Encoding Hybrid Logic into Higher-Order Logic

Chad E. Brown
cebrown@ags.uni-sb.de

Universität des Saarlandes, Saarbrücken, Germany
Simple Types

Simple Types $\mathcal{T}$:

- $o$ (truth values)
- $\iota$ (individuals)
- $(\alpha \beta)$ (functions from $\beta$ to $\alpha$)
Simple Types

Simple Types $\mathcal{T}$:

- $o$ (truth values)
- $i$ (individuals)
- $(\alpha \beta)$ (functions from $\beta$ to $\alpha$)

$(\alpha \beta \gamma)$ abbreviates $((\alpha \beta) \gamma)$
Simple Types

Simple Types \( \mathcal{T} \):

\[ o \] (truth values)

\[ \iota \] (individuals)

\( (\alpha \beta) \) (functions from \( \beta \) to \( \alpha \))

\( o\iota \) “sets of individuals” (characteristic functions of sets)

\( o(o\iota) \) “sets of sets”
Simply Typed \( \lambda \)-Terms

Terms:

\[ x_\alpha \] Variables \((\mathcal{V})\)

\[ A_\alpha \] Parameters \((\mathcal{P})\)

\[ c_\alpha \] Logical Constants \((\mathcal{S})\)

\[ (F_{\alpha\beta} B_\beta)_\alpha \] Application

\[ (\lambda y_\beta A_\alpha)_{\alpha\beta} \] \(\lambda\)-abstraction
Simply Typed $\lambda$-Terms

Terms:

- $x_\alpha$: Variables ($\mathcal{V}$)
- $A_\alpha$: Parameters ($\mathcal{P}$)
- $c_\alpha$: Logical Constants ($\mathcal{S}$)
- $(F_\alpha B_\beta)_\alpha$: Application
- $(\lambda y_\beta A_\alpha)_{\alpha \beta}$: $\lambda$-abstraction

Equality of terms: $\alpha \beta \eta$
Simply Typed $\lambda$-Terms

Terms:

- $x_\alpha$ Variables ($V$)
- $A_\alpha$ Parameters ($P$)
- $c_\alpha$ Logical Constants ($S$)
- $(F_{\alpha\beta} B_\beta)_\alpha$ Application
- $(\lambda y_\beta A_\alpha)_{\alpha\beta}$ $\lambda$-abstraction

Equality of terms: $\alpha\beta\eta$

- $\alpha$-conversion Changing Bound Variables
- $\beta$-reduction $((\lambda y_\beta A_\alpha) B) \xrightarrow{\beta} [B/y]A$
- $\eta$-reduction $(\lambda y_\beta (F_{\alpha\beta} y)) \xrightarrow{\eta} F$ ($y_\beta \notin \text{Free}(F)$)
Simply Typed $\lambda$-Terms

Terms:

- $x_\alpha$ Variables ($\mathcal{V}$)
- $A_\alpha$ Parameters ($\mathcal{P}$)
- $c_\alpha$ Logical Constants ($\mathcal{S}$)
- $(F_{\alpha\beta} B_\beta)_\alpha$ Application
- $(\lambda y_\beta A_\alpha)_{\alpha\beta}$ $\lambda$-abstraction

Equality of terms: $\alpha\beta\eta$

Every term has a unique $\beta\eta$-normal form, up to $\alpha$-conversion.
Logical Constants

\(\neg_{oo}\) – negation

\(\lor_{ooo}\) – disjunction

\(\Pi_{o(o\alpha)}^{\alpha}\) – universal quantification over type \(\alpha\)

\(=^{\alpha}_{o\alpha\alpha}\) – equality
(A_o \lor B_o) \text{ means } (\lor_{ooo} A_o B_o)

(A_o \supset B_o) \text{ means } (\neg A_o \lor B_o)

(\forall x_\alpha A_o) \text{ means } (\Pi_{o(o\alpha)}^\alpha \lambda x_\alpha A_o).

(\exists x_\alpha A_o) \text{ means } (\neg \forall x_\alpha \neg A_o).
Church’s Type Theory:

Simply typed $\lambda$-calculus with the signature

$$\{\neg, \vee\} \cup \{\Pi^\alpha \mid \alpha \in T\}$$

(and perhaps a description or choice operator).
Church’s Type Theory:

- Simply typed λ-calculus with the signature

\[ \{\neg, \lor\} \cup \{\Pi^\alpha \mid \alpha \in \mathcal{T}\} \]

(and perhaps a description or choice operator).

- Axioms of Extensionality
Church’s Type Theory:

- Simply typed $\lambda$-calculus with the signature
  \[ \{\neg, \vee\} \cup \{\Pi^\alpha \mid \alpha \in T\} \]
  (and perhaps a description or choice operator).
- Axioms of Extensionality
- Axiom of Description or Choice
Church’s Type Theory:

- Simply typed $\lambda$-calculus with the signature

  \[
  \{\neg, \lor\} \cup \{\Pi^\alpha \mid \alpha \in \mathcal{T}\}
  \]

  (and perhaps a description or choice operator).

