인공지능

22년 삼성 AI 전문가과정
6월 7일 화요일 6교시
장병탁

7차시 : Rule-Based Systems

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang

Lecture Overview

인공지능

7차시 : Rule-Based Systems

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang

Introduction

- **First-Order Logic** (Previous lecture)
  - Intractable in practical applications
  - Confronting the real world
- **Horn clauses (PROLOG)**
  - Less expressive
  - More efficient
- **Rule-based expert systems**
  - Rule-based systems
  - Expert systems
  - Knowledge-based systems

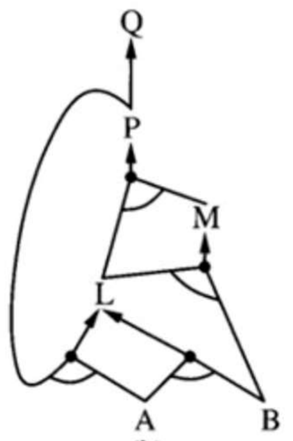
Reasoning Using Horn Clauses

- **Proof by AND/OR Tree**

Horn Clause

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

(a)

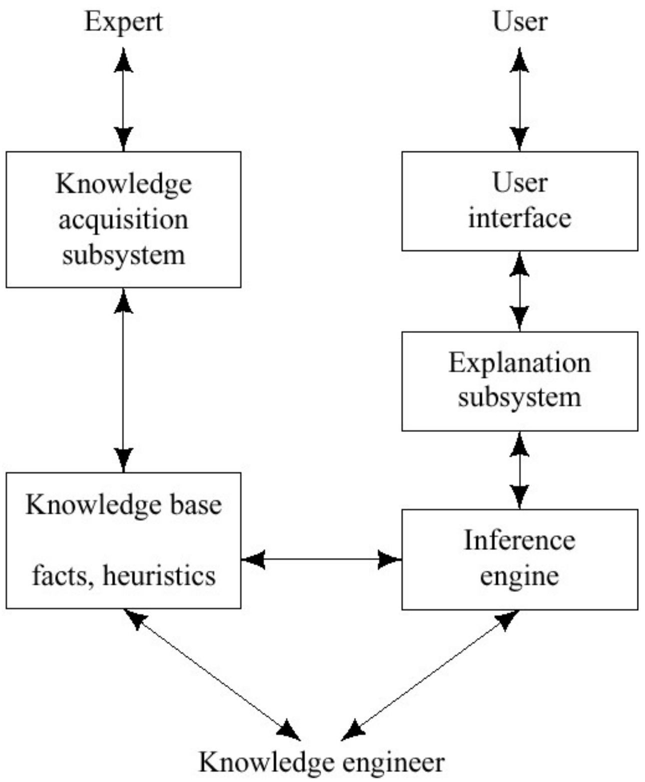


(b)

Figure 7.15 (a) A simple knowledge base of Horn clauses. (b) The corresponding AND–OR graph.

Rule-Based Expert Systems

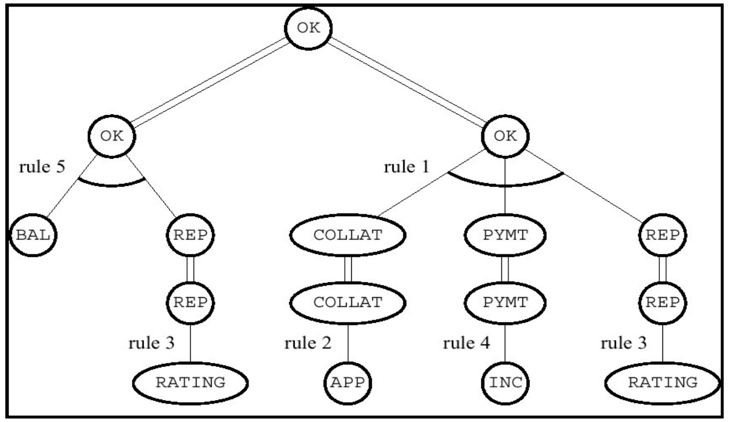
- Expert Systems
  - Inference engine & Knowledge base
  - "AI Programs that achieve expert-level competence in solving problems by bringing to bear a body of knowledge" [Feigenbaum, McCorduck & Nii 1988]
- Expert systems vs. knowledge-based systems
- Rule-based expert systems
  - Often based on reasoning with propositional logic Horn clauses.



