Knowledge Representation

06 January 2021 10:41

Objectives:

- Defining knowledge and reasoning
- Expert systems
- Knowledge as rules
- Rule based systems
- Forward and backward chaining
- Conflict resolution
- Assumption-based reasoning

Defining Knowledge & Reasoning

What is knowledge?

A *relation defined by the propositional attitude between a knower and a proposition* (a simple declarative sentence)
e.g. John knows that Abraham Lincoln was president, or John hopes to see a shooting star tonight

What matters is whether the proposition is true or false - this is what defines the state of the world according to an agent

Not all knowledge is propositional - e.g. "Jon knows Mary well"

Reasoning

- Explicitly representing all propositions believed to be true is difficult
- Reasoning bridges the gap between what is represented (known true propositions) and what is believed by an agent
- Let KB be a set of propositions believed to be true and alpha be a proposition not in KB
- alpha is said to be logically entailed by KB, KB | = alpha is we believe alpha to be implicitly true given the propositions in the KB
- Knowledge representation language need to have a well-defined notion of entailment
  - What does it mean for a proposition to be true or false?
  - Consider what else we can decide is true or false based on that knowledge

The expressiveness of a representation language has a direct impact on the computational complexity of the reasoning process

Can we compute all the propositions that are entailed by the KB? (logically complete)
Can we guarantee that any proposition believed to be true as a result of reasoning is indeed true? (logically sound)

Knowledge Representation Hypothesis

Any mechanically embodied intelligent process will be comprised of structural ingredients that

- we as external observers naturally take to represent a propositional account of knowledge that the overall process exhibits
- independent of such external semantic attribution, play a formal but causal and essential role in engendering the behaviour that manifests that knowledge

In other words, a process contains a collection of propositions which it believes to be true, and reasons with the propositions during its operation

Propositions
Typically it is more natural to represent knowledge as a logical formulae rather than a table of information
- with formulae, it is easier to check correctness and debug formulae
- can incrementally add to a formulae easily
- can extend with infinitely many variables and domains

For efficient reasoning, we can exploit the Boolean nature of such formulae

A proposition is a statement that is true or false, which naturally can be represented as a logical formulae

Semantics
When creating a KB, we must choose the variables that will be used to build propositions
These should have meaning to the KB designer

We then give the system knowledge about the domain and can then make enquiries
- knowledge will take the form of a clause, h <- b which consists of two parts
- the body, b is a logical formulae that can evaluate to true or false
- The head, h is a variable that can be derived to be true as a result of b being true

Notably, the system doesn't understand the meaning of the symbols

View of Semantics

Users view
- the user must decide the task domain - intended interpretation
- A variable must be associated with each proposition you want to represent
- tell the system clauses that are true in the intended interpretation - this is known as axiomatizing the domain
- if KB |= alpha then alpha must be true in the intended interpretation
- Users interpret the systems responses using the intended interpretation of the symbols

Computers View
- the system does not have access to the intended interpretation
- It is only aware of the KB
- The system can determine if a particular formula is a logical consequence of KB, but does not understand what that formula really means

Summary of Knowledge Representation
- We talked about defining knowledge and reasoning
- We now know what it means for a system to reason about something
- There is a disconnect between the semantic meaning we give to symbols, and how the computer interprets the symbols

Expert Systems

A computer system with a KB that takes some knowledge and reasons about it, and provides advice, responses and courses of action

- Simulates human reasoning

- Performs reasoning over a representation of human knowledge
- Solves problems with heuristic or approximate methods

Who is an Expert?

- Process knowledge focussed on a specific domain
- Capable of solving problems
- Capable of explaining how they solve problems

