Today's Plan

Our Target

Model Construction

Classifying Class Theories

Order and Continuity

Order Models
OUR TARGET
\[ \alpha \in \{ x : \phi(x) \} \iff \phi(\alpha) \]
\( \alpha \in \lambda x. \phi(x) \iff \phi(\alpha) \)
Russell's Paradox

\{x : x \not\in x\} \in \{x : x \not\in x\} \iff \{x : x \not\in x\} \not\in \{x : x \not\in x\}
Russell's Paradox

\{x : x \not\in x\} \in \{x : x \not\in x\} \text{ iff } \{x : x \not\in x\} \not\in \{x : x \not\in x\}

In general,

\{x : F(x \in x)\} \in \{x : F(x \in x)\} \text{ iff } 
F(\{x : F(x \in x)\} \in \{x : F(x \in x)\})
The Heterological Paradox

\[ \lambda x. (x \notin x) \in \lambda x. (x \notin x) \text{ iff } \lambda x. (x \notin x) \notin \lambda x. (x \notin x) \]
The Heterological Paradox

\[ \lambda x. (x \not\in x) \in \lambda x. (x \not\in x) \iff \lambda x. (x \not\in x) \not\in \lambda x. (x \not\in x) \]

In general,

\[ \lambda x. F(x \in x) \in \lambda x. F(x \in x) \iff F(\lambda x. F(x \in x) \in \lambda x. F(x \in x)) \]
Extensionality

If $a$ and $b$ have the same members, then $a = b$. 
If $a$ and $b$ have the same members, then $a = b$.

$$
\Gamma, x \in a \vdash x \in b, \Delta \quad \Gamma, x \in b \vdash x \in a, \Delta
$$

$$
\Gamma \vdash a = b, \Delta
$$
Extensionality

If \( a \) and \( b \) have the same members, then \( a = b \).

\[
\Gamma, x \in a \vdash x \in b, \Delta \quad \Gamma, x \in b \vdash x \in a, \Delta
\]

\[
\Gamma \vdash a = b, \Delta
\]

(Extensionality will not play a significant role in what follows.)
MODEL
CONSTRUCTION
What are Models For?

*Defining* validity.
What are Models For?

Defining validity.

Providing counterexamples, including proving non-triviality.
What are Models For?

Defining validity.

Providing counterexamples, including proving non-triviality.

Relating theories.
What are Models For?

- Defining validity.
- Providing counterexamples, including proving non-triviality.
- Relating theories.
- Giving a sense of what the theory can be about.
What are Models For?

Defining validity.

Providing counterexamples, including proving non-triviality.

Relating theories.

Giving a sense of what the theory can be about.

Motivating the theory.
What are Models For?

Defining validity.

Providing counterexamples, including proving non-triviality.

Relating theories.

Giving a sense of what the theory can be about.

Motivating the theory.
What are Models For?

**Defining validity.**

Providing *counterexamples*, including *proving non-triviality*.

**Relating theories.**

Giving a sense of what the theory can be *about*.

**Motivating the theory.**
These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be about.
These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be about.
These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be about.
These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be about.

\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}

an \(a\) where \(a = \{a\}\)
ZFC and its Cousins: Anti-Foundation

These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be about.

\[
\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}
\]

an \(a\) where \(a = \{a\}\)
These models are good for (1) relating ZFC to AFA,
These models are good for (1) relating ZFC to AFA, (2) motivating a choice of the anti-foundation axiom,
ZFC and its Cousins: Anti-Foundation

These models are good for (1) *relating* ZFC to AFA, (2) motivating a choice of the anti-foundation axiom, and (3) explaining what the theory could be *about*.
If $x$ is a variable and $M$ is a term, $\lambda x. M$ is a term.
If $x$ is a variable and $M$ is a term, $\lambda x. M$ is a term.

For any terms, $M$ and $N$, $MN$ is $M$ applied to $N$. 
Untyped $\lambda$ Calculus

If $x$ is a variable and $M$ is a term, $\lambda x.M$ is a term.