- Axioms of Extensionality

- Axiom of Description or Choice

- Axiom of Infinity
Church’s Type Theory

Church’s Type Theory:

- Simply typed $\lambda$-calculus with the signature

$$\{\neg, \vee\} \cup \{\Pi^\alpha \mid \alpha \in T\}$$

(and perhaps a description or choice operator).

- Axioms of Extensionality
- Axiom of Description or Choice
- Axiom of Infinity

Elementary Type Theory ($\mathcal{T}_{PS}$ – for automated theorem proving)
Church’s Type Theory:

- Simply typed $\lambda$-calculus with the signature

\[ \{\neg, \lor\} \cup \{\Pi^\alpha \mid \alpha \in T\}\]

(and perhaps a description or choice operator).

- Axioms of Extensionality

- Axiom of Description or Choice

- Axiom of Infinity

Extensional Type Theory ($\text{TPS}$ and $\text{LEO}$ – automated theorem proving)
Multi-Modal Logic

“Propositional” Symbols:

\[ PROP = \{ P, Q, \ldots \} \]
“Propositional” Symbols:

\[PROP = \{P, Q, \ldots\}\]

“Modalities”:

\[MOD = \{R, S, \ldots\}\]
Multi-Modal Logic

“Propositional” Symbols:

\[ PROP = \{ P, Q, \ldots \} \]

“Modalities”:

\[ MOD = \{ R, S, \ldots \} \]

Multi-Modal WFF’s (\( \varphi, \psi, \ldots \)):

\[ P \neg \varphi \lor \psi \langle R \rangle \varphi \langle [R] \varphi \]
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P$ with some predicate $P$.
- Associate each $R$ with some relation $R$.
- $ST_x(P(x)) = P(x)$.
- $ST_x(R(x; y)) = R(x; y)$.
- $ST_x(hRi) = \exists y (R(x; y))$.
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$. 

Note: $ST$ translates to a predicate on $x$. 
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.
- $ST_x(P) = \overline{P}(x)$
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.
- $ST_x(P) = \overline{P}(x)$
- $ST_x(\neg \varphi) = \neg ST_x(\varphi)$
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in \text{PROP}$ with some predicate $\overline{P}$.
- Associate each $R \in \text{MOD}$ with some relation $\overline{R}$.
- $ST_x(P) = \overline{P}(x)$
- $ST_x(\neg \varphi) = \neg ST_x(\varphi)$
- $ST_x(\varphi \lor \psi) = ST_x(\varphi) \lor ST_x(\psi)$
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.
- $ST_x(P) = \overline{P}(x)$
- $ST_x(\neg \varphi) = \neg ST_x(\varphi)$
- $ST_x(\varphi \lor \psi) = ST_x(\varphi) \lor ST_x(\psi)$
- $ST_x(\langle R \rangle \varphi) = \exists y(\overline{R}(x, y) \land ST_y(\varphi))$
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.

$ST_x(P) = \overline{P}(x)$

$ST_x(\neg \varphi) = \neg ST_x(\varphi)$

$ST_x(\varphi \lor \psi) = ST_x(\varphi) \lor ST_x(\psi)$

$ST_x(\langle R \rangle \varphi) = \exists y(\overline{R}(x, y) \land ST_y(\varphi))$

$ST_x([R] \varphi) = \forall y(\overline{R}(x, y) \supset ST_y(\varphi))$
Multi-Modal Logic

Standard Translation to First-Order (relative to $x$):

- Associate each $P \in PROP$ with some predicate $\overline{P}$.
- Associate each $R \in MOD$ with some relation $\overline{R}$.
- $ST_x(P) = \overline{P}(x)$
- $ST_x(\neg \varphi) = \neg ST_x(\varphi)$
- $ST_x(\varphi \lor \psi) = ST_x(\varphi) \lor ST_x(\psi)$
- $ST_x(\langle R \rangle \varphi) = \exists y(\overline{R}(x, y) \land ST_y(\varphi))$
- $ST_x([R] \varphi) = \forall y(\overline{R}(x, y) \supset ST_y(\varphi))$

Note: $ST$ translates to a predicate on $x$. 
Multi-Modal Logic

Translation to Higher-Order (map to type $o_\ell$):
Multi-Modal Logic

Translation to Higher-Order (map to type $o_{\ell}$):

- Associate each $P \in PROP$ with some predicate $\overline{P}_{o_{\ell}}$. 
Multi-Modal Logic

Translation to Higher-Order (map to type $o_l$):

