Outline

1. Knowledge-based agents
2. Wumpus world
3. Logic in general—models and entailment
4. Propositional (Boolean) logic
5. Equivalence, validity, satisfiability
6. Inference rules and theorem proving
   - forward chaining
   - backward chaining
   - resolution
7. Propositional knowledge-based agents
8. Boolean circuit agents

Resolution

Conjunctive Normal Form (CNF—universal)

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

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Resolution

Conjunctive Normal Form (CNF—universal)

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

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

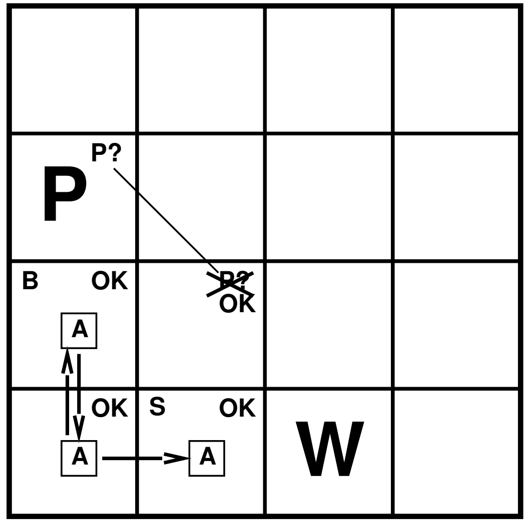
Resolution inference rule (for CNF): complete for propositional logic

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

where ℓ_i and m_j are complementary literals.

- Example:

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



- Resolution is sound and complete for propositional logic
- Soundness:

$$\frac{C \vee l \qquad D \vee \neg l}{C \vee D}$$

where ℓ_i and m_j are complementary literals.
Soundness: Prove $C \vee l, D \vee \neg l \models C \vee D$

Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)$.

   $$(B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \lor \beta$.

   $$(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg(P_{1,2} \lor P_{2,1}) \lor B_{1,1})$$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $$(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land ((\neg P_{1,2} \land \neg P_{2,1}) \lor B_{1,1})$$

4. Apply distributivity law ($\lor$ over $\land$) and flatten:

   $$(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1})$$

Resolution algorithm


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

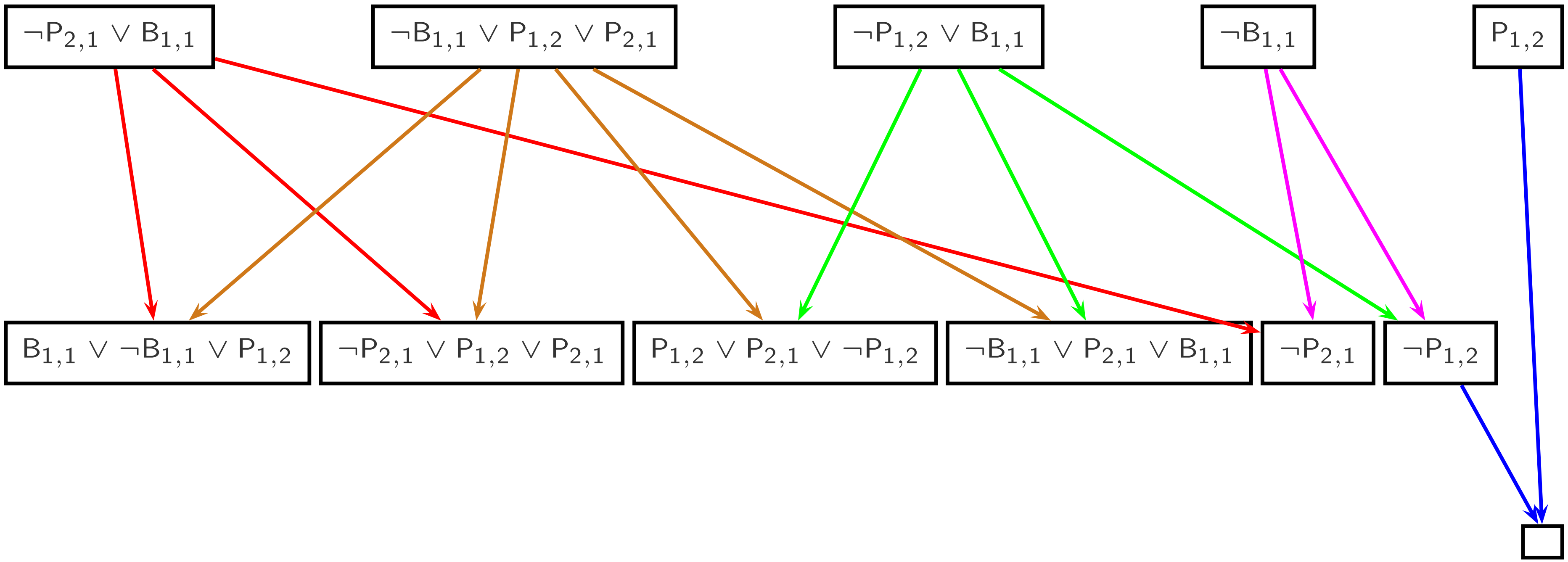
```
function PL-RESOLUTION(KB, α) returns true or false
    inputs: KB, the knowledge base, a sentence in propositional logic
            α, the query, a sentence in propositional logic

    clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
    new ← { }
    loop do
        for each Ci, Cj in clauses do
            resolvents ← PL-RESOLVE(Ci, Cj)
            if resolvents contains the empty clause then return true
            new ← new ∪ resolvents
        if new ⊆ clauses then return false
        clauses ← clauses ∪ new
```

Resolution example

$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$ $\alpha = \neg P_{1,2}$

Resolution example

$$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1} \quad \alpha = \neg P_{1,2}$$



7. Propositional knowledge-based agents

Agents Based on Propositional Logic

- Agents that use inference and a knowledge base
  - Finding pits and wumpuses using logical inference
  - Keeping track of location and orientation
- Circuit-based agents

Agent Knowledge Base

- Instance of the generic **knowledge-based agents**:

```
function KB-AGENT(percept) returns an action
    static:  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
```

- Agent that reasons about **location of pits**, **wumpuses**, and **safe squares**.
- Modelling starts with a KB that states "**physics**" of the Wumpus world:

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \land \neg W_{1,1}$

Physics of the Wumpus World

- $[1, 1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i, j]$, a breeze can arise. . .

[$n = 4$**,** 64 **sentences]**

Physics of the Wumpus World

- $[1, 1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i, j]$, a breeze can arise...

[$n = 4$, 64 sentences]

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i,j]$, a breeze can arise... **[$n = 4$, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i,j]$, a stench can arise **[$n = 4$, 64 sentences]**

Physics of the Wumpus World

- $[1, 1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i, j]$, a breeze can arise. . .

[$n = 4$**, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i, j]$, a stench can arise **[**$n = 4$**, 64 sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i,j]$, a breeze can arise. . . **[**$n = 4$**,** 64 **sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i,j]$, a stench can arise **[**$n = 4$**,** 64 **sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

- There is **exactly** one Wumpus

Physics of the Wumpus World

- $[1, 1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i, j]$, a breeze can arise. . .

  [$n = 4$**, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i, j]$, a stench can arise **[**$n = 4$**, 64 sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

- There is **exactly** one Wumpus
  - . . . at least one $W_{1,1} \vee W_{1,2} \vee \ldots W_{4,3} \vee W_{4,4}$

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i,j]$, a breeze can arise… **[**$n = 4$**, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i,j]$, a stench can arise **[**$n = 4$**, 64 sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

- There is **exactly** one Wumpus
  - … at least one $W_{1,1} \vee W_{1,2} \vee \ldots W_{4,3} \vee W_{4,4}$
  - … at most one $?$

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i,j]$, a breeze can arise. . . **[**$n = 4$**, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i,j]$, a stench can arise **[**$n = 4$**, 64 sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

- There is **exactly** one Wumpus
  - . . . at least one $W_{1,1} \vee W_{1,2} \vee \ldots W_{4,3} \vee W_{4,4}$
  - . . . at most one $?$
    - For any two squares, one must be empty

Physics of the Wumpus World

- $[1,1]$ contains no pit or wumpus: $\neg P_{1,1} \wedge \neg W_{1,1}$
- For each square $[i,j]$, a breeze can arise. . . **[**$n = 4$**, 64 sentences]**

$$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i+1,j} \vee P_{i-1,j})$$

- For each square $[i,j]$, a stench can arise **[**$n = 4$**, 64 sentences]**

$$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i+1,j} \vee P_{i-1,j})$$

- There is **exactly** one Wumpus
  - . . . at least one $W_{1,1} \vee W_{1,2} \vee \ldots W_{4,3} \vee W_{4,4}$
  - . . . at most one ?
    - For any two squares, one must be empty
    - For x squares we get $\frac{x(x-1)}{2}$ sentences $\neg W_{1,1} \vee \neg W_{1,2}$

