Lecture 19
Artificial Intelligence

Khola Naseem
khola.naseem@uet.edu.pk

Knowledge representation

Knowledge representation

- Humans are best at understanding, reasoning, and interpreting knowledge.
- Human knows things, which is knowledge and as per their knowledge they perform various actions in the real world.
- But how machines do all these things comes under knowledge representation and reasoning.
- Representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language

Knowledge representation

- **What to Represent:**
- Following are the kind of knowledge which needs to be represented in AI systems:
- Object: All the facts about objects in our world domain. E.g., Guitars contains strings, cats have for legs.
- Events: Events are the actions which occur in our world.
- Performance: It describe behavior which involves knowledge about how to do things.
- Meta-knowledge: It is knowledge about what we know.
- Facts: Facts are the truths about the real world

Knowledge representation

- **What to Represent:**
- Knowledge-Base: The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the Sentences (Here, sentences are used as a technical term and not identical with the English language).
- Each sentence is expressed in a language called a knowledge representation language and represents assertion about the world
- Each time the agent program is called, it does three things.
  - First, it TELLs the knowledge base what it perceives.
  - Second, it ASKs the knowledge base what action it should perform. In the process of answering this query, extensive reasoning may be done about the current state of the world, about the outcomes of possible action sequences, and so on.
  - Third, the agent program TELLs the knowledge base which action was chosen, and the agent executes the action

Knowledge representation

- **Types of knowledge**

Knowledge representation

- Types of knowledge
- 1. Declarative Knowledge:
  - Declarative knowledge is to know about something.
  - It includes concepts, facts, and objects.
  - It is also called descriptive knowledge and expressed in declarative sentences.
  - What is known about a situation, e.g. it is sunny today, and cherries are red, car has 4 tyres
- 2. Procedural Knowledge
  - Procedural knowledge is a type of knowledge which is responsible for knowing **how to do something.**
  - It can be directly applied to any task.
  - It includes rules, strategies, procedures, agendas, etc.
  - Procedural knowledge depends on the task on which it can be applied
  - e.g., how to drive a car

Knowledge representation

- **Types of knowledge**
- 3. Meta-knowledge:
  - Knowledge about the other types of knowledge is called Meta-knowledge.
  - the knowledge that blood pressure is more important for diagnosing a medical condition than eye color
- 4. Heuristic knowledge:
  - Heuristic knowledge is representing knowledge of some experts in a filed or subject.
  - Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.
  - if I start seeing shops, I am close to the market.

Knowledge representation

- **Types of knowledge**

- 5. Structural knowledge:
  - Structural knowledge is basic knowledge to problem-solving.
  - It describes relationships between various concepts such as kind of, part of, and grouping of something.
  - e.g. how the various parts of the car fit together to make a car,

Knowledge representation

Logical Representation

- Logical representation is a language with some definite rules which deal with propositions and has no ambiguity in representation.
- Logical representation means drawing a conclusion based on various conditions.
- Also, it consists of precisely defined syntax and semantics which supports the sound inference.
- Each sentence can be translated into logics using syntax and semantics.

Knowledge representation

Logical Representation

- Syntax vs semantics: if statement

Syntax	Semantics
• It decides how we can construct legal sentences in logic. • It determines which symbol we can use in knowledge representation. • Also, how to write those symbols.	• Semantics are the rules by which we can interpret the sentence in the logic. • It assigns a meaning to each sentence.

- Types:
  - Predicate logic
  - Propositional logic-> true OR false
- Example:

Spot is a dog *dog(Spot)*

-

All dogs have tails $\forall x{:}dogs(x)\text{->}hastail(x)$

Spot has a tail *hastail(Spot)*

Credit: Khola Naseem

Knowledge representation

Logical Representation

- Propositional logic-> true OR false
- Types:
  - Simple, Complex
  - These are joined together to form more complex statements by logical connectives, expressing simple ideas such as and, or, not, if…then…,iff.
- There are standard symbols for these:
  - $\wedge$ stands for “and”,
  - $\vee$ stands for “or”,
  - $\neg$ stands for “not”,
  - $\Rightarrow$ / $\rightarrow$ stands for “if … then …”,
  - $\Leftrightarrow$ stands for “if and only if”.

Knowledge representation

Logical Representation

- Propositional logic-> true OR false

$Sentence$	$\rightarrow$	$AtomicSentence \mid ComplexSentence$
$AtomicSentence$	$\rightarrow$	$True \mid False \mid P \mid Q \mid R \mid \ldots$
$ComplexSentence$	$\rightarrow$	$(\ Sentence\ ) \mid [\ Sentence\ ]$
	$\mid$	$\neg\ Sentence$
	$\mid$	$Sentence \land Sentence$
	$\mid$	$Sentence \lor Sentence$
	$\mid$	$Sentence \Rightarrow Sentence$
	$\mid$	$Sentence \Leftrightarrow Sentence$
OPERATOR PRECEDENCE	:	$\neg, \land, \lor, \Rightarrow, \Leftrightarrow$

Figure 7.7 A BNF (Backus–Naur Form) grammar of sentences in propositional logic, along with operator precedences, from highest to lowest.

Example:

$$\neg A \land B$$

Knowledge representation

Logical Representation

- Propositional logic-> semantic:
  - $\neg P$ is true iff P is false in m.
  - $P \wedge Q$ is true iff both P and Q are true in m.
  - $P \vee Q$ is true iff either P or Q is true in m.
  - $P \Rightarrow Q$ is true unless P is true and Q is false in m.
  - $P \Leftrightarrow Q$ is true iff P and Q are both true or both false in m.

- Truth table:

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*

Knowledge representation

Logical Representation

- Propositional logic-> Standard logical equivalences

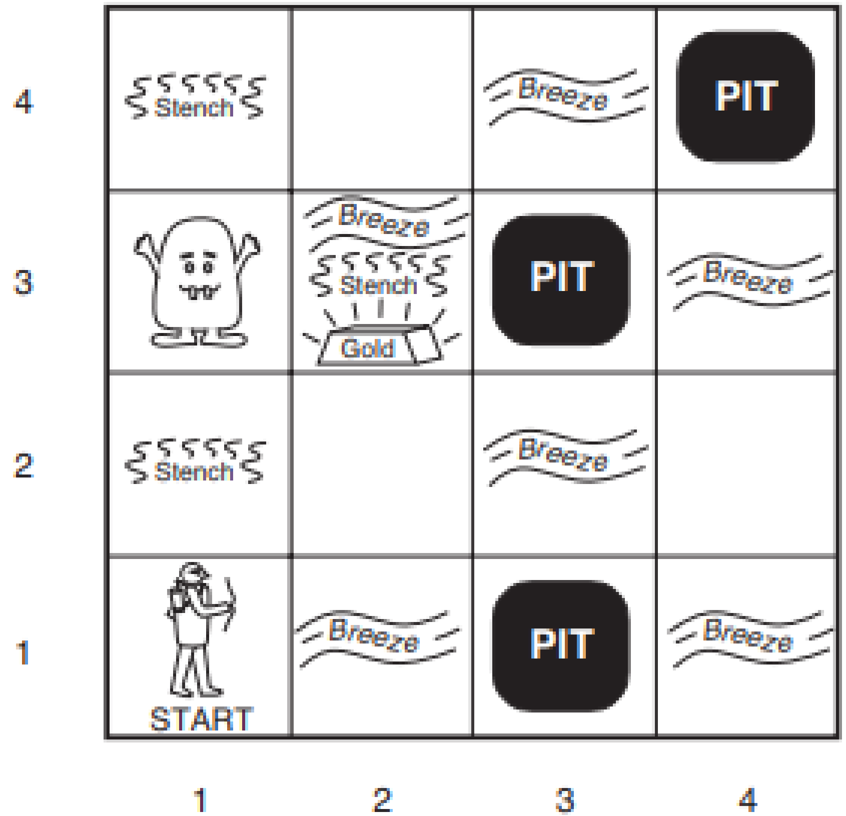
.

$$
\begin{aligned}
(\alpha \land \beta) &\equiv (\beta \land \alpha) \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) \\
\neg(\neg\alpha) &\equiv \alpha \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \lor \beta) \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))
\end{aligned}
$$

Knowledge representation

Logical Representation

- A typical wumpus world.
- https://thiagodnf.github.io/wumpus-world-simulator/

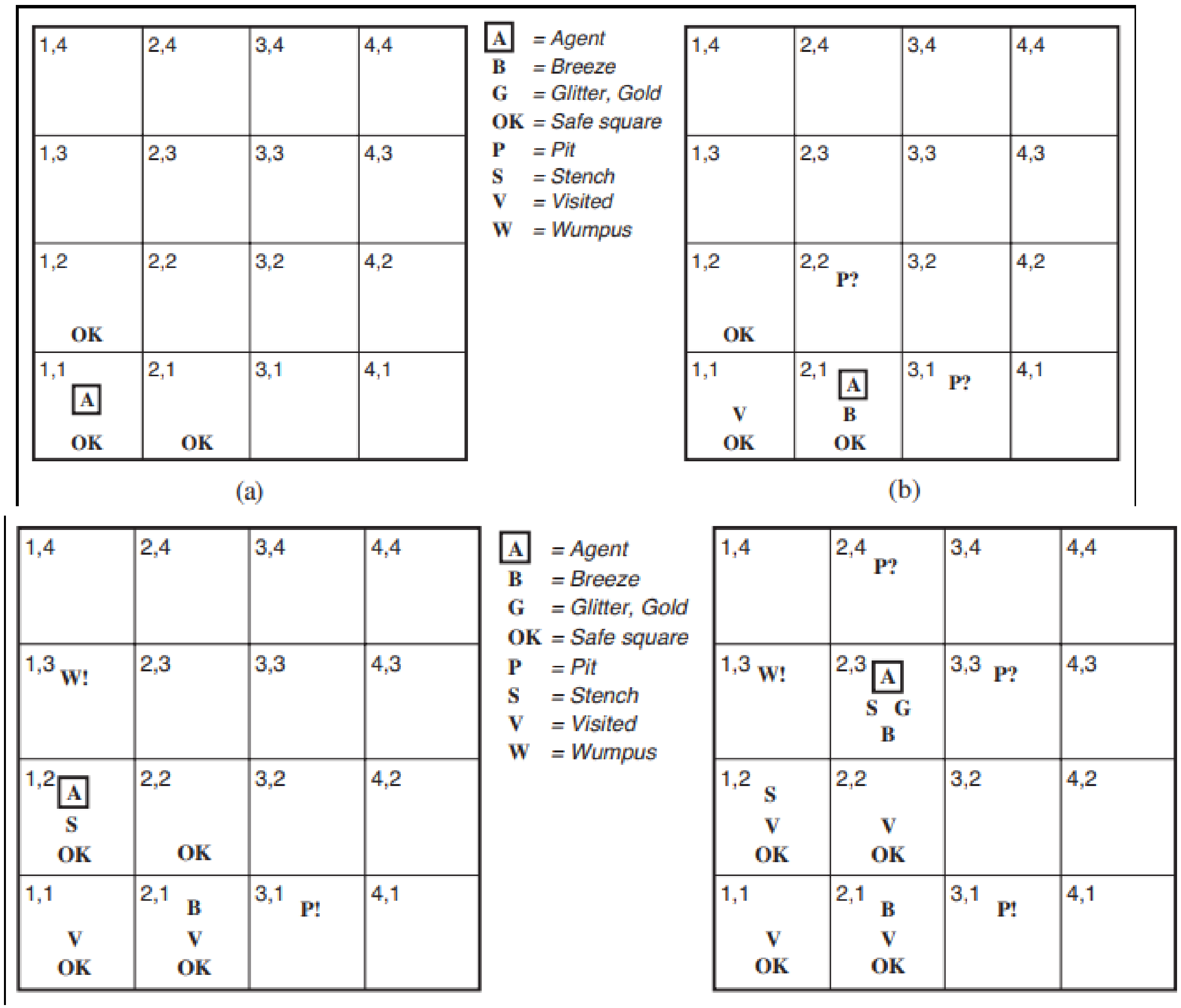


Credit: Khola Naseem

Knowledge representation

Logical Representation

- A typical wumpus world.



Knowledge representation

Logical Representation

- A typical wumpus world.

$P_{x,y}$ is true if there is a pit in $[x, y]$.
$W_{x,y}$ is true if there is a wumpus in $[x, y]$, dead or alive.
$B_{x,y}$ is true if the agent perceives a breeze in $[x, y]$.
$S_{x,y}$ is true if the agent perceives a stench in $[x, y]$.

There is no pit in [1,1]:

$$R_1 : \quad \neg P_{1,1} .$$

A square is breezy if and only if there is a pit in a neighboring square. This has to be stated for each square; for now, we include just the relevant squares:

$$R_2 : \quad B_{1,1} \quad \Leftrightarrow \quad (P_{1,2} \lor P_{2,1}) .$$
$$R_3 : \quad B_{2,1} \quad \Leftrightarrow \quad (P_{1,1} \lor P_{2,2} \lor P_{3,1}) .$$

$$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$$
$$S_{1,1} \Leftrightarrow (W_{1,2} \lor W_{2,1})$$

Knowledge representation

Logical Representation

- Propositional logic-> true OR false
- Types:
  - Simple, Complex
- example of a statement written in propositional calculus:
  - Suppose that R stands for “It is raining”, G stands for “I have got a coat”, W stands for “I will get wet”. The statement

$$\mathbf{R} \land \neg \mathbf{G} \Rightarrow \mathbf{W}$$

  - is a way of writing "If it is raining and I have not got a coat, then I will get wet."

Knowledge representation

Logical Representation

- Predicate calculus :
- make statements about objects, and the properties of objects, and the relationships between objects (propositional calculus can't).
- It contains predicates – statements like this:
- a(S)
- or this: b(S, T)
  - that mean S has the property a, or S and T are connected by the relationship b.
  - Example:

$$Brother(KingJohn, RichardTheLionheart)$$

Knowledge representation

Logical Representation

- Predicate calculus :
- Complex:
- $$Sibling(KingJohn, Richard) \Rightarrow Sibling(Richard, KingJohn)$$

Consider the interpretation in which
$Richard \rightarrow$ Richard the Lionheart
$John \rightarrow$ the evil King John
$Brother \rightarrow$ the brotherhood relation

Under this interpretation, $Brother(Richard, John)$ is true just in case Richard the Lionheart and the evil King John are in the brotherhood relation in the model

Knowledge representation

Logical Representation

- Predicate calculus
- Example of a statement written
- Suppose that c stands for "the cat", m stands for "the mat", s stands for "sits on", b stands for "black", f stands for "fat", h stands for "happy". The statement
- $(f(c) \wedge b(c) \wedge s(c,m)) \Rightarrow h(c)$
- is a way of writing "If the fat black cat sits on the mat then it is happy".

Knowledge representation

Logical Representation

- Predicate calculus

➢ As well as having the same logical connectives as propositional calculus, predicate calculus has two quantifiers,

- $\forall$ meaning “for all”,
- $\exists$ meaning “there exists”.

Knowledge representation

Logical Representation

- Predicate calculus

➢ $\forall$ meaning "for all",

- $\forall \langle variables \rangle \; \langle sentence \rangle$

  Everyone at Berkeley is smart:
  $\forall x \;\; At(x, Berkeley) \;\Rightarrow\; Smart(x)$

  $\forall x \;\; P$ is true in a model m iff P is true with x being **each** possible object in the model

  Roughly speaking, equivalent to the conjunction of instantiations of P

$$
\begin{array}{l}
\quad (At(KingJohn, Berkeley) \;\Rightarrow\; Smart(KingJohn)) \\
\wedge \; (At(Richard, Berkeley) \;\Rightarrow\; Smart(Richard)) \\
\wedge \; (At(Berkeley, Berkeley) \;\Rightarrow\; Smart(Berkeley)) \\
\wedge \; \ldots
\end{array}
$$

Knowledge representation

Logical Representation

- Predicate calculus

➢ ∀ meaning “for all”,

- Common mistake to avoid

Typically, $\Rightarrow$ is the main connective with $\forall$

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

$$\forall x \;\; At(x, Berkeley) \wedge Smart(x)$$

means “Everyone is at Berkeley and everyone is smart”

Knowledge representation

Logical Representation

- Predicate calculus
- Existential Quantification

$\exists \langle variables \rangle \ \ \langle sentence \rangle$

Someone at Stanford is smart:
$\exists x \ \ At(x, Stanford) \land Smart(x)$

$\exists x \ \ P$ is true in a model m iff P is true with x being **some** possible object in the model

Roughly speaking, equivalent to the disjunction of instantiations of P

$$\begin{aligned} & (At(KingJohn, Stanford) \land Smart(KingJohn)) \\ \lor \ & (At(Richard, Stanford) \land Smart(Richard)) \\ \lor \ & (At(Stanford, Stanford) \land Smart(Stanford)) \\ \lor \ & \ldots \end{aligned}$$

Knowledge representation

Logical Representation

- Predicate calculus
- Existential Quantification:

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

Knowledge representation

Logical Representation

- Predicate calculus
- 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 can be expressed using the other

$\forall x \ Likes(x, IceCream) \qquad \neg\exists x \ \neg Likes(x, IceCream)$

$\exists x \ Likes(x, Broccoli) \qquad \neg\forall x \ \neg Likes(x, Broccoli)$

Credit: Khola Naseem

Knowledge representation

Logical Representation

- Predicate calculus
- Properties of Quantifiers

Brothers are siblings

$\forall x, y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" is symmetric

$\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x).$

One's mother is one's female parent

$\forall x, y \;\; Mother(x, y) \;\Leftrightarrow\; (Female(x) \wedge Parent(x, y)).$

A first cousin is a child of a parent's sibling

$\forall x, y \;\; FirstCousin(x, y) \;\Leftrightarrow\; \exists p, ps \;\; Parent(p, x) \wedge Sibling(ps, p) \wedge Parent(ps, y)$

Knowledge representation

Logical Representation

- Predicate calculus
- Properties of Quantifiers

$term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object

E.g., definition of (full) $Sibling$ in terms of $Parent$:

$$\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; [\neg(x = y) \land \exists\, m, f \;\; \neg(m = f) \land Parent(m, x) \land Parent(f, x) \land Parent(m, y) \land Parent(f, y)]$$

Knowledge representation

Logical Representation

- Predicate calculus

$$
\begin{array}{rcl}
\textit{Sentence} & \rightarrow & \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} & \rightarrow & \textit{Predicate} \mid \textit{Predicate}(\textit{Term},\ldots) \mid \textit{Term} = \textit{Term} \\
\textit{ComplexSentence} & \rightarrow & (\ \textit{Sentence}\ ) \mid [\ \textit{Sentence}\ ] \\
& \mid & \neg\ \textit{Sentence} \\
& \mid & \textit{Sentence} \land \textit{Sentence} \\
& \mid & \textit{Sentence} \lor \textit{Sentence} \\
& \mid & \textit{Sentence} \ \Rightarrow\ \textit{Sentence} \\
& \mid & \textit{Sentence} \ \Leftrightarrow\ \textit{Sentence} \\
& \mid & \textit{Quantifier}\ \textit{Variable},\ldots\ \textit{Sentence} \\
\\
\textit{Term} & \rightarrow & \textit{Function}(\textit{Term},\ldots) \\
& \mid & \textit{Constant} \\
& \mid & \textit{Variable} \\
\\
\textit{Quantifier} & \rightarrow & \forall \mid \exists \\
\textit{Constant} & \rightarrow & A \mid X_1 \mid \textit{John} \mid \cdots \\
\textit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\textit{Predicate} & \rightarrow & \textit{True} \mid \textit{False} \mid \textit{After} \mid \textit{Loves} \mid \textit{Raining} \mid \cdots \\
\textit{Function} & \rightarrow & \textit{Mother} \mid \textit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} & : & \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

Credit: Khola Naseem

Knowledge representation

Logical Representation

- Predicate calculus
- Example of statement written in predicate calculus using these quantifiers:
- Suppose that d stands for “is a day”, p stands for “is a person”, mo stands for “is mugged on”, mi stands for “is mugged in”, S stands for Soho, x stands for some unspecified day and y stands for some unspecified person.
  - $\forall x(d(x) \rightarrow \exists y(p(y) \wedge mo(y, x) \wedge mi(y, S)))$
  - expresses the idea "Someone is mugged in Soho every day."

Knowledge representation

Representing facts with First-Order Logic

- Ali is a professor
- All professors are persons
- Salman is the dean
- Deans are professors
- All professors consider dean a friend or don’t know him
- Everyone is a friend of someone
- People only criticize people that are not their friends
- ali criticized Salman

Knowledge representation

Representing facts with First-Order Logic

- isProf (Ali)
  - Ali is a professor
- $\forall x(\text{isProf}(\text{x}) \rightarrow \text{isPerson}(\text{x}))$
  - All professors are persons
- isDean(Salman)
  - Salman is the dean
- $\forall x(\text{isDean}(\text{x}) \rightarrow \text{isProf}(\text{x}))$
  - Deans are professors
- $\forall x \forall y \left(isProf(x) \wedge isDean(y) \rightarrow friendOf(x, y) \vee \neg knows\,(x, y)\right)$
  - All professors consider dean a friend or don't know him

Credit: Khola Naseem

Knowledge representation

Representing facts with First-Order Logic

- $\forall x \exists y \; (friendOf(x, y))$
  - Everyone is a friend of someone
- $\forall x \forall y (isPerson(x) \wedge isPerson(y) \wedge criticise(x, y) \rightarrow \neg friendOf(x, y))$
  - People only criticize people that are not their friends
- $criticize(Ali, Salman)$
  - Ali criticized Salman
- Question: is Salman not friend of Ali?
  - $\neg freindOf(Salman, Ali)$

Knowledge representation

Representing facts with First-Order Logic

- How Machine "sees" it

$Ali = A$
$Salman = B$
$isProf(x) = P1(x)$
$friendOf(x) = P2(x)$
$isPerson(x) = P3(x)$
$isDean(x) = P4(x)$
$knows(x, y) = P5(x, y)$
$criticize(x, y) = P6(x, y)$

Knowledge Base:

- P1(A)
- $\forall x(\text{P1(x)} \rightarrow \text{P3(x)})$
- P4(B)
- $\forall x(\text{P4(x)} \rightarrow P1(\text{x}))$
- $\forall x \forall y\ ((P1(x) \land P4(y) \rightarrow P2(x, y)) \lor \neg P5(x, y))$
- $\forall x \exists y\ (P2(x, y)$
- $\forall x \forall y (P3(x) \land P3(y) \land P6(x, y) \rightarrow \neg P2(x, y))$
- $P6(A, B)$
- **Question:** $\neg \boldsymbol{P2}(\boldsymbol{B}, \boldsymbol{A})$**?**

Credit: Khola Naseem

Knowledge representation

Representing facts with First-Order Logic

Knowledge Engineering

1. Identify the task.
2. Assemble the relevant knowledge.
3. Decide on a vocabulary of predicates, functions, and constants.
4. Encode general knowledge about the domain.
5. Encode a description of the specific problem instance.
6. Pose queries to the inference procedure and get answers.
7. Debug the knowledge base.

