Higher Pro-arrows: Towards a Model for Naturality Pretype Theory

Andreas Nuyts

KU Leuven, Belgium

HoTT/UF ’23
Vienna, Austria
April 23, 2023
Naturality TT: **Why?** (And what?)
Example problem in verified functional programming
\[ f : A \rightarrow B \]

\[ \text{Monad Type Monoid} \times \]

\[ M \quad N \quad \text{Monad} \]

\[ \text{List } A \quad \text{WriterT } (\text{List } A) \ M \quad \text{WriterT } (\text{List } A) \ N \]

\[ \text{List } B \quad \text{WriterT } (\text{List } B) \ M \quad \text{WriterT } (\text{List } B) \ N \]

\[ \text{Type Monoid} \quad \times \]
\[
\begin{array}{cccc}
M & N & \text{Monad} \\
\text{List } A & \text{WriterT} (\text{List } A) M & \text{WriterT} (\text{List } A) N \\
\text{List } B & \text{WriterT} (\text{List } B) M & \text{WriterT} (\text{List } B) N \\
\end{array}
\]
Monad

\[
\begin{align*}
A & \xrightarrow{f} B \\
\text{List } A & \xrightarrow{\text{List } f} \text{List } B \\
\text{WriterT (List } A \text{) } M & \xrightarrow{\text{WriterT (List } A \text{) } N} \\
\text{List } B & \xrightarrow{\text{List } f} \text{List } B \\
\text{WriterT (List } B \text{) } M & \xrightarrow{\text{WriterT (List } B \text{) } N}
\end{align*}
\]

\[\text{Monad} \qquad \text{Monad}\]

\[\text{Type} \xrightarrow{\text{List}} \text{Monoid}\]
\[\begin{align*}
M & \xrightarrow{g} N \\
A & \xrightarrow{f} B \\
\text{List } A & \xrightarrow{\text{List } f} \text{List } B \\
\text{List } (\text{List } A) M & \xrightarrow{\text{List } (\text{List } f) M} \text{List } (\text{List } B) M \\
\text{List } (\text{List } A) N & \xrightarrow{\text{List } (\text{List } f) N} \text{List } (\text{List } B) N
\end{align*}\]

\[\text{Monad} \quad \text{Monad} \quad \text{Monad}\]

\[\text{Type} \xrightarrow{\text{List}} \text{Monoid}\]

\[\in\]
A \xrightarrow{f} B
\downarrow \quad \downarrow
\text{List} f \quad \text{List} f
\text{List} A \quad \text{List} B
\text{WriterT (List A) M} \quad \text{WriterT (List A) N}
\text{WriterT (List B) M} \quad \text{WriterT (List B) N}
\text{Monad}
\text{Type} \xrightarrow{\text{List}} \text{Monoid}
\[ M \xrightarrow{g} N \]

\[ \text{Monad} \]

\[ \text{Type} \xrightarrow{\text{List}} \text{Monoid} \]
In plain DTT

Functoriality of \( \text{List} : \text{Type} \rightarrow \text{Monoid} \):
- Object action: \((\text{List} A, [], +++)\)
- Functorial action:
  - \( \text{List} f : \text{List} A \rightarrow \text{List} B \) (by recursion)
  - \( \text{List} f \) is a monoid morphism:
    - \( \text{List} f \) preserves \([]\) (trivial)
    - \( \text{List} f \) preserves \(++\) (by induction)
- + functor laws (by induction)

Functoriality of \( \text{WriterT} : \text{Monoid} \rightarrow \text{MonadTrans} \):
- Object action: \( \text{WriterT} W \in \text{MonadTrans} \)
  - Object action: \( \text{WriterT} W M \in \text{Monad} \)
  - Object action: Define \( \text{WriterT} W M A \)
  - Functorial action \( \text{WriterT} W M f \)
    - Functorial action laws
    - return & bind + naturality
- + functor laws

... Object action: \( \text{WriterT} W \in \text{MonadTrans} \)
- Functorial action \( \text{WriterT} W g \)
  - Respects return & bind
  - + functor laws
  - lift : \( M \rightarrow \text{WriterT} W M \) + naturality
    - Respects return & bind
- Funktorial action:
  - \( \text{WriterT} h : \text{WriterT} V \rightarrow \text{WriterT} W \)
  - \( \text{WriterT} h M A \)
    - Respects return, bind & lift
    - naturality w.r.t. \( A \)
    - naturality w.r.t. \( M \)
- + functor laws
Functoriality of List : \(\text{Type} \to \text{Monoid}\):

- **Object action**: \((\text{List} \, A, [], \text{++})\)
- **Functorial action**:
  - \(\text{List} \, f : \text{List} \, A \to \text{List} \, B\) *(by recursion)*
  - \(\text{List} \, f\) is a monoid morphism:
    - \(\text{List} \, f\) preserves \([]\) *(trivial)*
    - \(\text{List} \, f\) preserves \(\text{++}\) *(by induction)*

  + functor laws *(by induction)*

Functoriality of \(\text{WriterT} : \text{Monoid} \to \text{MonadTrans}\):

- **Object action**: \(\text{WriterT} \, W \in \text{MonadTrans}\)
- **Functorial action**:
  - \(\text{Functorial action} \, \text{WriterT} \, W \, g\)
    - Respects return & bind
    + functor laws
  - \(\text{lift} : M \to \text{WriterT} \, W \, M + \text{naturality}\)
    - Respects return & bind
- **Functorial action**:
  - \(\text{WriterT} \, h : \text{WriterT} \, V \to \text{WriterT} \, W\)
    - \(\text{WriterT} \, h \, M \, A\)
      - Respects return, bind & lift
    - naturality w.r.t. \(A\)
    - naturality w.r.t. \(M\)

  + functor laws

+ functor laws
In plain DTT

Functoriality of List : Type → Monoid:
- Object action: \((\text{List } A, [], ++)\)
- Functorial action:
  - \(\text{List } f : \text{List } A \to \text{List } B\) (by recursion)
  - \(\text{List } f\) is a monoid morphism:
    - \(\text{List } f\) preserves \([]\) (trivial)
    - \(\text{List } f\) preserves ++ (by induction)
- + functor laws (by induction)

Functoriality of WriterT : Monoid → MonadTrans
- Object action: \(\text{WriterT } W \in \text{MonadTrans}\)
  - Object action: \(\text{WriterT } W M \in \text{Monad}\)
    - Object action: Define \(\text{WriterT } W M A\)
    - Functorial action \(\text{WriterT } W M f\)
      - + functor laws
    - return & bind + naturality
- Functorial action: \(\text{WriterT } h : \text{WriterT } V \to \text{WriterT } W\)
  - \(\text{WriterT } h M A\)
    - Respects return, bind & lift
    - naturality w.r.t. \(A\)
    - naturality w.r.t. \(M\)
- + functor laws
Functoriality of \( \text{List} : \text{Type} \to \text{Monoid} \):

