Lecture 3: Logic and Rule-Based Reasoning

Dr. Roman V Belavkin

BIS3226

Contents

1 Introduction to Production Systems

Historical Background

400 B.C. Aristotle's work provide foundation of two-valued logic: every proposition is either True or False (excluded middle).

1847 Bool, G., defines the calculus of deductive reasoning, which we now call BOOLEAN LOGIC.

1943 Post, E. L. proved that solutions to enumerable problems can be represented by sets of IF-THEN rules.

1961 GENERAL PROBLEM SOLVER (GPS) by A. Newell and H. Simon.

1969 DENDRAL (Feigenbaum, Buchanan, Lederberg) was the first system that showed the importance of *domain-specific knowledge* (expertise).

1970s MYCIN (Buchanan & Shortliffe) medical diagnosis system introduced the use of *certainty factors*.

1982 R1 (aka XCON) by McDermott was the first commercial ES (by 1986 it was saving DEC $40 millions p.a.).

1970–90 First cognitive architectures: ACT (J. Anderson), SOAR (A. Newell).

1990–200x further development of ACT-R.

Intelligence vs Expertise

Question 1. *Consider the following experts: A doctor, chess master, financial wizard, a chef.*
What is different between them?
Can you say that one is more intelligent than another?

- Expertise and intelligence are not the same things (although they are related).
- Expertise requires long time to learn (e.g. it takes 6 years to become a doctor).
- Expertise is a large amount of knowledge (in some domain).
- Expertise is easily recalled.
- Intelligence allows you to use your expertise (apply the knowledge).
- Expertise enables you to find solutions much faster.

Production Systems

Definition 1 (Production system (rule-based system))**.** is a computer program that operates using a set of IF-THEN rules (*production rules*).

Example 2.

```
IF   saturday OR sunday                  THEN   go to cinema
IF   NOT (saturday OR sunday)            THEN   go to work
IF   go to cinema                        THEN   go outside
IF   go to work AND NOT at work          THEN   go outside
IF   NOT (can go outside)                THEN   stay home
IF   good weather                        THEN   can go outside
IF   raining                             THEN   have an umbrella
IF   raining AND have an umbrella        THEN   can go outside
```

- The order of rules is not specified.
- The condition defines if a rule applies or not.
- Production systems have become popular in AI programming, such as expert systems, agents and cognitive models.

2 Elements of Boolean Logic

Boolean logic and Boolean functions

Definition 3 (Boolean variable)**.** is any variable a, b that can only have two values:

$$a \in \{0,1\}\,, \qquad b \in \{\texttt{False},\texttt{True}\}$$

Definition 4 (Boolean function)**.** is a function $f : A \to B$ between Boolean variables.

- *Boolean logic* can be described as a complete system of Boolean functions that can be derived (or represented) using three elementary Boolean functions (operations):

  $\neg$ not (negation)

  $\wedge$ and (conjunction)

  $\vee$ or (disjunction)

- Boolean logic is equivalent to the logic (algebra) of sets.

Negation (NOT, $\neg$)

- Let a be a Boolean variable

$$a \in L = \{0, 1\} = \{\texttt{False}, \texttt{True}\}$$

  Definition 5 ($\neg : L \to L$)**.**

a	$\neg a$
0	1
1	0

- The value of $\neg a$ is $1 - a$.

- Negation is equivalent to the complement of a set (the set of all elements $b \notin A$). If U is the universal set, and $A \subset U$, then the complement of A is

$$\bar{A} = U - A$$

Question 2 (Double negation)**.** *What is the value of* $\neg\neg a =$?

Conjunction (AND, $\wedge$)

Definition 6 ($\wedge : L \times L \to L$)**.**

a	b	$a \wedge b$
0	0	0
1	0	0
0	1	0
1	1	1

- The value of $a \wedge b$ is the *minimum* of $\{a, b\}$.

- Conjunction is equivalent to the intersection of sets

$$A \cap B$$

Question 3 (Law of contradiction)**.** *What is the value of*

$$a \wedge \neg a =?$$

Disjunction (OR, $\vee$)

Definition 7 ($\vee : L \times L \to L$)**.**

a	b	$a \vee b$
0	0	0
1	0	1
0	1	1
1	1	1

- The value of $a \vee b$ is the *maximum* of $\{a, b\}$.
- Disjunction is equivalent to the union of sets

$$A \cup B$$

Question 4 (Law of the excluded middle)**.** *What is the value of*

$$a \vee \neg a = ?$$

Implication ($\to$, $\Rightarrow$)

Definition 8 ($\Rightarrow : L \times L \to L$)**.**

a	b	$a \Rightarrow b$
0	0	1
1	0	0
0	1	1
1	1	1

- The value of $a \Rightarrow b$ is the same as $\neg a \vee b$.
- Disjunction is equivalent to set inclusion

$$A \subset B$$

Question 5 (Inverse implication)**.** *Which of the implications below is equivalent to $a \Rightarrow b$?*

$$\neg a \Rightarrow \neg b \quad or \quad \neg a \Leftarrow \neg b$$

Equivalence ($\sim$)

