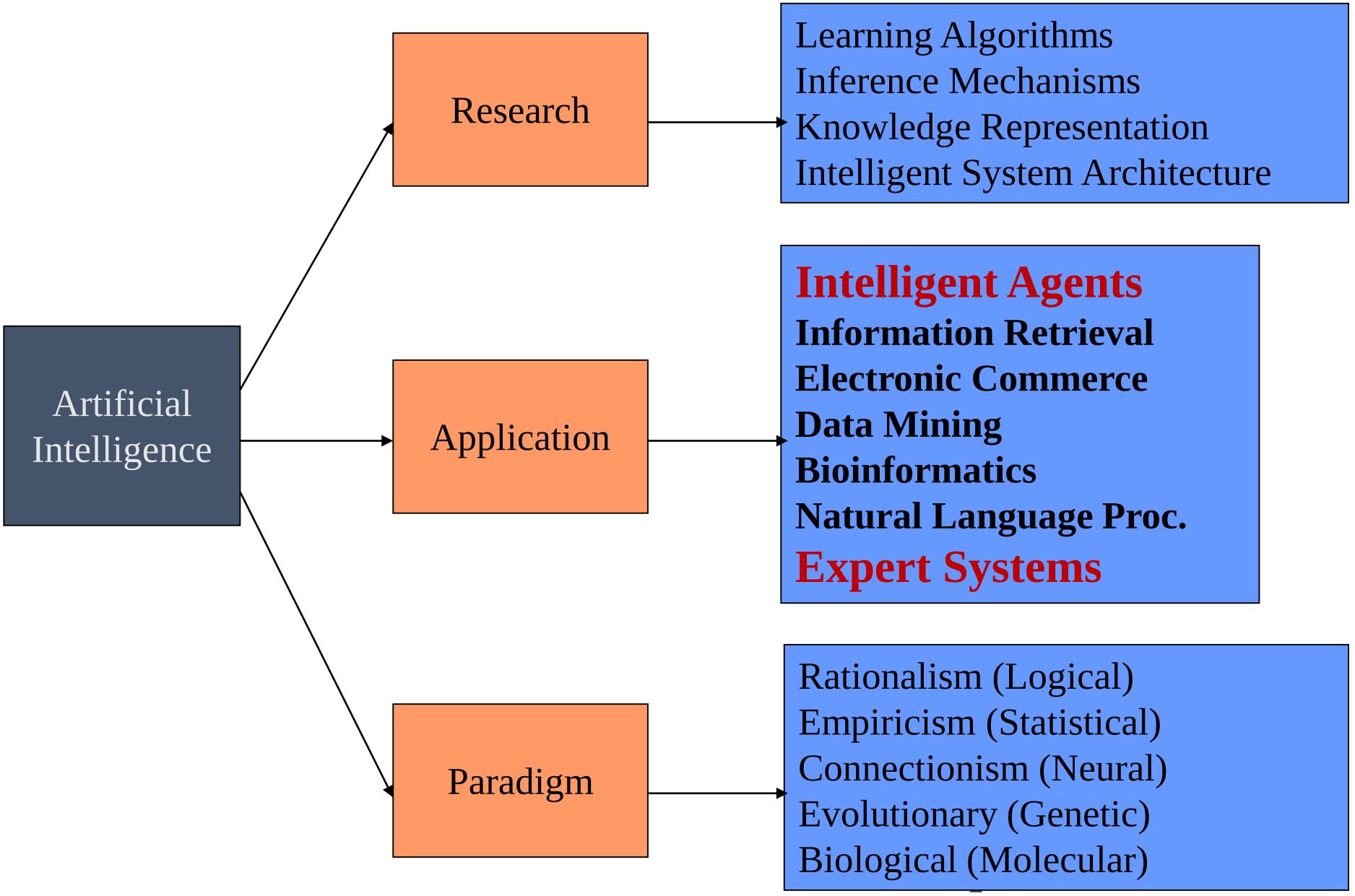
Expert System

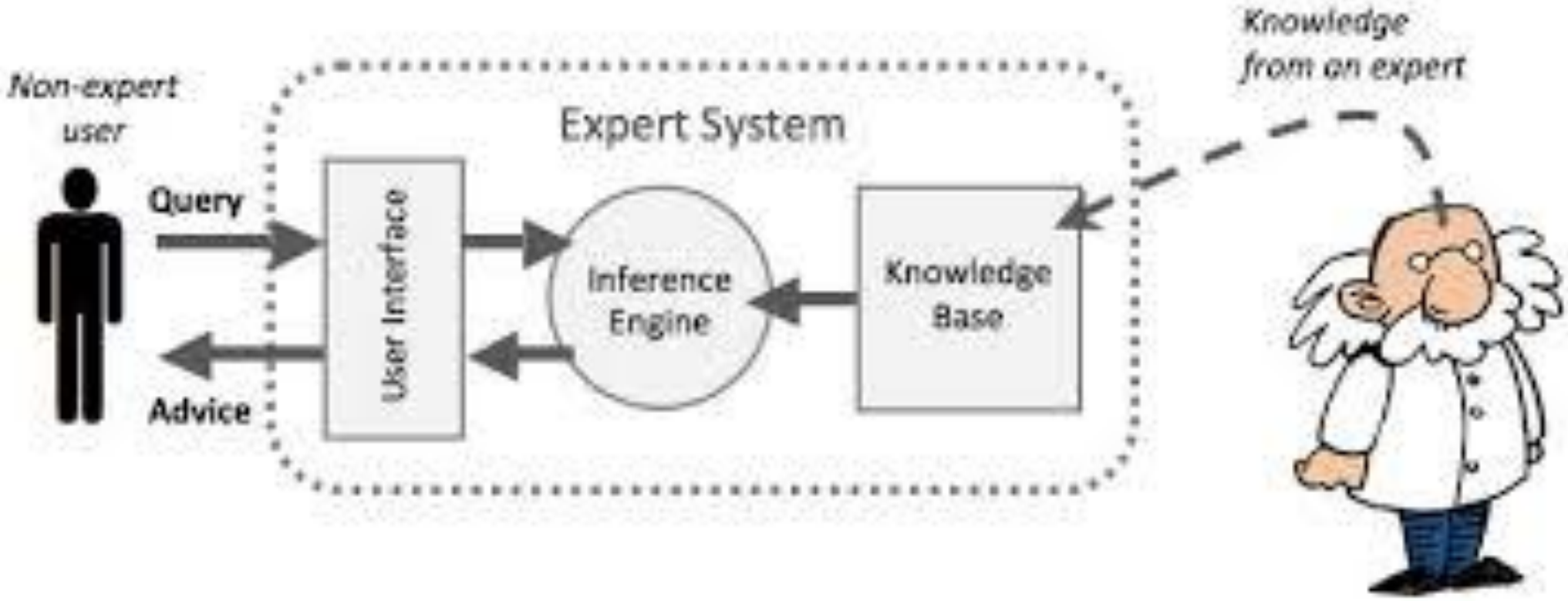
1. Expert system

   Definition, structure, first type of ES: rule-based ES
2. Rule-bases expert system

Research Areas and Approaches



Expert System



Basic concepts

- ES is a CBIS that uses **expert knowledge** to attain high-level decision performance in **narrowly defined problem domain** (Turban, 2011).
  - knowledge is extracted from human experts
  - knowledge must be encoded into a formal language
  - attempts to emulate expert's methodology and performance
  - both theoretical and practical (heuristic) knowledge

Basic concepts

- The features of ES:
  - Expertise – task-specific **knowledge** that experts possess.
    - Deep knowledge – knowledge base must contain complex knowledge, not easily found among non-experts.
    - Self-knowledge – capabilities of update their knowledge, learn from successes and failures.
  - **Reasoning** – reasoning mechanism.
  - **Explanation** – reasoning transparency

Applications of ES

- Customer support at Logitech
  - Interactive knowledge portal to provide Web-based self-help customer support. It emulates the way a human would interact with a customer,
    - allows customer to ask/describe problems in natural language
    - Carries on an intelligent conversation with the user.
- China's freight train system
  - Allocation of freight vehicles, determine what and how much to load on each vehicle.
- MYCIN (a classic application of ES) – 1970s
  - Diagnose bacterial infection of the blood.

