First Order Logic – Semantics (3A)

Young W. Lim
7/5/17

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using LibreOffice

Based on

Contemporary Artificial Intelligence,
R.E. Neapolitan & X. Jiang

Logic and Its Applications,
Burkey & Foxley

Examples of Terms

no expression involving a predicate symbol is a term.

$x \quad y \quad f(x) \quad g(x, y)$

$father(x)$ A function returns neither True nor False

The father of x

$Father(x)$ A predicate returns always True or False

Is x a father?

∀x love(x,y) : free variable y
∀x tall(x) : no free variable

Bound variable x
Free variable y

https://en.wikipedia.org/wiki/First-order_logic#Formation_rules

Terms

Terms

1. **Variables**. Any variable is a term.
2. **Functions**. Any expression **f**$(t_1,...,t_n)$ of n arguments is a term
   where each argument t_i is a term and **f** is a function symbol of valence n
   In particular, symbols denoting individual constants are
   0-ary function symbols, and are thus terms.

Only expressions which can be obtained
by finitely many applications of rules 1 and 2 are terms.

no expression involving a predicate symbol is a term.

https://en.wikipedia.org/wiki/First-order_logic#Formation_rules

Formulas

Formulas (wffs)

Predicate symbols.	$P(x) \qquad Q(x, y)$
Equality.	$x = f(y)$
Negation.	$\neg Q(x, y)$
Binary connectives.	$P(x) \land \neg Q(x, y)$
Quantifiers.	$\forall x, y \ (P(x) \land \neg Q(x, y))$

Only expressions which can be obtained
by finitely many applications of rules 1–5 are formulas.

The formulas obtained from the first two rules
are said to be **atomic formulas**.

https://en.wikipedia.org/wiki/First-order_logic#Formation_rules

Formulas

Formulas (wffs)

Predicate symbols. If **P** is an n-ary predicate symbol and t_1, ..., t_n are terms then $\mathbf{P}(t_1,...,t_n)$ is a formula.

Equality. If the equality symbol is considered part of logic, and t_1 and t_2 are terms, then $t_1 = t_2$ is a formula.

Negation. If φ is a formula, then ¬φ is a formula.

Binary connectives. If φ and ψ are formulas, then (φ → ψ) is a formula. Similar rules apply to other binary logical connectives.

Quantifiers. If φ is a formula and x is a variable, then ∀x φ (for all x, holds) and ∃x φ (there exists x such that φ) are formulas.

$P(x) \qquad Q(x, y)$

$x = f(y)$

$\neg Q(x, y)$

$P(x) \wedge \neg Q(x, y)$

$\forall x, y \; (P(x) \wedge \neg Q(x, y))$

https://en.wikipedia.org/wiki/First-order_logic#Formation_rules

Atoms and Compound Formulas

a formula that contains no logical connectives
a formula that has no strict subformulas

Atoms :
the simplest well-formed formulas of the logic.

$P(x) \qquad Q(x, y)$

Compound formulas :
formed by combining the atomic formulas using the logical connectives.

$$P(x) \wedge \neg Q(x, y)$$

$$\forall x, y \ (P(x) \wedge \neg Q(x, y))$$

https://en.wikipedia.org/wiki/Atomic_formula

Atomic Formula

for **propositional logic**
the atomic formulas are the propositional variables — p q

for **predicate logic**
the atoms are predicate symbols together with their arguments,
each argument being a term. — $P(x)$ $Q(x, f(y))$

In **model theory**
atomic formula are merely strings of symbols with a given signature
which may or may not be satisfiable with respect to a given model

https://en.wikipedia.org/wiki/Atomic_formula

A Signature

First specify a **signature**

Constant Symbols	$\{c_1, c_2, \dots c_n\} = D$
Predicate Symbols	$\{\mathbf{P}_1, \mathbf{P}_2, \dots \mathbf{P}_m\}$
Function Symbols	$\{\mathbf{f}_1, \mathbf{f}_2, \dots \mathbf{f}_l\}$

A Language

Determines the **language**

Given a language

A model is specified

A **domain of discourse**

a set of entities $\{entity_1, entity_2, \dots entity_n\}$

An **interpretation**

constant assignments $\{c_1, c_2, \dots c_n\} = D$

function assignments $\mathbf{f}_1(\,), \mathbf{f}_2(\,), \dots \mathbf{f}_l(\,)$

