Semantic Theory
Week 3 – Type Theory

Noortje Venhuizen

Universität des Saarlandes

Summer 2017

First-order logic

First-order logic talks about:

- Individual objects
- Properties of and relations between individual objects
- Generalization across individual objects (quantification)

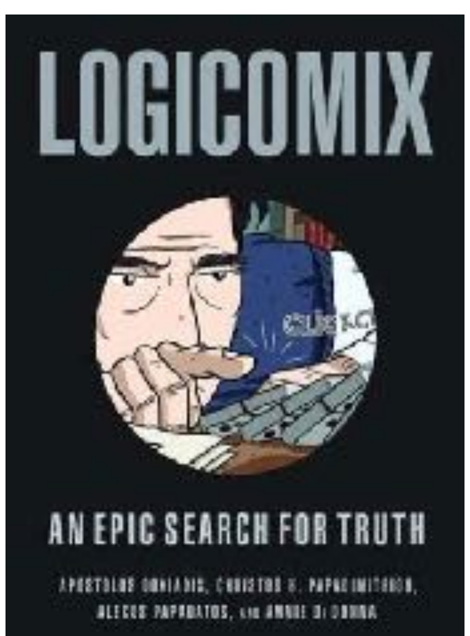
Limitations of first-order logic

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

Jumbo is a <u>small</u> elephant.	(Predicate modifiers)
Blond is a <u>hair color.</u>	(Second-order predicates)
<u>Yesterday</u>, it rained.	(Non-logical sentence operators)
Bill and John have <u>the same</u> hair color.	(Higher-order quantification)

What logical system can we use to capture this diversity?

Bertrand Russell



Q: Does the barber shave himself?

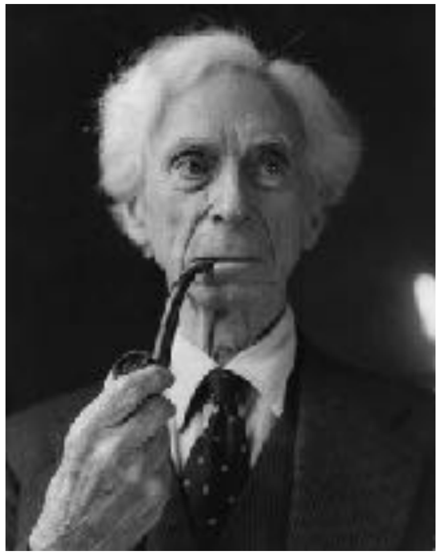
Russell's paradox

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

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

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

Q: does S belong to itself?



… we need a more restricted way of talking about *properties* and *relations between properties*!

Type Theory

Basic types:

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

Complex types:

- If **σ**, **τ** are types, then ⟨**σ, τ**⟩ is a type
  (representing a functor expression that takes a **σ** type expression as argument and returns a type **τ** expression; sometimes written as: **(σ→ τ)**).

Types & Function Application

Types of first-order expressions:

- Individual constants (Luke, Saarbrücken) : **e**
- One-place predicates (sleep, walk): $\langle \mathbf{e}, \mathbf{t} \rangle$
- Two-place predicates (read, admire): $\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle\rangle$
- Three-place predicates (give, introduce): $\langle \mathbf{e}, \langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle\rangle\rangle$

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

- sleep'(john') :: $\langle \mathbf{e}, \mathbf{t} \rangle(\mathbf{e}) \Longrightarrow \mathbf{t}$
- admire'(john') :: $\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle\rangle(\mathbf{e}) \Longrightarrow \langle \mathbf{e}, \mathbf{t} \rangle$

More examples of types

Types of higher-order expressions:

- Predicate modifiers (expensive, poor): ⟨⟨**e, t**⟩**,** ⟨**e, t**⟩⟩
- Second-order predicates (hair colour): ⟨⟨**e, t**⟩**, t**⟩
- Sentence operators (yesterday, possibly, unfortunately): ⟨**t, t**⟩
- Degree particles (very, too): ⟨⟨⟨**e, t**⟩**,** ⟨**e, t**⟩⟩**,** ⟨⟨**e, t**⟩**,** ⟨**e, t**⟩⟩⟩

Tip: If **σ**, **τ** are basic types, ⟨**σ, τ**⟩ can be abbreviated as **στ**. Thus, the type of predicate modifiers and second-order predicates can be more conveniently written as ⟨**et, et**⟩ and ⟨**et, t**⟩, 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.