- Object action: \((\text{List } A, [], +++)\)
- Functorial action:
  - \(\text{List } f : \text{List } A \to \text{List } B\) (by recursion)
  - \(\text{List } f\) is a monoid morphism:
    - \(\text{List } f\) preserves [] (trivial)
    - \(\text{List } f\) preserves +++ (by induction)

  + functor laws (by induction)

Functoriality of \(\text{WriterT} : \text{Monoid} \to \text{MonadTrans}\):

- Object action: \(\text{WriterT } W \in \text{MonadTrans}\)
- Object action: \(\text{WriterT } W M \in \text{Monad}\)
- Object action: Define \(\text{WriterT } W M A\)
- Functorial action \(\text{WriterT } W M f\)
  + functor laws
- return & bind + \(\naturality\) naturality

  ... Object action: \(\text{WriterT } W \in \text{MonadTrans}\)
  
  - Functorial action \(\text{WriterT } W g\)
    - Respects return & bind
    + functor laws
    - \(\text{lift} : M \to \text{WriterT } W M + \naturality\)
    - Respects return & bind

  - Functorial action:
    - \(\text{WriterT } h : \text{WriterT } V \to \text{WriterT } W\)
      - \(\text{WriterT } h M A\)
        - Respects return, bind & lift
      - \(\naturality\) naturality w.r.t. \(A\)
      - \(\naturality\) naturality w.r.t. \(M\)

  + functor laws
In HoTT (assuming \( f, g \) and \( h = \text{List } f \) are isos)

Functoriality of \( \text{List} : \text{Type} \to \text{Monoid} \):
- Object action: \((\text{List } A, [], ++)\)
- \( \mathbin{\text{ Functorial action:}} \)
  - \( \mathbin{\text{ List } f : \text{List } A \cong \text{List } B \text{ (by-recursion)}} \)
  - \( \mathbin{\text{ List } f \text{ is a monoid morphism:}} \)
    - \( \mathbin{\text{ List } f \text{ preserves } [] \text{ (trivial)}} \)
    - \( \mathbin{\text{ List } f \text{ preserves } ++ \text{ (by-ind.)}} \)
  + \( \mathbin{\text{ functor laws (by-induction)}} \)

Functoriality of \( \text{WriterT} : \text{Monoid} \to \text{MonadTrans} \):
- Object action: \( \text{WriterT } W \in \text{MonadTrans} \)
  + \( \mathbin{\text{ Functorial action:}} \)
    - \( \mathbin{\text{ WriterT } W g \text{ respects return & bind}} \)
    + \( \mathbin{\text{ functor laws}} \)
    - \( \mathbin{\text{ lift } : M \to \text{WriterT } W M \text{ + functor naturality}} \)
      - \( \mathbin{\text{ Respects return & bind}} \)
  + \( \mathbin{\text{ Functorial action:}} \)
    - \( \mathbin{\text{ WriterT } h : \text{WriterT } V \cong \text{WriterT } W \text{ respects return, bind & lift}} \)
    - \( \mathbin{\text{ naturality w.r.t. } A} \)
    - \( \mathbin{\text{ naturality w.r.t. } M} \)
  + \( \mathbin{\text{ functor laws}} \)
Functoriality of List : Type → Monoid:

- Object action: \((\text{List } A, [], ++)\)
- Functorial action:
  - \(\text{List } f : \text{List } A \rightarrow \text{List } B\) \((\text{by recursion})\)
  - \(\text{List } f\) is a monoid morphism:
    - \(\text{List } f\) preserves \([]\) \((\text{trivial})\)
    - \(\text{List } f\) preserves ++ \((\text{by ind.})\)

+ \(\text{functor laws} \text{ (by induction)}\)

Functoriality of WriterT : Monoid → MonadTrans

- Object action: \(\text{WriterT } W \in \text{MonadTrans}\)
  - Object action: \(\text{Define WriterT } W M A\)
  - Functorial action \(\text{WriterT } W M f\)
    + \(\text{functor laws}\)
    - \(\text{return & bind} + \text{naturality}\)

+ \(\text{functor laws} \text{ (by induction)}\)

Object action: \(\text{WriterT } W \in \text{MonadTrans}\)

- \(\text{Functorial action WriterT } W g\)
  - \(\text{Respects return & bind}\)
  + \(\text{functor laws}\)
  - \(\text{lift} : M \rightarrow \text{WriterT } W M + \text{naturality}\)
    - \(\text{Respects return & bind}\)

- \(\text{Functorial action: WriterT } h : \text{WriterT } V \rightarrow \text{WriterT } W\)
  - \(\text{WriterT } h M A\)
    - \(\text{Respects return, bind & lift}\)
  - \(\text{naturality w.r.t. } A\)
  - \(\text{naturality w.r.t. } M\)

+ \(\text{functor laws}\)
Variance and modalities

WriterT \(W M A\) : Monad is **covariant** w.r.t.
- \(W\) : Monoid
- \(M\) : Monad
- \(A\) : Type

ReaderT \(R M A\) is **contravariant** w.r.t.
- \(R\) : Type

return : \(A \rightarrow\) WriterT \(W M A\) is **natural** w.r.t.
- \(W\) : Monoid
- \(M\) : Monad
- \(A\) : Type

Ignoring variance

- HoTT: only consider **isomorphisms**
  😞 Not everything is an isomorphism.
- Param’ty: relations, not morphisms
  😞 Don’t know how to compute \(\text{fmap}\).

Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
### Variance and modalities

**Writer** $\text{Writer} W M A : \text{Monad}$ is **covariant** w.r.t.
- $W : \text{Monoid}$
- $M : \text{Monad}$
- $A : \text{Type}$

**Reader** $\text{Reader} R M A$ is **contravariant** w.r.t.
- $R : \text{Type}$

\[ \text{return} : A \to \text{Writer} W M A \text{ is } \text{natural} \text{ w.r.t.} \]
- $W : \text{Monoid}$
- $M : \text{Monad}$
- $A : \text{Type}$

### Ignoring variance

- HoTT: only consider **isomorphisms**
  - 😞 Not everything is an isomorphism.
- Param’ty: relations, not morphisms
  - 😞 Don’t know how to compute fmap.

### Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
### Variance and modalities

**WriterT W M A** : Monad is **covariant** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

**ReaderT R M A** is **contravariant** w.r.t.
- \( R \) : Type

**return** : \( A \rightarrow WriterT W M A \) is **natural** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

### Ignoring variance

- HoTT: only consider **isomorphisms**
  - 😞 Not everything is an isomorphism.
- Param’ty: relations, not morphisms
  - 😞 Don’t know how to compute fmap.

### Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

**Hence:**
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
**Variance and modalities**

WriterT \( W M A \) : Monad is **covariant** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

ReaderT \( R M A \) is **contravariant** w.r.t.
- \( R \) : Type

return : \( A \rightarrow \) WriterT \( W M A \) is **natural** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

**Ignoring variance**

- HoTT: only consider **isomorphisms**
  😞 Not everything is an isomorphism.
- Param’ty: **relations**, not morphisms
  😞 Don’t know how to compute \( \text{fmap} \).