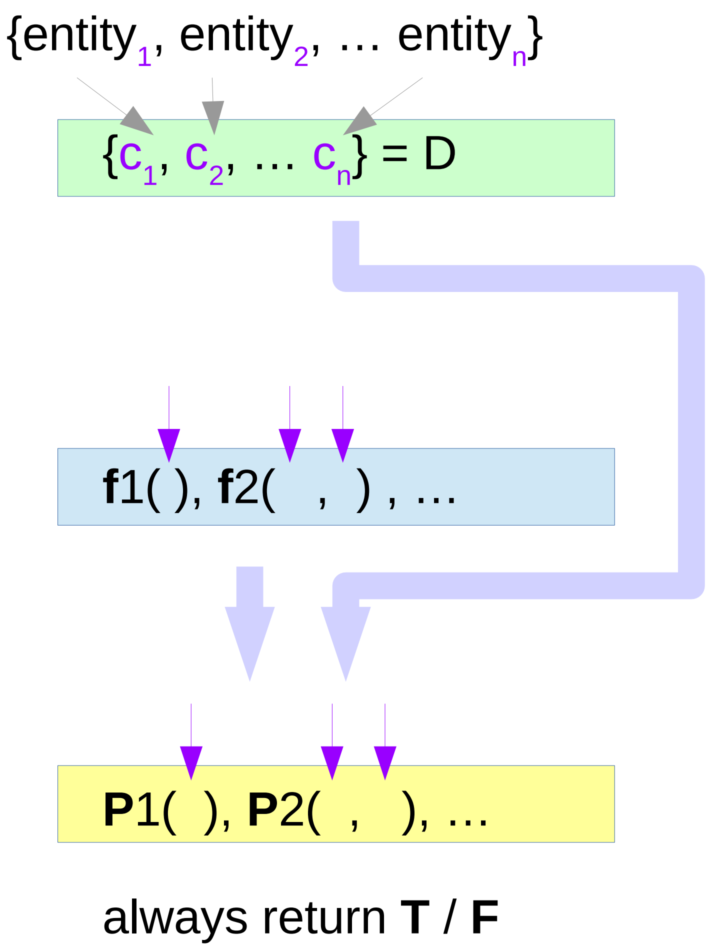
truth value assignments $\mathbf{P}_1(\,), \mathbf{P}_2(\,), \dots \mathbf{P}_m(\,)$

Interpretation

Constant assignments

Function assignments

Truth value assignments



Interpretation

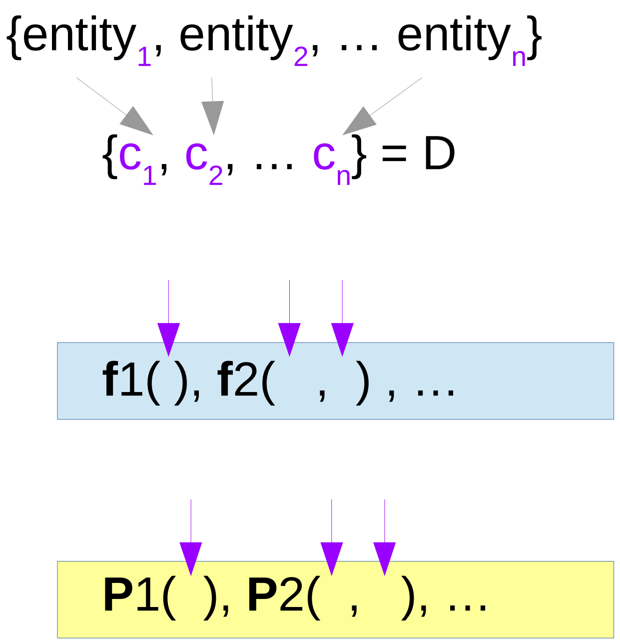
Propositional Logic

	A	B	
Interpretation I_1 ➡	T	T	
Interpretation I_2 ➡	T	F	
Interpretation I_3 ➡	F	T	
Interpretation I_4 ➡	F	F	

First Order Logic

Sentences

	P1( ) …	P2( ) …	S1	S2
Interpretation I_1 ➡	T	T		
Interpretation I_2 ➡	T	F		
Interpretation I_3 ➡	F	T		
Interpretation I_4 ➡	F	F		

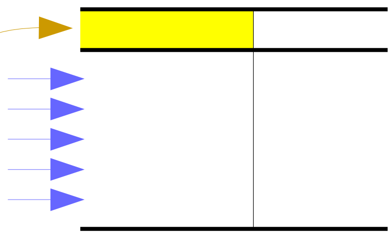


always return **T** / **F**

A Model

A **model** or a **possible world**:

Every **atomic proposition** is assigned a value **T** or **F**

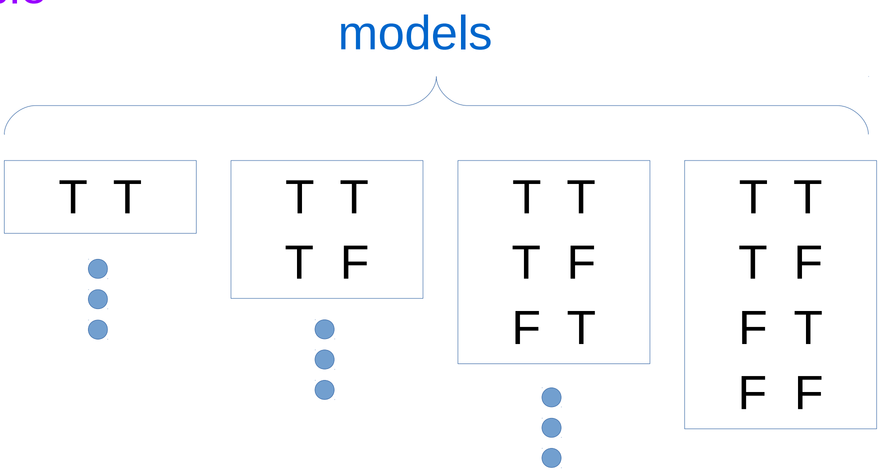


The set of **all** these assignments constitutes
A **model** or a **possible world**

