

Semantic Theory

Week 2 – Type Theory

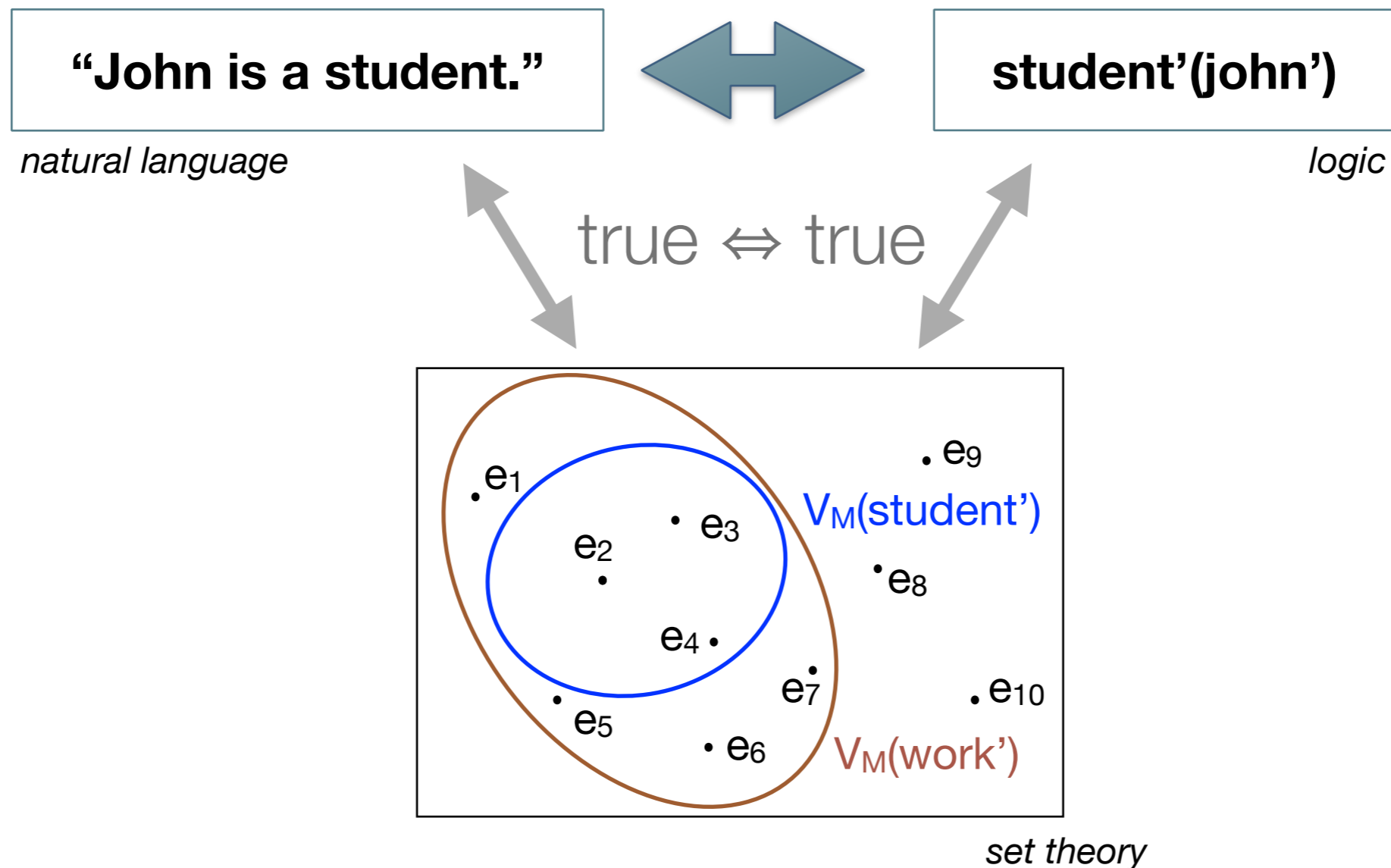
Noortje Venhuizen
Harm Brouwer

Universität des Saarlandes

Summer 2021

Truth-conditional semantics

- **Assumption:** Logical formula captures truth-conditions of NL sentence; they are true in the same possible models.