Structure of ES

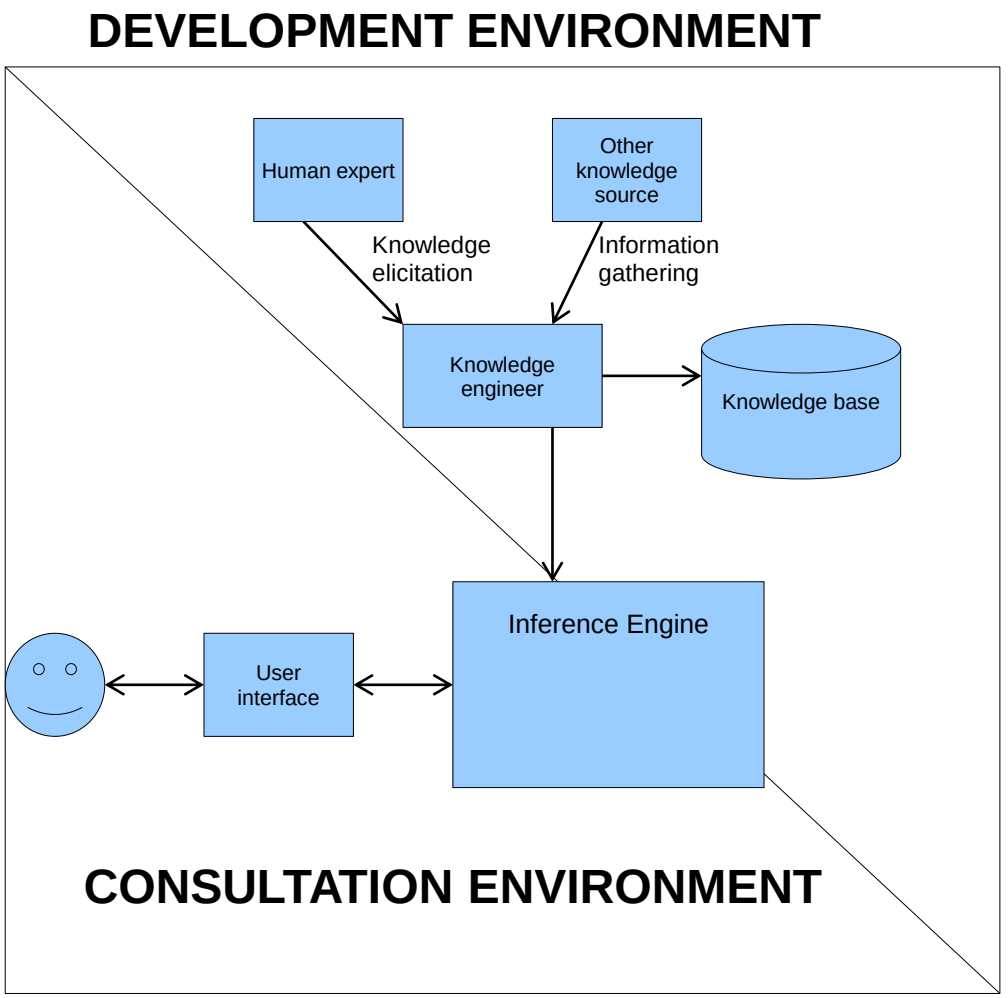
- Three major components of ES:
  - **Knowledge-base**
  - **Inference engine**
  - **User interface**

Inference Engine

- Essentially a computer program that provide a methodology for reasoning to give appropriate conclusion

User interface

- Tool for communication of user-expert → ideally in natural language, or graphical or textual Q&A approach.



Knowledge base

- The process of developing the knowledge base is called **knowledge engineering**.
- The **tasks** in knowledge engineering:
  - Knowledge **acquisition**
