Formal Logic

Lecture 6: The Syntax of Predicate Logic

Dr. Ioannis Votsis

ioannis.votsis@nchlondon.ac.uk
www.votsis.org

Predicate Logic

Propositional vs. predicate logic

DEDUCTIVE LOGIC

PROPOSITIONAL LOGIC

Formal study of arguments at the coarse-grained level of whole sentences.

NB: A.k.a. ‘propositional calculus’, ‘sentential logic’.

PREDICATE LOGIC

Formal study of arguments at the sub-sentential (and thus more fine-grained) level of sentences.

NB: A.k.a. ‘first-order logic’, ‘predicate calculus’.

The coarse-grained nature of L_1

- The coarseness of propositional logic is made obvious when we attempt to formalise some arguments in L_1.

Example Formalised:	**Natural Language Example**:
1. P	1. Zeno is a tortoise.
2. Q	2. All tortoises are toothless.
3. R	Therefore, Zeno is toothless.

- Although the argument is clearly valid in natural language, it's not (semantically or syntactically) valid in L_1.

- That's because each natural language sentence cannot but be formalised using a different sentence letter in L_1.

Peering inside a sentence

- That's where predicate logic comes in as it offers us a more expressive (read: more fine-grained) formal language.
- To be precise, it allows us to peer inside the structure of natural language sentences.
- In particular, it allows us to track similarities in the internal structure of those sentences.

Natural Language Example Revisited:

1. **Zeno** is a **tortoise**.
2. All **tortoises** are **toothless**.

Therefore, **Zeno** is **toothless**.

Designator-predicate form

- In more detail, predicate logic allows us to track and formalise designator-predicate (a.k.a. subject-predicate) form.

- **Designators** (proper names and definite descriptions) are singular noun phrases that denote unique objects.

- **Predicates** denote properties that we ascribe to objects.

Designators:	*Predicates*:
Anthony (Grayling)	... is 70 years old.
the Drawing Room (at NCH)	... is spacious.
Garfield (the cat)	... is known to be lazy.
Lima	... is the capital of Peru.

One-place predicates

- One-place (a.k.a. unary) predicates ascribe properties to the unique objects identified by single designators.

Aristophanes is a playwright.

Tom is a cat.

Jerry has big ears.

Usain Bolt runs fast.

Surfing is fun.

NB: Where **designators** are in **green** and **predicates** in **red**.

Many-place predicates

- Many-place predicates ascribe properties to two or more objects. They're known as binary, ternary, etc, predicates.

Othello loves Desdemona.	2-place
The senate acquitted Trump.	2-place
Manchester is between London and Edinburgh.	3-place
Big Bird is taller than Elmo, Bert and Oscar.	4-place
The queen gives parliament her speech.	3-place

NB: Where **designators** are in **green** and **predicates** in **red**.

Predicates may be distributed

- One and the same predicate may be distributed across a natural language sentence.

Manchester is between London and Edinburgh.

1st part: is between — 2nd part: and

Big Bird is taller than Elmo, Bert and the Cookie Monster.

1st part: is taller than — 2nd part: and the

The engineer is loosening the nut with the wrench.

1st part: is loosening — 2nd part: with

The syntax of L_2

- Predicate logic employs an artificial language – we call it 'L_2'. The expressions of L_2 consists of:

  non-logical terms:
  * **upper case letters** denoting **predicates**, e.g. *P*, *Q* and *R*.
  * **lower case letters** at the start of the alphabet denoting **constants**, e.g. *a*, *b* and *c*.
  * **lower case letters** at the end of the alphabet denoting **variables**, e.g. *x*, *y* and *z*.

  logical terms:
  * **symbols** denoting **logical connectives**, i.e. ¬, &, ∨, →, ↔.
  * **symbols** denoting **quantifiers**, i.e. ∀, ∃.

  NB: An infinite # of predicates, constants and variables can be produced via natural number subscripts, e.g. P_3, a_6, x_{14}.

The *arity* of a predicate

- Halbach (2010) uses the convention of natural number superscripts to indicate the *arity* of a predicate.

  Examples:

C^2dz	2-place (binary)
A^1a	1-place (unary)
P^3ayc	3-place (ternary)
R	0-place (a sentence)

- It should be clear that the superscripts are redundant as the number of constants + variables fixes the arity of a predicate.