      For $n = 4$ we have 16 squares $W_{i,j}$ and **[**120 **sentences]**

Provable and Possibly Safe Squares

- A square $[i, j]$ is provably safe, if $KB \models (\neg P_{i,j} \land \neg W_{i,j})$
  There is no pitch and no wumpus
- A square $[i, j]$ is possibly safe, if $KB \not\models (P_{i,j} \lor W_{i,j})$
  It cannot deduce that there is a pitch or a wumpus

Wumpus World Agent using PL

```
function PL-Wumpus-Agent(percept) returns an action
    inputs:  percept, a list, [stench,breeze,glitter]
    static:  KB, a knowledge base, initially containing the ``physics'' of the wumpus world
             x, y, orientation, the agent's position (init. [1,1]) and orientation (init. right)
             visited, an array indicating which squares have been visited, initially false
             action, the agent's most recent action, initially null
             plan, an action sequence, initially empty

    update x,y,orientation, visited based on action
    if stench then Tell(KB,S_x,y) else Tell(KB,¬ S_x,y)
    if breeze then Tell(KB,B_x,y) else Tell(KB,¬ B_x,y)
    if glitter then action ← grab
    else if plan is nonempty then action ← Pop(plan)
    else if for some fringe square [i,j], Ask(KB,(¬ P_i,j ∧ ¬ W_i,j)) is true or
              for some fringe square [i,j], Ask(KB,(P_i,j ∨ W_i,j)) is false then do
          plan ← A*-Graph-Search(Route-Problem([x,y],orientation,[i,j],visited))
          action ← Pop(plan)
    else action ← a randomly chosen move
    return action
```

Keeping track of location and orientation

So far: Agent maintains *location* and *orientation* outside of *KB*.
Now: Integrate information into *KB*

- First try: Use locations $L_{i,j}$ to indicate that the agent is $[i, j]$

Keeping track of location and orientation

So far: Agent maintains *location* and *orientation* outside of *KB*.
Now: Integrate information into *KB*

- First try: Use locations $L_{i,j}$ to indicate that the agent is $[i,j]$
  - Init: $L_{1,1}$

Keeping track of location and orientation

So far: Agent maintains *location* and *orientation* outside of *KB*.
Now: Integrate information into *KB*

- First try: Use locations $L_{i,j}$ to indicate that the agent is $[i, j]$
  - Init: $L_{1,1}$
  - Sentences:

$$L_{1,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{2,1}\ (1)$$

$$L_{2,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{3,1}\ (2)$$

$$L_{3,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{4,1}\ (3)$$

$$L_{4,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{4,1}\ (4)$$

and analogously for FacingLeft, FacingUp, FacingDown, Forward, TurnLeft, TurnRight, and all $L_{i,j}$

[$n = 4$**,** $16 \times 4 \times 3 = 192$ **Sentences]**

Keeping track of location and orientation

So far: Agent maintains *location* and *orientation* outside of *KB*.
Now: Integrate information into *KB*

- First try: Use locations $L_{i,j}$ to indicate that the agent is $[i, j]$
  - Init: $L_{1,1}$
  - Sentences:

$$L_{1,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{2,1} \; (1)$$

$$L_{2,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{3,1} \; (2)$$

$$L_{3,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{4,1} \; (3)$$

$$L_{4,1} \wedge \text{FacingRight} \wedge \text{Forward} \Rightarrow L_{4,1} \; (4)$$

  and analogously for FacingLeft, FacingUp, FacingDown, Forward, TurnLeft, TurnRight, and all $L_{i,j}$

  [$n = 4$**,** $16 \times 4 \times 3 = 192$ **Sentences]**

- Strange: Apply (1) **adds** $L_{2,1}$, and so on...
  So in which location is the agent then? $L_{1,1}??, L_{2,1}??, \ldots$

Need to track location over time!

Use locations $L^t_{i,j}$ to indicate that Wumpus is in $[i, j]$ at time t.

- Init: $L^1_{1,1}$

Need to track location over time!

Use locations $L^t_{i,j}$ to indicate that Wumpus is in $[i, j]$ at time t.

- Init: $L^1_{1,1}$
- Sentences:

$$L^n_{1,1} \wedge \mathsf{FacingRight}^t \wedge \mathsf{Forward}^t \Rightarrow L^{n+1}_{2,1}$$

$$L^n_{2,1} \wedge \mathsf{FacingRight}^t \wedge \mathsf{Forward}^t \Rightarrow L^{n+1}_{3,1}$$

$$L^n_{3,1} \wedge \mathsf{FacingRight}^t \wedge \mathsf{Forward}^t \Rightarrow L^{n+1}_{4,1}$$

$$L^n_{4,1} \wedge \mathsf{FacingRight}^t \wedge \mathsf{Forward}^t \Rightarrow L^{n+1}_{4,1}$$

and analogously for $\mathsf{FacingLeft}$, $\mathsf{FacingUp}$, $\mathsf{FacingDown}$, $\mathsf{Forward}$, $\mathsf{TurnLeft}$, $\mathsf{TurnRight}$, and all $L^t_{i,j}$

Allowing the agent to do at most $t = 100$ steps

[$n = 4$**,** $16 \times 4 \times 3 \times 100 = 19.200$ **Sentences]**

8. Boolean circuit agents

To track location and orientation, for the knowledge-based agent we have

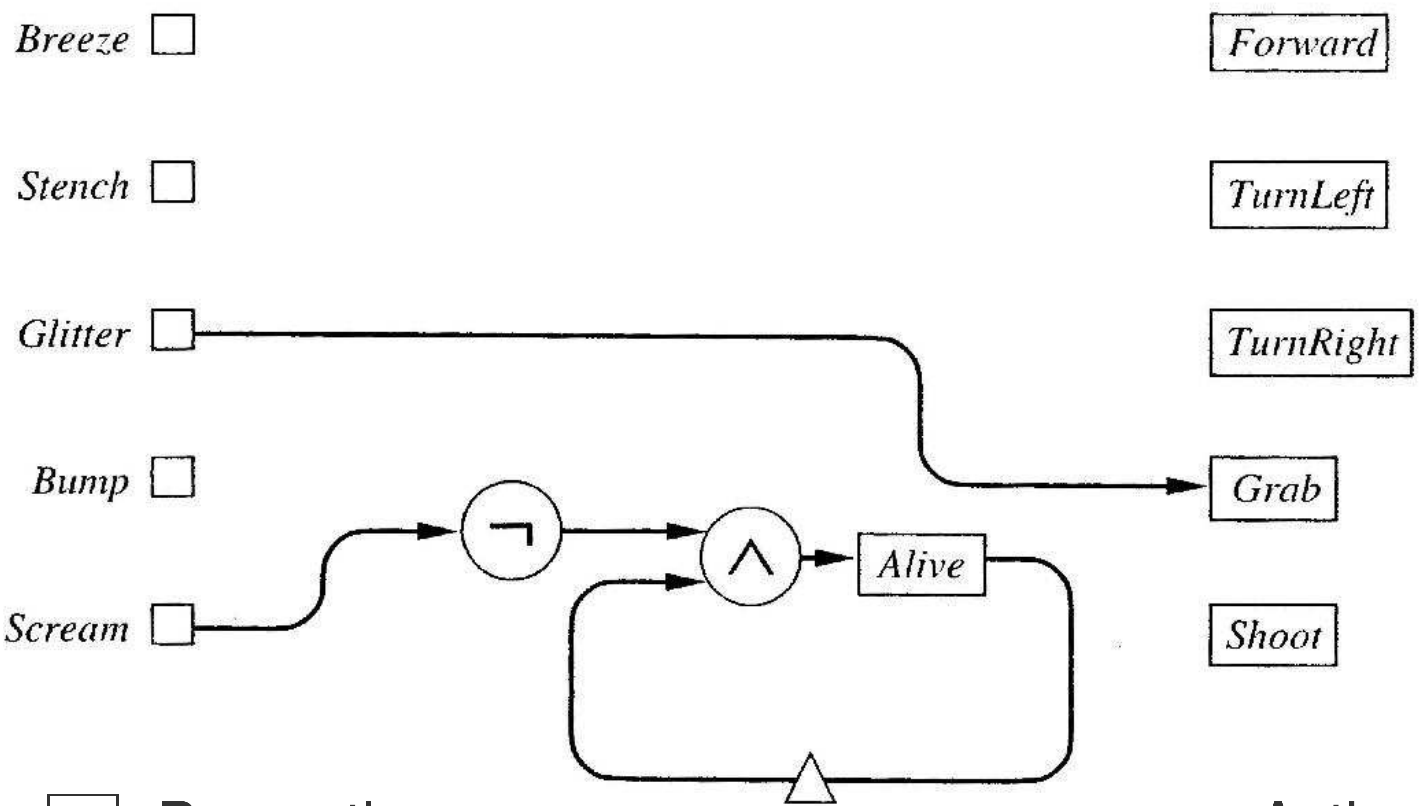
- ☹ to impose an upper bound of steps it can make (t)
- ☹ and nevertheless get tousands of sentences...
  ...most of which are never used!!
  Only one out of 192 $\approx 0.52\%$

