Formal Logic

Lecture 10: Predicate Logic with Identity

Dr. Ioannis Votsis

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

Syntax, Semantics and Formalisation

Revisiting L_2 (1)

- L_2 is rich enough to express many natural language subtleties but it has various limitations.

- One such limitation is that some valid arguments in natural language cannot be expressed as valid in L_2.

Mary Ann Evans is the same person as George Eliot. Mary Ann Evans is a writer. Therefore, George Eliot is a writer. **VALID**

$\{Img, Wm\} \vDash Wg$ **INVALID**

where I: ... is the same person as...
W: ... Is a writer
m: Mary Ann Evans
g: George Eliot

Revisiting L_2 (2)

- Another limitation of L_2 concerns its ability to capture sentences about quantities.

- Suppose we want to translate the natural language sentence ‘There are at least two spheres’ into L_2.

- It is tempting to offer the following formalisation:

  $(\exists x)(\exists y)(Sx\&Sy)$ where S: ... is a sphere

- This won’t work as *x* and *y* may pick out the same object, i.e. it comes out true even in a world with one sphere!

A new language: $L_=$

- To deal with such cases the new symbol '=' is introduced into predicate logic. This leads to a new language: $L_=$.

  NB: Halbach (2010) calls this '$L_=$' while Bergmann et al. (2014) call it 'PLE', which stands for 'predicate logic extended'.

- We treat the new symbol as a special binary predicate that expresses numerical identity.

- It is special because unlike other predicates its meaning remains constant throughout.

- That's why it is categorised as a *logical constant* like the logical connectives, e.g. '&', and the operators, e.g. '∃'.

The syntax of $L_=$

- The syntax of $L_=$ is the same as that of L_2, the sole exception being the modification of the notion of an atomic formula:

  "All atomic formulae of L_2 are atomic formulae of $L_=$. Furthermore, if s and t are variables or constants, then s=t is an atomic formula of $L_=$." (Halbach 2010: 167).

  Examples (of the new kind of atomic formula):

x=y	c=f
z=x	x=a
a=b	y=y

- Well-formed formulae in $L_=$ are the same as those in L_2 with the addition of all the atomic formulae that involve identity.

More examples

- **Examples of formulae in $L_=$:**

¬Rf	¬Rf&y=y
(∀x)(Px→c=d)	(∃x)(∃y)¬x=y
(∃x)(∀y)Qxy∨Rj	(∀x)¬(∃y)(Pxy→x=y)
(∀x)(∀y)(∀z)((x=y&y=z)→x=z)	¬a=b

NB: The negation in '¬a=b' ranges over the atomic formula 'a=b'. Some logicians use the inequality symbol '≠', e.g. a≠b.

Examples of expressions that are *not* formulae in $L_=$:

a	¬a
a=¬b	¬&x=y
(∀x)↔Dxa	(∀x)↔x=a
(∀∃z)(Az∨Bz)	(∀∃z)(Az∨z=z)

The semantics of $L_=$

- As with the syntax of $L_=$, its semantics is only a little different from the semantics of L_2.

- We say that $\phi=\psi$ *if and only if* ϕ and ψ denote the same object.

- A consequence of this definition is that $\neg\phi=\psi$ if and only if ϕ and ψ denote distinct objects.

  NB: We read $\neg\phi=\psi$ as 'It is not the case that ϕ equals ψ'.

- Another consequence of the definition is that $\phi=\phi$ is true and $\neg\phi=\phi$ is false under any interpretation.

Formalisation

- Natural language expressions like ‘is equal to’ or ‘is identical to’ are straightforwardly translated using identity in L=.

Natural Language Sentence	**Formalised in $L_=$:**
Superman is identical to Clark Kent.	s=c

where s: Superman and c: Clark Kent

- *Sometimes* the term ‘is’ on its own invites such a translation.

Natural Language Sentence	**Formalised in $L_=$:**
Batman is Bruce Wayne.	b=w

where b: Batman and w: Bruce Wayne

Formalisation: Expressing quantities

Natural Language Sentence:	**Formalised in $L_=$:**
There is at least one thing.	$(\exists x)x=x$
There are at least two things.	$(\exists x)(\exists y)\neg x=y$
There is exactly one thing.	$(\exists x)(\forall y)x=y$
There is at most one thing.	$(\forall x)(\forall y)x=y$
There is at least one piano.	$(\exists x)Px$
There is exactly one piano.	$(\exists x)(Px\&(\forall y)(Py\rightarrow x=y))$
There is at most one piano.	$(\forall x)(\forall y)((Px\&Py)\rightarrow x=y)$
There are at least two pianos.	$(\exists x)(\exists y)((Px\&Py)\&\neg x=y)$
There are exactly two pianos.	$(\exists x)(\exists y)(((Px\&Py)\&\neg x=y)\&$ $(\forall z)(Pz\rightarrow(z=x\vee z=y)))$
There are at most two pianos.	$(\forall x)(\forall y)(\forall z)(((Px\&Py)\&Pz)\rightarrow$ $((x=y\vee y=z)\vee x=z))$

Formalisation: Generalising

- One can thus generalise as follows (where $n>1$):

  There are at least *n* pianos.
  $(\exists x_1)\ldots(\exists x_n)((Px_1\&\ldots\ \&Px_n)\&(\neg x_1=x_2\&\ldots\ \&\neg x_{n-1}=x_n))$

  There are exactly *n* pianos.
  $(\exists x_1)\ldots(\exists x_n)(((Px_1\&\ldots\ \&Px_n)\&(\neg x_1=x_2\&\ldots\ \&\neg x_{n-1}=x_n))\&$
  $(\forall x_{n+1})(Px_{n+1}\rightarrow(x_{n+1}=x_1\vee\ldots\ \vee x_{n+1}=x_n)))$

  There are at most *n* pianos.
  $(\forall x_1)\ldots(\forall x_{n+1})(((Px_1\&Px_2)\&\ldots Px_{n+1})\rightarrow(x_1=x_2\vee\ldots\ \vee x_n=x_{n+1}))$

The End