Definition 9 ($\sim : L \times L \to L$)**.**

a	b	$a \sim b$
0	0	1
1	0	0
0	1	0
1	1	1

- The value of $a \sim b$ is the same as $(a \vee b) \wedge (\neg a \vee \neg b)$.
- Logical equivalence is the same as set equivalence:

$$A \subset B \quad \text{and} \quad A \supset B\,, \quad A \equiv B$$

Summary of Elementary Logical Operations

a	b	$\neg a$	$a \wedge b$	$a \vee b$	$a \Rightarrow b$	$a \sim b$
0	0	1	0	0	1	1
1	0	1	0	1	0	0
0	1	0	0	1	1	0
1	1	0	1	1	1	1

Duality (De Morgan's) laws :

$$\neg(a \wedge b) = \neg a \vee \neg b \,, \quad \neg(a \vee b) = \neg a \wedge \neg b$$

Absorption laws :

$$a \wedge (a \vee b) = a \,, \quad a \vee (a \wedge b) = a$$

3 Problem Solving

Problem Solving

- A problem can be described as some set of states $\omega \in \Omega$ or configurations of a system (the problem space) with one or several initial states ω_0 and one or several goal states g.
- At each moment $t \in [0, T]$ (step, cycle) the problem solver performs some actions that result in transitions

$$\omega_t \mapsto \omega_{t+1}$$

- The problem exists while

$$\omega_t \neq g$$

- The problem is solved when the goal is reached:

$$\omega_t = g \quad \text{or} \quad \omega_t \sim g$$

Remark 1. *Problem space can be considered as a choice set $(\Omega, \lesssim)$, and the goal state as the top element $\top \in \Omega$ (the optimal choice).*

The 'Water Jugs' Problem

- You have one 8–litre jug full of water.
- You also have two empty jugs: one can hold 3 and the other 5 litres.
- The goal is to share the water equally between two people:

$$8 \mapsto 4 + 4$$

- We can represent a problem state as (a, b, c), where a, b and c are the amounts of water in the 3, 5 and 8–litre jugs respectively.
- Initial state is $(0, 0, 8)$, and the goal state is $(0, 4, 4)$.

The 'Water Jugs' Problem Space

$$(0,8,0)$$
$$(1,7,0)(0,7,1)$$
$$(2,6,0)(1,6,1)(0,6,2)$$
$$(3,5,0)(2,5,1)(1,5,2)(0,5,3)$$
$$(4,4,0)(3,4,1)(2,4,2)(1,4,3)(0,4,4)$$
$$(5,3,0)(4,3,1)(3,3,2)(2,3,3)(1,3,4)(0,3,5)$$
$$(6,2,0)(5,2,1)(4,2,2)(3,2,3)(2,2,4)(1,2,5)(0,2,6)$$
$$(7,1,0)(6,1,1)(5,1,2)(4,1,3)(3,1,4)(2,1,5)(1,1,6)(0,1,7)$$
$$(8,0,0)(7,0,1)(6,0,2)(5,0,3)(4,0,4)(3,0,5)(2,0,6)(1,0,7)(0,0,8)$$

4 Development and Operation of ES

Declarative and Procedural Knowledge

The knowledge in production systems is often divided into:

Declarative : these are propositions or facts describing the current state of the problem (e.g. which facts are known to be true).

Procedural : these are logical rules (implications $a \Rightarrow b$):

IF *condition* THEN *action*

The rules are used to change a problem state (e.g. by inferring new facts). The IF part is called the *left-hand-side* (or the *antecedent*, *premise*). The THEN part is called the *right-hand-side* (or the *consequent*).

Remark 2. *Evidence suggests that these types of knowledge are encoded in different parts of the brain or even use different mechanisms. Procedural knowledge is usually difficult to describe, but also harder to forget.*

Declarative

Example 10. Barney is a dog Dog has four legs 4 is a sum of 2 and 2

- In ACT–R, declarative facts are called *chunks*, and they are encoded as

```
(<name> ISA <chunk-type> slot1 <value> ...)
```

- The *chunk-type* should be defined

```
(chunk-type <name> <slot1> <slot2> ...)
```

Example 11. `(chunk-type dog name legs)`
`(barney ISA dog name Barney legs four)`

Procedural

Example 12.

IF	*Barney is a dog*	THEN	*Barney has four legs*
IF	*the price is high*	THEN	*sell*

- In Act–r, rules have the form

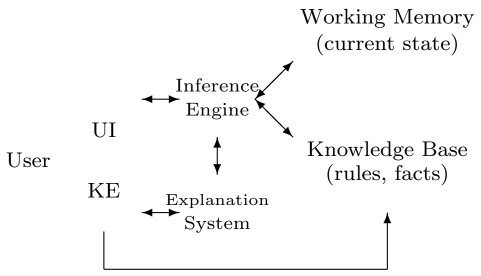
  ```
  (p <condition>
  ==>
     <action>
  )
  ```

- The condition part usually starts with the `=goal>` chunk and then some other chunks.

Example 13.
```
(p =goal>
    isa dog
==>
   =goal>
    legs four)
```

Architecture of ES

Main components of a classical rule-based Expert System (ES):



Knowledge base, working memory, inference engine, explanation system, user interface and knowledge base editor.

