KNOWLEDGE REPRESENTATION

What is knowledge?

- facts, information, and skills acquired through experience or education; the theoretical or practical understanding of a subject.
- Knowledge = information + rules

TYPES OF KNOWLEDGE

- There are 5 types of knowledge:
- 1) Procedural knowledge
- 2) Declarative knowledge
- 3) Meta knowledge
- 4) Heuristic knowledge
- 5) Structural knowledge

1)Procedural Knowledge

- Gives information/ knowledge about how to achieve something.
- Describes how to do things provides set of directions of how to perform certain tasks.
- **Procedural knowledge**, also known as imperative **knowledge**, is the **knowledge** exercised in the performance of some task.
- It depends on targets and problems.
- Example
- How to drive a car?

2)Declarative knowledge

- Its about statements that describe a particular object and its attributes , including some behavior in relation with it.
- *"Can this knowledge be true or false*?"
- It is non-procedural, independent of targets and problem solving.
- Example
- It is sunny today and chemise are red.

3)Meta Knowledge

- It's a knowledge about knowledge and how to gain them.
- Example
- The knowledge that blood pressure is more important for diagnosing a medical condition then eyes color.

4)Heuristic Knowledge

- Representing knowledge of some expert in a field or subject.
- Rules of thumb.
- Heuristic Knowledge are sometimes called shallow knowledge.
- Heuristic knowledge are empirical as opposed to deterministic.

5)Structural Knowledge

- Describes what relationship exists between concepts/ objects.
- Describe structure and their relationship.
- Example
- How to various part of car fit together to make a car, or knowledge structures in term of concepts, sub concepts and objects.

Knowledge Representation

- **What to Represent?**

Let us first consider what kinds of knowledge might need to be represented in AI systems:

- **Objects**
  - -- Facts about objects in our world domain. *e.g.* Guitars have strings, trumpets are brass instruments.
- **Events**
  - -- Actions that occur in our world. *e.g.* Steve played the guitar in Frank's Band.

- **Performance**
  - -- A behavior like *playing the guitar* involves knowledge about how to do things.

- **Meta-knowledge**
  - -- knowledge about what we know.

Thus in solving problems in AI we must represent knowledge and there are two entities to deal with:

- **Facts**
  - truths about the real world and what we represent. This can be regarded as the *knowledge level*
- **Representation of the facts**
  - which we manipulate. This can be regarded as the *symbol level* since we usually define the representation in terms of symbols that can be manipulated by programs.

- We can structure these entities at two levels
- **the knowledge level**
  - -- at which facts are described
- **the symbol level**
  - -- at which representations of objects are defined in terms of symbols that can be manipulated in programs.

There are multiple approaches and scheme that comes to mind when we begin to think about representation.