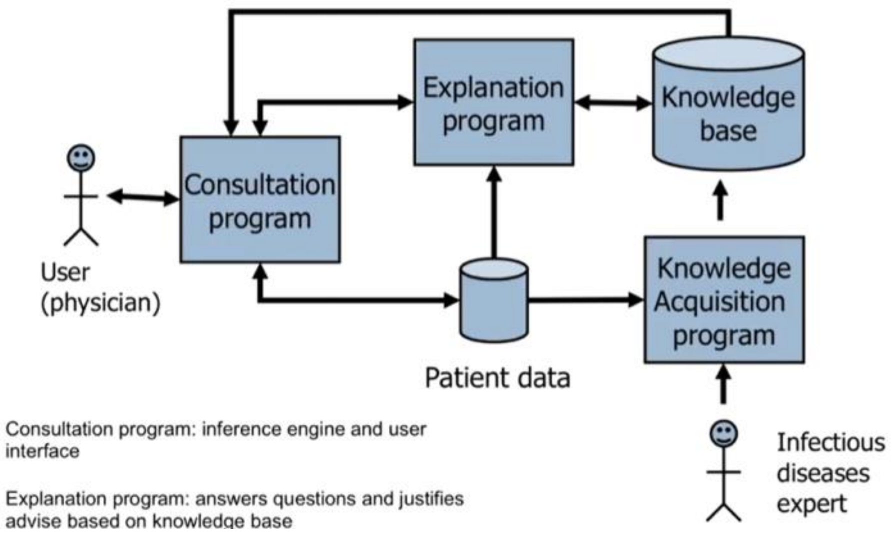
Example - The MYCIN System

Developed by Stanford

Provides advice to a physician on selection of antibiotics for treating infections
- generates hypotheses and weights them based on evidence provided
- Makes therapy recommendations

The Problem

Blood infections therapy
Therapy process - identify organism involved
choose the most appropriate drug or combination of drugs


Explanation program
Knowledge base
Consultation program
User (physician)
Knowledge Acquisition program
Patient data
Infectious diseases expert
Consultation program: inference engine and user interface
Explanation program: answers questions and justifies advise based on knowledge base

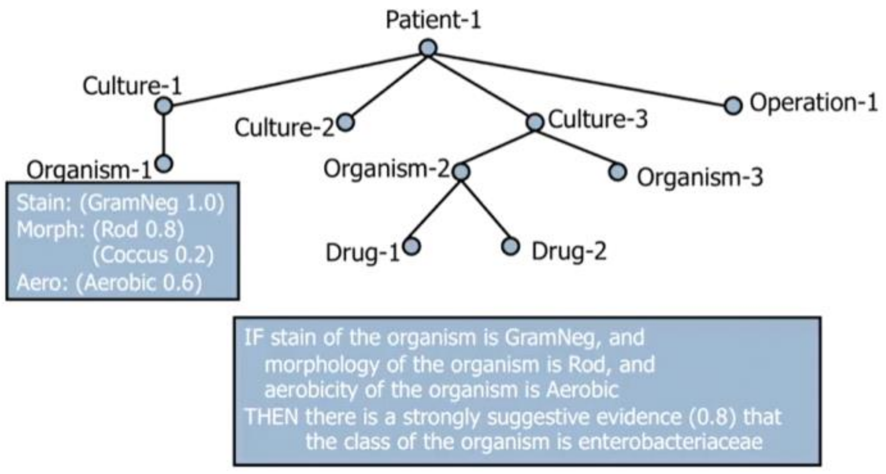
MYCIN Knowledge Representation

- KB contains rules in the form of *if condition and condition 2 ...*
- The rule tally states how certain the conclusion is, given that the conditions are satisfied
- The certainty associated with a conclusion is a function of the combined certainties of the conditions are the rule tally

Also stored in the KB

- Lists such as a list of all known organisms
- Knowledge tables containing clinical data
- A classification of clinical parameters according to the context in which they apply - e.g. are they attributes of patient or organism?

MYCIN Patient Data


Patient-1
Culture-1
Culture-2
Culture-3
Operation-1
Organism-1
Organism-2
Organism-3
Drug-1
Drug-2
Stain: (GramNeg 1.0)
Morph: (Rod 0.8)
(Coccus 0.2)
Aero: (Aerobic 0.6)
IF stain of the organism is GramNeg, and
morphology of the organism is Rod, and
aerobicity of the organism is Aerobic
THEN there is a strongly suggestive evidence (0.8) that
the class of the organism is enterobacteriaceae