Lecture Overview

- Confronting the Real World
  - Knowledge-based system, Predicate calculus
- Reasoning Using Horn Clauses
  - Horn Clauses, PROLOG programs
- Rule-Based Expert Systems
  - Expert Systems, Consulting system

1. $COLLAT \wedge PYMT \wedge REP \supset OK$
2. $APP \supset COLLAT$
3. $RATING \supset REP$
4. $INC \supset PYMT$
5. $BAL \wedge REP \supset OK$



Outline (Lecture 7)

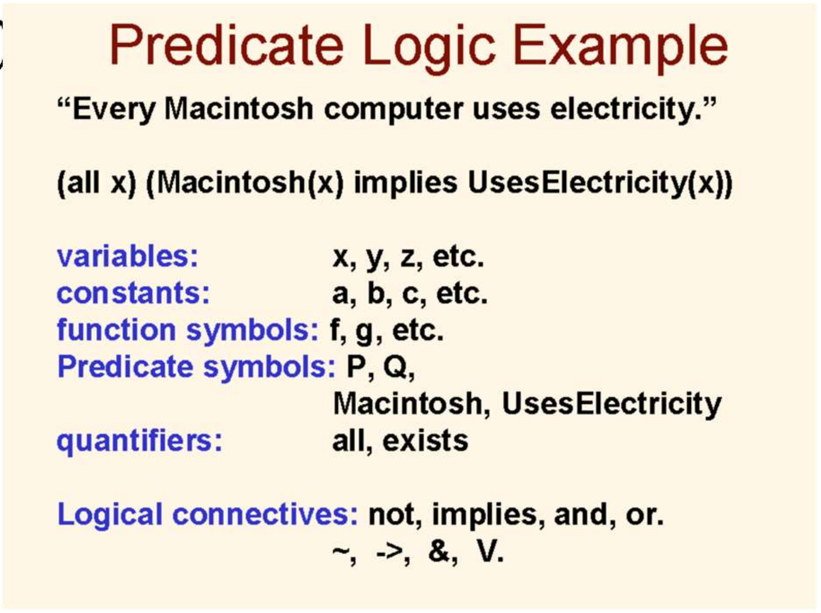
7.1 Confronting the Real World

7.1 Confronting the Real World (1/3)

- Knowledge-based systems
  - Programs that reason over extensive knowledge bases.
  - Do the methods scale up sufficiently well for practical applications?
- Three major theoretical properties of logical reasoning systems
  - Soundness
    - To be confident that an inferred conclusion is true
  - Completeness
    - To be confident that inference will eventually produce any true conclusion
  - Tractability
    - To be confident that inference is feasible

7.1 Confronting the Real World (2/3)

- **Predicate calculus (first-order logic)**
  - Resolution refutation is sound and complete, but semi-decidable.
    - Semi-decidability makes the predicate calculus inherently intractable.
    - Even on problems for which resolution refutation terminates, the procedure is NP-hard.
    - This fact has led many to despair of using formal, logical methods for large-scale reasoning problems.



Predicate Logic Example

"Every Macintosh computer uses electricity."

(all x) (Macintosh(x) implies UsesElectricity(x))

variables:	x, y, z, etc.
constants:	a, b, c, etc.
function symbols:	f, g, etc.
Predicate symbols:	P, Q, Macintosh, UsesElectricity
quantifiers:	all, exists
Logical connectives:	not, implies, and, or. ~, ->, &, V.

Predicate logic example. (n.d.). Retrieved February 23, 2022, from https://courses.cs.washington.edu/courses/cse341/03sp/slides/Prolog2/sld006.htm

7.1 Confronting the Real World (3/3)

- To make reasoning more efficient
  - We could use procedures that might occasionally prove an untrue formula.
  - We could use procedures that might not be guaranteed to find proofs of true formulas.
  - We could use a language that is less expressive than the full predicate calculus.
    - Reasoning is typically more efficient with Horn clauses.