**Naturality TT**

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
Variance and modalities

WriterT $W M A$ : Monad is **covariant** w.r.t.
- $W$ : Monoid
- $M$ : Monad
- $A$ : Type

ReaderT $R M A$ is **contravariant** w.r.t.
- $R$ : Type

return : $A \rightarrow$ WriterT $W M A$ is **natural** w.r.t.
- $W$ : Monoid
- $M$ : Monad
- $A$ : Type

Ignoring variance

- HoTT: only consider **isomorphisms**
  - 😞 Not everything is an isomorphism.
- Param’ty: **relations**, not morphisms
  - 😞 Don’t know how to compute fmap.

Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:

- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
Variance and modalities

WriterT \( W M A \) : Monad is **covariant** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

ReaderT \( R M A \) is **contravariant** w.r.t.
- \( R \) : Type

\text{return} : A \rightarrow \text{WriterT} \( W M A \) is **natural** w.r.t.
- \( W \) : Monoid
- \( M \) : Monad
- \( A \) : Type

---

Ignoring variance

- HoTT: only consider **isomorphisms**
  😞 Not everything is an isomorphism.
- Param’t’y: **relations**, not morphisms
  😞 Don’t know how to compute \( \text{fmap} \).

---

Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’t’y when necessary
Variance and modalities

**WriterT W M A** : Monad is **covariant** w.r.t.
- W : Monoid
- M : Monad
- A : Type

**ReaderT R M A** is **contravariant** w.r.t.
- R : Type

**return : A ↦ WriterT W M A** is **natural** w.r.t.
- W : Monoid
- M : Monad
- A : Type

Ignoring variance

- **HoTT**: only consider **isomorphisms**
  - 😞 Not everything is an isomorphism.
- Param’ty: **relations**, not morphisms
  - 😞 Don’t know how to compute fmap.

Naturality TT

- Preserve isomorphisms
- Preserve relations
- Keep track of action on morphisms

Hence:
- Use functoriality/naturality when possible
- Use HoTT when applicable
- Use param’ty when necessary
Pretypes: A Note on Fibrancy
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

Fibrant types have operations for these:

We ignore fibrancy for now:

- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

⇒ It’s a pretype system

| Directed | functorial | Transport along morphisms |
| Segal    | Composition of morphisms |
| Rezk     | Isomorphism-path univalence |
| HoTT     | Comp. of & transp. along paths |
| Param’ty | discrete   | Homog. bridges express equality |
|          | …          | …                         |

Andreas Nuyts

Higher Pro-arrows: Towards a Model for Naturality Pretype Theory
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

**Fibrant** types have **operations** for these:

We **ignore** fibrancy for now:

- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

⇒ It’s a **pretype** system

<table>
<thead>
<tr>
<th>Directed</th>
<th>HoTT</th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Kan</td>
</tr>
<tr>
<td>Transport along morphisms</td>
<td>Comp. of &amp; transp. along paths</td>
</tr>
</tbody>
</table>

| Segal | Par’m’ty |
| Composition of morphisms | discrete |
| Isomorphism-path univalence | Homog. bridges express equality |

... ...
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

**Fibrant** types have operations for these:

We ignore fibrancy for now:

- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

⇒ It’s a pretype system

<table>
<thead>
<tr>
<th>Directed</th>
<th>Segal</th>
<th>Rezk</th>
<th>HoTT</th>
<th>Kan</th>
<th>Param’ty</th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Transport</td>
<td>Isomorphism-path</td>
<td>Comp. of &amp; transp.</td>
<td>Comp. of &amp; transp.</td>
<td>discrete</td>
</tr>
<tr>
<td></td>
<td>along morphisms</td>
<td>univalence</td>
<td>along paths</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segal</td>
<td>Composition</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>of morphisms</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rezk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HoTT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Param’ty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>discrete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Andreas Nuyts
Higher Pro-arrows: Towards a Model for Naturality Pretype Theory 10/19
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

Fibrant types have operations for these:

We ignore fibrancy for now:

- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

⇒ It’s a pretype system

<table>
<thead>
<tr>
<th>Directed</th>
<th></th>
<th>HoTT</th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Transport along morphisms</td>
<td>Kan</td>
</tr>
<tr>
<td>Segal</td>
<td>Composition of morphisms</td>
<td>Comp. of &amp; transp. along paths</td>
</tr>
<tr>
<td>Rezk</td>
<td>Isomorphism-path univalence</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Param’ty</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>discrete</td>
<td>Homog. bridges express equality</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

Fibrant types have operations for these:

We ignore fibrancy for now:
- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]
⇒ It’s a pretype system

<table>
<thead>
<tr>
<th>Directed</th>
<th>HoTT</th>
<th>Param’ty</th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Kan</td>
<td>discrete</td>
</tr>
<tr>
<td>Transport along morphisms</td>
<td>Comp. of &amp; transp. along paths</td>
<td>Homog. bridges express equality</td>
</tr>
<tr>
<td>Segal</td>
<td>Rezk</td>
<td></td>
</tr>
<tr>
<td>Composition of morphisms</td>
<td>Isomorphism-path univalence</td>
<td></td>
</tr>
</tbody>
</table>

Andreas Nuyts
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

**Fibrant** types have **operations** for these:

We **ignore** fibrancy for now:

- Functoriality & Segal fibrancy are brittle
  \(\Rightarrow\) need to consider pretypes anyway

- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

\(\Rightarrow\) It’s a **pretype** system

<table>
<thead>
<tr>
<th>Directed</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Transport along morphisms</td>
</tr>
<tr>
<td>Segal</td>
<td>Composition of morphisms</td>
</tr>
<tr>
<td>Rezk</td>
<td>Isomorphism-path univalence</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HoTT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Kan</td>
<td>Comp. of &amp; transp. along paths</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Param’ty</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>discrete</td>
<td>Homog. bridges express equality</td>
</tr>
</tbody>
</table>

\[\ldots\]
A note on fibrancy

A presheaf model of DTT can account for the existence of paths/morphisms/bridges/…

**Fibrant** types have **operations** for these:

We **ignore** fibrancy for now:

- Functoriality & Segal fibrancy are brittle ⇒ need to consider pretypes anyway
- There are promising techniques for defining fibrancy internally:
  - Contextual fibrancy [BT21, Nuy20]
  - Amazing right adjoint [LOPS18] & Transpension [ND21]
  - Internal fibrant replacement monad [Nuy20]

