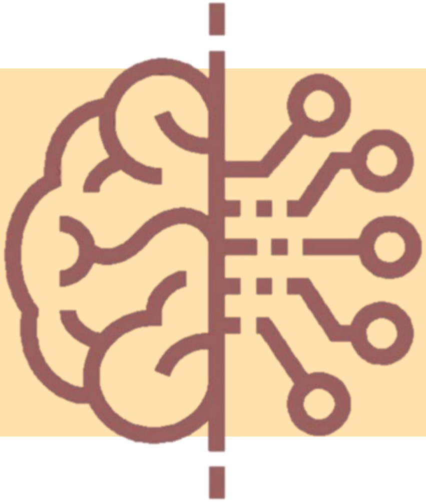
Intelligent Agent

(Knowledge Representation)

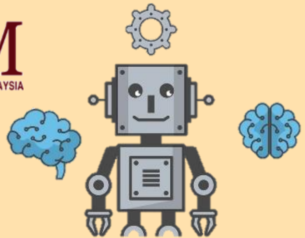


Artificial Intelligence

School of Computing
Universiti Teknologi Malaysia

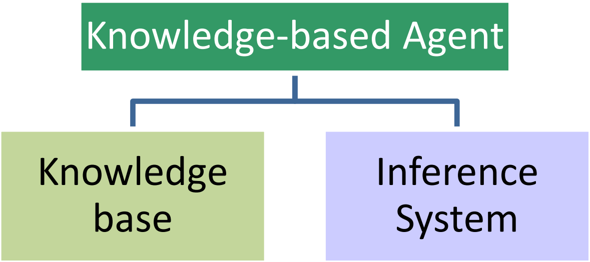
Outline

1. Agents in Artificial Intelligence (in Ch6)
2. Intelligent Agents, PEAS Representation (in Ch6)
3. Knowledge-based Agent
4. Knowledge Representation
5. Knowledge Representation Techniques
6. Inference system



Knowledge-based Agent

- An intelligent agent needs knowledge about the real world for taking decisions and reasoning to act efficiently.
- Knowledge-based agents are those agents who have the capability of maintaining an internal state of knowledge, reason over that knowledge, update their knowledge after observations and take actions. These agents can represent the world with some formal representation and act intelligently.
- Knowledge-based agents are composed of two main parts:
  - **Knowledge-base**
  - **Inference system**



Knowledge-based Agent

Knowledge base

Inference System

Task #1

Open ChatGPT, post this question.

For this case study. Let’s say I want to create a chatbot that can answer questions about different types of fruits. You would need to represent the knowledge about each fruit in a way that the chatbot can understand. This could include information such as the name of the fruit, its color, its taste, and its nutritional value. Can u give example of the Knowledge Base and Inference system.

The chatbot

Given the case study:

- For example, let’s say you want to create a chatbot that can answer questions about different types of fruits. You would need to represent the knowledge about each fruit in a way that the chatbot can understand. This could include information such as the name of the fruit, its color, its taste, and its nutritional value.

Knowledge Base:

- Would include structured information about various fruits. Each fruit is represented with properties like name, color, taste, and nutritional value. Here's an example of a simplified knowledge base:

```
knowledge_base = {
   'Apple': {
      'Color': ['Red', 'Green'],
      'Taste': 'Sweet and slightly tart',
      'Nutritional Value': 'Rich in Vitamin C and dietary fiber'
   },
   'Banana': {
      'Color': 'Yellow',
      'Taste': 'Sweet and creamy',
      'Nutritional Value': 'High in potassium and Vitamin B6'
   },
   'Orange': {
      'Color': 'Orange',
      'Taste': 'Sweet and citrusy',
      'Nutritional Value': 'Excellent source of Vitamin C'
   },
   # Additional fruits and their properties can be added here
}
```

The chatbot

Given the case study:

- For example, let’s say you want to create a chatbot that can answer questions about different types of fruits. You would need to represent the knowledge about each fruit in a way that the chatbot can understand. This could include information such as the name of the fruit, its color, its taste, and its nutritional value.

Inference System:

- Responsible for reasoning over the knowledge base to answer user queries. It uses logic to derive answers based on the information in the knowledge base. Here's a simplified inference system:

```
def answer_question(knowledge_base, fruit, question):
    if fruit in knowledge_base:
        if question == 'Color':
            return f"{fruit}'s color is {knowledge_base[fruit]['Color']}."
        elif question == 'Taste':
            return f"{fruit} tastes {knowledge_base[fruit]['Taste']}."
        elif question == 'Nutritional Value':
            return f"{fruit} is {knowledge_base[fruit]['Nutritional Value']}."

    # If the fruit or question is not found in the knowledge base
    return "I don't have that information."

# Example usage:
user_fruit = 'Apple'
user_question = 'Taste'
response = answer_question(knowledge_base, user_fruit, user_question)
print(response)
```

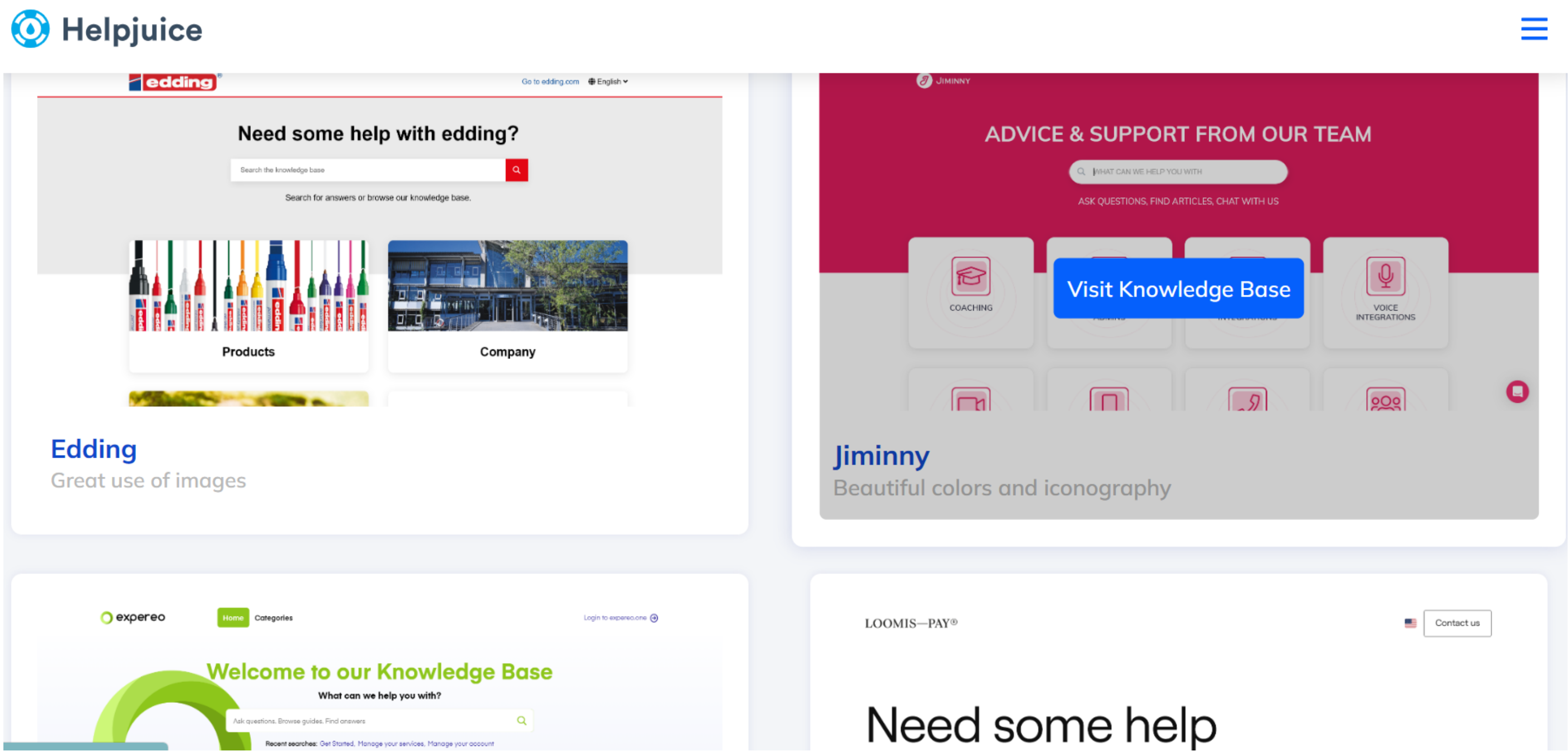
Knowledge-based Agent

A knowledge-based agent must able to do the following:

- An agent should be able to represent states, actions, etc.
- An agent should be able to incorporate new percepts
- An agent can update the internal representation of the world
- An agent can deduce the internal representation of the world
- An agent can deduce appropriate actions

Knowledge-based **Agent:** Example

https://helpjuice.com/knowledge-base-examples



Need some help with edding?

Search for answers or browse our knowledge base.

Edding

Great use of images

ADVICE & SUPPORT FROM OUR TEAM

ASK QUESTIONS, FIND ARTICLES, CHAT WITH US

Jiminny

Beautiful colors and iconography

Welcome to our Knowledge Base

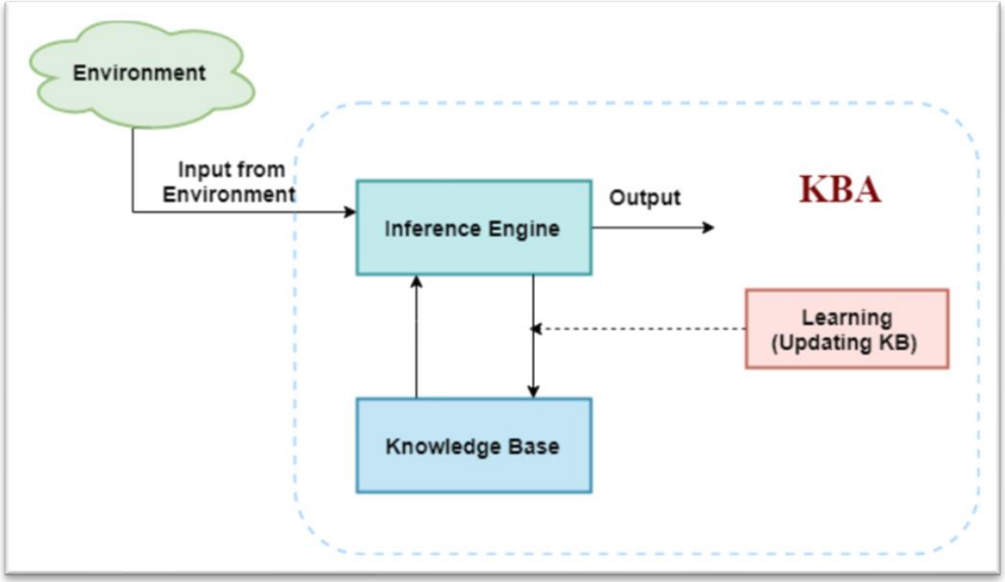
What can we help you with?