- Associate each $P \in PROP$ with some predicate $\overline{P}_{ol}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{oll}$. 
Multi-Modal Logic

Translation to Higher-Order (map to type $o_\ell$):

- Associate each $P \in PROP$ with some predicate $\overline{P}_{o_\ell}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{o_\ell}$.
- $HST(P) = \overline{P}$
Multi-Modal Logic

Translation to Higher-Order (map to type $o_{\ell}$):

- Associate each $P \in PROP$ with some predicate $P_{o_{\ell}}$.
- Associate each $R \in MOD$ with some relation $R_{o_{\ell}}$.
- $HST(P) = P$
- $HST(\neg \varphi) = (\lambda x_{\ell} \neg (HST(\varphi)x))$
Multi-Modal Logic

Translation to Higher-Order (map to type $o\ell$):

- Associate each $P \in PROP$ with some predicate $\overline{P}_{o\ell}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{o\ell}$.
- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\lambda x_l \neg (HST(\varphi)x))$
- $HST(\varphi \lor \psi) = (\lambda x_l ((HST(\varphi)x) \lor (HST(\psi)x)))$
Multi-Modal Logic

Translation to Higher-Order (map to type $o_\ell$):

- Associate each $P \in PROP$ with some predicate $P_{o_\ell}$.
- Associate each $R \in MOD$ with some relation $R_{o_\ell}$.

- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\lambda x_i \neg (HST(\varphi)x))$
- $HST(\varphi \lor \psi) = (\lambda x_i ((HST(\varphi)x) \lor (HST(\psi)x)))$
- $HST(\langle R \rangle \varphi) = \lambda x_i \exists y_i ((\overline{R}xy) \land (HST(\varphi)y))$
Multi-Modal Logic

Translation to Higher-Order (map to type $o_\ell$):

- Associate each $P \in PROP$ with some predicate $\overline{P}_{o_\ell}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{o_\ell}$.

$$HST(P) = \overline{P}$$

$$HST(\neg \varphi) = (\lambda x_{\ell} \neg (HST(\varphi)x))$$

$$HST(\varphi \lor \psi) = (\lambda x_{\ell} ((HST(\varphi)x) \lor (HST(\psi)x)))$$

$$HST(\langle R \rangle \varphi) = \lambda x_{\ell} \exists y_{\ell}((\overline{R}xy) \land (HST(\varphi)y))$$

$$HST([R]\varphi) = \lambda x_{\ell} \forall y_{\ell}((\overline{R}xy) \supset (HST(\varphi)y))$$
Multi-Modal Logic

Translation to Higher-Order (map to type $o_l$):

- Associate each $P \in PROP$ with some predicate $P_{o_l}$.
- Associate each $R \in MOD$ with some relation $R_{o_l}$.
- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\lambda x_i \neg (HST(\varphi)x))$
- $HST(\varphi \lor \psi) = (\lambda x_i ((HST(\varphi)x) \lor (HST(\psi)x)))$
- $HST(\langle R \rangle \varphi) = \lambda x_i \exists y_i ((\overline{R}x y) \land (HST(\varphi)y))$
- $HST(\lbrack R \rbrack \varphi) = \lambda x_i \forall y_i ((\overline{R}x y) \supset (HST(\varphi)y))$
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_{oL}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{oL}$.
- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\lambda x_i (\neg (HST(\varphi)x)))$
- $HST(\varphi \lor \psi) = (\lambda x_i ((HST(\varphi)x) \lor (HST(\psi)x)))$
- $HST([R] \varphi) = \lambda x_i \forall y_i ((\overline{R} x y) \supset (HST(\varphi)y))$
- $HST(\langle R \rangle \varphi) = \lambda x_i \exists y_i ((\overline{R} x y) \land (HST(\varphi)y))$
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_{ov}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{ov}$.
- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\lambda x_i \neg (HST(\varphi) x))$
- $HST(\varphi \lor \psi) = (\lambda x_i ((HST(\varphi) x) \lor (HST(\psi) x)))$
- $HST(\langle R \rangle \varphi) = \lambda x_i \exists y_i ((\overline{R} x y) \land (HST(\varphi) y))$
- $HST(\lbrack R \rbrack \varphi) = \lambda x_i \forall y_i ((\overline{R} x y) \supset (HST(\varphi) y))$
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_{ol}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{ol}$.

\[
HST(P) = \overline{P}
\]

\[
HST(\neg \varphi) = (\neg HST(\varphi))
\]

\[
HST(\varphi \lor \psi) = (\lambda x_i ((HST(\varphi)x) \lor (HST(\psi)x)))
\]

\[
HST([R]\varphi) = \lambda x_i \exists y_i ((\overline{R}x y) \land (HST(\varphi)y))
\]

\[
HST(\langle R \rangle \varphi) = \lambda x_i \forall y_i ((\overline{R}x y) \supset (HST(\varphi)y))
\]