  - Knowledge representation
    - Production rules
    - Semantic network, frame-based
    - Decision tables
    - Decision tree
    - Predicate logic
  - Knowledge validation

Knowledge Acquisition: Non-automatic methods

- Interviews with domain experts
  - (Extracting knowledge from a human is often called knowledge elicitation)
- Iterative process, hard to get right first time.
- Human experts usually find it very difficult to state all the data relevant for a given problem.

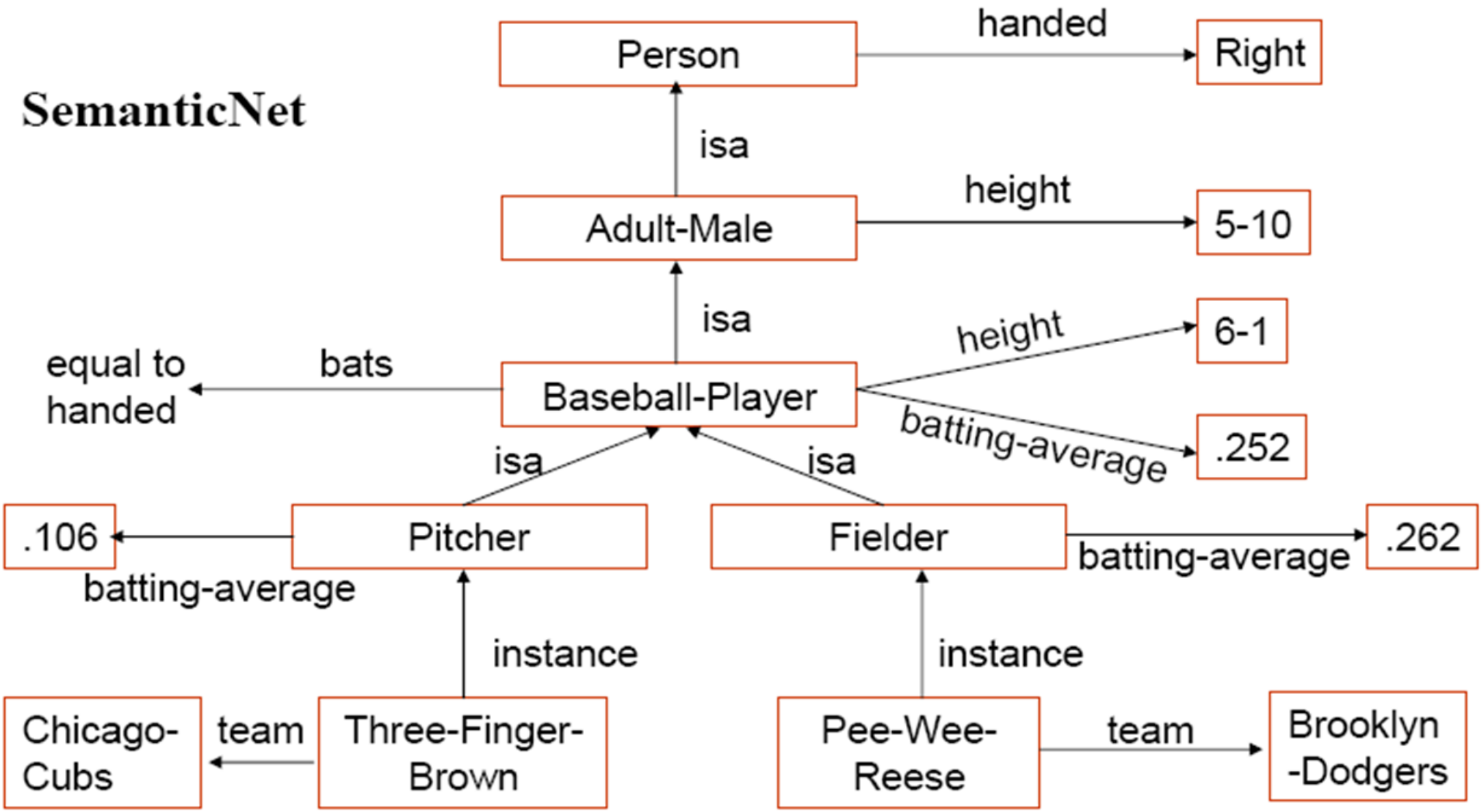
Knowledge Acquisition: automatic and semi-automatic methods (for expert systems)

