Knowledge Representation

Knowledge Representation?

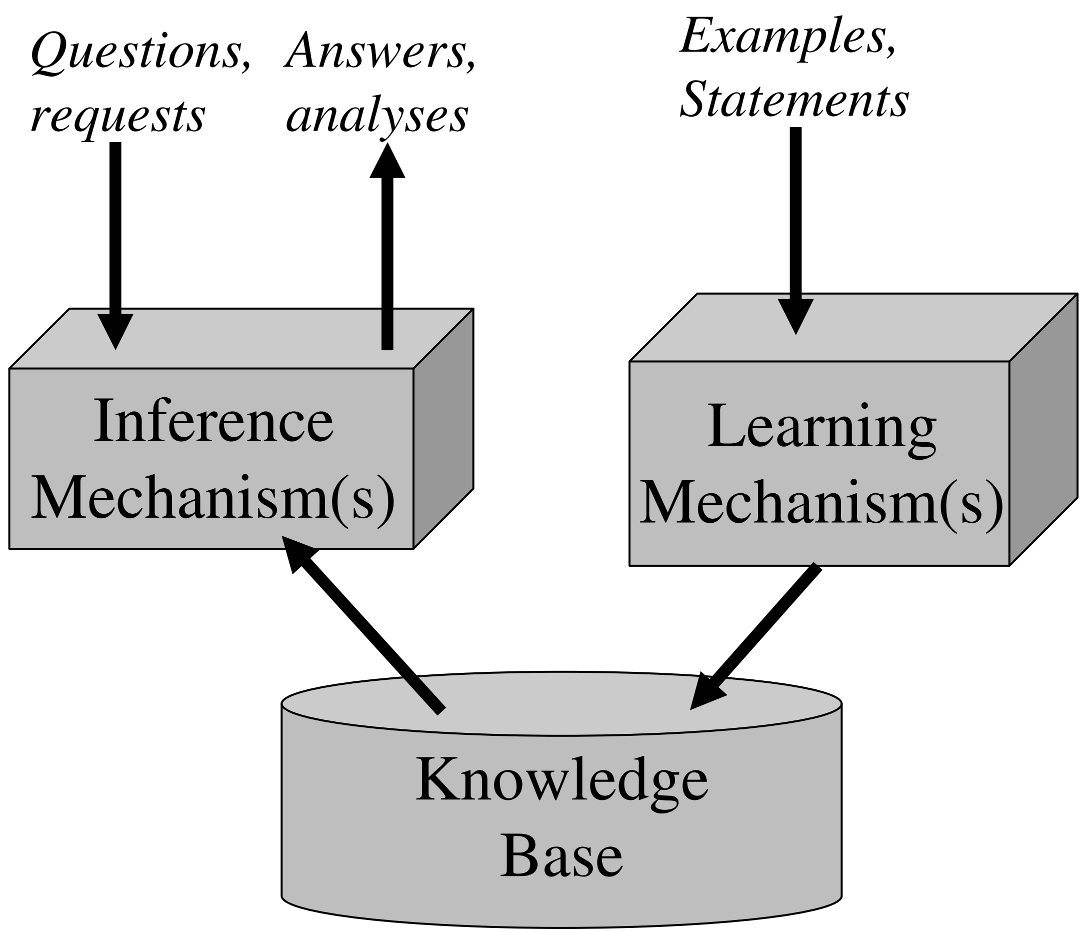
- Ambiguous term
  - “The study of how to put knowledge into a form that a computer can reason with” (Russell and Norvig)
- Originally couple w/ linguistics
- Lead to philosophical analysis of language

Representation Representation Representation

- Think about knowledge, rather than data in AI
- Facts
- Procedures
- Meaning
  - Cannot have intelligence without knowledge
- Always been very important in AI
- Choosing the wrong representation
  - Could lead to a project failing
- Still a lot of work done on representation issues

How knowledge representations are used in cognitive models

- Contents of KB is part of cognitive model
- Some models hypothesize multiple knowledge bases.



Representations for Problem solving techniques

- For certain problem solving techniques
  - The “best” representation has already been worked out
  - Often it is an obvious 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)
- But what if you have a new project?
  - What kind of general representations schemes are there?

Four General Representation Types

- Logical Representations
- Semantic Networks
- Production Rules
- Frames

Logical Representations
What is a Logic?

