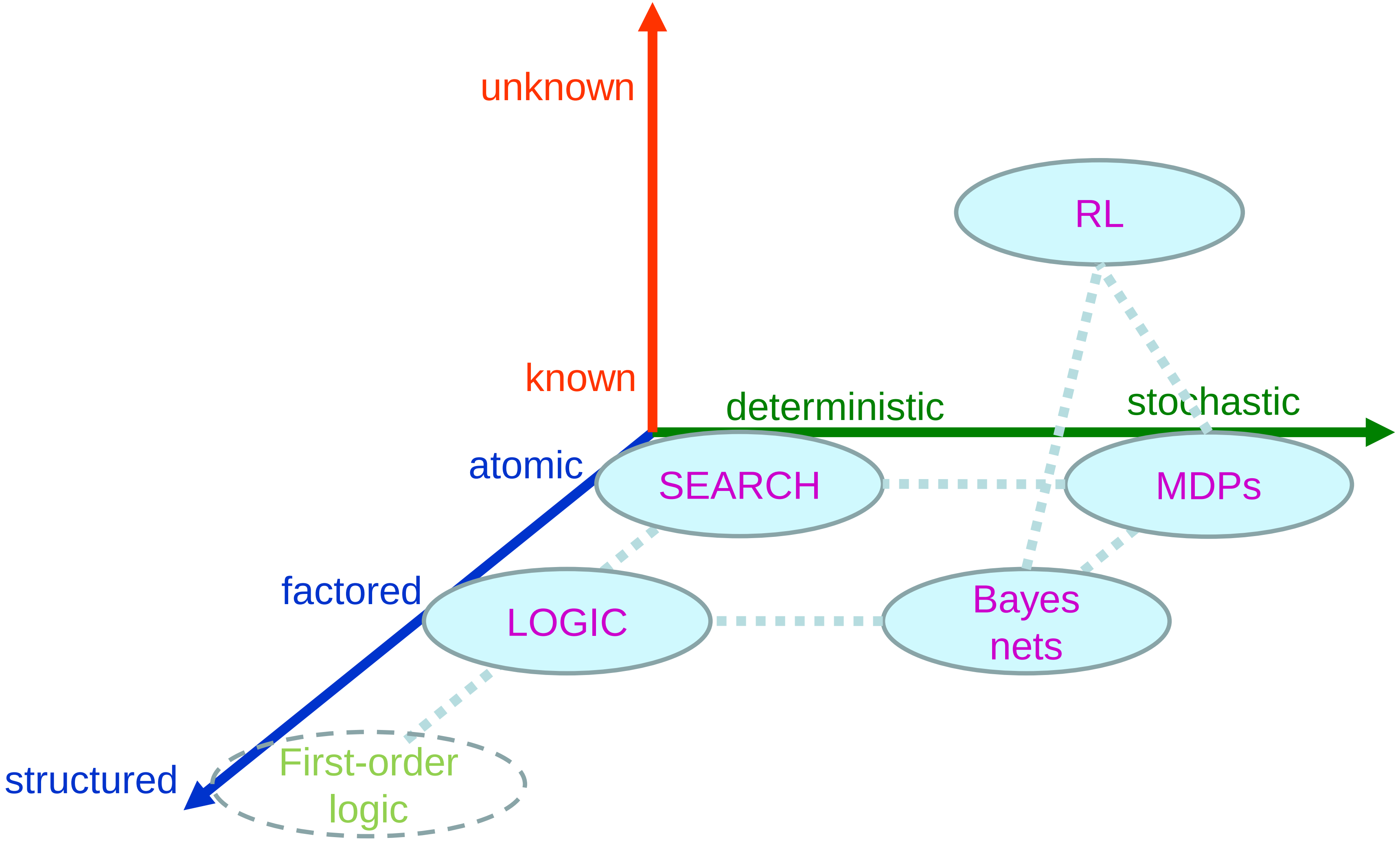
Artificial Intelligence

Introduction to Logic



Outline

1. Introduction to logic
   - Basic concepts of knowledge, logic, reasoning
   - Propositional logic: syntax and semantics
2. Propositional logic: inference
3. Agents using propositional logic
4. First-order logic

Knowledge Representation

Intended role of knowledge representation in AI is to reduce problems of intelligent action to search problems. *-- Ginsberg, 1993*

An Analogy between AI Problems and Programming

Programming

1. Devise an algorithm to solve the problem
2. Select a programming language in which the algorithm can be encoded
3. Capture the algorithm in a program
4. Run the program