MYCIN Control Structure

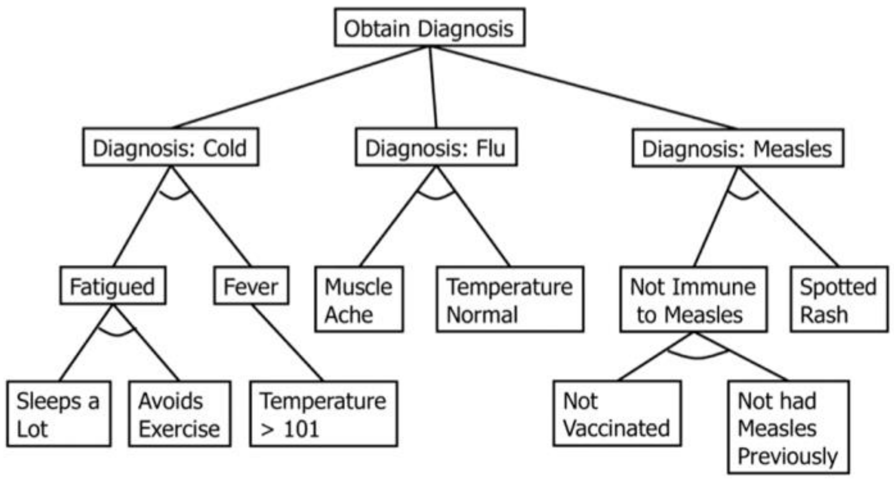
Consultation is a search through a tree of goals

- child nodes in the tree are sub goals that must be achieved to achieve the goal represented by the parent node

MYCIN top level goal specified as the following rule

- If there is an organism which requires therapy, and consideration has been given to any other organism requiring therapy
- THEN compile a list of possible therapies and determine the best one in the list
- The action part of this rule is the root node of the tree of goals

Leaves of the 3 are facts such as lab data - satisfy this goal by asking the user



Rule Based Systems

Rules as Knowledge

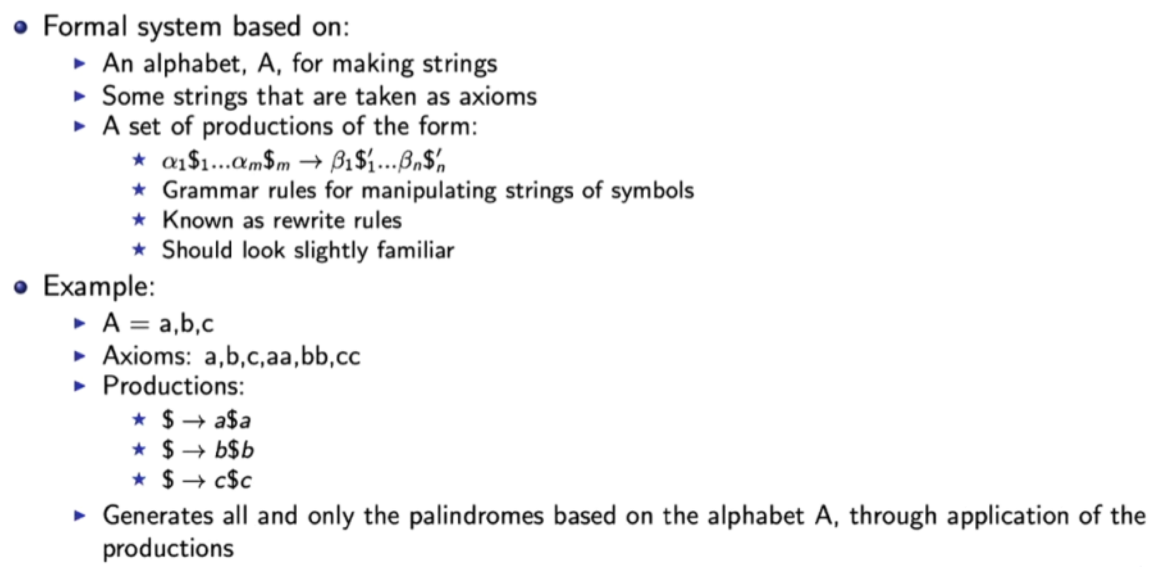
Information can be stored as **propositions**

People tend to associate intelligent behaviour with regularities in behaviour - rational people act consistently

Production rules are a formalism that has been used in automata theory, formal grammars and the design of programming languages

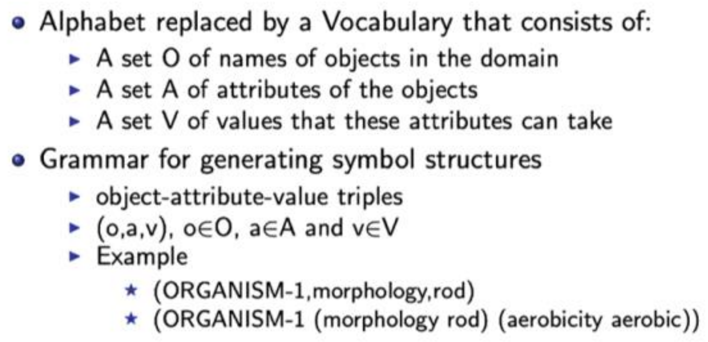
In expert systems, this is referred to as a **condition action rule** or situation action rule

