6. Propositional logic: a very simple logic

By: Alex S.

Mar, 2024

1. First-order logic

- Weakly fitted for uncertain knowledge
- Hard to describe any type of heuristics
- Due to amount of sentences, it is hard to describe complex problems

Relations between knowledge

Ontology	Epistemology
Related to the real nature, assumptions about the real world	Related to probable knowledge and states via representation languages
• Propositional logic assumes that there are facts. • Predicate logic assumes that the world is made of **objects** that have *true* or *false* relations. • Special(temporal) logic assumes that the world is ordered by time moments or intervals.	Full confidence that sentence is either *true* or *false*
• Probability theory with stochastic environments.	Order of confidence in interval $[0;1]$
• Theory of confidence? [*Haven't found any reference to the correct term*] - facts exists with the probability that the expert have entailed. • Fuzzy logic - facts exists with some order of truthfulness	

2. Propositional logic syntax

Syntax is made of **alphabet**(defines symbols) + **grammar**(defines how to construct sentences correctly).

1. **Proposition symbols**: big latin alphabet letters, most often starting from P: $P, Q, R, \ldots$, but any other letter can be used as well, e.g.

   $P_{1,3}$ - pit is in the square $(1,3)$
2. **Boolean symbols**: *T* for true and *F* for false.
3. **Logical connectives**:
   - negation(NOT): $\neg$, e.g.

     $\neg P_{1,3}$ - pit is not in the square $(1,3)$
   - conjunction(AND): $\wedge$, &
   - disjunction(OR): $\vee$
   - implication(IF…,THEN…): $\Rightarrow$, $\rightarrow$ (implies)
   - equivalence(IF …, AND ONLY IF …): $\equiv$, $\Leftrightarrow$
4. **Aid symbols**: different types of parentheses $[()]$

Every proposition/boolean symbol is an **atomic** sentence, e.g.

$P_{1,3}$ - pit is in the square $(1,3)$ - is **atomic**

Complex sentences are constructed from simpler sentences(atomics), using parentheses and **logical connectives**, e.g.

$P_{1,3} \vee \neg P_{1,3}$ - pit is or is not in the square $(1,3)$ - is a **complex sentence**

Example of logic sentence notation:

Cyclone(P) *consequences are* ***strong wind***(Q) *and* ***big waves***(R)*, but* ***anticyclone***(S) *consequences are* ***no wind***(T) *at all.*

We can separate certain parts of the sentence into atomic sentences described by one letter in parenthesis and get:

$$(P \rightarrow (Q \wedge R)) \wedge (S \rightarrow T)$$

We can decrease number of parenthesis with **logical connectives relation power**:

1. The biggest relation is for **negation**, exactly by the symbol.
   - $\neg P \wedge Q$ - not P, then and Q
   - $\neg(P \wedge Q)$ - first $P \wedge Q$, then $\neg$
2. **Conjunction and disjunction**:
   - $P \vee Q \vee R$ - is incorrect, order of operations is not defined, need to specify either
   - $(P \vee Q) \vee R$ or $P \vee (Q \vee R)$
3. **Implication** works on the whole expression on the right or left side:
   - $P \rightarrow Q \wedge R$ - is ok, as well as $Q \wedge R \rightarrow P$ - implication works after **AND** operation.
4. **Equivalence** waits until both sides are resolved:
   - $P \wedge Q \rightarrow R \equiv S$ - waits until $P \wedge Q \rightarrow R$ and S are resolved.

3. Semantics

Atomic sentences are easy:

- *True* is true in every model and *False* is false in every model.
- The truth value of every proposition symbol must be specified directly in the model. For example, in the model m, P must be false or true.

For complex sentences, we have five rules, which hold for any subsentences P and Q in any model m (here “iff” means “if and only if”):

- $\neg P$ is true iff P is false in m
- $P \land Q$ is true iff both P and Q are true in m
- $P \lor Q$ is true iff either P or Q is true in m
- $P \to Q$ is true unless P is true and Q is false in m
- $P \equiv Q$ is true iff P and Q are both true or both false in m

The rules can also be expressed with **truth tables**. True is represented by 1 and False by 0:

P	Q	$\neg P$	$P \land Q$	$P \lor Q$	$P \to Q$	$P \equiv Q$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Table 1: Truth table for the five logical connectives

Equivalences:

- $\neg\neg P \equiv P$
- $\neg P \to Q \equiv P \lor Q$
- $P \to Q \equiv \neg P \lor Q$ - resolution law
- $P \to Q \equiv \neg Q \to \neg P$
    - We can prove that the equivalence is correct by using **truth tables**:

P	Q	$P \to Q$
0	0	1
0	1	1
1	0	0
1	1	1

$\neg P$	$\neg Q$	$\neg Q \to \neg P$
1	1	1
1	0	1
0	1	0
0	0	1

- De Morgan’s law:
    - $\neg(P \lor Q) \equiv \neg P \land \neg Q$
    - $\neg(P \land Q) \equiv \neg P \lor \neg Q$
- Commutative property:
    - $P \land Q \equiv Q \land P$
    - $P \lor Q \equiv Q \lor P$
- Associative property:
    - $(P \land Q) \land R \equiv P \land (Q \land R)$
- Distributive property:
    - $P \lor (Q \land R) \equiv (P \lor Q) \land (P \lor R)$
    - $P \land (Q \lor R) \equiv (P \land Q) \lor (P \land R)$

4. Inference procedure

If sentence α is true in model m, we say that m **satisfies** α or m **is a model of** α.

Notation: $M(\alpha)$ - set of all models of α.

Logical entailment between sentences - the idea that a sentence *follows logically* from another sentence:

$$\alpha \vDash \beta$$

where α and β are sentences.

Entailment means that in every model in which α is true, β is also true.

$$\alpha \vDash \beta \text{ if and only if } M(\alpha) \subseteq M(\beta)$$

where $M(\alpha)$ is a subset of $M(\beta)$.

If $\alpha \vDash \beta$, then α is *stronger* assertion than β: it rules out more possible worlds. E.g. the sentence $x = 0$ entails the sentence $xy = 0$. In any model where x is zero, it is the case the xy is zero too.

Inference must be **sound** and it is preferable for it to be **complete**.

Meta-form of the entailment:

$$\frac{\alpha}{\beta} \equiv \alpha \vDash \beta$$

where α is a **premise** - a proposition — a true or false declarative statement, used in an argument to prove the truth of another proposition called the conclusion, β is the **conclusion**.

4.1. MODUS PONENS

$$\frac{\alpha \to \beta, \alpha}{\beta} \equiv \alpha \to \beta, \alpha \vDash \beta$$

where α and $\alpha \to \beta$ are premises, and β is a conclusion.

- In the KB $\alpha \to \beta$ represents sentences IF… THEN…
- α is the parameter that comes from sensors and goes into working memory
- After working memory is loaded a sentence is being searched, so that $\alpha \equiv T$ and $\alpha \to \beta \equiv T$

MODUS PONENS can be proved to be **sound** by using **truth table**. Whenever premises α and $\alpha \to \beta$ are *True*, and the sentence β is also *True* in every case, then it can be inferred and procedure is **sound**.

α	β	$\alpha \to \beta$
0	0	1
0	1	1
1	0	0
1	**1**	**1**

Table 2: Truth table for the MODUS PONENS

Since premises $\alpha \equiv T$ and $\alpha \to \beta \equiv T$ holds *True*, and conclusion $\beta \equiv T$, then it concludes that the law is **sound**.

Example:

- $\alpha \to \beta$: $(\text{WumpusAhead} \land \text{WumpusAlive} \to \text{Shoot})$
- α: $(\text{WumpusAhead} \land \text{WumpusAlive})$
- Therefore with MODUS PONENS we can infer that $\beta \equiv \text{Shoot}$

AND-elimination

- Allows to infer separate conjuncts, that are guaranteed to be *True*:

$$\frac{\alpha_1 \land \alpha_2 \land \ldots \land \alpha_n}{\alpha_1, \alpha_2, \ldots, \alpha_n}$$

Example:

- $\alpha_1 \land \alpha_2$: $(\text{WumpusAhead} \land \text{WumpusAlive})$
- We can infer that $\alpha_1 \equiv \text{WumpusAhead}$ and $\alpha_2 \equiv \text{WumpusAlive}$

AND-inclusion

$$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \land \alpha_2 \land \ldots \land \alpha_n}$$

OR-inclusion

$$\frac{\alpha}{\alpha \lor \beta \lor \gamma \lor \ldots}$$

Double negation

$$\frac{\neg\neg\alpha}{\alpha}$$

4.2. Unit resolution

$$\frac{\alpha \lor \beta, \neg\beta}{\alpha}$$

- can also be called as a **proof by contradiction**

Example:

- If there's a pit(P) in one of $(1,1), (2,2), (3,1)$ and it's not in $(2,2)$, then it's in $(1,1)$ or $(3,1)$.
- Symbolically:

$$\frac{P_{1,1} \lor P_{2,2} \lor P_{3,1}, \neg P_{2,2}}{P_{1,1} \lor P_{3,1}}$$

4.3. Full resolution

- Comes from the unit resolution as

$$\frac{\alpha \lor \beta, \neg\beta \lor \gamma}{\alpha \lor \gamma} \equiv \frac{\neg\alpha \to \beta, \beta \to \gamma}{\neg\alpha \to \gamma}$$

Example:

$$\frac{P_{1,1} \lor P_{3,1}, \neg P_{1,1} \lor \neg P_{2,2}}{P_{3,1} \lor \neg P_{2,2}}$$

α	β	γ	$\alpha \lor \beta$	$\neg\beta \lor \gamma$	$\alpha \lor \gamma$
0	0	0	0	1	0
0	0	1	0	1	1
0	1	0	1	0	0
0	1	1	**1**	**1**	**1**
1	0	0	**1**	**1**	**1**
1	0	1	**1**	**1**	**1**
1	1	0	1	0	1
1	1	1	**1**	**1**	**1**

Table 3: Full resolution soundness proof

In Table 3 there are 4 cases where premises $\alpha \lor \beta$ and $\neg\beta \lor \gamma$ are true. Since all conclusions $\alpha \lor \gamma$ are also true in all those 4 cases, we can conclude that **full resolution** is **sound**.

4.4. MODUS TOLLENS

$$\frac{\alpha \to \beta, \neg\beta}{\neg\alpha}$$

- E.g. there is a breeze in the square(α sentence), then the pit is in the nearest squares(β).
- **MT**: The pit is not in the nearest squares($\neg\alpha$), then the breeze was not perceived($\neg\alpha$).

4.5. Abduction rule

$$\frac{\alpha \to \beta, \beta}{\alpha}$$

As seen in Table 4, the rule is not **sound**, since the second row does not produce a truthful conclusion α.

α	β	$\alpha \to \beta$
0	0	1
0	**1**	**1**
1	0	0
1	**1**	**1**

Table 4: Truth table for the abduction rule

4.6. Equivalence rule

$$\frac{\alpha \equiv \beta}{(\alpha \to \beta) \land (\beta \to \alpha)}$$

$$\frac{(\alpha \to \beta), (\beta \to \alpha)}{\alpha \equiv \beta}$$

5. Conjunctive normal form

Every sentence of propositional logic is logically equivalent to a conjunction of clauses.

A sentence expressed as a conjunction of clauses is said to be in **conjunctive normal form** or **CNF**.

E.g. convert $S_{1,1} \equiv W_{1,2} \lor W_{2,1}$ into **CNF**:

$$S_{1,1} \equiv W_{1,2} \lor W_{2,1}$$

$$(S_{1,1} \to W_{1,2} \lor W_{2,1}) \land (W_{1,2} \lor W_{2,1} \to S_{1,1})$$

$$(\neg S_{1,1} \lor W_{1,2} \lor W_{2,1}) \land (\neg(W_{1,2} \lor W_{2,1}) \lor S_{1,1})$$

$$(\neg S_{1,1} \lor W_{1,2} \lor W_{2,1}) \land ((\neg W_{1,2} \land \neg W_{2,1}) \lor S_{1,1})$$

$$(\neg S_{1,1} \lor W_{1,2} \lor W_{2,1}) \land ((\neg W_{1,2} \lor S_{1,1}) \land (\neg W_{2,1} \lor S_{1,1}))$$

The last line represents **CNF**.

6. Propositional logic weaknesses

- Requires to process large amounts of sentences
- No way to create a generic sentence with variables
- It doesn't allow to efficiently deal with perceptions, with no ability to apply time constraint. E.g. bumping into the wall - cannot limit the agent to do a different action on next time iteration
- Can use only sentences that produce *True* or *False*

7. Predicate logic

- Uses idea that the world is made of objects with relations between them. Relations are either *True* or *False*.
- Has 2 parts:
  1. predicates that define relations
  2. arguments to the predicates, also known as **terms**. Argument types:
     - objects, e.g. **John**
     - relations, e.g. **is a part of**
     - properties, e.g. **is red**
     - functions, e.g. **father(John)**

Predicate logic uses:

- Symbols for defining **constants**($P, A, B, \ldots$)
- Variables ($x, y, \ldots$) for defining constant types
- Functions($f(x, y, \ldots) \to z$) that maps every variable onto function value set. f is called the function constant, $x, y, \ldots$ are the function's term.
- Predicates make statements about objects that are *True* or *False*($p(x, y, \ldots) \to \{T, F\}$), where p is the predicate constant, but $x, y, \ldots$ are the arguments.