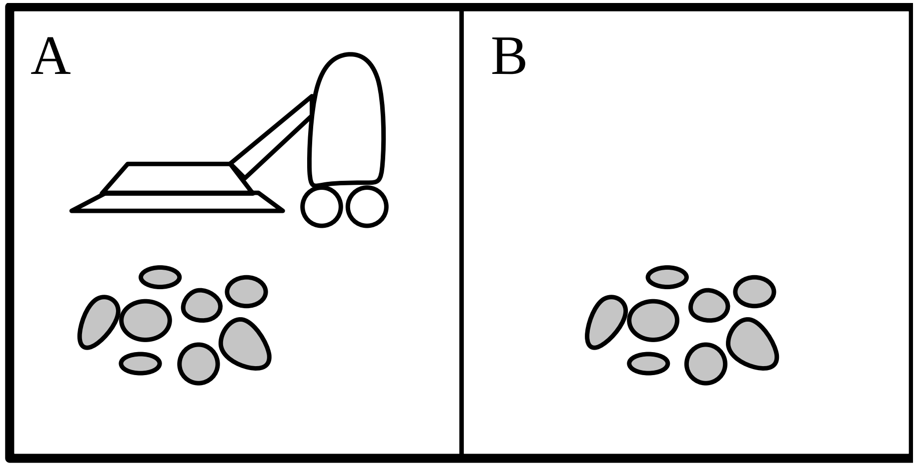
Artificial Intelligence

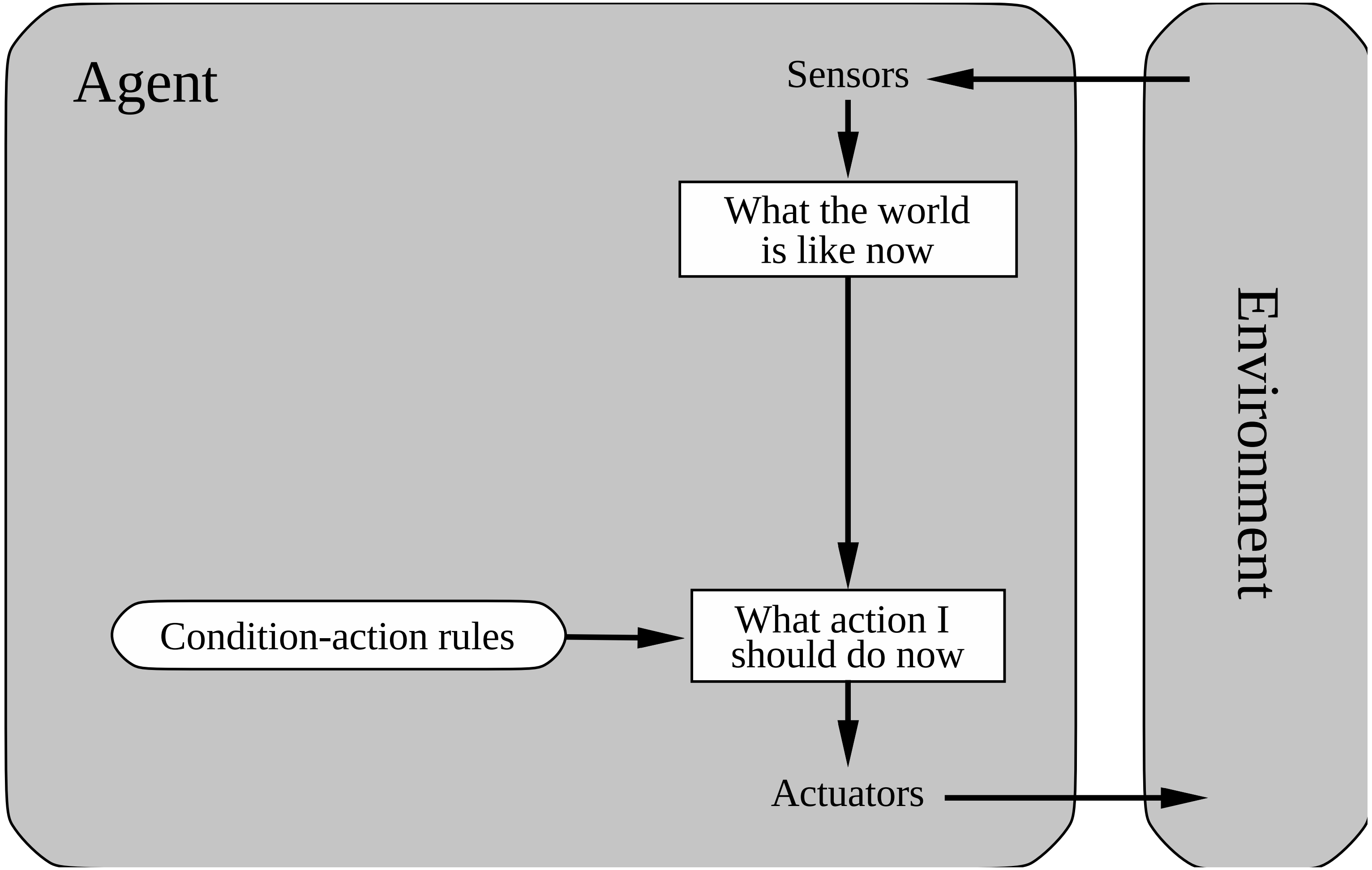
1. Identify the knowledge needed to solve the problem
2. Select a language in which the knowledge can be represented
3. Write down the knowledge in the language
4. Use the consequences of the knowledge to solve the problem

It is the final step that usually involves search

Simple Reflex Agent

Condition	Action
[A,Clean]	Right
[A,Dirty]	Suck
[B,Clean]	Left
[B,Dirty]	Suck





Simple Reflex Agent and Knowledge-Based Agent

- Recall the simple reflex agent

This agent keeps track of the state of the external world using its "update" function.

loop forever
  Input *percepts*
  state ← Update-State(*state*, *percept*)
  rule ← Rule-Match(*state, rules*)
  action ← Rule-Action[*rule*]
  Output *action*
  state ← Update-State(*state*, *action*)
end

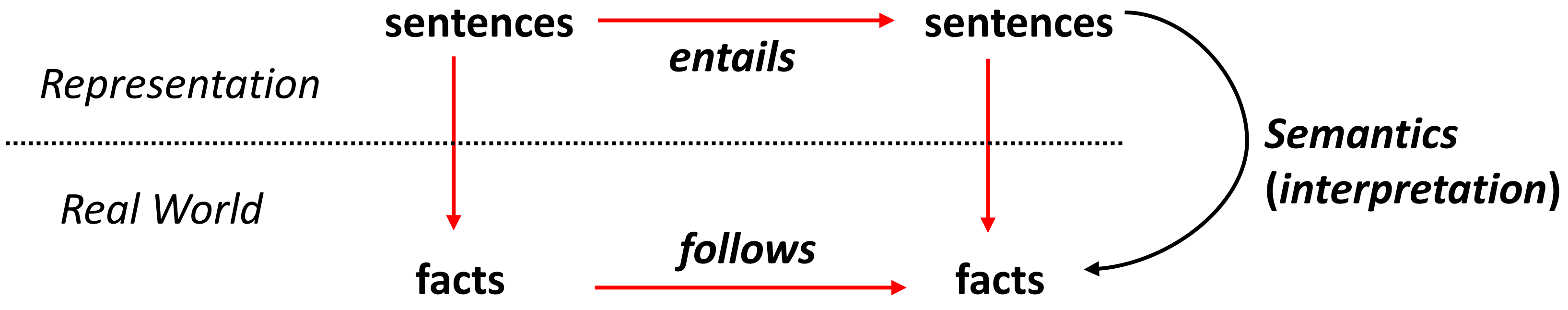
- A knowledge-based agent represents the state of the world using a set of sentences called a knowledge base.

loop forever
  Input *percepts*
  KB ← tell(*KB*, make-sentence(*percept*))
  action ← ask(*KB*, *action-query*)
  Output *action*
  KB ← tell(*KB*, make-sentence(*action*))
end

At each time instant, whatever the agent currently perceived is stated as a sentence, e.g. "I am hungry".

Knowledge Representation and Inference

- The knowledge representation language provides a declarative representation of real-world objects and their relationships



- Fundamental Requirements
  - The “ask” operation should give an answer that **follows** from the knowledge base (i.e., what has been told)
  - It is the **inference mechanism** that determines what follows from the knowledge base

Knowledge Representation and Inference

- Knowledge base = set of sentences in a formal language
- Declarative approach to building an agent (or other system):
  - ***Tell*** it what it needs to know (or have it ***Learn*** the knowledge)
  - Then it can ***Ask*** itself what to do—answers should follow from the KB
- For Example:
  - **Tell**: Father of Dipak is Ramesh
  - **Tell**: Jyoti is Dipak sister
  - **Tell**: Dipak father is the same as Jyoti sister father
  - **Ask**: Who is Jyoti father?

