# Knowledge Representation

Dr. Keerthy A S

# Knowledge Representation

- Issues
- History of AI representational schemes
- Conceptual Graphs
- Alternatives to explicit Representation
- Agent based and distributed problem solving.

# What is knowledge representation?

- Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
- Information representing 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.
- KR is not just storing data into some database, but also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

# What to Represent?

- **Object:** All the facts about objects in our world domain. E.g., Guitars contain strings; trumpets are brass instruments.
- **Events:** Events are the actions that occur in our world.
- **Performance:** It describes behavior that 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 and what we 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 Sentences (technical terms and not identical to the English language).
- **Knowledge** is awareness or familiarity gained by experiences of facts, data, and situations.

# Types of knowledge


- **5. Structural knowledge:**
  - It describes relationships between various concepts such as kind of, part of, and grouping of something.
  - It describes the relationship that exists between concepts or object

**1. Declarative Knowledge:**

- To know about something - concepts, facts, and objects.
- Expressed in declarative sentences & simpler than procedural language.

**2. Procedural Knowledge/Imperative.**

- Responsible for knowing how to do something.
- can be directly applied to any task & includes rules, strategies, procedures, agendas, etc.
- depends on the task on which it can be applied.

**3. Meta-knowledge:**

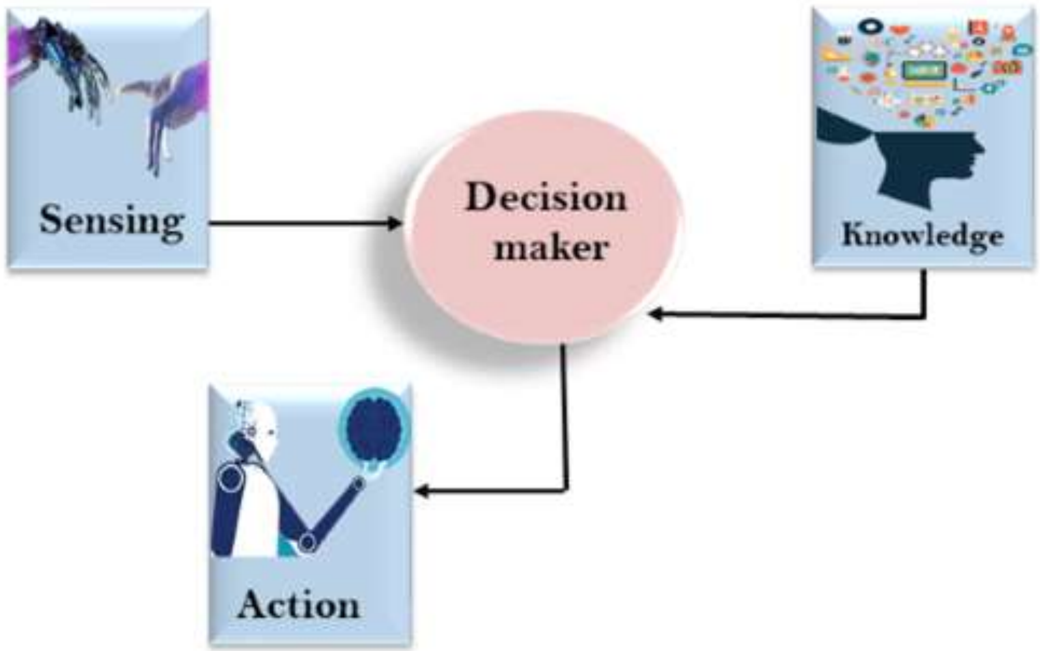
- Knowledge about the other types of knowledge

**4. Heuristic knowledge:**

- Representing knowledge of some experts in a field or subject.
- Rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

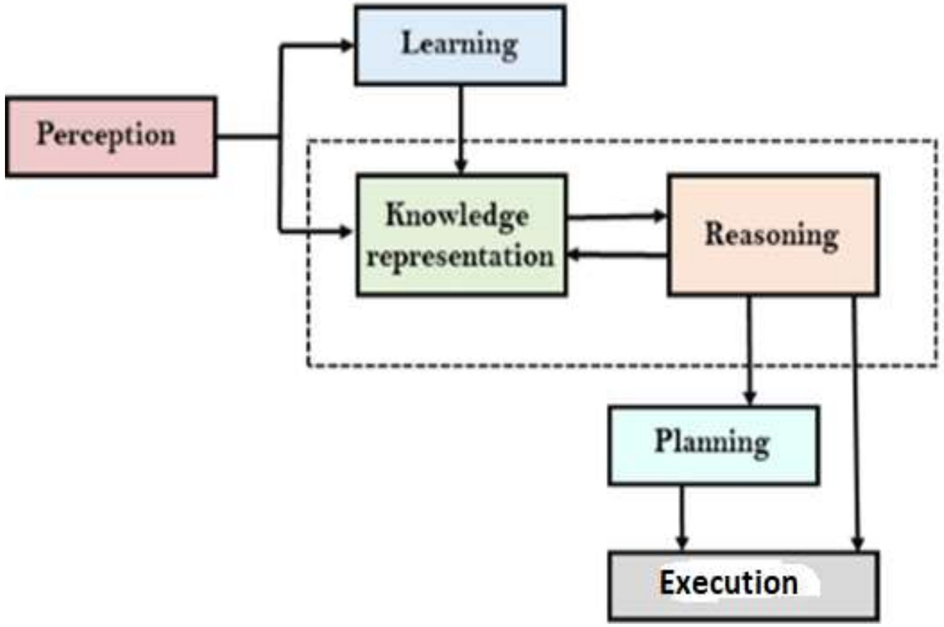
# Relation between knowledge and intelligence

- Knowledge of real-worlds plays a vital role in intelligence.
- Knowledge plays an important role in demonstrating intelligent behavior in AI agents. An agent is only able to accurately act on some input when he has some knowledge or experience about that input.




- The decision-maker acts by sensing the environment and using knowledge.
- if the knowledge part is not present then, it cannot display intelligent behavior.

# AI knowledge cycle




- AI system has a **Perception** component by which it retrieves information from its environment. It can be visual, audio, or another form of sensory input.
- The **learning** component is responsible for learning from data captured by Perception comportment.
- The main components are **knowledge representation** and **Reasoning**. These two components are involved in showing the intelligence of machine-like humans. These two components are independent of each other but also coupled together.
- The **planning** and **execution** depend on the analysis of Knowledge representation and reasoning.