- Programs which compile **dependency networks** during interviews with experts:
  - MOLE (Elsheman, 1988) works for systems which classify cases as instances of fixed categories. It builds an inference network similar to belief networks we will see later in the module.
  - SALT (Marcus and McDermott, 1989) works for open-ended sets of solutions, such as design problems; builds a dependency network and compiles into a set of production rules.
- Programs using learning:
  - Learning decision diagrams from a set of positive and negative instances of a concept (e.g. when to approve a loan application)
  - Learning rules from a set of positive and negative instances.

Knowledge base

- The process of developing the knowledge base is called **knowledge engineering**.
- The **tasks** in knowledge engineering:
  - Knowledge acquisition
  - Knowledge **representation**
    - Production rules
    - Semantic network, frame-based
    - Decision tables
    - Decision tree
    - Predicate logic
  - Knowledge **validation**

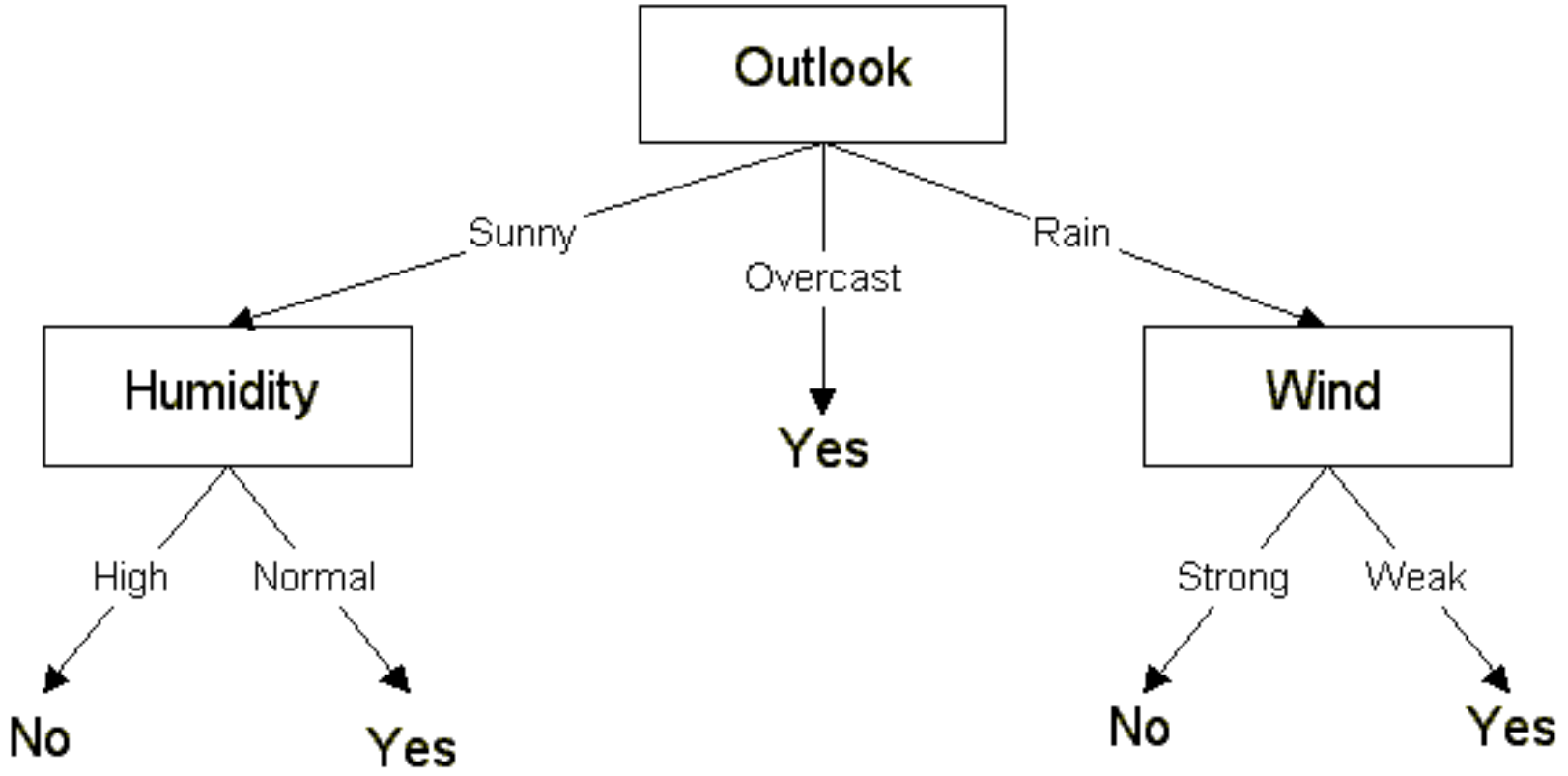
KR tech.: Semantic network



KR tech.: Decision Table

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Else
cash	Y	Y	Y	N	N	
order $>$ 100	Y	N	N	-	-	
order $\geq$ 50	Y	Y	N	-	-	
order $<$ 50	N	N	Y	-	-	
credit record good	-	-	-	Y	N	
give 20% discount	X					
give 10 % discount		X				
accept order			X	X		
reject order					X	
exception report						X

