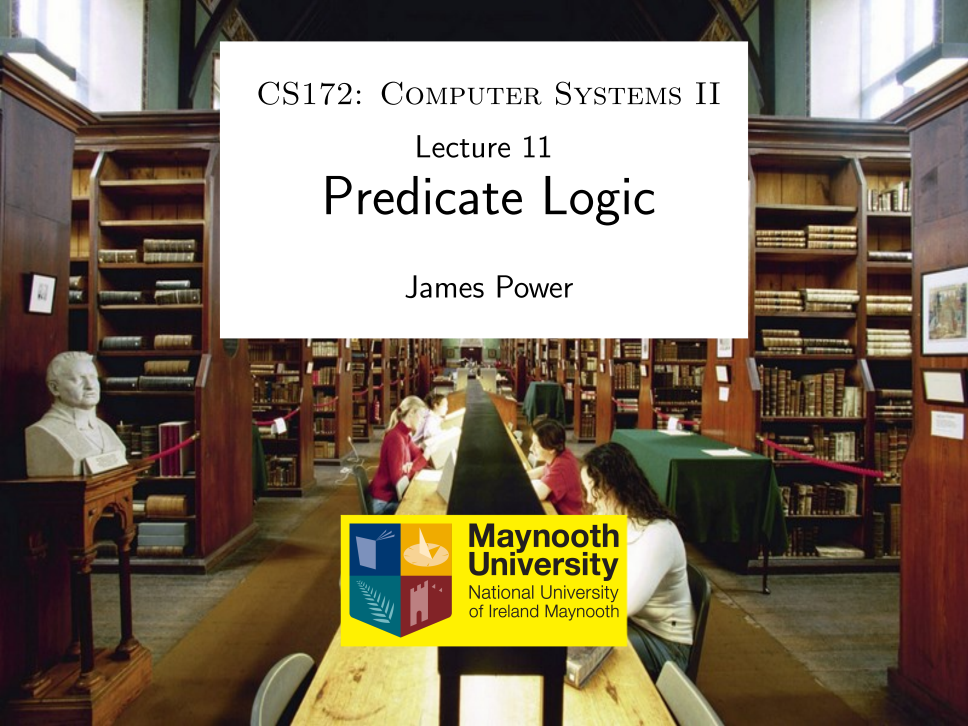


CS172: Computer Systems II

Lecture 11

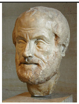
Predicate Logic

James Power

Maynooth University
National University of Ireland Maynooth

Progress of Logic

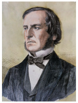
LiA §4

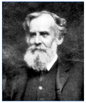


Aristotle, *Prior Analytics*, c. 350 BC

Augustus De Morgan, *Formal Logic or The Calculus of Inference*, 1847.

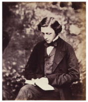
George Boole, *An Investigation of the Laws of Thought*, 1854.





John Venn *Symbolic Logic*, 1881.

Lewis Carroll *Symbolic Logic*, 1896.



Gottlob Frege, *Begriffsschrift: A Formal Language for Pure Thought*, 1879.

Limitations of Propositional Logic

Statement	**Propositional translation**
John reads	p
John walks	q
John sees Mary	r

- Propositional logic misses that these are all statements have something in common: they all concern *John*.

Limitations of Propositional Logic

Statement	Propositional translation	Predicate translation
John reads	p	
John walks	q	
John sees Mary	r	

- Propositional logic misses that these are all statements have something in common: they all concern *John*.
- Solution: Introduce predicates to represent properties and relations.

Statement	Predicate translation
x reads	Rx
x walks	Wx
x sees y	Sxy

Syntax: predicate symbols

Every predicate symbol has an arity, which is the number of operands it takes.

- Example: if L has an arity of 2, then it takes 2 arguments, Lxy
- Sometimes this is written explicitly, so that "L/2" is different from "L/1" (a kind of *overloading*)
- Names:

Arity	Name	
0	medadic	nullary
1	monadic	unary
2	dyadic	binary
3	triadic	ternary
n	n-adic	n-ary
	(Greek)	(Latin)

Syntax: *applying* predicate symbols to operands

- A predicate with **arity 2** represents a binary relation; for example we might write Bxy to indicate that x is bigger than y.
  Java analogy:
  ```
  public static boolean B(Person x, Person y)
  ```
- A predicate with **arity 1** represents a property or, equivalently, a (sub-)set; for example we might write Tx to indicate that x is tall.
  Java analogy:
  ```
  public static boolean T(Person x)
  ```
- A predicate with **arity 0** is really a proposition.
  ... analogous to a boolean variable in Java
- **Notation:** in predicate logic we write Lxy to state that "relationship L holds between x and y".
  ... analogous to asserting `L(x,y)` in Java
  ... or writing `(L x y)` in Lisp/ML/Haskell