Canonical Systems



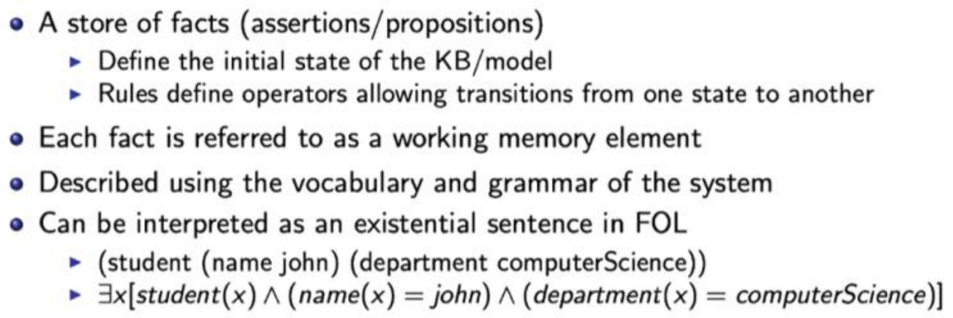
- Formal system based on:
  - An alphabet, A, for making strings
  - Some strings that are taken as axioms
  - A set of productions of the form:
    - $\alpha_1\$_1...\alpha_m\$_m \rightarrow \beta_1\$'_1...\beta_n\$'_n$
    - Grammar rules for manipulating strings of symbols
    - Known as rewrite rules
    - Should look slightly familiar
- Example:
  - A = a,b,c
  - Axioms: a,b,c,aa,bb,cc
  - Productions:
    - $\$ \rightarrow a\$a$
    - $\$ \rightarrow b\$b$
    - $\$ \rightarrow c\$c$
  - Generates all and only the palindromes based on the alphabet A, through application of the productions

Knowledge Representation



- Alphabet replaced by a Vocabulary that consists of:
  - A set O of names of objects in the domain
  - A set A of attributes of the objects
  - A set V of values that these attributes can take
- Grammar for generating symbol structures
  - object-attribute-value triples
  - (o,a,v), o∈O, a∈A and v∈V
  - Example
    - (ORGANISM-1,morphology,rod)
    - (ORGANISM-1 (morphology rod) (aerobicity aerobic))