# Approaches to knowledge representation

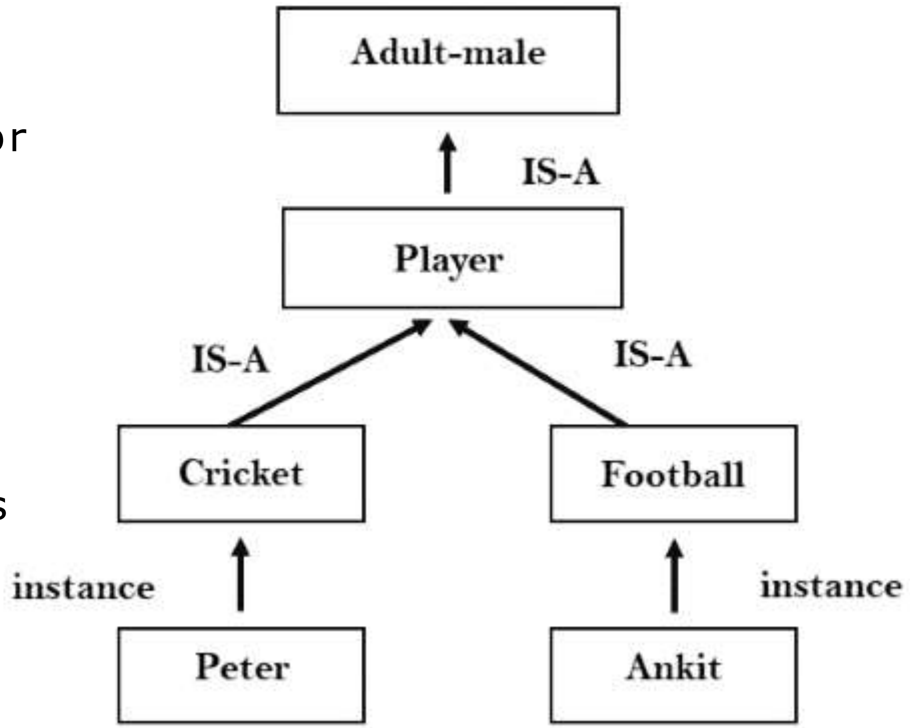
# 1. Simple relational knowledge

- It is the simplest way of storing facts which uses the relational method, and each fact about a set of objects is set out systematically in columns.
- This approach of knowledge representation is famous in database systems where the relationship between different entities is represented.
- This approach has little opportunity for inference.

| Player | Weight | Age |
| --- | --- | --- |
| Player1 | 65 | 23 |
| Player2 | 58 | 18 |
| Player3 | 75 | 24 |

# 2. Inheritable knowledge

- In the inheritable knowledge approach, all data must be stored in a hierarchy of classes.
- All classes should be arranged in a generalized form or a hierarchal manner.
- Apply inheritance property, elements inherit values from other members of a class.
- Shows a relation between instance and class, and it is called instance relation.
- Every individual frame can represent the collection of attributes and their value.



# 3. Inferential knowledge

- The inferential knowledge approach represents knowledge in the form of formal logic.
- This approach can be used to derive more facts.
- It guaranteed correctness.
- **Example:** Let's suppose there are two statements:
  - Marcus is a man
  - All men are mortal

**man(Marcus)**

**$\forall$x, man (x) ----------> mortal (x)**

# 4. Procedural knowledge

- Procedural knowledge approach uses small programs and codes which describe how to do specific things, and how to proceed.
- In this approach, one important rule is used which is the **If-Then rule**.
- With this knowledge, we can use various coding languages such as **LISP language** and **Prolog language**.
- We can easily represent heuristic or domain-specific knowledge using this approach.
- But it is not necessary that we can represent all cases in this approach.

# Requirements for knowledge Representation system

**1. Representational Accuracy:**

KR system should have the ability to represent all kinds of required knowledge.

**2. Inferential Adequacy:**

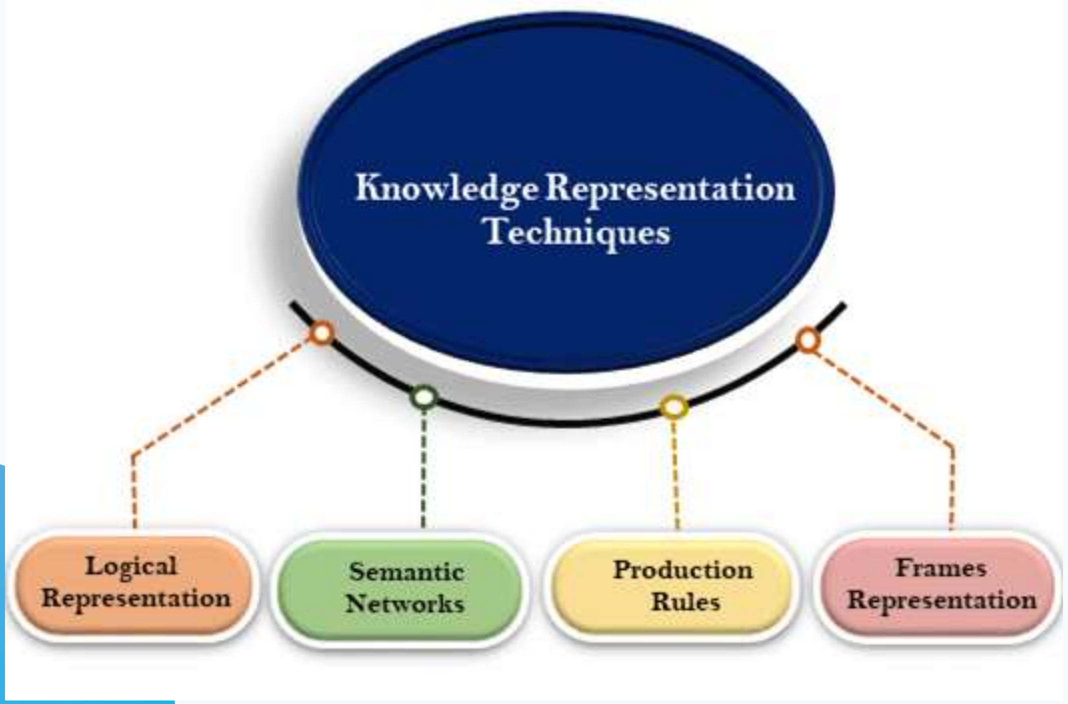
KR system should have the ability to manipulate the representational structures to produce new knowledge corresponding to the existing structure.

**3. Inferential Efficiency:**

The ability to direct the inferential knowledge mechanism in the most productive directions by storing appropriate guides.

**4. Acquisition efficiency-** The ability to acquire new knowledge easily using automatic methods.

# Techniques of knowledge representation



# 1. Logical Representation

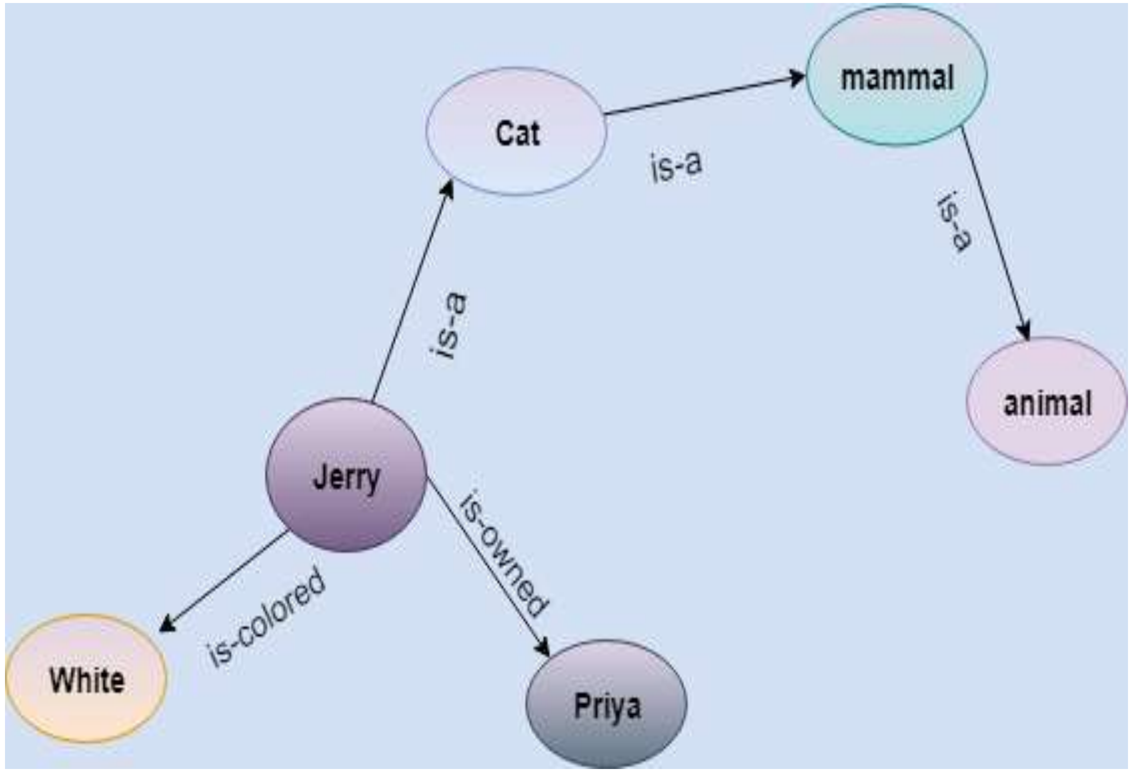
- Logical representation is a language with some concrete rules which deal with propositions and has no ambiguity in representation; drawing conclusions based on various conditions.
- Lays down important communication rules and consists of precisely defined syntax and semantics which supports sound inference.
- Each sentence can be translated into logic using syntax and semantics.
- Syntax:
  - Syntaxes are the rules which decide how we can construct legal sentences in 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 logic& assigning a meaning to each sentence.
- Logical representation can be categorized into
  - Propositional Logics
  - Predicate logics