Function application

(ii) If $\alpha \in WE_{\langle\sigma,\tau\rangle}$*, and* $\beta \in WE_{\sigma}$*, then* $\alpha(\beta) \in WE_{\tau}$

"John is a talented piano player"

$$\cfrac{\text{john} :: e \qquad \cfrac{\text{piano\_player} :: \langle e, t\rangle \qquad \text{talented} :: \langle\langle e, t\rangle, \langle e, t\rangle\rangle}{\text{talented(piano\_player)} :: \langle e, t\rangle}}{\text{talented(piano\_player)(john)} :: t}$$

Higher-order predicates

Higher-order quantification:

- *Bill has the same hair colour as John*

  $\exists C\ (\text{hair\_colour}(C) \wedge C(\text{bill}) \wedge C(\text{john}))$

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

Higher-order equality:

- For p, q ∈ CON_t, “p=q” expresses material equivalence: “p ↔ q”.
- For F, G ∈ $CON_{\langle e,\ t\rangle}$, “F=G” expresses co-extensionality: “∀x(Fx↔Gx)”
- For any formula ϕ of type *t*, ϕ=(x=x) is a representation of “ϕ is true”.

Type inferencing: examples

1. Yoda$_e$ is faster than Palpatine$_e$.
2. Yoda$_e$ is much faster than Palpatine$_e$.
3. Anakin$_e$ believes he will be a Jedi.
4. Obi-Wan$_e$ told [Qui-Gon Jinn]$_e$ he will take [the Jedi-exam]$_e$.
5. Obi-Wan$_e$ expects to pass.
6. Yoda$_e$ encouraged Obi-Wan$_e$ to take the exam.
7. [Han Solo]$_e$ fights because [the Dark Side]$_e$ is rising.
8. Wookiee$_{\langle e,t\rangle}$ is a hairier species than Ewok$_{\langle e,t\rangle}$.

Type Theory — Semantics [1]

Let **U** be a non-empty set of entities.

The domain of possible denotations $\mathbf{D_\tau}$ for every type $\boldsymbol{\tau}$ 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

Expressions of type $\boldsymbol{\sigma}$ *denote elements of* $\mathbf{D_\sigma}$

Characteristic functions

Many natural language expressions have a type ⟨**σ, t**⟩

Expressions with type ⟨**σ, t**⟩ 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 $\mathbf{D_{\sigma}}$

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 the persons John, Bill, Mary, Paul, and Sally. For selected types, we have the following sets of possible denotations:

- $D_t = \{0,1\}$
- $D_e = U = \{j, b, m, p, s\}$
- $D_{<e,t>} = \left\{ \begin{bmatrix} j \rightarrow 1 \\ b \rightarrow 0 \\ m \rightarrow 1 \\ p \rightarrow 0 \\ s \rightarrow 1 \end{bmatrix}, \begin{bmatrix} j \rightarrow 1 \\ b \rightarrow 1 \\ m \rightarrow 0 \\ p \rightarrow 1 \\ s \rightarrow 1 \end{bmatrix}, \begin{bmatrix} j \rightarrow 0 \\ b \rightarrow 1 \\ m \rightarrow 1 \\ p \rightarrow 0 \\ s \rightarrow 0 \end{bmatrix}, \ldots \right\}$

Alternative set notation: $D_{<e,t>} = \{\{j,m,s\},\{j,b,p,s\},\{b,m\},\ldots\}$

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:

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

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

Type Theory — Interpretation

Given a model structure M = ⟨U, V⟩ and a variable assignment g:

$[\![\alpha]\!]^{M,g}$ = V(α) if α is a constant
= g(α) if α is a variable

$[\![\alpha(\beta)]\!]^{M,g} = [\![\alpha]\!]^{M,g}([\![\beta]\!]^{M,g})$

$[\![\alpha = \beta]\!]^{M,g}$ = 1 iff $[\![\alpha]\!]^{M,g} = [\![\beta]\!]^{M,g}$

$[\![\neg\phi]\!]^{M,g}$ = 1 iff $[\![\phi]\!]^{M,g} = 0$

$[\![\phi \wedge \psi]\!]^{M,g}$ = 1 iff $[\![\phi]\!]^{M,g} = 1$ and $[\![\psi]\!]^{M,g} = 1$

$[\![\phi \vee \psi]\!]^{M,g}$ = 1 iff $[\![\phi]\!]^{M,g} = 1$ or $[\![\psi]\!]^{M,g} = 1$

…

For any variable v of type σ:

$[\![\exists v\phi]\!]^{M,g}$ = 1 iff there is a $d \in D_\sigma$ such that $[\![\phi]\!]^{M,g[v/d]} = 1$

$[\![\forall v\phi]\!]^{M,g}$ = 1 iff for all $d \in D_\sigma$: $[\![\phi]\!]^{M,g[v/d]} = 1$

Interpretation: Example

John is a talented piano player

$$\frac{\dfrac{\text{piano\_player} :: \langle e, t\rangle \qquad \text{talented} :: \langle\langle e, t\rangle, \langle e, t\rangle\rangle}{\text{john} :: e \qquad \text{talented(piano\_player)} :: \langle e, t\rangle}}{\text{talented(piano\_player)(john)} :: t}$$