⇒ It’s a **pretype** system

<table>
<thead>
<tr>
<th>Directed</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>functorial</td>
<td>Transport along morphisms</td>
</tr>
<tr>
<td>Segal</td>
<td>Composition of morphisms</td>
</tr>
<tr>
<td>Rezk</td>
<td>Isomorphism-path univalence</td>
</tr>
<tr>
<td><strong>HoTT</strong></td>
<td></td>
</tr>
<tr>
<td>Kan</td>
<td>Comp. of &amp; transp. along paths</td>
</tr>
</tbody>
</table>

| **Param’ty** |             |
| discrete    | Homog. bridges express equality |

... ...
Model-first Approach

The type system emerges from the model:
- A diagram of CwFs and adjunctions models an instance of MTT [GKNB20].
- An endofunctor on $\mathcal{W}$ models a substructural shape (e.g. $\mathbb{I}$) in $\operatorname{Psh}(\mathcal{W})$ giving rise to modalities $\exists(i : \mathbb{I}) \dashv P[i : \mathbb{I}] \dashv \forall(i : \mathbb{I}) \dashv \check{\exists}(i : \mathbb{I})$. This is the basis of the modal transpension type system (MTraS) [ND21].
Model-first Approach

The type system emerges from the model:

- A diagram of CwFs and adjunctions models an instance of MTT [GKNB20].
- An endofunctor on $W$ models a substructural shape (e.g. $\mathbb{I}$) in $\text{Psh}(W)$ giving rise to modalities
  
  \[
  \exists (i : \mathbb{I}) \dashv \exists [i : \mathbb{I}] \dashv \forall (i : \mathbb{I}) \dashv \forall [i : \mathbb{I}].
  \]

  This is the basis of the modal transpension type system (MTraS) [ND21].
The Model

- Pro-arrow equipments
- Degrees of Relatedness (RelDTT) [ND18]
- Tamsamani & Simpson’s model of higher category (graph) theory [CL04, Tam99, Sim11]

- Higher dimension
- Directify
- Heterogenize

Naturality (Pre)type Theory
A category $\mathcal{C}$ can be defined as a simplicial set $\mathcal{C} \in \text{Psh}(\Delta)$ satisfying the Segal condition.

A double category $\mathcal{C}$ has:
- objects
- horiz. arrows / (1)-arrows
- vertical arrows / (2)-arrows
- squares

and can be defined as a bisimplicial set $\mathcal{C} \in \text{Psh}(\Delta \times \Delta)$ satisfying the Segal condition in each dimension.

An $n$-fold category $\mathcal{C}$ is an $n$-fold simplicial set $\mathcal{C} \in \text{Psh}(\Delta^n)$ satisfying the Segal condition in each dimension.
**n-Fold Categories**

**Category**

A category $\mathcal{C}$ can be defined as a simplicial set $\mathcal{C} \in \text{Psh}(\Delta)$ satisfying the Segal condition.

**Double category**

A double category $\mathcal{C}$ has:
- objects
- horiz. arrows / (1)-arrows
- vertical arrows / (2)-arrows
- squares

and can be defined as a bisimplicial set $\mathcal{C} \in \text{Psh}(\Delta \times \Delta)$ satisfying the Segal condition in each dimension.

**n-Fold category**

An $n$-fold category $\mathcal{C}$ is an $n$-fold simplicial set $\mathcal{C} \in \text{Psh}(\Delta^n)$ satisfying the Segal condition in each dimension.

---

Pretypes!
**$n$-Fold Categories**

<table>
<thead>
<tr>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>A <strong>category</strong> $\mathcal{C}$ can be defined as a <strong>simplicial set</strong> $\mathcal{C} \in \text{Psh}(\Delta)$ satisfying the <strong>Segal condition</strong>.</td>
</tr>
</tbody>
</table>

**Double category**

<table>
<thead>
<tr>
<th>A <strong>double category</strong> $\mathcal{C}$ has:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• objects</td>
</tr>
<tr>
<td>• horiz. arrows / (1)-arrows</td>
</tr>
<tr>
<td>• vertical arrows / (2)-arrows</td>
</tr>
<tr>
<td>• squares</td>
</tr>
</tbody>
</table>

and can be defined as a **bisimplicial set** $\mathcal{C} \in \text{Psh}(\Delta \times \Delta)$ satisfying the **Segal condition** in each dimension.

---

**$n$-Fold Category**

An **$n$-fold category** $\mathcal{C}$ is an $n$-fold simplicial set $\mathcal{C} \in \text{Psh}(\Delta^n)$ satisfying the **Segal condition** in each dimension.
**Category**

A category $\mathcal{C}$ can be defined as a simplicial set $\mathcal{C} \in \text{Psh}(\Delta)$ satisfying the Segal condition.

**Double category**

A double category $\mathcal{C}$ has:

- objects
- horiz. arrows / (1)-arrows
- vertical arrows / (2)-arrows
- squares

and can be defined as a bisimplicial set $\mathcal{C} \in \text{Psh}(\Delta \times \Delta)$ satisfying the Segal condition in each dimension.
**Category**

A category $\mathcal{C}$ can be defined as a simplicial set $\mathcal{C} \in Psh(\Delta)$ satisfying the **Segal condition**.

---

**Double category**

A double category $\mathcal{C}$ has:
- objects
- horiz. arrows / (1)-arrows
- vertical arrows / (2)-arrows
- squares

and can be defined as a bisimplicial set $\mathcal{C} \in Psh(\Delta \times \Delta)$ satisfying the **Segal condition** in each dimension.

---

**$n$-Fold category**

An $n$-fold category $\mathcal{C}$ is an $n$-fold simplicial set $\mathcal{C} \in Psh(\Delta^n)$ satisfying the **Segal condition** in each dimension.

---

**Pretypes!**
The Twisted Prism Functor

$\Delta$ is a skeleton of $\text{FinLinOrd}$, hence $\Delta \simeq \text{FinLinOrd}$.

### Twisted Prism Functor [PK19]

$$\square \times \Box : \text{FinLinOrd} \rightarrow \text{FinLinOrd} : \quad W \mapsto W^{\text{op}} \sqcup < W$$

### Twisted Cube Category $\Box$ [PK19]

(Roughly) the subcategory of $\text{FinLinOrd}$ (or $\Delta$) generated by $\top$ and $\square \times \Box$.

### $n$-Fold category

An $n$-fold category $\mathcal{C}$ is an $n$-fold twisted cubical set $\mathcal{C} \in \text{Psh}(\Box^n)$ satisfying the Segal condition in each dimension.

An MTraS-shape $\Box$ modelled by $\square \times \Box$, reconciles the view of Hom as a contra-/covariant bifunctor with a view as a directed path type.
The Twisted Prism Functor

Δ is a skeleton of FinLinOrd, hence Δ ≃ FinLinOrd.

Twisted Prism Functor [PK19]

□ × □ : FinLinOrd → FinLinOrd :
W ↦ W^{op} ⊔ < W

An MTraS-shape I modelled by □ × □, reconciles the view of Hom as a contra-/covariant bifunctor with a view as a directed path type.