Working Memory

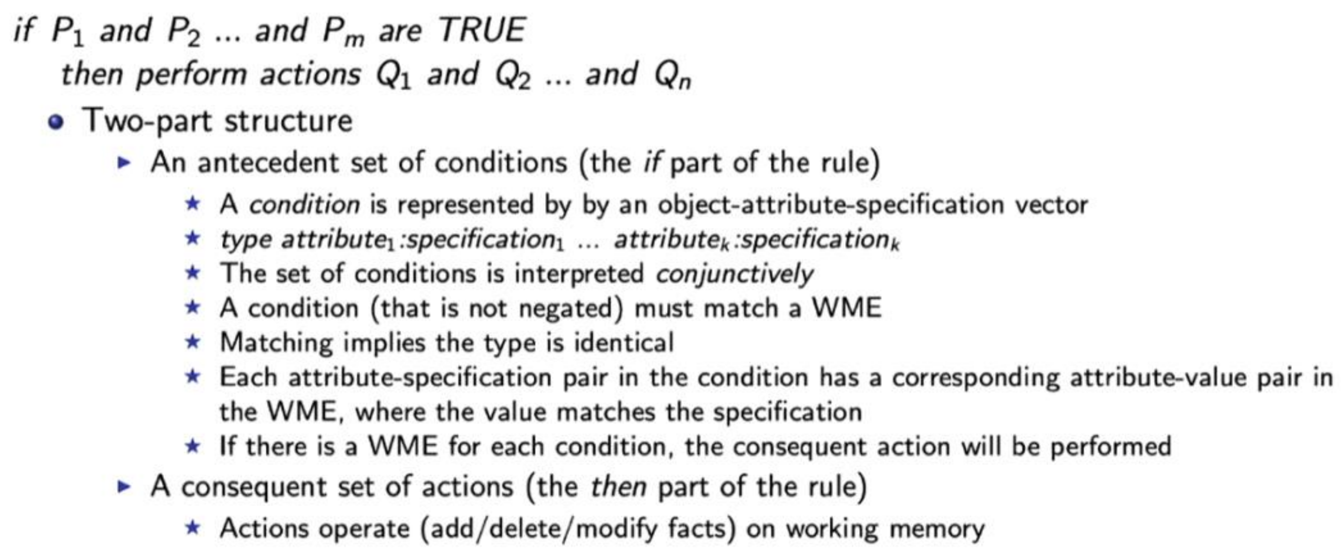


- A store of facts (assertions/propositions)
  - Define the initial state of the KB/model
  - Rules define operators allowing transitions from one state to another
- Each fact is referred to as a working memory element
- Described using the vocabulary and grammar of the system
- Can be interpreted as an existential sentence in FOL
  - (student (name john) (department computerScience))
  - $\exists x[student(x) \wedge (name(x) = john) \wedge (department(x) = computerScience)]$

Key Points

- Facts are working memory elements - we describe them with the grammar of the system
- We can describe the fact with an existential sentence in First Order Logic

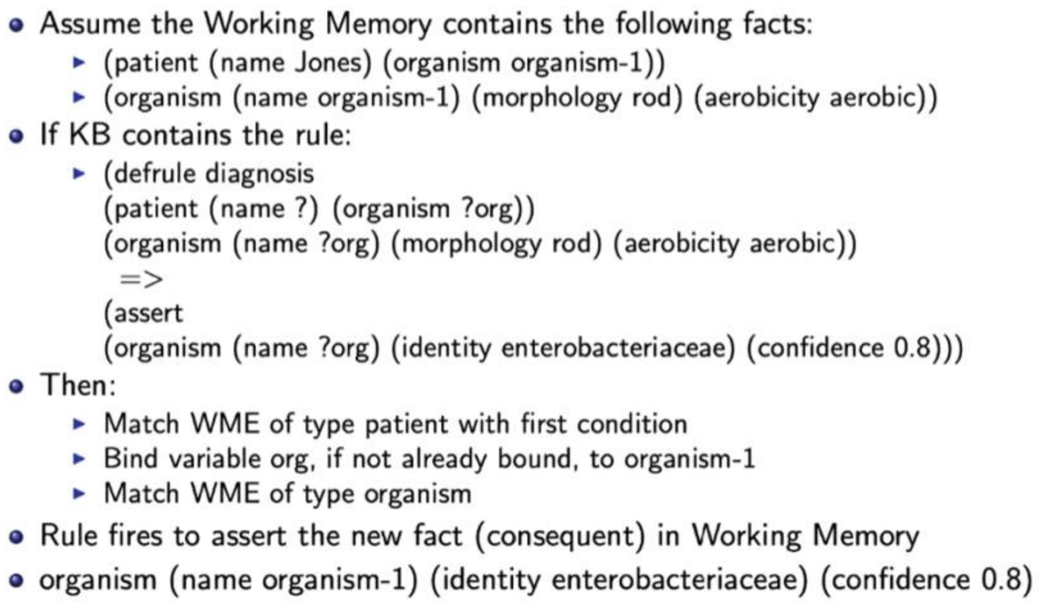
A Production Rule



if P_1 and P_2 ... and P_m are TRUE
then perform actions Q_1 and Q_2 ... and Q_n

- Two-part structure
  - An antecedent set of conditions (the *if* part of the rule)
    - ★ A *condition* is represented by by an object-attribute-specification vector
    - ★ *type attribute$_1$:specification$_1$ ... attribute$_k$:specification$_k$*
    - ★ The set of conditions is interpreted *conjunctively*
    - ★ A condition (that is not negated) must match a WME
    - ★ Matching implies the type is identical
    - ★ Each attribute-specification pair in the condition has a corresponding attribute-value pair in the WME, where the value matches the specification
    - ★ If there is a WME for each condition, the consequent action will be performed
  - A consequent set of actions (the *then* part of the rule)
    - ★ Actions operate (add/delete/modify facts) on working memory

CLIPS Syntax



- Assume the Working Memory contains the following facts:
  - (patient (name Jones) (organism organism-1))
  - (organism (name organism-1) (morphology rod) (aerobicity aerobic))
- If KB contains the rule:
  - (defrule diagnosis
    (patient (name ?) (organism ?org))
    (organism (name ?org) (morphology rod) (aerobicity aerobic))
    =>
    (assert
    (organism (name ?org) (identity enterobacteriaceae) (confidence 0.8)))
- Then:
  - Match WME of type patient with first condition
  - Bind variable org, if not already bound, to organism-1
  - Match WME of type organism
- Rule fires to assert the new fact (consequent) in Working Memory
- organism (name organism-1) (identity enterobacteriaceae) (confidence 0.8)