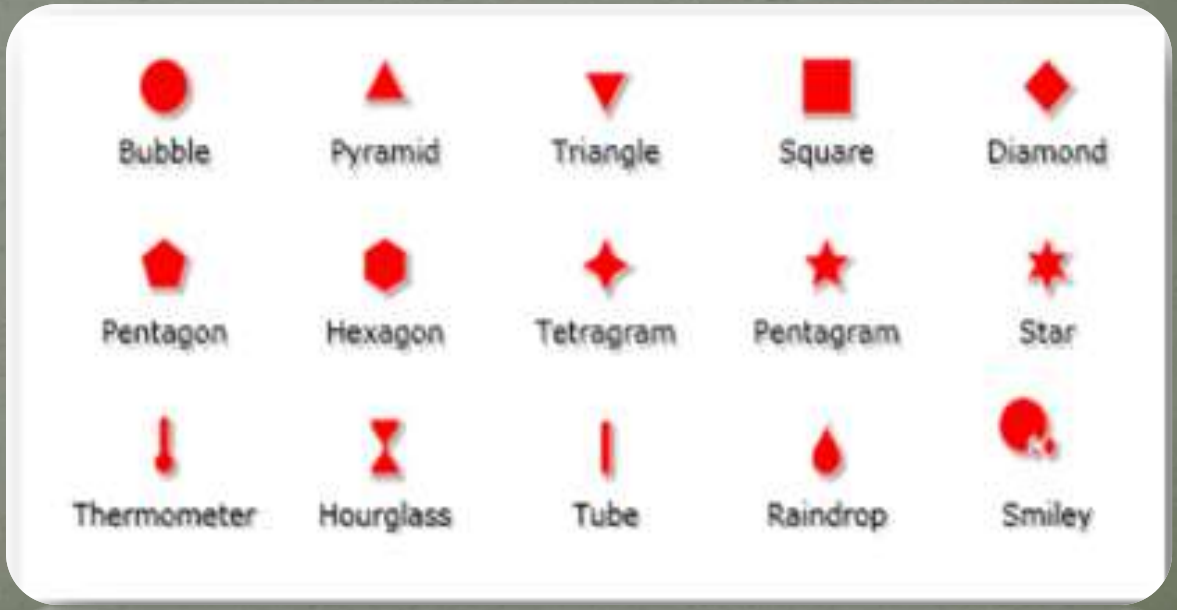
1)Pictures and symbols

2)Graphs and network

3)Numbers

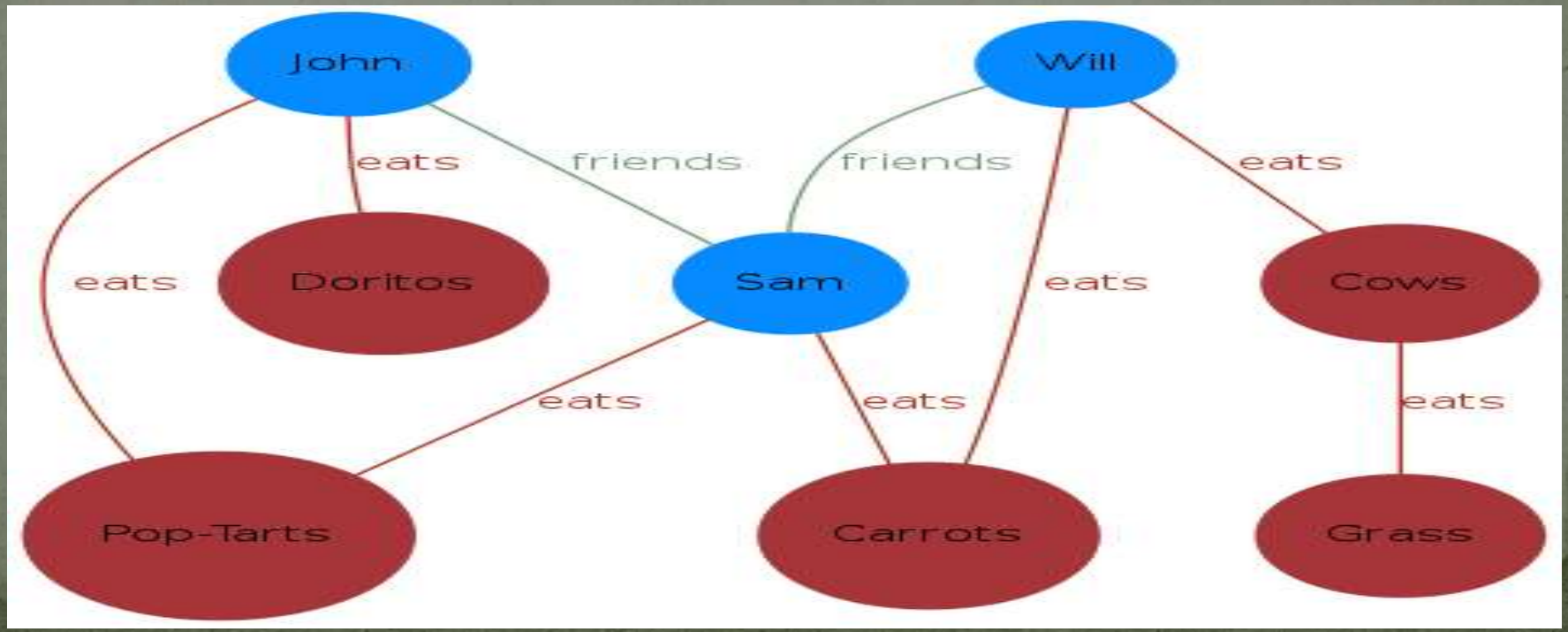
1)Pictures and symbols

- Pictorial representation are not easily translate to useful information because computer can't interpret pictures directly with out complex reasoning.
- Pictures are useful for human understanding.



2)Graph and network

- Allows relationship between objects to be incorporated.
- We can represent procedural knowledge using graphs.



3)Numbers

- Numbers are an integral part of knowledge representation used by humans.
- Numbers translate easily to computer representation.

- English or natural language is an obvious way of representing and handling facts.
- Logic enables us to consider the following fact:

  spot is a dog as

  dog(spot)

  We could then infer that all dogs have tails with:

  Vx : *dog(x)* → *hasatail(x)*

  We can then deduce:

  hasatail(Spot)

Using Knowledge (1)

We have briefly mentioned where knowledge is used in AI systems. Let us consider a little further to what applications and how knowledge may be used.

- **Learning** -- acquiring knowledge. This is more than simply adding new facts to a knowledge base.

  New data may have to be *classified* prior to storage for easy *retrieval, etc.*

  Duplication should be avoided.