Knowledge Base

Inference Engine

Knowledge Representation

- Knowledge Representation Techniques
  - Logical Calculus
  - Production Rule Systems
  - Structured Models

Logical Calculus

- The goal is find a way to
  - State knowledge explicitly
  - Draw conclusions from the stated knowledge

- Logic
  - A "logic" is a mathematical notation (a language) for stating knowledge
  - The main alternative to logic is "natural language" i.e., English, Marathi, etc.
  - As in natural language the fundamental unit is a “sentence” (or a statement)
  - Syntax and Semantics
  - Logical inference

Logic

- Logic is defined by:
  - A set of sentences
    - A sentence is constructed from a set of primitives according to syntax rules.
  - A set of interpretations
    - An interpretation gives a semantic to primitives. It associates primitives with values.
  - The valuation (meaning) function *V*
    - Assigns a value (typically the truth value) to a given sentence under some interpretation V

$$: \text{sentence} \times \text{interpretation} \rightarrow \{\text{True}, \text{False}\}$$

Propositional Logic

- Knowledge representation in logical and mathematical form.
- Proposition is a declarative statement which is either true or false.
  - It is AI course (T)
  - The Sun rises from West (F)
  - 3+3= 7 (F)
  - 5 is a prime number (T)
  - It is raining today (T/F)

  - Are you going out somewhere?
  - 2+3
  - x + 5 = 3
  - She is very talented

- Proposition: {Op, Σ, Value}
  - Operator : Op= {¬,∧,∨,⇒, ⇔,(,)}
  - Set of symbols/variable/atom/formulas/sentence: Σ = {A, B, …}.
  - Truth Value: V = {*t*, *f* }

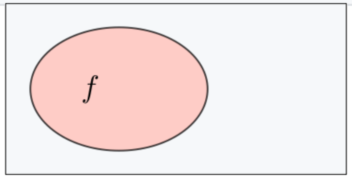
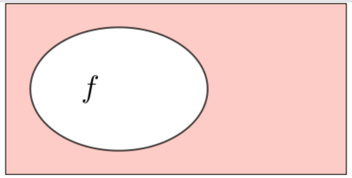
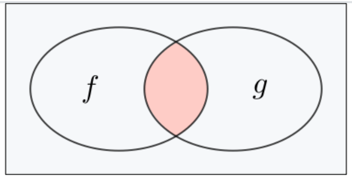
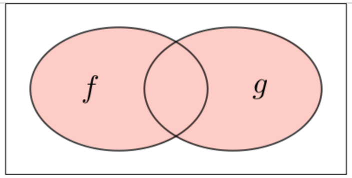
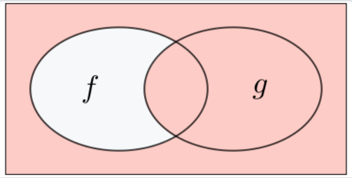
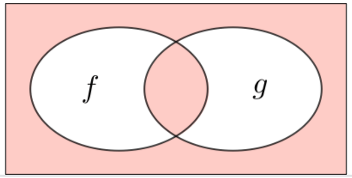
Syntax

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

Syntax

- Sentences in the propositional logic:
  - Atomic sentences (or formulas):
    - Constructed from constants $\{t, f\}$ and propositional symbols Σ
    - True, False are (atomic) sentences
    - All proposition symbols $a \in \Sigma$, are (atomic) sentence.
    - "*Light in the room is on*," "*It rains outside*" are (atomic) sentences
  - Composite sentences (or formulas):
    - Constructed from valid sentences via connectives
    - If A, B are sentences then
      - $\neg$ A, (A $\wedge$ B), (A $\vee$ B), (A $\Rightarrow$ B), (A $\Leftrightarrow$ B), (A $\vee$ B) $\wedge$ (A $\vee$ $\neg$ B)

Syntax

Name	Symbol	Meaning	Illustration
Affirmation	f	f	
Negation	$\neg f$	not f	
Conjunction	$f \land g$	f and g	
Disjunction	$f \lor g$	f or g	
Implication	$f \rightarrow g$	if f then g	
Biconditional	$f \leftrightarrow g$	f, that is to say g	

Semantics

- The semantics of propositional calculus is defined below.
  - Interpretation:
    - A mapping function $V : \Sigma \to \{t, f\}$, depending on whether the symbol is satisfied in the world
    - Assigns a truth value to every proposition variable.
    - Example: consider the two interpretation I_1 and I_2

Interpretation

- Suppose there are three sentence
  - *AI_is_fun*
  - *you_like_AI*
  - *you_are_happy*
- *Interpretation* I_1:
  - $V(AI\_is\_fun, I_1) = true$
  - $V(you\_like\_AI, I_1) = false$
  - $V(you\_are\_happy, I_1) = true$

- *AI_is_fun* in I_1 ⇨ *true*
- ¬ *AI_is_fun* in I_1 ⇨ *false*
- *you_like_AI* in I_1 ⇨ *false*
- ¬*you_like_AI* in I_1 ⇨ *true*
- *AI_is_fun* ∨ *you_like_AI* in I_1 ⇨ *true*
- *you_like_AI* → *AI_is_fun* in I_1 ⇨ *true*
- *AI_is_fun* → *you_like_AI* in I_1 ⇨ *false*
- *you_like_AI* ∧ *you_are_happy* → *AI_is_fun* in I_1 ⇨ *true*

Interpretation

- Suppose there are three sentence
  - *AI_is_fun*
  - *you_like_AI*
  - *you_are_happy*
- *Interpretation* I_2:
  - $V(AI\_is\_fun, I_2) = false$
  - $V(you\_like\_AI, I_2) = false$
  - $V(you\_are\_happy, I_2) = false$

- *AI_is_fun* in I_2 ⇨ *false*
- ¬ *AI_is_fun* in I_2 ⇨ *true*
- *you_like_AI* in I_2 ⇨ *false*
- ¬*you_like_AI* in I_2 ⇨ *true*
- *AI_is_fun* ∨ *you_like_AI* in I_2 ⇨ *false*
- *you_like_AI* → *AI_is_fun* in I_2 ⇨ *true*
- *AI_is_fun* → *you_like_AI* in I_2 ⇨ *true*
- *you_like_AI* ∧ *you_are_happy* → *AI_is_fun* in I_2 ⇨ *true*

Semantics

- For complex sentences
  - Determined using the standard rules of logic
  - Rows define all possible interpretations (worlds)

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*

Example

- Given a sentence
  - “If the humidity is high and the temperature is high, then one does not feel comfortable”
- Propositional Calculus:
  - “Humidity is high” $\wedge$ “Temperature is high” $\Rightarrow$ ~ “One feels comfortable”.

Example

- Given a sentence
  - "If the humidity is high and the temperature is high, then one does not feel comfortable"
- We have the following sentences:
  - *P*: "Humidity is high"
  - *Q*: "Temperature is high"
  - *R*: "One feels comfortable"
- Represented by: $((P \wedge Q) \Rightarrow (\sim R))$
- In which interpretation sentence is *false*

P	Q	R	$P \wedge Q$	$\sim R$	$P \wedge Q \rightarrow \sim R$
T	T	T	T	F	F
T	T	F	T	T	T
T	F	T	F	F	T
T	F	F	F	T	T
F	T	T	F	F	T
F	T	F	F	T	T
F	F	T	F	F	T
F	F	F	F	T	T

Properties: Logical equivalence

- Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other.