II as an MTraS-shape is better behaved on □:

Twisted Cube Category □ [PK19]
(Roughly) the subcategory of FinLinOrd (or Δ) generated by □ and □ × □.

n-Fold category

An n-fold category C is an n-fold twisted cubical set C ∈ Psh(□ⁿ) satisfying the Segal condition in each dimension.
The Twisted Prism Functor

The Twisted Prism Functor

\[ \Delta \text{ is a skeleton of FinLinOrd, hence } \Delta \simeq \text{FinLinOrd.} \]

Twisted Prism Functor [PK19]

Twisted Cube Category [PK19]

(Roughly) the subcategory of FinLinOrd (or \( \Delta \)) generated by \( \top \) and \( \square \times \square \).

An MTraS-shape \( \square \times \square \) modelled by \( \square \times \square \), reconciles the view of Hom as a contra-/covariant bifunctor with a view as a directed path type.

\[ \begin{array}{ccc}
\square \times \square : \text{FinLinOrd} & \rightarrow & \text{FinLinOrd} \\
W & \mapsto & W^{\text{op}} \cup < \ W \\
\end{array} \]

\[ a \rightarrow b \quad \mapsto \quad \begin{array}{ccc}
\square \times \square : \text{FinLinOrd} & \rightarrow & \text{FinLinOrd} \\
W & \mapsto & W^{\text{op}} \cup < \ W \\
\end{array} \]

\[ \begin{array}{ccc}
\square \times \square : \text{FinLinOrd} & \rightarrow & \text{FinLinOrd} \\
W & \mapsto & W^{\text{op}} \cup < \ W \\
\end{array} \]

\[ \begin{array}{ccc}
\square \times \square : \text{FinLinOrd} & \rightarrow & \text{FinLinOrd} \\
W & \mapsto & W^{\text{op}} \cup < \ W \\
\end{array} \]

An n-fold category

An n-fold category \( \mathcal{C} \) is an n-fold twisted cubical set \( \mathcal{C} \in \text{Psh}(\square^n) \) satisfying the Segal condition in each dimension.
The Twisted Prism Functor

\( \Delta \) is a skeleton of \( \text{FinLinOrd} \), hence \( \Delta \simeq \text{FinLinOrd} \).

Twisted Prism Functor [PK19]

\[ \square \times \square : \text{FinLinOrd} \to \text{FinLinOrd} : \]
\[ W \leftrightarrow W^{\text{op}} \cup < W \]

\( \square \times \square \) as an MTraS-shape is better behaved on \( \boxdot \):

Twisted Cube Category \( \boxdot \) [PK19]

(Roughly) the subcategory of \( \text{FinLinOrd} \) (or \( \Delta \)) generated by \( \top \) and \( \square \times \square \).

\( n \)-Fold category

An \( n \)-fold category \( C \) is an \( n \)-fold twisted cubical set \( C \in \text{Psh}(\boxdot^n) \) satisfying the Segal condition in each dimension.

An MTraS-shape \( \square \) modelled by \( \square \times \square \), reconciles the view of \( \text{Hom} \) as a contra-/covariant bifunctor with a view as a directed path type.
The Twisted Prism Functor

$\Delta$ is a skeleton of $\text{FinLinOrd}$, hence $\Delta \simeq \text{FinLinOrd}$.

**Twisted Prism Functor [PK19]**

$\square \times \mathbb{I} : \text{FinLinOrd} \to \text{FinLinOrd} : W \mapsto W^{op} \uplus \subset W$

$I$ as an MTraS-shape is better behaved on $\boxdot$:

**Twisted Cube Category $\boxdot$ [PK19]**

(Roughly) the subcategory of $\text{FinLinOrd}$ (or $\Delta$) generated by $\top$ and $\square \times \mathbb{I}$.

**$n$-Fold category**

An $n$-fold category $\mathcal{C}$ is an $n$-fold twisted cubical set $\mathcal{C} \in \text{Psh}(\boxdot^n)$ satisfying the Segal condition in each dimension.

An MTraS-shape $\mathbb{I}$ modelled by $\square \times \mathbb{I}$, reconciles the view of $\text{Hom}$ as a contra-/covariant bifunctor with a view as a directed path type.
An equipment \( \mathcal{C} \) is a double category with:
- objects
- arrows (\( \rightarrow \))
- pro-arrows (\( \nrightarrow \))
- squares

such that every arrow \( \varphi: x \rightarrow y \) has graph pro-arrows \( \varphi^\downarrow : x \nrightarrow y \) and \( \varphi^\uparrow : y \nrightarrow x \) such that (\ldots).

Example: Set
Set is an equipment with:
- sets
- functions
- relations
  - identity relation: equality
  - \( (R; S)(x, z) = \exists y. R(x, y) \land S(y, z) \)
- proofs that \( R(a, b) \Rightarrow S(f a, g b) \)
An **equipment** $\mathcal{C}$ is a **double category** with
- objects
- arrows ($\rightarrow$)
- pro-arrows ($\nrightarrow$)
- squares
such that every arrow $\varphi: x \rightarrow y$ has **graph** pro-arrows $\varphi^\dagger: x \nrightarrow y$ and $\varphi^\ddagger: y \nrightarrow x$ such that (...).

**Example: Set**
Set is an equipment with:
- sets
- functions
- relations
  - identity relation: equality
  - $(R; S)(x, z) = \exists y. R(x, y) \land S(y, z)$
  - proofs that $R(a, b) \Rightarrow S(f a, g b)$
An **equipment** $\mathcal{C}$ is a **double category** with
- objects
- arrows ($\rightarrow$)
- pro-arrows ($\nrightarrow$)
- squares
such that every arrow $\varphi : x \rightarrow y$ has **graph** pro-arrows $\varphi^\dagger : x \nrightarrow y$ and $\varphi^\ddagger : y \nrightarrow x$ such that ($\ldots$).

**Example: Set**

Set is an equipment with:
- sets
- functions
- relations
  - identity relation: equality
  - $(R; S)(x, z) =$
    \[ \exists y. R(x, y) \land S(y, z) \]
- proofs that $R(a, b) \Rightarrow S(f a, g b)$
An equipment $C$ is a double category with
- objects
- arrows ($\rightarrow$)
- pro-arrows ($\nrightarrow$)
- squares

such that every arrow $\varphi : x \rightarrow y$ has graph pro-arrows $\varphi^\ddagger : x \nrightarrow y$ and $\varphi^\dagger : y \nrightarrow x$ such that (...).

**Example: Cat**

Cat is an equipment with:
- categories
- functors
- profunctors
  - identity profunctor: $\text{Hom}$
  - $(\mathcal{P}; \mathcal{Q})(x, z) = \text{coend} \exists y. \mathcal{P}(x, y) \times \mathcal{Q}(y, z)$
  - $\forall a, b. \mathcal{P}(a, b) \Rightarrow \mathcal{Q}(F a, G b)$
Higher equipments

Set is ...

- 😞 A large set
- 😃 A category
- 😃 An equipment

Cat is ...

- 😞 A category
- 😃 A 2-category
- 😃 An equipment

Eqmnt is ...