A partial solution to remedy that problem: **circuit based agents**

Intuition:

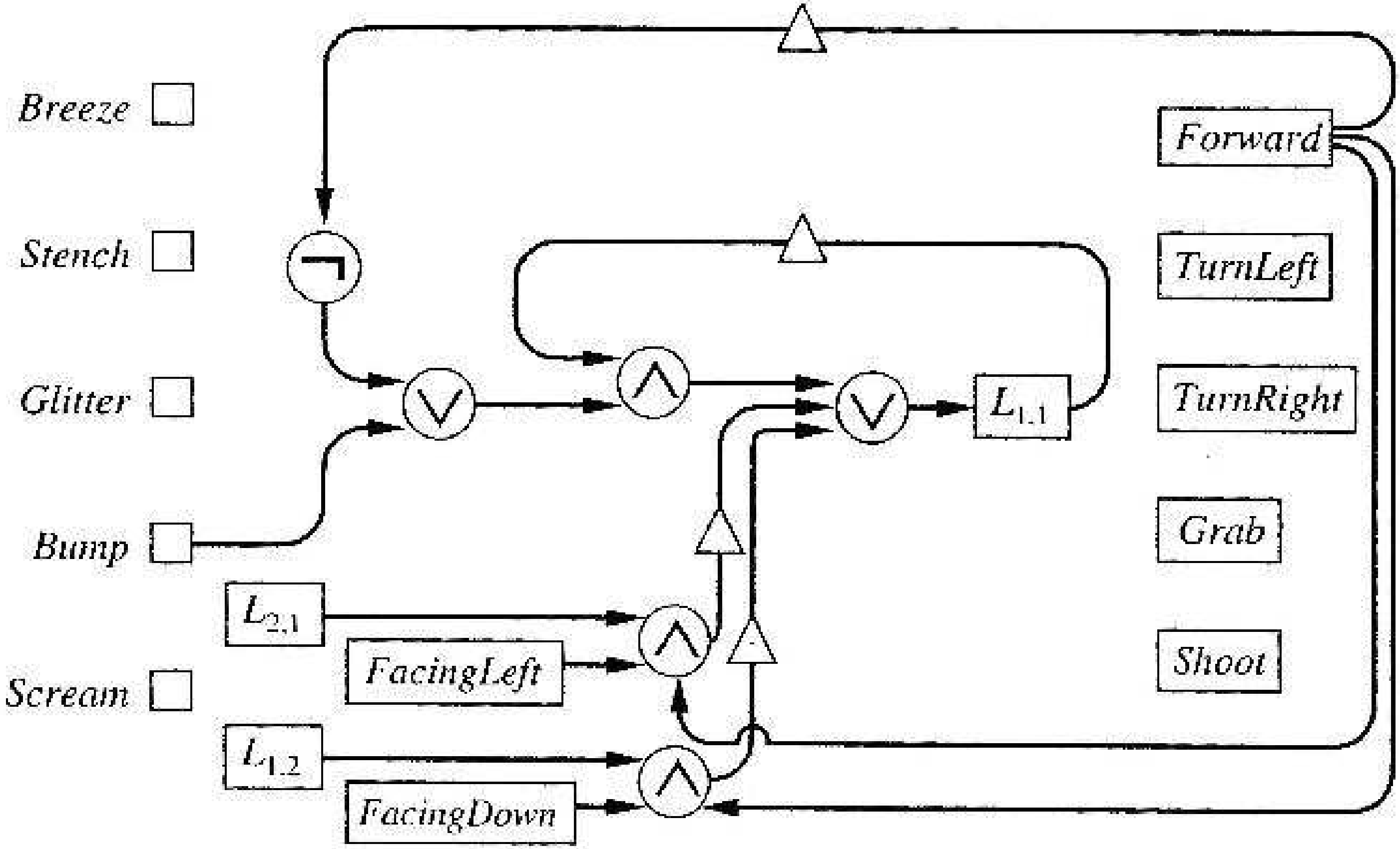
- Describe the transitions from t to $t + 1$
- Use recursion (How?)

Circuit-Agent Parts



- *Breeze* ☐: Perceptions — Actions *Forward*
- $\neg \mathrm{Scream}^t \wedge \mathrm{Alive}^t \Rightarrow \mathrm{Alive}^{t+1}$ — $\mathrm{Glitter}^t \Rightarrow \mathrm{Grab}^t$
- Propositional variables as **registers** *Alive*, *Forward*
- Logical connectives as **gates** (¬), (∧)

Circuit-Agent Parts



- $L^t_{2,1} \wedge \text{FacingLeft} \wedge \text{Forward}^t \Rightarrow L^{t+1}_{1,1}$
- $\neg\text{Forward}^t \wedge L^t_{1,1} \Rightarrow L^{t+1}_{1,1}$
- Sharing of "Inputs" via $\textcircled{\vee}$ **gates**

Unsatisfactory Things

- The KB contains many equivalences

  $$B_{i,j} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$$

  $$S_{i,j} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee P_{x-1,y})$$

  We would prefer (Why?) to express once and for all how breezes and stenches can arise.

Unsatisfactory Things

- The KB contains many equivalences

  $$B_{i,j} \Leftrightarrow (P_{x,y+1} \lor P_{x,y-1} \lor P_{x+1,y} \lor P_{x-1,y})$$
  $$S_{i,j} \Leftrightarrow (W_{x,y+1} \lor W_{x,y-1} \lor W_{x+1,y} \lor P_{x-1,y})$$

  We would prefer (Why?) to express once and for all how breezes and stenches can arise.

- For the knowledge-based agent, the KB contains thousands of sentences to track location and orientation

  $$L^{t}_{i,j} \land \mathrm{FacingRight} \land \mathrm{Forward} \Rightarrow L^{t+1}_{i+1,j}$$

  We would prefer to express once what happens if the agent goes to the right

Unsatisfactory Things

- The KB contains many equivalences

  $$B_{i,j} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$$
  $$S_{i,j} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee P_{x-1,y})$$

  We would prefer (Why?) to express once and for all how breezes and stenches can arise.

- For the knowledge-based agent, the KB contains thousands of sentences to track location and orientation

  $$L^{t}_{i,j} \wedge FacingRight \wedge Forward \Rightarrow L^{t+1}_{i+1,j}$$

  We would prefer to express once what happens if the agent goes to the right

- Is there another way than using time t to obtain consistent state updates?

  I.e., not ending with KBs where all $L_{1,1}, L_{2,1}, \ldots$ where $[i,j]$ have been visited?

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - syntax: formal structure of sentences
  - semantics: truth of sentences wrt models
  - entailment: necessary truth of one sentence given another
  - inference: deriving sentences from other sentences
  - soundess: derivations produce only entailed sentences
  - completeness: derivations can produce all entailed sentences
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Forward, backward chaining are linear-time, complete for Horn clauses
- Resolution is complete for propositional logic
- Propositional logic lacks expressive power

First-order logic

Chapter 8

Outline

- Why FOL?
- Syntax and semantics of FOL
- Fun with sentences
- Wumpus world in FOL


Why FOL?

Pros and cons of propositional logic

- 😊 Propositional logic is **declarative**: pieces of syntax correspond to facts

Pros and cons of propositional logic

- ☺ Propositional logic is **declarative**: pieces of syntax correspond to facts
- ☺ Propositional logic allows partial/disjunctive/negated information

  (unlike most data structures and databases)

Pros and cons of propositional logic

- 😈 Propositional logic is **declarative**: pieces of syntax correspond 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 $P_{1,2}$

Pros and cons of propositional logic

- Propositional logic is **declarative**: pieces of syntax correspond 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)

Pros and cons of propositional logic

- 😊 Propositional logic is **declarative**: pieces of syntax correspond 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"

Syntax and semantics of FOL

First-order 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 . . .

First-order 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, . . .

First-order 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, . . .
- Functions: father of, best friend, third inning of, one more than, end of . . .

