22년 삼성 AI 전문가과정
6월 7일 화요일 4교시
장병탁

인공지능

5차시 : Knowledge-Based Agents

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang

Lecture Overview

인공지능

5차시 : Knowledge-Based Agents

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang

Introduction

- **Problem-solving (Reflex) agents** (Previous lecture)
  - Knowledge is limited and inflexible, hidden in transition model $Result(s, a)$
  - Atomic representation of states is limited
- **Knowledge-based agents** (This lecture)
  - Represent knowledge about the world
  - Deduce the actions to take from the knowledge
- **Representing knowledge using logic**
  - Propositional logic (PL)
  - First-order logic (FOL)
- **First-order logic (FOL)**
  - More expressive than PL

Knowledge-Based Agents

A simple knowledge-based agent

```
function KB-AGENT(percept) returns an action
   persistent: KB, a knowledge base
               t, a counter, initially 0, indicating time

   TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
   action ← ASK(KB, MAKE-ACTION-QUERY(t))
   TELL(KB, MAKE-ACTION-SENTENCE(action, t))
   t ← t + 1
   return action
```

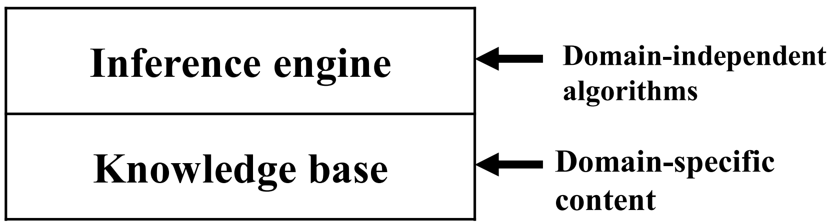


Figure 7.1 A generic knowledge-based agent. Given a percept, the agent adds the percept to its knowledge base, asks the knowledge base for the best action, and tells the knowledge base that it has in fact taken that action.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

Representing Knowledge in Logic

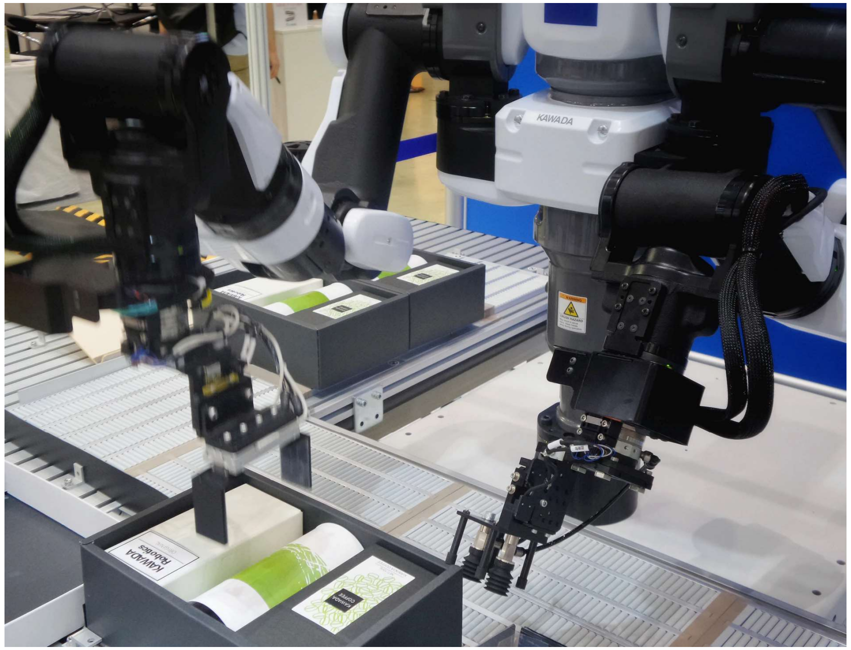


사진 출처 #1

Battery

Liftable

Movable

Knowledge Base (Rules)

$B \wedge L \Rightarrow M$

Sensors (Facts)

B

L

Query (Goal)

M?