- Lay down some concrete communication rules
  - In order to give information to agents, and get info
    - Without errors in communication (or at least, fewer)
- Think of a logic as a language
  - Many ways to translate from one language to another
- **Expressiveness**
  - How much of natural language (e.g., English)
    - We are able to translate into the logical language

Syntax and Semantics of Logics

- Syntax
  - How we can construct legal sentences in the logic
  - Which symbols we can use (English: letters, punctuation)
  - How we are allowed to write down those symbols
- Semantics
  - How we interpret (read) sentences in the logic
  - i.e., what the meaning of a sentence is
- Example: “All lecturers are six foot tall”
  - Perfectly valid sentence (syntax)
  - And we can understand the meaning (semantics)
  - This sentence happens to be false (there is a counterexample)

Propositional Logic

- Syntax
  - Propositions such as P meaning “it is wet”
  - Connectives: and, or, not, implies, equivalent

    $\wedge \quad \vee \quad \neg \quad \rightarrow \quad \leftrightarrow$
  - Brackets, T (true) and F (false)
- Semantics
  - How to work out the truth of a sentence
    - Need to know how connectives affect truth
    - E.g., “P and Q” is true if and only if P is true and Q is true
    - “P implies Q” is true if P and Q are true or if P is false
  - Can draw up truth tables to work out the truth of statements

First Order Predicate Logic

- More expressive logic than propositional
- Syntax allows
  - Constants, variables, predicates, functions and quantifiers
- So, we say something is true for all objects (universal)
  - Or something is true for at least one object (existential)
- Semantics
  - Working out the truth of statement
    - This can be done using rules of deduction

Example Sentence

- In English:
  - "Every Monday and Wednesday I go to John's house for dinner"
- In first order predicate logic:

$$\forall X\ ((\text{day\_of\_week}(X, \text{monday}) \vee \text{day\_of\_week}(X, \text{weds})) \longrightarrow (\text{go\_to}(\text{me}, \text{house\_of}(\text{john}) \wedge \text{eat}(\text{me}, \text{dinner}))).$$

- Note the change from "and" to "or"
  - Translating is problematic

Higher Order Predicate Logic

- More expressive than first order predicate logic
- Allows quantification over functions and predicates, as well as objects
- For example
  - We can say that all our polynomials have a zero at 17: $\forall$ f (f(17)=0).
- Working at the meta-level
  - Important to AI, but not often used

Other Logics

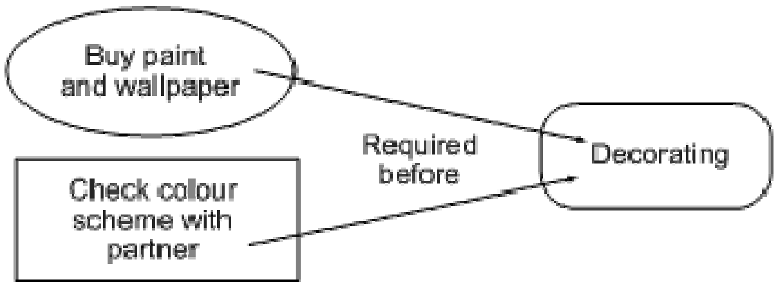
- Fuzzy logic
  - Use probabilities, rather than truth values
- Multi-valued logics
  - Assertions other than true and false allowed
    - E.g., “unknown”
- Modal logics
  - Include beliefs about the world
- Temporal logics
  - Incorporate considerations of time

Why Logic is a Good Representation

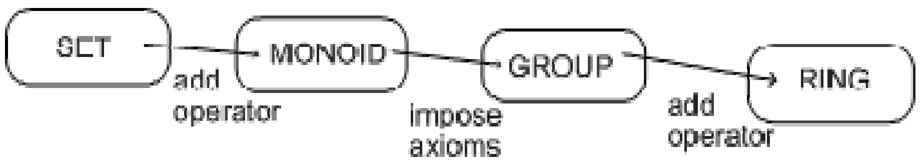
- Some of many reasons are:
  - It's fairly easy to do the translation when possible
  - There are whole tracts of mathematics devoted to it
  - It enables us to do logical reasoning
  - Programming languages have grown out of logics
    - Prolog uses logic programs (a subset of predicate logic)

Semantic Networks

- Logic is not the only fruit
- Humans draw diagrams all the time, e.g.,
  - E.g. causal relationships:

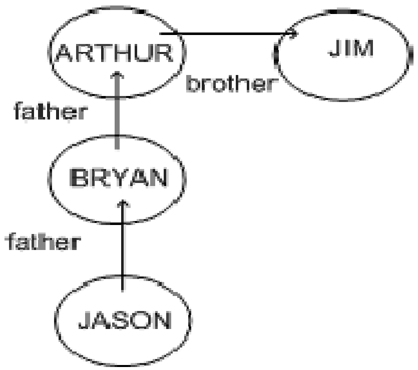


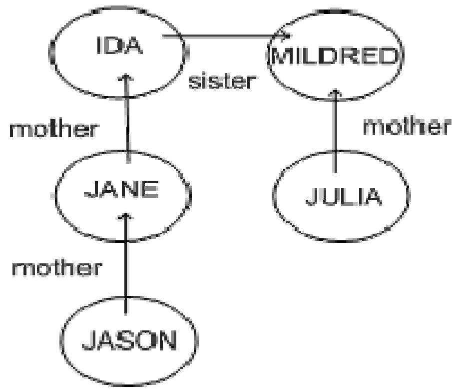
  - And relationships between ideas:



Graphical Representations

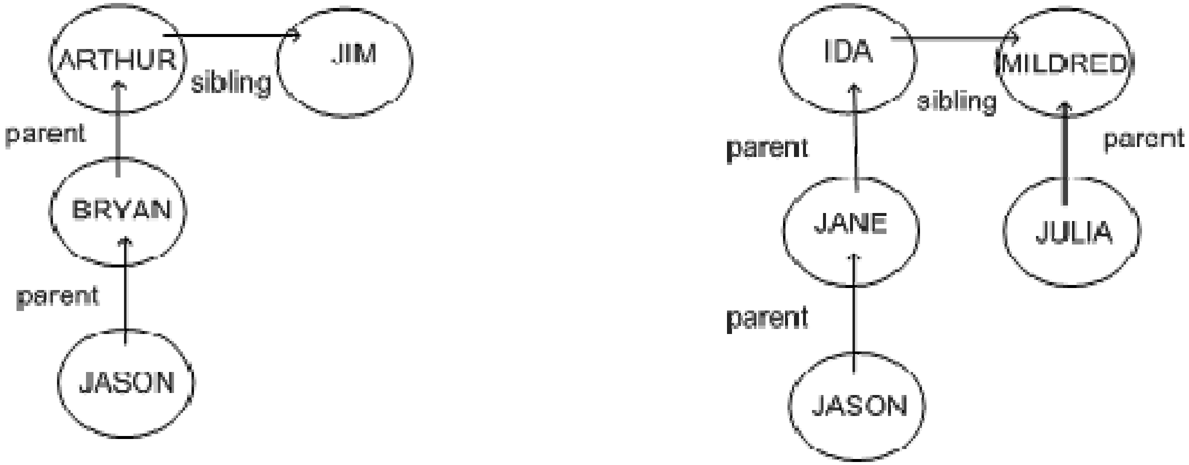
- Graphs are very easy to store inside a computer
- For information to be of any use
  - We must impose a formalism on the graphs





  - Jason is 15, Bryan is 40, Arthur is 70, Jim is 74
  - How old is Julia?

Better Graphical Representation



- Because the formalism is the same
  - We can guess that Julia's age is similar to Bryan's
- Limited the syntax to impose formalism

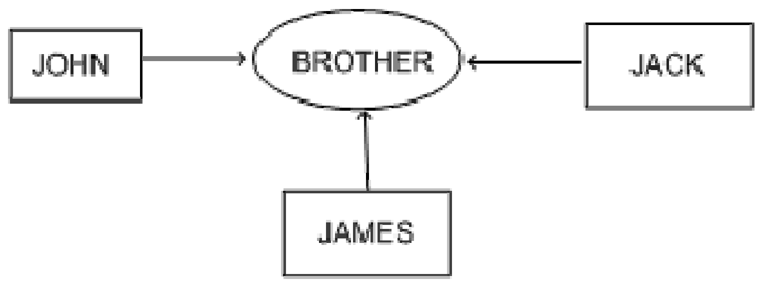
Semantic Network Formalisms

- Used a lot for natural language understanding
  - Represent two sentences by graphs
    - Sentences with same meaning have exactly same graphs
- Conceptual Dependency Theory
  - Roger Schank's brainchild
  - Concepts are nodes, relationships are edges
  - Narrow down labels for edges to a very few possibilities
  - Problem:
    - Not clear whether reduction to graphs can be automated for all sentences in a natural language

Conceptual Graphs

- John Sowa
- Each graph represents a single proposition
- Concept nodes can be:
  - Concrete (visualisable) such as restaurant, my dog spot
  - Abstract (not easily visualisable) such as anger
- Edges do not have labels
  - Instead, we introduce conceptual relation nodes
- Many other considerations in the formalism
  - See Russell and Norvig for details

Example Conceptual Graph



- Advantage:
  - Single relationship between multiple concepts is easily representable

