EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

**Faculty of Philosophy**
General Linguistics

# Semantics & Pragmatics SoSe 2020

## Lecture 5: Formal Semantics II (Predicate Logic)

**05/05/2020, Christian Bentz**

# Q&A

## Organization

- *When and how do we get to know about pass and fail for the exercise sheets?* – The tutors will send you an e-mail/moodle-message as soon as they have checked the sheets.
- I created a chat on moodle, where you can discuss questions about exercises and lectures. Please do not use it to exchange solutions to the exercises.

# Q&A

## Tutorials

- *What about the full stop?* – Whether or not to include punctuation generally is a pre-processing decision to which there is not necessarily a right or wrong answer. In the exam, I would be clear about which option to choose, or accept different solutions.
- *There is a mistake in Table 1 of the solutions, a should have the frequency 5.* – Yes, that is true.
- *Shouldn't we have whole bits, i.e. only whole numbers?* – In terms of implementation on a computer, yes. But in theory the entropy is defined over rational numbers bigger 0.
- *What is the maximum likelihood (plug-in) estimator?* – In this context, it simply means you plug in the normalized frequency of a unit as its probability, i.e. without any further smoothing.
- *Why $log_2$?* – This is a convention in information theory to get results in bits.

# Q&A

## Lecture 4

- *On slide 42 of Lecture 4 there was an error, the last valuation should be* $V((\neg(p \vee q) \rightarrow \neg\neg\neg q) \leftrightarrow r) = 1$ *instead of 0.*
  – True, I changed it.
- *Are construction trees the same as truth trees?*
  – No. Note that a construction tree is about finding out whether a given string is a valid propositional logic expression, while truth trees are constructed to find out whether a *whole inference* drawn based on propositional logical expressions is valid. In other words, a given propositional logic expression might be syntactically valid, but still lead to an invalid inference. For example, $p \leftrightarrow \neg p$ is perfectly syntactically valid, but since it is a contradiction, it is invalid as a logical inference.