Logics in general

Language	Ontological Commitment	Epistemological Commitment
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

Syntax of FOL: Basic elements

Constants	KingJohn, 2, UCB, ...
Predicates	Brother, $>$, ...
Functions	Sqrt, LeftLegOf, ...
Variables	x, y, a, b, ...
Connectives	$\wedge \quad \vee \quad \neg \quad \Rightarrow \quad \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall \; \exists$

Atomic sentences

Atomic sentence = $\text{predicate}(\text{term}_1, \ldots, \text{term}_n)$
or $\text{term}_1 = \text{term}_2$

Term = $\text{function}(\text{term}_1, \ldots, \text{term}_n)$
or constant or variable

E.g., $\text{Brother}(\text{KingJohn}, \text{RichardTheLionheart})$
$> (\text{Length}(\text{LeftLegOf}(\text{Richard})), \text{Length}(\text{LeftLegOf}(\text{KingJohn})))$

Complex sentences

Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \;\Rightarrow\; S_2, \quad S_1 \;\Leftrightarrow\; S_2$$

E.g. $Sibling(KingJohn, Richard) \;\Rightarrow\; Sibling(Richard, KingJohn)$

$>(1,2) \vee \leq(1,2)$

$>(1,2) \wedge \neg{>}(1,2)$

Truth in first-order logic

- 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

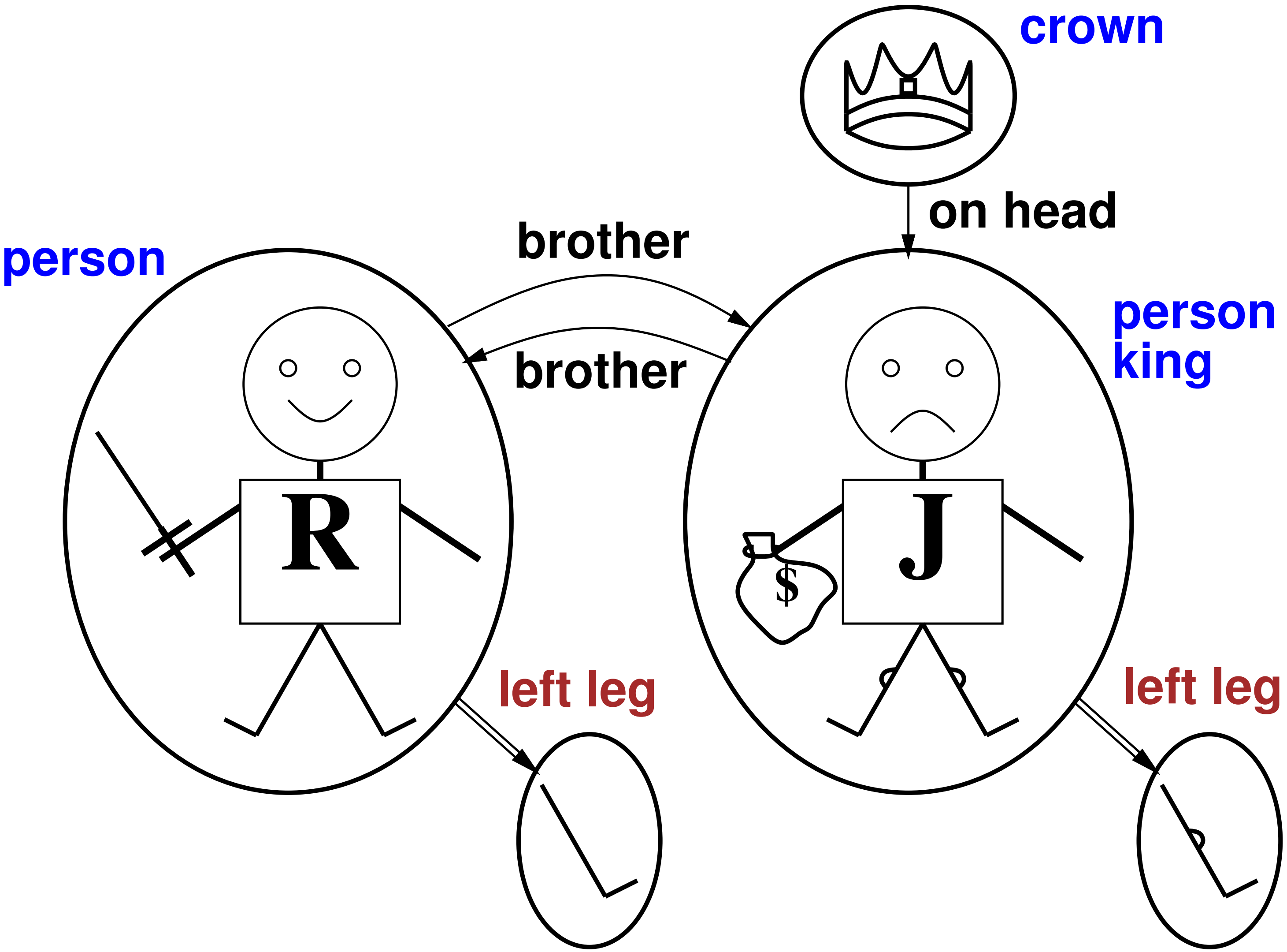
  constant symbols $\rightarrow$ objects

  predicate symbols $\rightarrow$ relations

  function symbols $\rightarrow$ functional relations

- An atomic sentence $\mathsf{predicate}(\mathsf{term}_1, \ldots, \mathsf{term}_n)$ is true
  iff the objects referred to by $\mathsf{term}_1, \ldots, \mathsf{term}_n$
  are in the relation referred to by $\mathsf{predicate}$

Models for FOL: Example



Truth example

Consider the interpretation in which

$$\begin{aligned} \mathsf{Richard} &\rightarrow \text{Richard the Lionheart} \\ \mathsf{John} &\rightarrow \text{the evil King John} \\ \mathsf{Brother} &\rightarrow \text{the brotherhood relation} \end{aligned}$$

Under this interpretation, $\mathsf{Brother}(\mathsf{Richard}, \mathsf{John})$ is true just in case Richard the Lionheart and the evil King John are in the brotherhood relation in the model

Models for FOL: Lots!

- Entailment in propositional logic can be computed by enumerating models
- We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

- For each k-ary predicate P_k in the vocabulary

Models for FOL: Lots!

- Entailment in propositional logic can be computed by enumerating models
- We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

- For each k-ary predicate P_k in the vocabulary
  - For each possible k-ary relation on n objects

Models for FOL: Lots!

- Entailment in propositional logic can be computed by enumerating models
- We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

- For each k-ary predicate P_k in the vocabulary
  - For each possible k-ary relation on n objects
    - For each constant symbol C in the vocabulary

Models for FOL: Lots!

- Entailment in propositional logic can be computed by enumerating models
- We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

- For each k-ary predicate P_k in the vocabulary
  - For each possible k-ary relation on n objects
    - For each constant symbol C in the vocabulary
      - For each choice of referent for C from n objects . . .

Models for FOL: Lots!

- Entailment in propositional logic can be computed by enumerating models
- We **can** enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

- For each k-ary predicate P_k in the vocabulary
  - For each possible k-ary relation on n objects
    - For each constant symbol C in the vocabulary
      - For each choice of referent for C from n objects . . .

Computing entailment by enumerating FOL models is not easy!

Universal quantification

- $\forall \langle \text{variables} \rangle \ \ \langle \text{sentence} \rangle$

Universal quantification

- $\forall \langle \text{variables} \rangle \ \ \langle \text{sentence} \rangle$
- Everyone at *Saarbrücken* is smart:

Universal quantification

- $\forall \langle \text{variables} \rangle \; \langle \text{sentence} \rangle$
- Everyone at *Saarbrücken* is smart:

$$\forall x \;\; \text{At}(x, Saarbruecken) \;\Rightarrow\; \text{Smart}(x)$$

Universal quantification

- $\forall \langle \text{variables} \rangle \ \ \langle \text{sentence} \rangle$
- Everyone at *Saarbrücken* is smart:

$$\forall x \;\; At(x, Saarbruecken) \;\Rightarrow\; Smart(x)$$

- $\forall x \;\; P$ is true in a model m iff P is true with x being **each** possible object in the model

Universal quantification

- $\forall \langle \text{variables} \rangle \ \langle \text{sentence} \rangle$
- Everyone at *Saarbrücken* is smart:

$$\forall x \ \ At(x, Saarbruecken) \ \Rightarrow \ Smart(x)$$

- $\forall x \ \ P$ is true in a model m iff P is true with x being **each** possible object in the model
- **Roughly** speaking, equivalent to the conjunction of instantiations of P