Definition of $\wedge$ $P \wedge \neg P \equiv False$ $P \wedge False \equiv False$ $P \wedge True \equiv P$	Idempotent Laws $p \vee p \equiv p$ $p \wedge p \equiv p$	DeMorgan's Laws $\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	Distributive Laws $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
Definition of $\vee$ $P \vee \neg P \equiv True$ $P \vee False \equiv P$ $P \vee True \equiv True$	Double Negation $\neg(\neg p) \equiv p$	Absorption Laws $p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Associative Laws $(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
	Commutative Laws $p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Implication Laws $p \rightarrow q \equiv \neg p \vee q$ $p \rightarrow q \equiv \neg q \rightarrow \neg p$	Biconditional Laws $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$ $p \leftrightarrow q \equiv \neg q \leftrightarrow \neg p$

Properties

- A proposition sentence:
  - **Valid** / **Tautology** iff sentence is true under every interpretation, e.g., $(P \vee \neg P)$
  - **Invalid** / **Contradiction** iff sentence is false under every interpretation, e.g., $(P \wedge \neg P)$
  - **Consistent** / **Satisfiable** iff sentence is true under at least one interpretation
  - **Inconsistent** / **Unsatisfiable** iff sentence is not made true under any interpretation

Interpretations and Models

- Every interpretation that satisfies a sentence is called a ***model*** of the sentence.
- A sentence is ***satisfiable*** if it has a model
  - There is at least one interpretation under which the sentence can evaluate to True.
- A sentence is ***valid*** if it is True in all interpretations
  - i.e., if its negation is not satisfiable (leads to contradiction)

P	Q	$P \vee Q$	$(P \vee Q) \wedge \neg Q$	$((P \vee Q) \wedge \neg Q) \Rightarrow P$
T	T	T	F	T
T	T	T	T	T
T	F	T	F	T
T	F	F	F	T

Knowledge Base

- A **knowledge base (KB)** is a set of propositions that the agent is given as being *true*. An element of the knowledge base is an *axiom*.
- A ***model*** of a set of propositions is an **interpretation in which all the propositions are** ***true*****.**

- **Example KB: {A $\vee$ C, B $\vee$ ¬C}**

- KB = **(A $\vee$ C)** $\wedge$ **(B $\vee$ ¬C)**

A	**B**	**C**	**A $\vee$ C**	**B $\vee$ ¬C**	**KB**
T	*T*	*T*	*T*	*T*	*T*
T	*T*	*F*	*T*	*T*	*T*
T	*F*	*T*	*T*	*F*	*F*
T	*F*	*F*	*T*	*T*	*T*
F	*T*	*T*	*T*	*T*	*T*
F	*T*	*F*	*F*	*T*	*F*
F	*F*	*T*	*T*	*F*	*F*
F	*F*	*F*	*F*	*T*	*F*

Propositional Inference: Entailment

- Given:
  - **KB** = set of propositional sentences
  - **α** = a propositional sentence / Query / Ask question

- Entailment: **KB $\models$ α** (read as "KB entails α" or "α logically follows from KB")
  - iff **α** is true in every model of **KB**

    OR

  - iff every interpretation that makes all sentences in **KB** *true* makes **α** also *true*.

    OR

  - iff every model of **KB** is also a model of **α**

Propositional Inference: Entailment

- Given:
  - **KB** = set of propositional sentences = $\{A \vee C, B \vee \neg C\}$
  - **α** = a propositional sentence = $A \vee B$

- Entailment: **KB $\models$ α**

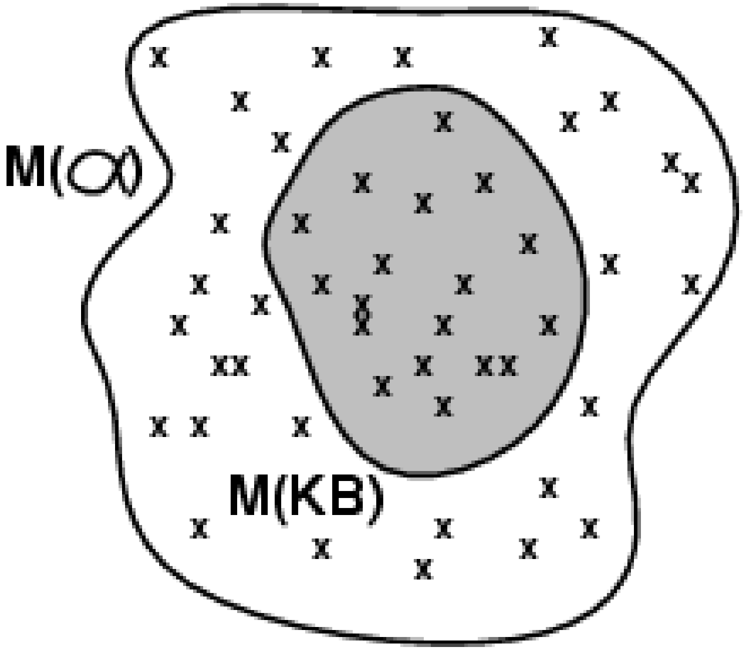
That is, no interpretation exists in which **KB** is true and **α** is false.

A	B	C	$A \vee C$	$B \vee \neg C$	KB	α
T	*T*	*T*	*T*	*T*	*T*	*T*
T	*T*	*F*	*T*	*T*	*T*	*T*
T	*F*	*T*	*T*	*F*	*F*	*T*
T	*F*	*F*	*T*	*T*	*T*	*T*
F	*T*	*T*	*T*	*T*	*T*	*T*
F	*T*	*F*	*F*	*T*	*F*	*T*
F	*F*	*T*	*T*	*F*	*F*	*F*
F	*F*	*F*	*F*	*T*	*F*	*F*

Model

- Semantics in Logic is in terms of Models: formally structured worlds with respect to which truth can be evaluated
  - A model *m* denotes an assignment of binary weights to propositional symbols.