# 2. Semantic Network Representation

- An alternative of predicate logic for knowledge representation.
- Represent knowledge in the form of graphical networks.
- The network consists of nodes representing objects and arcs which describe the relationship between those objects.
- Categorize the object in different forms and link those objects.
- Semantic networks are easy to understand and can be easily extended.
- Consists of mainly two types of relations:
    - IS-A relation (Inheritance)
    - Kind-of-relation

# Semantic Network Representation

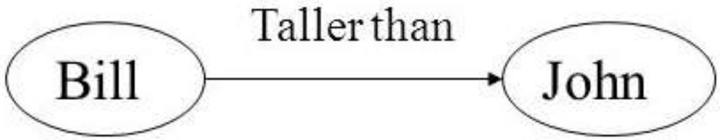
- Statements:
  - Jerry is a cat.
  - Cat is a mammal
  - Jerry is owned by Priya.
  - Jerry is white colored.
  - All Mammals are animal.



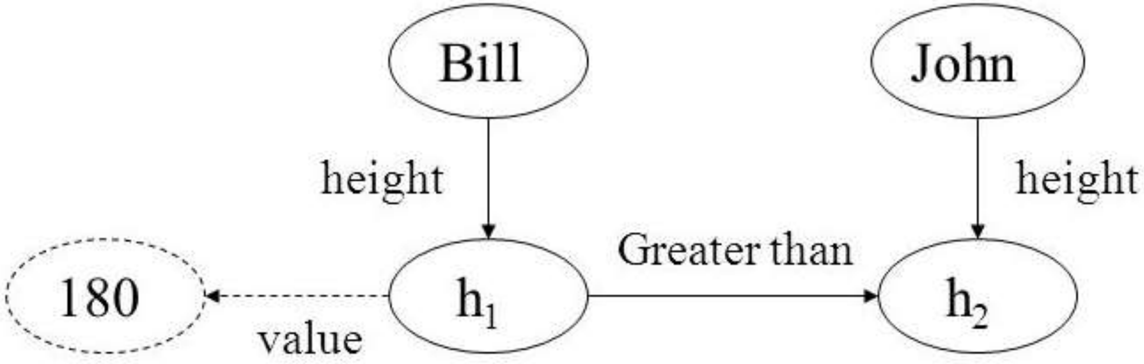
# Semantic Network Representation

"Bill is taller than John ."

- Non appropriate scheme :




- Appropriate scheme :

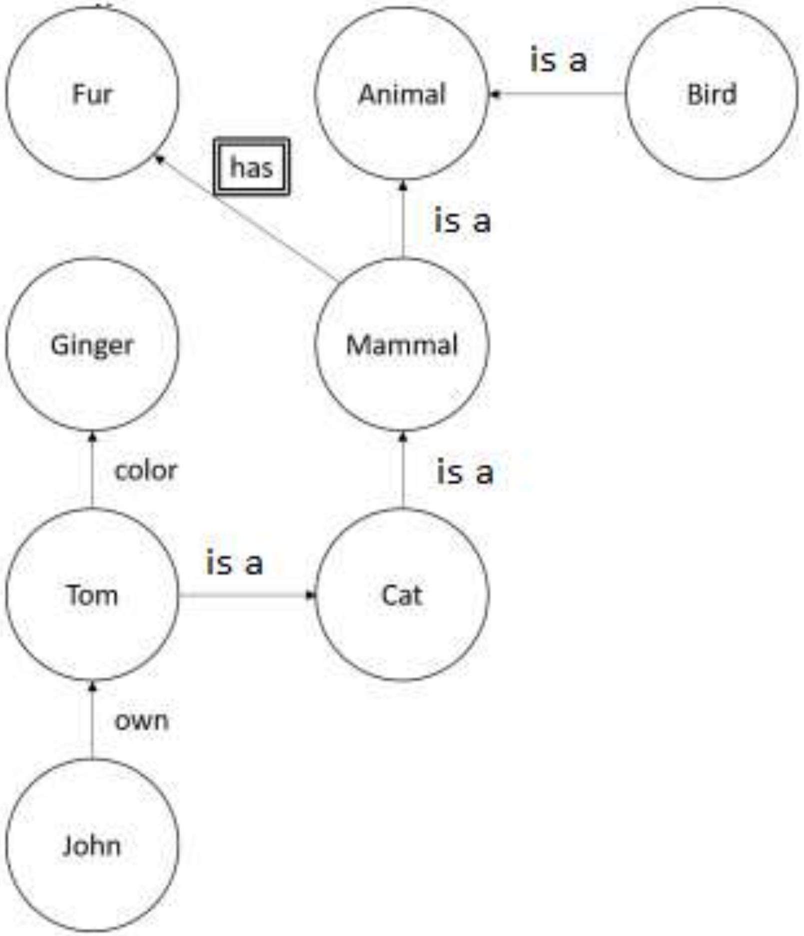


# Semantic Network Representation

- Mammals have fur.
- All mammals are animals.
- A bird is an animal.
- A cat is a mammal.
- Tom is a cat.
- Tom is owned by John.
- Tom is ginger in color.

# Semantic Network Representation

- Mammals have fur.
- All mammals are animals.
- A bird is an animal.
- A cat is a mammal.
- Tom is a cat.
- Tom is owned by John.
- Tom is ginger in color.



# Semantic Network

**Drawbacks:**

- Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions.
- Semantic networks try to model human-like memory to store information, but in practice, it is not possible to build such a vast semantic network.
- These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
- Semantic networks do not have any standard definition for link names.
- Networks are not intelligent and depend on the creator of the system.

**Advantages of Semantic network:**

- Semantic networks are a natural representation of knowledge.
- Semantic networks convey meaning in a transparent manner.
- They are simple and easily understandable.