Universal quantification

- $\forall \langle \text{variables} \rangle \quad \langle \text{sentence} \rangle$
- Everyone at *Saarbrücken* is smart:

$$\forall x \;\; At(x, Saarbruecken) \;\Rightarrow\; Smart(x)$$

- $\forall x \;\; P$ is true in a model m iff P is true with x being **each** possible object in the model
- **Roughly** speaking, equivalent to the conjunction of instantiations of P

$$\begin{array}{ll} & (At(KingJohn, Saarbruecken) \;\Rightarrow\; Smart(KingJohn)) \\ \wedge & (At(Richard, Saarbruecken) \;\Rightarrow\; Smart(Richard)) \\ \wedge & (At(Saarbruecken, Saarbruecken) \;\Rightarrow\; Smart(Saarbruecken)) \\ \wedge & \ldots \end{array}$$

A common mistake to avoid

- Typically, $\Rightarrow$ is the main connective with $\forall$
- Common mistake: using $\wedge$ as the main connective with $\forall$:

$$\forall x \;\; \mathsf{At}(x, Saarbruecken) \wedge \mathsf{Smart}(x)$$

means “Everyone is at *Saarbrücken* and everyone is smart”

Existential quantification

- $\exists \langle variables \rangle \ \ \langle sentence \rangle$
- Someone at Stanford is smart:

$$\exists x \ \ At(x, Stanford) \land Smart(x)$$

- $\exists x \ \ P$ is true in a model m iff P is true with x being **some** possible object in the model
- **Roughly** speaking, equivalent to the disjunction of instantiations of P

$$\begin{array}{ll} & (At(KingJohn, Stanford) \land Smart(KingJohn)) \\ \lor & (At(Richard, Stanford) \land Smart(Richard)) \\ \lor & (At(Stanford, Stanford) \land Smart(Stanford)) \\ \lor & \ldots \end{array}$$

Another common mistake to avoid

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

$$\exists x \;\; At(x, Stanford) \Rightarrow Smart(x)$$

  is true if there is anyone who is not at Stanford!

Encoding World Knowledge

- Brothers are siblings
  $\forall x, y \; Brother(x, y) \Rightarrow Sibling(x, y)$.

Encoding World Knowledge

- Brothers are siblings
  $\forall x, y \;\; \mathrm{Brother}(x, y) \Rightarrow \mathrm{Sibling}(x, y).$
- "Sibling" is symmetric
  $\forall x, y \;\; \mathrm{Sibling}(x, y) \Leftrightarrow \mathrm{Sibling}(y, x).$

Encoding World Knowledge

- Brothers are siblings
  $\forall x, y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y).$
- "Sibling" is symmetric
  $\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x).$
- One's mother is one's female parent
  $\forall x, y \;\; Mother(x, y) \;\Leftrightarrow\; (Female(x) \wedge Parent(x, y)).$

Encoding World Knowledge

- Brothers are siblings
  $\forall x, y \ \ \text{Brother}(x, y) \ \Rightarrow \ \text{Sibling}(x, y).$
- "Sibling" is symmetric
  $\forall x, y \ \ \text{Sibling}(x, y) \ \Leftrightarrow \ \text{Sibling}(y, x).$
- One's mother is one's female parent
  $\forall x, y \ \ \text{Mother}(x, y) \ \Leftrightarrow \ (\text{Female}(x) \wedge \text{Parent}(x, y)).$
- A first cousin is a child of a parent's sibling
  $\forall x, y \ \ \text{FirstCousin}(x, y) \ \Leftrightarrow \ \exists\, p, ps \ \ \text{Parent}(p, x) \wedge \text{Sibling}(ps, p) \wedge \text{Parent}(ps, y)$

Equality

- $\mathsf{term}_1 = \mathsf{term}_2$ is true under a given interpretation if and only if term_1 and term_2 refer to the same object
- Example:

  $1 = 2$ and $\forall x \;\; \times(\mathsf{Sqrt}(x), \mathsf{Sqrt}(x)) = x$ are satisfiable

  $2 = 2$ is valid
- Example: Definition of (full) Sibling in terms of Parent:

$$\forall x, y \;\; \mathsf{Sibling}(x, y) \;\Leftrightarrow\; \left[\begin{array}{lll} & \neg(x = y) & \\ \wedge & \exists\, m, f & \neg(m = f) \\ & & \wedge \mathsf{Parent}(m, x) \\ & & \wedge \mathsf{Parent}(f, x) \\ & & \wedge \mathsf{Parent}(m, y) \\ & & \wedge \mathsf{Parent}(f, y) \end{array} \right]$$

Wumpus World in FOL

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathsf{Tell}(\mathsf{KB}, \mathsf{Percept}([\mathsf{Smell}, \mathsf{Breeze}, \mathsf{None}], 5))$$
$$\mathsf{Ask}(\mathsf{KB}, \exists\, a \;\; \mathsf{Action}(a, 5))$$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathrm{Tell}(\mathrm{KB}, \mathrm{Percept}([\mathrm{Smell}, \mathrm{Breeze}, \mathrm{None}], 5))$$
$$\mathrm{Ask}(\mathrm{KB}, \exists\, a \;\; \mathrm{Action}(a, 5))$$

- Assume a rule $\mathrm{Smell} \Rightarrow \mathrm{Shoot}$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathsf{Tell}(\mathsf{KB}, \mathsf{Percept}([\mathsf{Smell}, \mathsf{Breeze}, \mathsf{None}], 5))$$
$$\mathsf{Ask}(\mathsf{KB}, \exists a\ \ \mathsf{Action}(a, 5))$$

- Assume a rule $\mathsf{Smell} \Rightarrow \mathsf{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathsf{Tell}(\mathsf{KB}, \mathsf{Percept}([\mathsf{Smell}, \mathsf{Breeze}, \mathsf{None}], 5))$$
$$\mathsf{Ask}(\mathsf{KB}, \exists\, \mathsf{a}\;\; \mathsf{Action}(\mathsf{a}, 5))$$

- Assume a rule $\mathsf{Smell} \Rightarrow \mathsf{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{\mathsf{a}/\mathsf{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathrm{Tell}(\mathrm{KB}, \mathrm{Percept}([\mathrm{Smell}, \mathrm{Breeze}, \mathrm{None}], 5))$$
$$\mathrm{Ask}(\mathrm{KB}, \exists a\ \ \mathrm{Action}(a, 5))$$

- Assume a rule $\mathrm{Smell} \Rightarrow \mathrm{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{a/\mathrm{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of plugging σ into S; e.g.,

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\text{Tell}(\text{KB}, \text{Percept}([\text{Smell}, \text{Breeze}, \text{None}], 5))$$
$$\text{Ask}(\text{KB}, \exists\, a \;\; \text{Action}(a, 5))$$

- Assume a rule $\text{Smell} \Rightarrow \text{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{a/\text{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of plugging σ into S; e.g.,
  - $S = \text{Smarter}(x, y)$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\text{Tell}(KB, \text{Percept}([\text{Smell}, \text{Breeze}, \text{None}], 5))$$
$$\text{Ask}(KB, \exists a \;\; \text{Action}(a, 5))$$

- Assume a rule $\text{Smell} \Rightarrow \text{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{a/\text{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of plugging σ into S; e.g.,
  - $S = \text{Smarter}(x, y)$
  - $\sigma = \{x/\text{Hillary}, y/\text{Bill}\}$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\mathrm{Tell}(KB, \mathrm{Percept}([\mathrm{Smell}, \mathrm{Breeze}, \mathrm{None}], 5))$$
$$\mathrm{Ask}(KB, \exists a \;\; \mathrm{Action}(a, 5))$$

- Assume a rule $\mathrm{Smell} \Rightarrow \mathrm{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{a/\mathrm{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of plugging σ into S; e.g.,
  - $S = \mathrm{Smarter}(x, y)$
  - $\sigma = \{x/\mathrm{Hillary}, y/\mathrm{Bill}\}$
  - $S\sigma = \mathrm{Smarter}(\mathrm{Hillary}, \mathrm{Bill})$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at $t = 5$:

$$\text{Tell}(\text{KB}, \text{Percept}([\text{Smell}, \text{Breeze}, \text{None}], 5))$$
$$\text{Ask}(\text{KB}, \exists\, a \;\; \text{Action}(a, 5))$$

- Assume a rule $\text{Smell} \Rightarrow \text{Shoot}$
- Does KB entail any particular actions at $t = 5$?
  Answer: Yes, $\{a/\text{Shoot}\}$ $\leftarrow$ substitution (binding list)
- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of plugging σ into S; e.g.,
  - $S = \text{Smarter}(x, y)$
  - $\sigma = \{x/\text{Hillary}, y/\text{Bill}\}$
  - $S\sigma = \text{Smarter}(\text{Hillary}, \text{Bill})$
- $\text{Ask}(\text{KB}, S)$ returns some/all σ such that $\text{KB} \models S\sigma$

Knowledge base for the wumpus world

- "Perception"

$$\forall b, g, t \;\; Percept([Smell, b, g], t) \;\Rightarrow\; Smelt(t)$$
$$\forall s, b, t \;\; Percept([s, b, Glitter], t) \;\Rightarrow\; AtGold(t)$$