Main Components of ES

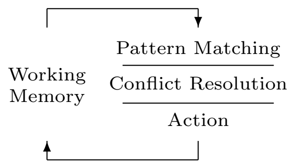
Knowledge Base contains all the knowledge of ES in a form of rules (procedural knowledge) and facts (declarative knowledge). Can be compared with a hard disk drive of PC, or long-term memory of the human brain.

Working Memory contains only the facts describing the current state of a problem. Can be compared with RAM of PC, or short-term memory of the human brain.

Inference Engine implements the reasoning process. In brief, it finds rules in the knowledge base that correspond to the contents of working memory and applies them to the problem.

Recognise-Act Cycle

Reasoning in rule-based systems occurs in cycles.
Each cycle consists of three stages:



Pattern Matching : (recognition) the contents of the working memory is compared with the rules. All rules that *match* the current problem state are selected into a *conflict set*.

Conflict Resolution : a single rule from the conflict set is selected.

Action : the action part of the selected rule is performed. It may result in changing the working memory.

Example: Weather Forecast ES

1	IF	*cyclone*	THEN	*clouds*
2	IF	*anticyclone*	THEN	*clear sky*
3	IF	*pressure is low*	THEN	*cyclone*
4	IF	*pressure is high*	THEN	*anticyclone*
5	IF	*arrow is down*	THEN	*pressure is low*
6	IF	*arrow is up*	THEN	*pressure is high*

Example: Weather Forecast ES

Cycle	Working Memory	Conflict set	Rule fired
0	arrow is up	6	6
1	arrow is up, pressure is high	6, 4	4
2	arrow is up, pressure is high, anticyclone	6, 4, 2	2
3	arrow is up, pressure is high, anticyclone, clear sky	6, 4, 2	Halt

Conflict Resolution

Sometimes, several rules can match the working memory, but only one has to be selected (hence the conflict). There are several strategies to deal with conflict resolution:

Refraction : once the rule has fired, it is not used again.

Recency : use the rule that has been used recently in such situation.

Specificity : use the rule with the more specific condition (more facts).

Priority : assign priority to rules (i.e. rank, utility, probability, cost, etc) and choose the one with the highest priority.

Parallel : fire all rules with separate lines of reasoning.

Other Components of ES

User interface may provide interaction facilities, where ES asks the user some questions. The answers are interpreted into facts in working memory and used in reasoning.

Knowledge base editor gives the possibility to examine and edit the knowledge base.

Rule-based ES can explain their reasoning by showing the trace of the rules applied to solve the problem. This is useful because human can understand the logics behind the solution.

5 Discussion

Limitations of Rule-Based Systems

IF	*x has wings*	THEN	*x is a bird*
IF	*x is a bird*	THEN	*x can fly*

Question 6. *How about an ostrich?*

- Rules may have exceptions.
- The knowledge can be task-specific and not transferable to other domains.
- There are other types of ES: model-based, case-based.

General (Weak) Methods

- In 1961, A. Newell and H. Simon wrote a program called *General Problem Solver* (GPS) that could solve many different problems using only a small set of rules.
- GPS used a strategy known as *means-ends analysis*.
- GPS produced solutions very similar to those people came up with.
- Methods that can be applied to a broad range of problems are called **weak** methods, because they use weak information about the problem domain, and their performance is also usually weak.

Knowledge-Based (Strong) Methods

- DENDRAL (Feigenbaum et al, 1969) was a program that used rules to infer molecular structure from spectral information. The challenge was that the number of possible molecules was so large, that it was impossible to check all of them using simple rules (weak method).
- The researchers consulted experts in chemistry and added several more specific rules to their program. The number of combinations the program had to test was reduced dramatically.
- DENDRAL demonstrated the importance of the *domain-specific* knowledge.

Knowledge Engineering

The process of designing an ES is called **knowledge engineering**. It consist of three stages:

Knowledge acquisition : the process of obtaining the knowledge from experts (by interviewing and/or observing human experts, reading specific books, etc).

Knowledge representation : selecting the most appropriate structures to represent the knowledge (lists, sets, scripts, decision trees, object-attribute-value triplets, etc).

Knowledge validation : testing that the knowledge of ES is correct and complete.

Main Areas of Application

The main areas of application of ES are (Waterman, 1986):

Interpretation — drawing high-level conclusions based on data.

Prediction — projecting probable outcomes.

Diagnosis — determining the cause of malfunctions, disease, etc.

Design — finding best configuration based on criteria.

Planning — proposing a series of actions to achieve a goal.
Monitoring — comparing observed behaviour to the expected behaviour.
Debugging and Repair — prescribing and implementing remedies.
Instruction — assisting students in learning.
Control — governing the behaviour of a system.

Advantages and Limitations of ES

Advantages:

- Increased productivity (find solutions much faster than humans).
- Availability of expertise (human experts can be at one place at a time).
- Can be used in dangerous environments (e.g. in space).

Limitations:

- Difficulty in engineering, especially acquiring the expertise.
- Mistrust by the users.
- Effective only in specific areas (areas of expertise).