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

# Overview



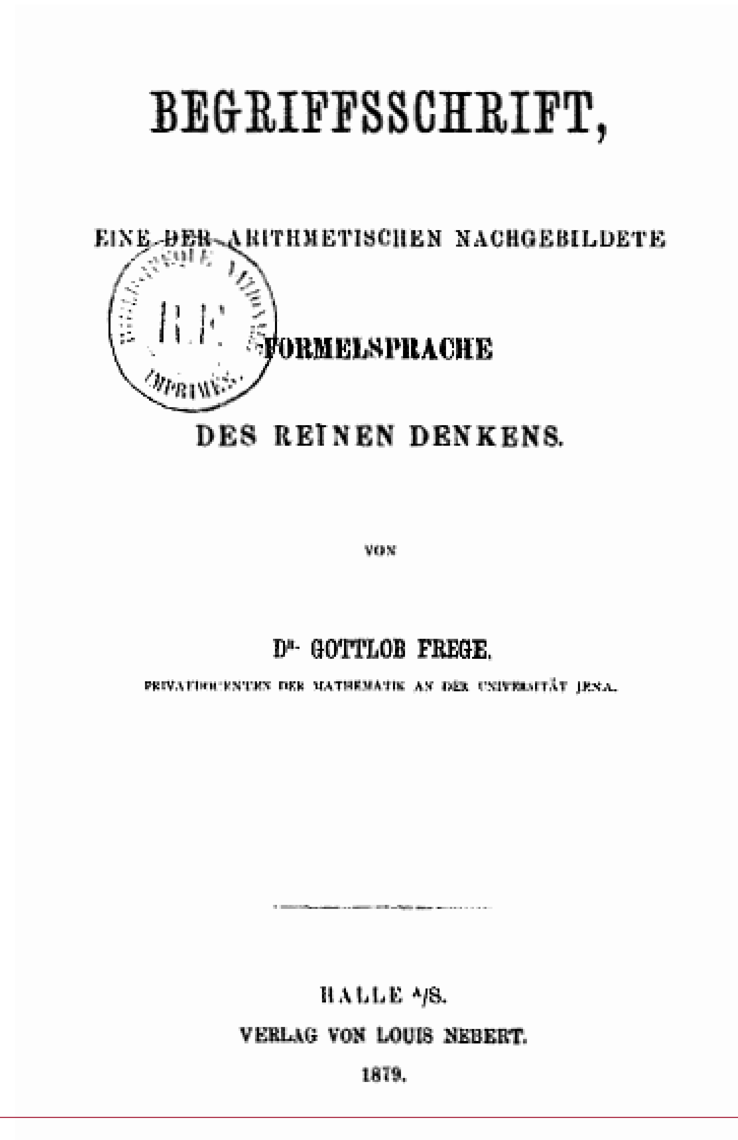
 

# Section 1: Recap of Lecture 4

"[...] fand ich ein Hindernis in der **Unzulänglichkeit der Sprache**, die bei aller entstehenden Schwerfälligkeit des Ausdruckes doch, je verwickelter die Beziehungen wurden, desto weniger die Genauigkeit erreichen liess, welche mein Zweck verlangte. Aus diesem Bedürfnisse ging der Gedanke der vorliegenden **Begriffsschrift** hervor."

Frege (1879). Begriffsschrift: Eine der arithmetischen nachgebildete Formelsprache des reinen Denkens, p. X.

Translation: [...] I found the **inadequacy of language** to be an obstacle; no matter how unwieldy the expressions I was ready to accept, I was less and less able, as the relations became more and more complex, to attain the precision that my purpose required. This **deficiency** led me to the idea of the present **ideography**.



BEGRIFFSSCHRIFT,

EINE DER ARITHMETISCHEN NACHGEBILDETE

FORMELSPRACHE

DES REINEN DENKENS.

VON

D^R. GOTTLOB FREGE,

PRIVATDOCENTEN DER MATHEMATIK AN DER UNIVERSITÄT JENA.

HALLE ᵃ/S.

VERLAG VON LOUIS NEBERT.

1879.



# Propositional Variables

"[...] as logical variables there are symbols which stand for statements (that is 'propositions'). These symbols are called **propositional letters**, or **propositional variables**. In general we shall designate them by the letters p, q, and r, where necessary with subscripts as in $p_1$, $q_2$, $r_3$, etc."

Gamut, L.T.F (1991). Volume 1, p. 29.

# Propositional Operators

We will here use the following operators (aka connectives):

| Operator | Alternative Symbols | Name | English Translation |
|---|---|---|---|
| $\neg$ | $\sim$, ! | negation | *not* |
| $\wedge$ | ., & | conjunction | *and* |
| $\vee$ | +, \|\| | disjunction (inclusive *or*) | *or* |
| XOR | EOR, EXOR, $\oplus$, $\veebar$ | exclusive *or* | *either ... or* |
| $\rightarrow$ | $\Rightarrow$, $\supset$ | material implication[1] | *if ..., then* |
| $\leftrightarrow$ | $\Leftrightarrow$, $\equiv$ | material equivalence[2] | *if, and only if ..., then* |

**Note:** We will here assume that the English translations and the operators themselves are indeed equivalent in their meanings. However, in language usage, this might not actually be the case.

[1] aka *conditional*.
[2] aka *biconditional*.



# Propositional Formulas

"The propositional letters and the **composite expressions** which are formed from them by means of connectives are grouped together as *sentences* or **formulas**. We designate these by means of the letters $\phi$ and $\psi$, etc. For these **metavariables**, unlike the variables p, q, and r, there is no convention that different letters must designate different formulas."

Gamut, L.T.F (1991). Volume 1, p. 29.



Examples:

$\phi \equiv \text{p}, \text{q}, \text{r}$, etc.

$\phi \equiv \neg\text{p}, \neg\text{q}, \neg\text{r}$, etc.

$\phi \equiv \text{p} \wedge \text{q}, \text{p} \vee \text{q}$, etc.

$\phi \equiv \neg(\neg\text{p}_1 \vee \text{q}_5) \rightarrow \text{q}$, etc.