- Although I do not intend to follow this convention, it's worth being familiar with it as some of Halbach's exercises require it.

Constants, variables and atomic formulae

- To understand well-formed formulae in L_2 we need the concepts of constant, variable and atomic formula.

  Constants (in L_2): These denote unique objects. They are the corresponding notion of natural language designators in L_2.

  Variables (in L_2): These can take a range of values, namely any constant within L_2.

  Atomic formula (in L_2):
  If P is a predicate letter of arity n and every term $t_1,\ldots, t_n$ is a variable or constant, then $Pt_1\ldots t_n$ is an atomic formula of L_2.

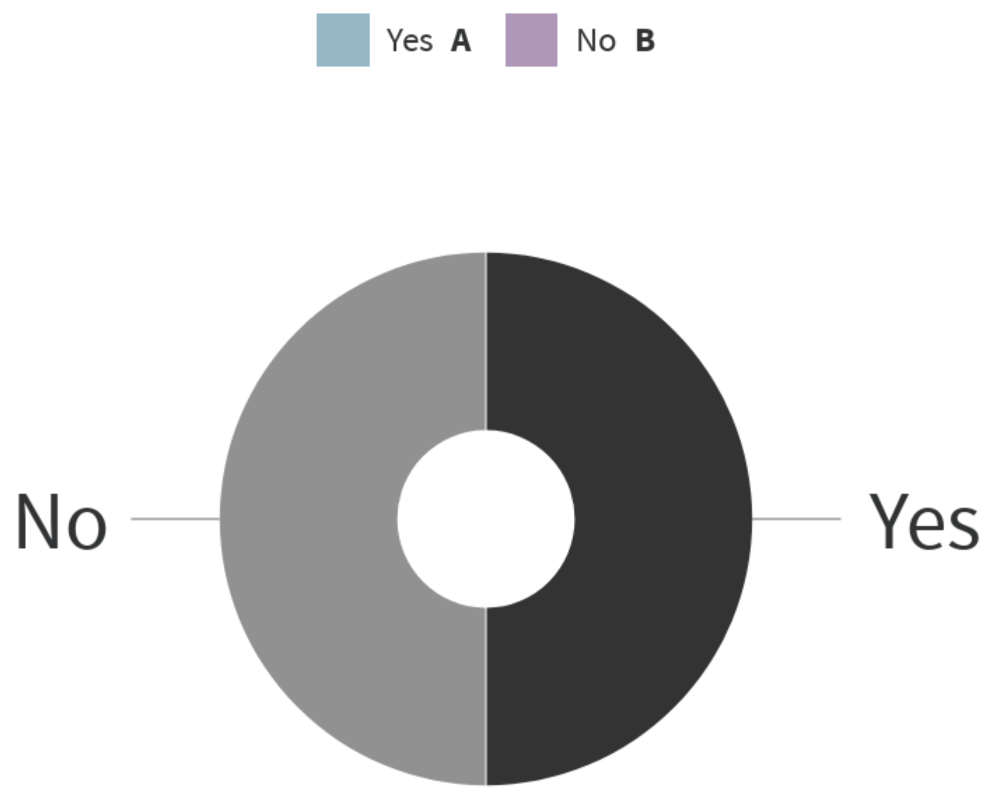
Formulae in L_2

- Definition of a (**well-formed**) **formula** in L_2 :