Recent searches: Get Started, Manage your services, Manage your account

Need some help

Knowledge-based Agent: The architecture

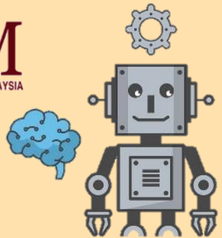
Generalized architecture for a knowledge-based agent



The knowledge-based agent (KBA) take input from the environment by perceiving the environment. The input is taken by the inference engine of the agent and which also communicate with KB to decide as per the knowledge store in KB. The learning element of KBA regularly updates the KB by learning new knowledge.

Knowledge base:

- Knowledge-base is a central component of a knowledge-based agent, it is also known as KB.
- It is a collection of sentences (here 'sentence' is a technical term and it is not identical to sentence in English).
- These sentences are expressed in a language which is called a **knowledge representation language**.
- The Knowledge-base of KBA stores fact about the world.



Knowledge Representation

Knowledge representation is a science that translate real knowledge to computer understandable format. Artificial intelligence represents knowledge not with the help of stored data in the system but with prior experiences and knowledge to act like an intelligent human

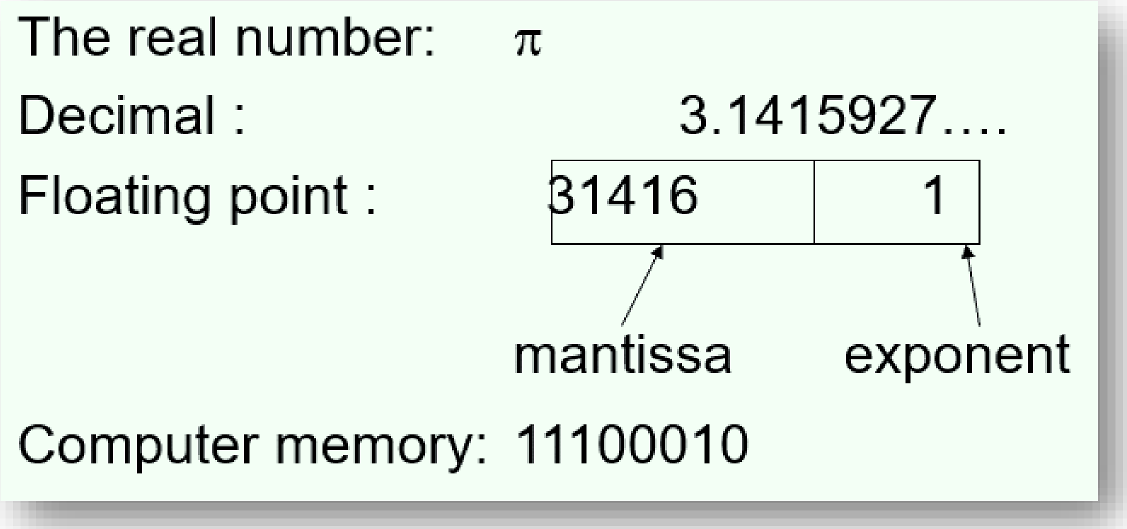
Example: Could a machine (google) answer the question “Who is the prime minister of Malaysia?”

- Background knowledge is necessary?
- How is this knowledge written down, or encoded so a computer can use it?
- How can it be written down efficiently? We can't write everything down.
- What do the formal representations mean? Semantics?

Knowledge Representation

Why need representation?

- Capture important features of a problem
- Make information accessible to problem solving procedure
- Abstraction
- Example:



The real number: π

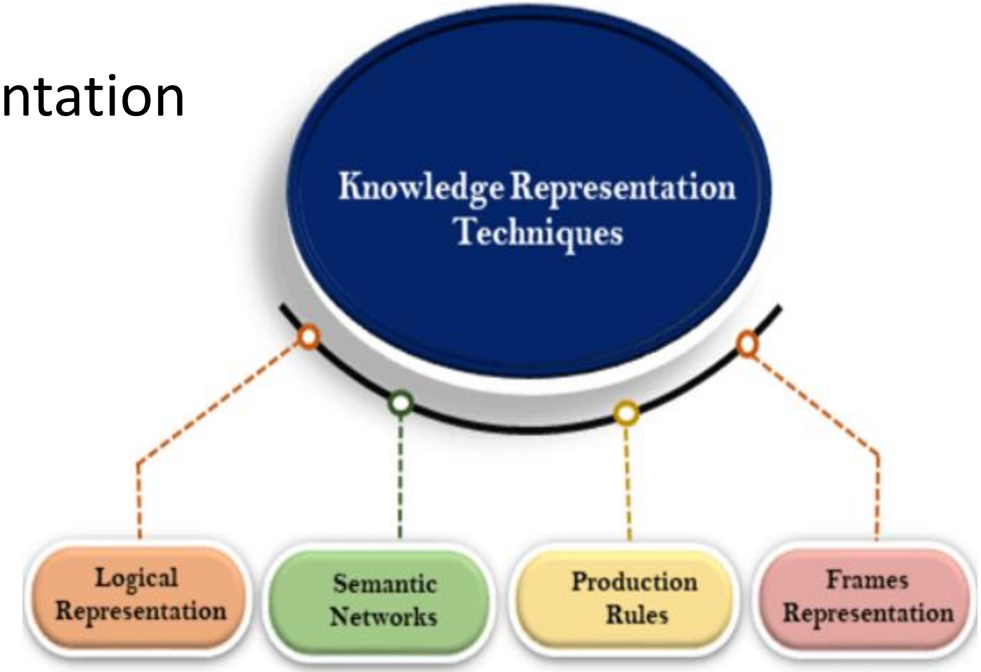
Decimal : 3.1415927….

Floating point : 31416 | 1

Computer memory: 11100010

Knowledge Representation Techniques

- There are mainly four ways of knowledge representation which are given as follows:
    - Logical Representation
    - Semantic Network Representation
    - Frame Representation
    - Production Rules



Knowledge Representation Techniques

Logical Representation

- Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation.
- Logical representation means drawing a conclusion based on various conditions. This representation lays down some important communication rules.
- It consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logics using syntax and semantics.
  - **Syntax**:
    - Syntaxes are the rules which decide how we can construct legal sentences in the logic.
    - It determines which symbol we can use in knowledge representation.
    - How to write those symbols.
  - **Semantics**:
    - Semantics are the rules by which we can interpret the sentence in the logic.
    - Semantic also involves assigning a meaning to each sentence.
- Logical representation can be categorized into mainly two logics:
  - Propositional Logics
  - Predicate logics

Knowledge Representation Techniques

Logical Representation : Propositional (PL)

- Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions.
- A proposition is a declarative statement which is either true or false. It is a technique of knowledge representation in logical and mathematical form.
  - **Example**:

a) It is Sunday. (T/F?)
b) The Sun rises from West. (T/F?)
c) 3+3 = 7 (T/F?)
d) 5 is a prime number. (T/F?)

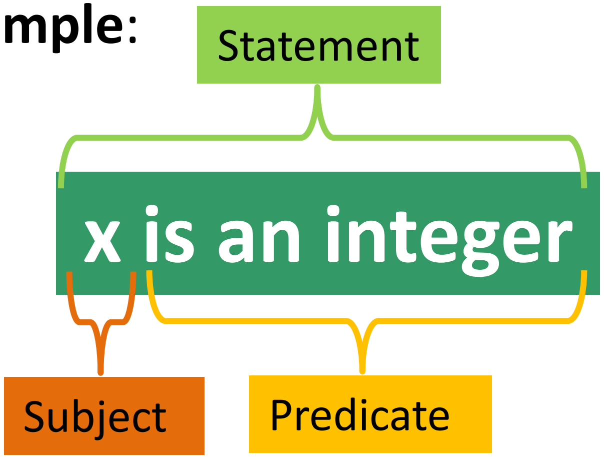
Syntax and Semantics

- **Syntax**
  - Rules for constructing legal sentences in the logic
  - Which symbols we can use (English: letters, punctuation)
  - How we are allowed to combine symbols
- **Semantics**
  - How we interpret (read) sentences in the logic
  - Assigns a meaning to each sentence
- Example: “All lecturers are seven foot tall”
  - A valid sentence (syntax)
  - And we can understand the meaning (semantics)
  - This sentence happens to be false (there is a counterexample)

Knowledge Representation Techniques

Logical Representation : Predicate (FOL)

- Predicate Logic is also known as First-Order Logic (FOL).
- Unfortunately, in propositional logic (PL), we can only represent the facts, which are either true or false.
- PL is not sufficient to represent the complex sentences or natural language statements. The propositional logic has very limited expressive power.
  - **Example**:



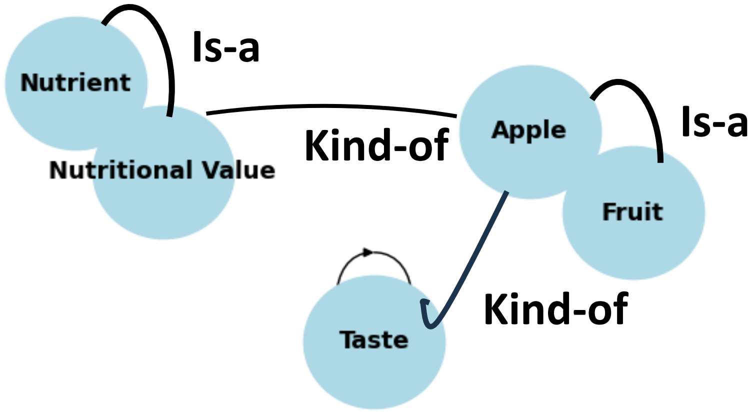
- **Subject**: Subject is the main part of the statement
- **Predicate**: A predicate can be defined as a relation, which binds two atoms together in a statement

Knowledge Representation Techniques

Semantic Network Representation

- This representation consist of mainly two types of relations:
  - IS-A relation (Inheritance)
  - Kind-of-relation

Semantic Network Representation for 'Apple' with IS-A and Kind-of Relations



Example: Following are some statements which we need to represent in the form of nodes and arcs.

Statements:

1. Apple is a fruit.
2. Nutritional value is a nutrient
3. Apple is tasty.
4. Apple has nutritional value.

In the above diagram, we have represented the different type of knowledge in the form of nodes and arcs. Each object is connected with another object by some relation.