With **term** we describe variables, functions and constants.

Complex sentences are mode of atomic sentences:

S is an atomic sentence, as well as

- $\neg S$
- $S \land P$
- $S \lor P$
- $S \to P$
- $S \equiv P$

E.g. complex sentence from atomic ones:
$\text{like}(x, y) \land \text{like}(z, y) \to \neg\ \text{like}(x, z)$

7.1. Quantifiers

$\forall$ - universal quantifier("for all")

$\exists$ - existential quantifier("there exists at least one")

$\exists!$ - there exists only one entity

$\forall x$ - quantifier complex

$\forall x\ p(x)$ - predicate works for all x, e.g.
$\forall x\ \text{like}(x, \text{Mary})$ - everyone likes Mary.

Quantifier domain limits all the variables, e.g.

$\forall x((p(x) \land q(y)) \lor r(x, y))$, where y is a free term.

Examples:

$\forall x(\text{cat}(x) \to \text{animal}(x))$ - every cat is the animal

$\forall x(\text{footballer}(x) \to \text{fast}(x))$ - every footballer is fast

Equivalences:

$\forall x\ p(x) \equiv p(x_1) \land p(x_2) \land \ldots \land p(x_n)$

$\exists x\ p(x) \equiv p(x_1) \lor p(x_2) \lor \ldots \lor p(x_n)$

- Quantifier mutual expressibility

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

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

- Free exchange of variables

$$\exists x\ p(x) \equiv \exists y\ p(y)$$
$$\forall x\ p(x) \equiv \forall y\ p(y)$$

- Quantifiers carry-out from parentheses

$$\forall(p(x) \land q(x)) \equiv \forall x\ p(x) \land \forall x\ q(x)$$
$$\exists(p(x) \land q(x)) \equiv \exists x\ p(x) \lor \exists x\ q(x)$$

7.2. Semantics

- **Sentences** are the same as **statements**
- **Interpretation domain** D - prescribes an object from the D domain to each constant:

$$D = \{\text{John}, \text{Paul}, \text{Ringo}, \text{George}\}$$
$$\text{John} = \text{const}$$

- Each **variable** has a non-empty subset of D domain:

$$x - \text{guitarist}\ \{\text{John}, \text{Paul}, \text{George}\} \subseteq D$$

- Each **function** with the volume N is defined with N objects(arguments) from D domain and the function defines a mapping:

$$f : D^N \rightarrow D$$

- Each **predicate** with the volume M defines M arguments from D domain and defines the mapping:

$$p : D^M \rightarrow \{T, F\}$$

where T - *True*, F - *False*

7.3. Inference

Inference must be **sound**, i.e. sentences must be truthful in every interpretation, e.g.

$$\forall x(p(x) \lor \neg p(x))$$

is truthful.

7.4. Substitution

In every truthful sentence it is possible to substitute a term, and in the result gather a truthful sentence:

$$\forall x\ p(x) : p(K)$$

where x is variable, K - constant, and $p(K)$ is truthful.

Example:

$$\text{TELL}(\text{KB}, \forall x(\text{student}(x) \rightarrow \text{human}(x)))$$
$$\text{TELL}(\text{KB}, \text{student}(\text{John}))$$

With substitution and *MODUS PONENS* we can infer that John is human:

$$\frac{(\text{student}(x) \rightarrow \text{human}(x)), \text{student}(\text{John})}{\text{human}(\text{John})}$$

Skolem normal form

It is easy to substitute $\forall$, for $\exists$ we should use **Skolem normal form**.

The simplest Skolem form is when $\exists$ stands alone. It is replaced with the constant(c):

$$\exists x\ p(x) \equiv p(c)$$

With universal quantifier we should apply a function $y = f(x)$ that maps every x onto y. E.g. every child has a father, later can be expressed as a function that defines a relation between father and each child:

Example 1:

$$\forall x \exists y\ \text{father}(x, y)$$
$$\forall x\ \text{father}(x, f(x))$$

Example 2:

Every human has brains:

$$\forall x(\text{human}(x) \rightarrow \exists y\ \text{brains}(y) \land \text{belongs}(x, y))$$
$$\forall x(\text{human}(x) \rightarrow \text{brains}(f(x)) \land \text{belongs}(x, f(x)))$$

7.5. Predicates for Wumpus World

- Predicate that defines if one square coordinates are close to the other square:

$$\forall x, y, a, b\ \ \text{close}((x, y), (a, b)) \equiv$$
$$(a, b) \in \{(x+1, y), (x-1, y), (x, y+1), (x, y-1)\}$$

- Predicate that defines that there is only one Wumpus existing on the map:

$$\exists!\ \text{wumpus}(x)$$
$$\exists\ \text{wumpus}(x) \land \forall y\ \text{wumpus}(y) \rightarrow (x = y)$$
$$-\ \text{if}\ \exists!\ \text{is not allowed}$$

- **Diagnostic laws** describe how observed facts are related to the consequences:

  If breeze is detected, then pit is in the closest squares:

$$\forall x(\text{breeze}(x) \rightarrow \exists r(\text{close}(r, x) \land \text{pit}(r)))$$

  where x, r are square coordinate tuples.