7.2 Reasoning Using Horn Clauses

7.2 Reasoning Using Horn Clauses (1/6)

- **Horn Clauses**
  - Clauses that have at most one positive literal.
  - Rule: λ_h : - $\lambda_{b_1}, \ldots, \lambda_{b_n}$
    - There is at least one negative literal and a single positive literal.
    - The literal λ_h is called the head of the clause.
    - The ordered list of literals, $\lambda_{b1}, \ldots, \lambda_{bn}$, is called body.
  - Fact λ_h : -
  - There may be no negative literals in the clause.
  - Goal :- $\lambda_{b_1}, \ldots, \lambda_{b_n}$
    - There may be no positive literal in the clause.
  - Horn clauses form the basis of the programming language PROLOG.

Horn Clause

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

(a)

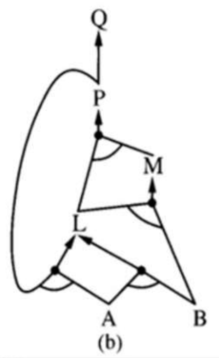


Figure 7.15 (a) A simple knowledge base of Horn clauses. (b) The corresponding AND–OR graph.

```
1.  :- MOVES
2.  BAT_OK :-
3.  LIFTABLE :-
4.  MOVES :- BAT_OK, LIFTABLE
```

7.2 Reasoning Using Horn Clauses (2/6)

Inference over PROLOG Clauses (1/2)

- **Resolution Operation**
  - A goal can resolve with a fact by unifying the fact with one of the literals in the goal.

    The resolvent is a new goal consisting of a list of all of the substitution instances of the other literals in the original goal.
  - A goal can resolve with a rule by unifying the head of the rule with one of the literals in the goal.

    The resolvent is a new goal formed by appending the list of substitution instances of all of the literals in the body of the rule to the front of the list of substitution instances of all of the other (nonresolved-upon) literals in the goal.

```
1. :- MOVES
2. BAT_OK :-
3. LIFTABLE :-
4. MOVES :- BAT_OK, LIFTABLE
```

7.2 Reasoning Using Horn Clauses (3/6)

Inference over PROLOG Clauses (2/2)

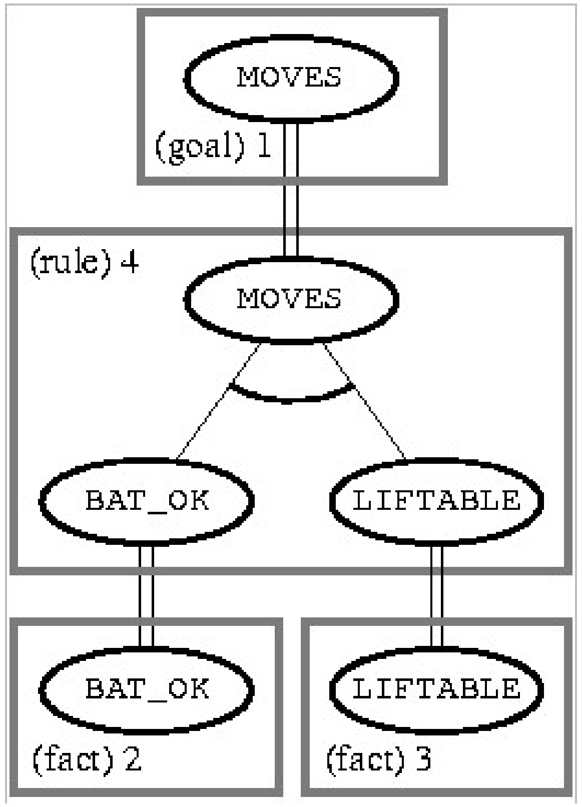
- **PROLOG programs**
  - The clauses are usually ordered in the following way: goal, facts, rules.
  - Proof of a goal clause succeeds when the new goal produced by a resolution is empty.
  - Depth-first, backtracking search procedure
    - A goal clause fails if the interpreter has tried all resolutions for one of the goal literals and none results in new goals that can be proved.
    - The interpreter backtracks to the previous goal clause and tries other resolutions on it.