Knowledge Representation Techniques

Frame Representation

- A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world.
- Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations.
- It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called facets.
    - The various aspects of a slot is known as Facets. Facets are features of frames which enable us to put constraints on the frames.
    - Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values.
- A frame is also known as slot-filter knowledge representation in artificial intelligence.
- Frames are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not much useful. Frames system consist of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base. The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Knowledge Representation Techniques

Frame Representation : Example

Let's take an example of a frame for a fruit:

Slots	Filters
Name	Apple
Color	[Red, Green]
Taste	Sweet and slightly tart
Nutrition	Vitamin C, Dietary Fiber

Knowledge Representation Techniques

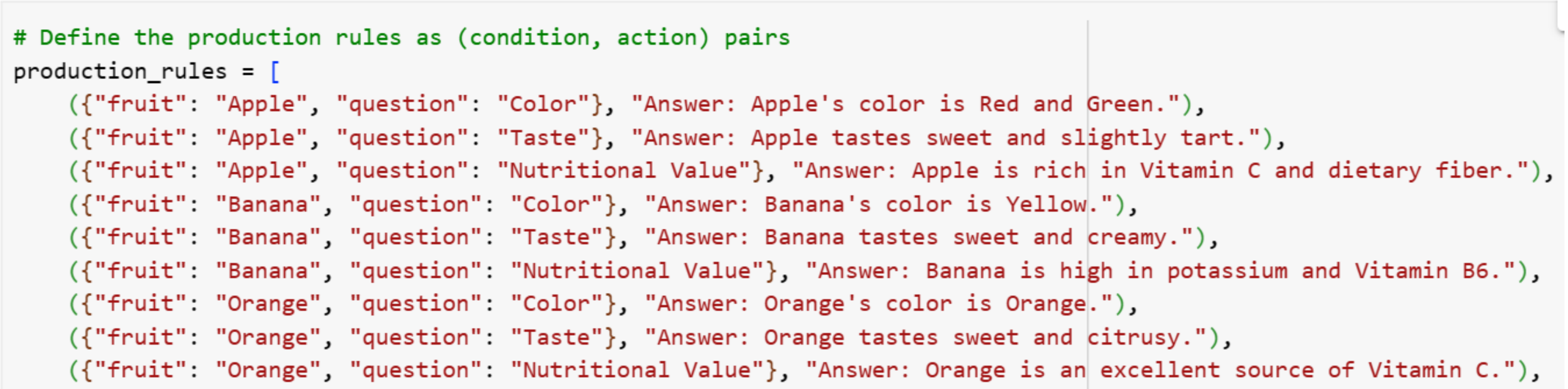
Production Rules

- Production rules system consist of (condition, action) pairs which mean, "If condition then action". It has mainly three parts.
  - ▶ **The set of production rules**
    - In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. And the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle
  - ▶ **Working Memory**
    - The working memory contains the description of the current state of problems-solving and rule can write knowledge to the working memory. This knowledge match and may fire other rules.
  - ▶ **The recognize-act-cycle**
    - If there is a new situation (state) generates, then multiple production rules will be fired together, this is called conflict set. In this situation, the agent needs to select a rule from these sets, and it is called a conflict resolution.

Knowledge Representation Techniques

Production Rules: Example

- Production rules system consist of (condition, action) pairs which mean, "If condition then action“



```
# Define the production rules as (condition, action) pairs
production_rules = [
    ({"fruit": "Apple", "question": "Color"}, "Answer: Apple's color is Red and Green."),
    ({"fruit": "Apple", "question": "Taste"}, "Answer: Apple tastes sweet and slightly tart."),
    ({"fruit": "Apple", "question": "Nutritional Value"}, "Answer: Apple is rich in Vitamin C and dietary fiber."),
    ({"fruit": "Banana", "question": "Color"}, "Answer: Banana's color is Yellow."),
    ({"fruit": "Banana", "question": "Taste"}, "Answer: Banana tastes sweet and creamy."),
    ({"fruit": "Banana", "question": "Nutritional Value"}, "Answer: Banana is high in potassium and Vitamin B6."),
    ({"fruit": "Orange", "question": "Color"}, "Answer: Orange's color is Orange."),
    ({"fruit": "Orange", "question": "Taste"}, "Answer: Orange tastes sweet and citrusy."),
    ({"fruit": "Orange", "question": "Nutritional Value"}, "Answer: Orange is an excellent source of Vitamin C."),
```

Forum activity:
Advantages vs Disadvantages

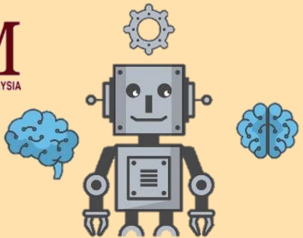
Identify the correct knowledge representation techniques given its advantages and disadvantages

Technique	Advantages	Disadvantages
1. ?	• Normally expressed in natural language. • It is highly modular, so we can easily remove, add or modify an individual rule.	• Does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses. • During the execution of the program, many rules may be active that lead inefficient system.
2. ?	• Makes the programming easier by grouping the related data. • It is comparably flexible and used by many applications in AI. • It is very easy to add slots for new attribute and relations. • It is easy to include default data and to search for missing values. • It is easy to understand and visualize.	• Inference mechanism is not be easily processed. • Inference mechanism cannot be smoothly proceeded by frame representation. • It has a much generalized approach.

Forum Activity:
Advantages vs Disadvantages

Identify the correct knowledge representation techniques given its advantages and disadvantages

Technique	Advantages	Disadvantages
3. ?	• It is a natural representation of knowledge. • Convey meaning in a transparent manner. • Simple and easily understandable.	• Take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible the solution does not exist in this network. • Inadequate representation as they do not have any equivalent quantifier, e.g., for all, for some, none, etc. • Does not have any standard definition for the link names. • Depend on the creator of the system.
4. ?	• Enables us to do logical reasoning. • Basis for the programming languages.	• This representations have some restrictions and are challenging to work with. • This representation technique may not be very natural, and inference may not be so efficient.

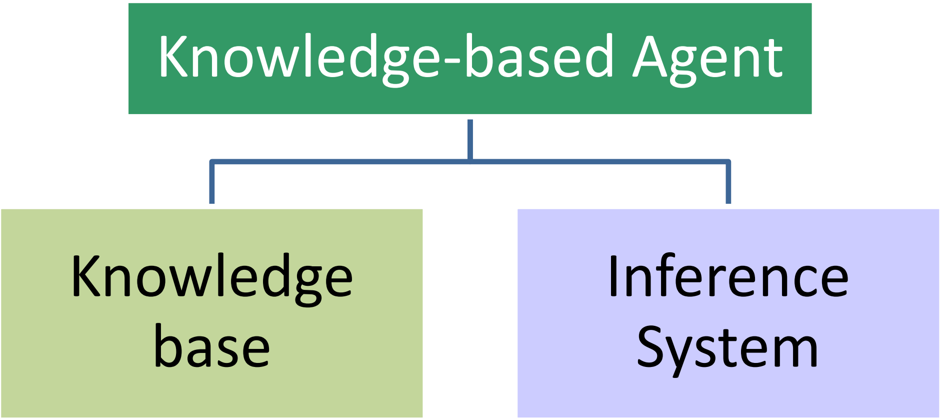


Inference System

Recap

Knowledge-based agents are composed of two main parts:

- ✓ **Knowledge-base**
- ✓ **Inference system**



Knowledge-based Agent

Knowledge base

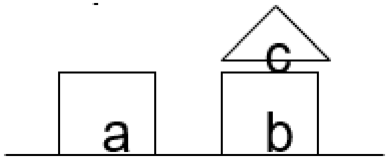
Inference System

Inference System

AI as representation

- A representational scheme should:
    - Be adequate to express all of the necessary information
    - Support efficient execution of the resulting code
    - Provide a natural scheme for expressing the required knowledge

- To support all these needs, AI representation language must:
    - Handle qualitative knowledge-(reasoning)



clear(c),clear(a), ontable(a), ontable(b)

on(c,b), cube(a), cube(b), pyramid(c)

- Predicates : clear, ontable, on, cube, pyramid

Inference System

AI as representation

- Allow new knowledge to be inferred from set of facts and rules

$$\forall X \neg \$Y \; on(X,Y) \Rightarrow clear\ (X)$$

-for all X, X is clear if there does not exist a Y such that Y is on X-

- Allow representation of general principles as well as specific solutions

- Capture complex semantic meaning

hassize(bluebird, small), hascovering(bird,feathers), hascolor(bluebird,blue), hasproperty(bird,flies)

- Allow for meta-level reasoning

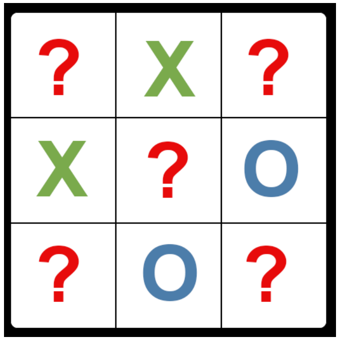
“knowing about what you know” – meta-knowledge

Inference System

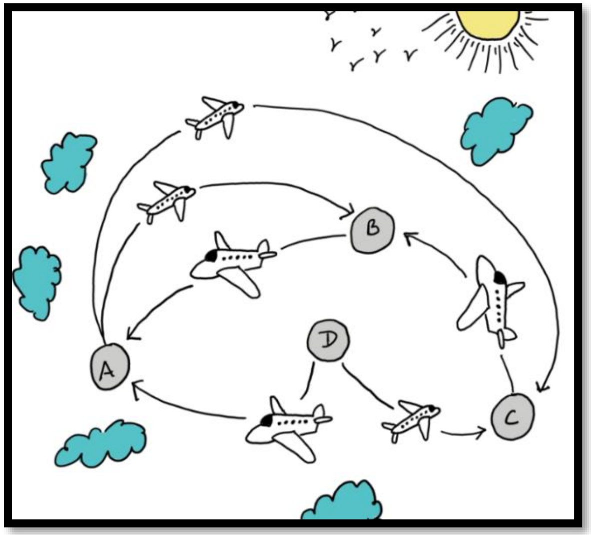
AI as representation and search

Problem solving as search:

Problem are solved by searching among alternative choices, is supported by a commonsense view of human problem solving.



?	X	?
X	?	O
?	O	?