- **Reasoning** -- Infer facts from existing data.

Using Knowledge(2)

If a system only knows:

- Davis is a Jazz Musician.
- All Jazz Musicians can play their instruments well.
- If things like *Is Davis a Jazz Musician?* or *Can Jazz Musicians play their instruments well?* are asked then the answer is readily obtained from the data structures and procedures.
- However a question like *Can Davis play his instrument well?* requires reasoning.

Properties for Knowledge Representation Systems

The following properties should be possessed by a knowledge representation system.

- **Representational Adequacy**
  - -- the ability to represent the required knowledge;
- **Inferential Adequacy**
  - - the ability to manipulate the knowledge represented to produce new knowledge corresponding to that inferred from the original;
- **Inferential Efficiency**
  - - the ability to direct the inferential mechanisms into the most productive directions by storing appropriate guides;
- **Acquisitional Efficiency**
  - - the ability to acquire new knowledge using automatic methods wherever possible rather than reliance on human intervention.

To date no single system optimises all of the above

Approaches to Knowledge Representation

Simple relational knowledge

- The simplest way of storing facts is to use a relational method where each fact about a set of objects is set out systematically in columns. This representation gives little opportunity for inference, but it can be used as the knowledge basis for inference engines.
- Simple way to store facts.
- Each fact about a set of objects is set out systematically in columns.
- Little opportunity for inference.
- Knowledge basis for inference engines.

Figure: Simple Relational Knowledge

Musician	Style	Instrument	Age
Miles Davis	Jazz	Trumpet	deceased
John Zorn	Avant Garde	Saxophone	35
Frank Zappa	Rock	Guitar	deceased
John Mclaughlin	Jazz	Guitar	47

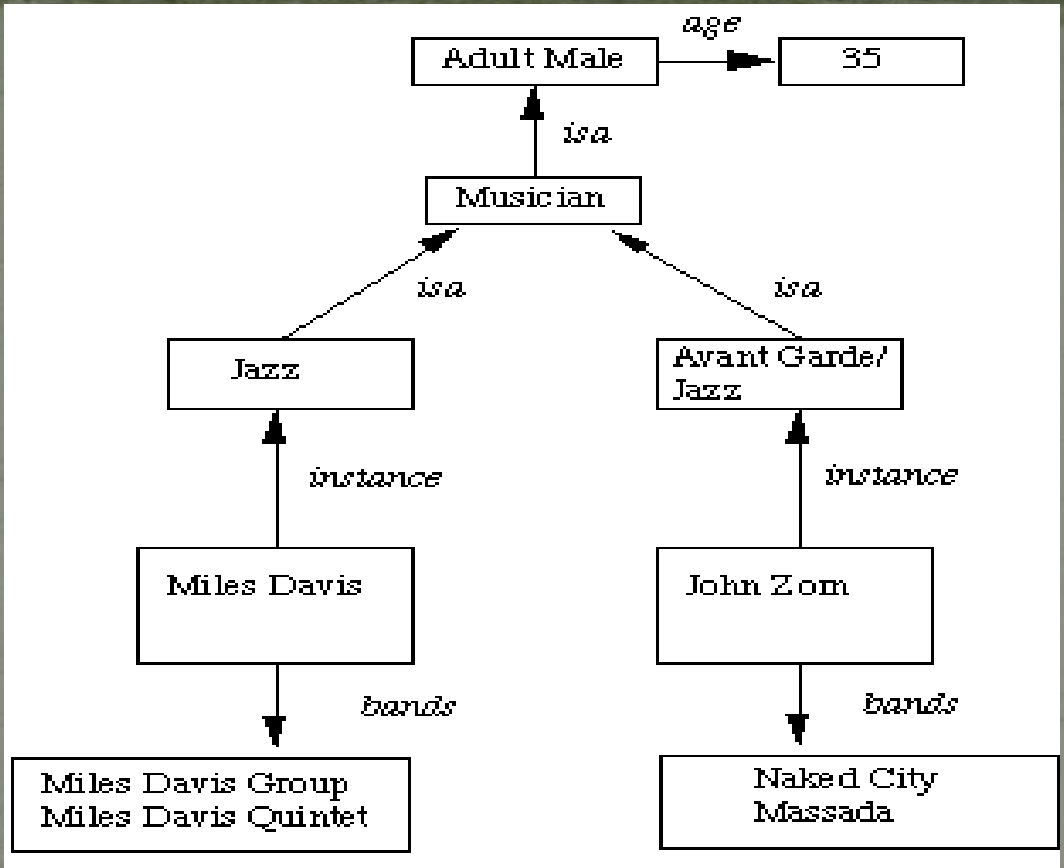
Inheritable knowledge

Relational knowledge is made up of objects consisting of

- attributes
- corresponding associated values.