For any terms, $M$ and $N$, $MN$ is $M$ applied to $N$.

$$(\lambda x.M)N = M[x := N].$$
Models of the Untyped $\lambda$ Calculus

You bump up against Cantor's Theorem.
Models of the Untyped λ Calculus

You bump up against Cantor's Theorem.
Models of the Untyped \(\lambda\) Calculus

\[
D \cong D \rightarrow D
\]

You bump up against Cantor’s Theorem.
$D \simeq [D \to D]$
[D → E]: the order preserving functions from (D, ⊑) to (E, ⊑).
[\mathcal{D} \rightarrow \mathcal{E}]: the order preserving functions from \((\mathcal{D}, \sqsubseteq)\) to \((\mathcal{E}, \sqsubseteq)\).

It’s ordered too: \(f \sqsubseteq g\) iff \((\forall x)(f(x) \sqsubseteq g(x))\).
The Scott Construction

\[ [D \to E] : \text{the order preserving functions from } (D, \sqsubseteq) \text{ to } (E, \sqsubseteq). \]

It’s ordered too: \( f \sqsubseteq g \iff (\forall x)(f(x) \sqsubseteq g(x)) \).

Embed \( D_i \) into \( [D_i \to D_i] = D_{i+1} \)
(Use the constant functions.)
The Scott Construction

\([D \rightarrow E]\): the order preserving functions from \((D, \sqsubseteq)\) to \((E, \sqsubseteq)\).

It’s ordered too: \(f \sqsubseteq g\) iff \((\forall x)(f(x) \sqsubseteq g(x))\).

Embed \(D_i\) into \([D_i \rightarrow D_i] = D_{i+1}\)
(Use the constant functions.)

Let \(D_\infty\) be the limit: \(D_\infty \cong [D_\infty \rightarrow D_\infty]\).

This is a model of the untyped \(\lambda\) calculus.
Truth Theories: Kripke, Woodruff, Gilmore, Brady

\[
\begin{align*}
M_0 & \quad \cdots \quad M_n & \quad M_{n+1} & \quad M_k & \quad M_{k+1} \\
0 & \quad A & \quad \sim & \quad T\langle A \rangle & \quad A & \quad = & \quad T\langle A \rangle
\end{align*}
\]
This is not like the other model constructions: the domain is constant—the terms \( f_x : \phi(x) \). This shows what the theory is about in only a very weak sense.
This is not like the other model constructions: 
the domain is constant—the terms \( \{ x : \phi(x) \} \).
This is not like the other model constructions: the domain is constant—the terms \( \{ x : \phi(x) \} \).

This shows what the theory is \textit{about} in only a very weak sense.
CLASSIFYING CLASS THEORIES
Gaps or Gluts?
Underlying Logic: Negation

Gaps or Gluts?

Paraconsistent or Paracomplete?
Do we have a conditional in the language?
Do we have a conditional in the language?

And if so, what is it like?
These decisions are not *that* important.
Underlying Logic: Not *that* important

These decisions are not *that* important.

The logic must allow for *fixed points*.
These decisions are not *that* important.

The logic must allow for *fixed points*.

For *any* sentence context $F(-)$, we need to allow for some $p$ to be *equivalent to* $F(p)$. If $c =_{df} \{ x : F(x \in x) \}$, then $c \in c$ *iff* $F(c \in c)$.
D

- D: the *ordinary* domain.
D: the *ordinary* domain.
- $D$: the *ordinary* domain.
- $\Omega$: truth values.
Semantic Values

- D: the *ordinary* domain.
- \( \Omega \): truth values.
- C: the classes
Semantic Values

\[ C \ (C \cup D) \rightarrow \Omega \]

- D: the *ordinary* domain.
- \( \Omega \): truth values.
- C: the classes
Semantic Values

\[ C \cong (C \cup D) \rightarrow \Omega \]

- **D**: the *ordinary* domain.
- **Ω**: truth values.
- **C**: the classes
Semantic Values

\[ C \cong [(C \cup D) \rightarrow \Omega] \]

- \(D\): the ordinary domain.
- \(\Omega\): truth values.
- \(C\): the classes
Extensionality

We won’t focus on extensionality here.
We won’t focus on extensionality here.