## The Vocabulary

We can now define a **language $L$ for propositional logic**. The “vocabulary” $A$ of $L$ consits of the propositional letters (e.g. p, q, r, etc.), the operators (e.g. $\neg$, $\wedge$, $\vee$, $\rightarrow$, etc.), as well as the round brackets ‘(’ and ‘)’. The latter are important to group certain letters and operators together. We thus have:

$$A = \{p, q, r, ..., \neg, \wedge, \vee, \rightarrow, ..., (, )\} \tag{1}$$

# The Syntax: Recursive Definition

Reminiscent of formal grammars of natural languages (see last years lecture on Phrase Structure Grammar), we now also need to define **syntactic rules** which allow for the symbols of the vocabulary to be combined yielding **well-formed expressions**. These rules are:

(i) Propositional letters in the vocabulary of *L* are formulas in *L*.

(ii) If $\psi$ is a formula in *L*, then $\neg\psi$ is too.

(iii) If $\phi$ and $\psi$ are formulas in *L*, then $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, and $(\phi \leftrightarrow \psi)$ are too.[3]

(iv) Only that which can be generated by the clauses (i)-(iii) in a finite number of steps is a formula in *L*.

Gamut, L.T.F (1991). Volume 1, p. 35.



[3] We could also add the *exclusive or* here as a connective.

# Example

- $(\neg(p \vee q) \rightarrow \neg\neg\neg q) \leftrightarrow r$ (iii, $\leftrightarrow$)
  - $(\neg(p \vee q) \rightarrow \neg\neg\neg q)$ (iii, $\rightarrow$)
    - $\neg(p \vee q)$ (ii)
      - $p \vee q$ (iii, $\vee$)
        - p (i)
        - q (i)
    - $\neg\neg\neg q$ (ii)
      - $\neg\neg q$ (ii)
        - $\neg q$ (ii)
          - $q$ (i)
  - r (i)

**Note**: The level of embedding is 3 here. The *biconditional* ($\leftrightarrow$) constitutes the highest level of embedding, the *conditional* ($\rightarrow$) the middle level, the *or-statement* ($\vee$) the lowest level. Importantly, on the right of each formula in the tree, we note in parentheses which clause licenses the formula. In the case of operator application, we also give the operator for completeness, e.g. (iii, $\leftrightarrow$).

# The Semantics of Propositional Logic

"The valuations we have spoken of [i.e. truth valuations of formulas] can now, in the terms just introduced [i.e. functions], be described as (unary)[4] **functions mapping formulas onto truth values**. But not every function with formulas as its domain and truth values as its range will do. A valuation must agree with the **interpretations of the connectives** which are given in their truth tables."

Gamut, L.T.F (1991). Volume 1, p. 35.



[4]An *unary* function is a function with a single argument, e.g. f(x). A *binary* functions could be f(x,y), a *ternary* function f(x,y,z), etc.

# Valuation Exercise



Assume the formula for which we created a construction tree above:

$$\phi \equiv (\neg(\mathrm{p} \vee \mathrm{q}) \rightarrow \neg\neg\neg\mathrm{q}) \leftrightarrow \mathrm{r}. \tag{2}$$

What is the value assigned by $V(\phi)$ given $V(\mathrm{p}) = 1$, $V(\mathrm{q}) = 0$, and $V(\mathrm{r}) = 1$?

**Solution**

To answer this question, the construction tree comes in handy, namely, we might want to start with valuation at the lowest level of embedding and then work our way up:

- $V(\mathrm{p} \vee \mathrm{q}) = 1$
- $V(\neg(\mathrm{p} \vee \mathrm{q})) = 0$
- $V(\neg\neg\neg\mathrm{q}) = 1$
- $V(\neg(\mathrm{p} \vee \mathrm{q}) \rightarrow \neg\neg\neg\mathrm{q}) = 1$
- $V((\neg(\mathrm{p} \vee \mathrm{q}) \rightarrow \neg\neg\neg\mathrm{q}) \leftrightarrow r) = 1$

# Beyond Propositional Logic

"The propositional logic outlined in this section is an important part of the logical metalanguage for semantic analysis, but it is not sufficient on its own because it is concerned only with **truth values** [of whole sentences]. We need a way to go beyond p and q, to represent the actual meanings of **the basic propositions** we are dealing with."

Kroeger (2019). Analyzing meaning, p. 66.