Inference System

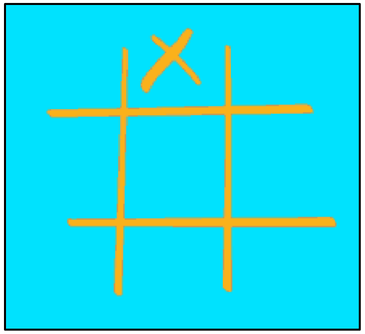
Search

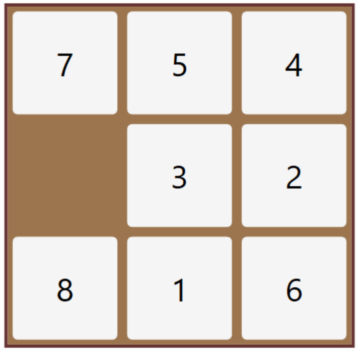
- Often no direct way to solve a problem
- You may know what moves are allowed but not *how* to put the moves into a sequence to solve a problem
- Can generate possibilities for next step and so on
- Considering full search space often too expensive. Too many possibilities (even for computers)

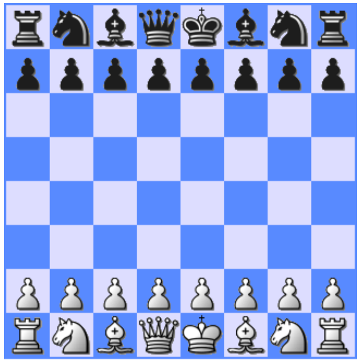
Inference System

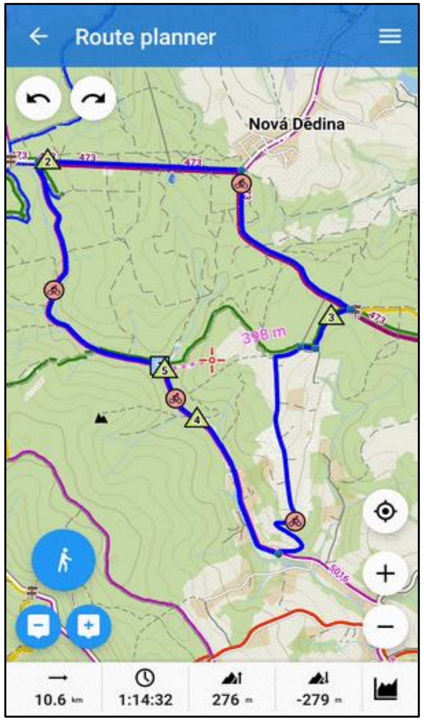
Search Example

- Planning a journey
    - Route finders
    - Timetabling
- Games
    - Tic Tac Toe
    - Chess
- Puzzles



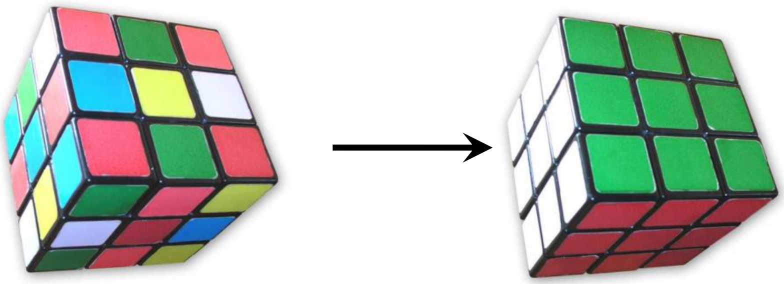


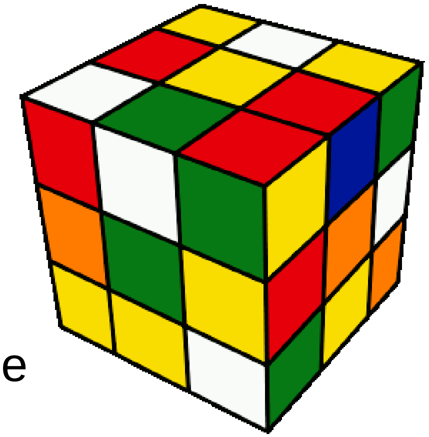




Inference System

Search Example: Rubik's Cube





- 43,252,003,274,489,856,000 combinations

- Up to 481,229,803,398,374,426,442,198,455,156,73 brute-force solution attempts
  - More than 15,259,696,962,150,381 years!!

- Need to look at heuristics or strategies, i.e. selecting the best options to lead to a solution.

Inference System

Reasoning and Inference

- If we know that elephants are mammals with four legs and that Barbar is an elephant, can we conclude that Barbar is a mammal with four legs?
- How do we deal with an elephant (and a mammal) but only has three legs?
- If we use formal *logic* as a knowledge representation language, logical proof can be used to allow us to infer new facts.

Inference System

Intelligent activity

Intelligent activity is achieved through the use of:

1. Symbol or pattern to represent significant aspects of a problem domain
2. Operations on these patterns to generate potential solutions to problems
3. Search to select a solution from among these possibilities

Inference System

AI Programming Languages

- Need a language good for symbolic manipulation (rather than numeric)

- LISP - short for List Processing. A functional programming language. Many AI systems have been written in LISP

- Prolog - Short for programming in logic. Based on a proof method in logic called resolution. Easy to get started and good for symbolic manipulation

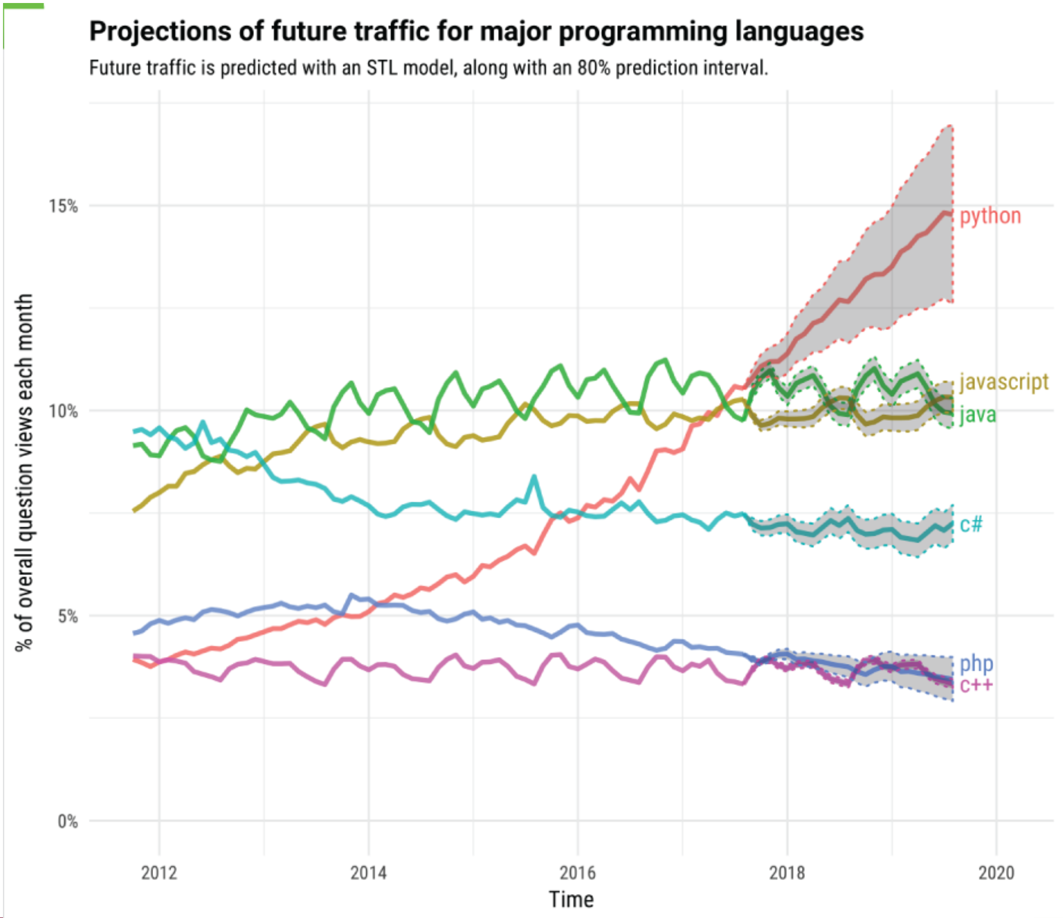
- But AI also can be written in any general-purpose language, such as Python, C, Java. Particularly, as AI programs become components of other larger systems rather than stand alone systems.

AI Programming Languages

- Python is becoming more and more popular among data scientists.

- According to StackOverflow, the popularity of Python is predicted to grow until 2020, at least

- This means it's easier to search for developers and replace team players if required. Also, the cost of their work may be not as high as when using a less popular programming language.



Inference System

AI Programming Languages: Python

Python as the best programming language for AI

1. A great library ecosystem
2. A low entry barrier
3. Flexibility
4. Platform independence
5. Readability
6. Good visualization options
7. Community support
8. Growing popularity

Knowledge Representation

Artificial Intelligence

School of Computing
Universiti Teknologi Malaysia

Premier Digital Tech University™

Outline

We will learn these topics:

1. AI-based knowledge representation
2. Propositional Logic
3. Predicate Logic

2.1 AI-based knowledge representation

Representation

- AI agents deal with knowledge (data)
  - Facts (believe & observe knowledge)
  - Procedures (how to knowledge)
  - Meaning (relate & define knowledge)
- Right representation is crucial
  - Early realisation in AI
  - Wrong choice can lead to project failure
  - Active research area

Choosing a Representation

- For certain problem solving techniques
  - 'Best' representation already known
  - Often a requirement of the technique
  - Or a requirement of the programming language (e.g. Prolog)
- Examples
  - First order theorem proving… first order logic
  - Inductive logic programming… logic programs
  - Neural networks learning… neural networks
- Some general representation schemes
  - Suitable for many different (and new) AI applications

But.. what is knowledge?

- Rules?
  - Production systems
- Classification?
  - Semantic networks
- Structure?
  - Frames
- Idealised and formalised knowledge?
  - Logic

AI as representation

A representational scheme should:

1. Be adequate to express all of the necessary information
2. Support efficient execution of the resulting code
3. Provide a natural scheme for expressing the required knowledge

What knowledge to be represented?

- **Object**
  There are many objects present in the human world. All information we have, related to all the objects, can be considered as a type of knowledge. For instance, a bus has wheels, and a guitar has strings, etc.

- **Events**
  Our understanding of the world is based on the idea we have about the various incidents that have occurred in our world. Thus, events refer to every action that happens in our world.

- **Performance**
  The term is used to explain human behavior or the way they perform certain actions during different situations.

- **Meta-knowledge**
  Knowledge about things we are already aware of.

- **Facts**
  The reality of the actual world and what we stand for.

- **Knowledge base**
  Knowledge Base abbreviated as KB is the most significant part element of knowledge-based agents. It refers to a set of information about any discipline, field, etc. For example, a knowledge base on road construction.

Types of knowledge representation

- **Declarative knowledge**
  It refers to the knowledge that lets us describe our world and it contains everything including ideas, facts, objects, etc… and therefore deals with the description of things.

- **Procedural knowledge**
  Also referred to as imperative knowledge. It refers to more complex concepts (how things work or behave) than declarative knowledge. Therefore, this knowledge is used to complete any task with specific rules, processes, and agendas and thus makes the system work efficiently. Also, this kind of knowledge relies on the task we are trying to finish.

- **Meta-knowledge**
  Meta-knowledge, as mentioned above, is a set of cognitive knowledge when combined. So, this is knowledge about different types of knowledge.

- **Heuristic knowledge**
  Heuristic knowledge is the knowledge imparted by experts of particular domains, topics, and areas, which they have acquired after years of experience. This kind of knowledge enables you to take a better approach to specific problems and make decisions.