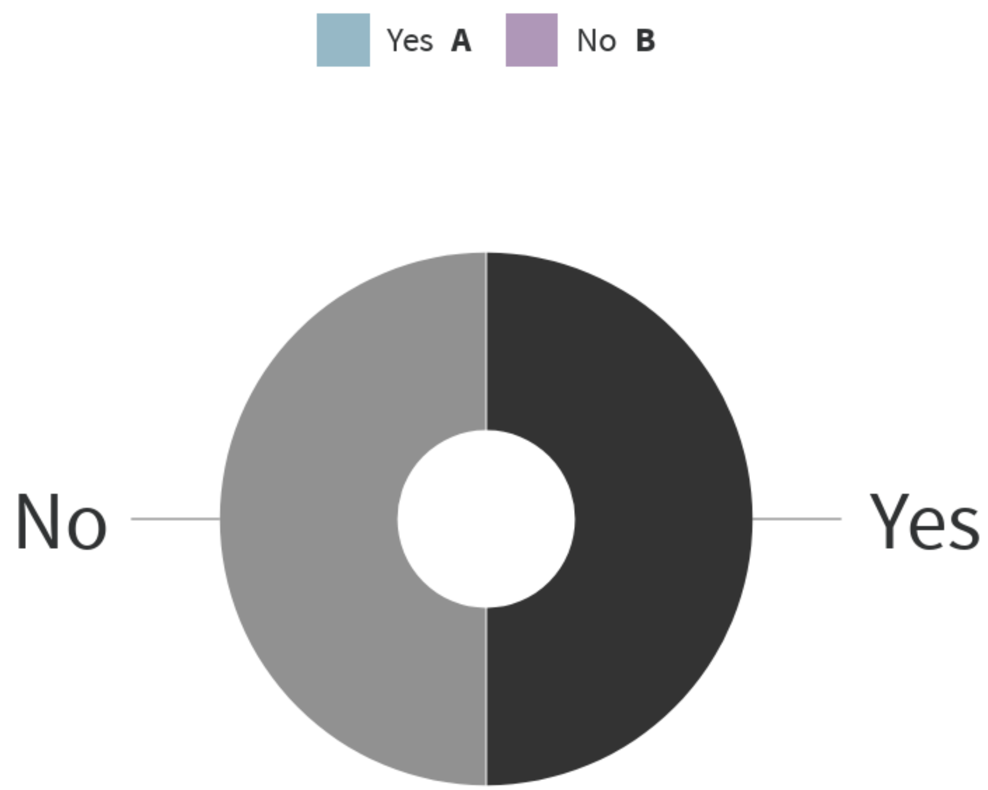
1. All atomic formuale of L_2 are formulae of L_2.
2. If ϕ, ψ are formulae of L_2, then $\neg\phi$, $(\phi \& \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$ and $(\phi \leftrightarrow \psi)$ are formulae of L_2.
3. If *v* is a variable and ϕ is a formula, then $(\forall v)\phi$ and $(\exists v)\phi$ are formulae of L_2.
4. Nothing else is a formula of L_2.

NB: Unlike with propositional logic, we are more careful here and use meta-variables ϕ, ψ in the definition of an L_2 formula.