First-order logic

First-order logic talks about:

- Individual objects: $V_M(\text{'john'}) = e$; $g(x) = e$
- Properties of and relations between individual objects:
 $\text{'happy'}(\text{'john'})$; $\text{'love'}(\text{'john'}, \text{'mary'})$
- Quantification over individual objects: $\forall x(\text{happy}(x))$

Limitations of first-order logic

FOL is not expressive enough to capture all meanings that can be expressed by basic natural language expressions:

Jumbo is a small elephant.

(Predicate modifiers)

Happy is a state of mind.

(Second-order predicates)

Yesterday, it rained.

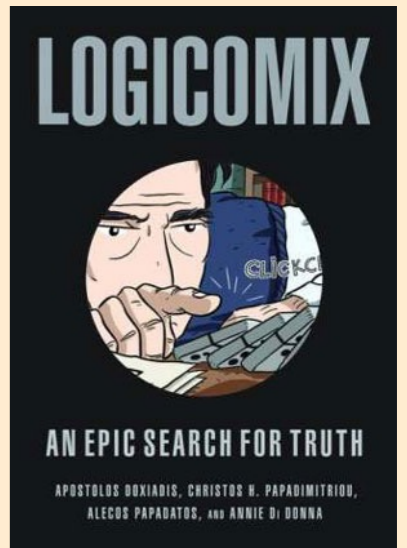
(Non-logical sentence operators)

Bill and John have the same hair color.

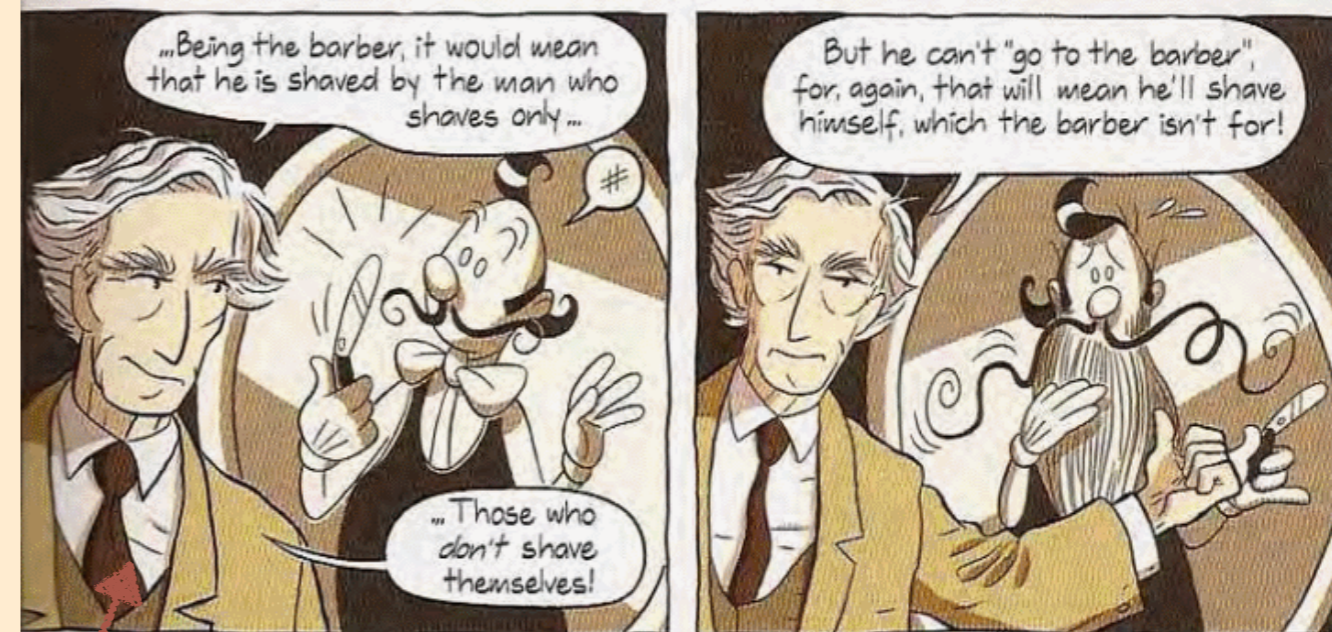
(Higher-order quantification)