We extend the base more by allowing inference mechanisms:

- **Property inheritance**
  - **elements inherit values from being members of a class.**
  - **data must be organized into a hierarchy of classes.**



Boxed nodes -- objects and values of attributes of objects.
Values can be objects with attributes and so on.
Arrows -- point from object to its value.
This structure is known as a **slot and filler structure, semantic network or a collection of frames.**

Inferential Knowledge

Represent knowledge as *formal logic*:

- *All dogs have tails*

V(x) : *dog(x)* → *hasatail(x)*

Advantages:

- A set of strict rules.
  - Can be used to derive more facts.
  - Truths of new statements can be verified.
  - Guaranteed correctness.
- Popular in AI systems. *e.g* Automated theorem proving

Procedural Knowledge

Basic idea:

- Knowledge encoded in some procedures
  - small programs that know how to do specific things, how to proceed.

Advantages:

- *Heuristic* or domain specific knowledge can be represented.
- *Extended logical inferences*, such as default reasoning facilitated.
- *Side effects* of actions may be modeled. Some rules may become false in time. Keeping track of this in large systems may be tricky.

What is Learning?

Learning is an important area in AI, perhaps more so than planning.

- Problems are hard -- harder than planning.
- A goal of AI is to enable computers that can be taught rather than programmed.
- *Learning* is a an area of AI that focuses on processes of self-improvement.
- Recognized Solutions are not as common as planning.

Why is it hard?

- Intelligence implies that an organism or machine must be able to adapt to new situations.
- It must be able to learn to do new things.
- This requires knowledge acquisition, inference, updating/refinement of knowledge base, acquisition of heuristics, applying faster searches, *etc.*

How can we learn?

- Many approaches have been taken to attempt to provide a machine with learning capabilities. This is because learning tasks cover a wide range of phenomena.

- Listed below are a few examples of how one may learn.

- **Skill refinement** -- one can learn by practicing
- **Knowledge acquisition** -- one can learn by experience and by storing the experience in a knowledge base.
- **Taking advice** -- Similar to rote learning although the knowledge that is input may need to be transformed (or *operationalised*) in order to be used effectively.

- **Problem Solving** -- if we solve a problem one may learn from this experience. The next time we see a similar problem we can solve it more efficiently. This does not usually involve gathering new knowledge but may involve reorganization of data or remembering how to achieve to solution.
- **Induction** -- One can learn from *examples*.
- **Discovery** -- Here one learns knowledge without the aid of a teacher.
- **Analogy** -- If a system can recognize similarities in information already stored then it may be able to transfer some knowledge to improve to solution of the task in hand.

Rote Learning

Rote Learning is basically *memorization*.

- Saving knowledge so it can be used again.
- Retrieval is the only problem.
- No repeated computation, inference or query is necessary.

A simple example of rote learning is *caching*

- Store computed values (or large piece of data)
- Recall this information when required by computation.
- Significant time savings can be achieved.
- Many AI programs (as well as more general ones) have used caching very effectively.

Memorization is a key necessity for learning.

Store v Compute

Rote Learning must not decrease the efficiency of the system.

- We be must able to decide whether it is worth storing the value in the first place.
- Consider the case of multiplication -- it is quicker to recompute the product of two numbers rather than store a large multiplication table.

How can we decide?

- **Cost-benefit analysis** -- Decide when the information is first available whether it should be stored. An analysis could weigh up amount of storage required, cost of computation, likelihood of recall.

- **Selective forgetting** -- here we allow the information to be stored initially and decide later if we retain it. Clearly the frequency of reuse is a good measure. We could tag an object with its *time of last use*. If the cache memory is full and we wish to add a new item we remove the least recently used object.

Learning by Taking Advice

There are two basic approaches to advice taking:

- **Take high level, abstract advice** and convert it into rules that can guide performance elements of the system.
- ***Develop sophisticated tools* such as knowledge base editors and debugging.** These are used to aid an expert to translate his expertise into detailed rules.

Learning by Problem Solving

There are three basic methods in which a system can learn from its own experiences.

1. Learning by Parameter Adjustment

The basic idea of *parameter adjustment* is to:

- Start with some estimate of the correct weight settings.
- Modify the weight in the program on the basis of accumulated experiences.
- Here the problem has an evaluation function that is represented as a polynomial of the form such as:

  $c_1t_1+c_2t_2+c_3t_3+\ldots\ldots\ldots\ldots$

- The *t* terms a values of features and the *c* terms are weights.

- **2. Learning by Macro Operators**

  The basic idea here is similar to Rote Learning:

  Avoid expensive recomputation

  Macro-operators can be used to group a whole series of actions into one.

