Introduction to Artificial Intelligence

Chapter 3: Knowledge Representation and Reasoning (2) Propositional Logic

Nguyễn Hải Minh, Ph.D
nhminh@fit.hcmus.edu.vn

Outline

- ❑Syntax
- ❑Semantics
- ❑A simple knowledge base

- ❑Logical Inference Problem
  - o Model-checking Approach
  - o Inference Rules Approach

- ❑CNF Form

Propositional logic: Syntax

- Propositional logic is the simplest logic – illustrates basic ideas
- Constants: TRUE or FALSE
- Symbols to stand for propositions (sentences): P, Q, R, P_1, $W_{1,3}$, …
- Logical connectives:
  - NOT $\neg$ Negation
  - AND $\wedge$ Conjunction
  - OR $\vee$ Disjunction
  - IMPLIES $\Rightarrow$ Implication (if... then)
  - Iff $\Leftrightarrow$ Equivalence, biconditional (if and only if)
- Literal: an atomic sentence (P) or negated atomic sentence ($\neg$P)

Backus-Naur Form (BNF) Grammar

$$
\begin{array}{rcl}
Sentence & \rightarrow & AtomicSentence \mid ComplexSentence \\
AtomicSentence & \rightarrow & True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence & \rightarrow & (\ Sentence\ ) \mid [\ Sentence\ ] \\
& \mid & \neg\ Sentence \\
& \mid & Sentence \land Sentence \\
& \mid & Sentence \lor Sentence \\
& \mid & Sentence \Rightarrow Sentence \\
& \mid & Sentence \Leftrightarrow Sentence \\
\\
\text{OPERATOR PRECEDENCE} & : & \neg, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

BNF – a formal grammar of propositional logic

Propositional logic: Semantics

- Each model specifies true/false for each proposition symbol
  - E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$
    false true false
- With these symbols, 8 possible models can be enumerated automatically.

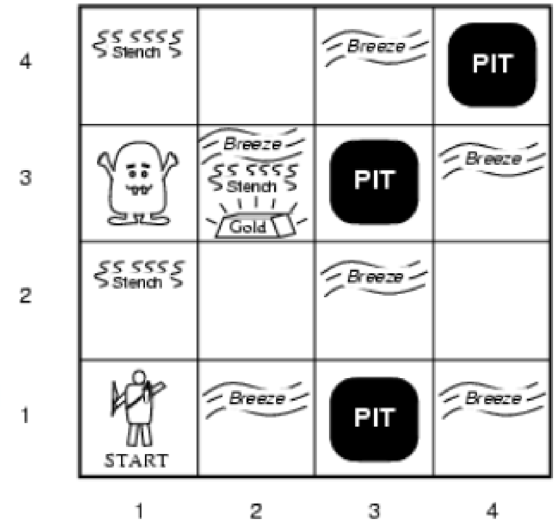
- Rules for evaluating truth with respect to a model *m*:
  - $\neg S$ is true iff S is false
  - $S_1 \wedge S_2$ is true iff S_1 is true and S_2 is true
  - $S_1 \vee S_2$ is true iff S_1 is true or S_2 is true
  - $S_1 \Rightarrow S_2$ is true iff S_1 is false or S_2 is true
  - i.e., is false iff S_1 is true and S_2 is false
  - $S_1 \Leftrightarrow S_2$ is true iff $S_1 \Rightarrow S_2$ is true and $S_2 \Rightarrow S_1$ is true

- Simple recursive process evaluates an arbitrary sentence,
  - e.g., $\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = true \wedge (true \vee false) = true \wedge true = true$

Truth tables for connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

A simple knowledge base: Wumpus world



- Symbols for each position $[i, j]$
  - $P_{i,j}$ is true if there is a pit in $[i, j]$
  - $W_{i,j}$ is true if there is a wumpus in $[i, j]$
  - $B_{i,j}$ is true if there is a breeze in $[i, j]$
  - $S_{i,j}$ is true if there is a stench in $[i, j]$
- Sentences in Wumpus world's KB:
  - R_1: $\neg P_{1,1}$
  - R_2: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
  - R_3: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
  - R_4: $\neg B_{1,1}$
  - R_5: $B_{2,1}$

Logical Inference Problem

- Given:
  - KB: A set of sentences
  - A sentence α
- Goal: answer the question: does the KB semantically entail α?
  - That is, KB |= α
- In other words:
  - In all interpretations in which sentences in KB are true, is α also true?