$\neg_{ol(ol)}$ is $(\lambda U_{ol} \lambda x_i \neg(Ux))$ (Complement)
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_o$.
- Associate each $R \in MOD$ with some relation $\overline{R}_o$.
- $\text{HST}(P) = \overline{P}$
- $\text{HST}(\neg \varphi) = (\neg \text{HST}(\varphi))$
- $\text{HST}(\varphi \lor \psi) = (\lambda x_i ((\text{HST}(\varphi) x) \lor (\text{HST}(\psi) x)))$
- $\text{HST}([R]\varphi) = \lambda x_i \exists y_i ((\overline{R} x y) \land (\text{HST}(\varphi) y))$
- $\text{HST}([R]\varphi) = \lambda x_i \forall y_i ((\overline{R} x y) \supset (\text{HST}(\varphi) y))$
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_{ol}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{ol}$.
- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\neg HST(\varphi))$
- $HST(\varphi \lor \psi) = (HST(\varphi) \lor HST(\psi))$
- $HST([R] \varphi) = \lambda x_1 \exists y_1((\overline{R} x y) \land (HST(\varphi) y))$
- $HST(\langle R \rangle \varphi) = \lambda x_1 \forall y_1((\overline{R} x y) \supset (HST(\varphi) y))$

$\lor_{ol(ol)} \text{ is } (\lambda U_{ol} \lambda V_{ol} \lambda x_1 . (U x) \lor (V x)) \text{ (Union)}$
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_{ov}$.
- Associate each $R \in MOD$ with some relation $\overline{R}_{ov}$.

- $HST(P) = \overline{P}$
- $HST(\neg \varphi) = (\neg HST(\varphi))$
- $HST(\varphi \lor \psi) = (HST(\varphi) \lor HST(\psi))$
- $HST(\langle R \rangle \varphi) = \lambda x_i \exists y_i ((\overline{R} x y) \land (HST(\varphi) y))$
- $HST([R] \varphi) = \lambda x_i \forall y_i ((\overline{R} x y) \supset (HST(\varphi) y))$
Encoding Multi-Modal Logic

- Associate each \( P \in PROP \) with some predicate \( \overline{P}_{o_1} \).
- Associate each \( R \in MOD \) with some relation \( \overline{R}_{o_2} \).

\[
HST(P) = \overline{P}
\]

\[
HST(\neg \varphi) = (\neg HST(\varphi))
\]

\[
HST(\varphi \lor \psi) = (HST(\varphi) \lor HST(\psi))
\]

\[
HST(\langle R \rangle \varphi) = (\Diamond \overline{R} \ HST(\varphi))
\]

\[
HST([R] \varphi) = \lambda x_i \forall y_i ((\overline{R} x y) \supset (HST(\varphi) y))
\]

\[
\Diamond_{o_1(o_1)} \ \text{is} \ (\lambda R_{o_2} \lambda U_{o_1} \lambda x_i \exists y_i . R x y \land U y)
\]
Encoding Multi-Modal Logic

- Associate each $P \in PROP$ with some predicate $\overline{P}_o$.
- Associate each $R \in MOD$ with some relation $\overline{R}_o$.
- $HST(P) = \overline{P}$
- $HST(\neg \phi) = (\neg HST(\phi))$
- $HST(\phi \lor \psi) = (HST(\phi) \lor HST(\psi))$
- $HST(\langle R \rangle \phi) = (\Diamond \overline{R} \ HST(\phi))$
- $HST([R] \phi) = \lambda x_1 \forall y_1 ((\overline{R} x y) \supset (HST(\phi) y))$
Associate each $P \in PROP$ with some predicate $\overline{P}_{o\ell}$.

Associate each $R \in MOD$ with some relation $\overline{R}_{o\ell}$.

$HST(P) = \overline{P}$

$HST(\neg \varphi) = (\neg HST(\varphi))$

$HST(\varphi \vee \psi) = (HST(\varphi) \vee HST(\psi))$

$HST(\langle R \rangle \varphi) = (\diamond \overline{R} \ HST(\varphi))$

$HST([R] \varphi) = (\Box \overline{R} \ HST(\varphi))$

$\Box_{o\ell(o\ell)(o\ell)} \text{ is } (\lambda R_{o\ell} \lambda U_{o\ell} \lambda x \lambda y . R x y \supset U y)$
Associate each $P \in PROP$ with some predicate $\overline{P}_o$.

Associate each $R \in MOD$ with some relation $\overline{R}_o$.