Example Sentences (Set 1):

p: John is hungry.
q: John is smart.
r: John is my brother.

Example Sentences (Set 2):

p: John snores.
q: Mary sees John.
r: Mary gives George a cake.

Note: Propositional logic assigns variables (p, q, r) to whole declarative sentences, and hence is "blind" to the fact that the first set of sentences shares both the same subject, and the copula construction, whereas the second set of sentences uses predicates of different valencies and different subjects and objects.

# Beyond Propositional Logic

A second major limitation of propositional logic is that it cannot take into account **quantifications**, and hence cannot decide on the truth values of the classical syllogisms below.

(1) Premise 1: ***All*** *men are mortal.*
Premise 2: *Socrates is a man.*

---

Conclusion: *Therefore, Socrates is mortal.*

(2) Premise 1: *Arthur is a lawyer.*
Premise 2: *Arthur is honest.*

---

Conclusion: *Therefore,* ***some (= at least one)*** *lawyer is honest.*

# Section 2: Predicate Logic (Basic Definitions)

# The Vocabulary

Similar as for propositional logic, we can define a **language** *L* **for predicate logic**. In this case, the "vocabulary" of *L* consits of

- a (potentially infinite) supply of **constant symbols** (e.g. a, b, c, etc.),
- a (potentially infinite) supply of **variable symbols** representing the constants (e.g. x, y, z, etc.),
- a (potentially infinite) supply of **predicate symbols** (e.g. A, B, C, etc.),
- the **connectives** (e.g. $\neg$, $\wedge$, $\vee$, $\rightarrow$, etc.),
- the **quantifiers** $\forall$ and $\exists$,
- as well as the round brackets '(' and ')'.
- (The equal sign '='.)

# Quantifiers

"Standard predicate logic makes use of two quantifier symbols: the **Universal Quantifier** ∀, and the **Existential Quantifier** ∃. As the mathematical examples [below] illustrate, these quantifier symbols **must introduce a variable**, and this variable is said to be bound by the quantifier."

Kroeger (2019) Analyzing meaning, p. 69.

| Examples: | Quantifier notation: |
|---|---|
| *For all x it is the case that x plus x equals x times two.* | $\forall x(x+x = 2x)$ |
| *There is some y for which y plus four equals y divided by three.* | $\exists y(y+4 = y/3)$ |

Note: The **parentheses** are used here to delimit the expression that the quantifier scopes over. This follows the notation by Gamut (1991), while Kroeger (2019) would use *square brackets* here.

# Predicates and Functions

The following types of symbols (sometimes referred to as "non-logical") are relevant for our analyses:

- **Predicate symbols**: these are typically given as upper case letters, and reflect relations between $n$ elements, where $n \geq 0$, and $n \in \mathbb{N}$ (i.e. natural numbers).[5]
- **Function symbols**: these are typically given with lower case letters ($f$, $g$, etc.), and take $n$ variables as their arguments (similar to predicates), e.g. $f(x)$, $f(x, y)$, etc. However, Gamut (1991) use upper case letters here, remember the valuation function $V(\phi)$ from the lecture on propositional logic.



[5] Zimmermann & Sternefeld (2013), p. 245 denote the set of all n-place predicates of a so-called *predicate logic lexicon* or *language L* as $PRED_{n,L}$.

# Predicates (Notational Confusion)

**Predicate symbols**: these are typically given as upper case letters, and reflect relations between $n$ elements, where $n \geq 0$, and $n \in \mathbb{N}$ (i.e. natural numbers). These are also called **n-ary** or **n-place predicate symbols**.

| Examples: | Kroeger: | Z & S: | **Gamut**: |
|---|---|---|---|
| *Socrates snores* | SNORE(s) | S(s) | $\mathbf{S_1 s}$ |
| *Peter is honest* | HONEST(p) | H(p) | **Hp** |
| *Mary sees Peter* | SEE(m,p) | S(m,p) | $\mathbf{S_2 m p_1}$ |
| *Mary gives Paul Lucy* | GIVE(m,p,l) | G(m,p,l) | $\mathbf{G m p_2 l}$ |

**Note**: In this lecture series, we will work with the Gamut (1991) notation, as most of the concepts and definitions here are developed according to the chapters in their introduction.