All possible worlds (assignments) are permissible

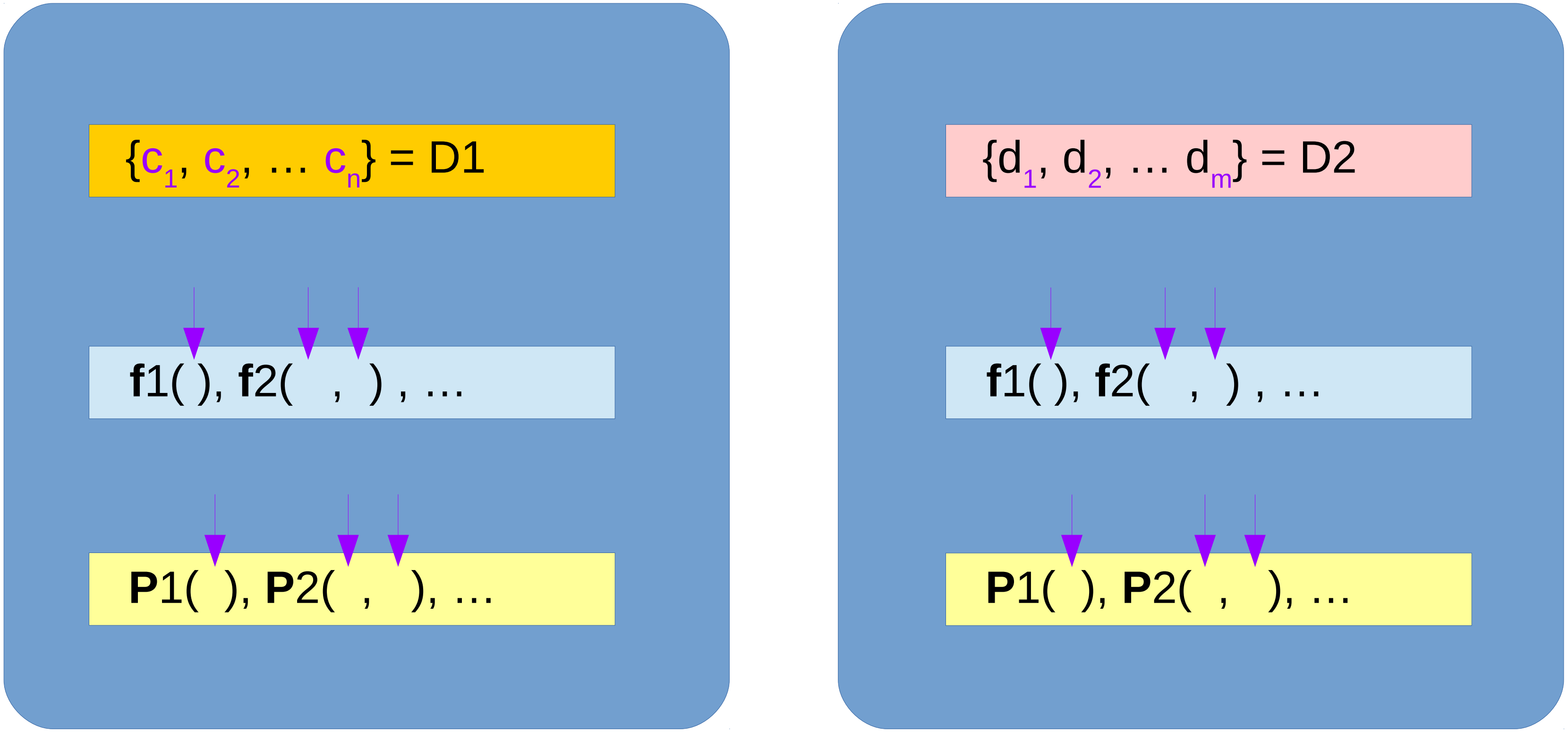
A	B	A∧B	A∧B ⇒ A
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	T