Kinds of Logic

Logics in general (formal languages)

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
• Propositional logic	• facts	• true/false/unknown
• First-order logic	• facts, objects, relations	• true/false/unknown
• Temporal logic	• facts, objects, relations, times	• true/false/unknown
• Probability theory	• facts	• degree of belief
• Fuzzy logic	• facts + degree of truth	• known interval value

Propositional Logic & Inference

Inference rules

- **Modus ponens**

  $\alpha \Rightarrow \beta, \ \ \alpha. \ \ \vDash \ \ \beta$

  $(\alpha_1, \dots, \alpha_n), (\alpha_1 \land \cdots \land \alpha_n \Rightarrow \beta) \ \ \vDash \ \ \beta$

- **And-elimination**

  $\alpha \land \beta \ \ \vDash \ \ \alpha$

- **Resolution**

  $(\alpha_1, \dots, \alpha_j, \dots, \alpha_n), \ \ (\alpha_1, \dots, \neg\alpha_j, \dots, \alpha_n)$

  $\vDash \ \ (\alpha_1, \dots, \alpha_{j-1}, \alpha_{j+1}, \dots, \alpha_n)$

Light => Lecture
Light
➔ *Lecture*

Light => Lecture
➔ ¬ *Light* ∨ *Lecture*

¬ *Light* ∨ *Lecture*
Light
➔ *Lecture*

❑ Family relationships: Sibling

- A sibling is another child of one's parents

$$\forall x, y\ Sibling(x, y) \Leftrightarrow x \neq y \land \exists p\ Parent(p, x) \land Parent(p, y)$$

- Brothers are siblings

$$\forall x, y\ Brother(x, y) \Rightarrow Sibling(x, y)$$

- Sibling is symmetric

$$\forall x, y\ Sibling(x, y) \Leftrightarrow Sibling(y, x)$$

- A first cousin is a child of a parent's sibling

$$\forall x, y\ FirstCousin(x, y) \Leftrightarrow \exists p, ps\ Parent(p, x) \land Sibling(ps, x) \land Parent(ps, y)$$

Lecture 5. Knowledge-Based Agents

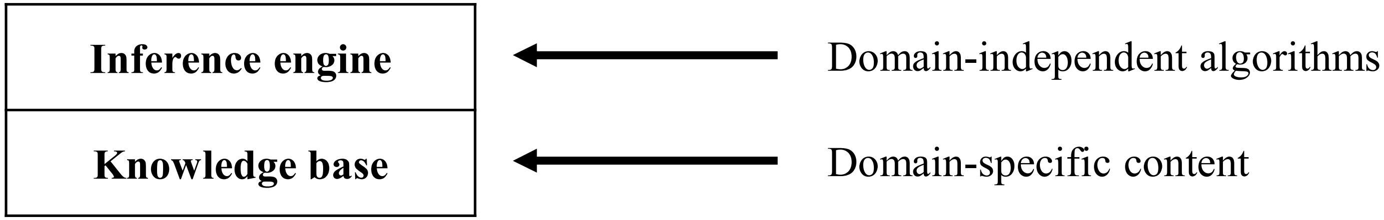
- Knowledge-based Agents
  - What is knowledge?
  - Logical Agents: Knowledge in logic
- Propositional Logic
  - Syntax and semantics
- Propositional Inference
  - Forward and backward chaining
  - Resolution inference
- First-Order Logic
  - Syntax and semantics
  - Using first-order logic

Outline (Lecture 5)

Stuart Russell & Peter Norvig (2021), Artificial Intelligence: A Modern Approach (4th Edition)

5.1. Knowledge-Based Agents

5.1 Knowledge-Based Agents (1/2)



- **Knowledge base = A set of sentences in a formal language**
- Declarative approach to building an agent (or other system):
  - **Tell** it what it needs to known
- Then it **Ask** itself what to do – answers should follow from knowledge base
- Agents can be viewed at the knowledge level: what they know or at the implementation level
  - i.e. data structures and algorithms in knowledge base