# Predicates

When there is no **concrete example sentence in English** (or any other language) that a predicate logic formulation refers to, then the notation might use some upper case letter which represents **some particular predicate** which is not further defined. Gamut do not use indices in this case.

| Z & S: | **Gamut**: |
|---|---|
| P(x) | **Ax** |
| Q(x) | **Bx** |
| R(x,y) | **Axy** |
| S(x,y,z) | **Bxyz** |

**Note**: Importantly, **this is different from a predicate variable**, typically denoted by upper case X, which will become relevant in second-order logic. A variable can stand in for *any* predicate logic constant defined in the language L.



EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

**Faculty of Philosophy**
General Linguistics

# Section 3: Translation to Predicate Logic

# Beyond Propositions

Consider the following (valid) **logical inference** below. If we formulate a propositional logic language to translate these sentences to, we simply represent them with propositional letters. Which, in fact, yields an invalid inference.

(3) Premise 1: Casper is bigger than John.
Premise 2: John is bigger than Peter.

---

Conclusion: *Therefore, Casper is bigger than Peter*.

(4) Premise 1: p
Premise 2: q

---

Conclusion: r

Gamut, L.T.F (1991). Volume 1, p. 66-67.

# The Predicate Logic Alternative

If, on the other hand, we translate these sentences into predicates reflecting the original relations between Casper, Peter, and John, then we get a better reflection of the natural language sentence structure. This can then be used to create a logically valid inference.

(5) Premise 1: Casper is bigger than John.
Premise 2: John is bigger than Peter.

---

Conclusion: *Therefore, Casper is bigger than Peter.*

(6) Premise 1: Bcj
Premise 2: Bjp

---

Conclusion: Bcp

# Translation Key

In order to translate a set of natural language sentences into predicate logic expressions unambiguously, we need a **translation key** listing the **predicates** and **constant symbols**.

Gamut, L.T.F (1991). Volume 1, p. 68.



**English sentences:**

(1) John is bigger than Peter or Peter is bigger than John.

(2) Alkibiades does not admire himself.

(3) If Socrates is a man, then he is mortal.

(4) Ammerbuch lies between Tübingen and Herrenberg.

(5) Socrates is a mortal man.

**Translation key:**

$a_1$: Alcibiades
$a_2$: Ammerbuch
j: John
p: Peter
s: Socrates
t: Tübingen
h: Herrenberg

Axy: x admires y
$B_1xy$: x is bigger than y
$B_2xyz$: x lies between y and z
$M_1x$: x is a man
$M_2x$: x is mortal

# Translation Examples

We can then translate the natural language sentences into predicate logic by further identifying the logical operators, i.e. connectives and negation.

Gamut, L.T.F (1991). Volume 1, p. 68.



| **English sentences:** | **Translations:** |
| --- | --- |
| (1) John is bigger than Peter **or** Peter is bigger than John. | (1) $B_1jp \vee B_1pj$ |
| (2) Alcibiades does **not** admire himself. | (2) $\neg Aa_1a_1$ |
| (3) **If** Socrates is a man, **then** he is mortal. | (3) $M_1s \rightarrow M_2s$ |
| (4) Ammerbuch lies between Tübingen and Herrenberg. | (4) $B_2a_2th$ |
| (5) Socrates is a mortal man. | (5) $M_1s \wedge M_2s$ |

# Translation with Quantifiers

If $\phi$ is an expression of predicate logic, then $\forall x\phi$ is called the **universal generalization** of $\phi$. Likewise, $\exists x\phi$ is called the **existiential generalization**.

Gamut, L.T.F (1991). Volume 1, p. 71.



| **English sentences:** | **Translations:** |
|---|---|
| (1) Everyone is friendly. | (1) $\forall$xFx |
| (2) Someone is friendly. | (2) $\exists$xFx |
| (3) No one is friendly. | (3) $\neg\exists$xFx |
| (4) Everyone is unfriendly. | (4) $\forall$x$\neg$Fx |

**Notes**: We have to add *Fx: x is friendly* to the key. Someone/somebody and no one/nobody are seen as equivalent. Further, note that while we would clearly consider (3) and (4) two different English sentences, the predicate logic translations are perfectly equivalent, i.e. $\neg\exists xFx \equiv \forall x\neg Fx$.