Every atomic proposition : A, B

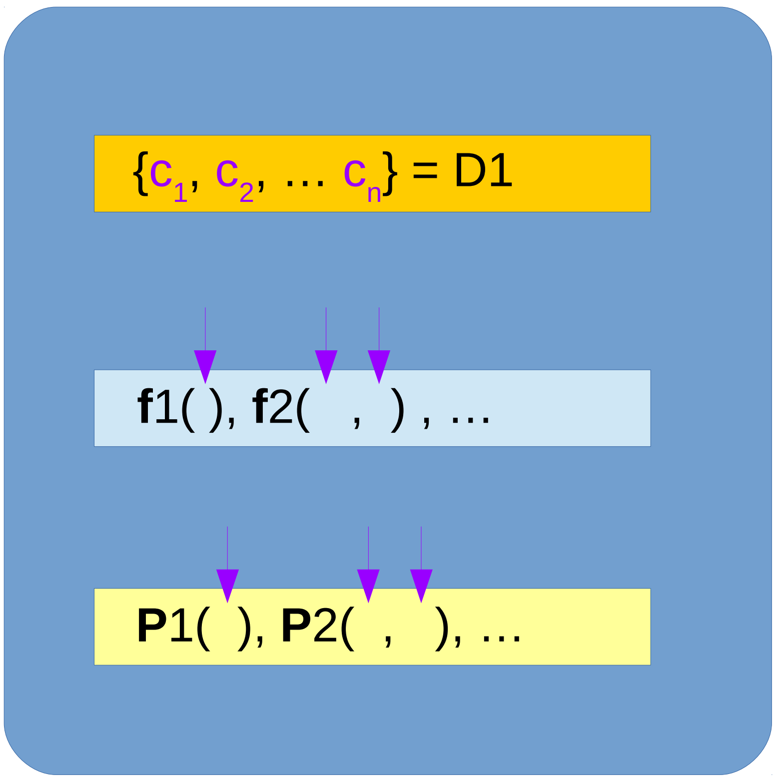


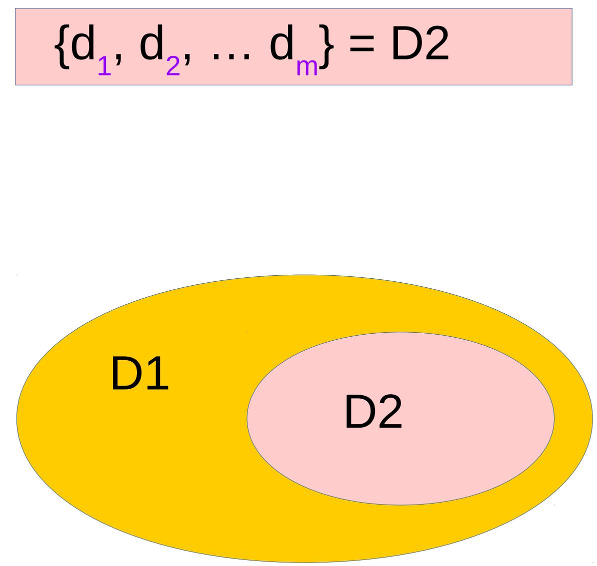
$2^4 = 16$

Models



Models





Truth values of sentences

Propositional Logic

	A	B	
Interpretation I_1 ➡	T	T	
Interpretation I_2 ➡	T	F	
Interpretation I_3 ➡	F	T	
Interpretation I_4 ➡	F	F	

First Order Logic

Sentences

	P1( )	P2( )	...	S1	S2
Interpretation I_1 ➡	T	T			
Interpretation I_2 ➡	T	F			
Interpretation I_3 ➡	F	T			
Interpretation I_4 ➡	F	F			

terms $\quad x \quad y \quad f(x) \quad g(x, y)$

atomic formulas $\quad P(x) \quad Q(x, y)$

formulas / sentences $\quad \forall x, y \ (P(x) \wedge \neg Q(x, y))$

Model Theory

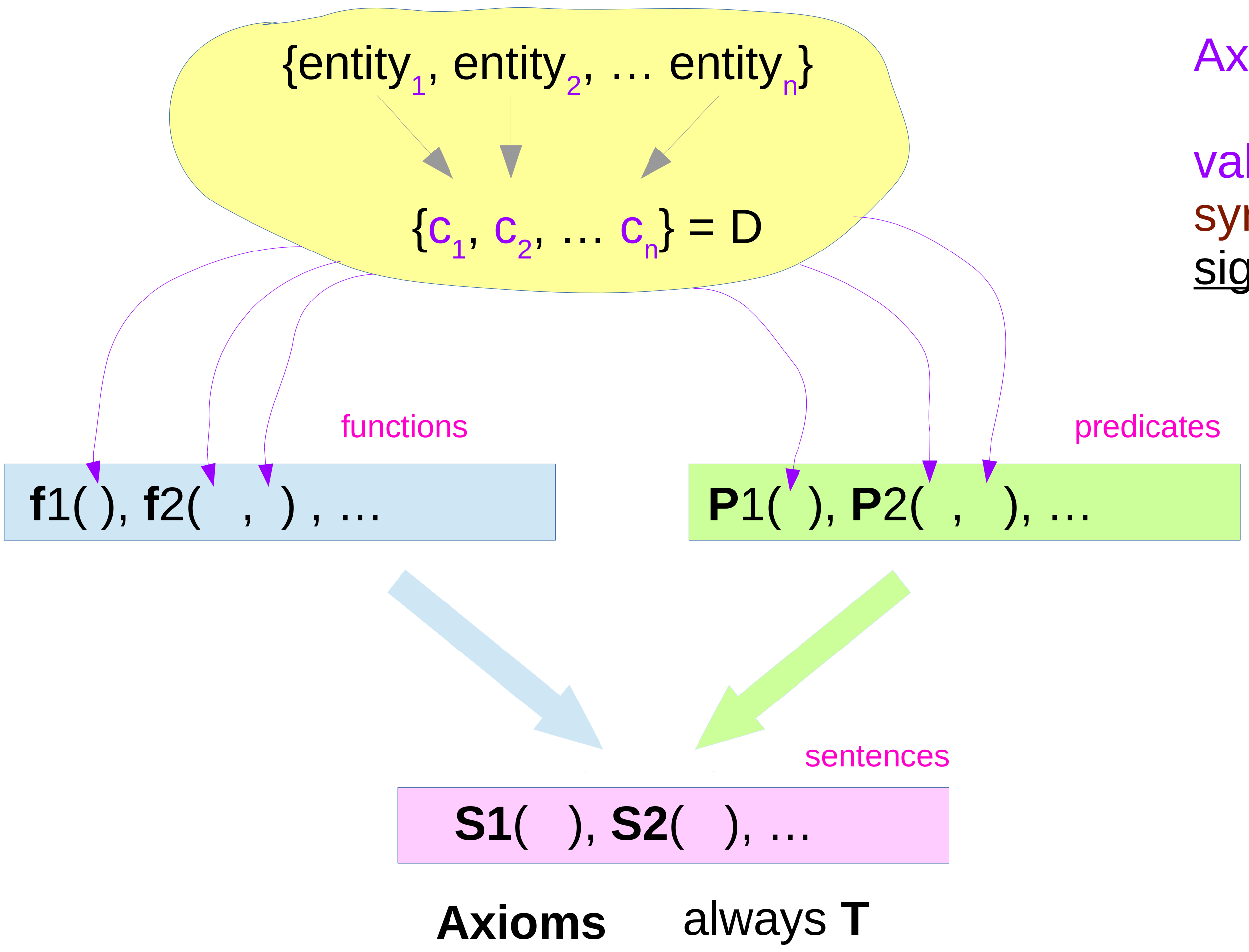
A first-order theory of a particular signature is
a set of axioms,
which are sentences
consisting of symbols from that signature.