- 😞 An equipment
- 😃 A 2-equipment

Eqmnt has:

- Objects Equipments
- Arrows Equipment functors
- Pro-arrows Equipment profunctors:
  - Contain arrows and pro-arrows
- Pro-pro-arrows Equipment pro-profunctors:
  - Contain pro-arrows

Squares ...

Cubes ...

Higher Equipment

An $n$-equipment is an $n$-fold category (…)

⇒ $\mathcal{C} \in \text{Psh}(\boxtimes^n_{\uparrow,\downarrow})$
Set is …
- A large set
- A category
- An equipment

Cat is …
- A category
- A 2-category
- An equipment

Eqmnt is …
- An equipment
- A 2-equipment

Eqmnt has:
- Objects
- Equipments
- Arrows
- Equipment functors
- Pro-arrows
- Equipment profunctors: Contain arrows and pro-arrows
- Pro-pro-arrows
- Equipment pro-profunctors: Contain pro-arrows
- Squares …
- Cubes …

Higher Equipment
An $n$-equipment is an $n$-fold category (…)

$\mathcal{C} \in \text{Psh}(\{\times_n\}^\uparrow, \downarrow)$
Higher equipments

Set is . . .
- A large set
- A category
- An equipment

Cat is . . .
- A category
- A 2-category
- An equipment

Eqmnt is . . .
- An equipment
- A 2-equipment

Eqmnt has:
- Objects Equipments
- Arrows Equipment functors
- Pro-arrows Equipment profunctors:
  - Contain arrows and pro-arrows
- Pro-pro-arrows Equipment pro-profunctors:
  - Contain pro-arrows
- Squares . . .
- Cubes . . .

Higher Equipment

An $n$-equipment is an $n$-fold category ( . . . )

$\Rightarrow C \in \text{Psh}(\times^n_{\uparrow \downarrow})$
Higher equipments

Set is ...
- A large set
- A category
- An equipment

Cat is ...
- A category
- A 2-category
- An equipment

Eqmnt is ...
- An equipment
- A 2-equipment

Eqmnt has:
- **Objects**  Equipments
- **Arrows**  Equipment functors
- **Pro-arrows**  Equipment profunctors:
  - Contain arrows and pro-arrows
- **Pro-pro-arrows**  Equipment **pro-profunctors**:
  - Contain pro-arrows

Squares ...
Cubes ...

Higher Equipment

An *n*-equipment is an *n*-fold category (…)

⇒ \( \mathcal{C} \in \text{Psh}(\times^n) \)
Set is . . .
  😞 A large set
  😎 A category
  😎 An equipment

Cat is . . .
  😞 A category
  😎 A 2-category
  😎 An equipment

Eqmnt is . . .
  😞 An equipment
  😎 A 2-equipment

Eqmnt has:
  Objects 
  Arrows Equipment functors
  Pro-arrows Equipment profunctors:
    Contain arrows and pro-arrows
  Pro-pro-arrows Equipment profunctors:
    Contain pro-arrows
  Squares . . .
  Cubes . . .

Higher Equipment
An $n$-equipment is an $n$-fold category (. . .)

$\mathcal{C} \in \text{Psh}(\mathcal{X}^n_\uparrow, \downarrow)$
Directifying “Degrees of Relatedness” [ND18]

Depth $n$ types

- $i$-edge relations $\bowtie_i$
  - $R : A \bowtie_{i+1}^U B$
    - is a container for $r : a \bowtie^R_i b$
    - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
  - $a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
  - Modalities change indices:

- $i$-jet (pro-$i^{-1}$-arrow) relations $\bowtie_i$
  - $J : A \bowtie_{i+1}^U B$
    - is a container for $j : a \bowtie^J_i b$
    - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
  - $(\downarrow, \uparrow) : a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
  - Modalities change indices & orientation:
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**

- $i$-edge relations $\rightsquigarrow_i$
- $R : A \rightsquigarrow_{i+1}^U B$
  is a container for $r : a \rightsquigarrow_i^R b$
  $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices:

**$n$-equipments**

- $i$-jet (pro$i^{-1}$-arrow) relations $\rightsquigarrow_i$
- $J : A \rightsquigarrow_{i+1}^U B$
  is a container for $j : a \rightsquigarrow_j^J b$
  $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $(\ddagger, \dagger) : a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices & orientation:
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**
- $i$-edge relations $\circ_i$
- \( R : A \circ_{i+1} B \)
  - is a container for \( r : a \circ_i b \)
  - \( U = \langle \text{disc} \mid U^\text{HS} \rangle \)
- \( a \circ_i b \Rightarrow a \circ_{i+1} b \)
- Modalities change indices:

**n-equipments**
- $i$-jet (pro\(i^{-1}\)-arrow) relations $\circ_i$
- \( J : A \circ_{i+1} B \)
  - is a container for \( j : a \circ_{i'} b \)
  - \( U = \langle \text{disc} \mid U^\text{HS} \rangle \)
- \((\dagger, \ddagger) : a \circ_i b \Rightarrow a \circ_{i+1} b \)
- Modalities change indices & orientation:
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**

- $i$-edge relations $\bowtie_i$
- $\mathcal{R} : A \bowtie^{U}_{i+1} B$
  is a container for $r : a \bowtie^\mathcal{R}_i b$
  $\mathcal{U} = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
- Modalities change indices:

```
\begin{align*}
  &a \bowtie_0 b & fa \bowtie_0 fb \\
  \downarrow & & \downarrow \\
  &a \bowtie_1 b & fa \bowtie_1 fb \\
  \downarrow & & \downarrow \\
  &a \bowtie_2 b & fa \bowtie_2 fb \\
\end{align*}
```

```
\begin{align*}
  &a \bowtie_0 b & a \bowtie_0 b \\
  \downarrow & & \downarrow \\
  &a \bowtie_1 b & a \bowtie_1 b \\
  \downarrow & & \downarrow \\
  &a \bowtie_2 b & a \bowtie_2 b \\
\end{align*}
```

```
\begin{align*}
  &par & con \\
  \text{par} & & \text{con}
\end{align*}
```

**$n$-equipments**

- $i$-jet ($\text{pro}^{i+1}$-arrow) relations $\bowtie_i$
- $\mathcal{J} : A \bowtie^{U}_{i+1} B$
  is a container for $j : a \bowtie^\mathcal{J}_i b$
  $\mathcal{U} = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $(\dagger, \ddagger) : a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
- Modalities change indices & orientation:

```
\begin{align*}
  &a \bowtie_0 b & fa \bowtie_0 fb \\
  \downarrow & & \downarrow \\
  &a \bowtie_1 b & fa \bowtie_1 fb \\
  \downarrow & & \downarrow \\
  &a \bowtie_2 b & fa \bowtie_2 fb \\
  \downarrow & & \downarrow \\
\end{align*}
```

```
\begin{align*}
  &a \bowtie_0 b & a \bowtie_0 b \\
  \downarrow & & \downarrow \\
  &a \bowtie_1 b & a \bowtie_1 b \\
  \downarrow & & \downarrow \\
  &a \bowtie_2 b & a \bowtie_2 b \\
\end{align*}
```

```
\begin{align*}
  &par & con \\
  \text{par} & & \text{con}
\end{align*}
```