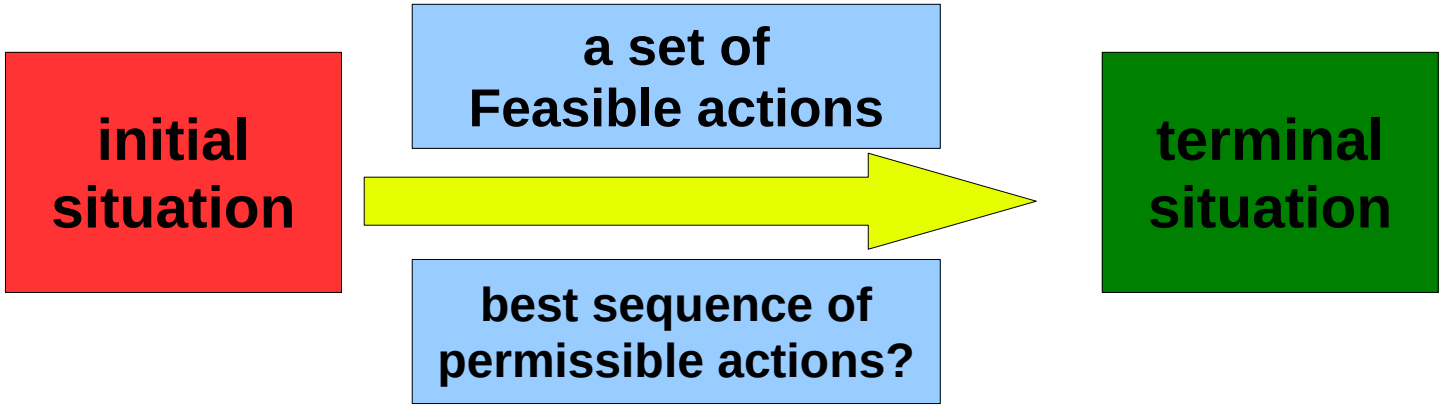
KR tech.: Decision tree



KR tech.: Predicate Logic

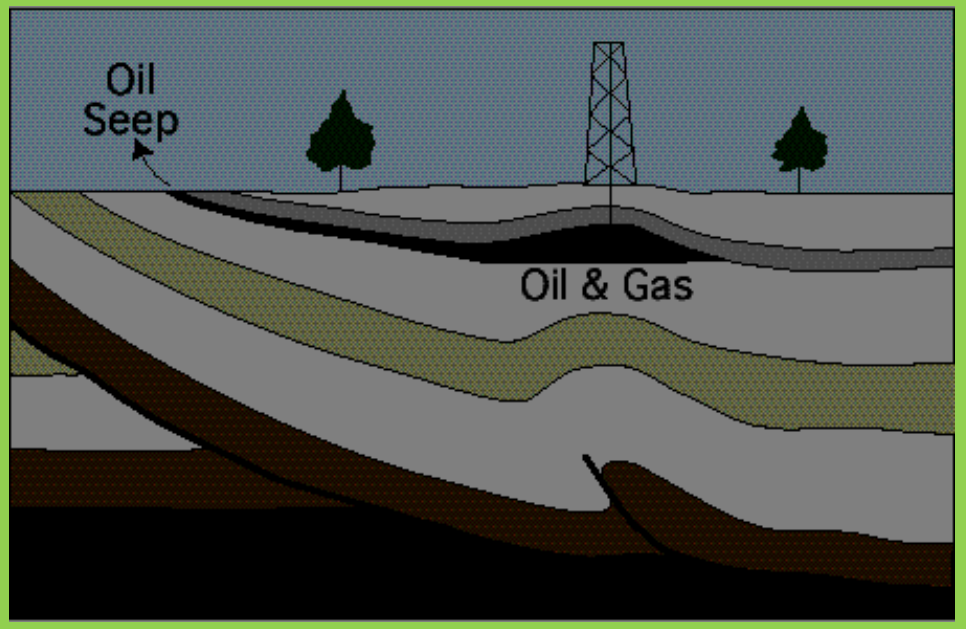
- More expressive logic than propositional
  - **Constants** are objects: john, apples
  - **Predicates** are properties and relations:
    - likes(john, apples)
  - **Functions** transform objects:
    - likes(john, fruit_of(apple_tree))
  - **Variables** represent any object: likes(X, apples)
  - **Quantifiers** qualify values of variables
    - True for all objects (Universal): $\forall$X. likes(X, apples)
    - Exists at least one object (Existential): $\exists$X. likes(X, apples)

Inference (reasoning)



- How a human expert solve problem?
  - Heuristic reasoning, or rules of thumb.
  - Only a relatively few classes of problems where experts use computationally feasible, mathematically precise methods.
  - A large part of what an expert system needs to know is the body of heuristics that specialists use in solving hard problems.

Example:



PROSPECTOR:
Used by geologists to identify sites for drilling or mining

How a geologist identify a promising site?

PUFF:
Medical system for diagnosis of respiratory conditions

How a doctor diagnose respiratory conditions?

Inference (reasoning)

- Some inference mechanisms:
  - Most expert systems are **rule-based**.
    - Possible inference mechanisms in rule-based expert systems are:
      - Backward chaining, goal-driven approach
      - Forward chaining, data-driven approach
  - **Case-Based** Reasoning
    - previous examples (cases) of the task and its solution are stored.
    - To solve a new problem the closest matching case is retrieved, and its solution or an adaptation of it is proposed as the solution to the new problem.
  - Etc.

TABLE 10.5 Generic Categories of Expert Systems

Category	*Problem Addressed*
Interpretation	Inferring situation descriptions from observations
Prediction	Inferring likely consequences of given situations
Diagnosis	Inferring system malfunctions from observations
Design	Configuring objects under constraints
Planning	Developing plans to achieve goals
Monitoring	Comparing observations to plans, flagging exceptions
Debugging	Prescribing remedies for malfunctions
Repair	Executing a plan to administer a prescribed remedy
Instruction	Diagnosing, debugging, and correcting student performance
Control	Interpreting, predicting, repairing, and monitoring system behaviors

General Problems Suitable for Expert Systems

- Interpretation systems
  - Surveillance, image analysis, signal interpretation
- Prediction systems
  - Weather forecasting, traffic predictions, demographics
- Diagnostic systems
  - Medical, mechanical, electronic, software diagnosis
- Design systems
  - Circuit layouts, building design, plant layout
- Planning systems
  - Project management, routing, communications, financial plans

General Problems Suitable for Expert Systems

- Monitoring systems
  - Air traffic control, fiscal management tasks
- Debugging systems
  - Mechanical and software
- Repair systems
  - Incorporate debugging, planning, and execution capabilities
- Instruction systems
  - Identify weaknesses in knowledge and appropriate remedies
- Control systems
  - Life support, artificial environment

Rule-Based Expert System

- A rule-based expert system is an expert system which works as a production system in which rules encode expert knowledge.
  - Rule-based expert systems, use human expert knowledge to solve real-world problems that normally would require human intelligence.
  - Expert knowledge is often represented in the form of rules or as data within the computer.

Rule-based ES: some applications

- Rule-based expert systems have played an important role in modern intelligent systems and their applications in:
  - strategic goal setting,
  - planning,
  - design,
  - scheduling,
  - fault monitoring,
  - diagnosis and so on.

Rule-based ES:

- A rule-based system consists of
  - if-then rules,
  - a bunch of facts, and
  - an interpreter controlling the application of the rules, given the facts.
- These *if-then rule statements* are used to formulate the conditional statements that comprise the complete knowledge base.

- A single if-then rule assumes the form ‘if x is A then y is B’
  - if-part of the rule ‘x is A’ → antecedent or premise,
  - then-part of the rule ‘y is B’ → consequent or conclusion.
- There are two broad kinds of inference engines used in rule-based systems:
  - forward chaining
  - backward chaining systems.