The set of axioms is
often finite or
recursively enumerable,
in which case the theory is called effective.

Sometimes theories often include
all logical consequences of the axioms.

https://en.wikipedia.org/wiki/First-order_logic#First-order_theories.2C_models.2C_and_elementary_classes

Axioms of a model theory



Axioms

valid sentences consisting of symbols from a particular signature.

Models

Propositional Logic

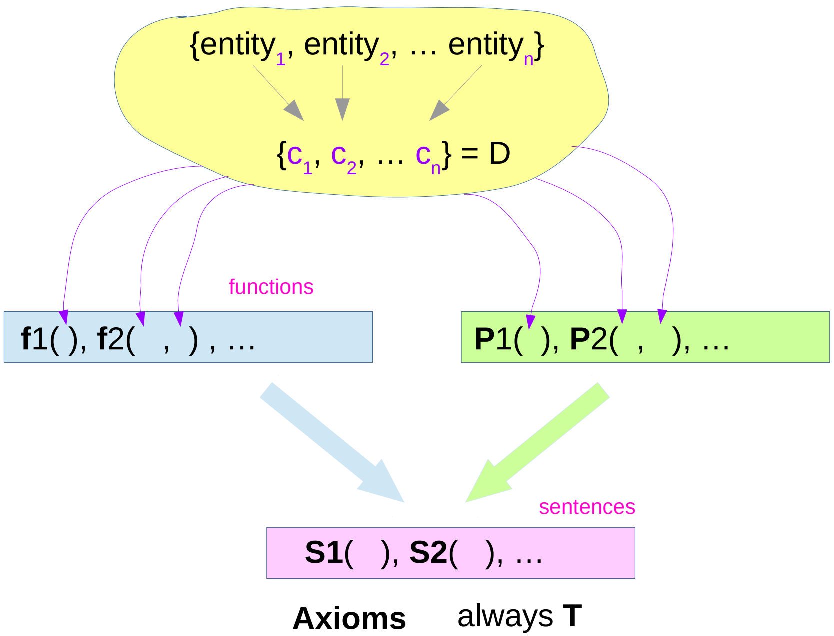
	A	B					
Interpretation I_1 ➡	T	T					
Interpretation I_2 ➡	T	F	T	T	T	➡	a model
Interpretation I_3 ➡	F	T	T	T	T	➡	a model
Interpretation I_4 ➡	F	F					

First Order Logic

Sentences

	P1( ) …	P2( ) …	S1	S2 …		
Interpretation I_1 ➡	T	T				
Interpretation I_2 ➡	T	F				
Interpretation I_3 ➡	F	T	T	T	➡	a model
Interpretation I_4 ➡	F	F				
			T	T	➡	a model
			T	T	➡	a model

Signature



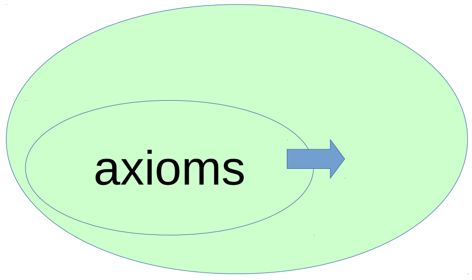
Logical Axioms

formulas in a formal language that are universally valid
formulas that are satisfied by every assignment of values (interpretations)

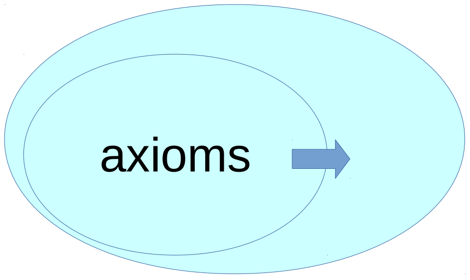
usually one takes as **logical axioms**
at least some minimal set of tautologies
that is sufficient for proving all tautologies in the language

in the case of predicate logic *more* **logical axioms** than that are required,
in order to prove logical truths that are not tautologies in the strict sense.

tautology



valid formulas



https://en.wikipedia.org/wiki/Axiom

Non-logical Axioms

formulas that play the role of theory-specific assumptions

reasoning about two different structures,
for example the natural numbers and the integers,
may involve the same **logical axioms**;

the purpose is to find out
what is special about *a particular structure*
(or set of structures, such as groups).