- Reflex: $\forall t \;\; AtGold(t) \;\Rightarrow\; Action(Grab, t)$
- Reflex with internal state: do we have the gold already?

$$\forall t \;\; AtGold(t) \wedge \neg Holding(Gold, t) \;\Rightarrow\; Action(Grab, t)$$

$Holding(Gold, t)$ cannot be observed

$\Rightarrow$ keeping track of change is essential

Deducing hidden properties

- Properties of locations: $\forall x, t \;\; At(Agent, x, t) \wedge Smelt(t) \;\Rightarrow\; Smelly(x)$
  $\forall x, t \;\; At(Agent, x, t) \wedge Breeze(t) \;\Rightarrow\; Breezy(x)$
- Squares are breezy near a pit:
  - Diagnostic rule—infer cause from effect

    $$\forall y \;\; Breezy(y) \;\Rightarrow\; \exists x \;\; Pit(x) \wedge Adjacent(x, y)$$

  - Causal rule—infer effect from cause

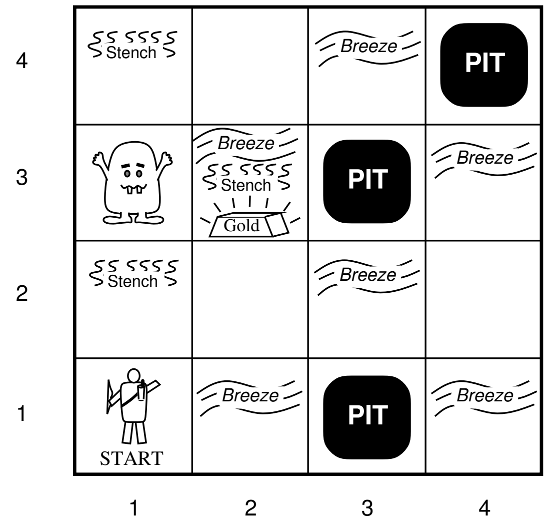
    $$\forall x, y \;\; Pit(x) \wedge Adjacent(x, y) \;\Rightarrow\; Breezy(y)$$

    Neither of these is complete—e.g., the causal rule doesn't say whether squares far away from pits can be breezy
  - Definition for the Breezy predicate:

    $$\forall y \;\; Breezy(y) \;\Leftrightarrow\; [\exists x \;\; Pit(x) \wedge Adjacent(x, y)]$$

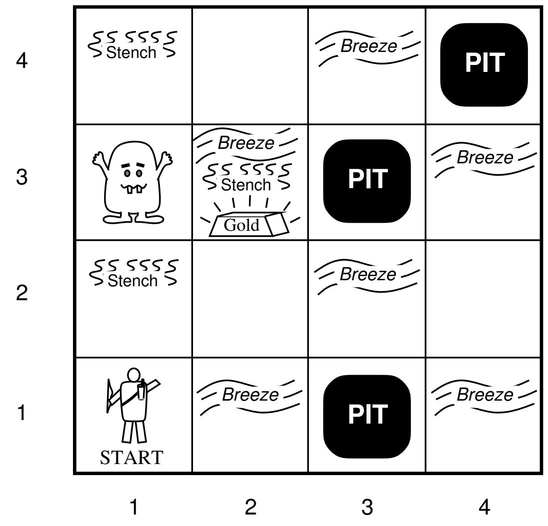
Wumpus Physics in FOL

- $B(x, y)$: There is a breeze in square $[x, y]$



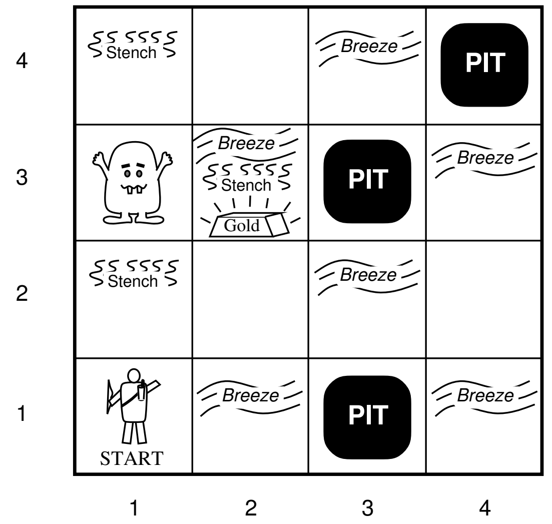
Wumpus Physics in FOL

- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$



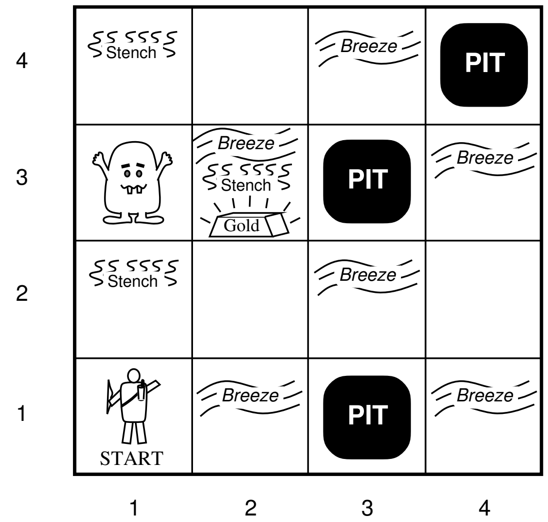
Wumpus Physics in FOL

- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t



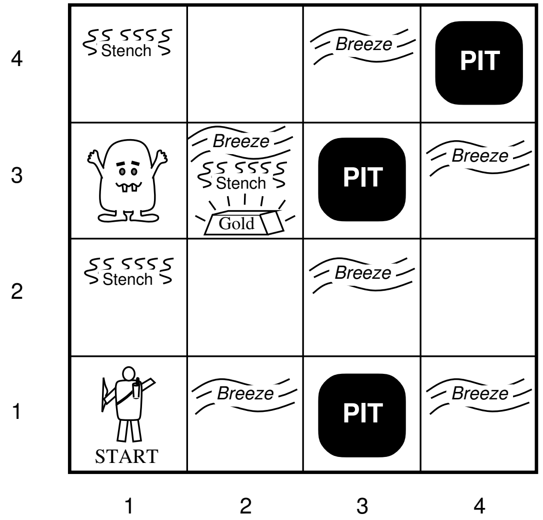
Wumpus Physics in FOL

- $B(x, y)$: There is a **breeze** in square $[x, y]$
- $S(x, y)$: There is a **stench** in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t
- Initial state: $L(1, 1, 1)$

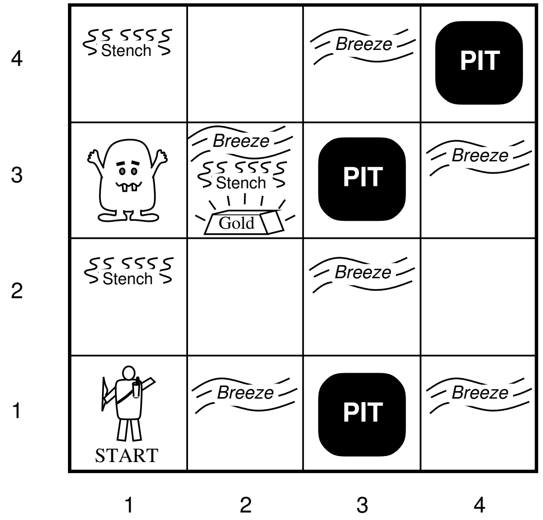


Wumpus Physics in FOL

- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t
- Initial state: $L(1, 1, 1)$
- Deriving knowledge
  $\forall x, y \,.\, B(x, y) \Leftrightarrow P(x, y+1) \vee P(x, y-1) \vee P(x+1, y) \vee P(x-1, y)$
  $\forall x, y \,.\, S(x, y) \Leftrightarrow W(x, y+1) \vee W(x, y-1) \vee W(x+1, y) \vee W(x-1, y)$