→ *What logically sound system can capture this diversity?*

Introducing Russell's paradox

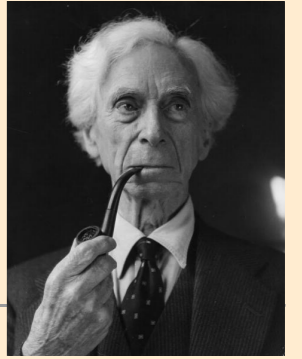


From: **Logicomix — An epic search for truth**; A. Doxiadis, C.H. Papadimitriou, A. Papadatos and A. Di Donna



Bertrand Russell

Russell's paradox for Higher-Order Logic



What if we extend the FOL interpretation of predicates, and simply interpret higher-order predicates as sets of sets of properties?

Then, for every predicate P , we can define a set $\{x \mid P(x)\}$ containing all and only those entities for which P holds.

Now what if we define a set $S = \{X \mid X \notin X\}$ representing the set of all sets that are not members of itself..

Paradox: does S belong to itself?

If it does, then S must satisfy its constraints, namely that it doesn't belong to itself, which is not possible if we assume it belongs to S .

If not, then S is a set that doesn't belong to itself, hence it belongs to S .

→ **Conclusion:** We need a more restricted way of talking about *properties and relations between properties!*

Type Theory

In Type Theory, all logical expressions are assigned a *type* (that may be basic or complex), which restricts how they can be combined.

Basic types:

- **e** – the type of individual terms (“entities”)
- **t** – the type of formulas (“truth-values”)



Complex types:

- If σ , τ are types, then $\langle \sigma, \tau \rangle$ is a type

↘ **Note: this is a “tau” (the Greek letter), not a “t”!**

This represents a functor expression that takes an expression of type σ as its argument and returns an expression of type τ ; this functor is sometimes written as $(\sigma \rightarrow \tau)$ or simply $(\sigma\tau)$ (as in Winter-EFS)

Types & Function Application

Types of first-order expressions:

- Individual constants (Luke, Death Star) : \mathbf{e} (Entity)
- One-place predicates (walk, jedi): $\langle \mathbf{e}, \mathbf{t} \rangle$ (Function from entities to truth values; a property)
- Two-place predicates (admire, fight with): $\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$ (Function from entities to properties)
- Three-place predicates (give, introduce): $\langle \mathbf{e}, \langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle \rangle$
(Function from entities to functions from entities to properties)

Function application: Combining a functor of complex type $\langle \mathbf{a}, \mathbf{\beta} \rangle$ with an appropriate argument of type \mathbf{a} , results in an expression of type $\mathbf{\beta}$: $\langle \mathbf{a}, \mathbf{\beta} \rangle(\mathbf{a}) \mapsto \mathbf{\beta}$

- $\text{jedi}'(\text{luke}') :: \langle \mathbf{e}, \mathbf{t} \rangle(\mathbf{e}) \implies \mathbf{t}$ (“luke is a jedi” has a truth value: true or false)
- $\text{admire}'(\text{luke}') :: \langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle(\mathbf{e}) \implies \langle \mathbf{e}, \mathbf{t} \rangle$ (“[to] admire luke” is a property)

More examples of types

Types of higher-order expressions:

- Predicate modifiers (expensive, small): $\langle\langle\mathbf{e}, \mathbf{t}\rangle, \langle\mathbf{e}, \mathbf{t}\rangle\rangle$
(Function from properties to properties)
- Second-order predicates (state of mind): $\langle\langle\mathbf{e}, \mathbf{t}\rangle, \mathbf{t}\rangle$
(Property of properties)
- Sentence operators (yesterday, unfortunately): $\langle\mathbf{t}, \mathbf{t}\rangle$
(Function from truth values to truth values)
- Degree particles (very, too): $\langle\langle\langle\mathbf{e}, \mathbf{t}\rangle, \langle\mathbf{e}, \mathbf{t}\rangle\rangle, \langle\langle\mathbf{e}, \mathbf{t}\rangle, \langle\mathbf{e}, \mathbf{t}\rangle\rangle\rangle$
“complex function”

Tip: If σ , τ are basic types, $\langle\sigma, \tau\rangle$ can be abbreviated as $\sigma\tau$. Thus, the type of predicate modifiers and second-order predicates can be more conveniently written as $\langle\mathbf{et}, \mathbf{et}\rangle$ and $\langle\mathbf{et}, \mathbf{t}\rangle$, respectively.

Type Theory — Vocabulary

Non-logical constants:

- For every type τ a (possibly empty) set of non-logical constants CON_τ (pairwise disjoint)

Variables:

- For every type τ an infinite set of variables VAR_τ (pairwise disjoint)

Logical symbols: $\forall, \exists, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, =$

Brackets: $(,)$

Type Theory — Syntax

For every type τ , the set of well-formed expressions WE_τ is defined as follows:

- (i) $CON_\tau \subseteq WE_\tau$ and $VAR_\tau \subseteq WE_\tau$;
- (ii) If $\alpha \in WE_{\langle\sigma, \tau\rangle}$, and $\beta \in WE_\sigma$, then $\alpha(\beta) \in WE_\tau$; (function application)
- (iii) If A, B are in WE_t , then $\neg A$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$ are in WE_t ;
- (iv) If A is in WE_t and x is a variable of arbitrary type, then $\forall xA$ and $\exists xA$ are in WE_t ;
- (v) If α, β are well-formed expressions of the same type, then $\alpha = \beta \in WE_t$;
- (vi) Nothing else is a well-formed expression.

This prevents us from running into Russell's paradox!

Type inferencing

Types can be derived for all expressions that constitute the logical form of a sentence, as defined by its syntactic structure.

“Luke is a talented jedi”

$$\text{talented}' :: \langle \langle e, t \rangle, \langle e, t \rangle \rangle \qquad \text{jedi}' :: \langle e, t \rangle$$

$$\text{luke}' :: e \qquad \text{talented}'(\text{jedi}') :: \langle e, t \rangle$$

$$\text{talented}'(\text{jedi}')(\text{luke}') :: t$$

Note: we here ignore the semantic contribution of “is” and “a” (see Winter, pg 61)

Type inferencing: examples

Recommended strategy: Start by describing the logical form of the sentences (how are expressions combined logically, based on the given syntactic bracketing), then derive types from there (see previous slide).

1. Yoda_e [is faster than Palpatine_e].
2. Yoda_e [is much [faster than]] Palpatine_e .
3. [[Han Solo_e fights] [because [[the Dark Side_e is rising]]].
4. Obi-Wan_e [[told [Qui-Gon Jinn_e] he will take [the Jedi-exam]] $_e$].

Higher-order predicates

Higher-order quantification:

- *Leia has the same hair colour as Padmé*

$$\exists C (\text{hair_colour}(C) \wedge C(l') \wedge C(p'))$$

$\langle\langle e, t \rangle, t \rangle$ $\langle e, t \rangle$ e

Higher-order equality:

- For $p, q \in \text{CON}_t$, “ $p=q$ ” expresses material equivalence: “ $p \leftrightarrow q$ ”.
- For $F, G \in \text{CON}_{\langle e, t \rangle}$, “ $F=G$ ” expresses co-extensionality: “ $\forall x(Fx \leftrightarrow Gx)$ ”.
- For any formula ϕ of type t , $\phi=(x=x)$ is a representation of “ ϕ is true”.