But we’ll *identify* classes by their extensions as much as possible.
Sharpening our Target

\[ C \cong [C \cup D \to \Omega] \]
Sharpening our Target

\[ C \cong [C \cup D \rightarrow \Omega] \]

\(\phi(x)\) gives a function \([C \cup D \rightarrow \Omega]\).

So we can find a class \(C\) to *match*.

\(a \in \{x : \phi(x)\}\) has the *same* value in \(\Omega\) as \(\phi(a)\).
ORDER AND CONTINUITY
Underlying Logic: Preservation
Underlying Logic: Preservation

Ω is ordered by ⊑.

\[ \sqsubseteq \]
$\Omega$ is ordered by $\sqsubseteq$.

All connectives & quantifiers are $\sqsubseteq$-order preserving.
Underlying Logic: Preservation

\[ \Omega \]

\( \Omega \) is ordered by \( \sqsubseteq \).

All connectives & quantifiers are \( \sqsubseteq \)-order preserving.

(If \( x \sqsubseteq x' \) and \( y \sqsubseteq y' \) then \( x \uparrow y \sqsubseteq x' \uparrow y' \), etc.)
Preservation on candidates for $\Omega$

In $Ł^3$, $1! = 0$; but $0! = 1$

Similar behaviour here.

In $\text{RM}^3$, $1! = 0$; but $0! = 1$.
Preservation on candidates for $\Omega$

$\begin{array}{c}
0 \\
\rightarrow \\
*
\rightarrow \\
1
\end{array}$

$K_3$ or LP

In $\mathcal{L}$, $1! = 0$; but $0! = 1$.

Similar behaviour here.

In $\mathcal{R}$, $1! = 0$; but $0! = 1$. 
Preservation on candidates for $\Omega$

K₃ or LP, but not Ł₃

In Ł₃, $\star \rightarrow \star$ is 1; but $1 \rightarrow 0$ is 0
Preservation on candidates for $\Omega$

In Ł3, $* \rightarrow *$ is 1; but $1 \rightarrow 0$ is 0

In RM3, $1 \rightarrow *$ is 0; but $1 \rightarrow 1$ is 1
Preservation on candidates for $\Omega$

K₃ or LP, but not Ł₃ or RM₃

In Ł₃, $* \rightarrow *$ is 1; but $1 \rightarrow 0$ is 0

In RM₃, $1 \rightarrow *$ is 0; but $1 \rightarrow 1$ is 1
Preservation on candidates for $\Omega$

$K_3$ or LP, but not $\mathcal{L}_3$ or $RM_3$

In $\mathcal{L}_3$, $* \rightarrow *$ is 1; but $1 \rightarrow 0$ is 0

In $RM_3$, $1 \rightarrow *$ is 0; but $1 \rightarrow 1$ is 1

FDE, but no robust conditionals.
Preservation on candidates for $\Omega$

K₃ or LP, but not Ł₃ or RM₃
In Ł₃, $\ast \rightarrow \ast$ is 1; but 1 → 0 is 0
In RM₃, 1 → $\ast$ is 0; but 1 → 1 is 1

FDE, but no robust conditionals.
Similar behaviour here.
Many other choices for $\Omega$ are possible.
Many other choices for $\Omega$ are possible.

Even $\{0, 1\}$ can be ordered: $0 \sqsubseteq 1$. Then $\land, \lor, 0, 1$ are order preserving, but $\neg$ and $\supseteq$ are not order preserving.
3: our choice of $\Omega$

\[ 0 \xleftarrow{} * \xrightarrow{} 1 \]
(I really don’t care if you think of * as true, or as untrue.)
ORDER MODELS
Given an order algebra $\Omega$, and a domain $D$ of urelements

\[ \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \text{ is a } \langle D, \Omega \rangle \text{ order model iff} \]

- $\uparrow$ is a partial order on $C$.
- $\ast : C \rightarrow [C \rightarrow D] \rightarrow \Omega$ is order preserving and invertible.
- $+ : [C \rightarrow D] \rightarrow \Omega$, where $+ = \ast ^{-1}$, is also order preserving.

