (x free_in φ ∧ x ≠ y)"
| "x free_in (□⁰φ)     = (x free_in φ)"
abbreviation is_not_free (infix "not'_free'_in" 900) where "x not_free_in φ ≡ ¬(x free_in φ)"
fun is_bound (infix "bound'_in" 900) where
  "x bound_in (r⁰(u,v)) = False"
| "x bound_in (¬⁰φ)     = (x bound_in φ)"
| "x bound_in (φ ∧⁰ ψ) = (x bound_in φ ∨ x bound_in ψ)"
| "x bound_in (∃⁰y. φ) = (x = y ∨ x bound_in φ)"
| "x bound_in (□⁰φ)     = (x bound_in φ)"
abbreviation is_not_bound (infix "not'_bound'_in" 900) where "x not_bound_in φ ≡ ¬(x bound_in φ)"
abbreviation not_in       (infix "not'_in"      900) where "x not_in φ ≡ x not_free_in φ ∧ x not_bound_in φ"
‹A fresh variable: strictly larger than every variable occurring in φ.›
primrec fresh ("fresh⁰(_)") where
  "fresh (r⁰(u,v)) = max (u+1) (v+1)"
| "fresh (¬⁰φ)     = fresh φ"
| "fresh (φ ∧⁰ ψ) = max (fresh φ) (fresh ψ)"
| "fresh (∃⁰x. φ) = max (x+1) (fresh φ)"
| "fresh (□⁰φ)     = fresh φ"
lemma L5: "x bound_in φ ⟹ x < (fresh⁰φ)" by (induct φ) auto
lemma L6: "x free_in φ ⟹ x < (fresh⁰φ)" by (induct φ) auto
lemma L7: "¬fresh⁰φ not_in φ"          using L5 L6 by blast
lemma L8: "max (fresh φ) (fresh ψ) not_free_in φ" by (metis L6 L7 max.absorb3 max_def)
lemma L9: "max (fresh φ) (fresh ψ) not_bound_in φ" by (metis L5 L7 max.absorb3 max_def)
‹Irrelevance lemma: updating a non-free variable does not affect truth.›
lemma L12: "y not_free_in φ ⟹ ((W,R,I,D),g(y:=c),w ⊨⁰ φ) = ((W,R,I,D),g,w ⊨⁰ φ)"
  by (induct φ arbitrary: g w) simp+ metis L4 L2)
‹Variable-for-variable substitution. Box descends transparently.›
primrec Subst ("[_←_]⁰(_)") where
  "[x←z](r⁰(u,v)) = r⁰(if x = u then z else u, if x = v then z else v)"
| "[x←z](¬⁰φ)     = ¬⁰([x←z]φ)"
| "[x←z](φ ∧⁰ ψ) = ([x←z]φ ∧⁰ [x←z]ψ)"
| "[x←z](∃⁰y. φ) = (if x = y then (∃⁰y. φ) else (∃⁰y. [x←z]φ))"
| "[x←z](□⁰φ)     = □⁰([x←z]φ)"
lemma L13 [simp]: "size ([x←z]φ) = size φ" by (induct φ) auto)
lemma L14 [simp]: "[x←x]φ = φ"          by (induct φ) auto)
lemma L16 [simp]: assumes "a ≠ x" shows "a not_free_in ([a←x]φ)" using assms by (induct φ) auto
‹Size-based induction principle (analogous to QBF's SInduct, with a Box clause).›
lemma SInduct:
  assumes "∧r u v. P (r⁰(u,v))"
    and "∧φ.   (∧ψ. size ψ < size φ          ⟹ P ψ) ⟹ P (¬⁰φ)"
    and "∧φ ψ. (∧χ. size χ < size φ + size ψ ⟹ P χ) ⟹ P (φ ∧⁰ ψ)"
    and "∧φ y. (∧ψ. size ψ ≤ size φ          ⟹ P ψ) ⟹ P (∃⁰y. φ)"
    and "∧φ.   (∧ψ. size ψ ≤ size φ          ⟹ P ψ) ⟹ P (□⁰φ)"
  shows "P φ"
using assms proof (induct "size φ" arbitrary: φ rule: less_induct) case less thus ?case by (induct φ) auto qed
```

MinFMLinHOL_deep_subst_lemma.thy (excerpt)

- is_free, is_bound, fresh (with lemmata L5–L9); irrelevance lemma L12
- Variable-for-variable substitution [x←z]ϕ; substitutability predicate is_subst_for; SubstitutionLemma
- Alphabetic renaming ren_for_subst; safe substitution $[x \leftarrow_r z]\phi$ (= L27, invoked by the ∃-case of the faithfulness proof)
- Size-based induction principles SInduct and QInduct
- Every modal clause is one line, dispatched by auto/force — the ◇ descends transparently, like ¬

A3 · Löwenheim–Skolem: the construction in detail

1 Witnesses

Hilbert choice picks an individual witnessing $\exists^{d}y.\phi$ and an R-successor world witnessing $\Diamond^{d}\varphi$; partial assignments over the free variables of ϕ, encoded via finite lists → witness sets stay countable

2 Simultaneous saturation

Mutual recursion on stages: $\mathcal{N}_{i+1}$ adds existential witnesses, $\mathcal{W}_{i+1}$ adds modal witnesses, over assignments/worlds from stage i; limits $\mathcal{N}$, $\mathcal{W}$ are countable unions of countable stages

3 Tarski–Vaught test

$\langle \mathcal{W},R,I,\mathcal{N} \rangle \subseteq_{e} \langle W,R,I,D \rangle$: both witness conditions hold by construction; coincidence lemma lifts finite restrictions to full assignments; worlds use truth-up-to-modal-depth

Packaged for the faithfulness proofs: MinS_ES (elementary-substructure assumption), MinS_ES_Univ (universal domains), MinS_specific (per-model construction of a bounded RR and a surjective gg → lemma valid_specific).