$[\![\text{talented(piano\_player)(john)}]\!]^{M,g}$

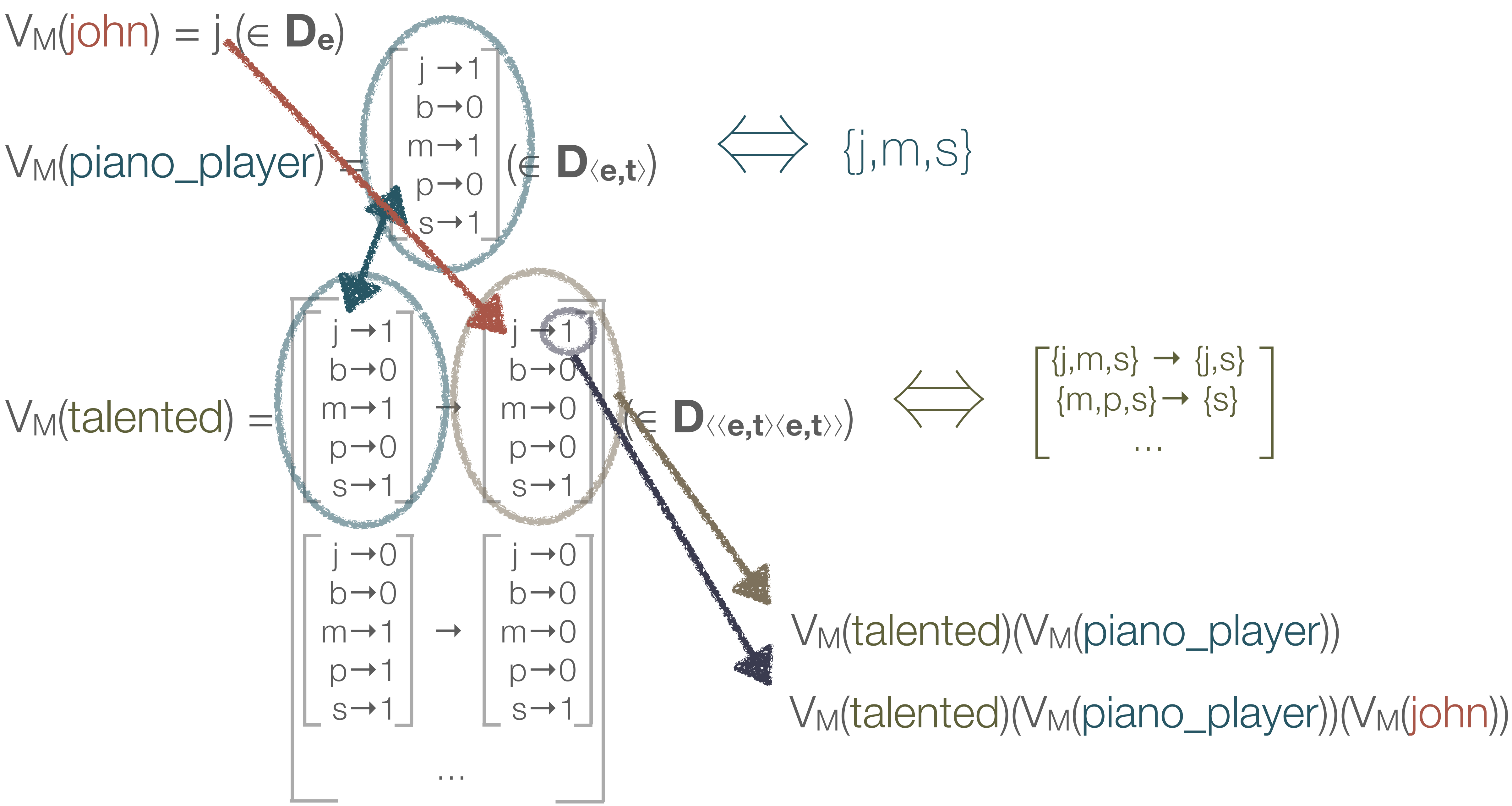
$= [\![\text{talented(piano\_player)}]\!]^{M,g} ([\![\text{john}]\!]^{M,g})$

$= [\![\text{talented}]\!]^{M,g}([\![\text{piano\_player}]\!]^{M,g}) ([\![\text{john}]\!]^{M,g})$

$= V_M(\text{talented})(V_M(\text{piano\_player}))(V_M(\text{john}))$

Interpretation: Example (cont.)

⟦*John is a talented piano player*⟧^{M,g} = V_M(talented)(V_M(piano_player))(V_M(john))



Defining the right model

Consider the following Model M:

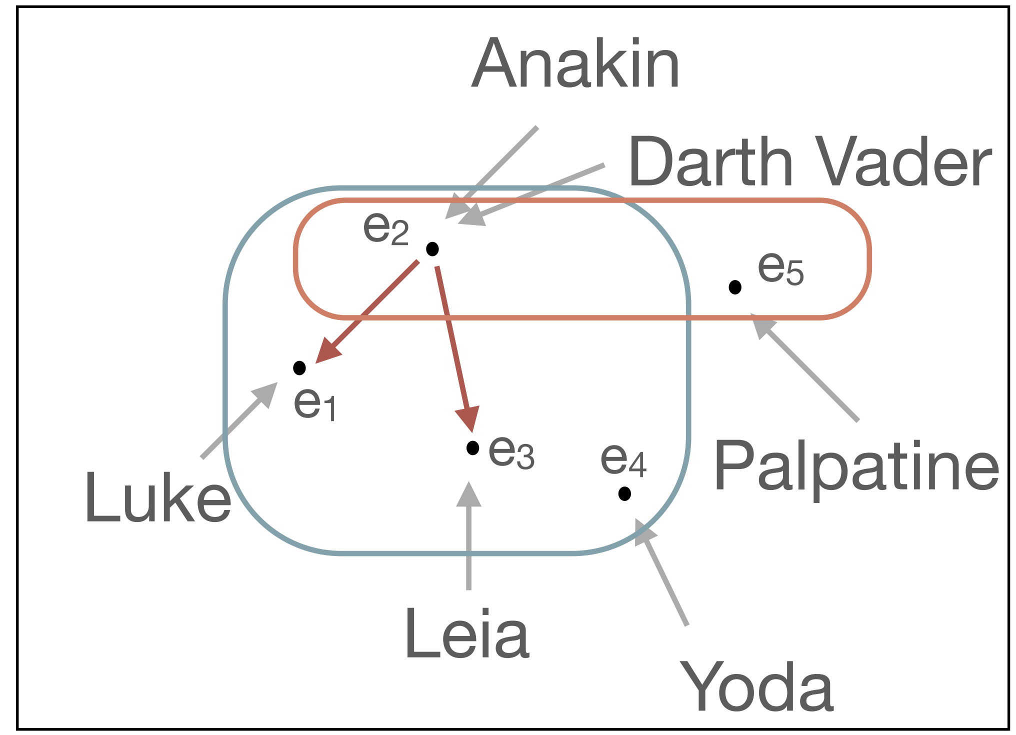
$D_e = U_M = \{e_1, e_2, e_3, e_4, e_5\}$