Write $\ast (c)$ as $c^\ast$ and $+ (f)$ as $f^+$. So $c^\ast + = c$ and $f^+ \ast = f$. 

- If $b \in C \in D$ and $c \in C$, then $c \ast (b)$ tells you whether $b$ is in $c$. 
Defining Order Models

Given an order algebra $\Omega$, and a domain $D$ of urelements

$\langle C, \sqsubseteq, \uparrow, \downarrow \rangle$ is a $\langle D, \Omega \rangle$ order model iff

- $\sqsubseteq$ is a partial order on $C$. 

- Write $\ast(c)$ as $c^\ast$ and $\ast(f)$ as $f^\ast$. So $c^\ast + = c$ and $f +^\ast = f$. 

- If $b \in C$ and $c \in C$, then $c^\ast(b)$ tells you whether $b$ is in $c$. 

Greg Restall

Fixed Point Models, for Theories of, Properties and Classes
Defining Order Models

Given an order algebra Ω, and a domain D of urelements

\[ \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \text{ is a } \langle D, \Omega \rangle \text{ order model iff} \]

- \( \sqsubseteq \) is a partial order on C.
- \( \uparrow : C \rightarrow [C \cup D \rightarrow \Omega] \) is order preservering and invertible.
Defining Order Models

Given an order algebra \( \Omega \), and a domain \( D \) of urelements

\[ \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \text{ is a } \langle D, \Omega \rangle \text{ order model iff} \]

- \( \sqsubseteq \) is a partial order on \( C \).
- \( \uparrow : C \rightarrow [C \cup D \rightarrow \Omega] \) is order preserving and invertible.
- \( \downarrow : [C \cup D \rightarrow \Omega] \rightarrow C \), where \( \downarrow = \uparrow^{-1} \), is also order preserving.
Defining Order Models

Given an order algebra $\Omega$, and a domain $D$ of urelements

$$\langle C, \sqsubseteq, \uparrow, \downarrow \rangle$$

is a $\langle D, \Omega \rangle$ order model iff

- $\sqsubseteq$ is a partial order on $C$.
- $\uparrow : C \rightarrow [C \cup D \rightarrow \Omega]$ is order preserving and invertible.
- $\downarrow : [C \cup D \rightarrow \Omega] \rightarrow C$, where $\downarrow = \uparrow^{-1}$, is also order preserving.

- Write ‘$\uparrow(c)$’ as ‘$c_{\uparrow}$’ and ‘$\downarrow(f)$’ as ‘$f_{\downarrow}$’. So $c_{\uparrow\downarrow} = c$ and $f_{\downarrow\uparrow} = f$. 
Defining Order Models

Given an order algebra \( \Omega \), and a domain \( D \) of urelements

\[
\langle C, \sqsubseteq, \uparrow, \downarrow \rangle \text{ is a } \langle D, \Omega \rangle \text{ order model iff}
\]

- \( \sqsubseteq \) is a partial order on \( C \).
- \( \uparrow : C \to [C \cup D \to \Omega] \) is order preserving and invertible.
- \( \downarrow : [C \cup D \to \Omega] \to C \), where \( \downarrow = \uparrow^{-1} \), is also order preserving.

- Write ‘\( \uparrow(c) \)’ as ‘\( c_{\uparrow} \)’ and ‘\( \downarrow(f) \)’ as ‘\( f_{\downarrow} \)’. So \( c_{\uparrow\downarrow} = c \) and \( f_{\downarrow\uparrow} = f \).
- If \( b \in C \cup D \) and \( c \in C \), then \( c_{\uparrow}(b) \) tells you whether \( b \) is in \( c \).
Membership is order preserving

If \( x \sqsubseteq x' \) and \( y \sqsubseteq y' \) then \( x \uparrow(y) \sqsubseteq x' \uparrow(y') \).
Membership is order preserving

If $x \sqsubseteq x'$ and $y \sqsubseteq y'$ then $x \uparrow(y) \sqsubseteq x' \uparrow(y')$.