7.2 Reasoning Using Horn Clauses (4/6)

Example of an AND/OR Tree (1/2)

- Body nodes are called *AND* nodes because they must all be proved.
- If there had been alternative resolutions possible, the search for a proof tree could have generated additional nodes, called *OR* nodes.

```
1.  :- MOVES
2.  BAT_OK :-
3.  LIFTABLE :-
4.  MOVES :- BAT_OK, LIFTABLE
```

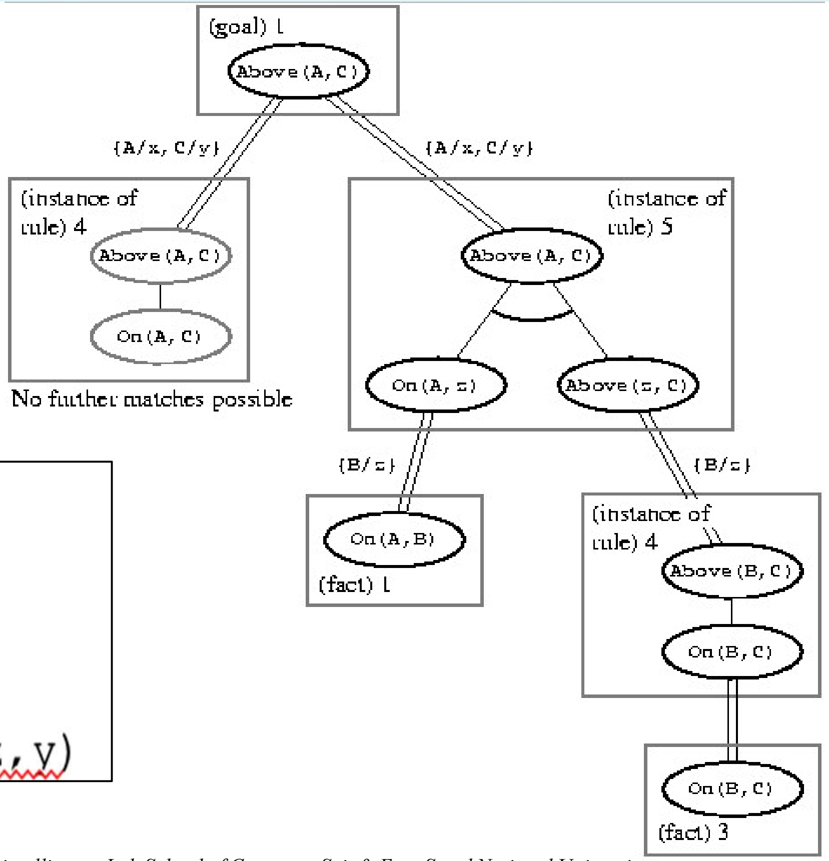


7.2 Reasoning Using Horn Clauses (5/6)

Example of an AND/OR Tree (2/2)

- Consistency requires that the same term be substituted for the variable throughout the proof tree.

```
1.  :- Above(A,C)
2.  On(A,B)  :-
3.  On(B,C)  :-
4.  Above(x,y)  :- On(x,y)
5.  Above(x,y)  :- On(x,z), Above(z,y)
```



7.2 Reasoning Using Horn Clauses (6/6)

Forward chaining system (e.g., OPS5)

- Reasoning proceeds forward, beginning with facts, chaining through rules, and finally establishing the goal.
  - A rule is applicable if each of the literals in its antecedent can be unified with a corresponding fact.
  - When more than one rule is applicable, some sort of external *conflict resolution* scheme is used to decide which rule will be applied.
- Model of aspects of human cognitive processing
  - Facts: *working* or *short-term* memory
  - Rules: *long-term* memory

7.3 Rule-Based Expert Systems

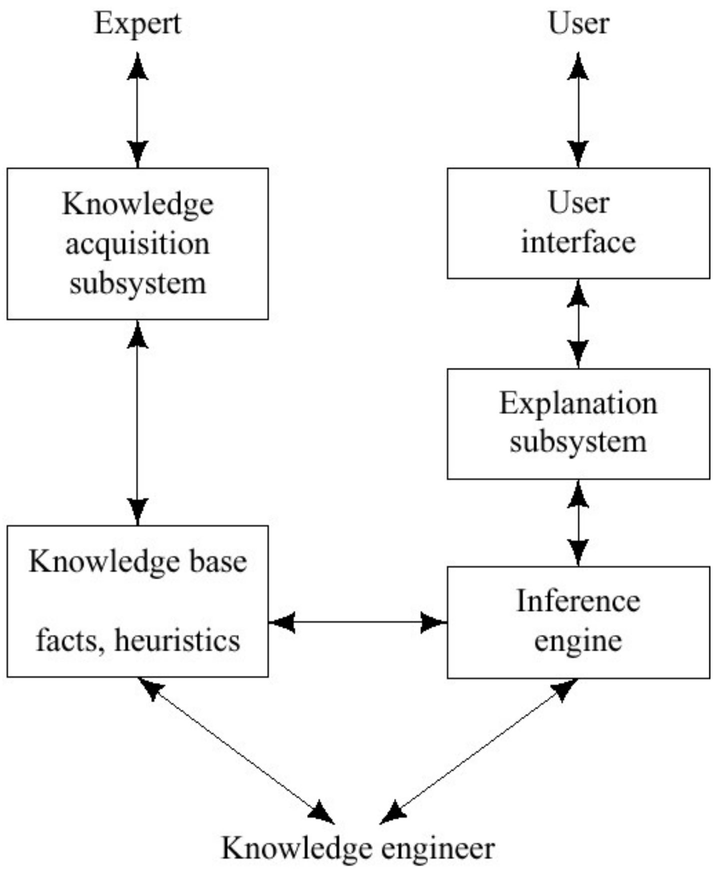
7.3 Rule-Based Expert Systems (1/9)

- ***Expert Systems***
  - One of the most successful applications of AI reasoning technique using facts and rules
  - "AI programs that achieve expert-level competence in solving problems by bringing to bear a body of knowledge [Feigenbaum, McCorduck & Nii 1988]"
- **Expert systems vs. knowledge-based systems**
- **Rule-based expert systems**
  - Often based on reasoning with propositional logic Horn clauses.

7.3 Rule-Based Expert Systems (2/9)

Structure of Expert Systems

- Knowledge Base
  - Consists of predicate-calculus facts and rules about subject at hand.
- Inference Engine
  - Consists of all the processes that manipulate the knowledge base to deduce information requested by the user.
- Explanation subsystem
  - Analyzes the structure of the reasoning performed by the system and explains it to the user.



7.3 Rule-Based Expert Systems (3/9)

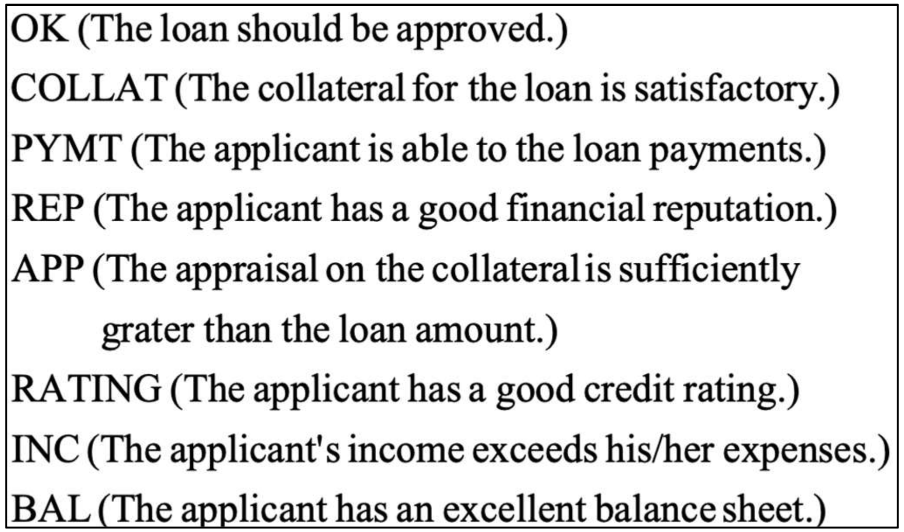
Structure of Expert Systems

- Knowledge acquisition subsystem
  - Checks the growing knowledge base for possible inconsistencies and incomplete information.
- User interface
  - Consists of some kind of natural language processing system or graphical user interfaces with menus.
- “Knowledge engineer”
  - Usually a computer scientist with AI training.
  - Works with an expert in the field of application in order to represent the relevant knowledge of the expert in a forms of that can be entered into the knowledge base.

7.3 Rule-Based Expert Systems (4/9)

Example: loan officer in a bank

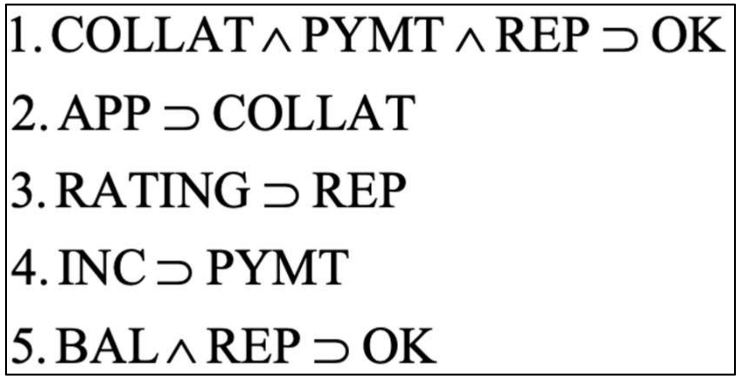
- *“Decide whether or not to grant a personal loan to an individual.”*



OK (The loan should be approved.)
COLLAT (The collateral for the loan is satisfactory.)
PYMT (The applicant is able to the loan payments.)
REP (The applicant has a good financial reputation.)
APP (The appraisal on the collateral is sufficiently
grater than the loan amount.)
RATING (The applicant has a good credit rating.)
INC (The applicant's income exceeds his/her expenses.)
BAL (The applicant has an excellent balance sheet.)

- Facts

- Rules



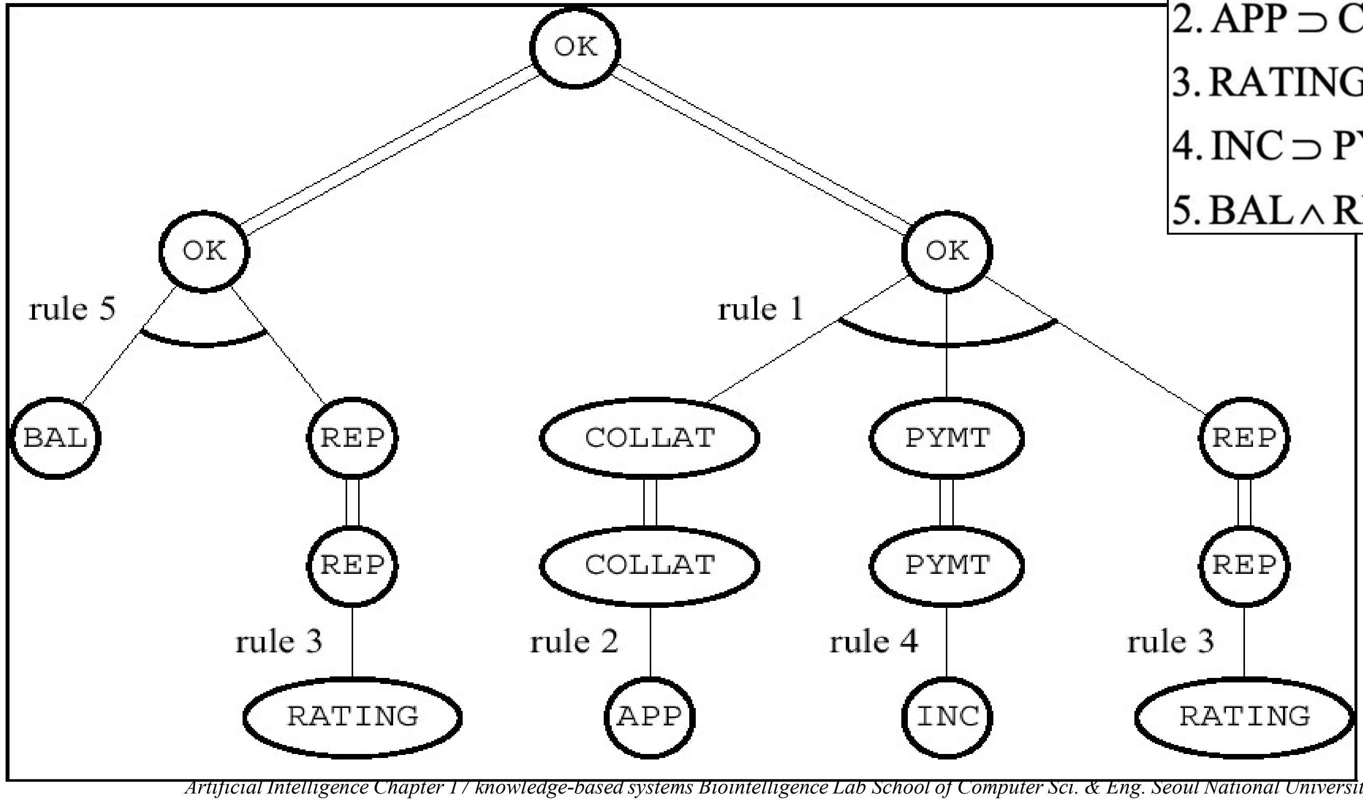
1. $\text{COLLAT} \wedge \text{PYMT} \wedge \text{REP} \supset \text{OK}$
2. $\text{APP} \supset \text{COLLAT}$
3. $\text{RATING} \supset \text{REP}$
4. $\text{INC} \supset \text{PYMT}$
5. $\text{BAL} \wedge \text{REP} \supset \text{OK}$