$HST(P) = \overline{P}$

$HST(\neg \varphi) = (\neg HST(\varphi))$

$HST(\varphi \lor \psi) = (HST(\varphi) \lor HST(\psi))$

$HST(\langle R \rangle \varphi) = (\Diamond \overline{R} HST(\varphi))$

$HST([R] \varphi) = (\Box \overline{R} HST(\varphi))$

Using these definitions, the translation is trivial.
Multi-Modal Fragment of Higher-Order

\[ PROP = \{ P, Q, \ldots \} \subseteq V_{ol} \cup P_{ol} \]
Multi-Modal Fragment of Higher-Order

\[ PROP = \{P, Q, \ldots\} \subseteq \mathcal{V}_{ol} \cup \mathcal{P}_{ol} \]

\[ MOD = \{R, S, \ldots\} \subseteq \mathcal{V}_{ol} \cup \mathcal{P}_{ol} \]
Multi-Modal Fragment of Higher-Order

\[ PROP = \{ P, Q, \ldots \} \subseteq V_{ol} \cup P_{ol} \]

\[ MOD = \{ R, S, \ldots \} \subseteq V_{ol} \cup P_{ol} \]

Multi-Modal Formulas are certain terms \((\varphi, \psi, \ldots)\) of type \(ol\):

\[ P | \neg \varphi | (\varphi \lor \psi) | (\Diamond R \varphi) | (\Box R \varphi) \]
Multi-Modal Fragment of Higher-Order

\[ \text{PROP} = \{P, Q, \ldots\} \subseteq V_{ol} \cup P_{ol} \]

\[ \text{MOD} = \{R, S, \ldots\} \subseteq V_{ol} \cup P_{ol} \]

Multi-Modal Formulas are certain terms \((\varphi, \psi, \ldots)\) of type \(oI\):

\[ P|\neg\varphi|(\varphi \lor \psi)|(\Diamond R \varphi)|(\Box R \varphi) \]

No inductive translation is required.
Two directions:

1. Generalize the types.
2. Extend to Hybrid Logic.
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots \} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha}$$
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots\} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha}$$

(Relations between $\alpha$ and $\beta$)

$$MOD^{\alpha,\beta} = \{R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots\} \subseteq \mathcal{V}_{o\beta\alpha} \cup \mathcal{P}_{o\beta\alpha}$$
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots\} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha}$$

(Relations between $\alpha$ and $\beta$)

$$MOD^{\alpha,\beta} = \{R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots\} \subseteq \mathcal{V}_{o\beta\alpha} \cup \mathcal{P}_{o\beta\alpha}$$

$\alpha$-Multi-Modal Formulas $MMF^\alpha$ (terms of type $o\alpha$):

$$P_{o\alpha} \vdash \lnot^\alpha \varphi \vdash (\varphi \lor \psi) \vdash (\diamond^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta}) \vdash (\square^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta})$$
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots\} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha}$$

(Relations between $\alpha$ and $\beta$)

$$MOD^{\alpha,\beta} = \{R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots\} \subseteq \mathcal{V}_{o\beta\alpha} \cup \mathcal{P}_{o\beta\alpha}$$

$\alpha$-Multi-Modal Formulas $MMF^\alpha$ (terms of type $o\alpha$):

$$P_{o\alpha} \vdash^{\alpha} \neg \varphi | (\varphi \lor^{\alpha} \psi) | (\Diamond^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta}) | (\Box^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta})$$

$\neg^{\alpha} is$ $\lambda U_{o\alpha} \lambda x_{\alpha} \neg (U x)$
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots\} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha}$$

(Relations between $\alpha$ and $\beta$)

$$MOD^{\alpha,\beta} = \{R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots\} \subseteq \mathcal{V}_{o\beta\alpha} \cup \mathcal{P}_{o\beta\alpha}$$

$\alpha$-Multi-Modal Formulas $MMF^\alpha$ (terms of type $o\alpha$):

$$P_{o\alpha} \vdash \lnot^\alpha \varphi | (\varphi \lor^\alpha \psi) | (\Diamond^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta}) | (\Box^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta})$$

$\lor^\alpha$ is $$(\lambda U_{o\alpha} \lambda V_{o\alpha} \lambda x. (U x) \lor (V x))$$
Generalizing the Types

(Properties on type $\alpha$)

$$PROP^\alpha = \{P_{o\alpha}, Q_{o\alpha}, \ldots\} \subseteq V_{o\alpha} \cup P_{o\alpha}$$

(Relations between $\alpha$ and $\beta$)

$$MOD^{\alpha,\beta} = \{R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots\} \subseteq V_{o\beta\alpha} \cup P_{o\beta\alpha}$$

$\alpha$-Multi-Modal Formulas $MMF^\alpha$ (terms of type $o\alpha$):

$$P_{o\alpha} \models \neg^\alpha \varphi \models (\varphi \lor^\alpha \psi) \models (\diamond^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta}) \models (\Box^{\alpha,\beta} R_{o\beta\alpha} \varphi_{o\beta})$$