Data-driven Rule-based Expert Systems

Use Forward Chaining:

- Given a certain set of facts, use the rules to generate new facts until the desired goal is reached.
- To forward chain the inference engine must:
  1. Match the condition patterns of rules against facts in working memory.
  2. If there is more than one rule that could be used (that could "fire"), select which one to apply (this is called conflict resolution).
  3. Apply the rule, maybe causing new facts to be added to working memory.
  4. Halt when some useful (or goal) conclusion is added (or until all possible conclusions have been drawn.)

Goal-driven Rule-based Expert Systems

Use Backward Chaining:

- Work backwards from a hypothesized goal, attempting to prove it by linking the goal to the initial facts.
- To backward chain from a goal the inference engine must:
    1. Select rules with conclusions matching the goal.
    2. Replace the goal by the rule's premises. These become sub-goals.
    3. Work backwards till all sub-goals are known to be true - either they are facts or the user provides the information.

Example

A production system IDENTIFIER, which identifies animals.

R1	IF	the animal has hair
	THEN	it is a mammal
R2	IF	the animal gives milk
	THEN	it is a mammal
R3	IF	the animal has feathers
	THEN	it is a bird
R4	IF	the animal flies
		the animal lays eggs
	THEN	it is a bird
R5	IF	the animal is a mammal
		the animal eats meat
	THEN	it is a carnivore

R6	IF	the animal is a mammal
		the animal has pointed teeth
		the animal has claws
		the animal's eyes point forward
	THEN	it is a carnivore
R7	IF	the animal is a mammal
		the animal has hooves
	THEN	it is an ungulate
R8	IF	the animal is a mammal
		the animal chews cud
	THEN	it is an ungulate AND
		it is even-toed
R9	IF	the animal is a carnivore
		the animal has a tawny colour
		the animal has dark spots
	THEN	it is a cheetah
R10	IF	the animal is a carnivore
		the animal has a tawny colour

- Given these facts in working memory initially:
  - the animal gives milk
  - the animal chews its cud
  - the animal has long legs
  - the animal has a long neck
- Establish by forward chaining that the animal is a giraffe.

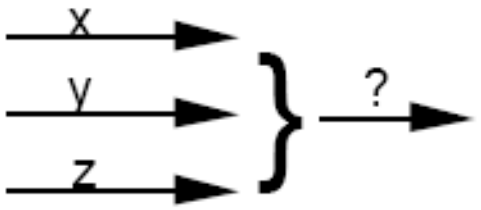
- Given the facts that:
  - the animal has hair
  - the animal has claws
  - the animal has pointed teeth
  - the animal's eyes point forward
  - the animal has a tawny colour
  - the animal has dark spots
- Establish by backward chaining that the animal is a cheetah.

- Data-driven search is suggested if:
  - All or most of the data is given in the problem statement (interpretation problems)
  - Large number of potential goals but few achievable in a particular problem instance.
  - It is difficult to formulate a goal or hypothesis.
  - Data-driven search can appear aimless but produces all solutions to a problem (if desired)

- Goal-driven search is suggested if:
  - A goal or hypothesis is given in the problem statement or can be easily formulated
  - (theorem-proving, diagnosis hypothesis testing).
  - There are a large number of rules that match the facts, producing a large number of conclusions - choosing a goal prunes the search space.
  - Problem data are not given (or easily available) but must be acquired as necessary (e.g. medical tests).

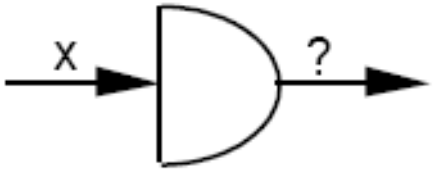
Handling uncertainty in Rule-based ES

- Uncertainty arises from abductive rules, heuristic rules or missing or unreliable data.
- To reason with uncertainty, we attach confidence measures to facts and to rules.



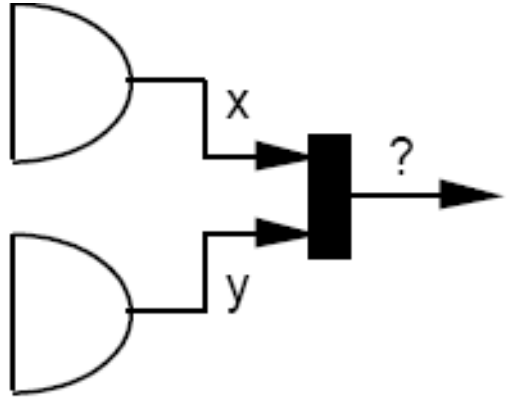
(1)

What confidence can I have in the conjunction of uncertain facts ?



