First Order Logic – Implication (4A)

Young W. Lim
2/8/17

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

PL: A Model

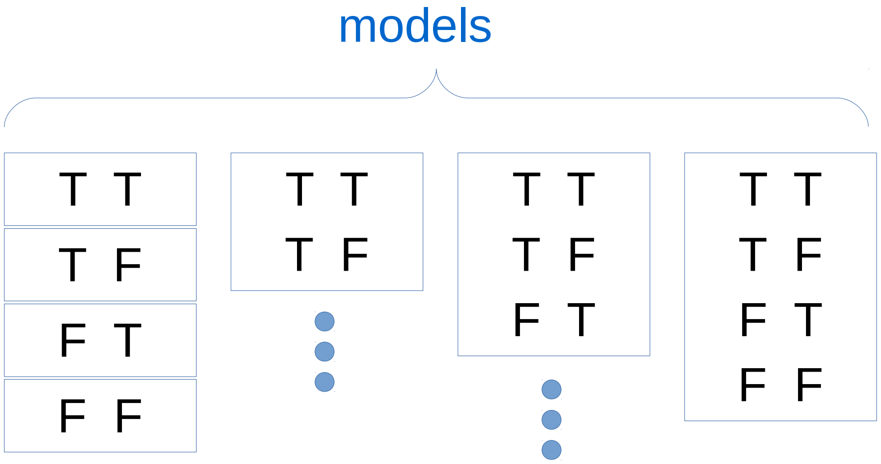
A **model** or a **possible world**:

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 permissiable

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

PL: Interpretation

An **interpretation** of a formal system is
the assignment of meanings to the symbols,
and **truth values** to the **sentences** of a formal system.

The study of interpretations is called formal semantics

Giving an interpretation is synonymous with
constructing a model.

An interpretation is expressed in a metalanguage,
which may itself be a formal language,
and as such itself is a syntactic entity.

https://en.wikipedia.org/wiki/Syntax_(logic)#Syntactic_consequence_within_a_formal_system

PL: Material Implication vs Logical implication

Given two propositions A and B,

If A ⇒ B is a tautology

It is said that A **logically implies** B (A ⭆ B)

Material Implication A ⇒ B (not a tautology)

Logical Implication A ⭆ B (a tautology)

A	B	A⇒B
T	T	T
T	F	F
F	T	T
F	F	T

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

tautology

A∧B ⭆ A

PL: Entailment

if **A**→**B** holds in every model then **A** ⊨ **B**,
and conversely if **A** ⊨ **B** then **A**→**B** is true in every model

any model that makes **A∧B** true

also makes **A** true   **A∧B ⊨ A**

No case : True ⇒ False

A	B	A⇒B
T	T	T
T	F	F
F	T	T
F	F	T

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

Entailment **A∧B ⊨ A**, or **A∧B ⇛ A**

PL: Validity and Soundness (1)

An argument form is **valid** if and only if

whenever the premises are **all true**, then conclusion is **true**.

An argument is valid if its argument form is valid.

If premises : **true**	→	then conclusion : **true**	
false		**true**	
false		**false**	
If true	→	**then <u>never</u>**	**false**

An argument is **sound** if and only if

it is valid and all its premises are true.

Always premises : **true** → **therefore** conclusion : **false**

http://math.stackexchange.com/questions/281208/what-is-the-difference-between-a-sound-argument-and-a-valid-argument

PL: Validity and Soundness (2)

A deductive argument is said to be **valid** if and only if

it takes a form that makes it *impossible*
for the premises to be **true** and the conclusion nevertheless to be **false**.

If	**true** →	**then never**	**false**

Otherwise, a deductive argument is said to be **invalid**.
for the premises to be **true** and the conclusion is **false**.

A deductive argument is **sound** if and only if

it is both **valid**, and all of its premises are **actually true**.

Otherwise, a deductive argument is **unsound**.

Always premises : **true**	→	**therefore** conclusion : **false**

http://www.iep.utm.edu/val-snd/

PL: Validity and Soundness (3)

A	B	A⇒B	A∧(A⇒B)	A∧(A⇒B)⇒B
T	T	T	T	**T**
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

valid

If premises : **true** then <u>never</u> conclusion : **false**

A	B	A⇒B	A∧(A⇒B)	A∧(A⇒B)⇒B
T	T	T	**T**	**T**
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

sound

Always premises : **true therefore** conclusion : **true**

http://www.iep.utm.edu/val-snd/

Formulas and Sentences

An **formula**

- A **atomic formula**
- The operator ¬ followed by a **formula**
- Two formulas separated by ∧, ∨, ⇒, ⇔
- A quantifier following by a variable followed by a formula

A **sentence**

- A **formula** with no free variables

∀x love(x,y)	: free variable y	: not a sentence
∀x tall(x)	: no free variable	: a sentence

Interpretation

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**

Satisfiable

If sentence s has value T under interpretation I:

I satisfies s $\qquad$ **I ⊨ s**

A sentence is **satisfiable**
if there is ***some*** interpretation under which it has value T

A formula that contains ***free variables*** is **satisfied** by an interpretation
if the formula has value T *regardless of* which individuals
from the domain of discourse are assigned to its ***free variables***

Valid

A formula is **valid**
if it is **satisfied** by *every* interpretation

Every **tautology** is a **valid** formula

A **valid** sentence: human(John) $\vee$ $\neg$human(John)

A **valid** sentence: $\exists x$ (human(x) $\vee$ $\neg$human(x))

A **valid** formula: loves(John, y) $\vee$ $\neg$loves(John, y)

True regardless of which individual
in the domain of discourse is assigned to y
This formula is true in every interpretation

Contradiction

A sentence is a **contradiction** if there is **no** interpretation that satisfies it

$\exists x\ (\text{human}(x) \land \neg \text{human}(x))$

not satisfiable under any interpretation

Logical Implication

Given tow formulas A and B, if **A** ⇒ **B** is **valid**:

A logically implies B **A** ⇛ **B**

human(John) ∧ (human(John) ⇒ mortal(John)) ⇛ mortal(John)

valid if it is **satisfied** by *every* interpretation

Logical Equivalence

Given tow formulas A and B, if **A** ⇔ **B** is **valid**:

A is **logically equivalent** **B** $\quad$ **A** ≡ **B**

(human(John) ⇒ mortal(John)) ≡ (¬ human(John) ∨ mortal(John))

valid if it is **satisfied** by *every* interpretation

Some Logical Equivalences

A and B are **variables** representing *arbitrary predicates*
A and B could have other arguments besides x

$\neg\exists x\ A(x) \equiv \forall x\ \neg A(x)$

$\neg\forall x\ A(x) \equiv \exists x\ \neg A(x)$

$\exists x\ (A(x) \lor B(x)) \equiv \exists x\ A(x) \lor \exists x\ B(x)$

$\forall x\ (A(x) \land B(x)) \equiv \forall x\ A(x) \land \forall x\ B(x)$

$\forall x\ A(x) \equiv \forall y\ A(y)$

$\exists x\ A(x) \equiv \exists y\ A(y)$

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