5.1 Knowledge-Based Agents (2/2)

A simple knowledge-based agent

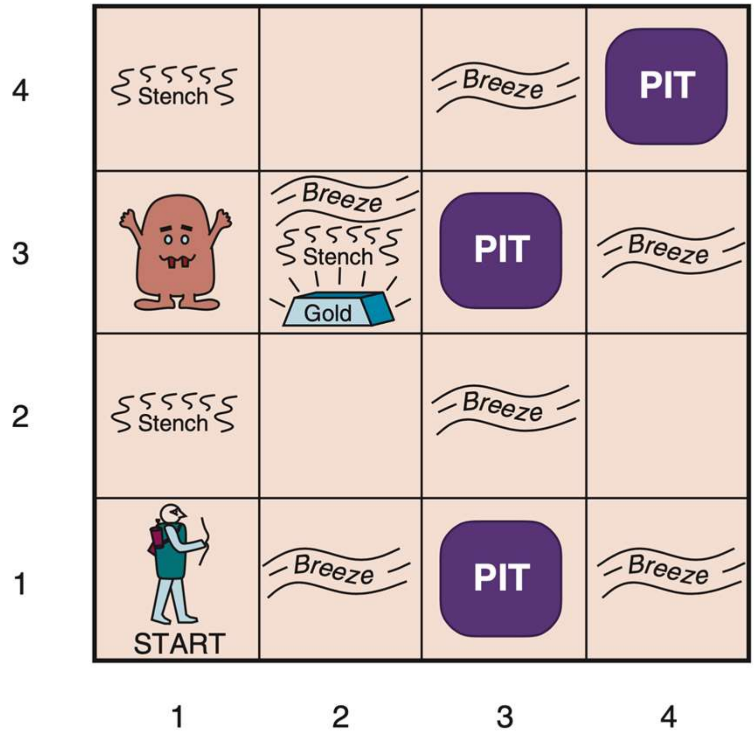
```
function KB-AGENT(percept) returns an action
   persistent: KB, a knowledge base
               t, a counter, initially 0, indicating time

   TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
   action ← ASK(KB, MAKE-ACTION-QUERY(t))
   TELL(KB, MAKE-ACTION-SENTENCE(action, t))
   t ← t + 1
   return action
```

Figure 7.1 A generic knowledge-based agent. Given a percept, the agent adds the percept to its knowledge base, asks the knowledge base for the best action, and tells the knowledge base that it has in fact taken that action.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

The Wumpus World



Sensors	Actuators	Knowledge Base
Stench		
Breeze		
Glitter	Forward	None → OK
Bump	TurnLeft	OK → MoveForward
Scream	TurnRight	Breeze → Pit
OK	Grab	Pit → TurnLeft
Pit	Shoot	Glitter → Gold
Visited	Climb	Gold → Grab
Wumpus		
None		

$B_{2,1}$: It breezes at (2, 1)

$P_{3,1}$: There is a pit at (3,3)

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.2 Propositional Logic

5.2 Propositional Logic (1/5)

Logic in general

- Logics are formal languages for representing information such that conclusions can be drawn.
- Syntax defines the sentences in the language.
- Semantics define the "meaning" of sentences.
  - i.e. Define truth of a sentence in a world.

Entailment

- Entailment means that one thing follows from another: $KB \vDash \alpha$
- Knowledge base KB entails sentence α iff α is true in all models where KB is true.

5.2 Propositional Logic (3/5)

Syntax

- **Proposition**: A statement which can be evaluated as true or false
  (but not both or neither)
  - Proposition symbols: $Liftable, Movable, L, M, P_1, P_2, S_1, S_2$
  - **Literals**: S (positive literal), $\neg S$ (negative literal)
- **(Propositional) sentences:** The proposition symbols P_1, P_2 are sentences
- If S is a sentence, $\neg S$ is a sentence (negation)
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (disjunction)
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction)
- If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (implication)
- If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (biconditional)

5.2 Propositional Logic (4/5)

Semantics

- Each model specifies true/false for each proposition symbol
  - i.e. $P_{1,2}: true, P_{2,2}: true, P_{3,1}: false$
- (With these symbols, 8 possible models, can be enumerated)

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*

Models of $P \vee Q$

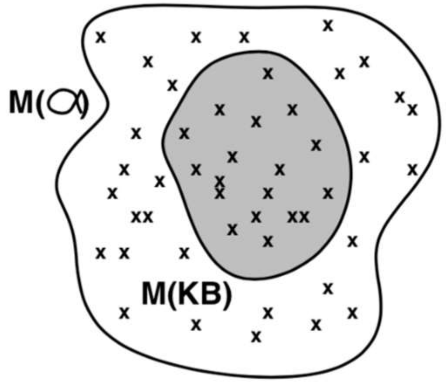
- *P=false, Q=true*
- *P=true, Q=false*
- *P=true, Q=true*