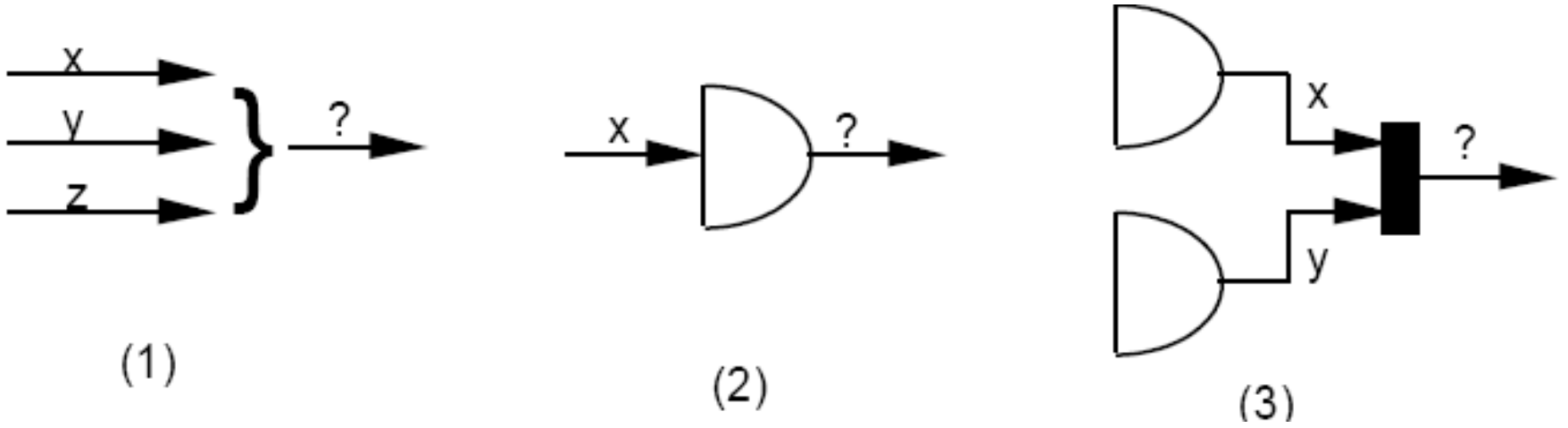
(2)

What confidence can I have in a particular rule's conclusion given the confidence of its premise ?



(3)

How do I combine confidences for supporting arguments ?



(1) If A and B and C then …

- If my confidence in A is x and in B is y and in C is z how confident am I about their conjunction (A and B and C)?
- S

(2) If D then E

- If my confidence in D is x how confident can I be in E?

(3) If the same fact F is deduced from (two) separate rules with confidences x and y,

- how confident am I in F?

A. Simple (Conservative) Rules

(1) Confidence in the conjunction is

min (x, y z)	a chain is as strong as its weakest link

(2) Confidence in rule conclusion is

x . *a*	*a* is the rule's attenuation factor

(3) Confidence in the multiply derived fact is

max (x, y)	a conclusion is no more certain than the strongest supporting argument.

B. Bayesian Probability Theory Based

Confidence measures are probabilities.

(1) Confidence in conjunction is $= x \cdot y \cdot z$

(2) Confidence in a rule is based on Bayes theorem

$$P(D_i,S)) = \frac{P(S|\ D_i) * P(D_i)}{\sum_{k=1}^{n} (P(S|\ D_k) * P(D_k))}$$

(there should be a summation symbol on the bottom line)

Uncertainty in MYCIN

If A: stain is gram positive
and B: morphology is coccus
and C: growth conformation is chains
then there is suggestive evidence (0.7) that
H: organism is streptococcus

0.7 is the measure of increase of belief (MB) of H given evidence A and B and C.

MB ranges 0 to 1.

Assigned by subjective judgement usually.

$$MB(H|E) = \frac{max[P(H|E),P(H)] - P(H)}{max[1,0] - P(H)}$$

Measures of **disbelief** also allowed. These also range 0 to 1.

$$MD(H|E) = \frac{min[P(H|E),P(H)] - P(H)}{min[1,0] - P(H)}$$

At the end a certainty factor CF = MB-MD is computed for each hypothesis.

The largest absolute CF values used to determine appropriate therapy. Weakly supported hypotheses $|CF| < 2$ are ignored.

Explanation Facility in Expert Systems

- Rule-based systems can be designed to answer questions like
  - HOW did you deduce this fact ? (i.e. how did we get here?)
- Explanation facilities are useful for debugging a rule base and for instilling confidence in users of the ES.
  - A tracer module records the rules that have been used. To answer HOW questions these rules are searched for one where the fact in question is a consequent → study MYCIN
  - The production system architecture provides the essential basis for explanation facility, and the facility contributes to the success and popularity of rule-based ES.

Expert System Shells

- An expert system shell is an expert system with an empty knowledge base, i.e.
  - An inference engine
  - User interface module
  - Tracer/explanation module
  - Knowledge base (rule) editor
  - Etc.
- Example:
  - EMYCIN is the shell of MYCIN
  - Jess
  - CLIPS