Knowledge Representation (KR)

Course Title: Computational Intelligence

Course Code: CSC401A

Vaishali R Kulkarni

Department of Computer Science and Engineering
Faculty of Engineering and Technology
M. S. Ramaiah University of Applied Sciences
Email: *vaishali.c*s.et@msruas.ac.in

Intelligent Agents

- Intelligent agents need knowledge about the world to choose the good actions or decisions
- Knowledge= set of sentences in formal language representing the world
- A sentence is an assertion of the world
- Acknowledge based agent consists of

knowledge base: content about specific world domain.

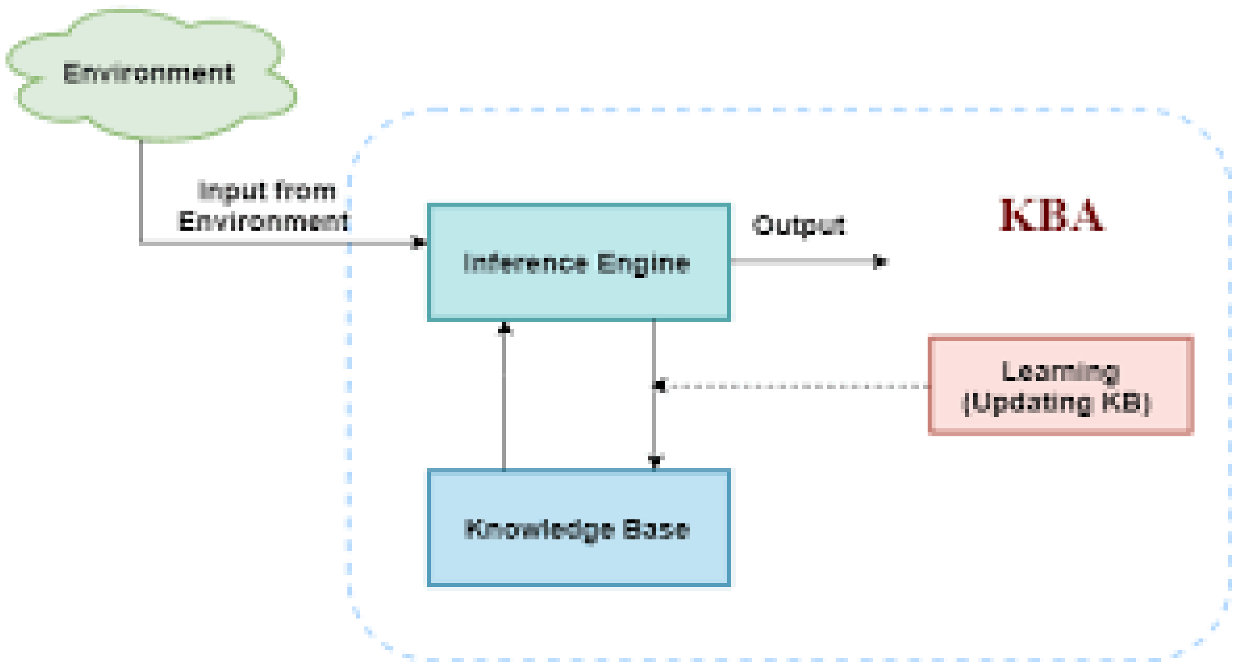
Inference Mechanism: Domain independent algorithm

Knowledge Representation (KR)

- An agent must be capable of:
- ➢ Represent states, actions etc.
- ➢ Incorporate new percepts
- ➢ Update internal representation of the world
- ➢ Deduce hidden properties of the world
- ➢ Deduce appropriate actions

Knowledge-based Agent(KBA)

Declarative: A KBA agent can be built by simply telling what it needs to do
Procedural: Encode desired behaviors directly as program code.
Minimizing the role of explicit representation and reasoning can result in much more efficient system.