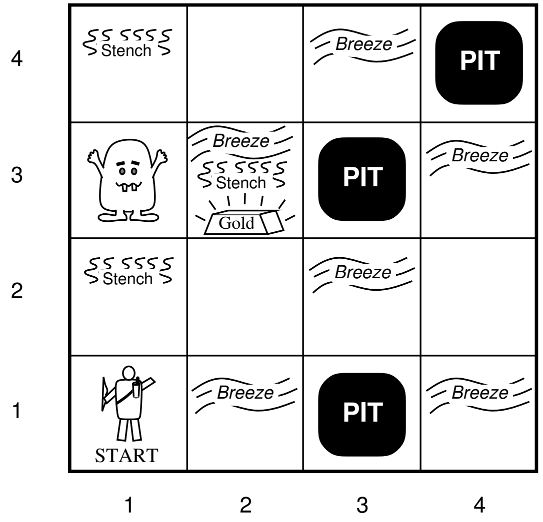
Wumpus Physics in FOL



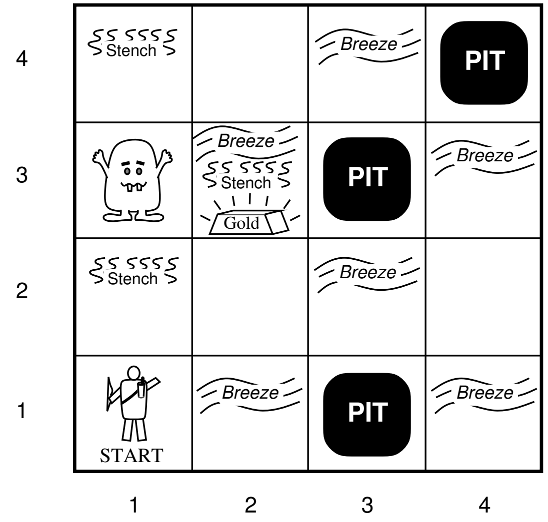
- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t
- Initial state: $L(1, 1, 1)$
- Deriving knowledge
  $\forall x, y \;.\; B(x, y) \Leftrightarrow P(x, y+1) \vee P(x, y-1) \vee P(x+1, y) \vee P(x-1, y)$
  $\forall x, y \;.\; S(x, y) \Leftrightarrow W(x, y+1) \vee W(x, y-1) \vee W(x+1, y) \vee W(x-1, y)$
- There is exactly one wumpus:

Wumpus Physics in FOL

- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t
- Initial state: $L(1, 1, 1)$
- Deriving knowledge
  $\forall x, y \,.\, B(x, y) \Leftrightarrow P(x, y+1) \lor P(x, y-1) \lor P(x+1, y) \lor P(x-1, y)$
  $\forall x, y \,.\, S(x, y) \Leftrightarrow W(x, y+1) \lor W(x, y-1) \lor W(x+1, y) \lor W(x-1, y)$
- There is exactly one wumpus:
  - At least one: $\exists x, y \,.\, W(x, y)$



Wumpus Physics in FOL

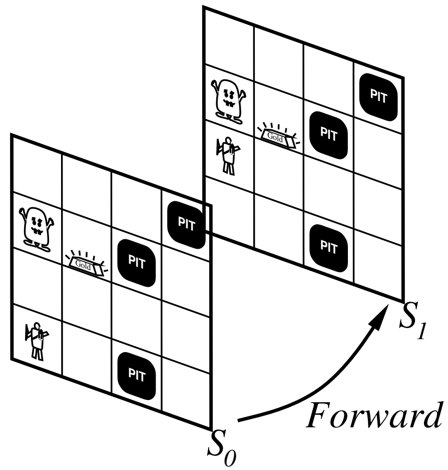


- $B(x, y)$: There is a breeze in square $[x, y]$
- $S(x, y)$: There is a stench in square $[x, y]$
- $L(t, x, y)$: The agent is in square $[x, y]$ at time t
- Initial state: $L(1, 1, 1)$
- Deriving knowledge
  $\forall x, y \,.\, B(x, y) \Leftrightarrow P(x, y + 1) \vee P(x, y - 1) \vee P(x + 1, y) \vee P(x - 1, y)$
  $\forall x, y \,.\, S(x, y) \Leftrightarrow W(x, y + 1) \vee W(x, y - 1) \vee W(x + 1, y) \vee W(x - 1, y)$
- There is exactly one wumpus:
  - At least one: $\exists x, y \,.\, W(x, y)$
  - At most one: $\forall x, y, u, v \,.\, (u \neq x \vee y \neq v) \Rightarrow (\neg W(x, y) \vee \neg W(u, v))$
    For any two (different) squares, one must be empty
    (was 120 sentences in PL)

Keeping track of change

- Facts hold in situations, rather than eternally
  E.g., $\mathsf{Holding(Gold, Now)}$ rather than just $\mathsf{Holding(Gold)}$
- Situation calculus is one way to represent change in FOL:
  - Adds a situation argument to each **non-eternal** predicate
    - Example: Now in $\mathsf{Holding(Gold, Now)}$ denotes a situation
    - State-dependent predicates are called fluents
  - Situations are connected by the Result function
    - Example: $\mathsf{Result(a, s)}$ is the situation that results from doing a in s

$$\mathsf{S1 = Result(Forward, S0)}$$



Describing actions I

- "Possibility" axiom — describe when an action is possible

$$\forall s \;\; AtGold(s) \;\Rightarrow\; Poss(Grab(Gold), s)$$

- "Effect" axiom—describe changes due to action

$$\forall s, x \;\; Poss(Grab(x), s) \;\Rightarrow\; Holding(x, Result(Grab(x), s))$$

- "Frame" axiom—describe **non-changes** due to action

$$\forall s \;\; HaveArrow(s) \;\Rightarrow\; HaveArrow(Result(Grab(Gold), s))$$

- Frame problem: find an elegant way to handle non-change
  (a) representation—avoid frame axioms
  (b) inference—avoid repeated "copy-overs" to keep track of state

Qualification and Ramification Problems

- Qualification problem: true descriptions of real actions require endless caveats—what if gold is slippery or nailed down or . . .
  - Description of the world is always an abstraction of the world
  - Q: How can we ensure that abstraction is adequate, i.e. we didn't leave out important aspects?
- Ramification problem: real actions have many secondary consequences—what about the dust on the gold, wear and tear on gloves, . . .
  - Q: What is changing location when you move your bag?

Describing actions II

- Successor-state axioms solve the representational frame problem

  Each axiom is "about" a **predicate** (not an action per se):

  Action a is possible in $s \Longrightarrow$

  $$\left(\begin{array}{l} \text{P true after applying } a \;\Leftrightarrow \\ \left[\begin{array}{ll} & \text{the action } a \text{ made P true} \\ \vee & \text{P true already and action } a \text{ did not make P false} \end{array} \right] \end{array} \right)$$

- For holding the gold:

  $$\begin{array}{l} \forall a, s \;\; Holding(Gold, Result(a, s)) \;\Leftrightarrow \\ \qquad [(a = Grab(Gold) \wedge AtGold(s)) \\ \qquad \vee (Holding(Gold, s) \wedge a \neq Release(Gold))] \end{array}$$

Making plans

- Initial condition in KB:

$$At(Agent, [1, 1], S_0)$$
$$At(Gold, [1, 2], S_0)$$

- Query: $Ask(KB, \exists s \; Holding(Gold, s))$
  i.e., in what situation will I be holding the gold?

- Answer: $\{s/Result(Grab, Result(Forward, S_0))\}$
  i.e., go forward and then grab the gold

- This assumes that the agent is interested in plans starting at S_0 and that S_0 is the only situation described in the KB

Making plans: A better way

- Represent plans as action sequences $[a_1, a_2, \ldots, a_n]$

- $\text{PlanResult}(p, s)$ is the result of executing p in s

- Then the query $\text{Ask}(KB, \exists\, p\ \ \text{Holding}(\text{Gold}, \text{PlanResult}(p, S_0)))$ has the solution $\{p/[\text{Forward}, \text{Grab}]\}$

- Definition of PlanResult in terms of Result:
  $\forall\, s\ \ \text{PlanResult}([\,], s) = s$
  $\forall\, a, p, s\ \ \text{PlanResult}([a|p], s) = \text{PlanResult}(p, \text{Result}(a, s))$

- Planning systems are special-purpose reasoners designed to do this type of inference more efficiently than a general-purpose reasoner

Summary

- First-order logic:

Summary

- First-order logic:
  - objects and relations are semantic primitives

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers

- Increased expressive power: sufficient to define wumpus world

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define wumpus world
- Situation calculus:

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define wumpus world
- Situation calculus:
  - conventions for describing actions and change in FOL

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define wumpus world
- Situation calculus:
  - conventions for describing actions and change in FOL
  - can formulate planning as inference on a situation calculus KB

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers

- Increased expressive power: sufficient to define wumpus world

- Situation calculus:
  - conventions for describing actions and change in FOL
  - can formulate planning as inference on a situation calculus KB

Next: Inference in First-Order Logic

Inference in first-order logic

Chapter 9

Outline

- Reducing first-order inference to propositional inference
- Unification
- Generalized Modus Ponens
- Forward and backward chaining
- Logic programming
- Resolution

A brief history of reasoning

450 B.C.	Stoics	propositional logic, inference (maybe)
322 B.C.	Aristotle	"syllogisms" (inference rules), quantifiers
1565	Cardano	probability theory (propositional logic + uncertainty)
1847	Boole	propositional logic (again)
1879	Frege	first-order logic
1922	Wittgenstein	proof by truth tables
1930	Gödel	$\exists$ complete algorithm for FOL
1930	Herbrand	complete algorithm for FOL (reduce to propositional)
1931	Gödel	$\neg\exists$ complete algorithm for arithmetic
1960	Davis/Putnam	"practical" algorithm for propositional logic
1965	Robinson	"practical" algorithm for FOL—resolution

- Resolution: **Fully Generalized Modus Ponens** + **Unification**
- Completeness by reduction to completeness of resolution for propositional logic.