The *formal language* of predicate logic

LiA §4.5

1. Symbols for constants: $a, b, c, \ldots$
2. Symbols for variables: $x, y, z, \ldots$
3. Symbols for predicates: $A, B, C, \ldots$ or $P, Q, R, \ldots$
4. The five propositional operators: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
5. Two quantifiers:

$$(\forall x \cdots\cdot) \qquad \text{"for all } x \cdots\text{"}$$

$$(\exists x \cdots\cdot) \qquad \text{"there exists an } x \cdots\text{"}$$

First examples: one quantifier

LiA §4.1

Someone walks

Some boy walks

A boy walks

John sees a girl

A girl sees John

A girl sees herself

Everyone walks

Every boy walks

Every girl sees Mary

First examples: one quantifier

LiA §4.1

Someone walks

Some boy walks

A boy walks

John sees a girl

A girl sees John

A girl sees herself

Everyone walks

Every boy walks

Every girl sees Mary

Constants	
j	John
m	Mary

Predicates	
Bx	"x is a boy"
Gx	"x is a girl"

Predicates	
Wx	"x walks"
Sxy	"x sees y"

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Example of a syntactically correct formula:

$$(\forall\, x \cdot Sx \rightarrow (\exists\, y \cdot (\, Ty \vee Vxy) \wedge (\forall\, z \cdot \neg\, Rxyz)))$$

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Example of a syntactically correct formula:

$$(\forall x \cdot Sx \rightarrow (\exists y \cdot (Ty \vee Vxy) \wedge (\forall z \cdot \neg Rxyz)))$$

$$(\forall z \cdot \qquad\qquad)$$

Quantify the variable z

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Example of a syntactically correct formula:

$$(\forall x \cdot Sx \rightarrow (\exists y \cdot (Ty \lor Vxy) \land (\forall z \cdot \neg Rxyz)))$$

$$(\exists y \cdot \qquad\qquad\qquad (\forall z \cdot \qquad))$$

Quantify the variable y

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Example of a syntactically correct formula:

$$(\forall\, x \cdot Sx \rightarrow (\exists\, y \cdot (\, Ty \vee Vxy) \wedge (\forall\, z \cdot \neg\, Rxyz)))$$

$$(\forall\, x \cdot \qquad (\exists\, y \cdot \qquad\qquad\qquad (\forall\, z \cdot \qquad)))$$

Quantify the variable x

Building up a *syntactically correct* formula

LiA §4.2

Once you have determined the constant and predicate symbols you want to use,

- you introduce (and *quantify*) the variables using for-all/there-exists
- and *join them up* using the (propositional) operators

Example of a syntactically correct formula:

$$(\forall\, x \cdot Sx \to (\exists\, y \cdot (\,Ty \vee Vxy) \wedge (\forall\, z \cdot \neg\, Rxyz)))$$

$$(\forall\, x \cdot \qquad (\exists\, y \cdot \qquad\qquad\qquad (\forall\, z \cdot \qquad)))$$

$$Sx \to (\qquad (\,Ty \vee Vxy) \wedge \qquad \neg\, Rxyz\ \ )$$

The formula *without* the quantifiers

Scope of a variable

Just like in Java, when we "declare" a variable using a quantifier it brings it into scope, until the matching end-parenthesis.

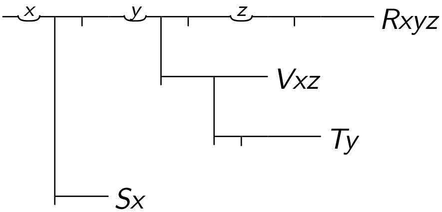
Example:

$$(\forall x \cdot Sx \ \rightarrow \ (\exists y \cdot \ (Ty \ \vee \ Vxy) \ \wedge \ (\forall z \cdot \neg Rxyz)))$$

- Identify the scope of x, y and z
- Any variable under the scope of a quantifier is bound
- Any variable with *no* matching quantifier is free

Frege introduces quantification and scope

$$(\forall x \cdot Sx \rightarrow \neg(\forall y \cdot (\neg Ty \rightarrow Vxz) \rightarrow \neg(\forall z \cdot \neg Rxyz)))$$



Gottlob Frege,
Begriffsschrift: A Formal Language for Pure Thought,
1879.

Example: syllogistic statements

LiA §4.2

All four kinds of sentence from Aristotle's syllogisms can be expressed in predicate logic

A	All A are B	$(\forall x \cdot Ax \rightarrow Bx)$
I	Some A are B	$(\exists x \cdot Ax \wedge Bx)$
E	All A are not B	$(\forall x \cdot Ax \rightarrow \neg Bx)$
	No A is B	$\neg(\exists x \cdot Ax \wedge Bx)$
O	Some A are not B	$(\exists x \cdot Ax \wedge \neg Bx)$
	Not all A are B	$\neg(\forall x \cdot Ax \rightarrow Bx)$

Common idiom: Note the use of

implies with for-all — conjunction with there-exists

Example: beyond syllogisms

- We can talk about relations between objects:

John sees Mary:

Mary sees John:

John gives Mary the book:

- We can express more complicated quantifier patterns:

Everyone sees someone:

Someone sees everyone:

Everyone is seen by someone:

Someone is seen by everyone:
