Intelligent Decision Support Systems

(Part XI – MODEL-DRIVEN TECHNIQUES IN DECISION SUPPORT: EXPERT-BASED MODELS (2))

Miquel Sànchez-Marrè

Intelligent Data Science and Artificial Intelligence Research Centre (IDEAI-UPC)
Knowledge Engineering and Machine Learning Group (KEMLG-UPC)

Computer Science Dept.
Universitat Politècnica de Catalunya · Barcelona Tech

miquel@cs.upc.edu
http://www.cs.upc.edu/~miquel

Course 2020/2021

https://kemlg.upc.edu
PART 11 – MODEL-DRIVEN TECHNIQUES IN DECISION SUPPORT (2)

Expert-based Models

https://kemlg.upc.edu
Backward Reasoning

- It is an inductive method. The deductive direction is broken:

\[ B, A \rightarrow B \rightarrow \top A \]

- Led by a goal: Hypothesis to be validated. The reasoning chain must be reconstructed in reverse order.

- Each step implies new subgoals or subhypothesis required to be validated.
Backward Reasoning: Operation

- Initialize the *Fact Base* with an initial set of facts
- Initialize the *list of hypothesis /goals* to be verified
- **While** there are hypothesis to be verified **do**
  - Validate the first hypothesis of the list
- **EndWhile**

To validate the hypothesis means to:

- **if** it is already validated → remove it from the list
  - Check whether it is already verified in the *Fact Base*
- **else** use the *Knowledge Base* and the *Fact Base* to validate it

- Select one rule
- Add the premises of the rule as new sub-goals to be validated instead of the hypothesis
Backward Reasoning: Advantages

- The problem resolution is directed by the goals. Only the necessary knowledge and facts are considered to solve the problem.

- The problem solving is actually the exploration of an and/or graph.
### Backward Reasoning: Example 1 (I)

<table>
<thead>
<tr>
<th>Knowledge Base</th>
<th>Fact Base</th>
<th>Goal/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1: A ∨ B → C</td>
<td>A</td>
<td>H??</td>
</tr>
<tr>
<td>R2: C → D</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>R3: E ∨ F → G</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R4: A → E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R5: D → G</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R6: A ∨ G → H</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Backward Chaining: Example 1 (II)

H → A → R6
H → G

H → A → R6
H → G

H → A → E → R6
H → G

H → A → E → R6
H → G

H → A → E → R6
H → G

H → A → E → R6
H → G

H → A → E → R6
H → G

H → A → E → R6
H → G

R3
R5

R3
R5

R4

R4
Backward Reasoning: Example 1 (III)

Let us suppose that F can be asked

We must go back and reconsider other options
Backward Reasoning: Example 1 (IV)
Backward Reasoning: Example 1 (V)
Backward Reasoning: Example 2a (1)

Knowledge Base

R1: B(X,Y) → A(X,Y)
R2: A(X,Y) → A(Y,X)
R3: A(X,Y) ∧ A(Y,Z) → A(X,Z)

Fact Base

A(a,b)  B(a,c)
A(c,b)  B(a,b)
A(d,a)  B(b,c)
A(e,a)  B(d,a)
A(f,a)  B(f,a)

Conflict resolution strategy

First rule in order

None Fact is questionable

Goal

A(f,b)?
Backward Reasoning: Example 2a (1)

A(f,b) → A(b,f) → B(f,b) No → A(f,b)

A(f,b) cycle!!

B(b,y) Yes, y=c

A(y/c,f) → B(f,c) No → A(f,c)

A(c,f) cycle!!

B(c,f) No

B(b,y) No

R1

R2

R3
Backward Reasoning: Example 2a (2)

\[ (*) \quad R3 \quad A(f,y) \quad R1 \quad B(f,y) \text{ Yes, } y=a \]

\[ R3 \quad A(y/a,c) \quad R1 \quad B(a,c) \text{ Yes} \]

In 13 steps !!
Backward Reasoning: Example 2b

**Knowledge Base**

R1: $B(X,Y) \rightarrow A(X,Y)$
R2: $A(X,Y) \rightarrow A(Y,X)$
R3: $A(X,Y) \land A(Y,Z) \rightarrow A(X,Z)$

**Goal**

$A(f,b)\?$

**Fact Base**

$A(a,b)$  $B(a,c)$
$A(c,b)$  $B(a,b)$
$A(d,a)$  $B(b,c)$
$A(e,a)$  $B(d,a)$
$A(f,a)$  $B(f,a)$

**Conflict resolution strategy**

**Most specific rule**

None fact is questionable
Backward Reasoning: Example 2b (2)

A(f,b)

R3

A(f,y) Yes, y=a

A(y/a,b) Yes

In 1 step !!
Knowledge Base Modularization (1)

- A *module* is the basic unit of knowledge structuration. It is a set of related rules:
  - Similar or equal conclusions
  - Similar conditions
  - Treatment of the same subdomain

- Each *module* usually contains:
  - <The-module-identifier>
  - <List-of-conclusions/diagnostics of the module>
  - <The module rules>
  - <The module meta-rules>
  - ...

© Miquel Sànchez i Marrè, KEMLG-IDEAI, 2020
Knowledge Base Modularization (2)

- Example 1: Diagnosis task in a WWTP

\[
\text{(module MAIN} \\
\text{((RPR01} \\
\text{  IF (high-COD-exit) \\
\text{   (high-conc-Zn-inflow) \\
\text{   (high-conc-Zn-bioReactor) \\
\text{ THEN} \\
\text{   (class1)}) \\
\text{RPR02} \\
\text{  IF (normal-COD-exit) \\
\text{   (high-DO) \\
\text{ THEN} \\
\text{   (class2)}) \\
\text{...)} \\
\text{(MRCL1} \\
\text{  IF (class1) \\
\text{ THEN} \\
\text{   (EXPLORE-MODULES MCLASS1)) \\
\text{MRCL2} \\
\text{  IF (class2) \\
\text{ THEN} \\
\text{   (EXPLORE-MODULES MCLASS2)) \\
\text{...))))}
\]
Example 2: Identification task of the species of a tree

- Group of Genres
  - Genre
    - VCR
    - Cedrus
  - Species
    - Black Pine
    - Red Pine
    - White Pine
    - Pinioner Pine
    - Juniperus
    - Taxus
Knowledge Base Modularization (4)

(module Main

((R1
   IF (leaves-length average)
   (width-leaves <5cm)
   THEN (group PAJTC))

(R2
   IF (leaves-length average)
   (width-leaves >5cm)
   (leaves-simplicity 1)
   (number-main-nervous 1)
   (leaves-outline entire)
   THEN (group VCR))

...)

((MR1
   IF (group PAJTC)
   THEN (EXPLORE-MODULES PAJTC))

(MR2
   IF (group VCR)
   THEN (EXPLORE-MODULES VCR))

... ))

(module PAJTC

((RPAJCT1
   IF (number-sheathed-leaves 2)
   THEN (genre pinus))

(RPAJCT2
   IF (number-sheathed-leaves 1)
   THEN (group AJT)))

...)

((MRPAJCT1
   IF (genre pinus)
   THEN (EXPLORE-MODULES pinus)

(MRPAJCT2
   IF (group AJT)
   THEN (EXPLORE-MODULES AJT))))

(module Pinus

((RPinus1
   IF (top-of-the-tree umbrella)
   THEN (species pinioner-pine))

(RPinus2
   IF (top-of-the-tree other)
   (length-leave >3cm&<6cm)
   THEN (species red-pine black-pine))

... ))
Meta-Knowledge / Meta-Reasoning

- **Meta-reasoning** ≡ reasoning over the own reasoning
  - Controlling **how** and **when** to apply the knowledge
  - Implicit Meta-knowledge
    - Conflict resolution strategy (criteria)
    - In first-generation Expert Systems: Artificial Premises to control rule applicability
  - Explicit Meta-knowledge
    - Introduction of **meta-rules** (Davis, 1980): Rules acting over rules
    - Separation between **control** and **knowledge**
    - **Unified reasoning mechanism**: Inference engine used both by rules and by meta-rules
    - **Strategy** concept: Necessary elements ordered for the problem solving process
Meta-rules (1)

- A *meta-rule* is the control unit over the knowledge
- Different kinds of *meta-rules*:
  - Meta-rules over *rules*
    - Activate rules / deactivate rules
  - Meta-rules over *modules*
    - Kind of reasoning in the modules (forward, backward)
    - Cut level in the minimum certainty of the rules
    - Rule subsumption
  - Meta-rules over *strategies*
    - *Strategy*: ordered set of modules to be explored
    - Exceptions
  - Meta-rules over *actuation plans*
    - Which strategy should be applied first when more than one are available
Meta-rules (2)

- The antecedent will have the same form of a rule
- The consequent is composed by special propositions
- Admitted consequents usually are the following:
  - ACTIVATE-RULES <list of rule identifiers>
  - DEACTIVATE-RULES <list of rule identifiers>
  - FORWARD
  - BACKWARD <list of facts to be validated>
  - EXPLORE-MODULES <list of module identifiers>
  - STOP-MODULE ;;; stop the exploration of the module
  - HALT ;;; stop the execution of the system
Meta-rules: examples

(MR-PRIMSET01
  IF (OK-BOMB) 1.0
  THEN (DEACTIVATE-RULES RDECP005 RDECP006 RDECP007 RDECP008 RDECP009 RDECP019 RDECP020))

(MR-ESTR01
  IF (CLASS1) POSSIBLE
  THEN (EXPLORE-MODULES C1))

(MR-03024
  IF (SIDA) POSSIBLE
  THEN (EXPLORE-MODULES BACTERIANA-ATÍPICA PNEUMOCISTIS-CARINI TBC CITOMEGALOVIRUS CRIPTOCOC NOCARDIA ASPERGILLUS PNEUMOCOC ENTEROBACTÈRIES))

(MR-02012
  IF (AGE < 14) SURE
  THEN (HALT))
General Cycle of an Inference Engine with Meta-Rules

1. **Detection**: Obtaining the set of applicable *rules*
   - Rule Conflict Set Formation

2. **Selection**: Selection of the *rule* to be applied / fired
   - Rule Conflict Set Resolution

3. **Application**: Application/Firing of the selected *rule*
   - Rule Inference

4. **Detection**: Obtaining the set of applicable *meta-rules*
   - Meta-rule Conflict Set Formation

5. **Selection**: Selection of the *meta-rule* to be applied / fired
   - Meta-rule Conflict Set Resolution

6. **Application**: Application/Firing of the selected *meta-rule*
   - Meta-rule Inference
Hybrid Reasoning

- **Problem solving in a hybrid way → Bidirectional Search**

- The change of the strategy is done through *meta-rules*

- **Advantages:** Improves the performance of *backward reasoning* and the combinatorial explosion of *forward reasoning*.
  - Avoiding impasses
  - When there are no clear goals
Hybrid Reasoning: Example 1a (1)

Knowledge Base

R1: \( F \land G \rightarrow A \)
R2: \( A \land B \land F \rightarrow K \)
R3: \( D \land E \rightarrow B \)
R4: \( I \rightarrow A \)
R5: \( C \rightarrow A \)
R6: \( J \rightarrow D \)
R7: \( I \land D \land J \rightarrow F \)
R8: \( H \rightarrow I \)
R9: \( H \rightarrow E \)
R10: \( F \land G \land E \rightarrow M \)
R11: \( M \land J \land B \rightarrow L \)

Goals

K??
L??

Conflict resolution strategy
Most specific rule

Meta-knowledge
MR1: \( A \land \text{Forward?} \rightarrow \text{Backward (K)} \)
MR2: \( B \land \text{Forward?} \rightarrow \text{Backward (K)} \)
MR3: \( F \land \text{Forward?} \rightarrow \text{Backward (K)} \)

Facts Base

C, J, H, G

© Miquel Sànchez i Marrè, KEMLG-IDEAI, 2020
Hybrid Reasoning: Example 1a (2)

Forward Reasoning without using explicit meta-knowledge

There are no rules with 3 or 2 premises to be applied

(1) A (R5)  
(2) D (R6)  
(3) I (R8)  
(4) F (R7)  
(5) A (R1) ✓  
(6) A (R4) ✓  
(7) E (R9)  
(8) M (R10)  
(9) B (R3)  
(10) K (R2) !!  
(11) L (R11) !!
Hybrid Reasoning: Example 1b (1)

Knowledge Base

R1: \( F \land G \rightarrow A \)
R2: \( A \land B \land F \rightarrow K \)
R3: \( D \land E \rightarrow B \)
R4: \( I \rightarrow A \)
R5: \( C \rightarrow A \)
R6: \( J \rightarrow D \)
R7: \( I \land D \land J \rightarrow F \)
R8: \( H \rightarrow I \)
R9: \( H \rightarrow E \)
R10: \( F \land G \land E \rightarrow M \)
R11: \( M \land J \land B \rightarrow L \)

Goals

K??
L??

Conflict resolution strategy
Most specific rule

Meta-knowledge
MR1: \( A \land \text{Forward?} \rightarrow \text{Backward (K)} \)
MR2: \( B \land \text{Forward?} \rightarrow \text{Backward (K)} \)
MR3: \( F \land \text{Forward?} \rightarrow \text{Backward (K)} \)

Facts Base

C, J, H, G
Hybrid Reasoning: Example 1b (2)

Hybrid Reasoning using meta-knowledge

(1) A (R5) → meta-rule MR1 → backward (K)

K

A yes

D → J yes

B

E → H yes

F *

I

H yes

F

D yes

J yes

K !!
User Interface

- User interaction through NL
- Functionalities
  - Introduce the problem data
  - Ask questions to the user
    - About facts
    - Requiring confirmations
  - Ask questions to the system
    - About the problem solving (Why?)
    - About assumptions (What if?)
    - About the state of the Facts Base
Explanation Module

- Credibility of the system
- Explanations / justifications en les regles/meta-regles
- Typical functionalities
  - Why? --> Goals the system wants to solve
  - How? --> Chain of reasoning until the current time
- Two layers of explanation
  - Show/Display --> Trace of the reasoning (rules and facts)
  - Justification --> Reasons for the reasoning strategies, for the goals, why certain questions are asked, etc.
- Explanations
  - Prefixed/precomputed text
  - NL generation depending on the context
Knowledge Acquisition Module

- This module allows the experts to supply their knowledge to the system.
- The experts can define the Fact Base:
  - The dictionary of the system can be defined, specifying the static properties of the facts.
  - Some dynamic property can also be defined.
  - Updating and removal of facts information can be done too.
- In addition, the Knowledge Base can be defined:
  - The experts can supply the different rules which code the expert knowledge.
  - The experts can define the different modules of the Knowledge Base.
- Some other functionalities like a consistency analysis of the KB can be available.
Knowledge Engineer Interface

- The Knowledge Engineer Interface (KEI) allows the Knowledge Engineer to have access to several modules and components of the system to make a fine tuning of it.

- The KEI allows the knowledge engineer to access to the knowledge acquisition module to make some maintenance of the Fact Base and/or the Knowledge base.

- The KEI also lets the knowledge engineer to access to the inference engine module, for modifying the conflict resolution strategies, etc.

- The KEI provides access to the meta-reasoning module, to define/update the meta-rules, the strategies, etc.
Advantages of Rule-Based Systems

- Adequate in ill-structured domains
- Efficient in diagnosis and classification tasks
- Auto-explanation ability
- Easiness for user communication
- Extensions are easily constructed (approximate reasoning)
Limitations of Rule-Based Systems

- Fragility
- Difficulty with controlling the reasoning
- Low reusability of Knowledge Bases
- Unable to learn
- Knowledge Acquisition problem
- Validation problem
History of Expert Systems (1)

- Oldest ones ≈ 1965
- DENDRAL (1965-1970)
  - Mass spectrograph and magnetic resonance of organic molecules
- META-DENDRAL (1970)
  - Heuristic rules construction from data
- MACSYMA
  - Manipulation of algebraic formula
History of Expert Systems (2)

- MYCIN (1972-1976)
  - Diagnosis of infectious illness in blood
  - 400 rules
  - Reasoning with uncertainty

- EMYCIN (1980)
  - Exports MYCIN’s control system
  - First Expert System Environment (shell)

- HEARSAY-II (1975)
  - Natural Language Interpretation (hear + say)
  - 1000 words
History of Expert Systems (3)

- PROSPECTOR (1977)
  - Mining Exploration
  - Another uncertainty reasoning model

- R1/XCON (1980)
  - Computer System’s Configuration
  - DEC, ≈ 200,000 rules

- INTERNIST (1982)
  - Internal Medicine Diagnosis
  - 500,000 - 1,000,000 rules
History of Expert Systems (4)

- CENTAUR (1983)
  - Diagnosis of pulmonary diseases
  - Rules and prototypes

- MOLE (1986)
  - Expert System classification shell

- TEST (1987)
  - Troubleshooting Expert System Tool
  - Diagnosis / classification

- VT (1988)
  - Vertical Transportation
  - Elevator System design
Intelligent Data Science and Artificial Intelligence (IDEAI-UPC)

Miquel Sànchez-Marrè
miquel@cs.upc.edu

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA

https://kemlg.upc.edu