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.2 Propositional Logic (2/5)

Models

- Models are formally structured worlds with respect to which truth can be evaluated.
- m is a model of a sentence α if α is true in m.
  - model m := interpretation or assignment θ that makes α true
- $M(\alpha)$ is the set of all models of α.
  - m = *(P=false, Q=true)* is a model of α = *"$P \lor Q$"*
  - M(*"$P \lor Q$"*) = { (*P=false, Q=true*), (*P=true, Q=false*), (*P=true, Q=true*) }
- Then $KB \vDash \alpha$ iff $M(KB) \subseteq M(\alpha)$



Inference

- $KB \vdash_i \alpha$ = sentence α can be derived from KB by procedure i
- **Soundness**: i is sound if, whenever $KB \vdash_i \alpha$, also true that $KB \vDash \alpha$
- **Completeness**: i is complete if, whenever $KB \vDash \alpha$, also true that $KB \vdash_i \alpha$

5.2 Propositional Logic (5/5)

Truth tables for inference

$KB \vDash \alpha$

- Enumerate rows (different assignments to symbols): if KB is true in row, check that α is too.
- **Inference by enumeration** (or **model checking**): Enumeration of all models sound and complete

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	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*

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

명제 논리

5.3 Propositional Theorem Proving

5.3 Propositional Theorem Proving (1/10)

So far, we have shown how to determine entailment by model checking (or inference by enumeration): enumerating models and showing that the sentences must hold in all models. Here we show how entailment can be done by theorem proving.

Theorem Proving

- **Logical Equivalence**
  - Sentences α and β are logically equivalent if they are true in the same set of models.
- **Validity**
  - A sentence is valid if it is true in all models
  - e.g. $True,\ A \vee \neg A,\ A \Rightarrow A,\ \big(A \wedge (A \Rightarrow B)\big) \Rightarrow B$
- **Satisfiability**
  - A sentence is satisfiable if it is true in some model
  - A sentence is unsatisfiable if it is true in no models: e.g. $A \wedge \neg A$

5.3 Propositional Theorem Proving (2/10)

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

Figure 7.11 Standard logical equivalences. The symbols α, β, and γ stand for arbitrary sentences of propositional logic.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.3 Propositional Theorem Proving (3/10)

Inference rules

- **Modus Ponens**
  - $\dfrac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$

    whenever any sentences of the form $\alpha \Rightarrow \beta$ and α are given, then the sentence β can be inferred.

Light => Lecture
Light
➔ *Lecture*

- **And-Elimination**
  - $\dfrac{\alpha \wedge \beta}{\alpha}$

Light ∧ *Loud*
➔ *Light*

Light => *Lecture*
➔(in CNF form)
¬ *Light* ∨ *Lecture*

¬ *Light* ∨ *Lecture*
Light
➔ *Lecture*

- **Resolution**
  - $(\alpha_1, \ldots, \alpha_j, \ldots, \alpha_n),\ \ (\alpha_1, \ldots, \neg\alpha_j, \ldots, \alpha_n) \vDash (\alpha_1, \ldots, \alpha_{j-1}, \alpha_{j+1}, \ldots, \alpha_n)$

5.3 Propositional Theorem Proving (4/10)

Horn Clauses, Definite Clauses, and Conjunctive Normal Form (CNF)

$$
\begin{aligned}
\mathit{CNFSentence} &\rightarrow \mathit{Clause}_1 \wedge \cdots \wedge \mathit{Clause}_n \\
\mathit{Clause} &\rightarrow \mathit{Literal}_1 \vee \cdots \vee \mathit{Literal}_m \\
\mathit{Fact} &\rightarrow \mathit{Symbol} \\
\mathit{Literal} &\rightarrow \mathit{Symbol} \mid \neg \mathit{Symbol} \\
\mathit{Symbol} &\rightarrow P \mid Q \mid R \mid \ldots \\
\mathit{HornClauseForm} &\rightarrow \mathit{DefiniteClauseForm} \mid \mathit{GoalClauseForm} \\
\mathit{DefiniteClauseForm} &\rightarrow \mathit{Fact} \mid (\mathit{Symbol}_1 \wedge \cdots \wedge \mathit{Symbol}_l) \Rightarrow \mathit{Symbol} \\
\mathit{GoalClauseForm} &\rightarrow (\mathit{Symbol}_1 \wedge \cdots \wedge \mathit{Symbol}_l) \Rightarrow \mathit{False}
\end{aligned}
$$

Figure 7.12 A grammar for conjunctive normal form, Horn clauses, and definite clauses. A CNF clause such as $\neg A \vee \neg B \vee C$ can be written in definite clause form as $A \wedge B \Rightarrow C$.