- **3. Learning by Chunking**

  Chunking involves similar ideas to Macro Operators and originates from psychological ideas on memory and problem solving.

Weak Slot and Filler Structures

Why use this data structure?

- It enables attribute values to be retrieved quickly
  - assertions are indexed by the entities
  - binary predicates are indexed by first argument. *E.g. team(Mike-Hall , Cardiff).*
- Properties of relations are easy to describe .
- It allows ease of consideration as it embraces aspects of object oriented programming.

- A *slot* is an attribute value pair in its simplest form.
- A *filler* is a value that a slot can take -- could be a numeric, string (or any data type) value or a pointer to another slot.

We will study two types:

- Semantic Nets.
- Frames.

The major idea is that:

- The meaning of a concept comes from its relationship to other concepts, and that,
- The information is stored by interconnecting nodes with labeled arcs.

4 types of knowledge representation in AI

- 1) Logical representation
- 2) Production rule
- 3) Semantic networks
- 4) Frame representation

1)LOGICAL REPRESENTATION

In order to give information to agent and get info without errors in communication.

Logic is based on truth.

There are 2 types of LR

1)propositional logic(PL)

2)first order predicate logic(FOL)

Propositional logic

- **Proposition :** A proposition is classified as a declarative sentence which is either true or false.

  eg: *1) It rained yesterday.*

- **Propositional symbols/variables**: P, Q, S, ... (**atomic sentences**)
- Sentences are combined by **Connectives**:

$\wedge$ ...and	[conjunction]
$\vee$ ...or	[disjunction]
$\Rightarrow$ ...implies	[implication / conditional]
$\Leftrightarrow$ ..is equivalent	[biconditional]
$\neg$ ...not	[negation]

First-order Predicate Logic

- First-order predicate calculus (FOPL) was developed by logicians to extend the expressiveness of Propositional Logic.
- It is generalization of propositional logic that permits reasoning about world entities (objects) as well as classes and subclasses of objects.
- Prolog is also based on FOPL.
- Predicate logic uses variables and quantifiers which is not present in propositional logic.

2) PRODUCTION RULE

- Consist of <condition,action>pairs.
- Agent check if a conditions holds then give a new situation(state).
- Production rule are belong to and same as propositional logic.

Example

- Not all that glitter is gold.
- ¬V(x)=Glitter(x)-›Gold(x)

3) Semantic Nets

❖**What is Semantic Net ?**

- **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.**
  - ***Nodes*** **are sometimes referred to as objects. Nodes are to represent physical objects, concepts, or situation.**
  - ***Arcs*** **as links or edges. The links are used to express relationships.**

- **It is also known as associative net due to the association of one node with other.**
- **Semantic networks are easy to understand and can be easily extended.**

Count…

- This representation consist of mainly two types of relations:

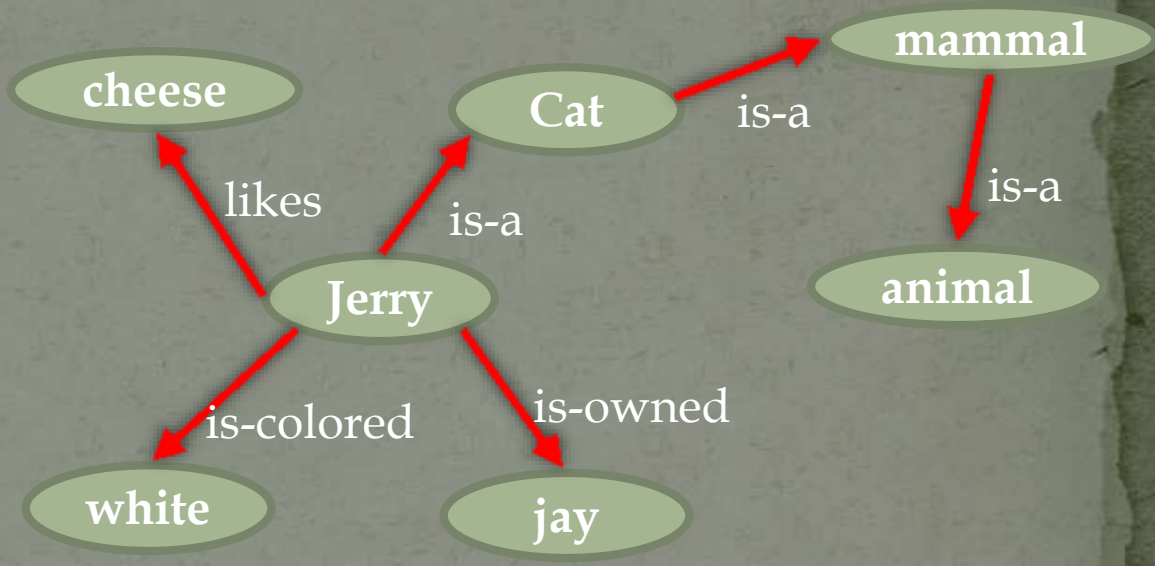
a) IS-A relation (Inheritance).
b) A-Kind-of relation (AKO).

a) IS-A relation :
   - IS-A means "is an instance of" and refers to a specific member of a class.
   - A class is related to the mathematical concept of a set in that it refers to a group of objects.

b) A-Kind-of relation :
   - The AKO relation is used to relate one class to another.
   - AKO relates generic nodes to generic nodes while the IS-A relates an instance or individual to a generic class.