- If a Sentence ***α*** is true in model ***m***
  - ***m*** satisfies ***α*** (***m*** is model of ***α***)
  - ***M***(**α**): Set of all models of **α**



Inference and Entailment

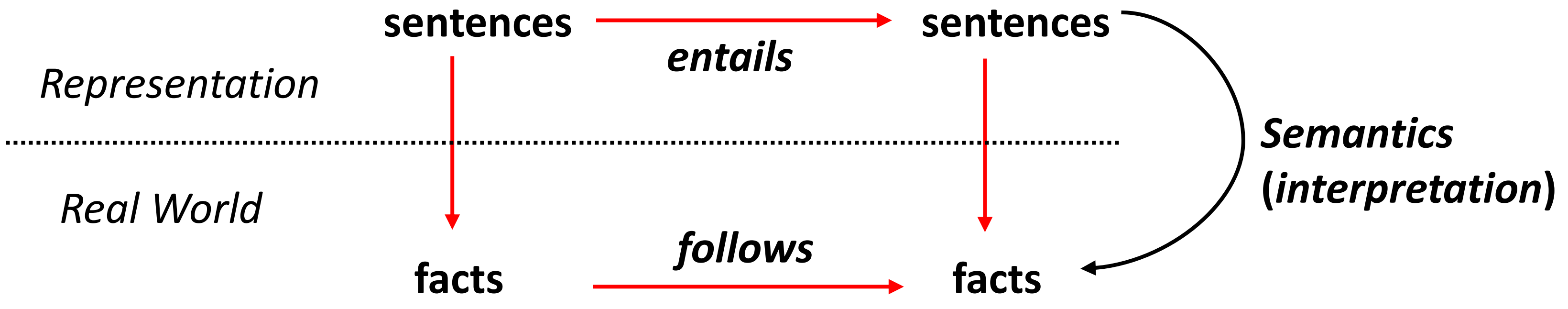
$$
\begin{tikzcd}
KB \arrow[r, "\vdash", "\text{derivation}"'] \arrow[d, "\text{interpretation}"'] & Q \arrow[d, "\text{interpretation}"] \\
\mathrm{Mod}(KB) \arrow[r, "\models", "\text{entailment}"'] & \mathrm{Mod}(Q)
\end{tikzcd}
$$

syntactic level
(formula)

semantic level
(interpretation)

Inference and Entailment

- **Entailment** reflects the relation of one fact in the world following from the others according to logic



- Knowledge base **KB** entails sentence **α** iff **α** is true in all worlds where **KB** is true

Inference Procedures

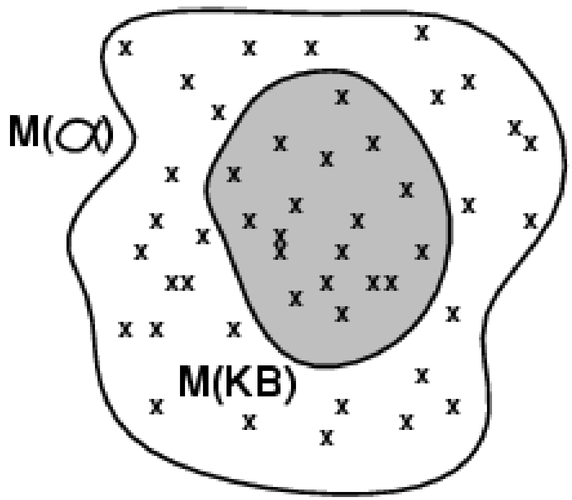
- Inference is a process by which conclusions are reached.
  - We want to implement the inference process on a computer !!

$$KB \vdash_i \alpha$$

- Sentence **α** can be inferred/derived/deduct from ***KB*** by procedure ***i***
  - ***i*** derived **α** from **KB**
- Algorithmic procedure that manipulate sentences in the input **KB** to produce **α** as an output

Inference Procedures Properties

- ***Soundness***: *i* is sound if whenever ***KB*** $\vdash_i$ **α**, then ***KB*** $\models$ **α**
  - *No wrong inferences but maybe not all true statements can be derived*
  - Derives only entailed sentences
- ***Completeness***: *i* is complete if whenever ***KB*** $\models$ **α**, then ***KB*** $\vdash_i$ **α**
  - *All true sentences can be derived, but maybe some wrong extra ones as well*
  - Derive any sentence that is entailed



A	B	C	A ∨ C	B ∨ ¬C	KB	α
T	*T*	*T*	*T*	*T*	*T*	*T*
T	*T*	*F*	*T*	*T*	*T*	*T*
T	*F*	*T*	*T*	*F*	*F*	*T*
T	*F*	*F*	*T*	*T*	*T*	*T*
F	*T*	*T*	*T*	*T*	*T*	*T*
F	*T*	*F*	*F*	*T*	*F*	*T*
F	*F*	*T*	*T*	*F*	*F*	*F*
F	*F*	*F*	*F*	*T*	*F*	*F*

Logical Inference Problem Formulation

- Given:
  - KB = set of propositional sentences
  - α = a propositional sentence / Query / Ask question
- Does a KB semantically entail ? **KB $\models \alpha$?**
- **Question:** Is there a procedure (program) that can decide this problem in a finite number of steps?
- **Answer:** Yes. Logical inference problem for the propositional logic is decidable.

Inference: proofs

- How we design the sound and complete procedure that answers:

$$KB \models \alpha \quad ?$$

- Method 1: ***model-checking***
  - For every possible world, if KB is true make sure that is α true too
  - OK for propositional logic (finitely many worlds); not easy for first-order logic
- Method 2: ***theorem-proving***
  - Search for a sequence of proof steps (applications of ***inference rules***) leading from KB to α
  - E.g., from $P \wedge (P \Rightarrow Q)$, infer Q by ***Modus Ponens***

Inference: Model Checking

- Given:
  - **KB** = set of propositional sentences = {A ∨ C, B ∨ ¬C}
  - **α** = a propositional sentence = A ∨ B

- Entailment: **KB ╞ α**

A	B	C	A ∨ C	B ∨ ¬C	KB	α
T	*T*	*T*	*T*	*T*	*T*	*T*
T	*T*	*F*	*T*	*T*	*T*	*T*
T	*F*	*T*	*T*	*F*	*F*	*T*
T	*F*	*F*	*T*	*T*	*T*	*T*
F	*T*	*T*	*T*	*T*	*T*	*T*
F	*T*	*F*	*F*	*T*	*F*	*T*
F	*F*	*T*	*T*	*F*	*F*	*F*
F	*F*	*F*	*F*	*T*	*F*	*F*

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

  $symbols \leftarrow$ a list of the proposition symbols in KB and α
  return TT-CHECK-ALL($KB, \alpha, symbols, \{\,\}$)

---

function TT-CHECK-ALL($KB, \alpha, symbols, model$) **returns** *true* or *false*
  if EMPTY?($symbols$) **then**
    if PL-TRUE?($KB, model$) **then return** PL-TRUE?($\alpha, model$)
    else return *true* // *when KB is false, always return true*
  else do
    $P \leftarrow$ FIRST($symbols$)
    $rest \leftarrow$ REST($symbols$)
    return (TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = true\}$)
      and
      TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = false\}$))

- PL-True returns true if the sentence holds within the model
- For *n* symbols, time complexity is $O(2^n)$, space complexity is $O(n)$

Inference: Model Checking

- Problem given:
  - KB = set of propositional sentences
  - α = a propositional sentence / Query / Ask question
- KB $\models$ α ?
  - We need to *check all possible interpretations* for which the KB is true (models of KB) whether α is true for each of them
- Truth table:
  - Enumerates truth values of sentences for all possible interpretations (assignments of True/False values to propositional symbols)
- Limitation:
  - The method of truth tables is a very inefficient since we need to evaluate a formula for each of 2^n possible models (interpretations), where n is the number of distinct atoms in the formula.