$x \uparrow(y) \sqsubseteq x \uparrow(y')$ — $y \sqsubseteq y'$ and $x \uparrow$ is order preserving.
Membership is order preserving

If $x \sqsubseteq x'$ and $y \sqsubseteq y'$ then $x \uparrow(y) \sqsubseteq x' \uparrow(y')$.

$x \uparrow(y) \sqsubseteq x \uparrow(y')$ — $y \sqsubseteq y'$ and $x \uparrow$ is order preserving.

$x \uparrow \sqsubseteq x' \uparrow$ — $x \sqsubseteq x'$ and $\uparrow$ is order preserving.
Membership is order preserving

If \( x \sqsubseteq x' \) and \( y \sqsubseteq y' \) then \( x_{\uparrow}(y) \sqsubseteq x'_{\uparrow}(y') \).

\[
\begin{align*}
    x_{\uparrow}(y) & \sqsubseteq x_{\uparrow}(y') & \text{— } & y \sqsubseteq y' \text{ and } x_{\uparrow} \text{ is order preserving.} \\
    x_{\uparrow} & \sqsubseteq x'_{\uparrow} & \text{— } & x \sqsubseteq x' \text{ and } \uparrow \text{ is order preserving.} \\
    x_{\uparrow}(y') & \sqsubseteq x'_{\uparrow}(y') & \text{— } & \text{by the definition of } \sqsubseteq \text{ for functions.}
\end{align*}
\]
Given a \( \langle D, 3 \rangle \) order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),
Interpreting a Language

Given a \( \langle D, 3 \rangle \) order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),

- An assignment \( \alpha \), takes variables to values in \( C \cup D \).
Given a $\langle D, 3 \rangle$ order model $\mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle$, 

- An assignment $\alpha$, takes variables to values in $C \cup D$.
- $[[x]]_{\mathcal{M}, \alpha} = \alpha(x)$ is the interpretation of the variable $x$. 

Connectives and quantifiers are interpreted as usual. 

(Connectives and quantifiers are order preserving functions on $\langle C \cup D, 3 \rangle$.)
Given a \( \langle D, 3 \rangle \) order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),

- An assignment \( \alpha \), takes variables to values in \( C \cup D \).
- \( [x]_{\mathcal{M}, \alpha} = \alpha(x) \) is the interpretation of the variable \( x \).
- (We abbreviate this \( [x] \) when \( \mathcal{M} \) and \( \alpha \) is clear.)
Interpreting a Language

Given a \langle D, 3 \rangle order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),

- An assignment \( \alpha \), takes variables to values in \( C \cup D \).
  - \( \llbracket x \rrbracket_{\mathcal{M}, \alpha} = \alpha(x) \) is the interpretation of the variable \( x \).
- (We abbreviate this \( \llbracket x \rrbracket \) when \( \mathcal{M} \) and \( \alpha \) is clear.)

- \( \llbracket s \in t \rrbracket_{\mathcal{M}, \alpha} \) is \( \llbracket t \rrbracket_{\uparrow}(\llbracket s \rrbracket) \) when \( \llbracket t \rrbracket \in C \),
  and is 0 when \( \llbracket t \rrbracket \in D \).
Interpreting a Language

Given a \( \langle D, 3 \rangle \) order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),

- An assignment \( \alpha \), takes variables to values in \( C \cup D \).
  - \( \llbracket x \rrbracket_{\mathcal{M}, \alpha} = \alpha(x) \) is the interpretation of the variable \( x \).
  - (We abbreviate this \( \llbracket x \rrbracket \) when \( \mathcal{M} \) and \( \alpha \) is clear.)
  - \( \llbracket s \in t \rrbracket_{\mathcal{M}, \alpha} \) is \( \llbracket t \rrbracket_{\uparrow}(\llbracket s \rrbracket) \) when \( \llbracket t \rrbracket \in C \), and is 0 when \( \llbracket t \rrbracket \in D \).
- Connectives and quantifiers are interpreted as usual.
Given a \( \langle D, 3 \rangle \) order model \( \mathcal{M} = \langle C, \sqsubseteq, \uparrow, \downarrow \rangle \),