Horn clause: Disjunctions of max one positive literal

A

$\neg A \vee B$

$\neg A \vee \neg B \vee C$ [$\Leftrightarrow A \wedge B \Rightarrow C$]

Conjunctive normal form (CNF): conjunction of clauses

$(\ldots \vee \ldots) \wedge (\ldots \vee \ldots) \wedge (\ldots \vee \ldots)$

KB is a conjunction of clauses (CNF)

$C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.3 Propositional Theorem Proving (5/10)

Proof methods

- **Forward chaining**
  - LHS to RHS
  - Data-driven
- **Backward chaining**
  - RHS to LHS
  - Goal-driven
- **Resolution refutation (theorem proving)**
  - $KB \vDash \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable
  - Prove α by *reduction ad absurdum*
  - Proof by refutation or proof by contradiction

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.3 Propositional Theorem Proving (6/10)

Example

Is Q True?

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

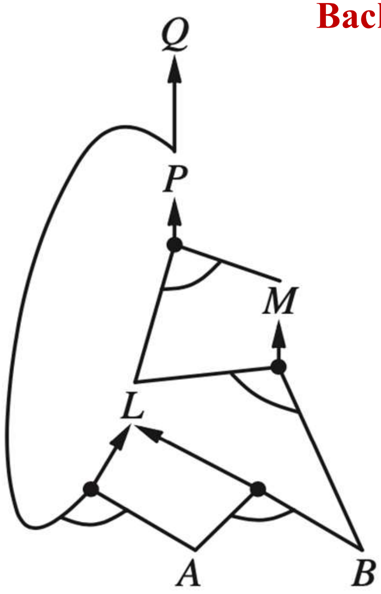
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

(a)



(b)

Backward chaining

Forward chaining

deduction vs induction (ML)

Figure 7.16 (a) A set of Horn clauses. (b) The corresponding AND–OR graph.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.3 Propositional Theorem Proving (7/10)

Forward and backward chaining

- **Forward chaining (FC)**
  - Fire any rule whose premises are satisfied in the KB, add its conclusion to the KB, until query is found.
  - FC is data-driven, cf. automatic, unconscious processing.
  - May do lots of work that is irrelevant to the goal.
  - FC derives every atomic sentence that is entailed by KB.
- **Backward chaining (BC)**
  - Work backwards from the query q
  - Avoid loops: check if new subgoal is already on the goal stack.
  - Avoid repeated work: check if new subgoal.
  - BC is goal-driven, appropriate for problem-solving.

6.5 Propositional Theorem Proving (8/10)

Resolution

Conjunctive Normal Form (CNF—universal)

$$\textbf{conjunction} \text{ of } \underbrace{\textbf{disjunctions} \text{ of } \textbf{literals}}_{\textbf{clauses}}$$

E.g., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

Resolution inference rule (for CNF): complete for propositional logic

$$\frac{\ell_1 \lor \cdots \lor \ell_k, \qquad m_1 \lor \cdots \lor m_n}{\ell_1 \lor \cdots \lor \ell_{i-1} \lor \ell_{i+1} \lor \cdots \lor \ell_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \lor \cdots \lor m_n}$$

where ℓ_i and m_j are complementary literals. E.g.,

$$\frac{P_{1,3} \lor P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Resolution is sound and complete for propositional logic