Inference Rule for Logic

- Modus ponens

$$\frac{p \Rightarrow q, \quad p}{q} \quad \begin{matrix} \leftarrow \text{ premise} \\ \leftarrow \text{ conclusion} \end{matrix}$$

- If both sentences in the premise are true then conclusion is true.
- The modus ponens inference rule is sound.
  - We can prove this through the truth table.

p	q	$p \Rightarrow q$	$p \Rightarrow q \wedge p$
T	T	T	T
T	F	F	F
F	T	T	F
F	F	T	F

Rule of Inference

- Generating the conclusions from evidence and facts is termed as Inference.

Simplification	Modus Ponens	Modus Tollens	Hypothetical Syllogism
$p \wedge q$	p $p \rightarrow q$	$\neg q$ $p \rightarrow q$	$p \rightarrow q$ $q \rightarrow r$
Therefore, p	Therefore, q	Therefore, $\neg p$	Therefore, $p \rightarrow r$
Conjunction	**Addition**	**Resolution**	**Disjunctive Syllogism**
p q	p	$p \vee q$ $\neg p \vee r$	$p \vee q$ $\neg p$
Therefore, $p \wedge q$	Therefore, $p \vee q$	Therefore, $q \vee r$	Therefore, q
Universal Instantiation	**Universal Generalization**	**Existential Instantiation**	**Existential Generalization**
$\forall x P(x)$	$P(c)$	$\exists x P(x)$	$P(c)$
Therefore, $P(c)$	Therefore, $\forall x P(x)$	Therefore, $P(c)$	Therefore, $\exists x P(x)$

Theorem Proving: Inference Rule

- Checks only entries for which KB is True.
- Represent sound inference patterns repeated in inferences
- Can be used to generate new (sound) sentences from the existing ones

- (MP) Modes Ponens $\dfrac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$
- (AI) And-Introduction $\dfrac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$
- (OI) Or-Introduction $\dfrac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \cdots \vee \alpha_n}$
- (AE) And-Elimination $\dfrac{\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_n}{\alpha_i}$
- (NE) Negation-Elimination $\dfrac{\neg\neg\alpha}{\alpha}$
- (UR) Unit Resolution $\dfrac{\alpha \vee \beta, \quad \neg\beta}{\alpha}$
- (R) General Resolution $\dfrac{\alpha \vee \beta, \quad \neg\beta \vee \gamma}{\alpha \vee \gamma}$

Using Inference Rules

- Given: KB = $\{(A \lor B) \Rightarrow C \land (B \lor \neg D)), A \land D\}$
- Prove: $(B \lor D) \land C$

1.	$(A \lor B) \Rightarrow C \land (B \lor \neg D)$	
2.	$A \land D$	
3.	$(A \lor B) \Rightarrow C$	: Using 1 and AE
4.	A	: Using 2 and AE
5.	$B \lor \neg D$	: Using 1 and AE
6.	D	: Using 2 and AE
7.	B	: Using 5, 6 and UR
8.	$A \lor B$	: Using 4, 7 and OI
9.	C	: Using 3, 8 and MP
10.	$B \lor D$	: Using 6, 7 and OI
11.	$(B \lor D) \land C$	: Using 9, 10 and AI

Note: in each of the steps in the proof we could have applied other rules to derive new sentences, thus the inference problem is really a search problem:

Initial state = ***KB***
Goal state = **conclusion to be proved**
Operators = **?**

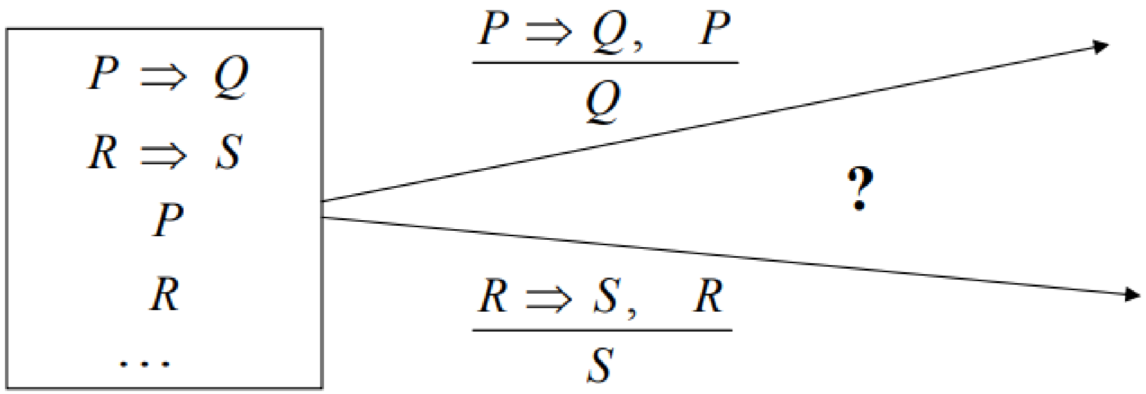
Example

- Given: KG = $\{P \wedge Q, P \Rightarrow R, (Q \wedge R) \Rightarrow S\}$
- Prove: KG $\models$ S

1. $P \wedge Q$
2. $P \Rightarrow R$
3. $(Q \wedge R) \Rightarrow S$
4. P : Using 1 and AE
5. R : Using 2, 4 and MP
6. Q : Using 1, and AE
7. $(Q \wedge R)$: Using 5, 6 and AI
8. S : Using 3, 7 and MP

Logic inferences and search

- To show that sentence holds for a KB
  - we may need to apply a number of sound inference rules
- Problem: many possible rules can be applied in the next step



- This is an instance of a search problem:
- Truth table method (from the search perspective)
  - blind enumeration and checking

Propositional Definite Clauses

- An *atomic proposition* or *atom* (or *facts*) is the same as in propositional calculus.
- A body is an atom or a conjunction of atoms. $a \wedge b$
- A definite clause is either an ***atom*** a, called an atomic clause, or of the form $a \Rightarrow b$, called a ***rule***,
  - where b, the head, is an atom and a is a body.
- A knowledge base is a set of definite clauses.
- If "$a_1 \wedge \ldots \wedge a_m \Rightarrow b$" is a *definite clause* in the knowledge base,
  - each $KB \models a_i$, then $KB \models b$.

Bottom-up Proof Procedure: Forward Chaining

```
procedure DCDeductionBU(KB)
    Inputs
        KB: a set of definite clauses
    Output
        Set of all atoms that are logical consequences of KB
    Local
        C is a set of atoms
    C := {}
    repeat
        select "h ← b1 ∧ ... ∧ bm" in KB where bi ∈ C for all i, and h ∉ C
        C := C ∪ {h}
    until no more definite clauses can be selected
    return C
```

If "$\boldsymbol{h} \leftarrow \boldsymbol{b}_1 \wedge \ldots \wedge \boldsymbol{b}_m$" is a definite clause in the knowledge base,

- each $\boldsymbol{b}_i$ has been derived,
- then ***h*** can be derived.