Thus non-logical axioms are not tautologies.

https://en.wikipedia.org/wiki/Axiom

Mathematical Discourse

Also called

- **postulate**
- **axioms in mathematical discourse**

this does not mean that it is claimed
that they are true in some absolute sense

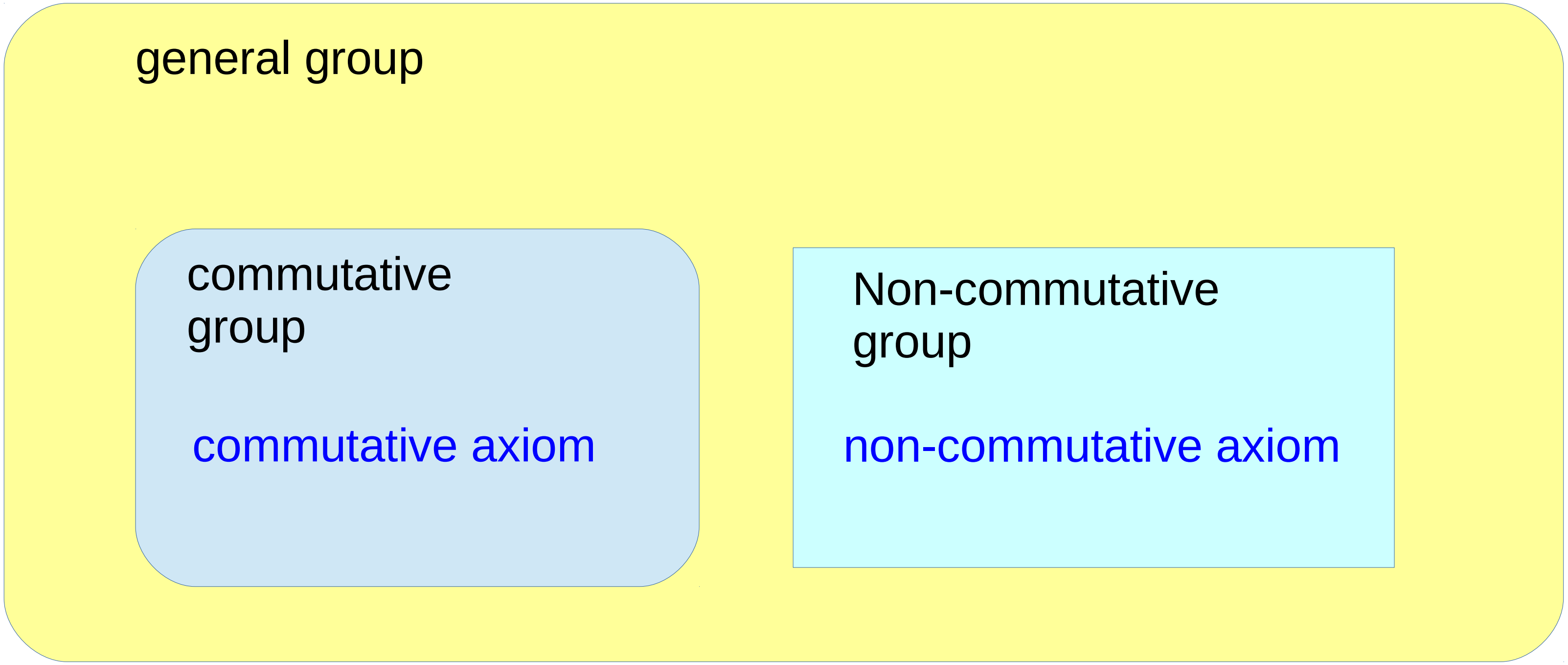
an elementary basis for a formal logic system

A deductive system

- **axioms** (non-logical)
- **rules of inference**

https://en.wikipedia.org/wiki/Axiom

Need not be tautologies



this does not mean that it is claimed
that they are true in some absolute sense

- Commutative axiom
- Non-commutative axiom

https://en.wikipedia.org/wiki/Axiom

Model Theory

The axioms are considered
to *hold* within the theory and

From axioms other sentences that *hold* within the theory can be derived.

A first-order structure that satisfies **all** sentences in a given theory
is said to be a **model** of the theory.

An elementary class is the set of **all** structures satisfying a particular theory.

These classes are a main subject of study in model theory.

https://en.wikipedia.org/wiki/First-order_logic#First-order_theories.2C_models.2C_and_elementary_classes

Truth values of sentences

Entailment in propositional logic can be computed
By enumerating the possible worlds (i.e. model checking)

How to enumerate possible worlds in FOL?