Solving the Logical Inference Problem

- Example:
  - Given KB in Wumpus World, decide if there is a pit in [1,2] or not:
    - KB $|= P_{1,2}$?

- 3 approaches:
  - Model-checking (by enumeration)
  - Inference Rules
  - Conversion to the inverse SAT problem (Resolution refutation)

1. Model-checking approach

- Other name:
  - Inference by enumeration
- Check if α is true in every model in which KB is true.
  - E.g, Wumpus's KB: 7 symbols → $2^7 = 128$ models
  - Draw a truth table for checking

Truth table for the KB of Wumpus World

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

A truth table constructed for the KB of Wumpus World

No pit in [1,2]

Inference by enumeration

- Depth-first enumeration of all models is sound and complete

```
function TT-ENTAILS?(KB, α) returns true or false
   inputs: KB, the knowledge base, a sentence in propositional logic
           α, the query, a sentence in propositional logic

   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, { })
```

```
function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
      if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
      else return true // when KB is false, always return true
   else do
      P ← FIRST(symbols)
      rest ← REST(symbols)
      return (TT-CHECK-ALL(KB, α, rest, model ∪ {P = true})
             and
             TT-CHECK-ALL(KB, α, rest, model ∪ {P = false }))
```


- For *n* symbols, time complexity is *O(2^n)*, space complexity is *O(n)*

2. Inference Rules Approach

- Other name:
  - Theorem proving
- Applying rules of inference directly to the sentences in KB to construct a proof of the desired sentence without consulting models
  - → Efficient than model checking if the number of models is large but length of proof is short
- New concepts:
  - Logical equivalence
  - Validity
  - Satisfiability

Logical equivalence

- Two sentences are logically equivalent iff true in same models: $\alpha \equiv ß$ iff $\alpha \models \beta$ and $\beta \models \alpha$

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Validity and satisfiability

- A sentence is valid if it is true in all models,
  - e.g., *True*, $A \lor \neg A$, $A \Rightarrow A$, $(A \land (A \Rightarrow B)) \Rightarrow B$
- Validity is connected to inference via the Deduction Theorem:
  - $KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid
- A sentence is satisfiable if it is true in some model
  - e.g., $A \lor B$, C
- A sentence is unsatisfiable if it is true in no models
  - e.g., $A \land \neg A$
- Satisfiability is connected to inference via the following:
  - $KB \models \alpha$ if and only if $(KB \land \neg \alpha)$ is unsatisfiable

Validity and satisfiability

- Validity and satisfiability are connected:
  - α is valid iff $\neg\alpha$ is unsatisfiable;
  - α is satisfiable iff $\neg\alpha$ is not valid.

- Result:
  - $\alpha \models \beta$ *if and only if the sentence* $(\alpha \land \neg\beta)$ *is unsatisfiable.*

Excercise

- Check the validity and satisfiability of the following sentence using the Truth table
  1. $A \wedge B \Rightarrow A \vee C$
  2. $A \vee B \Rightarrow A \wedge C$
  3. $(A \vee B) \wedge (\neg B \vee C) \Rightarrow A \vee C$
  4. $(A \vee \neg B) \Rightarrow A \wedge B$

Apply Inference Rules to derive a Proof

- Proof:
  - A chain of conclusions leads to the desired goal
- Example sound rules of inference:

RULE	PREMISE	CONCLUSION
Modus Ponens	$A, A \rightarrow B$	B
AND Introduction	A, B	$A \wedge B$
AND Elimination	$A \wedge B$	A
OR Introduction	A	$A \vee X$
Double Negation	$\neg\neg A$	A
Unit Resolution	$A \vee B, \neg B$	A
Resolution	$A \vee B, \neg B \vee C$	$A \vee C$

Inference Rules in Wumpus World

- ❑KB: $R_1 \rightarrow R_5$
- ❑Proof: $\neg P_{1,2}$
- ❑Apply inference rules:
  - Bi-conditional elimination to R_2:
    - R_6: $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
  - And-Elimination to R_6:
    - R_7: $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
  - Logical equivalence for contrapositives
    - R_8: $(\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$
  - Modus Ponens with R_8 and the percept R_4
    - $R_9 : \neg(P_{1,2} \vee P_{2,1})$
  - De Morgan's rule:
    - $R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$

Searching for Proof
→ Can apply
Searching Algorithms

finding a proof can be more efficient because the proof can ignore irrelevant propositions, no matter how many of them there are.

Proof by Resolution Inference Rule

- Problem of Proof by Inference Rules:
  - If the rules are inadequate, then the goal is not reachable → the algorithm is not complete
- Resolution Rule:
  - A single inference rule

$$\alpha \lor \beta, \neg \beta \lor \gamma \mid- \alpha \lor \gamma$$

  - Or: $\neg\alpha \Rightarrow \beta, \beta \Rightarrow \gamma \mid- \neg \alpha \Rightarrow \gamma$
  - Yields complete inference algorithm when coupled with any complete search algorithm

Soundness of Resolution Rule

α	β	γ	α ∨ β	¬ β ∨ γ	α ∨ γ
F	F	F	F	T	F
F	F	T	F	T	T
F	T	F	T	F	F
F	T	T	T	T	T
T	F	F	T	T	T
T	F	T	T	T	T
T	T	F	T	F	T
T	T	T	T	T	T

We highlighted the cases when both premises are true
The resolution rule is sound because the conclusions are true in all cases (here 4) where the premises are true

Resolution in Wumpus World

- KB:
  - $R_1 \rightarrow R_{10}$
  - $R_{11} : \neg B_{1,2}$
  - $R_{12} : B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$
- Proof by inference rules:
  - $R_{13} : \neg P_{2,2}$
  - $R_{14} : \neg P_{1,3}$
  - $R_{15} : P_{1,1} \vee P_{2,2} \vee P_{3,1}$

Resolution in Wumpus World

- KB:
  - $R_1 \rightarrow R_{10}$
  - $R_{11} : \neg B_{1,2}$
  - $R_{12} : B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$
- Proof by inference rules:
  - $R_{13} : \neg P_{2,2}$
  - $R_{14} : \neg P_{1,3}$
  - $R_{15} : P_{1,1} \vee P_{2,2} \vee P_{3,1}$

Resolves 2 complemantary literals

Resolvent:
$R_{16}: P_{1,1} \vee P_{3,1}$

Resolution in Wumpus World

- KB:
  - $R_1 \rightarrow R_{10}$
  - $R_{11} : \neg B_{1,2}$
  - $R_{12} : B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$
- Proof by inference rules:
  - R_1: $\neg P_{1,1}$
  - R_{16}: $P_{1,1} \vee P_{3,1}$

Resolves 2 complemantary literals

Resolvent:
R_{17}: $P_{3,1}$

→ *R_{16} & R_{17} are examples of the Unit resolution inference rule*

Conjunctive Normal Form (CNF)

- Resolution rule applies only to clauses (disjunctions of literals)

→ Need to convert all sentences in KB into clauses (CNF form)

- Example: convert $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ into CNF

$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

→ A conjunction of 3 clauses

Conversion to CNF

1. Remove implication and equivalence
   - $(P \Rightarrow Q)$ becomes $(\neg P \lor Q)$
   - $(P \Leftrightarrow Q)$ becomes $(P \Rightarrow Q) \land (Q \Rightarrow P)$, then becomes $(\neg P \lor Q) \land (\neg Q \lor P)$