Simple Example of Semantic net

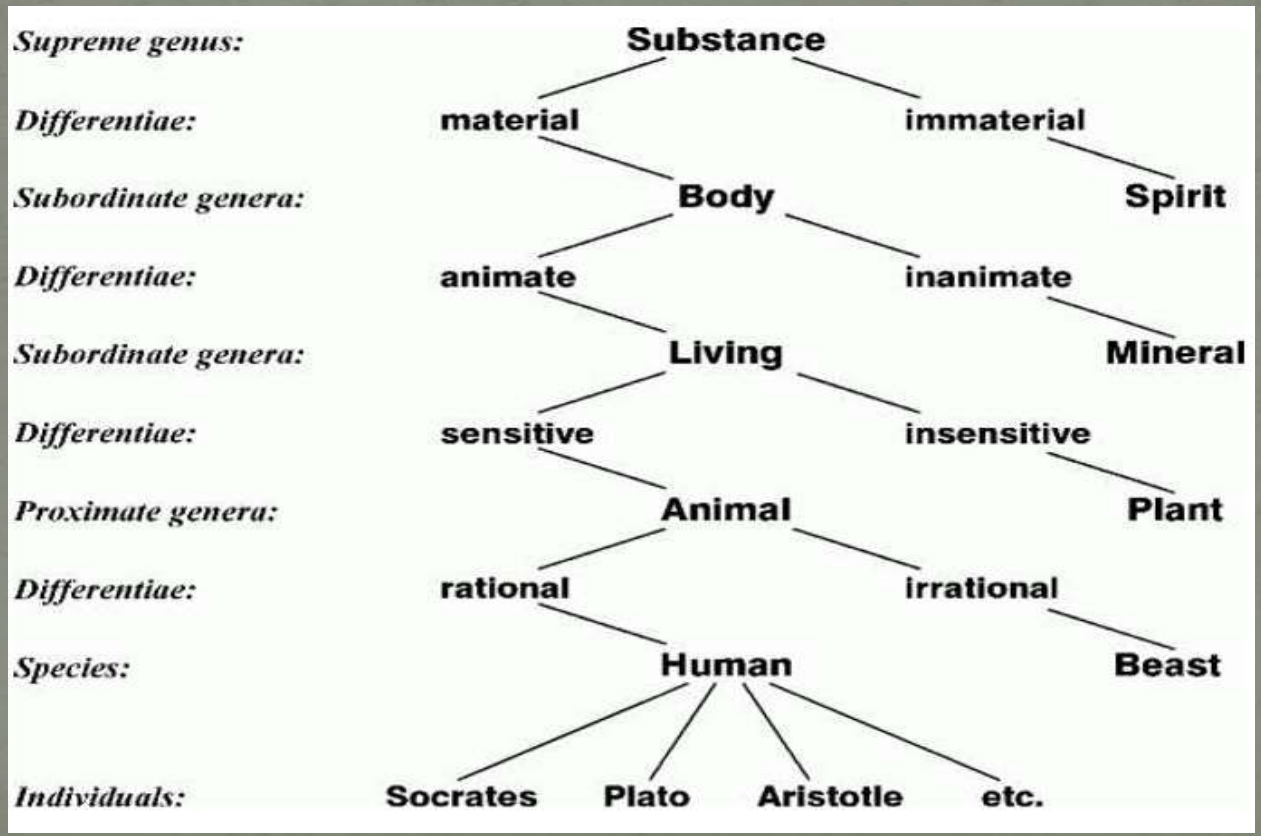
- Following are some statements which we need to represent in the form of nodes and arcs.
- Statements :
  1) Jerry is a cat.
  2) Jerry is a mammal.
  3) Jerry is owned by Jay.
  4) Jerry is white colored.
  5) All Mammals are animal.
  6) Jerry likes cheese.



- 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.