Knowledge representation

Knowledge engineer

General Knowledge

1. All professors are persons
2. Deans are professors
3. All professors consider dean a friend or don’t know him
4. Everyone is a friend of someone

Specific Problem

1. Ali is a professor
2. People only criticize people that are not their friends
3. Salman is the dean
4. Ali criticized Salman

Query

5. Is Salman not friend of Ali?

Credit: Khola Naseem

Knowledge representation

Logical Representation

- Predicate calculus
- "Every rich person owns a house. Susan is rich. Susan is a person. Therefore Susan owns a house.“
- Convert these statements into predicate calculus (I've used x, y, & z for variables. Susan is a constant).

$\forall x\ [(person(x) \wedge rich(x)) \rightarrow \exists y(house(y) \wedge owns(x,y))]$

rich(Susan).
person(Susan).
The conclusion:
$\exists z(house(z) \wedge owns(Susan,z))$.

Knowledge representation

Semantic Network Representation

- Semantic networks are alternative of predicate logic for knowledge representation.
- In Semantic networks, we can represent our knowledge in the form of graphical networks.
- This network consists of nodes representing objects and arcs which describe the relationship between those objects.
- This representation consist of mainly two types of relations:
  - IS-A relation (Inheritance)
  - Kind-of-relation

Knowledge representation

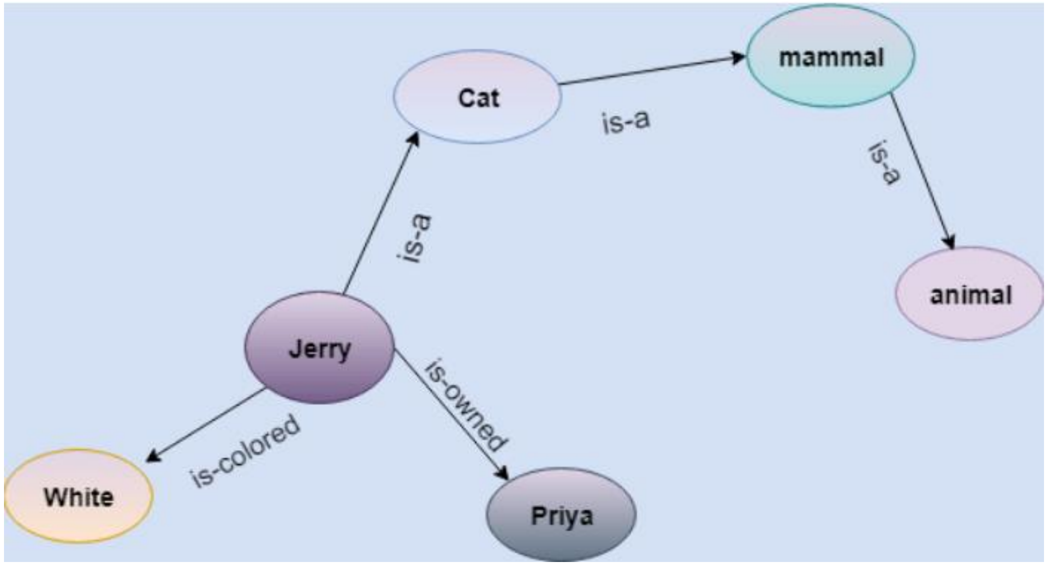
Semantic Network Representation

- **Example:** Following are some statements which we need to represent in the form of nodes and arcs.
- Statements:
  - Jerry is a cat.
  - Jerry is a mammal
  - Jerry is owned by Priya.
  - Jerry is brown colored.
  - All Mammals are animal.

Knowledge representation

Semantic Network Representation

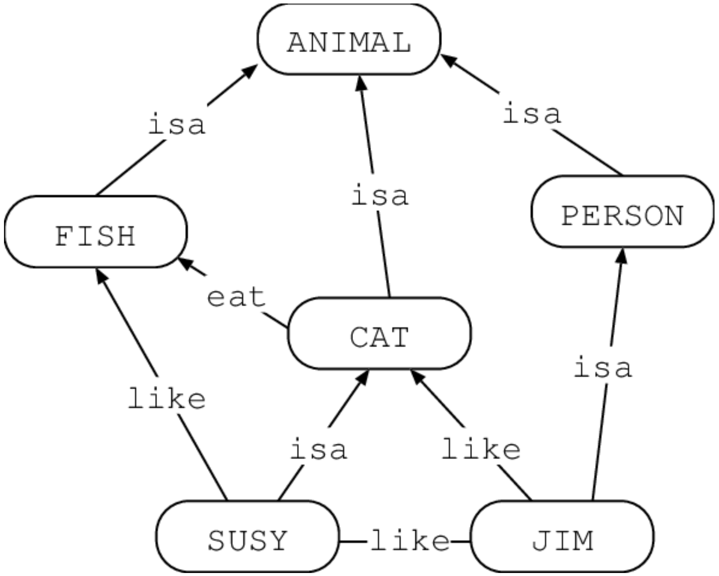
- **Example:** Following are some statements which we need to represent in the form of nodes and arcs.
- Statements:
  - Jerry is a cat.
  - Jerry is a mammal
  - Jerry is owned by Priya.
  - Jerry is brown colored.
  - All Mammals are animal.



Knowledge representation

Semantic Network Representation

- Example:
- SUSY is a CAT and JIM is a PERSON. JIM's pet is SUSY. They both like each other. SUSY is a CAT and CAT is an ANIMAL. SUSY like Fish and fish is an animal. person is also an animal

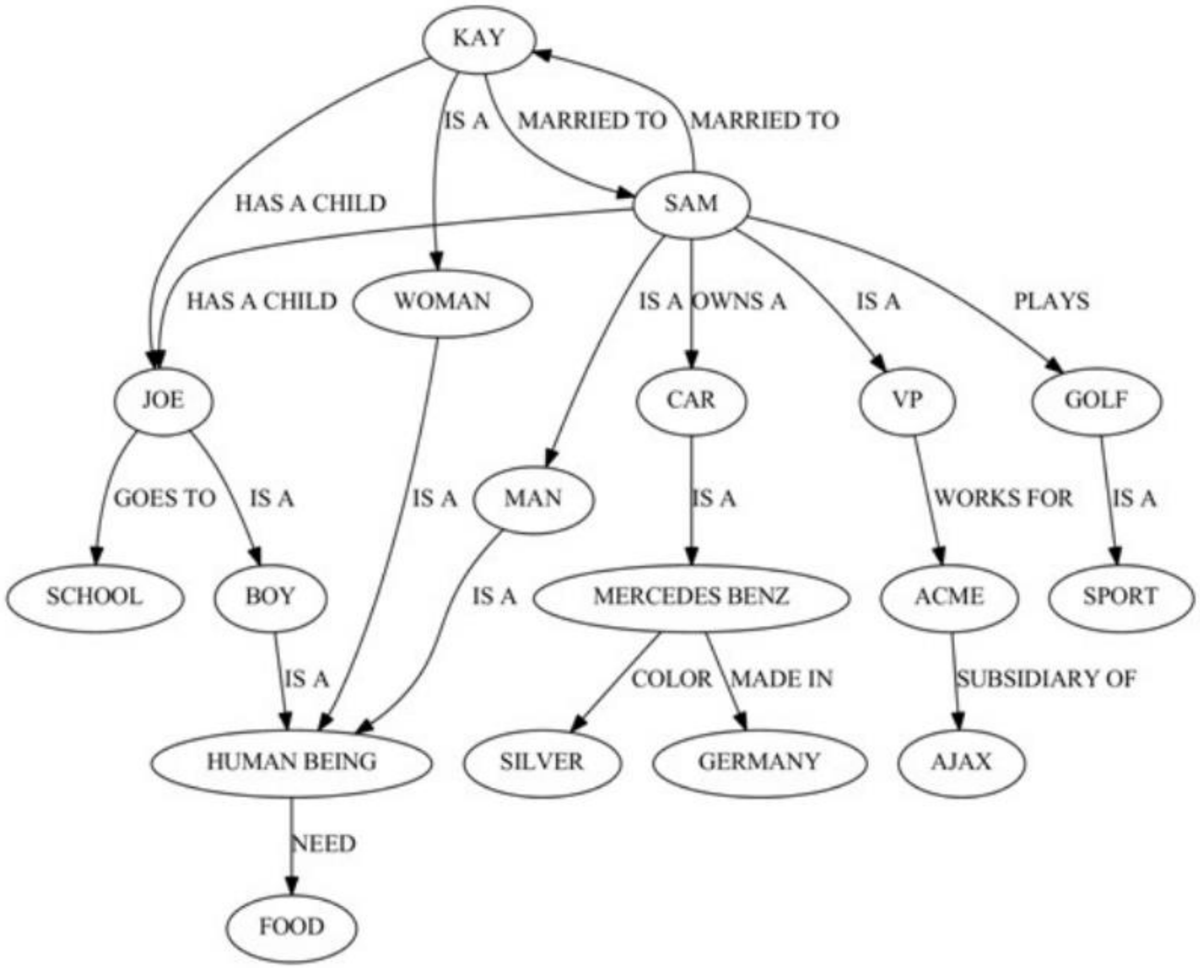


Credit: Khola Naseem

Knowledge representation

Semantic Network Representation

- Example:



Knowledge representation

Semantic Network Representation

- **Disadvantage:**
  - Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions.
  - it might be possible in the worst case scenario that after traversing the entire tree, we find that the solution does not exist in this network

Prolog(PROgramming in LOGic):

- we defines facts and rules and give this to the logic program
- Prolongs is the most widely used language to have been inspired by logic programming research.
- Some features:
  - Prolog uses logical variables. These are not the same as variables in other languages. Programmers can use them as ‘holes’ in data structures that are gradually filled in as computation proceeds.
  - It will look and reason, using available facts and rules, and then tells us an answer (or answers)

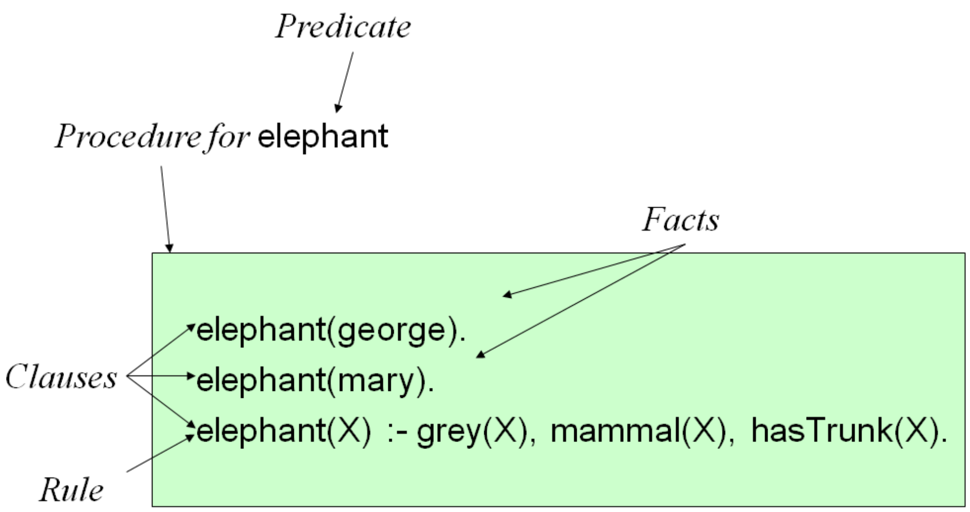
Prolog(PROgramming in LOGic):