  If breeze is not detected:

$$\forall s(\neg\ \text{breeze}(s) \rightarrow \forall r(\text{close}(r, s) \land \neg\ \text{pit}(r))) \equiv$$
$$\forall s(\neg\ \text{breeze}(s) \rightarrow \neg \exists r(\text{close}(r, s) \land \text{pit}(r)))$$

where s, r are square coordinate tuples.

- **Reason-consequence** laws:

  If square r is a pit, then the closest squares s has a breeze:

  $$\forall r(\text{pit}(r) \to (\forall s \ \text{close}(r, s) \to \text{breeze}(s)))$$

- The changes in the world can be described with **result** function:

$$\forall x, s(\text{is}(x, s) \land \text{movable}(x) \to \text{hold}(x, \text{result}(\text{grab}(x, s))))$$

  where x is a movable item, s is the square from which the item was grabbed

- Example of **the axiom of the effect of the action**:

  $$\forall x, s(\neg \ \text{hold}(x, \text{result}(\text{release}(x, s))))$$

  where x is a movable item, s is the starting square where it is allowed to release

- Example of **frame axioms**:

  $$\begin{gathered}\forall a, x, s(\neg \ \text{hold}(x, s) \land (a \neq \text{grab}(x)) \\ \to \neg \ \text{hold}(x, \text{result}(a, s))\end{gathered}$$

  where a is the action, x is a movable item, s is the starting square where it is allowed to release. This predicate describes what happens when *hold* action is not active, i.e. if we do not grab then we do not hold anything.

  $$\begin{gathered}\forall a, x, s(\text{hold}(x, s) \land (a \neq \text{release}(x)) \\ \to \text{hold}(x, \text{result}(a, s))\end{gathered}$$

  This predicate describes that if we do not release, then we certainly should be holding some item.

If we want to describe that some action will be true in later times we can describe it with the following pseudocode:

True later ≡ truthful action that changes state ∨ (predicate that already holds the truth ∧ action that does not make an action false)

E.g.

$$\begin{gathered}\forall a, x, s(\text{hold}(x, \text{result}(a, s)) \equiv \\ \equiv ((a = \text{grab}(x) \land \text{is}(x, s) \land \text{movable}(x)) \lor \\ \lor (\text{hold}(x, s) \land (a \neq \text{release}(x))))\end{gathered}$$

where predicate describes that the result of holding an item is that item x is movable, it is in the square s and we have grabbed it. Or we already hold this item and we did not release it yet.

8. Knowledge building for logical agents

Knowledge Base is created by business specialist and knowledge engineer.

The biggest bottleneck is the knowledge acquisition(elicitation).

An example of this problem: **The bear has very small brains(Winny Puhh), therefore it is stupid**. System cannot inference this sentence unless there is a KB that can sequence to this conclusion.

Let's create KB:

1. Puhh is a bear:

   $\text{bear}(P)$

2. All bears are animals:

   $\forall b \ (\text{bear}(b) \to \text{animal}(b))$

3. All animals are physical objects:

   $\forall a \ (\text{animal}(a) \to \text{object}(a))$

4. All animals have brains:

   $\forall a \ (\text{animal}(a) \to \text{brains}(a))$

5. Brains are the part of the animal:

   $\forall a \ (\text{belongs}(\text{brains}(a), a))$

6. If one physical object belongs to other, then it is also an object:

   $\forall x, y \ (\text{belongs}(x, y) \land \text{object}(y) \to \text{object}(x))$

7. Every object has a size K:

   $\forall x \ (\text{object}(x) \to \exists K \ (\text{size}(x) = K))$

8. A relative size of different objects:

   $\forall v, w\left(\text{relative size}(v, w) \to \frac{\text{size}(v)}{\text{size}(w)}\right)$

9. Puhh's brains relative size:

   $\text{relative size}(\text{brains}(P), \text{brains}(B)) =$ very small

   where B is a typical bear

From this KB we can now infer that Puhh has very small brains, he is a bear, an animal and he has brains.