The Rule Interpreter

Recognise-act cycle:

- Match the antecedent condition of rules against elements in the working memory
- if more than one rule antecedent matches ("can fire") choose one of the rules based on some conflict resolution strategy
- Apply the rule (add and delete elements of working memory)
- Back to Match

Cycle halts when no rules become active or if action of rule fired is to halt

Controlling Inference Behaviour

- Global control - domain independent, hard coded within the inference engine
- Local Control - domain dependent, coded in the form of meta rules - reason about which rule to fire rather than about objects in the domain

Summary of Rule-Based Systems

We have set up our rule based systems with a working memory
we can build rules that allow us to reason
we know the process that we use to work out what conclusions can be drawn from KB (antecedence)

Forward & Backward Chaining

- When facts match a rules condition, the rule fires
- Producing a solution from a KB is akin to a logical proof
- We are demonstrating that something logically follows from the initial state of the system
- There is no way for information to get added that isn't a consequence of the existing data
- We call the series of rules that we first the inference chain
- There are two main ways to build a chain - forward chaining and backward chaining

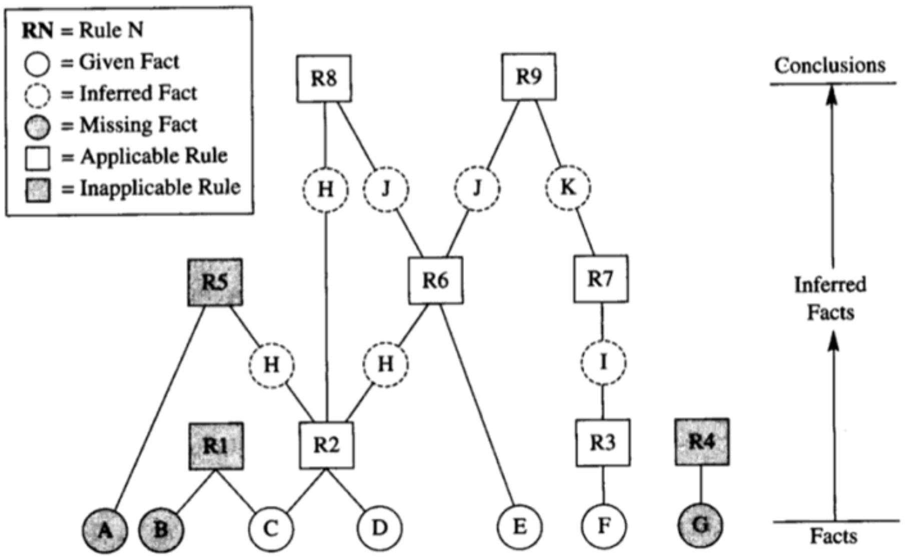
Forward Chaining

- Data driven - start with known data (facts)
- facts that can be inferred will be inferred even if they are not related to the goal

bottom-up ground proof procedure

Is a simple procedure of matching production rules that can be fired

- Select rules that produce new WME for the KB
- This is repeated until no rules can fire, then we can check to see if the desired solution is now in the KB
- *Is both logically sound and complete*
- Can spend a lot of time processing rules that do not contribute to the goal



Backward Chaining

Goal driven - system has a goal and the inference engine attempts to find the evidence to prove it
Only use data which is needed

top-down definite clause proof procedure

- Start from the query and work backwards to determine if it is a logical consequence of the KB
- Our query will be a clause made up of a number of WME, that we want the KB to contain

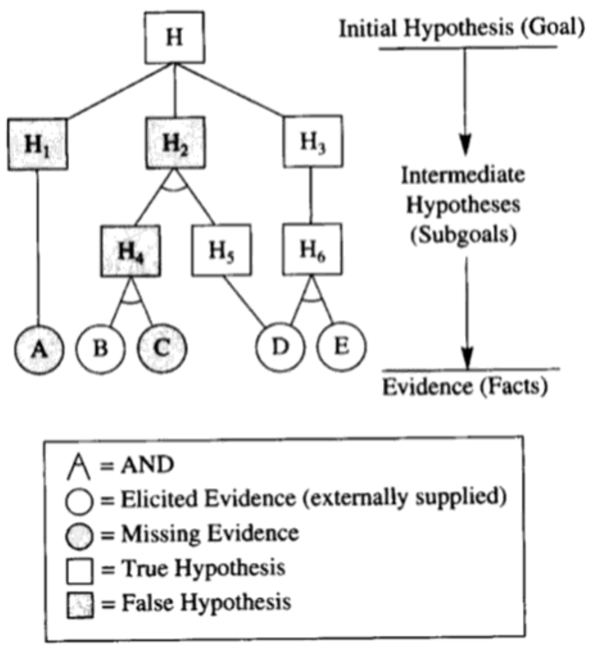
- We select an element of the clause and find a production rule that results in that element being added to the KB
- We replace the element with the condition of the production rule
- We repeat this until the query clause is entirely made up of elements that already exist in the KB