Andreas Nuyts
Higher Pro-arrows: Towards a Model for Naturality Pretype Theory 17/19
Directifying “Degrees of Relatedness” [ND18]

**Depth \(n\) types**
- \(i\)-edge relations \(\bowtie_i\)
- \(R : A \bowtie_{i+1} B\)
  is a container for \(r : a \bowtie_i^R b\)
  \(U = \langle \text{disc} \mid U^{\text{HS}} \rangle\)
- \(a \bowtie_i b \Rightarrow a \bowtie_{i+1} b\)
- Modalities change indices:

**\(n\)-equipments**
- \(i\)-jet (\(\text{pro}^{i-1}\)-arrow) relations \(\bowtie_i\)
- \(J : A \bowtie_{i+1}^U B\)
  is a container for \(j : a \bowtie_i^J b\)
  \(U = \langle \text{disc} \mid U^{\text{HS}} \rangle\)
- \((\check{\vdash}, \check{\dashv}) : a \bowtie_i b \Rightarrow a \bowtie_{i+1} b\)
- Modalities change indices & orientation:
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**

- $i$-edge relations $\rightsquigarrow_i$
- $R : A \rightsquigarrow_{i+1}^U B$
  - is a container for $r : a \rightsquigarrow_i^R b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices:

$$
\begin{align*}
& a \rightsquigarrow_0 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_1 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_2 b
\end{align*}
\quad
\begin{align*}
& a \rightsquigarrow_0 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_1 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_2 b
\end{align*}
$$

**$n$-equipments**

- $i$-jet (pro$^{i-1}$-arrow) relations $\rightsquigarrow_i$
- $J : A \rightsquigarrow_{i+1}^U B$
  - is a container for $j : a \rightsquigarrow_i^J b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $(\vdash, \dashv) : a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices & orientation:

$$
\begin{align*}
& a \rightsquigarrow_0 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_1 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_2 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_2 b
\end{align*}
\quad
\begin{align*}
& a \rightsquigarrow_0 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_1 b \\
\downarrow & \hspace{1cm} \downarrow \\
& a \rightsquigarrow_2 b
\end{align*}
$$
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**
- $i$-edge relations $\rightsquigarrow_i$
- $R : A \rightsquigarrow_{i+1}^U B$
  - is a container for $r : a \rightsquigarrow_i^R b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices:

**$n$-equipments**
- $i$-jet (pro$^{i-1}$-arrow) relations $\rightsquigarrow_i$
- $J : A \rightsquigarrow_{i+1}^U B$
  - is a container for $j : a \rightsquigarrow_i^J b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $(\prescript{\dagger}{\ddagger}) : a \rightsquigarrow_i b \Rightarrow a \rightsquigarrow_{i+1} b$
- Modalities change indices & orientation:

```
\begin{tikzcd}
a \rightsquigarrow_0 b \\
a \rightsquigarrow_1 b \\
a \rightsquigarrow_2 b \\
\end{tikzcd}
```

```
\begin{tikzcd}
a \rightsquigarrow_0 b \\
a \rightsquigarrow_1 b \\
a \rightsquigarrow_2 b \\
\end{tikzcd}
```

```
\begin{tikzcd}
a \rightsquigarrow_0 b \\
a \rightsquigarrow_1 b \\
a \rightsquigarrow_2 b \\
\end{tikzcd}
```

```
\begin{tikzcd}
a \rightsquigarrow_0 b \\
a \rightsquigarrow_1 b \\
a \rightsquigarrow_2 b \\
\end{tikzcd}
```
Directifying “Degrees of Relatedness” [ND18]

**Depth $n$ types**
- $i$-edge relations $\bowtie_i$
- $R : A \bowtie_{i+1} B$
  - is a container for $r : a \bowtie_i^R b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
- Modalities change indices:

**$n$-equipments**
- $i$-jet (pro$^{i-1}$-arrow) relations $\bowtie_i$
- $J : A \bowtie_{i+1} B$
  - is a container for $j : a \bowtie_i^J b$
  - $U = \langle \text{disc} \mid U^{\text{HS}} \rangle$
- $(\x, \dagger) : a \bowtie_i b \Rightarrow a \bowtie_{i+1} b$
- Modalities change indices & orientation:
2-category (Tamsamani & Simpson)
An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only
- (1)-arrows \( \sim \) 1-arrows
- (1,2)-squares \( \sim \) 2-arrows
can be non-trivial.

\[
\begin{array}{ccc}
a & \xrightarrow{id^{(2)}} & a \\
\varphi \downarrow & \Rightarrow & \chi \\
c & \xrightarrow{id^{(2)}} & c
\end{array}
\]

Recall the equipment Cat:

\[
\begin{array}{ccc}
\mathcal{A} & \xrightarrow{P} & \mathcal{B} \\
F \downarrow & \Rightarrow & G \\
\mathcal{C} & \xrightarrow{Q} & \mathcal{D}
\end{array}
\]

\[\forall a, b. \hom(F a, G b) \Rightarrow \mathcal{Q}(F a, G b)\]

\[\forall a. \hom(F a, G a)\]
Tamsamani & Simpson’s model of higher category theory

2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only
- (1)-arrows \( \sim \) 1-arrows
- (1,2)-squares \( \sim \) 2-arrows

can be non-trivial.

\[ a \xrightarrow{\text{id}^{(2)}} a \]
\[ \varphi \quad \Rightarrow \quad \chi \]
\[ c \xrightarrow{\text{id}^{(2)}} c \]

\[ \begin{array}{ccc}
a & \xrightarrow{\varphi} & a \\
| & \Downarrow & | \\
c & \xrightarrow{\chi} & c \\
\end{array} \]

\[ \begin{array}{ccc}
a & \xrightarrow{\varphi} & a \\
| & \Downarrow & | \\
c & \xrightarrow{\chi} & c \\
\end{array} \]

Recall the equipment Cat:

\[ \begin{array}{ccc}
\mathcal{A} & \xrightarrow{P} & \mathcal{B} \\
\downarrow F & & \downarrow G \\
\mathcal{C} & \xrightarrow{Q} & \mathcal{D} \\
\end{array} \]

\[ \forall a, b. \text{Hom}(F a, G b) \Rightarrow Q(F a, G b) \]

\[ \forall a. \text{Hom}(F a, G a) \]
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only
- (1)-arrows $\sim$ 1-arrows
- (1,2)-squares $\sim$ 2-arrows
can be non-trivial.

$n$-category (Tamsamani & Simpson)

An $n$-category is an **$n$-fold** category where only
- (1)-arrows, $\sim$ 1-arrows
- (1,2)-squares, $\sim$ 2-arrows
- (1,2,3)-cubes, $\sim$ 3-arrows …
can be non-trivial.