$\diamond^{\alpha,\beta}$ is $(\lambda R_{o\beta\alpha} \lambda U_{o\beta} \lambda x_\alpha \exists y_\beta . R x y \land U y)$
Generalizing the Types

(Properties on type $\alpha$)

\[ \text{PROP}^\alpha = \{ P_{o\alpha}, Q_{o\alpha}, \ldots \} \subseteq \mathcal{V}_{o\alpha} \cup \mathcal{P}_{o\alpha} \]

(Relations between $\alpha$ and $\beta$)

\[ \text{MOD}^{\alpha,\beta} = \{ R_{o\beta\alpha}, S_{o\beta\alpha}, \ldots \} \subseteq \mathcal{V}_{o\beta\alpha} \cup \mathcal{P}_{o\beta\alpha} \]

$\alpha$-Multi-Modal Formulas $\text{MMF}^\alpha$ (terms of type $o\alpha$):

\[ P_{o\alpha} \vdash^{\alpha} \varphi \vdash (\varphi \lor^{\alpha} \psi) \vdash (\diamond^{\alpha,\beta} R_{o\beta\alpha} \varphi) \vdash (\square^{\alpha,\beta} R_{o\beta\alpha} \varphi) \]

$\square^{\alpha,\beta}$ is $\lambda R_{o\beta\alpha} \lambda U_{o\beta} \lambda x_{\alpha} \forall y_{\beta}. R x y \supset U y$
Examples

Let $\in_{\omega(\omega)}$ stand for $[\lambda U_{\omega} \lambda x_\alpha. U x]$. 
Examples

Let \( \bar{\in}^{o\alpha}_{o\alpha} \) stand for \([\lambda U_{o\alpha} \lambda x_{\alpha}. U x]\).

Intuitively, \((\bar{\in} U x)\) means \(x \in U\).
Examples

Let \( \epsilon_{o\alpha(o\alpha)} \) stand for \( \lambda U_{o\alpha} \lambda x_{\alpha}.Ux \).
Intuitively, \( (\epsilon \; U \; x) \) means \( x \in U \).
Let \( P_{o\alpha} \in PROP^{\alpha} \).
Examples

Let $\bar{\in}_{o\alpha(o\alpha)}$ stand for $[\lambda U_{o\alpha} \lambda x_{\alpha}. U x]$.
Intuitively, $(\bar{\in} U x)$ means $x \in U$.
Let $P_{o\alpha} \in PROP^{o\alpha}$.

$[\bar{\in}] P$ is in $MMF^{o\alpha}$
Examples

Let $\overline{\in}_{o\alpha(o\alpha)}$ stand for $[\lambda U_{o\alpha}\lambda x_{\alpha}.U x]$. Intuitively, $(\overline{\in} U x)$ means $x \in U$.

Let $P_{o\alpha} \in PROP^\alpha$.

- $[\overline{\in}]P$ is in $MMF^{o\alpha}$
  
  This is true at $Q_{o\alpha}$ iff $Q$ is a subset of $P$. TPS can prove equivalence automatically (expanding definitions and working in higher-order logic).
Examples

Let $\bar{\in}_{\circ\alpha (o\alpha)}$ stand for $[\lambda U_{o\alpha} \lambda x_{\alpha}. Ux]$. Intuitively, $(\bar{\in} U x)$ means $x \in U$. Let $P_{o\alpha} \in PROP_{\alpha}$.

- $[\bar{\in}] P$ is in $MMF^{o\alpha}$
- $[\bar{\in}](P \land \neg P)$
Examples

Let $\bar{e}_{o\alpha(o\alpha)}$ stand for $[\lambda U_{o\alpha} \lambda x_{\alpha}. U x]$.

Intuitively, $(\bar{e} U x)$ means $x \in U$.

Let $P_{o\alpha} \in PROP^{\alpha}$.

- $[\bar{e}] P$ is in $MMF^{o\alpha}$
- $[\bar{e}] (P \land \neg P)$

This is true at $Q_{o\alpha}$ iff $Q$ is empty. Automatic Proof by TPS
Examples

Let \( \varepsilon_{o\alpha(o\alpha)} \) stand for \( [\lambda x_{\alpha}\lambda U_{o\alpha}.Ux] \).
Examples

Let $\in_{o\alpha(\alpha)}$ stand for $[\lambda x_\alpha \lambda U_{o\alpha}. U x]$. Intuitively, $(\in x U)$ means $x \in U$. 
Examples

Let $\in_{o\alpha(o\alpha)}$ stand for $[\lambda x_\alpha \lambda U_{o\alpha}. Ux]$.
Let $OPEN_{o(o\alpha)} \in PROP^{o\alpha}$. 
Examples

Let $\varepsilon_{o\alpha(o\alpha)}$ stand for $[\lambda x_{\alpha} \lambda U_{o\alpha}. U x]$.
Let $OPEN_{o(o\alpha)} \in PROP^{o\alpha}$.

\[\langle \varepsilon \rangle OPEN \text{ is in } MMF^{\alpha}\]
Examples

Let $\in_{o\alpha(o\alpha)}$ stand for $[\lambda x_\alpha \lambda U_{o\alpha}. Ux]$.
Let $OPEN_{o(o\alpha)} \in PROP^{o\alpha}$.