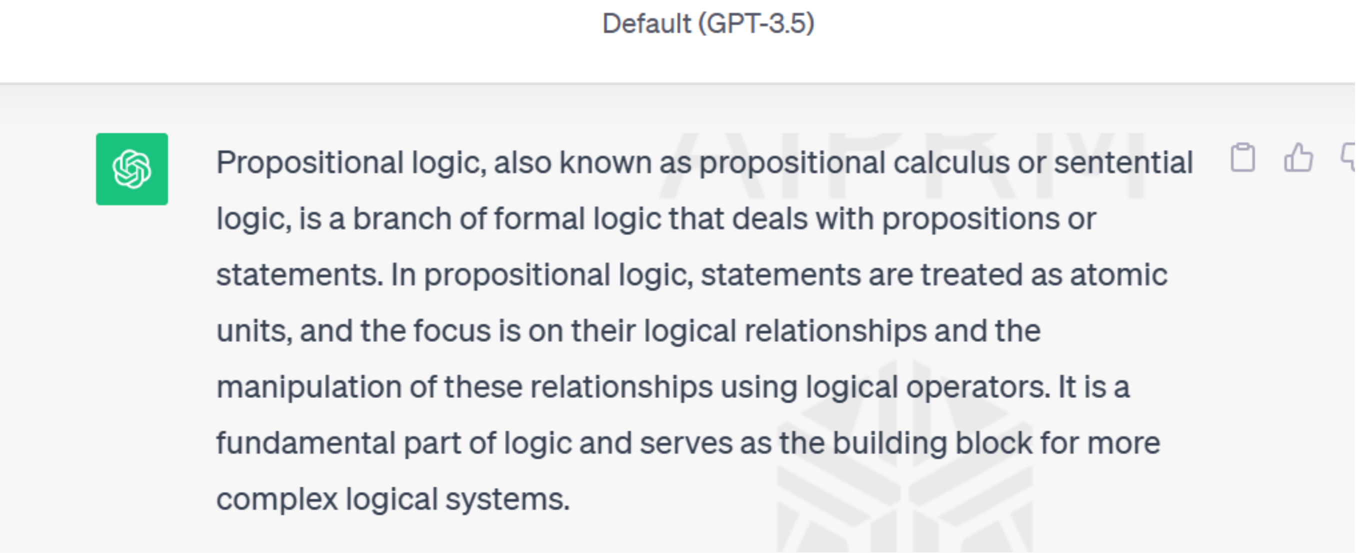
- **Structural knowledge**
  This type of knowledge serves as the basic form of knowledge for solving problems in the real world and helps to establish a connection between ideas or objects and their description.

2.2 Propositional Logic

What is a Logic?

- A language with concrete rules
  - No ambiguity in representation (may be other errors!)
  - Allows unambiguous communication and processing
  - Very unlike natural languages e.g. English
- Many ways to translate between languages
  - A statement can be represented in different logics
  - And perhaps differently in same logic
- **Expressiveness** of a logic
  - How much can we say in this language?
- Not to be confused with logical reasoning
  - Logics are languages, reasoning is a process (may **use** logic)

Ask ChatGPT ..what is propositional logic?



Default (GPT-3.5)

Propositional logic, also known as propositional calculus or sentential logic, is a branch of formal logic that deals with propositions or statements. In propositional logic, statements are treated as atomic units, and the focus is on their logical relationships and the manipulation of these relationships using logical operators. It is a fundamental part of logic and serves as the building block for more complex logical systems.

Basic Logic Operations

- The NOT operation: ¬
- The AND operation: ∧
- The OR operation: ∨
- The “implication” operation: →
- The “equivalence” operation: ↔ or ≡

Logic in General (1)

- *Logics* are formal languages for representing information such that conclusions can be drawn
- When people talk about logic they often mean *propositional* or *first-order predicate* logic.

Logic in General (2)

- A *logic* usually has a well defined syntax, semantics and proof theory.
- The *syntax* of a logic defines the syntactically acceptable objects of the logic, or *well-formed formulae*.
- The *semantics* of a logic associate each formula with a *meaning*.
- The *proof theory* is concerned with manipulating formulae according to certain rules.

Propositional Logic

- The syntax of propositional logic is constructed from *propositions* and *connectives*.
- A **proposition** is a statement that is either true or false but not *both*

Propositional Logic

- Syntax
  - Propositions, e.g. “it is wet”
  - Connectives: and, or, not, implies, iff (equivalent)
  - Brackets, T (true) and F (false)
- Semantics (Classical AKA Boolean) $\wedge \quad \vee \quad \neg \quad \rightarrow \leftrightarrow$
  - Define how connectives affect truth
    - “P and Q” is true if and only if P is true and Q is true
  - Use **truth tables** to work out the truth of statements

How to determine Propositions??

- It is possible to determine whether any given statement is a proposition by prefixing it with

  It is true that . . .

  and seeing whether the result makes grammatical sense.
- Propositions are often abbreviated using *propositional variables* eg p, q, r.
- Thus we must associate the propositional variable with its meaning i.e.

  Let p be *Pak Lah is Prime Minister*.

Criteria to determine Proposition

- **Declarative Form**: A proposition must be in the form of a declarative sentence, which means it makes a statement or assertion. Questions, commands, exclamations, and expressions of emotion are typically not considered propositions. Look for sentences that provide information or assert something.

  Proposition: "The Earth orbits the Sun."

  Not a Proposition: "What time is it?"

  Not a Proposition: "Close the door!"

- **Truth Value**: A proposition must have a definite truth value, which can be either true or false, but not both. Statements that are subjective, vague, or dependent on context may not be considered propositions. Ambiguous statements that could be true or false in different contexts may also not qualify.

  Proposition: "Water boils at 100 degrees Celsius."

  Not a Proposition: "It's a nice day." (The truth of this statement depends on the speaker's perspective.)

  Not a Proposition: "I like pizza." (The truth depends on the individual's preferences.)

- **Clarity**: A proposition should be clear and unambiguous. If a statement is unclear or open to multiple interpretations, it may not be considered a proposition.

  Proposition: "The cat is on the mat."

  Not a Proposition: "It's somewhere around here." (Unclear and lacking specificity.)

Propositions???

- Pak Lah is Prime Minister.
- What's the time?
- 2 + 3 = 5
- ‘Phone’ has five letters.
- 2 + 3 = 6
- Oh dear!
- ‘Work’ has five letters.
- the reactor is on
- *All* elephants have 4 legs
- I like AI class

Connectives

- Propositions may be combined with other propositions to form **compound propositions**. These in turn may be combined into further propositions.
- The connectives that may be used are

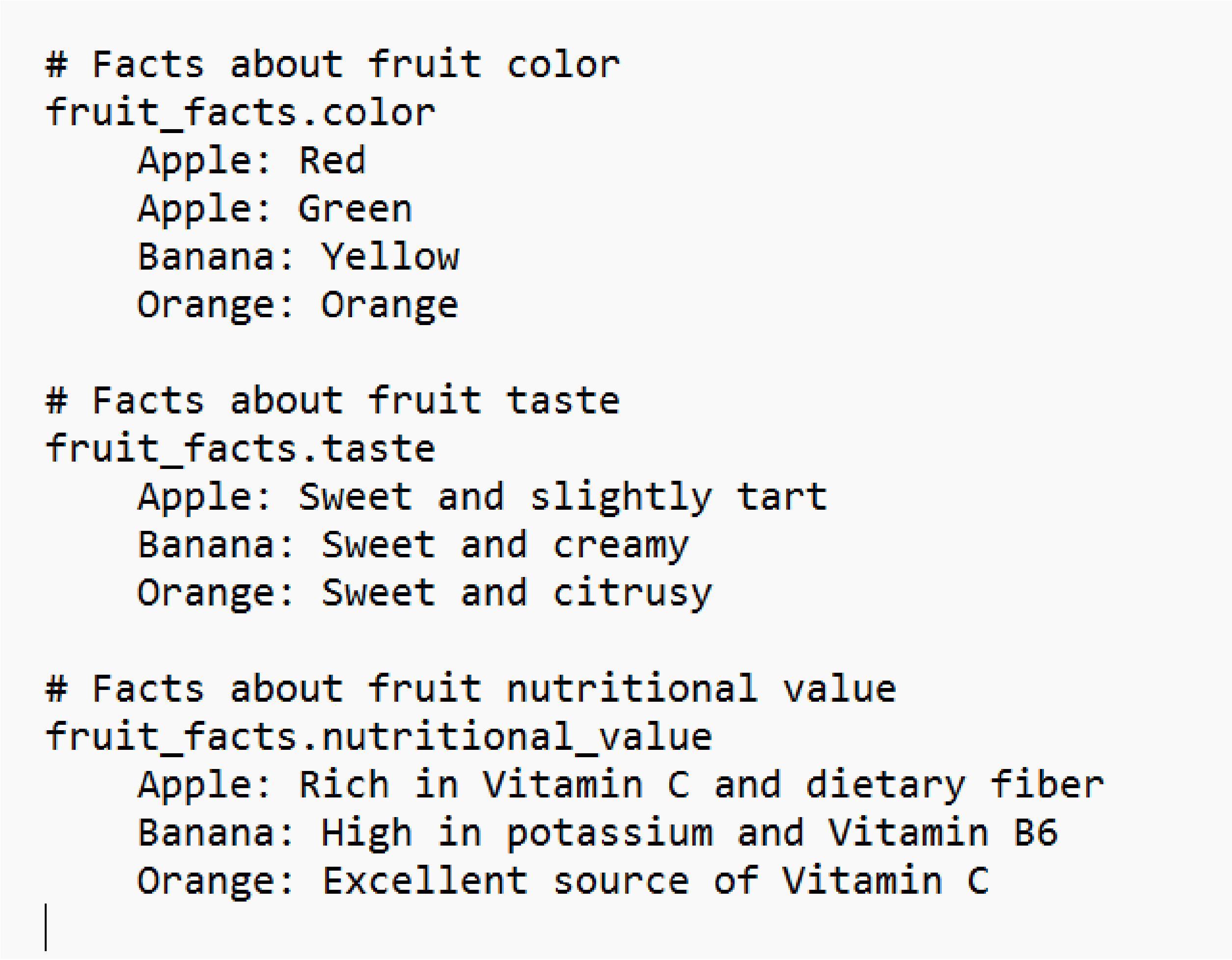
$\wedge$	and	conjunction (& or .)
$\vee$	or	disjunction (| or +)
$\neg$	not	negation (~)
$\Rightarrow$	if . . .then	implication ($\rightarrow$)
$\leftrightarrow$	if and only if	equivalence ($\Leftrightarrow$)

- Some books use different notations. Some of these are given in parentheses.

Propositional Logic: Syntax

- The proposition symbols $P1$, $P2$ etc are sentences
- If S is a sentence, $\neg S$ is a sentence (*negation*)
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (*conjunction*)
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (*disjunction*)
- If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (*implication*)
- If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (*equivalence*)

How to convert all these facts in PL?



```
# Facts about fruit color
fruit_facts.color
    Apple: Red
    Apple: Green
    Banana: Yellow
    Orange: Orange

# Facts about fruit taste
fruit_facts.taste
    Apple: Sweet and slightly tart
    Banana: Sweet and creamy
    Orange: Sweet and citrusy

# Facts about fruit nutritional value
fruit_facts.nutritional_value
    Apple: Rich in Vitamin C and dietary fiber
    Banana: High in potassium and Vitamin B6
    Orange: Excellent source of Vitamin C
```

Propositional Logic: Semantic

Each model species true/false for each proposition symbol

E.g.

P	Q	R
true	*true*	*false*

Why 8?