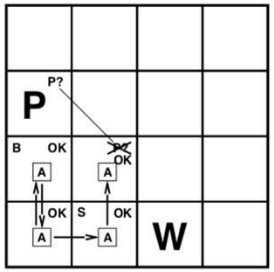


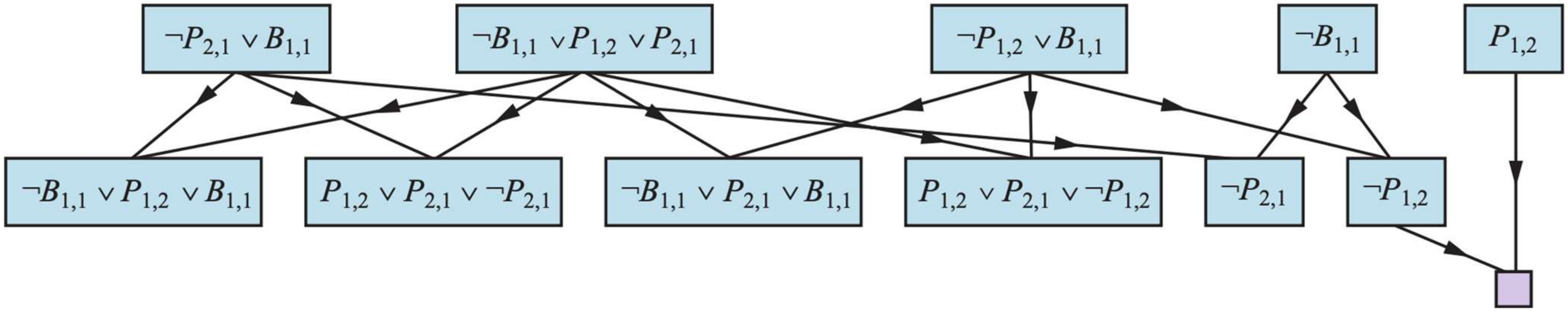
사진 출처 #11

5.3 Propositional Theorem Proving (9/10)

Resolution Refutation: show $KB \wedge \neg\, \alpha$ is unsatisfiable (proof by contradiction)

- $KB = \left(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})\right) \wedge \neg B_{1,1}$ $\quad \neg\alpha = P_{1,2}$
- $Cf.\ B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ ➔ $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Example



<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

5.3 Propositional Theorem Proving (10/10)

Resolution Algorithm for Propositional Logic

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

Figure 7.13 A simple resolution algorithm for propositional logic. PL-RESOLVE returns the set of all possible clauses obtained by resolving its two inputs.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

Agents Based on Propositional Logic

A Hybrid Agent

- Maintains and updates a knowledge base as well as current plan.
- Uses logical inference by asking questions of the knowledge base, to work out which squares are safe, and which have yet to be visited.
- Constructs a plan based on a decreasing priority of goals.
- Route planning is done with A* search.

Logical State Estimation

- The process of updating the belief state as new percepts arrive.
- Use a logical sentence involving proposition symbols associated with the current time step.
- *Approximate* state estimation represent belief states as conjunctions of literals (1-CNF formulas)
- As time goes along, some information may be lost.

Summary (Part I)

1. Knowledge is contained in agents in the form of **sentences** in a **knowledge representation language** that are stored in a **knowledge base.**
2. A representation language is defined by its **syntax**, which specifies the structure of sentences, and its **semantics**, which defines the **truth** of each sentence in each **possible world** or **model.**
3. The relationship of **entailment** between sentences is crucial to our understanding of reasoning.
4. The **propositional logic** is a simple language consisting of proposition symbols and logical connectives, which does not scale to environments of unbounded size.
5. The **inference rules** are patterns of sound inference that can be used to find proofs.
6. The **resolution rule** yields a complete inference algorithm for knowledge bases that are expressed in conjunctive normal form.
7. Forward chaining and backward chaining are very natural reasoning algorithms for knowledge bases in Horn form.
8. The **local search** methods such as WalkSAT can be used to find solutions. Such algorithms are sound but not complete.

1차 술어 논리

5.4 First-Order Logic

5.4 First-Order Logic (1/8)

Propositional logic

- Propositional logic is declarative: pieces of syntax respond to facts
- Propositional logic allows partial/disjunctive/negated information
  - (unlike most data structures and databases)
- Propositional logic is compositional:
  - meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- Meaning in propositional logic is context-independent
  - (unlike natural language, where meaning depends on context)
- Propositional logic has very limited expressive power
  - (unlike natural language)
  - e.g. cannot say pits cause breezes in adjacent squares except by writing one sentence for each square