- An assignment \( \alpha \), takes variables to values in \( C \cup D \).
  \[ \llbracket x \rrbracket_{\mathcal{M}, \alpha} = \alpha(x) \] is the interpretation of the variable \( x \).
- (We abbreviate this \( \llbracket x \rrbracket \) when \( \mathcal{M} \) and \( \alpha \) is clear.)
  \[ \llbracket s \in t \rrbracket_{\mathcal{M}, \alpha} \text{ is } \llbracket t \rrbracket \uparrow (\llbracket s \rrbracket) \text{ when } \llbracket t \rrbracket \in C, \]
  and is 0 when \( \llbracket t \rrbracket \in D \).
- Connectives and quantifiers are interpreted as usual.
  (Connectives and quantifiers are order preserving functions on 3 or \( C \cup D \rightarrow 3 \).)
Extending the Language with Terms

\[ \{ x : \phi(x) \} \]
Extending the Language with Terms

\[ \{x : \phi(x)\} \]

Since \( \llbracket \phi(x) \rrbracket_m,\alpha[x := v] \) is order preserving in \( v \) we can use that function, in \( [C \cup D \rightarrow 3] \), to select the extension of \( \{x : \phi(x)\} \).
Extending the Language with Terms

\[ \{ x : \phi(x) \} \]

Since \( [\phi(x)]_m,\alpha[x := v] \) is order preserving in \( v \),
we can use that function, in \([C \cup D \rightarrow 3]\),
to select the extension of \( \{ x : \phi(x) \} \).

\[ \llbracket \{ x : \phi(x) \} \rrbracket_m,\alpha = (\lambda v. [\phi(x)]_m,\alpha[x := v]) \downarrow \]
Strong Comprehension

\[ [t \in \{ x : \phi(x) \}]_{m, \alpha} \]
Strong Comprehension

\[
[t \in \{x : \phi(x)\}]_\mathcal{M},\alpha = \llbracket\{x : \phi(x)\}\rrbracket_\alpha(\llbracket t \rrbracket_\alpha)
\]

(I’ve dropped reference to \(\mathcal{M}\) as it is constant throughout.)
\[ [t \in \{ x : \phi(x) \}]_{\mathcal{M},\alpha} = \llbracket \{ x : \phi(x) \} \rrbracket_{\alpha \uparrow} (\llbracket t \rrbracket_{\alpha}) \]
\[ = (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha[x := v]}_{\downarrow \uparrow}) (\llbracket t \rrbracket_{\alpha}) \]

(I’ve dropped reference to \( \mathcal{M} \) as it is constant throughout.)
\[[t \in \{x : \phi(x)\}]_m,\alpha \quad = \quad \llbracket \{x : \phi(x)\} \rrbracket_{\alpha \uparrow} (\llbracket t \rrbracket_{\alpha})
= \quad (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha[x := v]})(\downarrow\uparrow (\llbracket t \rrbracket_{\alpha}))
= \quad (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha[x := v]})(\llbracket t \rrbracket_{\alpha})\]

(I’ve dropped reference to \(M\) as it is constant throughout.)
Strong Comprehension

\[
[t \in \{x : \phi(x)\}]_{m,\alpha} = \llbracket \{x : \phi(x)\} \rrbracket_{\alpha \uparrow} (\llbracket t \rrbracket_{\alpha}) \\
= (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha[x := v]} \downarrow \uparrow (\llbracket t \rrbracket_{\alpha}) \\
= (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha[x := v]} (\llbracket t \rrbracket_{\alpha}) \\
= \llbracket \phi(x) \rrbracket_{\alpha[x := \llbracket t \rrbracket_{\alpha}}
\]

(I’ve dropped reference to \(M\) as it is constant throughout.)
[t ∈ \{x : \phi(x)\}]_{m,\alpha} = \llbracket\{x : \phi(x)\}\rrbracket_{\alpha}^{\uparrow}(\llbracket t \rrbracket_{\alpha})

= (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha}[x := v])_{\downarrow\uparrow}(\llbracket t \rrbracket_{\alpha})

= (\lambda v. \llbracket \phi(x) \rrbracket_{\alpha}[x := v])(\llbracket t \rrbracket_{\alpha})