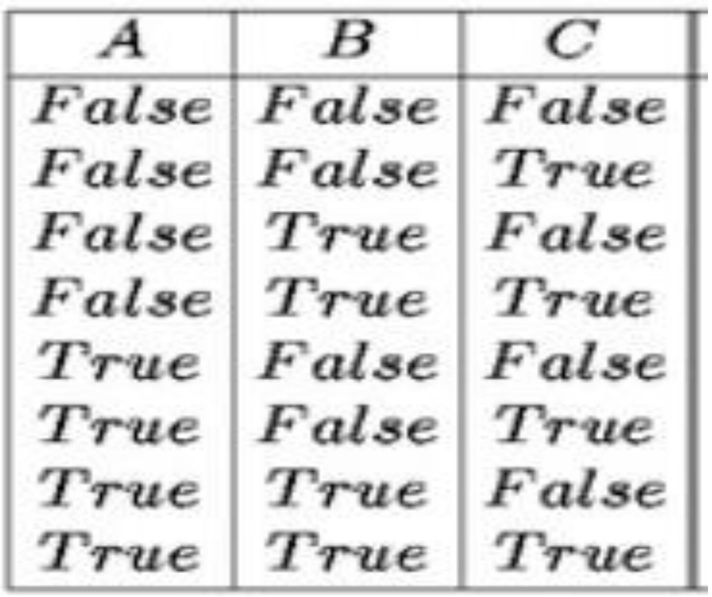
(With these symbols, 8 possible models, can be enumerated automatically.)

Rules for evaluating truth with respect to a model *m* are determined by *truth tables*

Propositional Logic: Semantic

- Why 8 models?

P	Q	R
true	*true*	*false*



A	B	C
$False$	$False$	$False$
$False$	$False$	$True$
$False$	$True$	$False$
$False$	$True$	$True$
$True$	$False$	$False$
$True$	$False$	$True$
$True$	$True$	$False$
$True$	$True$	$True$

How many variables? 3 variables P, Q and R
How many models? 2^3 = 8 models

$\wedge$ And (also called *conjunction*)

- The *conjunction* ‘p *AND* q’, written $p \wedge$. of two propositions is true when both p and q are true, false otherwise.
- We can summarise the operation of $\wedge$ using a Truth table. Rows in the table give all possible setting of the propositions to true (T) or false (F).

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

Natural Language Meaning (1)

p	Its Apple.
q	Its Red.
$p \wedge q$	Its Apple and its red. Its Apple but its red. Its Apple. Its red.

Natural Language Meaning (2)

Caution: semantics?

p	I took a shower
q	I woke up
$p \wedge q$	"I took a shower and I woke up"
$q \wedge p$	"I woke up and I took a shower".

Logically the same! *WE* may see difference.

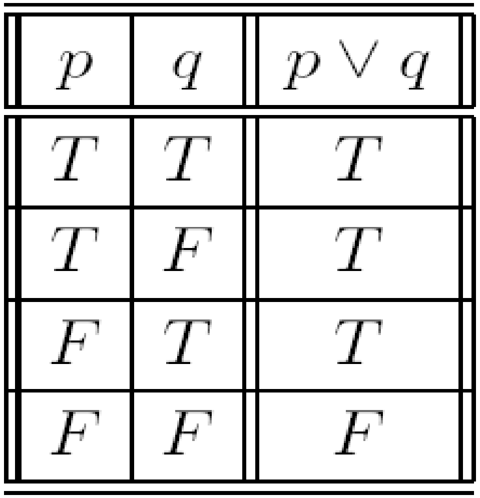
The word *both* is often useful eg. *I both took a shower and I woke up*.

$\vee$ Or

Also called *disjunction.*

The disjunction ‘p OR q’, written $p \vee q$, of two propositions is true when p or q (or both) are true, false otherwise.

Sometimes called *inclusive or*.



p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

Natural Language Meaning

p	It’s Monday.
q	It’s raining.
$p \vee q$	It’s Monday or it is raining.

The word *either* is often useful eg. *either it’s Monday or it is raining*.

It *also* includes the case of rain on a Monday!

$\neg$ Not

Also known as *negation*

- The negation 'NOT p' of a proposition (or $\neg p$) is true when p is false and is false otherwise. $\neg p$ may be read that it is false that p.

p	$\neg p$
T	F
F	T

- Negation is a *unary* connective. It only takes one argument.
- Conjunction and disjunction were both *binary* connectives.

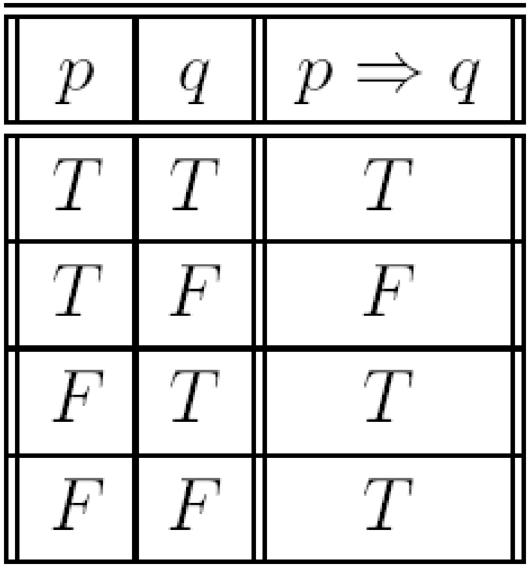
Natural Language Meaning

p	Logic is easy.
$\neg p$	It is false that logic is easy.
	It is not the case that logic is easy.
	Logic is not easy.

$\Rightarrow$ If . . . then . .

Also known as **implication**

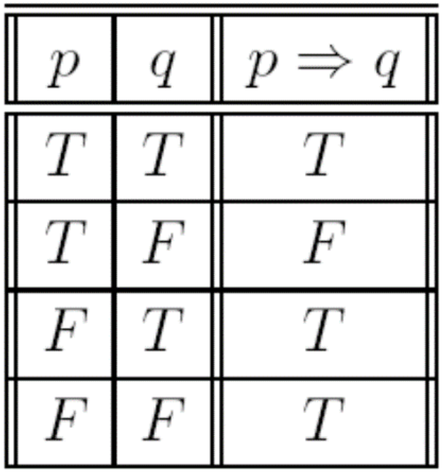
The implication 'p IMPLIES q', written $p \Rightarrow q$, of two propositions is true when either p is false or q is true, and false otherwise. An if…then statement will be false when p is true and q is false, and will be true for all other cases



p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Natural Language Meaning

p	I study hard.
q	I get rich.
$p \Rightarrow q$	If I study hard then I get rich.
	Whenever I study hard, I get rich.
	That I study hard implies I get rich.
	I get rich, if I study hard.



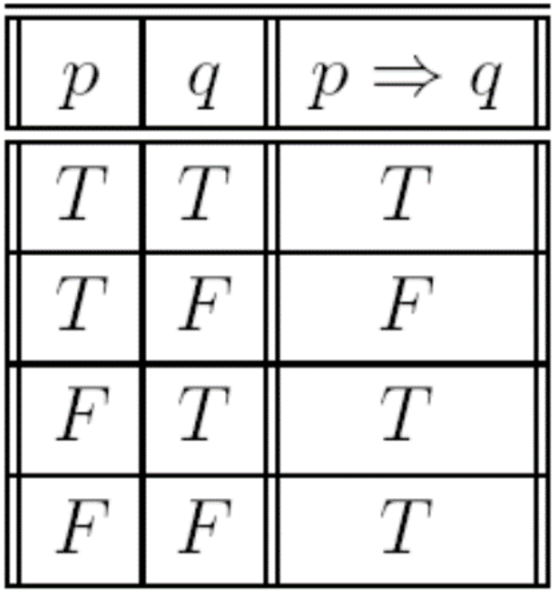
p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

More About Implication

$p \Rightarrow q$ is true in the following situations

- I study hard and I get rich; or
- I don't study hard and I get rich; or
- I don't study hard and I don't get rich.

The last two situations, i.e. when p is false (I don't study hard) we can't say whether I will get rich or not. However if I've studied hard but failed to become rich then the proposition is clearly false.



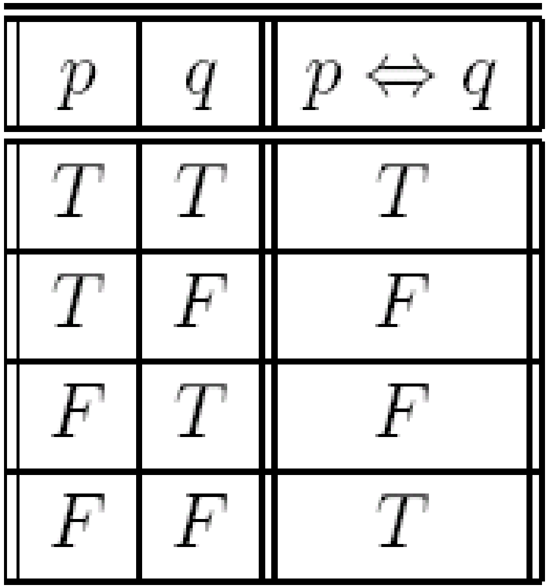
p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Contrapositive

$p \Rightarrow q$	If I study hard then I get rich.
$q \Rightarrow p$	If I get rich then I study hard. (the **converse.**)
$\neg q \Rightarrow \neg p$	If I don't get rich then I don't study hard. (the **contrapositive.**)

⇔ Biconditional

- Also known as iff (if and only if) or the biconditional.
- The biconditional, written as $p \Leftrightarrow q$, of two propositions is true when both p and q are true or when both p and q are false, and false otherwise.



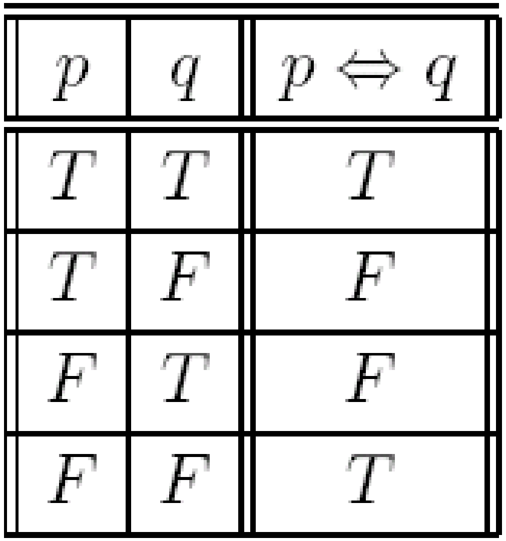
p	q	$p \Leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Biconditional

p A shape is a rectangle.

q A shape has four sides.