Backwards chaining is non-deterministic, based on the choice of production rules

Can stop early if one of the elements of the query cannot be derived

Because our queries are made up of conjunctives, if one element cannot be derived from the KB, *then the whole query cannot be derived*

However, when choosing the production rule that derives the element, we may also hit a dead end
- this does not mean we have to stop, merely we must select a new production rule
- we can only definitively say an element cannot be derived if we have explored all the clauses



Askable-clauses
- The only way to receive new information is from the user
- It can be tedious to have the user input all the information they know
- Instead ,we can define some clauses as askable, meaning it is information that KB can ask the user about

Conflict Resolution
non-syntactic errors in rule based systems:
- an incorrect answer is produce - a clause that should be false has been interpreted to be true
- an answer was not produced
- a program gets stuck in an infinite loop
- the system asks irrelevant questions - this requires investigation and assessment of the KB

Debugging Incorrect Answers
- suppose some clause / variable was proved false in the intended interpretation
- there must be some rule in the KB that was used to prove that clause

- either one of the variables in the rule is false in the intended interpretation - we can debug this by asking the user
- all of the variables are true in the intended interpretation - the rule itself is wrong and should be reassessed

Missing Answers

If a variable is true in the intended interpretation but could not be proved, then either
- there is no appropriate rule for the variable
- there is a rule and it did not fire when it should have
- this means one of the variables of the condition should be true but isn't
- this means we can solve this recursively, finding all the variables that should be true but do not have a rule

Infinite Loops

- A KB can get stuck in an infinite loop if there are rules that are cyclical - if you converted the KB into a directed graph, we will see a cycle
- A cyclical nature is symptomatic of another bug
- Rules should be reassessed to help ensure an acyclic KB
- Forward chaining cannot get stuck in infinite loops it only fires if it will add new elements to the working energy

Conflict Resolution

The firing of a rule may affect the activation of other ules as it affects the facts stored in the database
The method for choosing which rule to fire when more than one rule can be fired in a given inference cycle is called conflict resolution

Basic Conflict Resolution

- Fire rules in order of appearance in the KB - rules order will have a strong impact on the outcome - implicit knowledge
- Rule priority - make explicit the order in which the rules may fire

Specificity

- Fire the most specific rule
- More conditions means that it's more relevant to the current situation

Recency

- Fire the rule that use the data most recently entered into the working memory

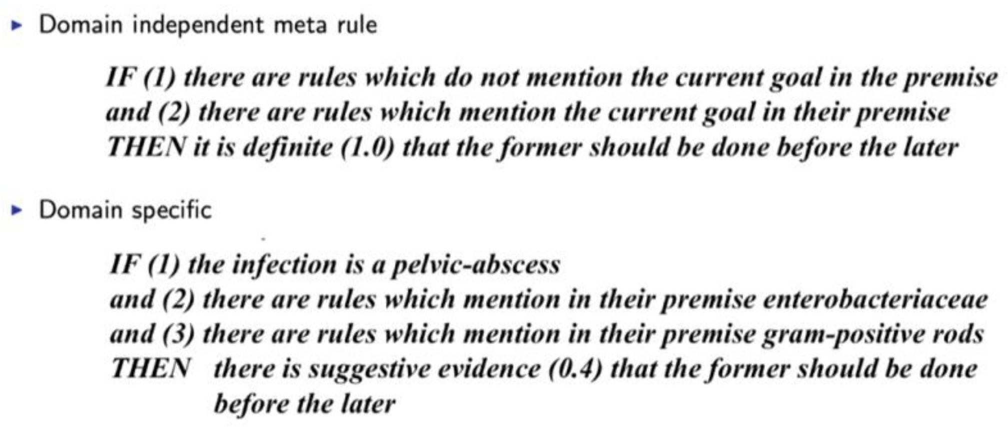
Refractoriness

- A rule is only allowed to fire once on the same data
- Prevents loops in inference