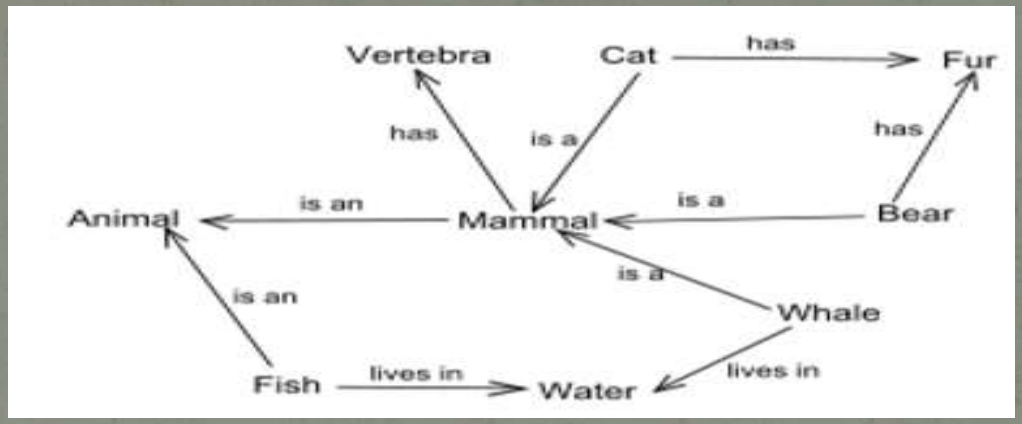
Kinds of Semantic Nets

- **Definitional Networks :** Emphasize the subtype or is-a relation between a concept type and a newly defined subtype.

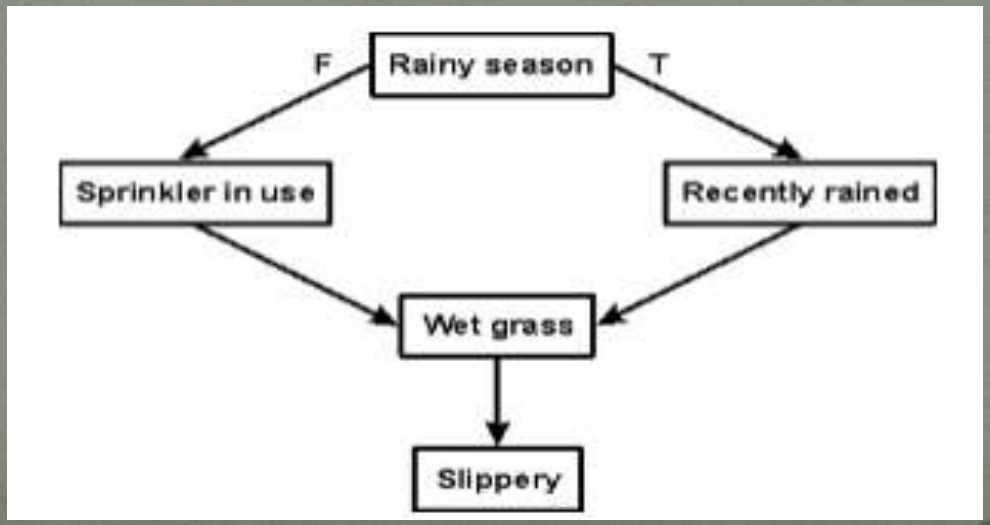


Kinds of Semantic Nets

- **Assertional Networks :** Designed to assert propositions.

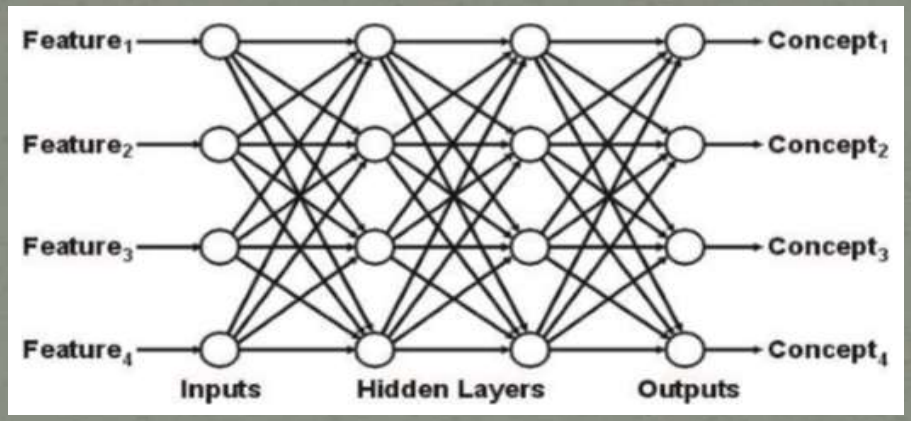


- **Implicational Networks :** Uses implication as the primary relation for connecting nodes.