Techniques in KR

- Logical Representation
- Semantic Network Representation
- Frame Representation
- Production Rules

Logical Representation

- It is a language with concrete rules that deals with propositions and it has no ambiguity
- It consists of precisely defined syntax and semantics
- In KR, each sentence is translated into logic using precise syntax and semantics.
- Syntax is well defined sentence in the knowledge. Semantics defines the meaning of sentence.
- Logical representation can be
  - Propositional Logic (PL)
  - Predicate Logic

Propositional Logic (PL)

- PL is the simplest logic. A proposition refers to a declarative statement that is true or false.
- If a **proposition** is true, then we say it has a truth value of "true"; if a **proposition** is false, its truth value is "false". For **example**, "Grass is green", and "2 + 5 = 5" are **propositions**. The first **proposition** has the truth value of "true" and the second "false".
- PL can not be predicate

Propositional Logic (PL)

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 five logical Connectives

logic	Name	rank
~	Negation	1
∧	Conjunction	2
V	Disjunction	3
=>	Conditional	4
⇔	Biconditional	5

not
and
or
Implies
If and only if(biconditional)

PL Examples

- *If a person is cool or funny, then he is popular:* $c \lor f \Rightarrow p$
- *A person is popular only if he is either cool or funny:* $p \Rightarrow c \lor f$
- *A person is popular if and only if he is either cool or funny:* $p \Leftrightarrow c \lor f$
- *There is no one who is both cool and funny:* $\neg(c \land f)$
- *Suppose we were to imagine a person who is cool and funny and popular, i.e. the proposition constants c and f and p are all true. Which of our sentences are true and which are false.*
- $c \lor f \Rightarrow p$
- $(1 \lor 1) \Rightarrow 1$
- $1 \Rightarrow 1$
- 1

Propositional language

- Example of Conditional Statement – “If you do your homework, you will not be punished.” Here, "you do your homework" is the hypothesis, p, and "you will not be punished" is the conclusion, q. Then PL is $p \rightarrow q$
- A – It is hot, B-It is humid C- It is raining
- If it is humid then it is hot. PL is $B \rightarrow A$
- If it is hot and humid, then it is not raining
- $(A \land B) \rightarrow \neg C$
- Drawbacks of PL: using Pl, we cannot represent all, some or none. Eg: All students are mature. Some students are good.
- Statements cannot be expressed using their properties

PL

- In the topic of Propositional logic, we have seen that how to represent statements using propositional logic. But unfortunately, in propositional logic, 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. Consider the following sentence, which we cannot represent using PL logic.

- "Some humans are intelligent", or
- "Sachin likes cricket."
- To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.

First-Order logic

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- FOL is sufficiently expressive to represent the natural language statements in a concise way.
- First-order logic is also known as Predicate logic or First-order predicate logic. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.
- First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
- Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
- Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between
- Function: Father of, best friend, third inning of, end of, ...
- As a natural language, first-order logic also has two main parts:
- Syntax and semantics

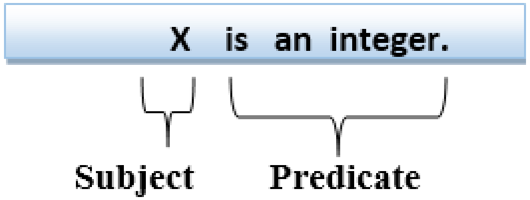
Syntax of First-Order logic:

- The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

Atomic & Complex Sentences:

- Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
- We can represent atomic sentences as Predicate (term1, term2, ..., term n).
- Example: Ravi and Ajay are brothers: => Brothers(Ravi, Ajay), Browny is a cat: => cat (Browny).
- Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
- 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.
- Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.

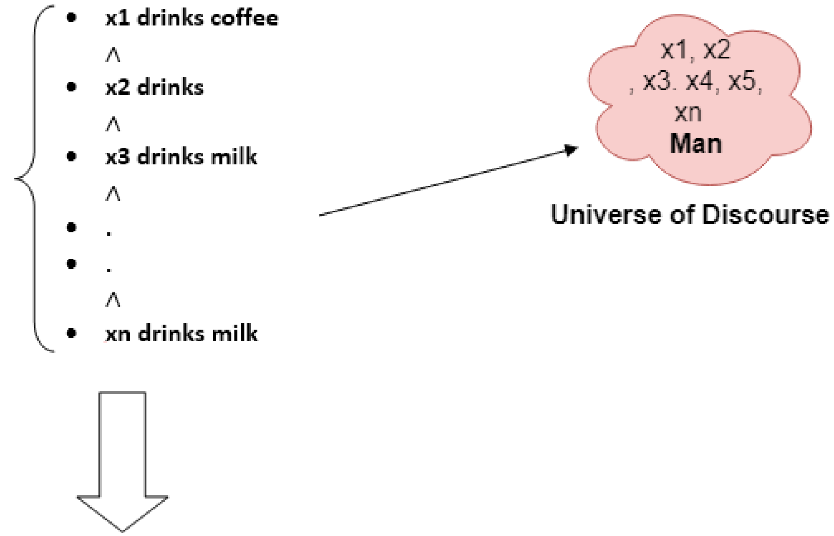


Quantifiers in First-order logic:

- A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:
- Universal Quantifier, (for all, everyone, everything)
- Existential quantifier, (for some, at least one).
- Universal Quantifier: Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

The Universal quantifier

- The Universal quantifier is represented by a symbol ∀, which resembles an inverted A. If x is a variable, then ∀x is read as: For all x, For each x and For every x.
- All man drink coffee.
- Let a variable x which refers to a cat so all x can be represented in UOD as below:
- **∀x man(x) → drink (x, coffee).**
- It will be read as: There are all x where x is a man who drink coffee.