# 3. Frame Representation

- A frame is a record-like structure that consists of a collection of attributes and their values to describe an entity in the world.
- Frames divide knowledge into substructures by representing stereotypical situations.
- It consists of a collection of slots of type and size, and slot values which are called facets.
- Facets are features of frames that enable us to put constraints on the frames.
- **slot-filter knowledge representation -** 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.
- Frames system consist of a collection of frames which are connected

# Examples

1: Frame for a book

2: Peter is a doctor by profession, and his age is a 25 year old bachelor and weighs 78kg.

| Slots | Filters |
|---|---|
| **Title** | Artificial Intelligence |
| **Genre** | Computer Science |
| **Author** | Peter Norvig |
| **Edition** | Third Edition |
| **Year** | 1996 |
| **Page** | 1152 |

| Slots | Filter |
|---|---|
| **Name** | Peter |
| **Profession** | Doctor |
| **Age** | 25 |
| **Marital status** | Single |
| **Weight** | 78 |

# Frame Representation

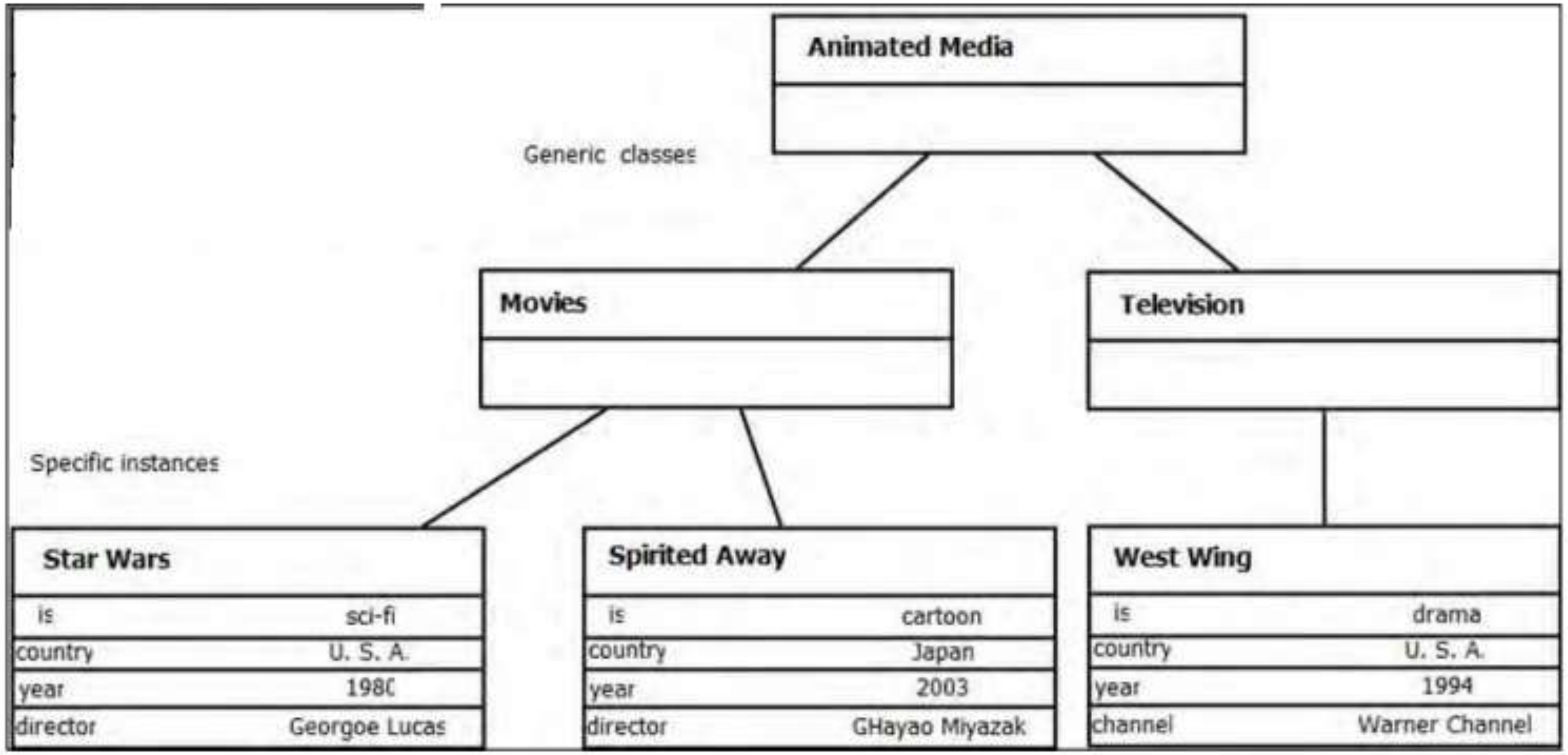
**Advantages:**

- The frame knowledge representation makes the programming easier by grouping the related data.
- The frame representation is comparably flexible and used by many applications in AI.
- It is very easy to add slots for new attributes and relations.
- It is easy to include default data and search for missing values.
- Frame representation is easy to understand and visualize.

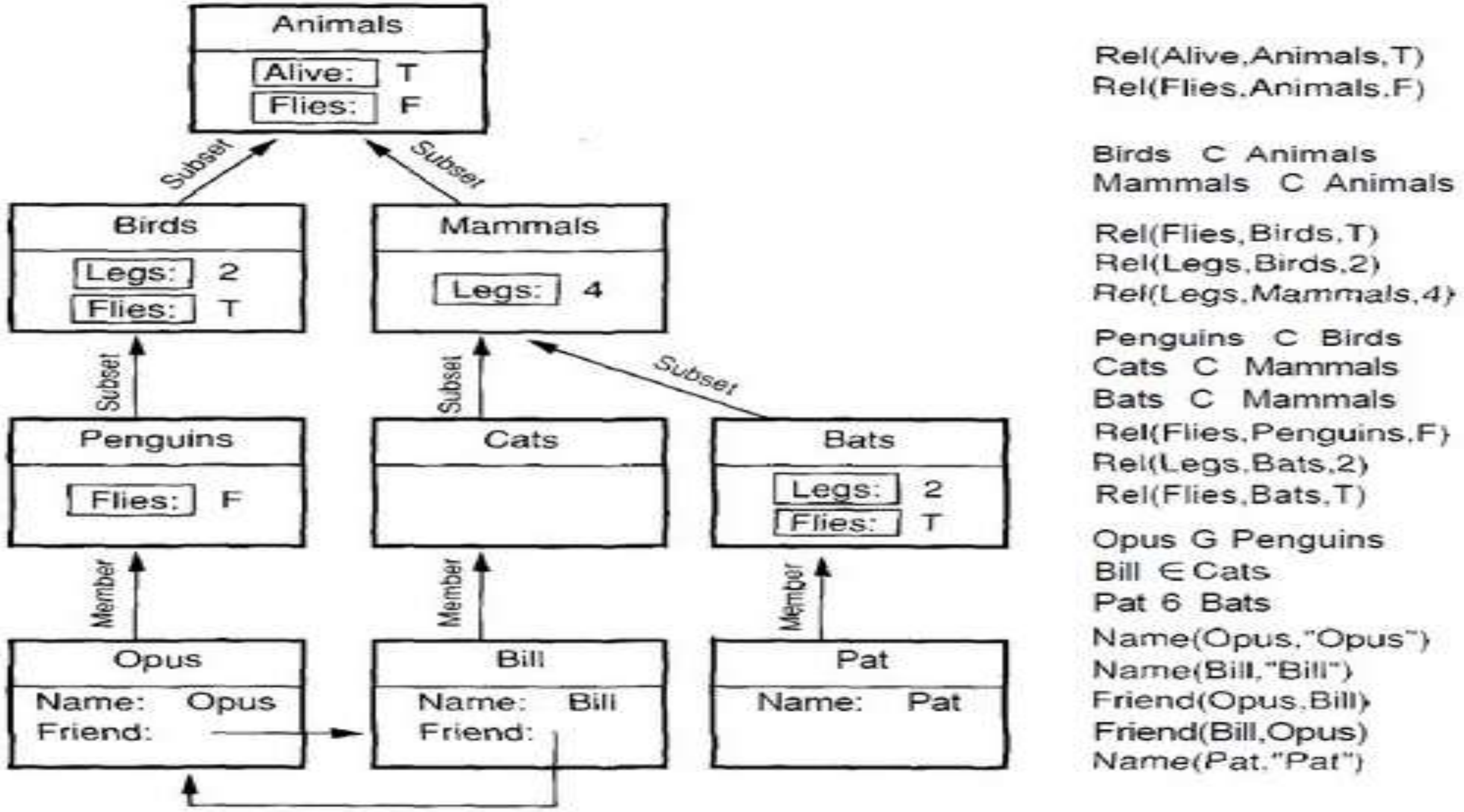
**Disadvantages:**

- In frame system inference mechanism is not easily processed.
- The inference mechanism cannot be smoothly proceeded by frame representation.
- Frame representation has a much more generalized approach.

# Frame Representation



# Frame Representation




**(a) A frame-based knowledge base** **(b) Translation into first-order logic**

# 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
    - Working Memory
    - The recognize-act-cycle
- 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.
- The action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle.
- 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.
- 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.

# Production Rules

- Example:
- IF (at bus stop AND bus arrives) THEN action (get into the bus)
- IF (on the bus AND paid AND empty seat) THEN action (sit down).
- IF (on bus AND unpaid) THEN action (pay charges).
- IF (bus arrives at destination) THEN action (get down from the bus).

# Production Rules

Advantages :

- The production rules are expressed in natural language.
- The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages :

- The production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for future use.
- During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

# Scripts

- A *script* is a structure that prescribes a set of circumstances which could be expected to follow on from one another.
- A thought sequence or a chain of situations which could be anticipated.
- a number of slots or frames but with more specialized roles.
- Scripts are beneficial because:
    - Events tend to occur in known runs or patterns.
    - Causal relationships between events exist.
    - Entry conditions exist which allow an event to take place
    - Prerequisites exist upon events taking place.
- *E.g.* when a student progresses through a degree scheme or when a purchaser buys a house.

**Components:**

- **Entry Conditions** - these must be satisfied before events in the script can occur.
- **Results**- Conditions that will be true after events in script occur.
- **Props**- Slots representing objects involved in events.
- **Roles**- Persons involved in the events.
- **Track**- Variations on the script. Different tracks may share components of the same script.
- **Scenes**- The sequence of *events* that occur. *Events* are represented in *conceptual dependency* form.

# Robbing a bank

*Props*

- Gun, *G*.
- Loot, *L*.
- Bag, *B*
- Get away car, *C*.

*Roles* might be:

- Robber, R.
- Cashier, *M*.
- Bank Manager, *O*.
- Policeman, *P*.

Getting a gun.

- Hold up a bank.
- Escape with the money.

*Entry Conditions*

- R is poor.
- R is destitute.

*Results*

- R has more money.
- *O* is angry.
- *M* is in a state of shock.
- *P* is shot.

**Script: ROBBERY** *Track: Successful Snatch*

*Props:*
G = Gun,
L = Loot,
B= Bag,
C = Get away car.

*Roles:*
R = Robber,
M = Cashier,
O = Bank Manager,
P = Policeman.

*Entry Conditions:*
R is poor.
R is destitute.

*Results:*
R has more money.
O is angry.
M is in a state of shock.
P is shot.

*Scene 1: Getting a gun*

R PTRANS R into Gun Shop
R MBUILD R choice of G
R MTRANS choice.
R ATRANS buys G

(go to scene 2)

*Scene 2 Holding up the bank*

R PTRANS R into bank
R ATTEND eyes M, O and P
R MOVE R to M position
R GRASP G
R MOVE G to point to M
R MTRANS "Give me the money or ELSE" to M
P MTRANS "Hold it Hands Up" to R
R PROPEL shoots G
P INGEST bullet from G
R ATRANS L to M
R ATRANS L puts in bag, B
R PTRANS exit
O ATRANS raises the alarm

(go to scene 3)

*Scene 3: The getaway*

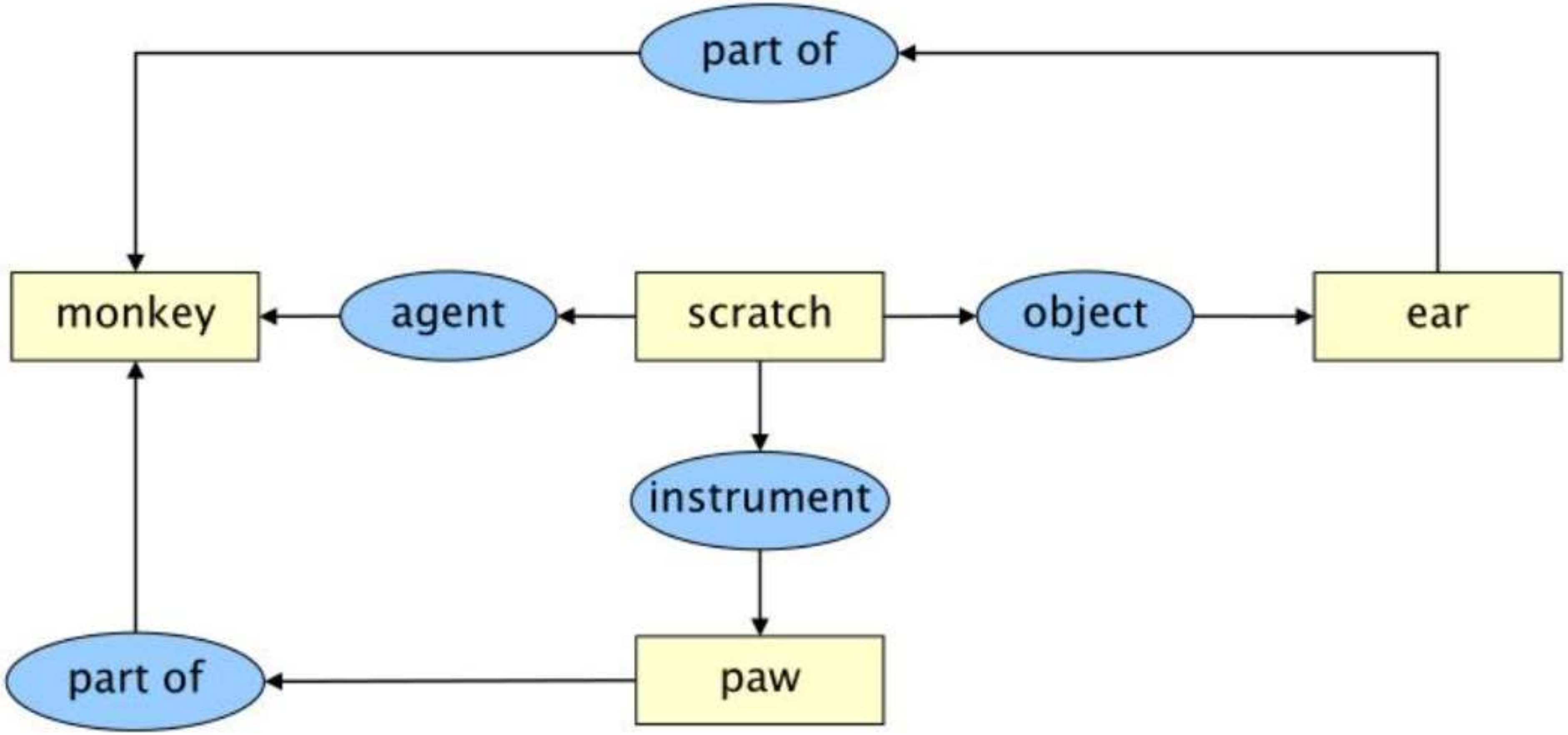
R PTRANS C

## Conceptual Graphs

- Network knowledge representation schema
- rooted in the association theory of meaning
- very much used in the problem of natural language processing
- A conceptual Graph is a complete bipartite-oriented graph, where each node is either a concept or a relation between two concepts, there is one or two edges each going to concepts, and each concept may represent another conceptual graph

# Conceptual Graphs

*A monkey scratches its ear with a paw*



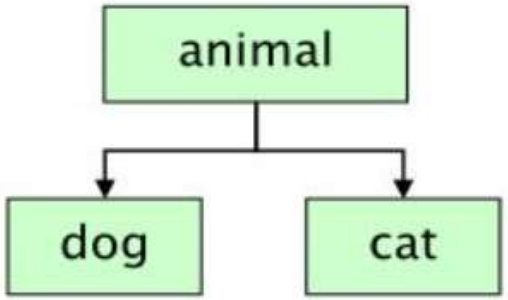
## Conceptual Graphs

- each concept has got its type and an instance

  general concept – a concept with a wildcard instance


  specific concept – a concept with a concrete instance


- there exists a hierarchy of types subtype:




- concept w is specialisation of concept v if type(v)>type(w) or instance(w)::type(v)

# Conceptual Graphs

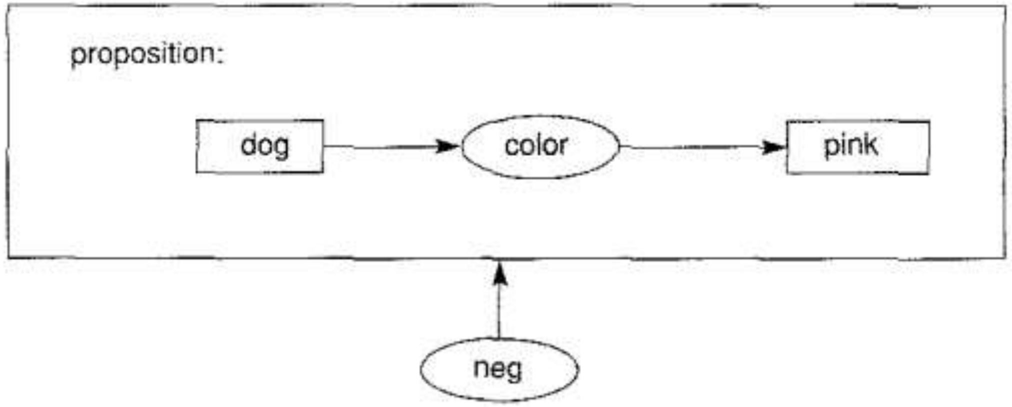
- FOPL transformation to CG
  - for each node → predicate
  - general concept → variable, specific concept → atom
    type:instance → type(instance)
  - relation → n-ary predicat relation(in1, in2, …, inn) with arguments conncecting neighbouring concepts
  - CG is existencionally quantified conjunction of these predicates


$$\exists\ X\ (dog(emma)\ \wedge\ color(emma,X)\ \wedge\ brown(X))$$

$$\exists X \exists Y\ (dog(X) \wedge color(X,Y) \wedge brown(Y)).$$

# Conceptual Graphs and Logic

$$\forall X \forall Y (\neg (dog(X) \wedge color(X,Y) \wedge pink(Y))).$$

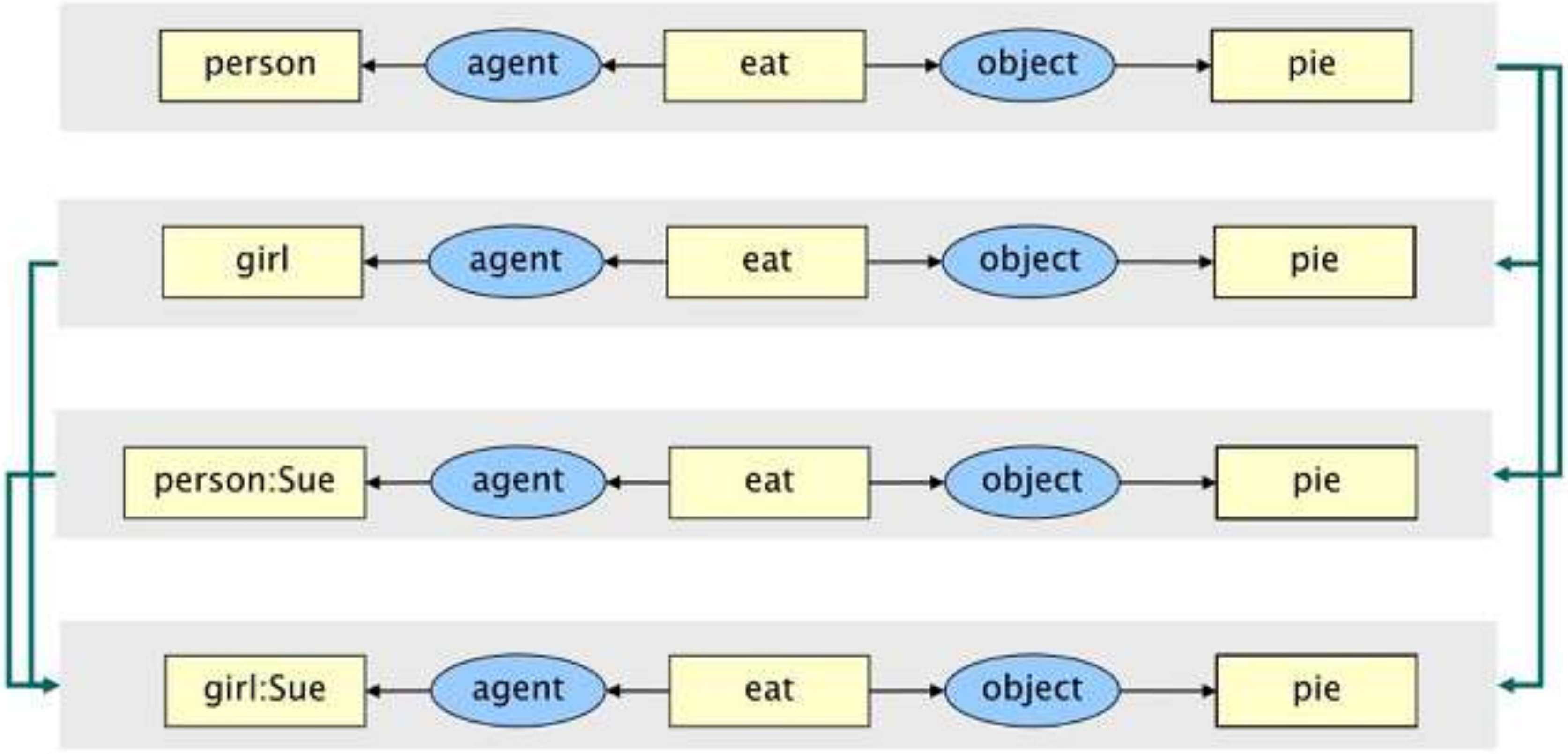



There are no pink dogs.

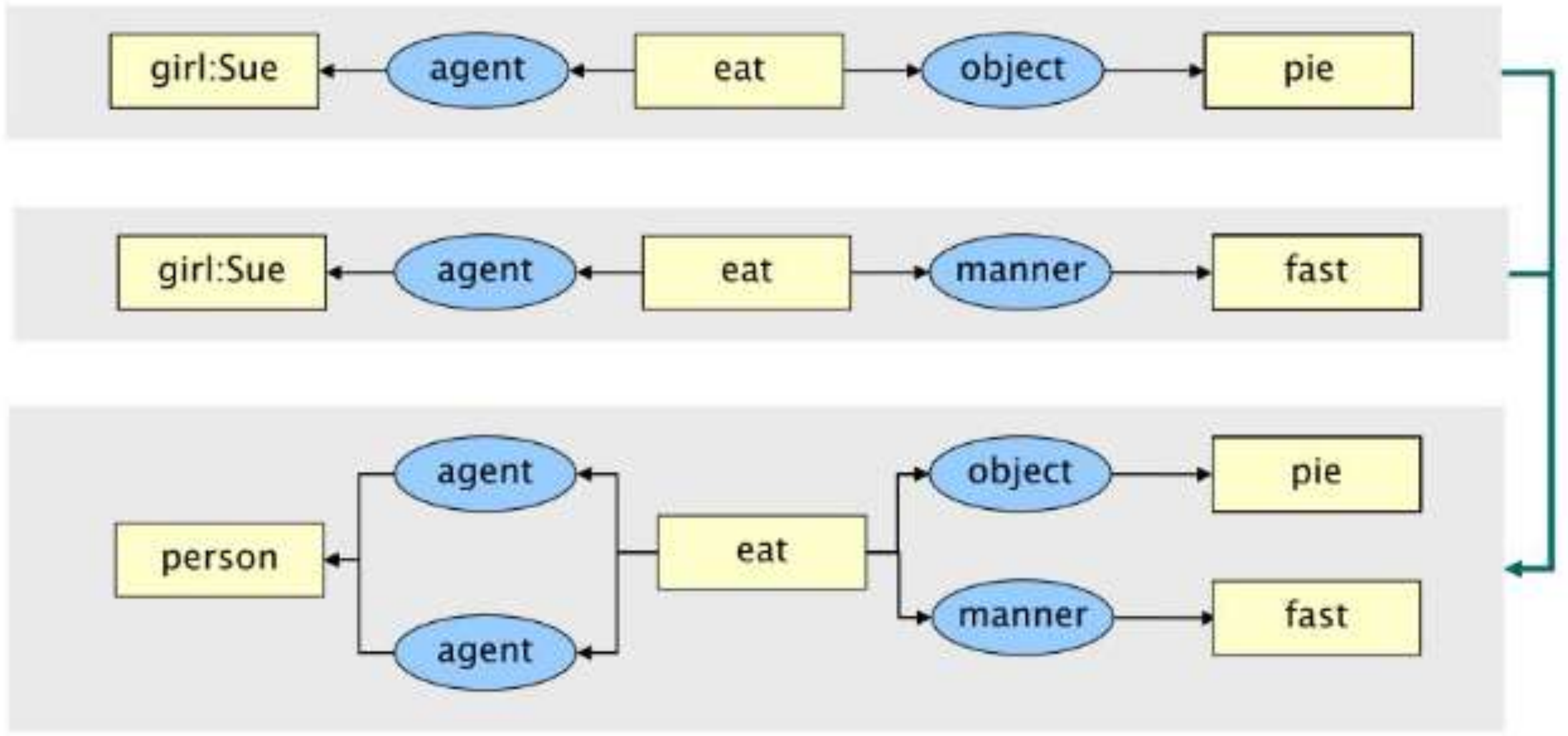
# Conceptual Graph: Generalization and Specialization

- canonic conceptual graph is sensible representation of knowledge that can be but does not necessary need to be true
- canonic formation rules formalise rules of inference between two graph for while preserving canonicity
  - copy – identical cloning of a graph
  - restriction – substituting a concept in a graph with its specialisation
  - join – joining two graphs via shared concept
  - simplification – deleting identical relations