$\langle \in \rangle OPEN$ is in $MMF^\alpha$
This is true at $x_\alpha$ iff $x$ is in some $P$ in $OPEN$. 

Examples

Let \( \in_{\alpha(o\alpha)} \) stand for \([\lambda x_{\alpha} \lambda U_{o\alpha}.Ux]\).
Let \( OPEN_{o(o\alpha)} \in PROP^o\alpha \).

\[ (\in)OPEN \text{ is in } MMF^\alpha \]
This is true at \( x_{\alpha} \) iff \( x \) is in some \( P \) in \( OPEN \).
Represents union of a collection
Examples

Let $\in_{\alpha}(o_{\alpha})$ stand for $[\lambda x_{\alpha} \lambda U_{\alpha}. U x]$. Let $OPEN_{o(\alpha)} \in PROP^{\alpha}$.

$\langle \in \rangle OPEN$ is in $MMF^{\alpha}$
This is true at $x_{\alpha}$ iff $x$ is in some $P$ in $OPEN$. Represents union of a collection
(Automatic proof of equivalence by TPS)
Hybrid Logic

Multi-Modal Plus Nominals:

\[ NOM = \{ i, j, \ldots \} \]
Hybrid Logic

Multi-Modal Plus Nominals:

\[ NOM = \{ i, j, \ldots \} \]

“At” operator:

\[ \@_i \varphi \]
Hybrid Logic

Multi-Modal Plus Nominals:

\[ NOM = \{ i, j, \ldots \} \]

“At” operator:

\[ @_i \varphi \]

Maybe downarrow binder:

\[ \downarrow x. \varphi(x) \]
First-Order Translation

- Associate nominals $i$ with variable $\bar{i}$.
- $ST_x(i) = (x = \bar{i})$
First-Order Translation

- Associate nominals $i$ with variable $\overline{i}$.
- $ST_x(i) = (x = \overline{i})$
- $ST_x(@_i \varphi) = [\overline{i}/x] ST_x(\varphi)$
First-Order Translation

- Associate nominals $i$ with variable $\overline{i}$.

- $ST_x(i) = (x = \overline{i})$

- $ST_x(@i\varphi) = [\overline{i}/x]ST_x(\varphi)$

- $ST_x(y) = (x = y)$ (y is a “nominal variable”)
First-Order Translation

- Associate nominals $i$ with variable $\bar{i}$.
- $ST_x(i) = (x = \bar{i})$
- $ST_x(@i\varphi) = [\bar{i}/x]ST_x(\varphi)$
- $ST_x(y) = (x = y)$ (y is a “nominal variable”)
- $ST_x(\downarrow y.\varphi) = [x/y]ST_x(\varphi)$
Nominals at type $\alpha$:

$$NOM^{\alpha} = \{i_\alpha, j_\alpha, \ldots\} \subseteq V_\alpha \cup P_\alpha$$
Encoding Hybrid Logic with General Types

Nominals at type $\alpha$:

$$NOM^\alpha = \{i_\alpha, j_\alpha, \ldots\} \subseteq \mathcal{V}_\alpha \cup \mathcal{P}_\alpha$$

$\alpha$-Hybrid Formula $HF^\alpha$ is a term of type $o_\alpha$ constructed as $\alpha$-Multi-Modal Plus:
Nominals at type $\alpha$:

$$NOM^\alpha = \{i_\alpha, j_\alpha, \ldots\} \subseteq \mathcal{V}_\alpha \cup \mathcal{P}_\alpha$$

$\alpha$-Hybrid Formula $HF^\alpha$ is a term of type $o\alpha$ constructed as $\alpha$-Multi-Modal Plus:

$$\bigcirc (U^{\alpha} i_\alpha)$$ where $U^{\alpha}$ is $(\lambda x_\alpha \lambda y_\alpha (x = y))$
Nominals at type $\alpha$:

$$NOM^\alpha = \{i_\alpha, j_\alpha, \ldots\} \subseteq \mathcal{V}_\alpha \cup \mathcal{P}_\alpha$$

$\alpha$-Hybrid Formula $HF^\alpha$ is a term of type $o_\alpha$ constructed as $\alpha$-Multi-Modal Plus:

- $(U^\alpha i_\alpha)$ where $U^\alpha$ is $(\lambda x_\alpha \lambda y_\alpha (x = y))$

