# Meaning Postulates

(1) *John is a bachelor*

(2) *John is married to Mary*

- Predicate logic is insensitive to the meaning of words.
- **Meaning postulates** can be thought of as additional axioms or premises used to exclude unwanted models.
- Examples:
  - $\forall x(\text{bachelor}(x) \rightarrow \neg\exists y\ \text{married-to}(x, y))$
  - $\forall x \forall y(\text{sell}(x, y) \rightarrow \text{buy}(y, x))$
  - $\forall x(\text{thumb}(x) \rightarrow \exists y(\text{hand}(y) \wedge \text{part-of}(x, y)))$



# True, Valid, Satisfiable

- **A formula φ is true in a model structure M** iff $[\![\varphi]\!]^{M,g} = 1$ for every variable assignment g
- **A formula φ is valid (⊨ φ)** iff φ is true in all model structures
- **A formula φ is satisfiable** iff there is at least one model structure M such that φ is true in M



# Satisfiable? Valid?

(1) $\forall x F(x) \rightarrow \exists x F(x)$

(2) $\exists x \forall y \Phi \rightarrow \forall y \exists x \Phi$

(3) $\exists x(F(x) \wedge \neg F(x))$

(4) $\exists x F(x) \vee \neg F(x)$

## Entailment

- **A set of formulas Γ is (simultaneously) satisfiable** iff there is a model structure M such that every formula in Γ is true in M ("M satisfies Γ," or "M is a model of Γ")
- **Γ is contradictory** if Γ is not satisfiable.
- **Γ entails a formula φ (Γ ⊨ φ)** iff φ is true in every model structure that satisfies Γ



## Example [⇒ Blackboard]

(1) *Not every blond student passed*

(2) *Not every student passed*



## Some logical laws

- **Quantifier negation**
  - $\neg\forall x\varphi \Leftrightarrow \exists x\neg\varphi$
- **Quantifier distribution**
  - $\forall x(\varphi \wedge \Psi) \Leftrightarrow \forall x\varphi \wedge \forall x\Psi$
  - $\exists x(\varphi \vee \Psi) \Leftrightarrow \exists x\varphi \vee \exists x\Psi$
- **Quantifier (in-)dependence**
  - $\forall x\forall y\varphi \Leftrightarrow \forall y\forall x\varphi$
  - $\exists x\exists y\varphi \Leftrightarrow \exists y\exists x\varphi$
  - $\exists x\forall y\varphi \Rightarrow \forall y\exists x\varphi$ (but not vice versa)

## Some logical laws

- **Quantifier movement**
  - $\varphi \rightarrow \forall x \Psi \;\Leftrightarrow\; \forall x(\varphi \rightarrow \Psi)$
  - $\varphi \rightarrow \exists x \Psi \;\Leftrightarrow\; \exists x(\varphi \rightarrow \Psi)$
  - $\forall x \Psi \rightarrow \varphi \;\Leftrightarrow\; \forall x(\Psi \rightarrow \varphi)$
  - $\exists x \Psi \rightarrow \varphi \;\Leftrightarrow\; \exists x(\Psi \rightarrow \varphi)$
- … provided that x does not occur free in φ



## First-order logic

- Formulas of first-order logic can talk about properties of and relations between individuals.
- Constants and variables denote individuals.
- Quantification is restricted to quantification over individuals.



## Limits of first-order logic

- First-order logic is not expressive enough to capture the full range of meaning of natural language:
  - Modification (“good student”, “former professor”)
  - Sentence embedding verbs (“knows that …”)
  - Higer order quantification (“have the same hair color”)
  - …
- First-order logic does not support compositional semantics construction.

# Limits of first-order logic

- **The principle of compositionality (recap):** *The meaning of a complex expression is a function of the meanings of its parts and of the syntactic rules by which they are combined* (cited from Partee &al., 1993)
- **Compositional semantics construction:**
  - compute meaning representations for sub-expressions.
  - combine them to obtain a meaning representation for a complex expression.
- *a man walks* $\mapsto \exists x(\text{man}'(x) \wedge \text{walk}'(x))$
  - *a man* $\mapsto$ **(?)**
  - *walks* $\mapsto$ **(?)**

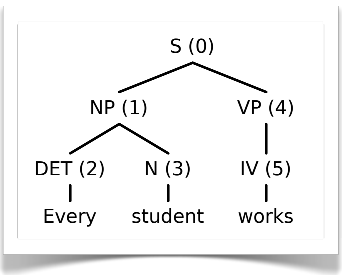


# Type Theory

- *a man walks* $\mapsto \exists x(\text{man}'(x) \wedge \text{walk}'(x))$
  - *a man* $\mapsto$ **(?)**
  - *walks* $\mapsto$ **(?)**
- Type theory generalizes first-order predicate logic
  - flexible syntax
  - higher order predicates and variables
  - lambda operator



# Semantics Construction




(2) $\mapsto \lambda P \lambda Q \forall x(P(x) \rightarrow Q(x))$

(3) $\mapsto$ student'

(1) $\mapsto \lambda P \lambda Q \forall x(P(x) \rightarrow Q(x))(\text{student}')$

$\Rightarrow_\beta \lambda Q \forall x(\text{student}'(x) \rightarrow Q(x))$

(4) = (5) $\mapsto$ work'

(0) $\mapsto \lambda Q \forall x(\text{student}'(x) \rightarrow Q(x))(\text{work}')$

$\Rightarrow_\beta$ $\mathbf{\forall x(\text{student}'(x) \rightarrow \text{work}'(x))}$