**English sentences:**

(1) Socrates admires someone.

(2) Socrates is admired by someone.

(3) All teachers are friendly.

(4) Some teachers are friendly.

(5) Some friendly people are teachers.

(6) All teachers are unfriendly.

(7) Some teachers are unfriendly.

**Translations:**

(1) $\exists$yAsy

(2) $\exists$xAxs

(3) $\forall$x(Tx$\rightarrow$Fx)

(4) $\exists$x(Tx$\wedge$Fx)

(5) $\exists$x(Fx$\wedge$Tx)

(6) $\forall$x(Tx$\rightarrow \neg$Fx)

(7) $\exists$x(Tx$\wedge\neg$Fx)



**Notes**:
We have to add *Tx: x is a teacher* to the key.

Due to the so-called commutatitivity of $\wedge$, i.e. $\phi \wedge \psi \equiv \psi \wedge \phi$, we have that the predicate logic expressions in (4) and (5) are seen as equivalent too. However, the asymmetry might be seen as actually relevant in the natural language examples.

Generally, we have that $\forall$x$\neg\phi \equiv \neg\exists$x$\phi$.

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Faculty of Philosophy
General Linguistics

# Section 4: The Syntax of Predicate Logic

# The Syntax: Recursive Definition

Given the vocabulary of $L$ we define the following clauses to create formulas of $L$.

(i) If A is an n-ary predicate letter in the vocabulary of $L$, and each of $t_1, \ldots, t_n$ is a constant or a variable in the vocabulary of $L$, then $At_1, \ldots, t_n$ is a formula in $L$.

(ii) If $\phi$ is a formula in $L$, then $\neg\phi$ is too.

(iii) If $\phi$ and $\psi$ are formulas in $L$, then $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, and $(\phi \leftrightarrow \psi)$ are too.

(iv) If $\phi$ is a formula in $L$ and $x$ is a variable, then $\forall x\phi$ and $\exists x\phi$ are formulas in L.

(v) Only that which can be generated by the clauses (i)-(iv) in a finite number of steps is a formula in $L$.

Gamut, L.T.F (1991). Volume 1, p. 75.

# Examples of Valid and Invalid Formulas

| Formula | Rule Applied |
|---|---|
| $Aa$ ✓ | (i) |
| $Ax$ ✓ | (i) |
| $Aab$ ✓ | (i) |
| $Axy$ ✓ | (i) |
| $\neg Axy$ ✓ | (i) and (ii) |
| $Aa \rightarrow Axy$ ✓ | (i) and (iii) |
| $\forall x(Aa \rightarrow Axy)$ ✓ | (i),(iii), and (iv) |
| $\forall x Aa \rightarrow Axy$ ✓ | (i),(iii), and (iv) |
| $a$ x | – |
| $A$ x | – |
| $\forall$ x | – |
| $\forall(Axy)$ x | – |

# Building Construction Trees

Just as for propositional logic expressions, we can also **build construction tress for predicate logic expressions** (if they are correctly derived).

$$\exists x(Tx \wedge Fx) \rightarrow \exists x(Fx \wedge Tx) \quad (iii, \rightarrow)$$

$$\exists x(Tx \wedge Fx) \quad (iv, \exists) \qquad\qquad \exists x(Fx \wedge Tx) \quad (iv, \exists)$$

$$Tx \wedge Fx \quad (iii, \wedge) \qquad\qquad Fx \wedge Tx \quad (iii, \wedge)$$

$$Tx \quad (i) \qquad Fx \quad (i) \qquad Fx \quad (i) \qquad Tx \quad (i)$$

**Note**: The **number of branchings** (depth of embedding) is still given by the number of connectives. **Quantifiers behave like negation** here in the sense of creating unary branches. Also, note how we now have predicates as **atomic formulas**, i.e. *terminal symbols*, rather than single letters representing propositions.

# Definition: Quantifier Scope

"If $\forall x\psi$ is a subformula of $\phi$, then $\psi$ is called the scope of this particular occurrence of the quantifier $\forall x$ in $\phi$. The same applies to occurrences of the quantifier $\exists x$."