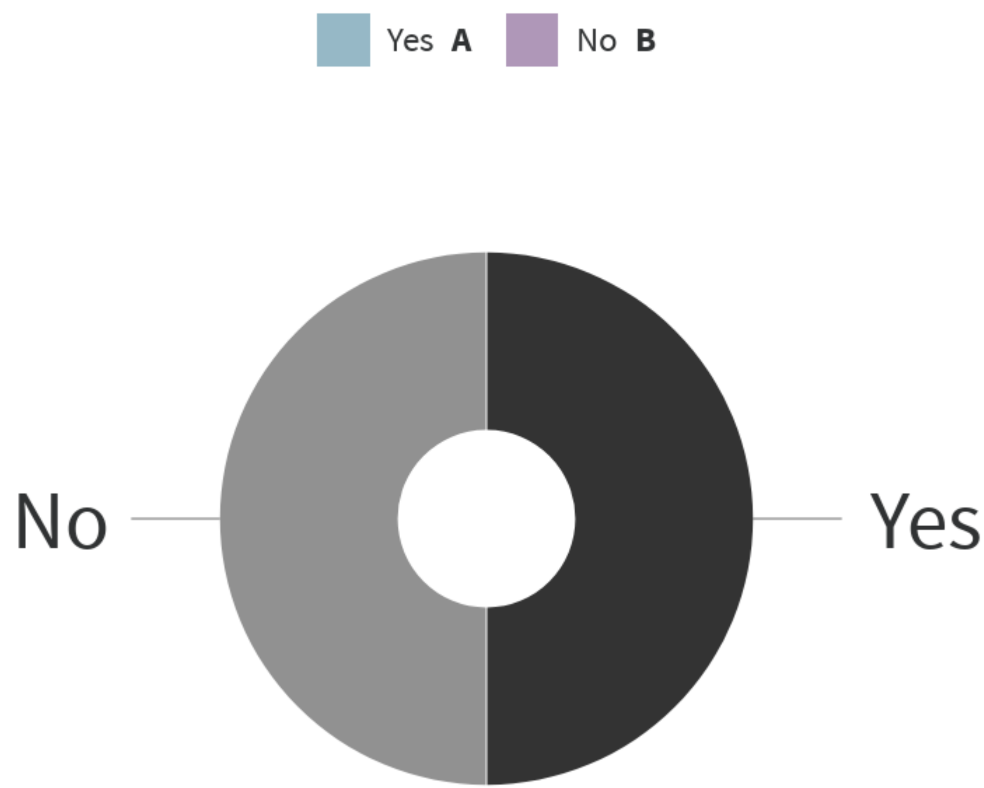
Is this a formula of L2? (∀x)&Pxa



Is this a formula of L2? (∀x)(∀y)¬Sxy



Is this a formula of L2? $\neg(\forall\exists x)(Qx \vee Rx)$



Formulae in L_2: Examples

- **Examples of L_2 formulae**:

¬Ab	(Aa&Aa)
(Adz&(Baef∨Cabcd))	((Dc→Bdx)↔(¬¬Caa→Gy))
(∀x)(Pxa→Qx)	(∀x)(∀y)¬Sxy
¬(∀x)(Ad→Bx)	(∀x)¬(∃y)(Pxy→Ry)

Examples of expressions which are not L_2 formulae:

¬&Ab	(Ab&B¬c)
(Aaz&&By)	(Ag →Bhd)Ca
(∀x)&Pxa	(∀¬x)(∀y)Sxy
¬(∀∃x)(Qx∨Rx)	(∀x)¬(∃y&z)(Uxy→Ez)

Quantifiers and scope

- To properly understand quantifiers we need to have a proper understanding of scope.

  "The scope of a quantifier in a formula P of PL is the quantifier itself and the subformula Q that immediately follows the quantifier" (Bergmann et al. 2014: 271).

  Examples (scope underlined):

  $\underline{(\exists y)(Fyz \,\&\, Gzy)}$

  $Hx \rightarrow \underline{(\forall y)Fxy}$

  $\underline{(\exists w)(Gwa \rightarrow Fa)} \leftrightarrow Hw$

  $\underline{(\forall x)\neg\underline{(\exists y)(Pxy \rightarrow Ry)}}$

Quantifiers, scope and parentheses

- Parentheses are crucial in helping us fix the scope of quantifiers.

- The following two sentences are not identical in meaning:

1. $(\forall x)(Fx \supset Ga)$

2. $(\forall x)\ Fx \supset Ga$

NB: In the absence of parentheses, the scope extends only to the atomic formula to the immediate right of a quantifier.

No double booking

- Whenever you use more than one quantifier with *nested scopes* ensure that they *don't bind with the same variable*.

Correct usage:

$(\forall y)(\exists x)\ Jyx$

$(\forall y)(Hy \rightarrow Rb) \rightarrow (\forall y)\ By$

Incorrect usage:

$(\forall \mathbf{y})(\exists \mathbf{y})\ Jyx$

$(\forall \mathbf{x})(Qxc\ \&\ (\exists \mathbf{x})(Pxc\ \&\ Ra))$

Variables: Free and bound instances

- Roughly, a variable instance in an L_2 formula is free (it occurs freely) when it's not bound (tied) to a quantifier: ∀ or ∃.

Free	**Bound**
P**x**	(∀**x**)P**x**
Q**z**a	(∃**z**)Q**z**a
(∀y)(Py**x**→Ra)	(∀**y**)(P**y**x→Ra)

- Here's the definition: "(i) All occurrences of variables in atomic formulae are free. (ii) The occurrences of a variable that are free in ϕ and ψ are also free in ¬ϕ, (ϕ&ψ), (ϕ∨ψ), (ϕ→ψ) and (ϕ↔ψ). (iii) In a formula (∀*v*)ϕ or (∃*v*)ϕ no occurrence of the variable *v* is free; all occurrences of variables other than *v* that are free in ϕ are also free in (∀*v*)ϕ and (∃*v*)ϕ" (Halbach p. 85).

.

Sentences in L_2

- A formula ϕ in L_2 is a sentence in L_2 if and only if ϕ contains no free instances of variables.

- In other words, all sentences in L_2 are formulae in L_2 but not vice-versa.

Examples:

Pdae

Rc & Gc

(Eh ∨ Jc)↔Fb

(∀x)(∀y)¬¬Fxy

¬(∃y)(Ray&By)

Pab → (∃y)(¬Pby & (∀x)¬Qxy)

The End