Type Theory — Semantics [1]

Let \mathbf{U} be a non-empty set of entities.

The domain of possible denotations \mathbf{D}_τ for every type τ is given by:

- $D_e = U$
- $D_t = \{0, 1\}$
- $D_{\langle\sigma, \tau\rangle}$ is the set of all functions from D_σ to D_τ

For any type τ , expressions of type τ denote elements of the domain \mathbf{D}_τ

Characteristic functions

Many natural language expressions have a type $\langle \sigma, \mathbf{t} \rangle$

Expressions with type $\langle \sigma, \mathbf{t} \rangle$ are functions mapping elements of type σ to truth values: $\{0, 1\}$

Such functions with a range of $\{0, 1\}$ are called *characteristic functions*, because they uniquely specify a subset of their domain \mathbf{D}_σ

The characteristic function of set M in a domain U is the function $F_M: U \rightarrow \{0, 1\}$ such that for all $a \in U$, $F_M(a) = 1$ iff $a \in M$.

NB: For first-order predicates, the FOL representation (using sets) and the type-theoretic representation (using characteristic functions) are equivalent.

Interpretation with characteristic functions: example

For $M = \langle U, V \rangle$, let U consist of five entities. For selected types, we have the following sets of possible denotations:

- $D_t = \{0, 1\}$

- $D_e = U = \{e_1, e_2, e_3, e_4, e_5\}$

- $D_{\langle e, t \rangle} = \left\{ \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 0 \end{bmatrix}, \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 0 \end{bmatrix}, \dots, \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix}, \dots, \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix}, \dots \right\}$

Alternative set notation: $D_{\langle e, t \rangle} = \{\{\}, \{e_1\}, \dots, \{e_2, e_3, e_5\}, \dots, \{e_1, e_2, e_5\}, \dots\}$

Type Theory — Semantics [2]

A model structure for a type theoretic language is a tuple $\mathbf{M} = \langle \mathbf{U}, \mathbf{V} \rangle$ such that:

- \mathbf{U} is a non-empty domain of individuals
- \mathbf{V} is an interpretation function, which assigns to every $\mathbf{a} \in \mathbf{CON}_{\tau}$ an element of \mathbf{D}_{τ} (where τ is an arbitrary type)

The variable assignment function g assigns to every typed variable $\mathbf{v} \in \mathbf{VAR}_{\tau}$ an element of \mathbf{D}_{τ}

Type Theory — Interpretation

Given a model structure $M = \langle U, V \rangle$ and a variable assignment g :

- $\llbracket \alpha \rrbracket^{M,g} = V(\alpha)$ if α is a constant
 $= g(\alpha)$ if α is a variable
- $\llbracket \alpha(\beta) \rrbracket^{M,g} = \llbracket \alpha \rrbracket^{M,g}(\llbracket \beta \rrbracket^{M,g})$
- $\llbracket \alpha = \beta \rrbracket^{M,g} = 1$ iff $\llbracket \alpha \rrbracket^{M,g} = \llbracket \beta \rrbracket^{M,g}$
- $\llbracket \neg \phi \rrbracket^{M,g} = 1$ iff $\llbracket \phi \rrbracket^{M,g} = 0$
- $\llbracket \phi \wedge \psi \rrbracket^{M,g} = 1$ iff $\llbracket \phi \rrbracket^{M,g} = 1$ and $\llbracket \psi \rrbracket^{M,g} = 1$
- $\llbracket \phi \vee \psi \rrbracket^{M,g} = 1$ iff $\llbracket \phi \rrbracket^{M,g} = 1$ or $\llbracket \psi \rrbracket^{M,g} = 1$
- ...