- $(@^{\alpha,\beta} j_\beta \varphi_{o_\beta})_{o_\alpha}$ where $@^{\alpha,\beta}$ is $\lambda z_\beta \lambda V_{o_\beta} \lambda x_\alpha. V z$

(Note this does not depend $x$.)
Encoding Hybrid Logic with General Types

Nominals at type $\alpha$:

$$NOM^\alpha = \{i_\alpha, j_\alpha, \ldots\} \subseteq \mathcal{V}_\alpha \cup \mathcal{P}_\alpha$$

$\alpha$-Hybrid Formula $HF^\alpha$ is a term of type $o\alpha$ constructed as $\alpha$-Multi-Modal Plus:

- $(U^\alpha i_\alpha)$ where $U^\alpha$ is $(\lambda x_\alpha \lambda y_\alpha (x = y))$

- $(\@^{\alpha, \beta} j_\beta \varphi_{o\beta})_{o\alpha}$ where $\@^{\alpha, \beta}$ is $\lambda z_\beta \lambda V_{o\beta} \lambda x_\alpha. V z$

- $(\downarrow^\alpha (\lambda i_\alpha \cdot \varphi_{o\alpha}))$ (where $i_\alpha$ is a “nominal variable”)

where $\downarrow^\alpha$ is $\lambda W_{o\alpha\alpha} \cdot \lambda x_\alpha (W x x)$
Examples

\[ P_{o\alpha} \in PROP^\alpha \]
Examples

\[ P_{o\alpha} \in PROP^\alpha \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]
Examples

\[ P_{o\alpha} \in PROP^{\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

\[ (U^{o\alpha} e) \supset [\in](P \land \neg P) \]
Examples

\[ P_{o\alpha} \in PROP^{\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

\[(U^{o\alpha}e) \supset [\exists](P \land \neg P)\]
Valid if \(e\) is empty.
Examples

\[ P_{o\alpha} \in PROP^{o\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

- \((U^{o\alpha} e) \supset [\in](P \land \neg P)\)
  
  Valid if \(e\) is empty.

- \((U^{o\alpha} e) \equiv [\in](P \land \neg P)\)
Examples

\[ P_{o\alpha} \in PROP^{\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

\[ (U^{o\alpha}e) \supset [\in](P \land \neg P) \]
Valid if \( e \) is empty.

\[ (U^{o\alpha}e) \equiv [\in](P \land \neg P) \]
Valid if \( e \) is the unique empty set (extensionality).
Examples

\[ P_{o\alpha} \in PROP^{o\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

- \( (U^{o\alpha}e) \supset [\in](P \land \neg P) \)
  Valid if \( e \) is empty.

- \( (U^{o\alpha}e) \equiv [\in](P \land \neg P) \)
  Valid if \( e \) is the unique empty set (extensionality).

- \( @^{\beta,(o\alpha)} e [\in] (P \land \neg P) \)
Examples

\[ P_{o\alpha} \in PROP^{o\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

\begin{itemize}
  \item \((U^{o\alpha} e) \supset [\in] (P \land \neg P)\)
    Valid if \(e\) is empty.
  \item \((U^{o\alpha} e) \equiv [\in] (P \land \neg P)\)
    Valid if \(e\) is the \textit{unique} empty set (extensionality).
  \item @\((\beta, o\alpha) e [\in] (P \land \neg P)\)
    True at \(b_\beta\) iff \(e\) is empty (does not depend on \(b\)).
\end{itemize}
Examples

\[ P_{o\alpha} \in PROP^{\alpha} \]
\[ e_{o\alpha} \in NOM^{o\alpha} \]

- \((U^{o\alpha} e) \supset [\in](P \land \neg P)\)
  Valid if \(e\) is empty.

- \((U^{o\alpha} e) \equiv [\in](P \land \neg P)\)
  Valid if \(e\) is the unique empty set (extensionality).

- \(\@^{\beta,(o\alpha)} e [\in] (P \land \neg P)\)
  True at \(b_{\beta}\) iff \(e\) is empty (does not depend on \(b\)).

Automatic proofs in \(TPS\)
Using $\lambda$-terms, the standard translation from Multi-Modal (and Hybrid) Logic to first-order logic becomes an easier translation into higher-order logic.
Conclusion

Using $\lambda$-terms, the standard translation from Multi-Modal (and Hybrid) Logic to first-order logic becomes an easier translation into higher-order logic.

The translation shows Hybrid Logic is a natural fragment of higher-order logic.
Conclusion

- Using $\lambda$-terms, the standard translation from Multi-Modal (and Hybrid) Logic to first-order logic becomes an easier translation into higher-order logic.

- The translation shows Hybrid Logic is a natural fragment of higher-order logic.

- By generalizing over types, we obtain a typed form of Hybrid Logic.