Universal instantiation (UI)

Every instantiation of a universally quantified sentence is entailed by it:

$$\frac{\forall \mathsf{v} \ \ \alpha}{\textsc{Subst}(\{\mathsf{v}/\mathsf{g}\}, \alpha)}$$

for any variable v and ground term g.

E.g., $\forall \mathsf{x} \ \ \mathsf{King(x)} \land \mathsf{Greedy(x)} \ \Rightarrow \ \mathsf{Evil(x)}$ yields

$$\mathsf{King(John)} \land \mathsf{Greedy(John)} \ \Rightarrow \ \mathsf{Evil(John)}$$

$$\mathsf{King(Richard)} \land \mathsf{Greedy(Richard)} \ \Rightarrow \ \mathsf{Evil(Richard)}$$

$$\mathsf{King(Father(John))} \land \mathsf{Greedy(Father(John))} \ \Rightarrow \ \mathsf{Evil(Father(John))}$$

$$\vdots$$

Existential instantiation (EI)

- For any sentence α, variable v, and constant symbol k **that does not appear elsewhere in the knowledge base**:

$$\frac{\exists v \;\; \alpha}{\textsc{Subst}(\{v/k\}, \alpha)}$$

- Why new k? $KB := \{\neg Sunny(Today), \exists\, day \;\; Sunny(day)\}$
- Example: $\exists x \;\; Crown(x) \wedge OnHead(x, John)$ yields

  $$Crown(C_1) \wedge OnHead(C_1, John)$$

  provided C_1 is a new constant symbol, called a Skolem constant
- Another example: from $\exists x \;\; d(x^y)/dy = x^y$ we obtain

  $$d(e^y)/dy = e^y$$

  provided e is a new constant symbol

Existential instantiation contd.

- UI can be applied several times to **add** new sentences;
  - the new KB is logically equivalent to the old
- EI can be applied once to **replace** the existential sentence;
  - the new KB is **not** equivalent to the old,
  - but the new KB is satisfiable iff the old KB was satisfiable

Reduction to propositional inference

Suppose the KB contains just the following:

$$\forall x \;\; King(x) \land Greedy(x) \;\Rightarrow\; Evil(x)$$
$$King(John)$$
$$Greedy(John)$$
$$Brother(Richard, John)$$

Instantiating the universal sentence in **all possible** ways, we have

$$King(John) \land Greedy(John) \;\Rightarrow\; Evil(John)$$
$$King(Richard) \land Greedy(Richard) \;\Rightarrow\; Evil(Richard)$$
$$King(John)$$
$$Greedy(John)$$
$$Brother(Richard, John)$$

The new KB is propositionalized: proposition symbols are
$King(John),\; Greedy(John),\; Evil(John), King(Richard)$ etc.

Reduction contd.

- Claim:

  a ground sentence is entailed by new KB iff entailed by original KB

- Claim:

  every FOL KB can be propositionalized so as to preserve entailment

- Idea: propositionalize KB and query, apply resolution, return result
- Problem: with function symbols, there are infinitely many ground terms
  Example: $\mathsf{Father}(\mathsf{Father}(\mathsf{Father}(\mathsf{John})))$

Reduction contd.

- Theorem: Herbrand (1930). If a sentence α is entailed by an FOL KB, it is entailed by a **finite** subset of the propositional KB
- Idea: For $n = 0$ to ∞ do
  create a propositional KB by instantiating with depth-n terms
  see if α is entailed by this KB
- Problem: works if α is entailed, loops if α is not entailed
- Theorem: Turing (1936), Church (1936), entailment in FOL is semidecidable

Problems with propositionalization

- Propositionalization seems to generate lots of irrelevant sentences.
- Example: from

$$\forall x \;\; \mathrm{King}(x) \wedge \mathrm{Greedy}(x) \;\Rightarrow\; \mathrm{Evil}(x)$$
$$\mathrm{King}(\mathrm{John})$$
$$\forall y \;\; \mathrm{Greedy}(y)$$
$$\mathrm{Brother}(\mathrm{Richard}, \mathrm{John})$$

  it seems obvious that $\mathrm{Evil}(\mathrm{John})$, ...
- ...but propositionalization produces lots of facts such as $\mathrm{Greedy}(\mathrm{Richard})$ that are irrelevant
- With p k-ary predicates and n constants, there are $p \cdot n^k$ instantiations
- With function symbols, it gets much much worse!

Unification

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(\mathsf{x})$ and $\mathsf{Greedy}(\mathsf{x})$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(\mathsf{y})$
- $\theta = \{\mathsf{x}/\mathsf{John}, \mathsf{y}/\mathsf{John}\}$ works
- $\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{John}, \mathsf{Jane})$	
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{y}, \mathsf{OJ})$	
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{y}, \mathsf{Mother}(\mathsf{y}))$	
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{x}, \mathsf{OJ})$	

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(\mathsf{x})$ and $\mathsf{Greedy}(\mathsf{x})$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(\mathsf{y})$
- $\theta = \{\mathsf{x}/\mathsf{John}, \mathsf{y}/\mathsf{John}\}$ works
- $\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
Knows(John, x)	Knows(John, Jane)	{x/Jane}
Knows(John, x)	Knows(y, OJ)	
Knows(John, x)	Knows(y, Mother(y))	
Knows(John, x)	Knows(x, OJ)	

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(\mathsf{x})$ and $\mathsf{Greedy}(\mathsf{x})$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(\mathsf{y})$
- $\theta = \{\mathsf{x}/\mathsf{John}, \mathsf{y}/\mathsf{John}\}$ works
- $\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{John}, \mathsf{Jane})$	$\{\mathsf{x}/\mathsf{Jane}\}$
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{y}, \mathsf{OJ})$	$\{\mathsf{x}/\mathsf{OJ}, \mathsf{y}/\mathsf{John}\}$
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{y}, \mathsf{Mother}(\mathsf{y}))$	
$\mathsf{Knows}(\mathsf{John}, \mathsf{x})$	$\mathsf{Knows}(\mathsf{x}, \mathsf{OJ})$	

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(x)$ and $\mathsf{Greedy}(x)$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(y)$
- $\theta = \{x/\mathsf{John}, y/\mathsf{John}\}$ works
- $\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
Knows(John, x)	Knows(John, Jane)	{x/Jane}
Knows(John, x)	Knows(y, OJ)	{x/OJ, y/John}
Knows(John, x)	Knows(y, Mother(y))	{y/John, x/Mother(John)}
Knows(John, x)	Knows(x, OJ)	

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(\mathsf{x})$ and $\mathsf{Greedy}(\mathsf{x})$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(\mathsf{y})$
- $\theta = \{\mathsf{x}/\mathsf{John}, \mathsf{y}/\mathsf{John}\}$ works
- $\text{UNIFY}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
Knows(John, x)	Knows(John, Jane)	{x/Jane}
Knows(John, x)	Knows(y, OJ)	{x/OJ, y/John}
Knows(John, x)	Knows(y, Mother(y))	{y/John, x/Mother(John)}
Knows(John, x)	Knows(x, OJ)	fail

Unification

- We can get the inference immediately if we can find a substitution θ such that $\mathsf{King}(x)$ and $\mathsf{Greedy}(x)$ match $\mathsf{King}(\mathsf{John})$ and $\mathsf{Greedy}(y)$
- $\theta = \{x/\mathsf{John}, y/\mathsf{John}\}$ works
- $\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
$\mathsf{Knows}(\mathsf{John}, x)$	$\mathsf{Knows}(\mathsf{John}, \mathsf{Jane})$	$\{x/\mathsf{Jane}\}$
$\mathsf{Knows}(\mathsf{John}, x)$	$\mathsf{Knows}(y, \mathsf{OJ})$	$\{x/\mathsf{OJ}, y/\mathsf{John}\}$
$\mathsf{Knows}(\mathsf{John}, x)$	$\mathsf{Knows}(y, \mathsf{Mother}(y))$	$\{y/\mathsf{John}, x/\mathsf{Mother}(\mathsf{John})\}$
$\mathsf{Knows}(\mathsf{John}, x)$	$\mathsf{Knows}(x, \mathsf{OJ})$	fail

- How to compute “unifiers” for two given terms/literals?