A4 · FaithfulMS_all: statement and proof (sources)

$(\forall RR\ II\ gg\ WW.\ \text{MinS.ValM}\ RR\ II\ gg\ WW\ (\!|\phi|\!)) = \models^{d} \phi$ — two short tactic invocations in Isabelle

```
proof(unfold ValD'_def; safe)
  {
    fix R :: 𝒜 and I and g::ℰ and x
    interpret MinS_specific R I Univ Univ g φ x unfolding MinS_specific_def by simp
    assume "∀WW RR II gg. ⟨WW,RR,II,Range gg⟩ ⊆ᴇ ⟨Univ,RR,II,Univ⟩ ⟶
      MinS.ValM WW (MinS.DpToShM RR II gg WW φ)"
    hence "ValM (DpToShM φ)" using ES' by blast
    thus "⟨Univ,R,I,Univ⟩,g,x ⊨ᵈ φ" using FaithfulMD valid_specific w_in_WW by blast
  }
qed(simp add: MinS_ES.FaithfulMD_ES MinS_ES_def)
—‹Global faithfulness': quantifying over all locale interpretations of MinS_ES_Univ recovers exactly deep validity in
  the universal domains.›
theorem FaithfulMS_all': "(∀WW RR II gg. MinS_ES_Univ RR II gg WW ⟶
```

MinFMLinHOL_faithfulness_locale.thy — FaithfulMS_all_ES and FaithfulMS_all' (universal domains)

- Right-to-left: unfold deep validity, apply FaithfulMD. Left-to-right: instantiate MinS_specific at the arbitrary model, transfer via valid_specific (Löwenheim–Skolem), apply FaithfulMD
- Constants-level re-issue in MinFMLinHOL_faithfulness.thy via specification: countable seeds DD_0, WW_0 → elementary sub-model DD, WW, gg of ⟨Univ,RR,II,Univ⟩

A5 · Experiments: proofs and disproofs (sources)

```
lemma BF_s:  "⊨ˢ (∀ˢx. □ˢ Pˢ(x,x)) ⊃ˢ □ˢ (∀ˢx. Pˢ(x,x))" unfolding DefS by auto
lemma BF_m:  "⊨ᵐ (∀ᵐx. □ᵐ Pᵐ(x,x)) ⊃ᵐ □ᵐ (∀ᵐx. Pᵐ(x,x))" unfolding DefM by auto
lemma CBF_d: "⊨ᵈ □ᵈ (∀ᵈx. Pᵈ(x,x)) ⊃ᵈ (∀ᵈx. □ᵈ Pᵈ(x,x))" unfolding DefD by auto
lemma CBF_s: "⊨ˢ □ˢ (∀ˢx. Pˢ(x,x)) ⊃ˢ (∀ˢx. □ˢ Pˢ(x,x))" unfolding DefS by auto
lemma CBF_m: "⊨ᵐ □ᵐ (∀ᵐx. Pᵐ(x,x)) ⊃ᵐ (∀ᵐx. □ᵐ Pᵐ(x,x))" unfolding DefM by auto
—‹Existential variants of Barcan: under constant domain, ◇∃ and ∃◇ coincide.›
lemma Barcan_Dia_d:      "⊨ᵈ (◇ᵈ (∃ᵈx. Pᵈ(x,x))) ⊃ᵈ (∃ᵈx. ◇ᵈ Pᵈ(x,x))" unfolding DefD by auto
lemma Barcan_Dia_s:      "⊨ˢ (◇ˢ (∃ˢx. Pˢ(x,x))) ⊃ˢ (∃ˢx. ◇ˢ Pˢ(x,x))" unfolding DefS by auto
lemma Barcan_Dia_m:      "⊨ᵐ (◇ᵐ (∃ᵐx. Pᵐ(x,x))) ⊃ᵐ (∃ᵐx. ◇ᵐ Pᵐ(x,x))" unfolding DefM by auto
lemma Barcan_Dia_d_conv: "⊨ᵈ (∃ᵈx. ◇ᵈ Pᵈ(x,x)) ⊃ᵈ ◇ᵈ (∃ᵈx. Pᵈ(x,x))" unfolding DefD by auto
lemma Barcan_Dia_s_conv: "⊨ˢ (∃ˢx. ◇ˢ Pˢ(x,x)) ⊃ˢ ◇ˢ (∃ˢx. Pˢ(x,x))" unfolding DefS by auto
lemma Barcan_Dia_m_conv: "⊨ᵐ (∃ᵐx. ◇ᵐ Pᵐ(x,x)) ⊃ᵐ ◇ᵐ (∃ᵐx. Pᵐ(x,x))" unfolding DefM by auto
—‹Invalid schemata: nitpick reports finite countermodels.›
—‹Symmetry of P is not implied: nitpick produces a 2-element countermodel. (Note: needs full domain D.)›
lemma "⊨ᵈ' (∀ᵈx. ∀ᵈy. (Pᵈ(x,y) ⊃ᵈ Pᵈ(y,x)))" unfolding DefD apply simp nitpick oops
```

```
lemma "⊨ᵐ (∀ᵐx. ∀ᵐy. (Pᵐ(x,y) ⊃ᵐ Pᵐ(y,x)))" unfolding DefM nitpick oops
—‹Modal collapse φ ⊃ □φ fails. (Note: needs full domain D for ⊨d'.)›
lemma "⊨ᵈ' Pᵈ(x,x) ⊃ᵈ □ᵈ Pᵈ(x,x)" unfolding DefD apply simp nitpick oops