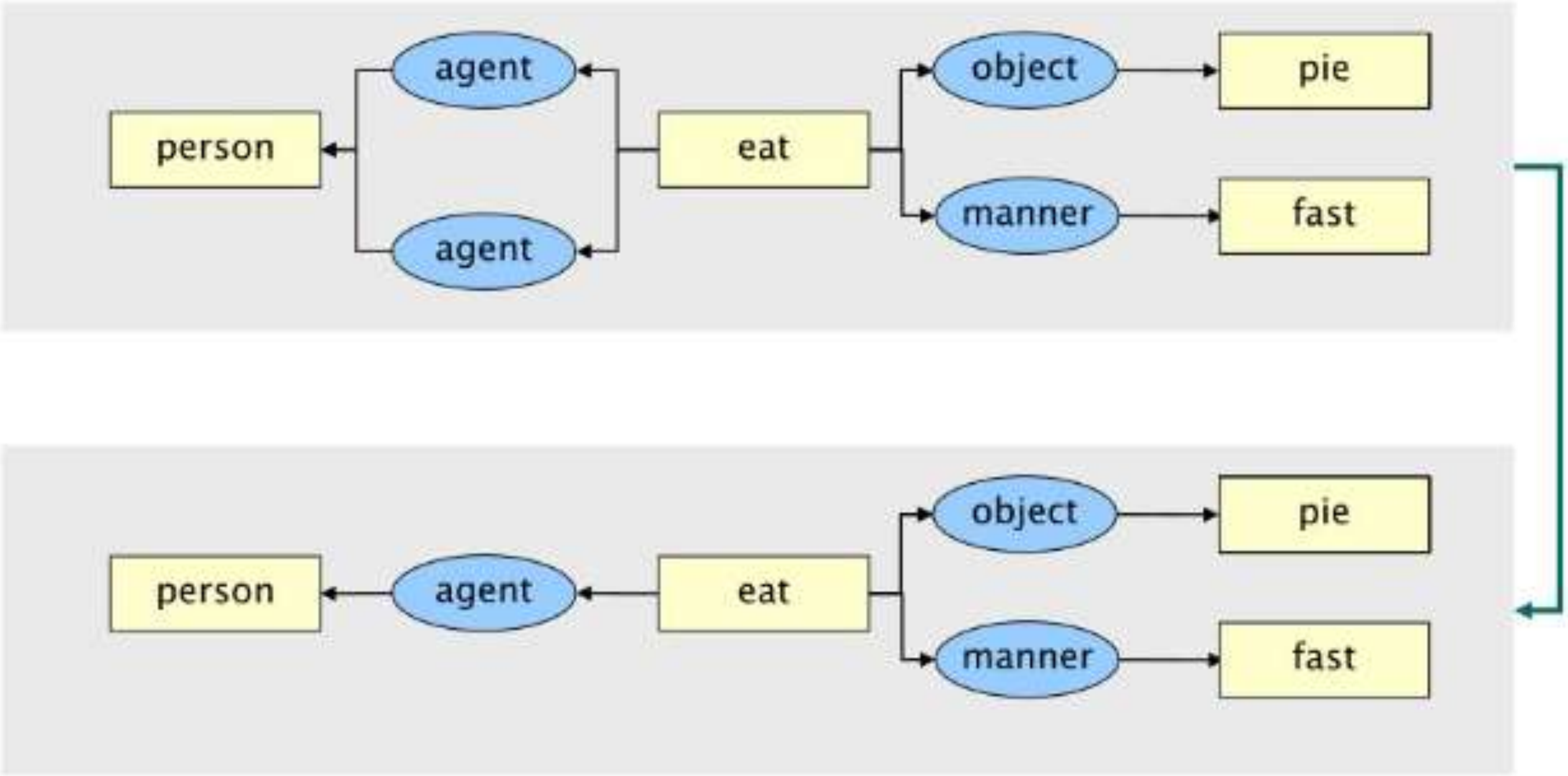
# Restriction of Concepts



# Joining Concepts



# Simplification of Concepts



# Issues in knowledge representation

**1. Important attributes**
There are two attributes shown in the diagram, **instance** and **isa.** Since these attributes support property of inheritance, they are of prime importance.

**2. Relationships among attributes**
Basically, the attributes used to describe objects are nothing but entities. However, the attributes of an object do not depend on the encoded specific knowledge.

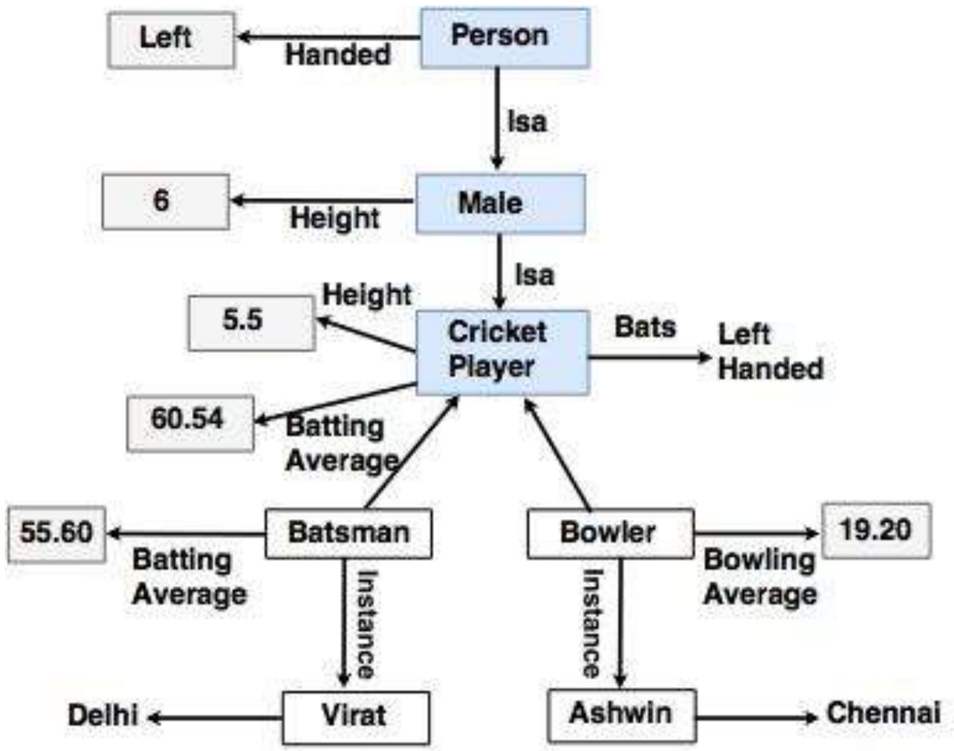



Fig: Inheratable Knowledge Representation

# Issues in knowledge representation

**3. Choosing the granularity of representation**

While deciding the granularity of representation, it is necessary to know the following:

**i.** What are the primitives and at what level should the knowledge be represented?

**ii.** What should be the number (small or large) of low-level primitives or high-level facts?

High-level facts may be insufficient to draw the conclusion while Low-level primitives may require a lot of storage.

**For example:** Suppose that we are interested in the following facts: **John spotted Alex**.

=> represented as **"Spotted (agent(John), object (Alex))"**

Such a representation can make it easy to answer questions such as: Who spotted Alex?

Suppose we want to know : "Did John see Sue?‟, Given only one fact, the user cannot discover that answer.

The user can add other facts, such as "Spotted (x, y) → saw (x, y)"

# Issues in knowledge representation

**4. Representing sets of objects:** Some properties of objects which satisfy the condition of a set together but not as individual.

**Example:** "There are more sheep than people in Australia", and "English speakers can be found all over the world.‟ These facts can be described by including an assertion to the sets representing people, sheep, and English.

**5. Finding the right structure as needed**
To describe a particular situation, it is always important to find access to the right structure. This can be done by selecting an initial structure and then revising the choice.

While selecting and reversing the right structure, it is necessary to solve the following problem statements.
They include the process on how to select an initial appropriate structure:

- Fill in the necessary details from the current situation.
- Determine a better structure if the initially selected structure is not appropriate to fulfill other conditions.
- Find the solution if none of the available structures is appropriate.
- Create and remember a new structure for the given condition.
- There is no specific way to solve these problems, but some of effective knowledge representation techniques have the potential to solve them.

# Alternatives to Explicit Representation

## Subsumption architecture (Brooks, MIT)

- claim: intelligence is the product of the interaction between an appropriately layered system and its environment
- architecture is a collection of task-handling behaviors, with each behavior accomplished via a finite state machine
- limited feedback between layers of behavior
- "… in simple levels of intelligence, explicit representations and models of the world simply get in the way. It turns out to be better to use the world as its own model." (Brooks)

## Copycat architecture (Mitchell & Hofstadter, Indiana)

- builds on representation techniques from semantic nets, blackboards, connectionist networks, and classifier systems
- supports semantic net-like representation that can evolve
- emphasizes analogical reasoning

# Agent-Based and Distributed Problem Solving

- Agent-based models are **computer simulations used to study the interactions between people, things, places, and time**.
- A multi-agent system is a computer program with problem solvers situated in interactive environments, which are each capable of flexible, autonomous, socially organized actions that can be, but need not be, directed towards predetermined objectives or goals.

## Features of Agent

- Situatedness:-the agent receives input from the environment in which it is active and can also effect changes within that environment
- Autonomous:- An agent can interact with its environment without the direct intervention of other agents.
- Flexible:-A flexible agent is both intelligently responsive as well as proactive depending on its current situation. A **responsive** agent receives stimuli from its environment and responds to them in an appropriate and timely fashion. A **proactive** agent does not simply respond to situations in its environment but is also able to be opportunistic, goal directed, and have appropriate alternatives for various situations.
- Social:-An agent can interact, as appropriate, with other software or human agents.

# Multi-agent Systems vs Distributed agent Systems

- Multi-agent systems are ideal for representing problems that include many problem-solving methods, multiple viewpoints, and multiple entities.
- multi-agent system refers to all types of software systems composed of multiple semi-autonomous components.
- The distributed agent system considers how a particular problem can be solved by a number of modules (agents) which cooperate by dividing and sharing the knowledge about the problem and its evolving solution.