So in shorthand notation, we can write it as :

Predicate Logic - example

- Every Gardner likes sun

  ∀x: Gardner(x)=>likes(x,sun)
- All purple mushrooms are poisonous
- ∀x: mushroom (x) ∧ purple(x)=> poisonous(x)
- Everyone is loyal to someone
- ∀x ∃ y:loyal(x,y) ie for all x there exists y that is loyal to x
- Everyone loves everyone
- ∀x ∀y : loves(x,y)

Predicate Logic - example

- Everyone loves everyone except himself
- $\forall x\ \forall y : loves(x,y) \land \neg\ loves(x,x)$
- All Indians were either loyal to Modi or hated him
- $\forall x$: Indian(x)=>Loyal(x, Modi) $\lor$ Hated(x,Modi)
- (LHS is instance and RHS is inference)

Predicate Logic - example

- People only try to kill rulers they are not loyal to
- $\forall x\ \forall y : person(x) \land Ruler(y) \land \neg\ loyal(x,y) \Rightarrow kill(x,y)$
- Is this correct?
- X is a person, y is a ruler and x is not loyal to y will kill ruler

Predicate Logic - example

- $\forall x \, \forall y : person(x) \wedge Ruler(y) \wedge \neg \, loyal(x,y) \Rightarrow kill(x,y)$
- This is wrong statement
- Suppose there are 1 crore people and all are not loyal, then they all will try to kill ruler
- $\forall x \, \forall y : person(x) \wedge Ruler(y) \wedge kill(x,y) \Rightarrow \neg \, loyal(x,y)$
- This is a correct version. If x is a person, y is a ruler and x tries to kill y, then x is not loyal to y

Predicate Logic - example

- Anyone who is married and has more than one spouse is a bigamist (Criminal)

$\forall x$: Married(x) $\wedge$ (no. of spouse(x) > 1)=>bigamist(x)

$\forall x$: Married(x) $\wedge$ gt(no. of spouse(x),1)=>bigamist(x)

You can fool all the people some of the time

$\exists$ t $\forall x$: Person(x) => time(t) $\wedge$ can fool(x,t)

This means every person can be fooled some time

You can fool some of the people some of the time (idiots)

$\forall x$: $\exists$ t:Person(x) $\wedge$ time(t) => canfool(x,t)

- One’s husband is one’s spouse
- $\forall w : \exists\ h{:}Husband(h,w) \Rightarrow male(h) \land spouse(h,w)$
- For every instance of w, there exits a specific instance h, where h is husband to w then we can conclude h is spouse to w
- Parent and children are Inverse Relation
- $\forall p\ \forall c : parent(p,c) \Rightarrow child(c,p)$

Basic elements of predicate

- Constants: Modi, sachin
- Variables: h,w,x,y,p
- Predicates: people, brother
- Function: left, right, spouse,kill,fool
- Connectives: and, or, not, implies(=>)and iff⇔
- Eqaulity ==
- Quatifiers : Universal, existential

- All Birds fly
- Fly(bird)
- ∀x bird(x)=>fly(x)

Some boys play cricket: The predicate is play(boys,cricket)

∃x boys(x)=>play(x,y) here x= boys and y=cricket

Every man respects his parent

∀x man(x)->respects(x,parent)

Not all students like both maths and science

¬ ∀(x)[student(x)->like(x,maths) ∧like(x,science)

Liked, likeble, liking -→ like- Python Library

- Only one student failed in maths
- X = student, y = subjects and maths
- ∃(x)[student(x)->failed(x,maths) ∧ ∀(y)
- Free Variable:

Inference: deducing the facts

All kings who are greedy are evil

∀x king(x) ^ greedy(x) => evil(x)

References

- Artificial Intelligence A Modern Approach Third Edition by Stuart J. Russell and Peter Norvig