For any variable v of type σ :

- $\llbracket \exists v \phi \rrbracket^{M,g} = 1$ iff there is a $d \in D_\sigma$ such that $\llbracket \phi \rrbracket^{M,g[v/d]} = 1$
- $\llbracket \forall v \phi \rrbracket^{M,g} = 1$ iff for all $d \in D_\sigma$: $\llbracket \phi \rrbracket^{M,g[v/d]} = 1$

Interpretation: Example

Luke is a talented jedi

$\text{jedi}' :: \langle e, t \rangle$

$\text{talented}' :: \langle \langle e, t \rangle, \langle e, t \rangle \rangle$

$\text{luke}' :: e$

$\text{talented}'(\text{jedi}') :: \langle e, t \rangle$

$\text{talented}'(\text{jedi}')(\text{luke}') :: t$

$\llbracket \text{talented}'(\text{jedi}')(\text{luke}') \rrbracket^{M,g}$

$= \llbracket \text{talented}'(\text{jedi}') \rrbracket^{M,g} (\llbracket \text{luke}' \rrbracket^{M,g})$

$= \llbracket \text{talented}' \rrbracket^{M,g} (\llbracket \text{jedi}' \rrbracket^{M,g}) (\llbracket \text{luke}' \rrbracket^{M,g})$

$= V_M(\text{talented}')(V_M(\text{jedi}'))(V_M(\text{luke}'))$

Interpretation: Example (cont.)

$$\llbracket \text{Luke is a talented Jedi} \rrbracket^{M,g} = V_M(\text{talented}')(V_M(\text{jedi}'))(V_M(\text{luke}'))$$

$$V_M(\text{luke}') = e_1 \ (\in \mathbf{D}_e)$$

$$V_M(\text{jedi}') = \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix} \ (\in \mathbf{D}_{\langle e,t \rangle}) \iff \{e_1, e_3, e_5\}$$

$$V_M(\text{talented}') = \begin{bmatrix} \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix} \\ \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 1 \\ e_5 \rightarrow 1 \end{bmatrix} \\ \dots \end{bmatrix} \rightarrow \begin{bmatrix} \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix} \\ \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 0 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix} \\ \dots \end{bmatrix} \ (\in \mathbf{D}_{\langle\langle e,t \rangle, \langle e,t \rangle\rangle}) \iff \begin{bmatrix} \{e_1, e_3, e_5\} \rightarrow \{e_1, e_5\} \\ \{e_3, e_4, e_5\} \rightarrow \{e_5\} \\ \dots \end{bmatrix}$$

$$V_M(\text{talented}')(V_M(\text{jedi}'))$$

$$V_M(\text{talented}')(V_M(\text{jedi}'))(V_M(\text{luke}'))$$

Defining the right model

Consider the following Model M:

$$D_e = U_M = \{e_1, e_2, e_3, e_4, e_5\}$$

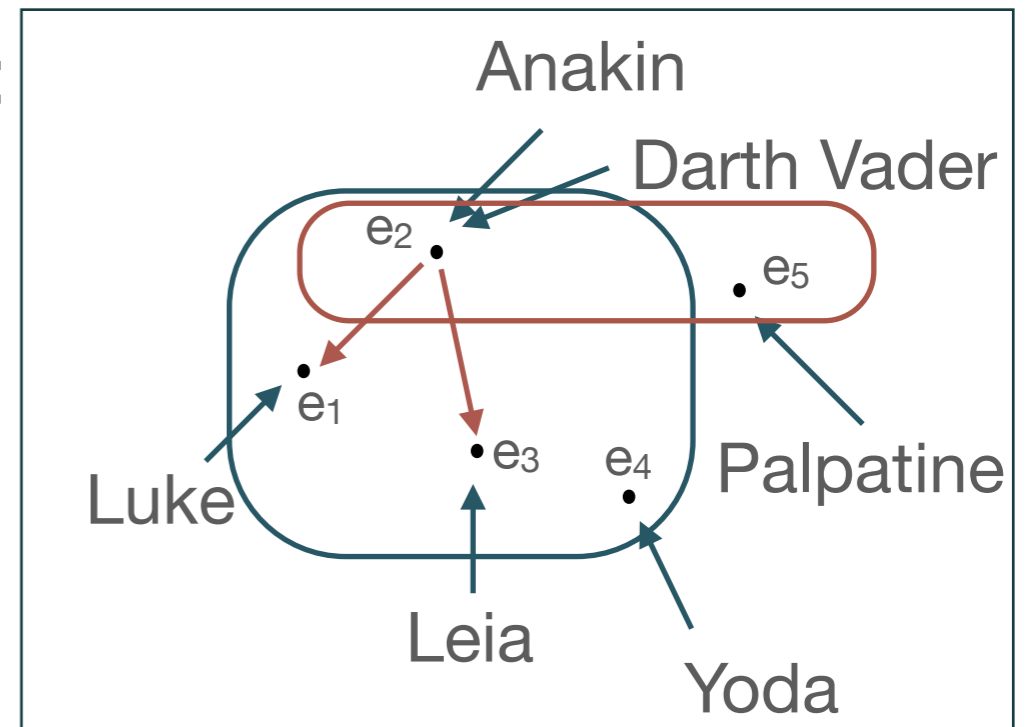
$$V_M(\text{anakin}'_e) = V_M(\text{darth_vader}'_e) = e_2$$

$$V_M(\text{jedi}'_{\langle e,t \rangle}) = \begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 1 \\ e_5 \rightarrow 0 \end{bmatrix}$$

$$V_M(\text{dark_sider}'_{\langle e,t \rangle}) = \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix}$$

$$V_M(\text{powerful}'_{\langle \langle e,t \rangle \langle e,t \rangle \rangle}) = \left[\begin{bmatrix} e_1 \rightarrow 1 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 1 \\ e_4 \rightarrow 1 \\ e_5 \rightarrow 0 \end{bmatrix} \rightarrow \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 1 \\ e_5 \rightarrow 0 \end{bmatrix}, \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix} \rightarrow \begin{bmatrix} e_1 \rightarrow 0 \\ e_2 \rightarrow 1 \\ e_3 \rightarrow 0 \\ e_4 \rightarrow 0 \\ e_5 \rightarrow 1 \end{bmatrix}, \dots \right]$$

M:



Note that here “powerful” is truth-preserving: Powerful $X_{\langle e,t \rangle} \models X_{\langle e,t \rangle}$

Meaning postulates

Some valid inferences in natural language:

- Bill is a poor piano player \models Bill is a piano player
- Bill is a blond piano player \models Bill is blond
- Bill is a former professor \models Bill isn't a professor

→ These entailments do not hold in type theory by definition. Why?

Meaning postulates: Restrictions on models that constrain the possible meanings of certain words

Adjective classes

Restrictive or Subsective adjectives (“poor”)

- $\llbracket \text{poor } N \rrbracket \subseteq \llbracket N \rrbracket$
- Meaning postulate: $\forall G \forall x (\text{poor}(G)(x) \rightarrow G(x))$

Intersective adjectives (“blond”)

- $\llbracket \text{blond } N \rrbracket = \llbracket \text{blond} \rrbracket \cap \llbracket N \rrbracket$
- Meaning postulate: $\forall G \forall x (\text{blond}(G)(x) \rightarrow (\text{blond}^*(x) \wedge G(x)))$
- NB: $\text{blond} \in \text{WE}\langle\langle e, t \rangle, \langle e, t \rangle\rangle \neq \text{blond}^* \in \text{WE}\langle e, t \rangle$

Privative adjectives (“former”)

- $\llbracket \text{former } N \rrbracket \cap \llbracket N \rrbracket = \emptyset$
- Meaning postulate: $\forall G \forall x (\text{former}(G)(x) \rightarrow \neg G(x))$

Reading material

- Winter: Elements of Formal Semantics (Chapter 3, Part I & II)
<http://www.phil.uu.nl/~yoad/efs/main.html>