7.3 Rule-Based Expert Systems (5/9)

- **To prove OK**
  - The inference engine searches for AND/OR proof tree using either backward or forward chaining.
- **AND/OR proof tree**
  - Root node: OK
  - Leaf node: facts
  - The root and leaves will be connected through the rules.
- **Using the preceding rule in a backward-chaining**
  - The user's goal, to establish OK, can be done *either* by proving *both* BAL and REP or by proving *each* of COLLAT, PYMT, and REP.
  - Applying the other rules, as shown, results in other sets of nodes to be proved.

7.3 Rule-Based Expert Systems (6/9)

By backward-chaining



1. $\text{COLLAT} \wedge \text{PYMT} \wedge \text{REP} \supset \text{OK}$
2. $\text{APP} \supset \text{COLLAT}$
3. $\text{RATING} \supset \text{REP}$
4. $\text{INC} \supset \text{PYMT}$
5. $\text{BAL} \wedge \text{REP} \supset \text{OK}$

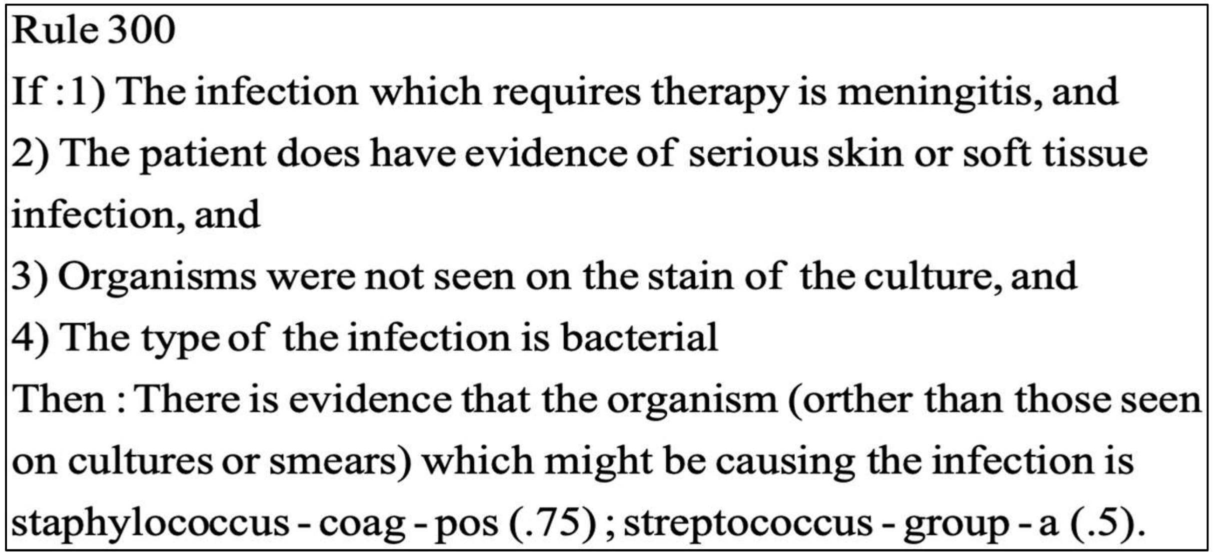
7.3 Rule-Based Expert Systems (7/9)