Example

KB

$a \leftarrow b \wedge c.$

$b \leftarrow d \wedge e.$

$b \leftarrow g \wedge e.$

$c \leftarrow e.$

$d.$

$e.$

$f \leftarrow a \wedge g.$

KB $\models a$

$\{\}$

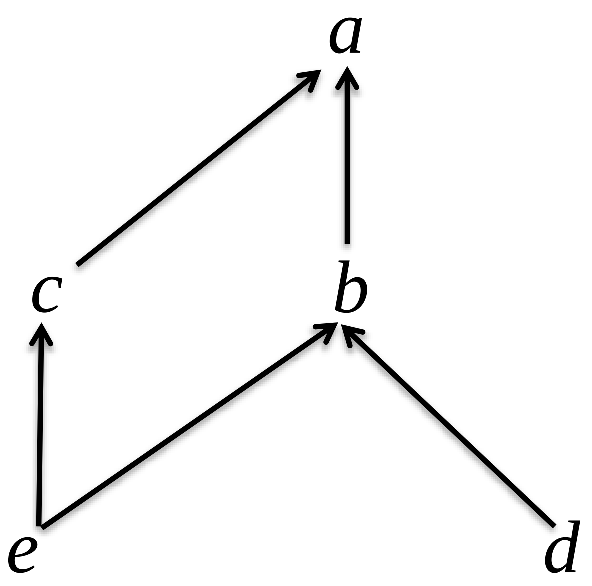
$\{d\}$

$\{e,d\}$

$\{c,e,d\}$

$\{b,c,e,d\}$

$\{a,b,c,e,d\}.$



Top-Down Proof Procedure: Backward Chaining

non-deterministic procedure *DCDeductionTD*(KB,Query)
    Inputs
        KB: a set definite clauses
        Query: a set of atoms to prove
    Output
        yes if $KB \models Query$ and the procedure fails otherwise
    Local
        G is a set of atoms
    $G := Query$
    repeat
        select an atom a in G
        choose definite clause "$a \leftarrow B$" in KB with a as head
        replace a with B in G
    until $G = \{\}$
    return *yes*

Example

KB

$a \leftarrow b \wedge c.$

$b \leftarrow d \wedge e.$

$b \leftarrow g \wedge e.$

$c \leftarrow e.$

$d.$

$e.$

$f \leftarrow a \wedge g.$

KB $\models a$

$yes \leftarrow a.$

$yes \leftarrow b \wedge c.$

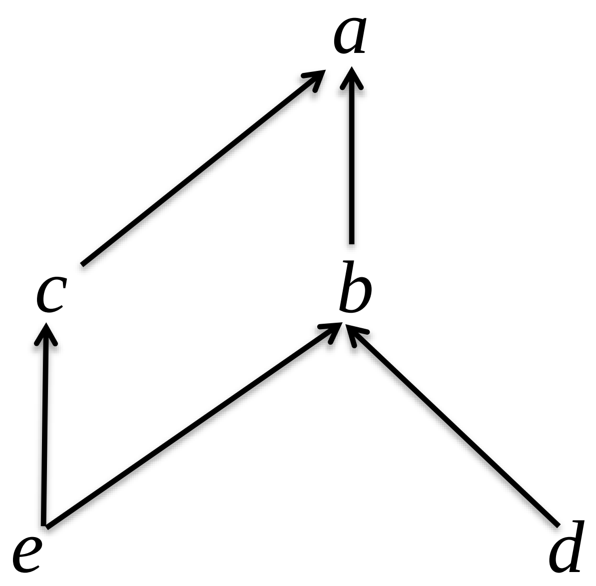
$yes \leftarrow d \wedge e \wedge c.$

$yes \leftarrow e \wedge c.$

$yes \leftarrow c.$

$yes \leftarrow e.$

$yes \leftarrow .$



Example

KB

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

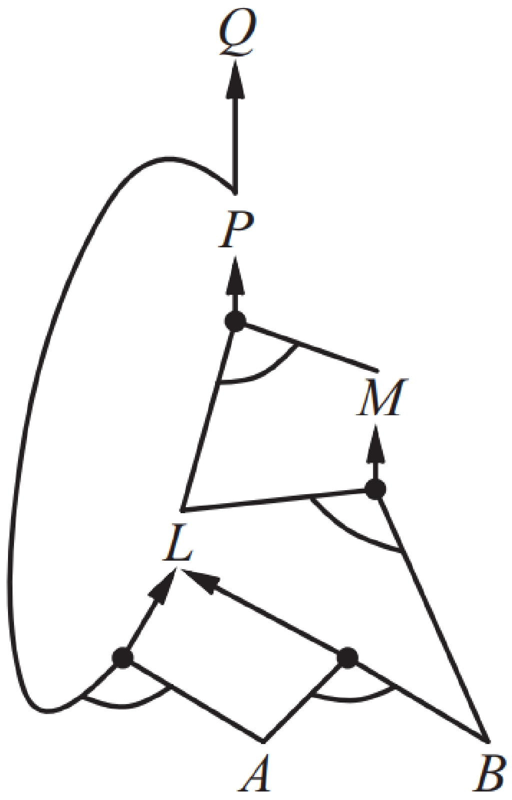
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

$\mathbf{KB} \models Q$



Example

- Assume the KB with the following rules and facts
  - **KB**: R1: $A \land B \Rightarrow C$ $\boldsymbol{\alpha} = E$?

    R2: $C \land D \Rightarrow E$

    R3: $C \land F \Rightarrow G$

    F1: A

    F2: B

    F3: D

Example

- Given: KB
  - sam_is_happy.
  - ai_is_fun.
  - worms_live_underground.
  - Night_time.
  - bird_eats_apple.
  - bird_eats_apple $\Rightarrow$ apple_is_eaten .
  - sam_is_in_room $\wedge$ night_time $\Rightarrow$ switch_1_is_up .
- Prove:
  - KB $\models$ bird_eats_apple.
  - KB $\models$ apple_is_eaten.
  - KB $\models$ switch_1_is_up

> - Until we consider computers with perception and the ability to act in the world
> - The computer does not know the meaning of the symbols.
> - It is the human that gives the symbols meaning.
> - All the computer knows about the world is what it is told about the world.

Proof Procedure

- Bottom-Up proof Approach
  - Forword Chaining
- Work **forward** from **KB** to query **α:**
  - Fire any rule whose premises are satisfied in the KB
  - Add its conclusion to the KB, until query **α** is found
- Data-driven, automatic, unconscious processing,
  - e.g., object recognition, routine decisions

- Top-Down proof Approach
  - Backward Chaining
- Work **backwards** from the query **α** to **KB**:
  - Check if **α** is known already, or
  - Prove by BC all premises of some rule concluding **α**
- Goal-driven, appropriate for problem-solving
  - e.g., Where are my keys? How do I get into a PhD program?