= \llbracket \phi(x) \rrbracket_{\alpha}[x := \llbracket t \rrbracket_{\alpha}]

= \llbracket \phi(t) \rrbracket_{m,\alpha}

(I’ve dropped reference to \(M\) as it is constant throughout.)
\[ [t \in \{x : \phi(x)\}]_{\mathcal{M},\alpha} = \langle\{x : \phi(x)\}\rangle_{\alpha}^{\uparrow}([t]_{\alpha}) \]
\[ = (\lambda v. [\phi(x)]_{\alpha[x := v]}^{\uparrow}[t]_{\alpha}) \]
\[ = (\lambda v. [\phi(x)]_{\alpha[x := v]}([t]_{\alpha})) \]
\[ = [\phi(x)]_{\alpha[x := [t]_{\alpha}]} \]
\[ = [\phi(t)]_{\mathcal{M},\alpha} \]

(I’ve dropped reference to \(\mathcal{M}\) as it is constant throughout.)
Logical Constants
Logical Constants

0 * 1
\( \Lambda = \{ x : 0 \} \)
\[ \Lambda = \{ x : 0 \} \quad x \in \Lambda \text{ is always false.} \]
\( \Lambda, \text{ } V \text{ and } \mathfrak{X} \)

\[
\Lambda = \{x : 0\} \quad x \in \Lambda \text{ is always false.}
\]

\[
V = \{x : 1\}
\]
Λ, V and Ξ

\[ \Lambda = \{ x : 0 \} \quad x \in \Lambda \text{ is always false.} \]

\[ V = \{ x : 1 \} \quad x \in V \text{ is always true.} \]
\(\Lambda = \{x : 0\}\)  \(x \in \Lambda\) is always false.

\(\forall = \{x : 1\}\)  \(x \in \forall\) is always true.

\(\mathcal{X} = \{x : *\}\)
\[ \Lambda = \{ x : 0 \} \quad x \in \Lambda \text{ is always false.} \]

\[ V = \{ x : 1 \} \quad x \in V \text{ is always true.} \]

\[ \mathcal{X} = \{ x : * \} \quad x \in \mathcal{X} \text{ is always *}. \]
Ordering the Classes

In fact, \([V] \sqsubseteq c\) for every class \(c \in C\).

From now, we'll use '∅', 'V' and 'V' as both the class terms in the language, and as their denotations, names for objects in \(C\).
In fact, $[V] \sqsubseteq c$ for every class $c \in C$.

From now, we'll use '$\emptyset$', 'V' and 'V' as both the class terms in the language, and as their denotations, names for objects in $C$. 
In fact, $[\mathcal{M}] \subseteq c$ for every class $c \in C$. 
Ordering the Classes

In fact, \([\mathcal{X}] \sqsubseteq c\) for every class \(c \in C\).

From now, we’ll use ‘\(\emptyset\)’, ‘\(\forall\)’ and ‘\(\mathcal{X}\)’ as both the class terms in the language, and as their denotations, names for objects in \(C\).
In a model $\mathcal{M}$, a class $c$ is **SHARP** iff for each object $b$ in $C \cup D$, $c^*(b)$ takes the value 0 or 1.
In a model $\mathcal{M}$, a class $c$ is **SHARP** iff for each object $b$ in $C \cup D$
$c(\uparrow b)$ takes the value 0 or 1

$\Lambda$ and $\forall$ are sharp.
In a model $\mathcal{M}$, a class $c$ is **SHARP** iff for each object $b$ in $C \cup D$
\[c_\uparrow(b)\] takes the value 0 or 1

$\Lambda$ and $\forall$ are sharp.

$\mathcal{W}$ is *not* sharp.
Almost No Classes are *Sharp*

If $c_{\uparrow}(b) = 1$ and $c_{\uparrow}(b') = 0$, then $c_{\uparrow}(\mathcal{X}) = \ast$. 
Almost No Classes are *Sharp*

If $c_{\uparrow}(b) = 1$ and $c_{\uparrow}(b') = 0$, then $c_{\uparrow}(\mathcal{X}) = \ast$.