For each number of domain number n from 1 to infinity
For each k-ary predicate Pk in the vocabulary
For each possible k-ary relation on n objects
For each constant symbol C in the vocabulary
For each choice of referent for C from n objects. ..

Computing entailment in this way is not easy.

https://www.cs.umd.edu/~nau/cmsc421/chapter08.pdf

Model – domain of discourse

1. a nonempty set D of **entities** called a **domain of discourse**
   - this domain is a set
   - each element in the set : entity
   - each constant symbol : one entity in the domain

If we considering all individuals in a class,
The constant symbols might be
‘Mary’, - an entity
‘Fred’, - an entity
‘John’, - an entity
‘Tom’ - an entity

Model – interpretation

2. an **interpretation**

   (a) an entity in D is assigned to each of the constant symbols.

      Normally, every entity is assigned to a constant symbol.

   (b) for each **function**,

      an entity is assigned to each possible input of entities to the **function**

   (c) the predicate '**True**' is always assigned the value T

      The predicate '**False**' is always assigned the value F

   (d) for every other **predicate**,

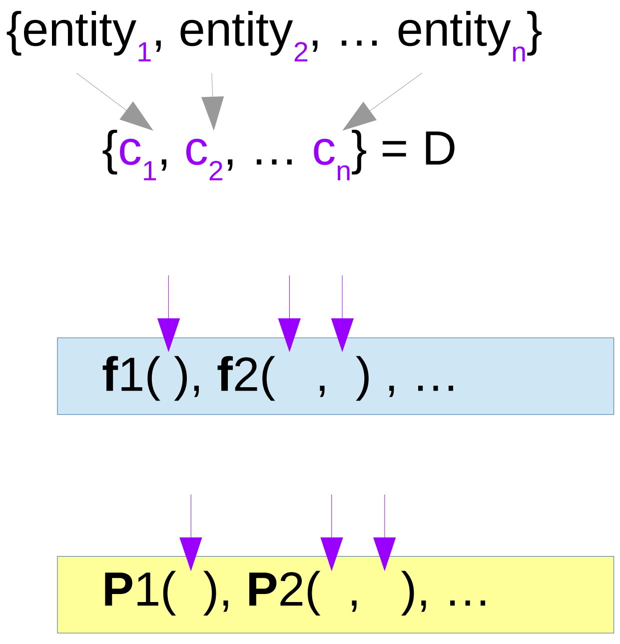
      the value T or F is assigned

      to each possible input of entities to the **predicate**

Each possible input of entities

Arity one: C(n, 1)
Arity two: C(n, 2)
Arity three: C(n, 3)
…

Arity one functions & predicates: C(n, 1)
Arity two: C(n, 2)
Arity three: C(n, 3)
…



always return **T** / **F**

Interpretation

Constant assignments

(a) an entity → the constant symbols.

Function assignments

(b) an entity → each possible input of entities to the **function**

Truth value assignments

(c) the value T →the predicate '**True**'
the value F → the predicate '**False**'

(d) for every other **predicate**,
the value T or F is assigned → every other predicate
to each possible input of entities to the **predicate**

Signature Model Examples A – (1)

Signature

1. constant symbols = { Mary, Fred, Sam }
2. predicate symbols = { married, young }
   married(x, y) : arity two
   young(x) : arity one

Model

1. domain of discourse D : the set of three particular *individuals*
   - this domain is a set
   - each element in the set : entity (= individuals)
   - each constant symbol : one entity in the domain (= one individual)
2. interpretation
   (a) a different individual is assigned to each of the constant symbols

   (a) an entity in D is assigned to each of the constant symbols.
   Normally, every entity is assigned to a constant symbol.

Signature Model Examples A – (2)

(b) for each **function**,
an entity is assigned to each possible input of entities to the **function**

(c) the predicate '**True**' is always assigned the value T
The predicate '**False**' is always assigned the value F

(d) the truth value assignments for every predicate

young(Mary) = F, young(Fred) = F, young(Sam) = T

married(Mary, Mary) = F, married(Mary, Fred) = T, married(Mary, Sam) = F
married(Fred, Mary) = T, married(Fred, Fred) = F, married(Fred, Sam) = F
married(Sam, Mary) = F, married(Sam, Fred) = F, married(Sam, Sam) = F

(d) for every other **predicate**,
the value T or F is assigned
to each possible input of entities to the **predicate**

(Mary, Mary), (Mary, Fred), (Mary, Sam)
(Fred, Mary), (Fred, Fred), (Fred, Sam)
(Sam, Mary), (Sam, Fred), (Sam, Sam)

Signature Model Examples B – (1)

Signature

1. constant symbols = { Fred, Mary, Sam }
2. predicate symbols = { love }   love(x, y) : arity two
3. function symbols = { mother }   mother(x) : arity one

Model

1. domain of discourse D : the set of three particular individuals
2. interpretation
   (a) a different individual is assigned to each of the constant symbols

   (b) the truth value assignments for every predicate
   love(Fred, Fred) = F, love(Fred, Mary) = F, love(Fred, Ann) = F
   love(Mary, Fred) = T, love(Mary, Mary) = F, love(Mary, Ann) = T
   love(Ann, Fred) = T, love(Ann, Mary) = T, love(Ann, Ann) = F

   (c) the function assignments
   mother(Fred) = Mary, mother(Mary) = Ann, mother(Ann) = - (no assignment)

Signature Model Examples B – (2)

2. interpretation

(a) a different individual is assigned to each of the constant symbols

(a) an entity in D is assigned to each of the constant symbols.
Normally, every entity is assigned to a constant symbol.