5.4 First-Order Logic (2/8)

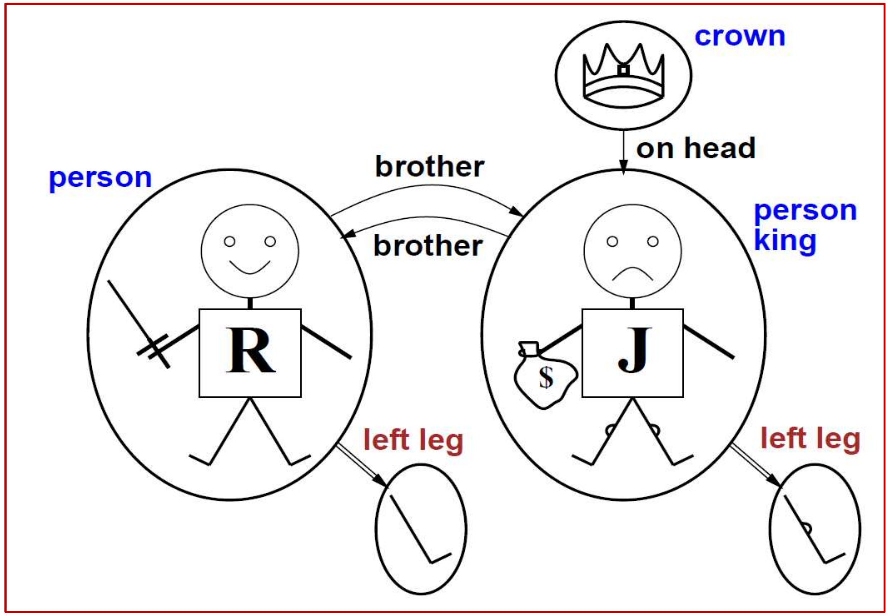
First-order logic

- More expressive than propositional logic
- Whereas propositional logic assumes world contains facts, first-order logic (like natural language) assumes the world contains
  - Objects: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries ...
  - Relations: red, round, bogus, prime, multistoried ..., brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, ...
    - Unary relations or properties
  - Functions: father of, best friend, third inning of, one more than, end of

5.4 First-Order Logic (3/8)

Models for FOL: Example

The models of a logical language are the formal structures that constitute the possible worlds under consideration. Each model links the vocabulary of the logical sentences to elements of the possible world, so that the truth of any sentence can be determined.



Objects:
- *Richard*
- *John*

Relations:
- *Brother*
- *OnHead*
- *Person*
- *King*
- *Crown*

Functions
- *LeftLeg*

사진 출처 #2

5.4 First-Order Logic (4/8)

Syntax and Semantics of FOL

Symbols and interpretations

- Constants *Richard* (interpretation: *“Richard the Lionheart”*)
- Predicates *Brother* (interpretation: brotherhood relation)
- Functions *LeftLeg*
- Variables $x, y, a, b \ldots$
- Connectives $\wedge, \vee, \neg, \Rightarrow, \Leftrightarrow, \ldots$
- Equality $=$
- Quantifiers $\forall, \exists$

5.4 First-Order Logic (5/8)

- **Terms**

  Richard, *LeftLeg*(*Richard*)

  x, $f(x)$

A term is an logical expression that refers to an object.

- **Atomic sentences**

  Brother(*Richard*, *John*)

  Married(*Father*(*Richard*), *Mother*(*John*))

A sentence is a predicate which can be evaluated as true or false.

- **Complex sentences**

  Brother(*Richard*, *John*) $\wedge$ *Brother*(*John*, *Richard*)

  $\neg$*King*(*Richard*) $\Rightarrow$ *King*(*John*)

5.4 First-Order Logic (6/8)

❑ Quantifiers

- **Universal quantifier**

  $\forall x \;\; King(x) \Rightarrow Person(x)$

- **Existential quantifier**

  $\exists x \;\; Crown(x) \land OnHead(x, John)$

Quantifiers express properties of entire collections of objects, instead of enumerating the objects by name.