Gamut, L.T.F (1991). Volume 1, p. 76.



Assume $\phi \equiv \neg\exists x\exists y(\forall z(\exists wAzw \rightarrow Ayz) \wedge Axy)$, we then have the following quantifiers and scopes for subformulas of $\phi$:

| **Quantifier** | **Scope** |
|---|---|
| $\exists w$ | $Azw$ |
| $\forall z$ | $\exists wAzw \rightarrow Ayz$ |
| $\exists y$ | $\forall z(\exists wAzw \rightarrow Ayz) \wedge Axy$ |
| $\exists x$ | $\exists y(\forall z(\exists wAzw \rightarrow Ayz) \wedge Axy)$ |

Note: The opening and closing brackets generally indicate the quantifier scope when connectives are involved, except for outer brackets, which can be dropped. If no connective is involved, then we don't need the brackets, e.g. $\exists wAzw$ rather than $\exists w(Azw)$.

# Definition: *Formula* vs. *Sentence*

There is a further distinction between **formulas** and **sentences** in predicate logic. Namely, sentences are a subset of formulas for which it holds that: "A sentence is a formula in *L* which **lacks free variables**."[6]

Gamut, L.T.F (1991). Volume 1, p. 77.



| Sentence | Not a Sentence (but Formula) |
|---|---|
| $Aa$ | $Ax$ |
| $\forall x(Fx)$ | $Fx$ |
| $\forall x(Ax \rightarrow \exists y By)$ | $Ax \rightarrow \exists y By$ |

[6]Free variables, in turn, are precisely defined by Gamut (1991), p.77 in their Definition 3. We will simply state here that a variable is free if it is not within the scope of a quantifier.

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

**Faculty of Philosophy**
General Linguistics

# Section 5: The Semantics of Predicate Logic

# Domain of Discourse

In predicate logic, we use quantifications as in *everybody is friendly*. This means we have to define the **domain of discourse** (*D*) as a set of entities (*e*), since statements of this type might be true or false in one domain (e.g. Hawaii), but not in another (e.g. Germany).

$$D = \{e_1, e_2, ..., e_i\} \text{ with } D \neq \{\}. \tag{3}$$

Gamut, L.T.F (1991). Volume 1, p. 88.

# Interpretation Functions

"The interpretation of the constants in $L$ will therefore be an attribution of some entity in $D$ to each of them, that is, a function with the set of constants in $L$ as its domain and $D$ as its range. Such functions are called **interpretation functions**."

$$I(c_i) = e_i. \tag{4}$$

"$I(c)$ is called the *interpretation of a constant* $c$, or its *reference* or its *denotation*, and if $e$ is the entity in $D$ such that $I(c) = e$, then $c$ is said to be one of $e$'s names ($e$ may have several different names)."

Gamut, L.T.F (1991). Volume 1, p. 88.



**Example**

$I = \{\langle m, e_1\rangle, \langle s, e_1\rangle, \langle v, e_1\rangle\}$
$I(m) = e_1$
$I(s) = e_1$
$I(v) = e_1$
Translation key: m: morning star; s: evening star; v: venus.

# Definition: A model **M** for language $L$[7]

"A model **M** for a language $L$ of predicate logic consists of a domain $D$ (this being a nonempty set) and an interpretation function $I$ which [...] conforms to the following requirements:

(i) if $c$ is a constant in $L$, then $I(c) \in D$;

(ii) if $B$ is an n-ary predicate letter in $L$, then $I(B) \subseteq D^n$."

Gamut, L.T.F (1991). Volume 1, p. 91.

**Example**

$D = \{e_1, e_2, e_3\}$
$I(m) = e_1$
$I(j) = e_2$
$I(p) = e_3$
$I(S) \subset D^2$
Translation key: j: John; p: Peter; m: morning star; Sxy: x sees y.

[7]The approach we follow here is called *Approach A* or *the interpretation of quantifiers by subsitution* in Gamut (1991), p. 89. There is also another alternative Approach B, which we do not consider here.