Recall the equipment Cat:
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only

- (1)-arrows \(\sim\) 1-arrows
- (1,2)-squares \(\sim\) 2-arrows

...can be non-trivial.

\[ a \xrightarrow{id^{(2)}} a \]
\[ \varphi \Rightarrow \chi \]
\[ c \xrightarrow{id^{(2)}} c \]
\[ \varphi \Rightarrow \chi \]
\[ c \]

Recall the equipment Cat:

\[ \mathcal{A} \xrightarrow{P} \mathcal{B} \]
\[ F \downarrow \quad \downarrow G \]
\[ C \xrightarrow{Q} D \]

\[ \forall a, b. \mathcal{P}(a, b) \Rightarrow \mathcal{Q}(F a, G b) \]

\[ \forall a. \text{Hom}(F a, G a) \]
Tamsamani & Simpson’s model of higher category theory

2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold) category** whose (2)-arrows are all trivial (id), so only
- (1)-arrows $\sim$ 1-arrows
- (1,2)-squares $\sim$ 2-arrows

can be non-trivial.

\[
\begin{array}{c}
\begin{array}{c}
a \\
\downarrow \varphi
\end{array}
\end{array}
\Rightarrow
\begin{array}{c}
\begin{array}{c}
\chi
\end{array}
\end{array}
\]

Recall the equipment Cat:

\[
\begin{array}{ccc}
\mathcal{A} & \xrightarrow{\mathcal{P}} & \mathcal{B} \\
F & \downarrow & G \\
\mathcal{C} & \xrightarrow{\mathcal{Q}} & \mathcal{D}
\end{array}
\]

\[
\forall a, b. \mathcal{P}(a, b) \Rightarrow \mathcal{Q}(F a, G b)
\]

\[
\cong \forall a. \text{Hom}(F a, G a)
\]

---

*n-category (Tamsamani & Simpson)*

An *n*-category is an **n-fold category** where only
- (1)-arrows, $\sim$ 1-arrows
- (1,2)-squares, $\sim$ 2-arrows
- (1,2,3)-cubes, $\sim$ 3-arrows . . .

can be non-trivial.
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only

- (1)-arrows $\sim$ 1-arrows
- (1,2)-squares $\sim$ 2-arrows

can be non-trivial.

$n$-category (Tamsamani & Simpson)

An $n$-category is an **$n$-fold** category where only

- (1)-arrows, $\sim$ 1-arrows
- (1,2)-squares, $\sim$ 2-arrows
- (1,2,3)-cubes, $\sim$ 3-arrows ...

can be non-trivial.

Recall the equipment $\text{Cat}$:
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only

- (1)-arrows $\sim$ 1-arrows
- (1,2)-squares $\sim$ 2-arrows

can be non-trivial.

$n$-category (Tamsamani & Simpson)

An $n$-category is an **$n$-fold** category where only

- (1)-arrows, $\sim$ 1-arrows
- (1,2)-squares, $\sim$ 2-arrows
- (1,2,3)-cubes, $\sim$ 3-arrows ...

can be non-trivial.

Recall the equipment $\text{Cat}$:

\[
\begin{array}{ccc}
A & \xrightarrow{\text{Hom}} & A \\
\downarrow F & & \downarrow G \\
C & \xrightarrow{\text{Hom}} & C
\end{array}
\]

\[
\forall a, b. \text{Hom}(a, b) \Rightarrow \text{Hom}(F a, G b)
\]

\[
\forall a. \text{Hom}(F a, G a)
\]
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only
- (1)-arrows \(\sim\) 1-arrows
- (1,2)-squares \(\sim\) 2-arrows can be non-trivial.

\[
\begin{array}{c}
a \\ \phi \\ c \\
\end{array}
\xrightarrow[	ext{id}(2)]{}
\begin{array}{c}
a \\ \chi \\ c \\
\end{array}
\]

Recall the equipment Cat:

\[
\begin{array}{ccc}
A & \xrightarrow{\text{Hom}} & A \\
F & \downarrow & G \\
C & \xrightarrow{\text{Hom}} & C \\
\end{array}
\]

\[
\forall \ a, b. \text{Hom}(a, b) \Rightarrow \text{Hom}(F a, G b)
\]

\[
\exists \ a. \text{Hom}(F a, G a)
\]

\(\Rightarrow\)

\section*{n-category (Tamsamani & Simpson)}

An \(n\)-category is an **\(n\)-fold** category where only
- (1)-arrows, \(\sim\) 1-arrows
- (1,2)-squares, \(\sim\) 2-arrows
- (1,2,3)-cubes, \(\sim\) 3-arrows \ldots

can be non-trivial.
2-category (Tamsamani & Simpson)

An 2-category is an **double (2-fold)** category whose (2)-arrows are all trivial (id), so only

- (1)-arrows \(\sim\) 1-arrows
- (1,2)-squares \(\sim\) 2-arrows

can be non-trivial.

\[
\begin{array}{c}
a \xrightarrow{\text{id}^{(2)}} a \\
\varphi \quad \Rightarrow \\
\chi \\
c \xrightarrow{\text{id}^{(2)}} c
\end{array}
\]

\[
\begin{array}{c}
a \\
\varphi \quad \Rightarrow \\
\chi \\
c
\end{array}
\]

Recall the equipment \(\text{Cat}\):

\[
\begin{array}{c}
\mathcal{A} \\
F \\
C
\end{array}
\xrightarrow{\mathcal{P}}
\begin{array}{c}
\mathcal{B} \\
G \\
D
\end{array}
\]

\[
\begin{array}{c}
\mathcal{A} \\
F \quad \Rightarrow \\
C
\end{array}
\]

\[
\begin{array}{c}
\mathcal{A} \\
F \\
C
\end{array}
\xrightarrow{\mathcal{P}}
\begin{array}{c}
\mathcal{B} \\
G \\
D
\end{array}
\]

\[
\begin{array}{c}
\mathcal{A} \\
F \quad \Rightarrow \\
C
\end{array}
\]

\[\forall a, b. \text{Hom}(a, b) \Rightarrow \text{Hom}(F a, G b) \]

\[\forall a. \text{Hom}(F a, G a) \]

\[\cong\]

\[\forall a, b. \text{Hom}(a, b) \Rightarrow \text{Hom}(F a, G b)\]

\[\forall a. \text{Hom}(F a, G a)\]

\[\cong\]
### Status of the model

- The building blocks are there (also further ahead!).
- Sort out details of base category & modalities. (Not success/failure but descriptive.)

### Conclusion

We are not stuck on higher directed type theory.

Thanks!

Questions?
Status of the model

- The building blocks are there (also further ahead!).
- Sort out details of base category & modalities. (Not success/failure but descriptive.)

Conclusion

We are **not** stuck on *higher* directed type theory.

Thanks!

Questions?