$\mathcal{X} \subseteq b$, so $c_{\uparrow}(\mathcal{X}) \subseteq c_{\uparrow}(b) = 1$. 
Almost No Classes are *Sharp*

If \( c_{\uparrow}(b) = 1 \) and \( c_{\uparrow}(b') = 0 \), then \( c_{\uparrow}(\mathcal{X}) = * \).

\( \mathcal{X} \sqsubseteq b \), so \( c_{\uparrow}(\mathcal{X}) \sqsubseteq c_{\uparrow}(b) = 1 \).

\( \mathcal{X} \sqsubseteq b' \), so \( c_{\uparrow}(\mathcal{X}) \sqsubseteq c_{\uparrow}(b') = 0 \).
Almost No Classes are Sharp

If $c \uparrow (b) = 1$ and $c \uparrow (b') = 0$, then $c \uparrow (\mathcal{X}) = *$.

$\mathcal{X} \sqsubseteq b$, so $c \uparrow (\mathcal{X}) \sqsubseteq c \uparrow (b) = 1$.

$\mathcal{X} \sqsubseteq b'$, so $c \uparrow (\mathcal{X}) \sqsubseteq c \uparrow (b') = 0$.

It follows that $c \uparrow (\mathcal{X}) = *$.
There is no *classical recapture* through crisp classes.

Once a class *includes* something and *excludes* something, it is *indecisive* about $\mathcal{X}$. 
There is no classical recapture through crisp classes

Once a class \textit{includes} something and \textit{excludes} something, it is \textit{indecisive} about \( \mathcal{X} \).

It follows that there are no \textit{crisp singletons}: objects \( \{a\} \) for which \( \llbracket a \in \{x\} \rrbracket = 1 \) and \( \llbracket b \in \{x\} \rrbracket = 0 \) for all other \( b \).
Singletons and Anti-Signetons: \( \{t\} \) and \( \{t\} \)

\[\text{\( \llbracket \{t\} \rrbracket_\alpha: \) (the class representative of) the function that}\]
\[\begin{align*}
\text{\quad – assigns } 1 \text{ to } x \text{ iff } \llbracket t \rrbracket_\alpha \sqsubseteq x, \\
\text{\quad – and } 0 \text{ to } x \text{ iff there is no } z \text{ where } x \sqsubseteq z \text{ and } \llbracket t \rrbracket_\alpha \sqsubseteq z, \\
\text{\quad – and } * \text{ otherwise.}
\end{align*}\]

\[\text{\( \llbracket \{t\} \rrbracket_\alpha: \) (the class representative of) the function that}\]
\[\begin{align*}
\text{\quad – assigns } 0 \text{ to } x \text{ iff } \llbracket t \rrbracket_\alpha \sqsubseteq x, \text{ and} \\
\text{\quad – and } 1 \text{ to } x \text{ if there is no } z \text{ where } x \sqsubseteq z \text{ and } \llbracket t \rrbracket_\alpha \sqsubseteq z, \\
\text{\quad – and } * \text{ otherwise.}
\end{align*}\]
From Here

- Study pure order models (where $D$ is empty), and impure order models.
- Find perspicuous ways to construct order models.
- Relate these constructions to other known model constructions.
- Axiomatise the logic of order models.
- Examine different motivations of order models.
Study *pure* order models (where $D$ is empty),
Study *pure* order models (where \(D\) is empty), ... and *impure* order models.
Study *pure* order models (where $D$ is empty),
... and *impure* order models.

Find perspicuous ways to *construct* order models.
From Here

- Study *pure* order models (where $D$ is empty),
  ... and *impure* order models.
- Find perspicuous ways to *construct* order models.
- Relate these constructions to other known model constructions.
From Here

- Study *pure* order models (where $D$ is empty), ... and *impure* order models.
- Find perspicuous ways to *construct* order models.
- Relate these constructions to other known model constructions.
- *Axiomatise* the logic of order models.
Study *pure* order models (where $D$ is empty),
... and *impure* order models.

Find perspicuous ways to *construct* order models.

Relate these constructions to other known model constructions.

*Axiomatise* the logic of order models.

Examine different *motivations* of order models.
THANK YOU!