- Prolog is a ‘declarative’ language
- Clauses are statements about what is true about a problem, instead of instructions how to accomplish the solution.
- An **atom(constant)** can contain letters, numbers, +, -, _, *, /, <, >, :, ., ~, &
- AN ATOM CANNOT START WITH _
- Variable start with Capital letter

	symbol
if	:-
and	,
or	;

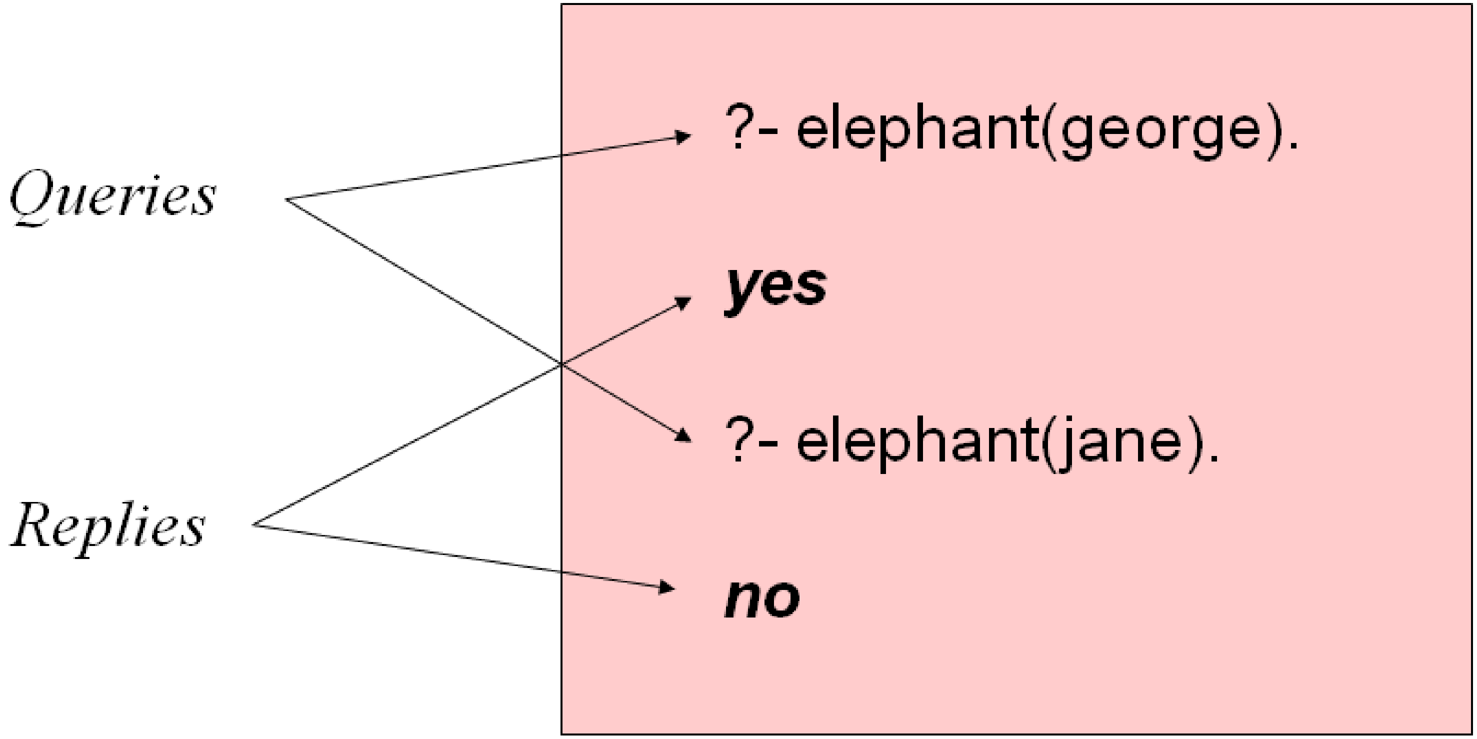
Prolog(PROgramming in LOGic):

- Structure of the program
  - Programs consist of procedures.
  - Procedures consist of clauses.
  - Each clause is a fact or a rule.
  - Programs are executed by posing queries.



