First-order logic

Reminder about functions & relations.

Cross product / Cartesian product of two sets A and B is given by

$$A \times B = \left\{ (a, b) \mid \begin{array}{l} a \in A \text{ and} \\ b \in B \end{array} \right\}.$$

A relation of arity n is just a subset of some n-arity cross product.

eg $<$ contains $(3,4)$
but not $(5,2)$

$$< \ \subseteq \mathbb{N} \times \mathbb{N}$$

$$< \ = \{ (0,1), (0,2), \ldots, \\ (1,2), (1,3), \ldots, \\ (n, n+1), \ldots \}$$

Functions

$$f(x) = 3x + 1$$

We can also think of functions as sets of pairs

$$f = \{(0,1), (1,4), (2,7), (3,10), (4,13), \dots\}$$

$$f : \mathbb{N} \to \mathbb{N} \quad \subseteq \mathbb{N} \times \mathbb{N}$$

name domain codomain

function outputs are unique. ie For a given input, the output is always the same. $f(3)$ is always 10.

Syntax of FOL

Terms are defined inductively :

function symbol

Term := Variable | $f(t_1, \dots, t_n)$

where variable is $x, y, z, \dots$

eg x is a term

$f'(x, y)$ is a term

$f(g(x), y, g(z), g(f'(x,y)))$

Formula := $P(t_1, \dots, t_n)$ | $t_1 = t_2$ |

predicate

Formula $\wedge$ Formula |

Formula $\vee$ Formula |

Formula $\rightarrow$ Formula |

$\neg$ Formula |

Formal $\leftrightarrow$ Formula |

$\exists$ Variable. Formula |

$\forall$ Variable. Formula

Example

If P has arity 1 and S has arity 2, and f has arity 1, and g has arity 2, then the following are formulae:

$$x = y$$

$$f(x) = g(x,y)$$

$$P(f(x))$$

$$S(x, g(y,y))$$

$$\forall x.\ S(x,x)$$

$$\exists y.\, P(f(y)) \wedge (\exists x.\ x = y).$$

Non-examples

$f(x)$ no, since no predicate or equality.

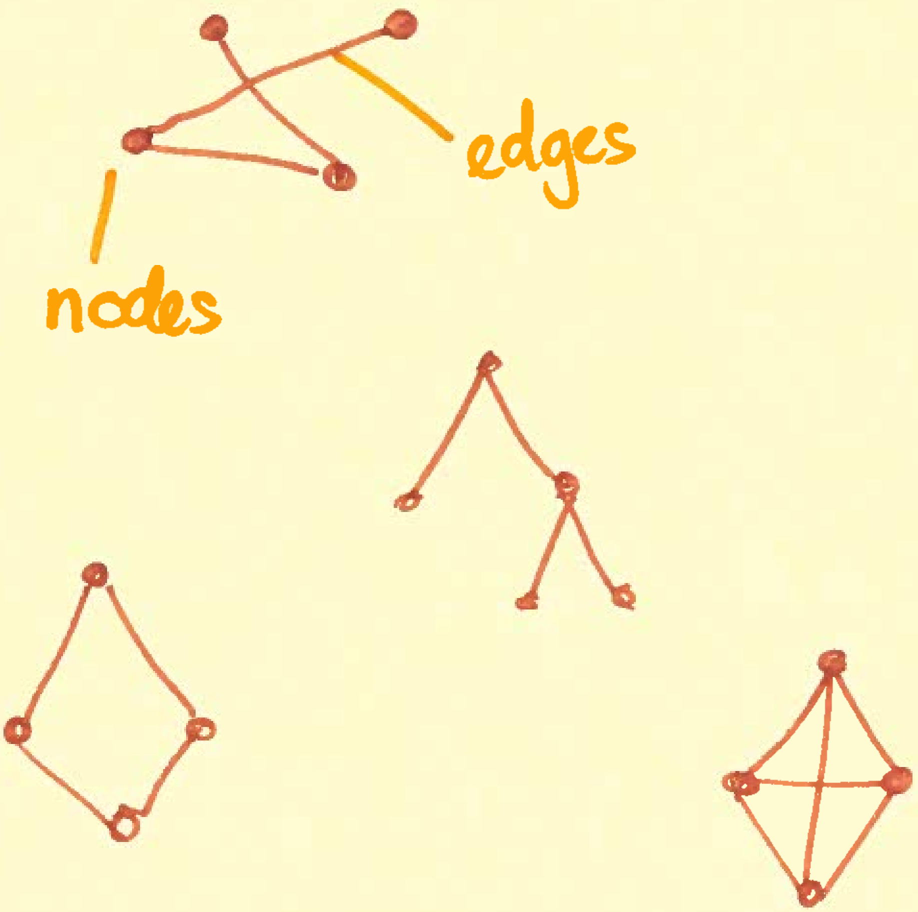
$P(x) = S(x,y)$ no, since equality between terms not formulae.

<u>knowledge-engineering</u>

1. Identify task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions & constants
4. Encode the general information about the ~~knowledge~~ domain
5. Try a specific instance
6. Pose queries
7. Debug

EXAMPLES

Graphs



Vocabulary

N unary predicate for nodes
E binary predicate for edges.

Encapsulating info:

- every edge connects two nodes

- $\forall x.\, N(x)$ is too restrictive
  $\times$ not everything is a node

- $\forall x.\, N(x) \rightarrow (\forall y.\, N(y) \rightarrow$
  $(E(x,y) \rightarrow E(y,x)))$

  If x and y are nodes and there is an edge between x and y, then there is an edge between y & x.