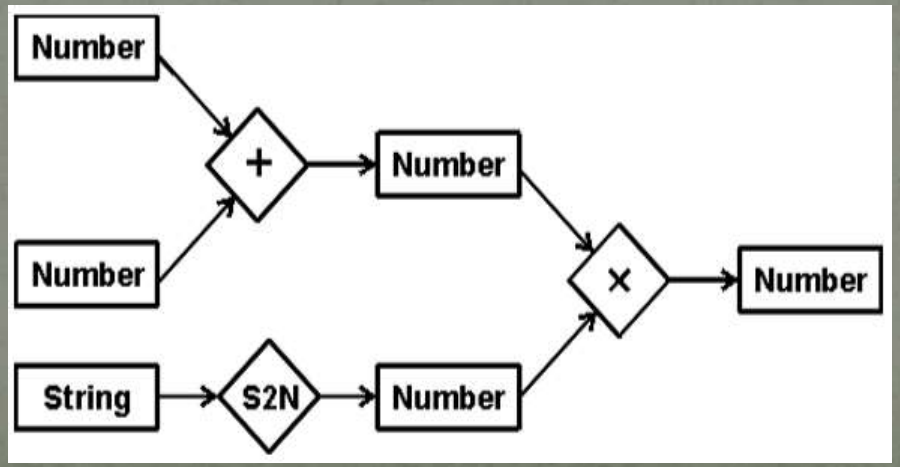


Kinds of Semantic Nets

- **Learning Networks** : Networks that build or extend their representations by acquiring knowledge from examples.



- **Executable Networks :** Contain mechanisms that can cause some change to the network itself.



Advantages of Semantic network

1) Semantic networks are a natural representation of knowledge.
2) Semantic networks convey meaning in a transparent manner.
3) These networks are simple and easily understandable.
4) Efficient.
5) Translatable to PROLOG w/o difficulty.

Disadvantages of Semantic network

1) 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.

2) These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.

3) Semantic networks do not have any standard definition for the link names.

4) These networks are not intelligent and depend on the creator of the system.

5) Semantic networks try to model human-like memory to store the information, but in practice, it is not possible to build such a vast semantic network.

Representation in a Semantic Net

The physical attributes of a person can be represented as in Fig.

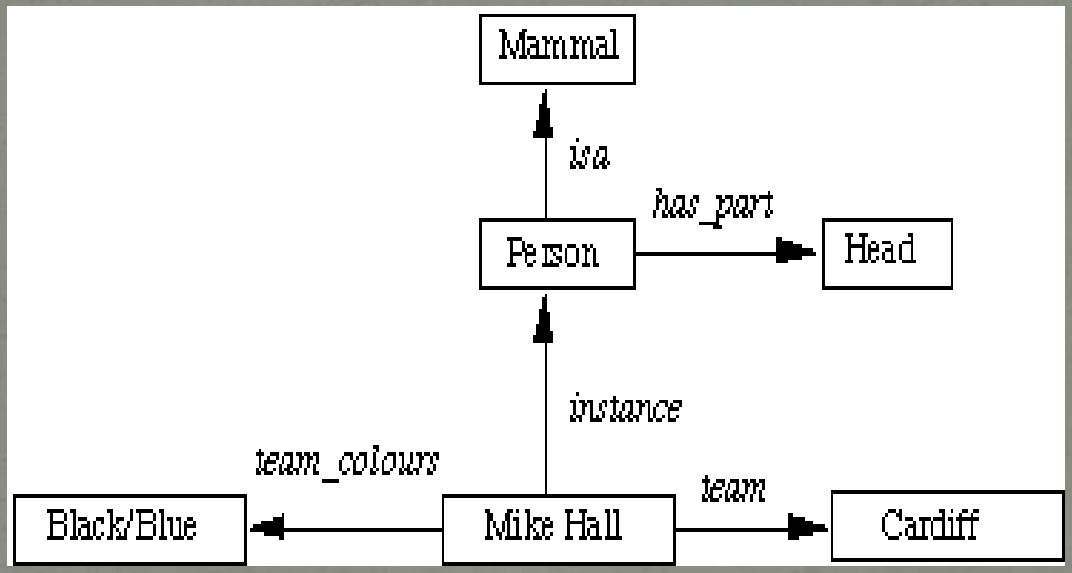


Fig: A Semantic Network

These values can also be represented in logic as: *isa(person, mammal)*, *instance(Mike-Hall, person) team(Mike-Hall, Cardiff)*

4)Frames

A *frame* is a collection of attributes or slots and associated values that describe some real world entity.

Frames can also be regarded as an extension to Semantic nets. As tasks became more complex the representation needs to be more structured. The more structured the system it becomes more beneficial to use frames.

Each frame represents:

- a class (set), or
- an instance (an element of a class).

Frame Knowledge Representation