(b) the truth value assignments

(b) for each **function**,
an entity is assigned to each possible input of entities to the **function**

love(Fred, Fred) = F, love(Fred, Mary) = F, love(Fred, Ann) = F
love(Mary, Fred) = T, love(Mary, Mary) = F, love(Mary, Ann) = T
love(Ann, Fred) = T, love(Ann, Mary) = T, love(Ann, Ann) = F

(c) the function assignments

(d) for every other **predicate**,
the value T or F is assigned
to each possible input of entities to the **predicate**

mother(Fred) = Mary, mother(Mary) = Ann, mother(Ann) = - (no assignment)

The truth value of sentences

The truth values of **all sentences** are assigned :

1. the truth values for **sentences** developed with the symbols **¬, ∧, ∨**, ⇒, ⇔ are assigned as in propositional logic.

2. the truth values for two terms connected by the **=** symbol is **T** if both terms refer to the same entity; otherwise it is **F**

3. the truth values for ∀x p(x) has value **T** if p(x) has value **T** for **every assignment** to x of an entity in the domain D; otherwise it has value **F**

4. the truth values for ∃x p(x) has value **T** if p(x) has value **T** for **at least one assignment** to x of an entity in the domain D; otherwise it has value **F**

5. the operator **precedence** is as follows **¬, =, ∧, ∨**, ⇒, ⇔

6. the **quantifiers** have precedence over the operators

7. **parentheses** change the order of the precedence

Formulas and Sentences

An **formula**
- A **atomic formula**
- The operator $\neg$ followed by a **formula**
- Two formulas separated by $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- A quantifier following by a variable followed by a formula

A **sentence**
- A **formula** with no free variables

$\forall x$ love(x,y)	: free variable y	: not a sentence
$\forall x$ tall(x)	: no free variable	: a sentence

Finding the truth value

Find the truth values of **all sentences**

1. ¬, ∧, ∨, ⇒, ⇔

2. = symbol

3. ∀x p(x)

4. ∃x p(x)

5. the **operator precedence** is as follows ¬, =, ∧, ∨, ⇒, ⇔

6. the **quantifiers** (∀, ∃) have precedence over the **operators**

7. **parentheses** change the order of the precedence

Sentence Examples (1)

Signature

Constant Symbols = {Socrates, Plato, Zeus, Fido}
Predicate Symbols = {human, mortal, legs} all arity one

Model
D: the set of these four particular individuals

Interpretation

(a) a different individual is assigned to each of the constant symbols

(b) the truth value assignment

human(Socrates)=T, human(Plato)=T, human(Zeus)=F, human(Fido)=F
mortal(Socrates)=T, mortal(Plato)=T, mortal(Zeus)=F, mortal(Fido)=T
legs(Socrates)=T, legs(Plato)=T, legs(Zeus)=T, legs(Fido)=T

Sentence Examples (2)

Sentence 1: human(Zeus) ∧ human(Fido) ∨ human(Socrates) = T
F ∧ F ∨ T

Sentence 2: human(Zeus) ∧ (human(Fido) ∨ human(Socrates)) = F
F ∧ (F ∨ T)

Sentence 3: ∀x human(x) = F
human(Zeus)=F, human(Fido)=F

Sentence 4: ∀x mortal(x) = F
mortal(Zeus)=F

Sentence 5: ∀x legs(x) = T
legs(Socrates)=T, legs(Plato)=T, legs(Zeus)=T, legs(Fido)=T

Sentence 6: ∃x human(x) = T
human(Socrates)=T, human(Plato)=T

Sentence 7: ∀x (human(x) ⇒ mortal(x)) = T

Sentence Examples (3)

Sentence 7: $\forall x$ (human(x) $\Rightarrow$ mortal(x)) = T

human(Socrates)=T,	mortal(Socrates)=T,	T $\Rightarrow$ T : T
human(Plato)=T,	mortal(Plato)=T,	T $\Rightarrow$ T : T
human(Zeus)=F,	mortal(Zeus)=F,	F $\Rightarrow$ F : T
human(Fido)=F	mortal(Fido)=T	F $\Rightarrow$ T : T

References

[1] en.wikipedia.org
[2] en.wiktionary.org
[3] U. Endriss, “Lecture Notes : Introduction to Prolog Programming”
[4] http://www.learnprolognow.org/ Learn Prolog Now!
[5] http://www.csupomona.edu/~jrfisher/www/prolog_tutorial
[6] www.cse.unsw.edu.au/~billw/cs9414/notes/prolog/intro.html
[7] www.cse.unsw.edu.au/~billw/dictionaries/prolog/negation.html
[8] http://ilppp.cs.lth.se/, P. Nugues,` An Intro to Lang Processing with Perl and Prolog