Meta Knowledge

- Knowledge about knowledge
- Concerns the use and control of domain knowledge in an expert system
- Represented in the form of metarules
- Determine the strategy for the use of task specific rules in the expert system
- May be domain independent but more likely to be domain dependent

Examples



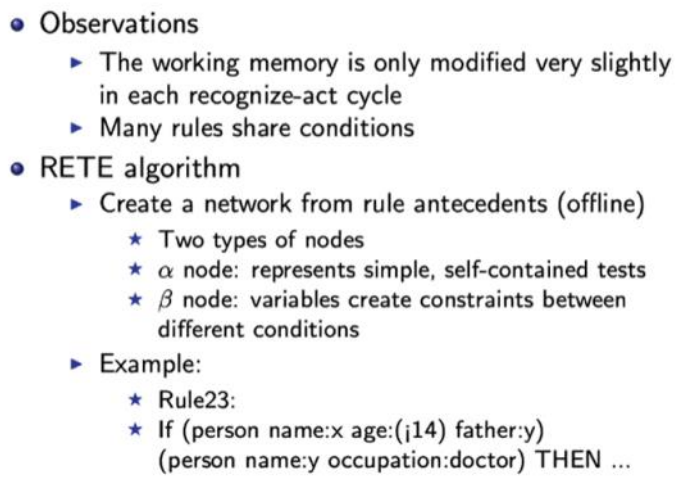
- Domain independent meta rule

  IF (1) there are rules which do not mention the current goal in the premise
  and (2) there are rules which mention the current goal in their premise
  THEN it is definite (1.0) that the former should be done before the later

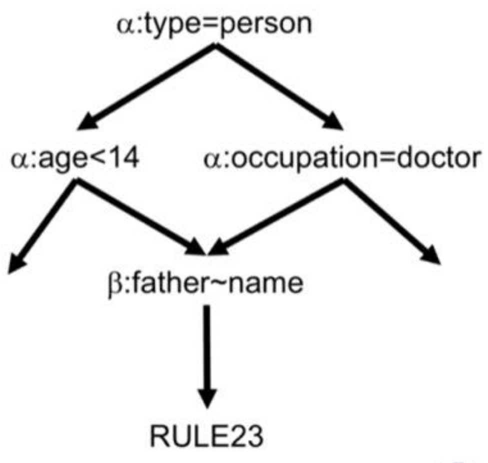
- Domain specific

  IF (1) the infection is a pelvic-abscess
  and (2) there are rules which mention in their premise enterobacteriaceae
  and (3) there are rules which mention in their premise gram-positive rods
  THEN there is suggestive evidence (0.4) that the former should be done before the later

Efficient Rule Matching



- Observations
  - The working memory is only modified very slightly in each recognize-act cycle
  - Many rules share conditions
- RETE algorithm
  - Create a network from rule antecedents (offline)
    - ⋆ Two types of nodes
    - ⋆ α node: represents simple, self-contained tests
    - ⋆ β node: variables create constraints between different conditions
  - Example:
    - ⋆ Rule23:
    - ⋆ If (person name:x age:(¡14) father:y) (person name:y occupation:doctor) THEN ...



RETE

During operation:

- Tokens representing new or changed WME's are passed through the network
- Tokens that make it through the network satisfy the rule
- If a token cannot move through the network it is reassessed when the corresponding WME is modified

Rule-Based Systems: Pros and Cons

- Easy mapping between expert humans and Boolean rules
- Each rule represents an independent piece of knowledge
- Separation of knowledge from the processing/control structure
- Ability to represent and reason with uncertain knowledge
- Exhaustive search through all rules during each inference cycle
- The Knowledge Acquisition Bottleneck
  - No independent learning
- Brittle

Assumption-Based Reasoning

Often we want agents to make assumptions rather than doing deduction from their knowledge
Abduction - an agent makes assumptions to explain observations

- for examples, it hypothesizes what could go wrong with a system to produce the observed symptoms
- Default reasoning - an agent makes assumptions of normality to make predictions

- for example, a delivery robot may want to assume Mary is in her office even if it is not true

Design & Recognition

Two different tasks use assumption-based reasoning

- **Design** - aim to design an artefact or plan
- **Recognition** - aim to find out what is true

Assumption-Based Framework

- A set of closed formula, F, called the facts
- A set H, called the possible hypothesis or assumable