Consulting system

- Attempt to answer a user's query by asking questions about the truth of propositions that they might know about.
- Backward-chaining through the rule is used to get to askable questions.
- If a user were to "volunteer" information, bottom-up, forward chaining through the rules could be used in an attempt to connect to the proof tree already built.
- The ability to give explanations for a conclusion
  - Very important for acceptance of expert system advice.
- Proof tree
  - Used to guide the explanation-generation process.

7.3 Rule-Based Expert Systems (8/9)

- In many applications, the system has access only to uncertain rules, and the user not be able to answer questions with certainty.
- **MYCIN** [Shortliffe 1976]: Diagnose bacterial infections.

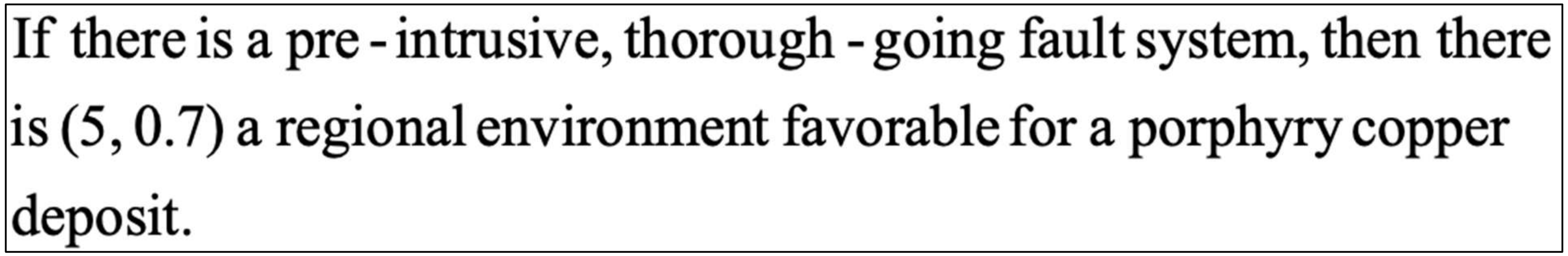


Rule 300
If : 1) The infection which requires therapy is meningitis, and
2) The patient does have evidence of serious skin or soft tissue infection, and
3) Organisms were not seen on the stain of the culture, and
4) The type of the infection is bacterial
Then : There is evidence that the organism (orther than those seen on cultures or smears) which might be causing the infection is staphylococcus - coag - pos (.75) ; streptococcus - group - a (.5).

Artificial Intelligence Chapter 17 Knowledge-based Systems Biointelligence Lab School of Computer Sci. & Eng. Seoul National University. - ppt download. SlidePlayer. (n.d.). Retrieved February 23, 2022, from https://slideplayer.com/slide/5260453/

7.3 Rule-Based Expert Systems (9/9)

- PROSPECTOR [Duda, Gaschnig & Hart 1979, Campbell, et al. 1982]
  - Reason about ore deposits.


If there is a pre-intrusive, thorough-going fault system, then there is (5, 0.7) a regional environment favorable for a porphyry copper deposit.

- The numbers (.75 and .5 in MYCIN, and 5, 0.7 in PROSPECTOR) are ways to represent the certainty or strength of a rule.
- The numbers are used by these systems in computing the certainty of conclusions.

Summary

- Knowledge-based systems are programs that reason over extensive knowledge bases.
- Three major theoretical properties of logical reasoning systems.
  → 1) Soundness, 2) Completeness, 3) Tractability
- Reasoning is typically more efficient with Horn clauses.
- Forward, backward chaining are linear-time, complete for Horn clauses. Resolution is complete for propositional logic.
  - Reasoning proceeds forward, beginning with facts, chaining through rules, and finally establishing the goal.
- Expert Systems: One of the most successful applications of AI reasoning technique using facts and rules.
  → Knowledge base, inference engine, explanation subsystem
- Inductive rule learning: Creates new rules (Ex. neural networks)
- Deductive rule learning: Enhances the efficiency of a system's performance (Ex. EBG)