2. Move negations inwards – Use De Morgan's
   - $\neg(P \land Q)$ becomes $(\neg P \lor \neg Q)$
   - $\neg(P \lor Q)$ becomes $(\neg P \land \neg Q)$
3. Distribute OR over AND
   - $P \lor (Q \land R)$ becomes $(P \lor Q) \land (P \lor R)$

Excercise

- Convert the following sentences into CNF:
  1. $(A \wedge B) \Rightarrow (C \Rightarrow D)$
  2. $P \vee Q \Leftrightarrow R \wedge \neg Q \Rightarrow P$

Resolution Algorithm

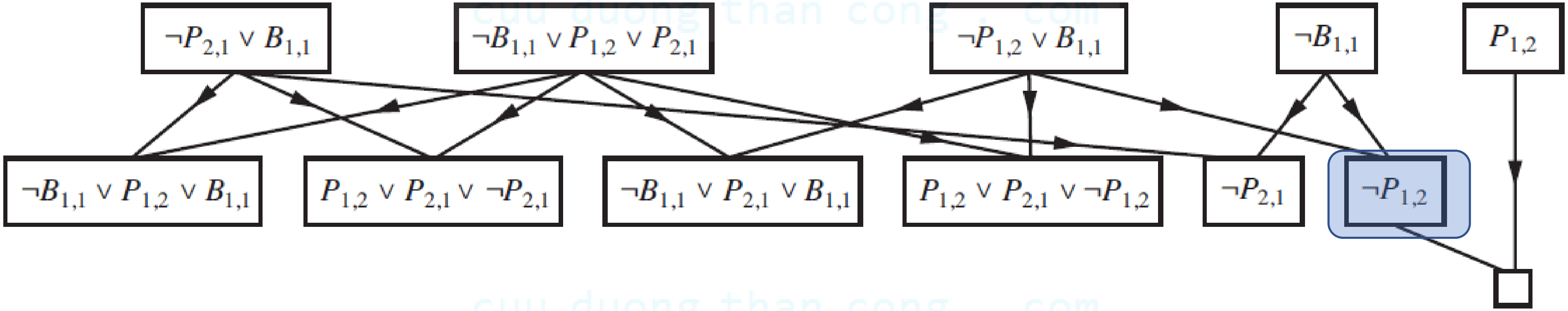
- Proof by contradiction, i.e., show $KB \land \neg \alpha$ unsatisfiable

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
  inputs: KB, the knowledge base, a sentence in propositional logic
    α, the query, a sentence in propositional logic

  $clauses \leftarrow$ the set of clauses in the CNF representation of $KB \land \neg\alpha$
  $new \leftarrow \{\ \}$
  loop do
    for each pair of clauses C_i, C_j **in** $clauses$ **do**
      $resolvents \leftarrow$ PL-RESOLVE(C_i, C_j)
      if $resolvents$ contains the empty clause **then return** *true*
      $new \leftarrow new \cup resolvents$
    if $new \subseteq clauses$ **then return** *false*
    $clauses \leftarrow clauses \cup new$

Resolution Algorithm – Example

- **Wumpus World:**
  - $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$
  - $\alpha = \neg P_{1,2}$



Excercise

- Given KB:
  1. $A \Rightarrow B$
  2. $A \Rightarrow C \vee D$
  3. $B \wedge D \Rightarrow E$
  4. $A \wedge B \Rightarrow F$
  5. A
- Check if the following sentences are entailed by KB?
  - F?
  - E?

Problem of Inference Rules

- Too many propositions to handle
  - The statement “Do not go forward if the Wumpus is in front of you” requires 16 squares x 4 orientations = 64 propositional rules
  - It will take thousands of rules to build an agent
- Change of the KB over time is difficult to represent
  - Standard technique is to index facts with the time when they’re true
  - This means we have a separate KB for every time point

Next week

- Chapter 3: Knowledge Representation and Reasoning (cont.)
  - Propositional Logic: Horn Forms