Logic Puzzle Solver

Sentence

Logic concerns with knowledge representation and reasoning.

Knowledge is represented by **sentences or propositions** in a particular **knowledge representation language**.

A **sentence or proposition** takes on one of two possible values: **True(T)** or **False(F)**.

Examples of propositions:

$1 + 5 = 6$ (T)

Boston is the capital of Rhode Island. (F)

Nonexamples:

Wake up!

Are you tired?

Propositional Logic

Propositional logic is a simple but powerful example of a **language representation**.

The syntax of propositional logic defines the allowable sentences.

The **atomic sentences** consist of a single proposition symbol. Each such symbol stands for a proposition that can be true or false.

P = It is raining.

Q = The ground is wet.

Connectives

Sentences can be combined to form more complex sentences. The five connectives that are used in propositional logic are:

Or ($\lor$)

And ($\land$)

Not ($\neg$)

Conditional(or Implication) ($\rightarrow$)

Biconditional ($\leftrightarrow$)

Examples

P = It is raining.
Q = The ground is wet.

$P \vee Q$ = It is raining or the ground is wet
$P \wedge Q$ = It is raining and the ground is wet
$\neg P$ = It is not raining
$P \rightarrow Q$ = If it's raining, then the ground is wet
$P \leftrightarrow Q$ = It's raining if and only if the ground is wet

Entailment

To reason in this propositional logic, we need to define logical **entailment** between sentences— the idea that a sentence **follows logically** from another sentence.

Let P and Q be sentences. The sentence P **entails** the sentence Q(or P **logically imply** Q) if every assignment of truth values to proposition symbols which makes P true also makes Q true.

In mathematical notation, we use $P \models Q$ to denote entailment.

The relation of entailment is familiar from arithmetic; we are happy with the idea that the sentence $x = 0$ entails the sentence $xy = 0$.

Entailment Example

Example: Let $P = (\neg A \vee B) \Rightarrow C$ and $Q = C$.

Does P |= Q?

No. This model makes P True but Q False.

Does Q |= P?

Yes, since every model which makes Q True(green) also makes P True(red).

A	B	C	P	Q=C
T	T	T	T	T
T	T	F	F	F
T	F	T	T	T
T	F	F	T	F
F	T	T	T	T
F	T	F	F	F
F	F	T	T	T
F	F	F	F	F

Knowledge-based Agent

Logical entailment can be applied to **derive conclusions**—that is, to carry out **logical inference**.

A human or artificial intelligence agent(e.g in games, smart assistants like Siri or Alexa) who is capable of reasoning does so from some prior knowledge.

A **knowledge base** or **KB** is a set of sentences. These sentences are taken to be true and represent some facts about the world.

Knowledge-based Agent

Let KB = $\{S_1, S_2, \ldots, S_n\}$ where each S_i represents a sentence of propositional logic.

Suppose that Q is some query and we wish to prove that this query follows logically from the knowledge base. That is, KB |= Q.

We can use the truth table to carry out this logical reasoning as follows.

- Enumerate all possible assignments for all the symbols in KB and Q.
- If every assignment which satisfies all the sentences in KB also satisfies Q, then our query Q is true and is entailed from the knowledge base.

This method of reasoning is called **model-checking**. It is **brute force** method which checks every possibility.

Checking using the truth table is tedious. We'll use a Python implementation which will do model checking for us.

```
from logic import *

R = Atom("It's raining.")
W = Atom("The ground is wet.")

# some examples
sentence1 = Not(R)            # It is not raining
sentence2 = Or(R, W)          # It's raining or the ground is wet.
sentence3 = And(R, W)         # It's raining and the ground is wet.
sentence4 = Implies(R, W)     # If it's raining, then the ground is wet.
sentence5 = Equiv(R, W)       # It's raining if and only if the ground is wet.
```

We’ll create a knowledge base object. We can “tell” information to the knowledge base or “ask” the knowledge base.

```
from logic import *
R = Atom("It's raining.")
W = Atom("The ground is wet.")

kb = createModelCheckingKB()        # Create the knowledge base
print(kb.tell(Implies(R, W)))       # Prints "I learned something.
print(kb.ask(W))                    # Prints "I don't know."
print(kb.tell(Not(W)))              # Prints "I learned something.
print(kb.ask(R))                    # Prints “No.”
print(kb.ask(Not(R)))               # Prints “Yes.”
```

A popular kind of logic puzzle:

The truth-tellers always tell the truth and liars always lie. Each character is either a truth-teller or a liar. Given a set of sentences spoken by each of the characters, determine whether they are truth-tellers or liars.

For these kinds of puzzle, you are usually given a hint like this:

Anna says, "Bob is a liar if Chad is truth-teller."

A = Anna is a truth-teller.

B = Bob is a truth-teller.

C = Chad is a truth-teller.

What Anna says is true if and only if Anna is a truth-teller. Thus, we should use the biconditional connective. So translating the above to propositional logic:

$$A \leftrightarrow (C \rightarrow \neg B)$$

References

1) Mendelson, Elliot. Introduction of Mathematical Logic. Taylor and Francis Group. 2015.
2) Russell, Stuart and Norvig, Peter. Artificial Intelligence, A Modern Approach. Pearson. 2016.