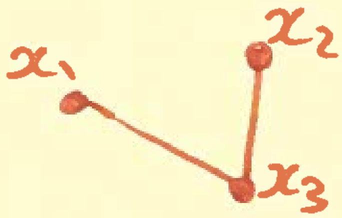
How to shrink this:

$$\forall x, y.\, (N(x) \wedge N(y)) \rightarrow \phi$$

$$\forall x.\, N(x) \rightarrow (\forall y.\, N(y) \rightarrow \phi)$$

Is this enough? Check it.

Use an example:



$N(x_1)$ is true
$N(x_2)$ is true
$N(x_3)$ is true

$E(x_1, x_3)$, $E(x_3, x_1)$
$E(x_2, x_3)$, $E(x_3, x_2)$ are
all true.

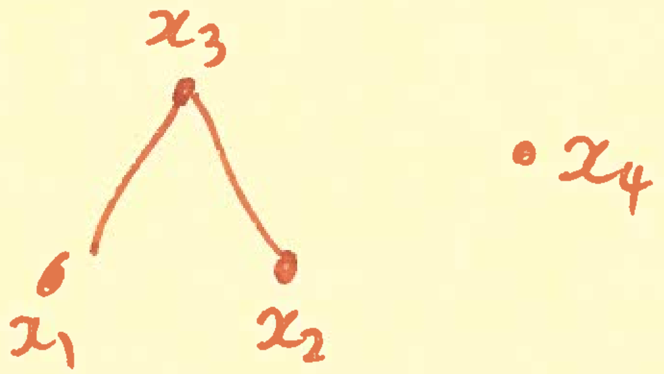
Is

$$\forall x, y. (N(x) \wedge N(y) \wedge E(x,y)) \to E(y,x).$$

true?

Yes. We could check further, but we will not in this case.

What is a graph, according to this?



We don't like this, so edit our axioms to rule it out.

Add – every ~~edge~~ node has at least one edge coming out of it.

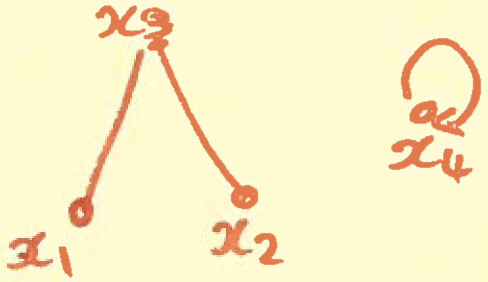
$$\forall x.\ N(x) \rightarrow (\exists y.\ E(x,y))$$

not quite right since y unrestricted.

$$\forall x.\ N(x) \rightarrow \exists y.(N(y) \wedge E(x,y)).$$

(Note: if didn't have other axiom, it would be $\forall x.\ N(x) \rightarrow \exists y.\left(N(y) \wedge \left(E(x,y) \vee E(y,x)\right)\right)$)

Is this enough? Depends on whether



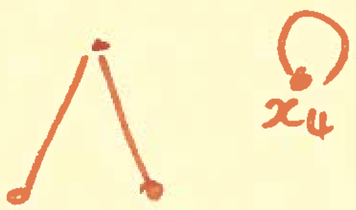
should be a graph.

If not, we need to refine.

We add the restriction that every node must be connected to <u>some other</u> node.

says x is not y.

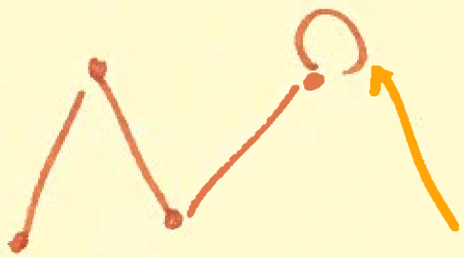
$$\forall x.\ N(x) \to (\exists y.\ \neg(x=y) \land N(y) \land E(x,y))$$



NOT A GRAPH NOW.

Since x_4 is not connected to some other node, only itself.

This is still a graph:



We don't like the self-edge. How do we rule it out?

$$\forall x.\ N(x) \rightarrow (\neg E(x,x))$$

We now have 3 axioms, and graph theory is anything that meets/satisfies those axioms.

The models/semantics are the pictures of the graphs.

eg

is a <u>model</u> for graph theory

EXAMPLES

Students taking modules.

- Every student takes at least one module.
- Every module is taken by at least one student.
- Every student has a unique ID.

Predicates : S is unary (for student)
M is unary (for module)
T is binary (for taken : $T(x,y)$ means x takes y.)

id is a unary function symbol.

1. $\forall x.\ S(x) \to (\exists y.\ M(y) \wedge T(x,y))$

2. $\forall x.\ M(x) \to (\exists y.\ S(y) \wedge T(y,x))$

3. $\forall x,y.\ (S(x) \wedge S(y) \wedge id(x) = id(y)) \to x = y.$

This would seem to work.

We can further restrict it. "Every student takes exactly three modules"

$$1'.\ \forall x.\ S(x) \to (\underbrace{\exists y_1, y_2, y_3.\ M(y_1) \wedge M(y_2) \wedge M(y_3) \wedge T(x,y_1) \wedge T(x,y_2) \wedge T(x,y_3)}_{\phi})$$

No need for y_1, y_2 and y_3 to be different!

Fix it to be three different modules

$$1''.\ \forall x.\ S(x) \to (\phi \wedge \underbrace{\neg(y_1 = y_2) \wedge \neg(y_2 = y_3) \wedge \neg(y_1 = y_3)}_{\phi'})$$

At least three now.

1''''.

$$\forall x.\ S(x) \rightarrow \Big(\phi' \wedge \forall z.\ M(z) \rightarrow \big[(z = y_1) \vee (z = y_2) \vee (z = y_3) \big] \Big)$$

We now have what we need.
We're taking exactly y_1, y_2 and y_3 and <u>nothing else</u>..