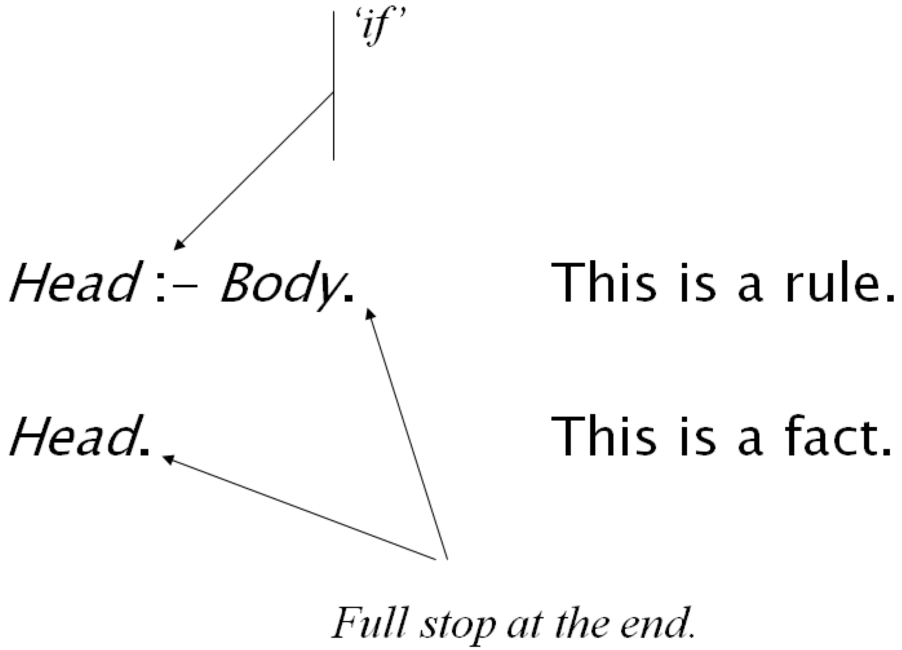
Prolog(PROgramming in LOGic):

- Structure of the program
  - queries.



Prolog(PROgramming in LOGic):

- Clauses: Facts and Rules.
- This is a rule where :- (if) says if the item on the right is
- true, then so is the item on the left



Prolog(PROgramming in LOGic):

- Basic data structure is term or tree.
- Prolog does not distinguish between inputs and outputs. It solves relations/predicates.
- Facts: ()
  - likes(john,mary).
  - likes(john,X). % Variables begin with capital
- Queries
  - ?- likes(X,Y).
  - X=john, Y=Mary.
  - ?- likes(X,X).
  - X=john.

Prolog(PROgramming in LOGic):

- Rule:
- Rules are used when you want to say that a fact depends on a group of facts

```
happy(albert).
happy(alice).
happy(bob).
happy(bill).
with_albert(alice).

runs(albert) :- happy(albert).
```

- Output:

```
?- runs(albert).
true.
```

Prolog(PROgramming in LOGic):

- Rule:
- Rules are used when you want to say that a fact depends on a group of facts
- Conjunction(and)
  - Comma is used for and

```
dances(alice) :- happy(alice), with_albert(alice).
```

- Output:

Prolog(PROgramming in LOGic):

- Rules
  - likes(ali,X) :- likes(X,ali). (:- = if)
  - likes(ali,X):- cat(X), likes(X,ali).
- Note: variables are dummy. Standarized apart
- Some Facts:
  - kitten(suzy).
  - likes(suzy,ali).
  - like(ali,X):-kitten(X), likes(X,ali).
- Query: ? - likes(ali,suzy).

```
?- like(ali,suzy).
true.
```

Prolog(PROgramming in LOGic):

- Rules and facts

```
father(a,b).
father(e,d).
mother(c,b).
mother(d,f).
parent(X,Y) :- father(X,Y).
parent(X,Y) :- mother(X,Y).
grandfather(X,Y):- father(X,Z),parent(Z,Y).
```

- Query:

```
?- grandfather(e,f).
true.
```

Prolog(PROgramming in LOGic):

- Rules and facts

```
father(a,b).
father(e,d).
mother(c,b).
mother(d,f).
parent(X,Y) :- father(X,Y).
parent(X,Y) :- mother(X,Y).
grandfather(X,Y):- father(X,Z),parent(Z,Y).
```

- Query:

```
?- grandfather(e,f).
true.
```

- Grandfater(e,f):-father(e,Z),parent(Z,f)
- Parent(d,f):-mother(d,f)

Prolog(PROgramming in LOGic):

- Rules and facts

  Tom is a cat.

  Tom caught a bird.

  Tom is owned by John.

  Tom is ginger in colour.

  Cats like cream.

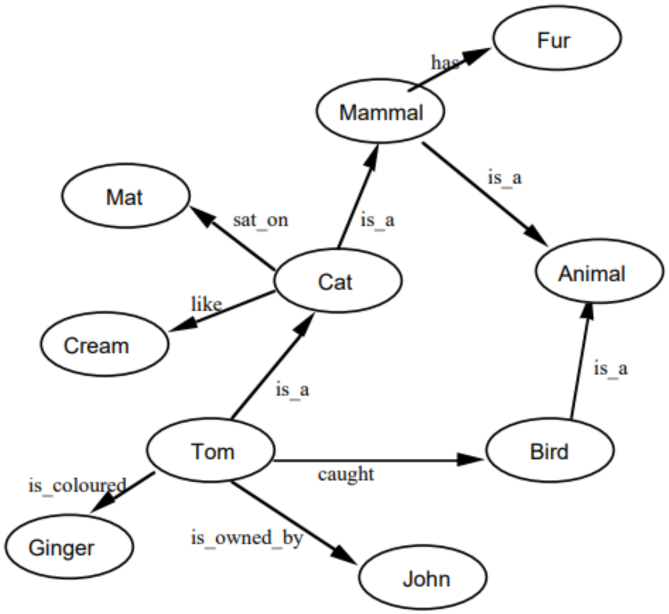
  The cat sat on the mat.

  A cat is a mammal.

  A bird is an animal.

  All mammals are animals.

  Mammals have fur.



- Tom has fur

Prolog(PROgramming in LOGic):

- Rules and facts

```
cat(tom).
cat(cat1).
mat(mat1).
sat_on(cat1,mat1).
bird(bird1).
caught(tom,bird1).
like(X,cream) :- cat(X).
mammal(X) :- cat(X).
has(X,fur) :- mammal(X).
animal(X) :- mammal(X).
animal(X) :- bird(X).
owns(john,tom).
is_coloured(tom,ginger).
```