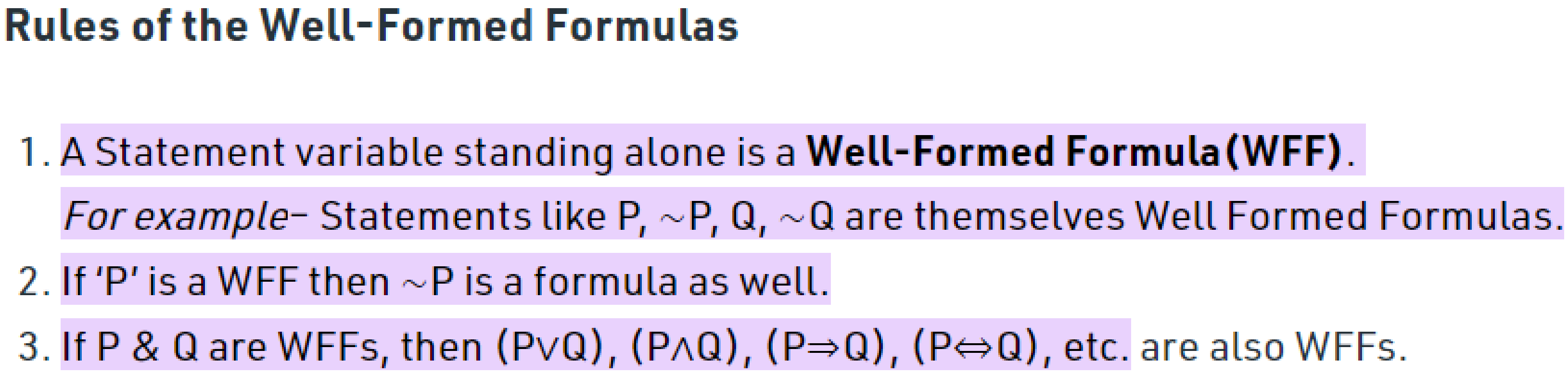
$p \Leftrightarrow q$ A shape is a rectangle if and only if it has four sides.



p	q	$p \Leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Well-Formed Formula (WFF)

Well-Formed Formula(WFF) is an expression consisting of variables (capital letters), parentheses, and connective symbols. An expression is basically a combination of operands & operators and here operands and operators are the connective symbols.

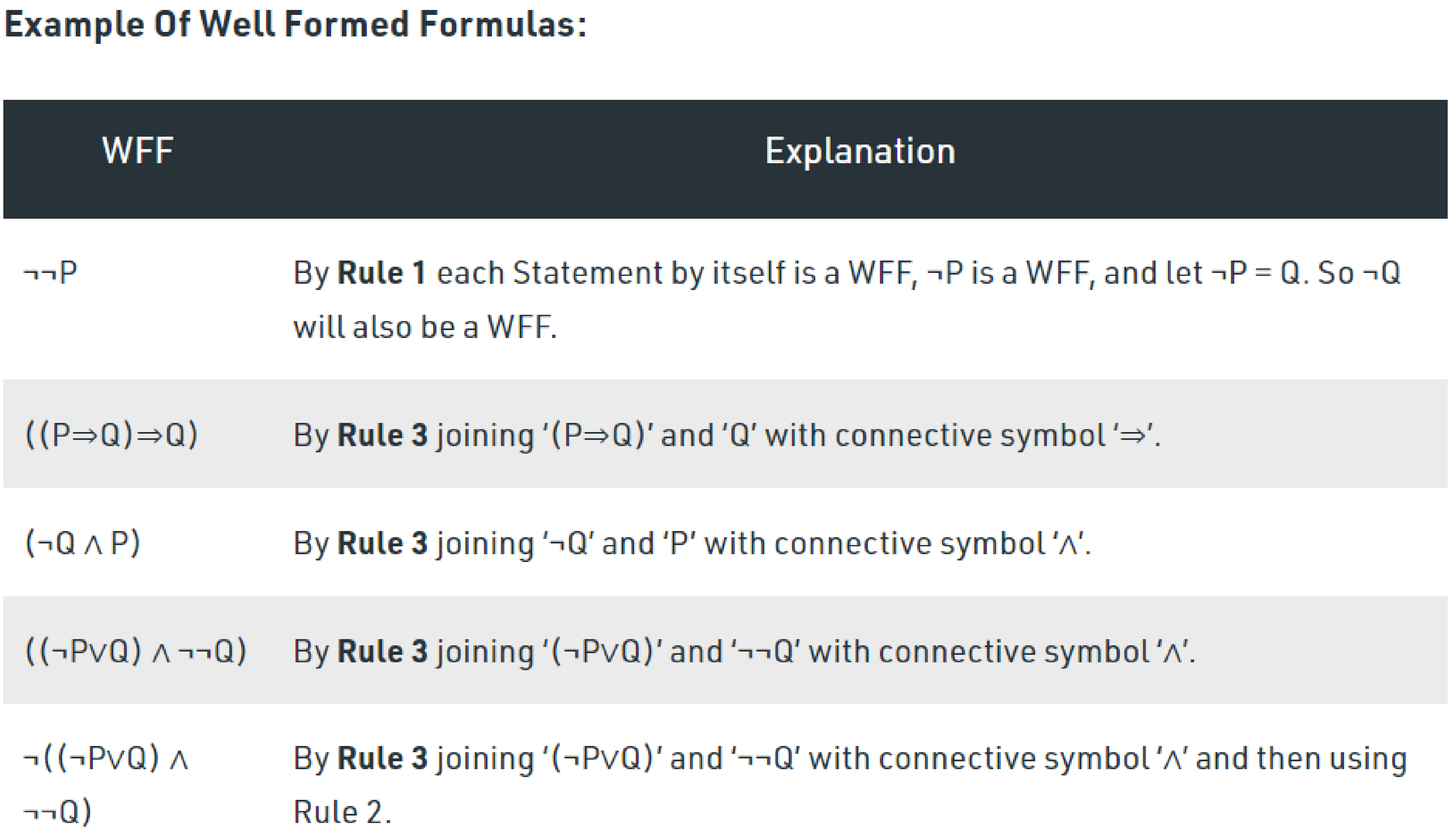


Rules of the Well-Formed Formulas

1. A Statement variable standing alone is a **Well-Formed Formula(WFF)**.

   For example– Statements like P, ~P, Q, ~Q are themselves Well Formed Formulas.
2. If ‘P’ is a WFF then ~P is a formula as well.
3. If P & Q are WFFs, then (P∨Q), (P∧Q), (P⇒Q), (P⇔Q), etc. are also WFFs.

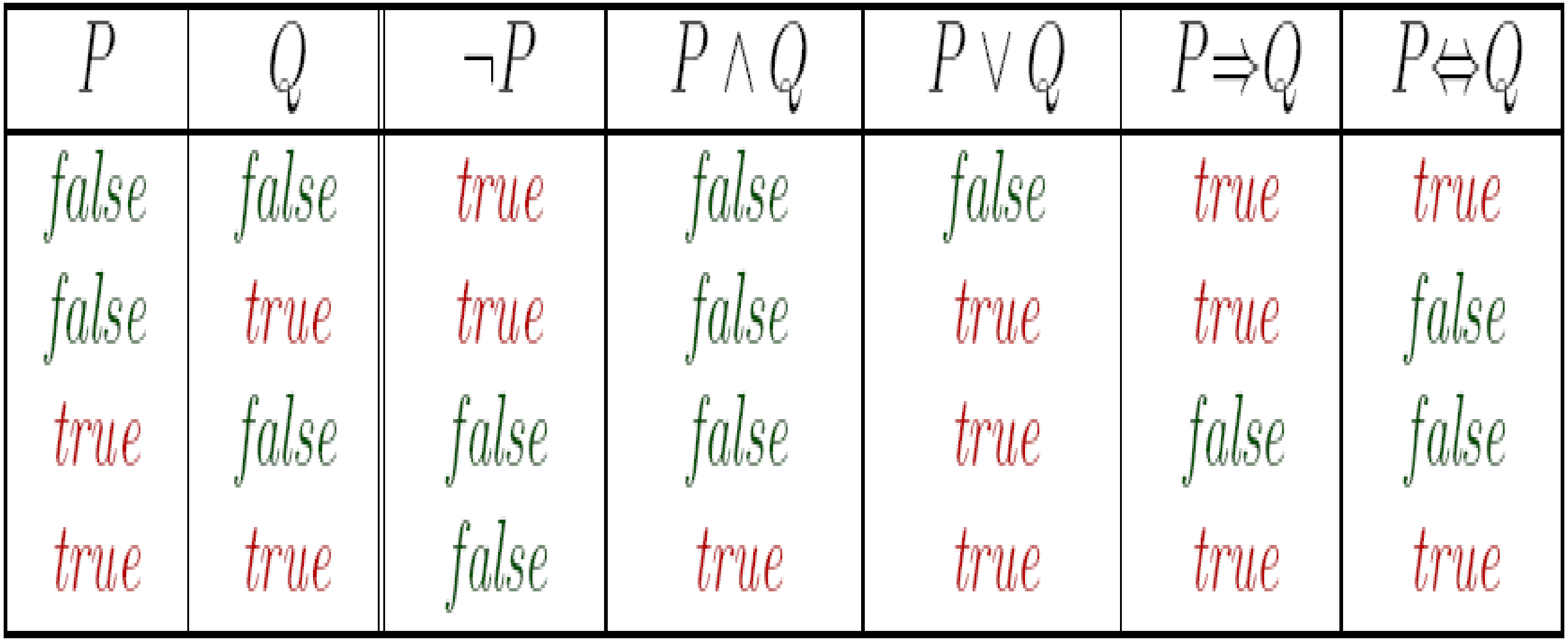
Well-Formed Formula (WFF)



Example Of Well Formed Formulas:

WFF	Explanation
¬¬P	By **Rule 1** each Statement by itself is a WFF, ¬P is a WFF, and let ¬P = Q. So ¬Q will also be a WFF.
((P⇒Q)⇒Q)	By **Rule 3** joining '(P⇒Q)' and 'Q' with connective symbol '⇒'.
(¬Q ∧ P)	By **Rule 3** joining '¬Q' and 'P' with connective symbol '∧'.
((¬P∨Q) ∧ ¬¬Q)	By **Rule 3** joining '(¬P∨Q)' and '¬¬Q' with connective symbol '∧'.
¬((¬P∨Q) ∧ ¬¬Q)	By **Rule 3** joining '(¬P∨Q)' and '¬¬Q' with connective symbol '∧' and then using Rule 2.

Recap: Truth Tables



P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Truth Tables for Compound Propositions

- Truth tables may be used to show **interpretations** of compound propositions.
- To draw up a truth table, construct a column for each proposition involved.
- You need 2^n rows for n propositions for all possible ways of setting the propositions to T and F.
- If we have 3 propositions, p; q; r, i.e. we need $2^3 = 8$ rows.
- Next, construct a column for each connective, the most deeply nested first.
- Evaluate each column using values for propositions or previous columns.

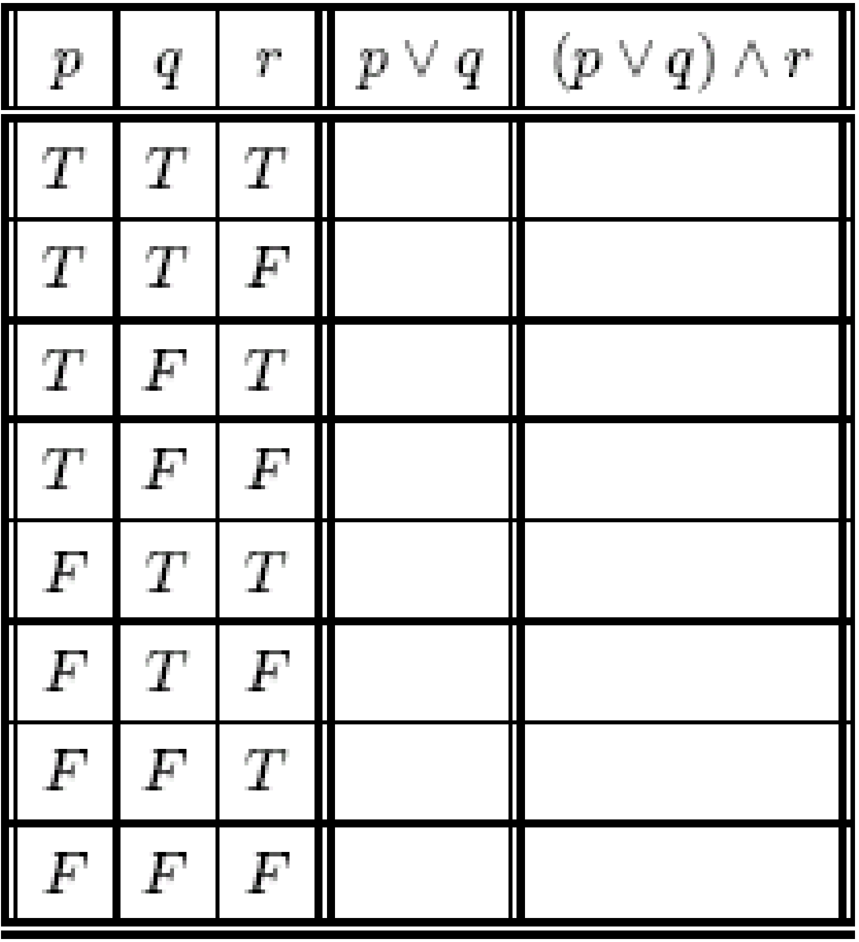
Exercise

p.. (my breakfast is) eggs.

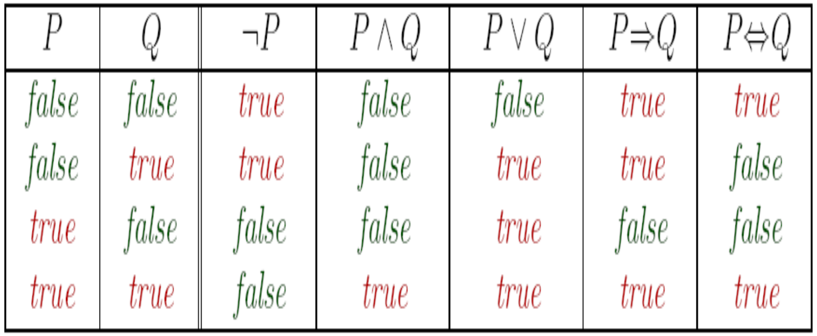
q.. (my breakfast is) cereal.

r.. (my breakfast is) toast.

- The statement ‘my breakfast is either eggs or cereal, and toast’ may be written in symbolic form as?
- Complete the output of the table.



p	q	r	$p \vee q$	$(p \vee q) \wedge r$
T	T	T		
T	T	F		
T	F	T		
T	F	F		
F	T	T		
F	T	F		
F	F	T		
F	F	F		



P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	false	true	false	false	true	true
false	true	true	false	true	true	false
true	false	false	false	true	false	false
true	true	false	true	true	true	true

2.3 Predicate Logic

Predicate Logic

- Propositional logic combines atoms
  - An atom contains no propositional connectives
  - Have no structure (Apple_is_red, John_likes_apples)
- **Predicates** allow us to talk about objects
  - Properties: is_red(apple)
  - Relations: likes(john, apples)
  - True or false
- In predicate logic each atom is a predicate
  - e.g. first order logic, higher-order logic

Predicate Calculus (First Order Logic)

- **Constants** are objects: john, apples
- **Predicates** are properties and relations:
  - likes(john, apples)
- **Functions** transform objects:
  - likes(john, fruit of(apple_tree))
- **Variables** represent any object: likes(X, apples)
- **Quantifiers** qualify values of variables
  - True for every objects (Universal): $\forall$X. likes(X, apples)
  - Exists at least one object (Existential): $\exists$X. likes(X, apples)

Example: FOL Sentence

- "Every rose has a thorn"

$$\forall X.(rose(X) \rightarrow \exists Y.(has(X,Y) \wedge thorn(Y)))$$

- For all X
  - if (X is a rose)
  - then there exists Y
    - (X has Y) and (Y is a thorn)

Example: FOL Sentence

- “On Mondays and Wednesdays I go to John’s house for dinner”

$$\forall X.((is\_mon(X) \vee is\_wed(X)) \rightarrow eat\_meal(me, houseOf(john), X))$$

- Note the change from “and” to “or”
  - Translating is problematic

First-Order Logic (FOL)

- FOL is a formal language for describing knowledge about objects, their properties and relations
- Constants and variables are associated with objects, while predicates describe relations between objects
- Unlike propositional logic, it can handle infinite domains through the use of quantifiers
- Generalized modus ponens is a sound inference method, but it is complete only for a subset of FOL (clauses with exactly one positive literal)
- Inference in FOL is only semi-decidable: can find a proof if one exists, but if a proof does not exist, the procedure may not halt.

First-Order Logic (FOL)

- A key element of FOL are *predicates*, which are used to describe objects, properties, and relationships between objects
  E.g. *On(x,y)*
- A *quantified statement* is a statement that applies to a class of objects
  E.g. $\forall x\ On(x, Table) \rightarrow Fruit(x)$
  - This means that there is only fruit on the table
  - The first element is called a *quantifier*, *x* is a *variable* and *Table* is a *constant*
  - *On* is a *predicate*
- The use of *quantifiers* allows FOL to handle *infinite domains*, while propositional logic can only handle finite domains.

First Order Predicate Calculus

- Includes 2 symbols:
  - Variable quantifiers
    - ∃ (existential) and
    - ∀ (universal)
- A quantifier followed by a variable and a sentence:
  - ∀ X likes(X,pizza) ;true for all X
  - ∃ Y friends(Y,amir) ;true if there is at least one

Universal Quantification

- Syntax: $\forall$*variables sentence*
- E.g. Everyone taking AI is smart.

  $\forall x\ Taking(x,AI) \rightarrow Smart(x)$
- Semantics: $\forall x\ S$ is equivalent to the *conjunction of instantiations* of S:

$$
\begin{array}{ll}
 & Taking(John,AI) \rightarrow Smart(John) \\
\wedge & Taking(Ann,AI) \rightarrow Smart(Ann) \\
\wedge & \ldots
\end{array}
$$

- Typically, $\rightarrow$ is the main connective with $\forall$.

Conjunction means AND

Example

- What does this statement mean:

  $\forall x\ Taking(x, AI) \land Smart(x)$

Example

- What does this statement mean:

  $\forall x\ Taking(x,AI) \wedge Smart(x)$

- Common mistake: using $\wedge$ as the main connective with $\forall$:

  $\forall x\ Taking(x,AI) \wedge Smart(x)$

  means "Everyone is taking AI and everyone is smart"

Existential Quantification

- Syntax: $\exists$*variables sentence*
- Someone taking AI is smart:

  $\exists x\ Taking(x,AI) \wedge Smart(x)$
- Semantics: $\exists x\ S$ is equivalent to the *disjunction of instantiations* of *S*

$$
\begin{aligned}
& (Taking(Ann,AI) \wedge Smart(Ann)) \\
\vee\ & (Taking(John,AI) \wedge Smart(John)) \\
\vee\ & \ldots
\end{aligned}
$$

- Typically, $\wedge$ is the main connective with $\exists$.

Disjunction means OR

Example

- What does this mean:

  $\exists x\ Taking(x, AI) \rightarrow Smart(x)$

Example

- What does this mean:

  $\exists x\ Taking(x, AI) \rightarrow Smart(x)$

- Common mistake: using → as the main connective with ∃:

  $\exists x\ Taking(x, AI) \rightarrow Smart(x)$ is true if there is anyone who is not taking AI!

Properties of Quantifiers

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- *$\exists x \forall y$ is not the same as $\forall y \exists x$*

$$\exists x \forall y Loves(x, y)$$

$$\forall y \exists x Loves(x, y)$$

Properties of Quantifiers

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- *$\exists x \forall y$ is not the same as $\forall y \exists x$*

$\exists x \forall y Loves(x, y)$

"There is a person who loves everyone in the world"

$\forall y \exists x Loves(x, y)$

"Everyone in the world is loved by at least one person"

Quantifier Duality

Each quantifier can be expressed using the other quantifier and negation:

- $\forall x$ *Likes(x,IceCream)* is equivalent to $\neg\exists x$ $\neg$*Likes(x,IceCream)*
- $\exists x$ *Likes(x,Broccoli)* is equivalent to $\neg\forall x$ $\neg$*Likes(x,Broccoli)*

Everyone likes ice cream == Not someone not liking ice cream (nobody do not like ice cream)

Someone like broccoli == Not everyone not liking broccoli

How to read quantified formulas

- When reading quantified formulas in English, **read them from left to right.** $\forall x$ can be read as "for every object x in the universe the following holds" and $\exists x$ can be read as "there exists an object x in the universe which satisfies the following" or "for some object x in the universe the following holds".
- For example, let the universe be the set of airplanes and let $F(x, y)$ denote "x flies faster than y". Then
- $\forall x \, \forall y \, F(x, y)$ can be translated initially as "For every airplane x the following holds: x is faster than every (any) airplane y". In simpler English it means "Every airplane is faster than every airplane (including itself !)".
- $\forall x \, \exists y \, F(x, y)$ can be read initially as "For every airplane x the following holds: for some airplane y, x is faster than y". In simpler English it means "Every airplane is faster than some airplane".
- $\exists x \, \forall y \, F(x, y)$ represents "There exist an airplane x which satisfies the following: (or such that) for every airplane y, x is faster than y". In simpler English it says "There is an airplane which is faster than every airplane" or "Some airplane is faster than every airplane".
- $\exists x \, \exists y \, F(x, y)$ reads "For some airplane x there exists an airplane y such that x is faster than y", which means "Some airplane is faster than some airplane".

Fun with Sentences

All x is the first cousin of (every cousin) y IFF a parent p with a child x whose a sibling ps has a child y

- Brothers are siblings

$$\forall x \forall y Brother(x, y) \rightarrow Sibling(x, y)$$

- “Sibling” is reflexive

$$\forall x \forall y Sibling(x, y) \leftrightarrow Sibling(y, x)$$

- One’s mother is one’s female parent

$$\forall x \forall y Mother(x, y) \leftrightarrow (Female(x) \wedge Parent(x, y))$$

- A first cousin is a child of a parent’s sibling

$$\begin{aligned} &\forall x \forall y FirstCousin(x, y) \leftrightarrow \exists p \exists ps Parent(p, x) \wedge \\ &\qquad Sibling(ps, p) \wedge Parent(ps, y) \end{aligned}$$

More Sample of Questions