$\exists x \;\; P$ is true in a model m iff P is true with x being some possible object in the model

❑ Equality

$Father(John) = Henry$

5.4 First-Order Logic (7/8)

$$
\begin{array}{rcl}
\mathit{Sentence} & \rightarrow & \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} & \rightarrow & \mathit{Predicate} \mid \mathit{Predicate}(\mathit{Term}, \ldots) \mid \mathit{Term} = \mathit{Term} \\
\mathit{ComplexSentence} & \rightarrow & (\, \mathit{Sentence} \,) \mid [\, \mathit{Sentence} \,] \\
& \mid & \neg\, \mathit{Sentence} \\
& \mid & \mathit{Sentence} \land \mathit{Sentence} \\
& \mid & \mathit{Sentence} \lor \mathit{Sentence} \\
& \mid & \mathit{Sentence} \;\Rightarrow\; \mathit{Sentence} \\
& \mid & \mathit{Sentence} \;\Leftrightarrow\; \mathit{Sentence} \\
& \mid & \mathit{Quantifier}\ \mathit{Variable}, \ldots\ \mathit{Sentence} \\
\\
\mathit{Term} & \rightarrow & \mathit{Function}(\mathit{Term}, \ldots) \\
& \mid & \mathit{Constant} \\
& \mid & \mathit{Variable} \\
\\
\mathit{Quantifier} & \rightarrow & \forall \mid \exists \\
\mathit{Constant} & \rightarrow & A \mid X_1 \mid \mathit{John} \mid \cdots \\
\mathit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\mathit{Predicate} & \rightarrow & \mathit{True} \mid \mathit{False} \mid \mathit{After} \mid \mathit{Loves} \mid \mathit{Raining} \mid \cdots \\
\mathit{Function} & \rightarrow & \mathit{Mother} \mid \mathit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} & : & \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

The syntax of first-order logic in Backus-Naur form (BNF).

사진 출처 #1

5.4 First-Order Logic (8/8)

❑ Truth in FOL

- Sentences are true with respect to a model and an interpretation
- Model contains ≥ 1 objects (domain elements) and relations among them
- Interpretation specifies referents for
  - Constant symbols → objects
  - Predicate symbols → relations
  - Function symbols → functional relations
- An atomic sentence $predicate(term_1, \dots, term_n)$ is true
  - iff the objects referred to by $term_1, \dots, term_n$ are in the relation referred to by $predicate$

5.5 Using First-Order Logic

5.5 Using First-Order Logic (1/2)

❑ Family relationships

- One's mother is one's female parent

  $\forall m, c \; Mother(c) = m \Leftrightarrow Female(m) \land Parent(m, c)$

- One's husband is one's male spouse

  $\forall w, h \; Husband(w, h) \Leftrightarrow Male(h) \land Spouse(h, w)$

- Male and female are disjoint categories

  $\forall x \; Male(x) \Leftrightarrow \neg Female(x)$

- Parent and child are inverse relations

  $\forall p, c \; Parent(p, c) \Leftrightarrow Child(c, p)$

- A grandparent is a parent of one's parent

  $\forall g, c \; Grandparent(g, c) \Leftrightarrow \exists p \; Parent(g, p) \land Parent(p, c)$

5.5 Using First-Order Logic (2/2)

❑ Family relationships: Sibling

- A sibling is another child of one's parent

$$\forall x, y\ Sibling(x, y) \Leftrightarrow x \neq y \land \exists p\ Parent(p, x) \land Parent(p, y)$$

- Brothers are siblings

$$\forall x, y\ Brother(x, y) \Rightarrow Sibling(x, y)$$

- Sibling is symmetric

$$\forall x, y\ Sibling(x, y) \Leftrightarrow Sibling(y, x)$$

- A first cousin is a child of a parent's sibling

$$\forall x, y\ FirstCousin(x, y) \Leftrightarrow \exists p, ps\ Parent(p, x) \land Sibling(ps, x) \land Parent(ps, y)$$

Summary

- Knowledge representation languages should be declarative, compositional, expressive, context independent, and unambiguous.
- Logics differ in their ontological commitments and epistemological commitments. While PL commits only to the existence of facts, FOL commits to the existence of objects and relations and thereby gains expressive power.
- The syntax of FOL extends the PL by adding terms to represent objects, and having universal and existential quantifiers to construct assertions about all or some of the possible values of the quantified variables.
- A possible world, or model, for first-order logic includes a set of objects and an interpretation that maps constant symbols to objects, predicate symbols to relations among objects, and function symbols to functions on objects.
- An atomic sentence is true just when the relation named by the predicate holds between the objects named by the terms. Extended interpretations, which map quantifier variables to objects in the model, define the truth of quantified sentences.
- Developing a KB in FOL requires a careful process of analyzing the domain, choosing a vocabulary, and encoding the axioms required to support the desired inferences.