Production Rule Representations

- Consists of <condition,action> pairs
- Agent checks if a condition holds
  - If so, the production rule “fires” and the action is carried out
  - This is a recognize-act cycle
- Given a new situation (state)
  - Multiple production rules will fire at once
  - Call this the **conflict set**
  - Agent must choose from this set
    - Call this **conflict resolution**
- Production system is any agent
  - Which performs using recognize-act cycles

Example Production Rule

102. After creating a new generalization G of Concept C
   - Consider looking for non-examples of G

- This was paraphrased
  - In general, we have to be more concrete
    - About exactly when to fire and what to do

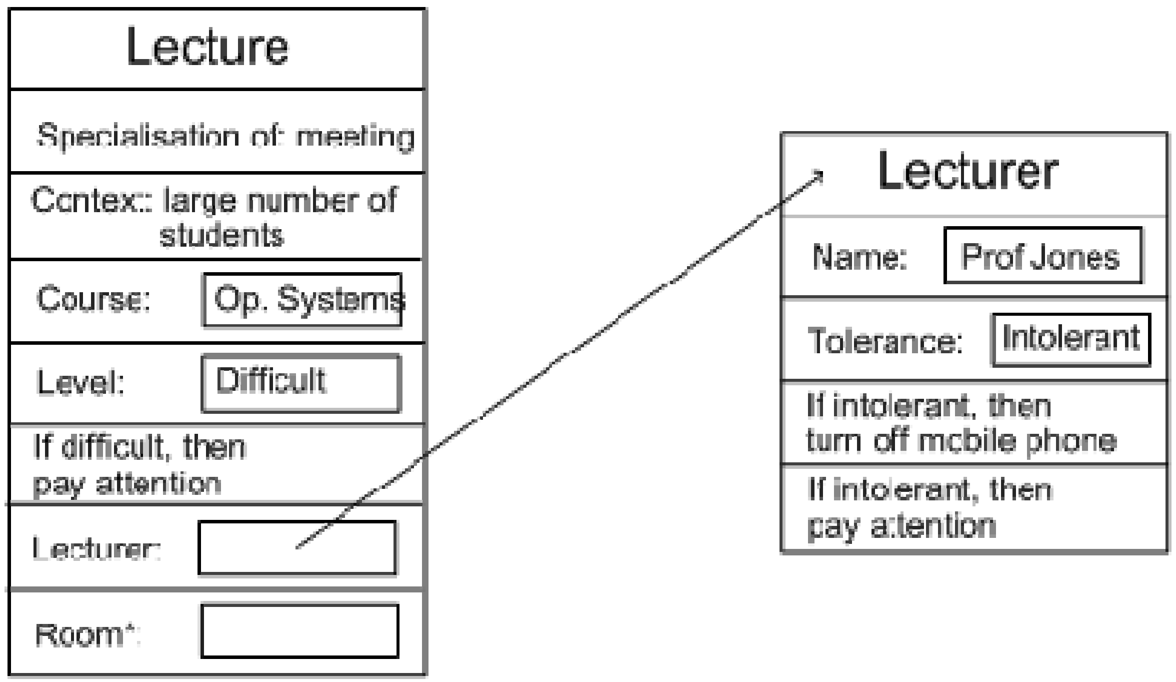
Frame Representations

- Information retrieval when facing a new situation
  - The information is stored in frames with slots
  - Some of the slots trigger actions, causing new situations
- Frames are templates
  - Which are to be filled-in in a situation
  - Filling them in causes an agent to
    - Undertake actions and retrieve other frames
- Frames are extensions of record datatype in databases
  - Also very similar to objects in OOP

Flexibility in Frames

- Slots in a frame can contain
  - Information for choosing a frame in a situation
  - Relationships between this and other frames
  - Procedures to carry out after various slots filled
  - Default information to use where input is missing
  - Blank slots - left blank unless required for a task
  - Other frames, which gives a hierarchy

Example Frame



Comparisons of KR Methods

- Rules
  - Adv.
    - simple syntax, easy to understand, simple interpreter, high modular, flexible
  - Disadv.
    - Hard to follow hierarchies, inefficient for large systems, not all knowledge can be expressed as rules

Comparisons of KR Methods

- Semantic Nets
  - Adv.
    - Easy to follow hierarchy, easy to trace association, flexible
  - Disadv.
    - Meaning attached to nodes might be ambiguous
    - exception handling is difficult
    - difficult to program

Comparisons of KR Methods

- Frames
  - Adv.
    - Expressive power, easy to set up slots for new properties and relations
    - easy to create specialized procedures
    - easy to include default information and detect missing values
  - Disadv.
    - Difficult to program
    - difficult for inference