$V_M(\text{Anakin}_e) = V_M(\text{Darth Vader}_e) = e_2$

$$V_M(\text{Jedi}_{\langle e,t\rangle}) = \begin{bmatrix} e_1 \to 1 \\ e_2 \to 1 \\ e_3 \to 1 \\ e_4 \to 1 \\ e_5 \to 0 \end{bmatrix} \quad V_M(\text{Dark\_Sider}_{\langle e,t\rangle}) = \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \\ e_3 \to 0 \\ e_4 \to 0 \\ e_5 \to 1 \end{bmatrix}$$

$$V_M(\text{Powerful}_{\langle\langle e,t\rangle\langle e,t\rangle\rangle}) = \begin{bmatrix} \begin{bmatrix} e_1 \to 1 \\ e_2 \to 1 \\ e_3 \to 1 \\ e_4 \to 1 \\ e_5 \to 0 \end{bmatrix} \to \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \\ e_3 \to 0 \\ e_4 \to 1 \\ e_5 \to 0 \end{bmatrix} \\ \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \\ e_3 \to 0 \\ e_4 \to 0 \\ e_5 \to 1 \end{bmatrix} \to \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \\ e_3 \to 0 \\ e_4 \to 0 \\ e_5 \to 1 \end{bmatrix} \\ \ldots \end{bmatrix}$$

M:



--> Powerful $X_{\langle e,t\rangle} \vDash X_{\langle e,t\rangle}$

Adjective classes & Meaning postulates

Some valid inferences in natural language:

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

These entailments do not hold in type theory. Why?

Meaning postulates: restrictions on models which constrain the possible meaning of certain words

Adjective classes & Meaning postulates (cont.)

Restrictive or Subsective adjectives ("poor")

- ⟦ poor N ⟧ ⊆ ⟦ N ⟧
- Meaning postulate: $\forall G \forall x(\text{poor}(G)(x) \rightarrow G(x))$

Intersective adjectives ("blond")

- ⟦ blond N ⟧ = ⟦ blond ⟧ ∩ ⟦ N ⟧
- Meaning postlate: $\forall G \forall x(\text{blond}(G)(x) \rightarrow (\text{blond}^*(x) \wedge G(x))$
- NB: blond $\in WE_{\langle\langle e, t\rangle, \langle e, t\rangle\rangle} \neq$ blond* $\in WE_{\langle e, t\rangle}$

Privative adjectives ("former")

- ⟦ former N ⟧ ∩ ⟦ N ⟧ = ∅
- Meaning postlate: $\forall G \forall x(\text{former}(G)(x) \rightarrow \neg G(x))$