# Definition: The valuation function $\mathbf{V}_M$

"If **M** is a model for $L$ whose interpretation function $I$ is a function of the constants in $L$ onto the domain $D$, then $\mathbf{V}_M$, the valuation $V$ based on $M$, is defined as follows:"

(i) If $Aa_1, \ldots, a_n$ is an atomic sentence in $L$, then $\mathbf{V}_M(Aa_1, \ldots, a_n) = 1$ if and only if $\langle I(a_1), \ldots, I(a_n)\rangle \in I(A)$.

(ii) $V_M(\neg\phi) = 1$ iff $V_M(\phi) = 0$.

(iii) $V_M(\phi \wedge \psi) = 1$ iff $V_M(\phi) = 1$ and $V_M(\psi) = 1$.

(iv) $V_M(\phi \vee \psi) = 1$ iff $V_M(\phi) = 1$ or $V_M(\psi) = 1$.

(v) $V_M(\phi \rightarrow \psi) = 0$ iff $V_M(\phi) = 1$ and $V_M(\psi) = 0$.

(vi) $V_M(\phi \leftrightarrow \psi) = 1$ iff $V_M(\phi) = V_M(\psi)$.

Gamut, L.T.F (1991). Volume 1, p. 91.

# Definition: The valuation function $\mathbf{V}_M$

(vii) $V_M(\forall x \phi) = 1$ iff $V_M([c/x]\phi) = 1$ for all constants c in L.

(viiii) $V_M(\exists x \phi) = 1$ iff $V_M([c/x]\phi) = 1$ for at least one constant c in L.

If $V_M(\phi) = 1$, then $\phi$ is said to be true in model **M**.

Gamut, L.T.F (1991). Volume 1, p. 91.

**Note:** The notation [c/x] means "replacing x by c". Note that this valuation works only for **sentences** of predicate logic as defined above. That is, it works for formulas that consist of atomic sentences and/or formulas with variables that are bound. For *formulas with free variables*, it does not work.

# Valuation Example

Given a Model of the world **M**, consisting of $D$ and $I$, and some formula $\phi$ which adheres to predicate logic syntax (and which consists of atomic sentences and or quantifications with bound variables), we can then evaluate the truth of $\phi$ as follows.

Model **M**

$D = \{e_1, e_2, e_3\}$
$I = \{\langle j, e_1\rangle, \langle p, e_2\rangle, \langle m, e_3\rangle, \langle S, \{\langle I(j), I(m)\rangle, \langle I(p), I(m)\rangle\}\rangle\}$
$I(S) = \{\langle I(j), I(m)\rangle, \langle I(p), I(m)\rangle\}$
Translation key: j: John; p: Peter; m: morning star; Sxy: x sees y.

## Valuation

“John sees the morning star”: $V_M$(Sjm) = 1 (according to (i))
“Everybody sees the morning star”: $V_M$(∀xSxm) = 0 (according to (vii))[8]

[8] This valuation gives 0 since the morning star (m) is a constant c in L, but it does not see itself, i.e. $\langle I(m), I(m)\rangle \notin S$.

# Summary

# Summary

- **Predicate logic** goes beyond propositional logic by, firstly, teasing apart predicates and their arguments, and secondly, introducing quantifiers.
- These changes lead to **adjustments in the formal definitions** of the syntax and semantics of the logical language L.
- While these adjustments enable a more precise translation of natural language sentences, there are also still plenty of **disagreements** with the predicate logic language L.

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

**Faculty of Philosophy**
General Linguistics

# References

# References

Gamut, L.T.F (1991). *Logic, Language, and Meaning. Volume 1: Introduction to Logic*. Chicago: University of Chicago Press.

Kroeger, Paul R. (2019). *Analyzing meaning. An introduction to semantics and pragmatics.* Second corrected and slightly revised version. Berlin: Language Science Press.

Zimmermann, Thomas E. & Sternefeld, Wolfgang (2013). *Introduction to semantics. An essential guide to the composition of meaning.* Mouton de Gruyter.



# Thank You.

Contact:

**Faculty of Philosophy**
General Linguistics
Dr. Christian Bentz
SFS Wihlemstraße 19-23, Room 1.24
chris@christianbentz.de
Office hours:
During term: Wednesdays 10-11am
Out of term: arrange via e-mail