lemma "⊨ᵐ Pᵐ(x,x) ⊃ᵐ □ᵐ Pᵐ(x,x)" unfolding DefM nitpick oops
—‹T-axiom □φ ⊃ φ fails without reflexivity. (Note: needs full domain D.)›
lemma "⊨ᵈ' □ᵈ Pᵈ(x,x) ⊃ᵈ Pᵈ(x,x)" unfolding DefD apply simp nitpick oops
lemma "⊨ᵐ □ᵐ Pᵐ(x,x) ⊃ᵐ Pᵐ(x,x)" unfolding DefM nitpick oops
—‹De-re/de-dicto distinction is collapsed under constant domain: ◇∃ implies ∃◇.›
lemma "⊨ᵈ ◇ᵈ (∃ᵈx. Pᵈ(x,x)) ⊃ᵈ (∃ᵈx. ◇ᵈ Pᵈ(x,x))" unfolding DefD by auto
lemma "⊨ˢ ◇ˢ (∃ˢx. Pˢ(x,x)) ⊃ˢ (∃ˢx. ◇ˢ Pˢ(x,x))" unfolding DefS by auto
lemma "⊨ᵐ ◇ᵐ (∃ᵐx. Pᵐ(x,x)) ⊃ᵐ (∃ᵐx. ◇ᵐ Pᵐ(x,x))" unfolding DefM by auto
end
```

MinFMLinHOL_experiments.thy (excerpts)

- Propositional tautologies, K, necessitation: auto after unfolding DefD / DefS / DefM
- Barcan & converse Barcan valid (constant domain) — uniform one-line tactics; ◇/∃ versions: ◇∃ and ∃◇ coincide
- nitpick: modal collapse φ ⊃ □φ fails by a 2-world model; □φ ⊃ φ fails without reflexive edges (T correspondence)
- Some disproofs use the full-domain relation $\models^{d\prime}$

A6 · Related work in detail

- Benzmüller & Paulson: minimal shallow embedding of quantified/higher-order modal logic in HOL (computational metaphysics) — no parallel deep embedding; faithfulness not mechanised inside HOL
- Benzmüller & Reiche (public announcement logic): shows the need for explicit evaluation-domain dependencies in a heavyweight shallow embedding — our maximal embedding generalises that pattern
- Kirst & Shillito 2025 (Coq): completeness for first-order bi-intuitionistic logic with constant domains — a syntactic metatheorem for an intuitionistic logic; ours is a semantic deep–shallow bridge with automated proof/disproof for a classical modal one
- From 2022: Löwenheim–Skolem for FOL in Isabelle — not for constant-domain Kripke semantics, nor deployed to bridge embeddings
- Other deep embeddings of FML target metalogical results (completeness, cut-elimination) rather than mutual faithfulness with shallow embeddings
- Locales are standard for algebraic structures (Ballarin) — using them to package shallow embeddings of object logics, with a global faithfulness theorem, appears to be new

The fix: shrink the domain, don’t grow the syntax

1 Idea

Replace D by a countable elementary N with the same truths — a nat-indexed gg can be onto N.

2 Downward Löwenheim–Skolem

Every model ⟨ W, R, I, D ⟩ has a countable elementary sub-model ⟨ W′, R, I, N⟩ — truth preserved both ways, for all ϕ.

3 Why elementary

Transfer is prove-at-N → lift-to-D; elementarity preserves nested ∃ / ◇ formulas — exactly the formulas that caused the problem.

4 Tarski–Vaught

Turns “preserves all ϕ” into “closed under ∃- and ◇-witnesses” — Hilbert choice + stage-wise saturation, countable at each stage.

Faithful at the full domain D — LS is the semantic license for minimalization while keeping V = nat