- *Complexity of BC can be much less than linear in size of KB*

Sentence transformations

- Propositional logic:
  - A sentence may include connectives: $\neg \;\; \vee \;\; \wedge \;\; \Rightarrow \;\; \Leftrightarrow$
  - A sentence may consist of multiple nested sentences
- Do we need all operators to represent a complex sentence?
  - No. We can rewrite a sentence in PL using an equivalent sentence with just operators
- Is it possible to limit the depth of the sentence structure?
  - Yes. Example: Normal forms.

$\neg \vee \wedge \Rightarrow \Leftrightarrow$

Normal forms

- Normal forms: This can simplify the inferences.
- Conjunctive normal form (CNF)
  - Conjunction of clauses (*clauses include disjunctions of literals*)
  - $(A \lor B) \land (\neg A \lor \neg C \lor D)$
- Disjunctive normal form (DNF)
  - Disjunction of terms (*terms include conjunction of literals*)
  - $(A \land \neg B) \lor (\neg A \land C) \lor (C \land \neg D)$

Conversion to CNF

Given: $A \Leftrightarrow (B \vee C)$

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

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

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

   $(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A)$

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

   $(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A)$

4. Apply **distributivity law** ($\wedge$ over $\vee$) and flatten:

   $(\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A)$

Properties: Logical equivalence

- Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other.

Definition of $\wedge$ $P \wedge \neg P \equiv False$ $P \wedge False \equiv False$ $P \wedge True \equiv P$	Idempotent Laws $p \vee p \equiv p$ $p \wedge p \equiv p$	DeMorgan's Laws $\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	Distributive Laws $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
Definition of $\vee$ $P \vee \neg P \equiv True$ $P \vee False \equiv P$ $P \vee True \equiv True$	Double Negation $\neg(\neg p) \equiv p$	Absorption Laws $p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Associative Laws $(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
	Commutative Laws $p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Implication Laws $p \rightarrow q \equiv \neg p \vee q$ $p \rightarrow q \equiv \neg q \rightarrow \neg p$	Biconditional Laws $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$ $p \leftrightarrow q \equiv \neg q \leftrightarrow \neg p$

Example

- Given: $\neg(A \Rightarrow B) \lor (C \Rightarrow A)$
  - $\neg(\neg A \lor B) \lor (\neg C \lor A)$
  - $(A \land \neg B) \lor (\neg C \lor A)$
  - $(A \lor \neg C \lor A) \land (\neg B \lor \neg C \lor A)$
  - $(A \lor \neg C) \land (\neg B \lor \neg C \lor A)$

- Given: $(P \Rightarrow (Q \Rightarrow R)) \Rightarrow (P \Rightarrow (R \Rightarrow Q))$
  - $\neg(\neg P \lor \neg Q \lor R) \lor (\neg P \lor \neg R \lor Q)$
  - $(P \land Q \land \neg R) \lor (\neg P \lor \neg R \lor Q)$
  - $(P \lor \neg P \lor \neg R \lor Q) \land (Q \lor \neg P \lor \neg R \lor Q) \land (\neg R \lor \neg P \lor \neg R \lor Q)$

Satisfiability (SAT) problem

- Determine whether a sentence in the conjunctive normal form (CNF) is satisfiable (i.e. can evaluate to true)
  - $(P \vee Q \vee \neg R) \wedge (\neg P \vee \neg R \vee S) \wedge (\neg P \vee Q \vee \neg T)$

- It is an instance of a constraint satisfaction problem (CSP):
  - Variables
    - Propositional symbols (P, R, T, S)
    - Values: True, False
  - Constraints
    - Every conjunct must evaluate to *true*, at least one of the literals must evaluate to *true*

Solving SAT

- Methods:
  - Backtracking: equivalent to the depth first search
    - Pick the variable, then pick its value (*T* or *F*)
    - Continue till all variables are assigned or till the partial assignment makes the sentence False
  - Iterative optimization methods:
    - Start from an arbitrary assignment of *T*, *F* values to symbols
    - Flip *T*, *F* values for one variable
    - Heuristics: prefer assignments that make more clauses true
    - Walk-SAT, simulated-annealing procedures

Inference problem and satisfiability

- Logical inference problem:
  - For all interpretations for which KB is true α is also true?
- Satisfiability:
  - Is there is some assignment (interpretation) under which a sentence evaluates to true
- Connection:
  - $KB \models \alpha$ if and only if $(KB \land \neg \alpha)$ is unsatisfiable
- Consequences:
  - Inference problem is NP-complete
  - Programs for solving the SAT problem can be used to solve the inference problem

Inferences with the CNF

- Is there an inference rule that is sufficient to support all inferences for the KB in the CNF form?

- *A rule that seems to fit very well the CNF form* is the ***resolution rule***.

Resolution rule

- Resolution rule
- *Sound inference rule* that works for the KB in the CNF form

- KB
- α

$$\frac{A \vee B, \quad \neg B \vee C}{A \vee C}$$

A	B	C	A ∨ B	¬B ∨ C	KB	α
T	*T*	*T*	*T*	*T*	*T*	*T*
T	*T*	*F*	*T*	*F*	*F*	*T*
T	*F*	*T*	*T*	*T*	*T*	*T*
T	*F*	*F*	*T*	*T*	*T*	*T*
F	*T*	*T*	*T*	*T*	*T*	*T*
F	*T*	*F*	*T*	*F*	*F*	*F*
F	*F*	*T*	*F*	*T*	*F*	*T*
F	*F*	*F*	*F*	*T*	*F*	*F*

Inferences with the resolution rule

- Resolution rule is sound
- Is it complete?
  - Is it possible to use it such that we start from the KB
  - Then prove the new facts and eventually prove the theorem if it is entailed?
- Not always = incomplete
  - Repeated application of the resolution rule to a KB in CNF may fail to derive new valid sentences
  - Example: We know: $(A \wedge B)$ We want to show: $(A \vee B)$
  - Resolution rule fails to derive it (incomplete ??)

Satisfiability inferences with the resolution rule

- Conversion to SAT:
  - Proof by contradiction
    - Disproving: (KB $\wedge \neg\alpha$)
    - Proves the entailment Important: KB $\models \alpha$
- Important
  - Resolution rule is sufficient to determine satisfiability/unsatisfiability of

$$(KB \wedge \neg \alpha)$$

  - For any KB and Query in the CNF
  - We say the resolution rule is refutation complete.

Resolution algorithm

- Algorithm:
  - Convert KB to the CNF form;
  - Apply iteratively the resolution rule starting from

    (KB ∧ ¬α)   (in CNF form)
- Stop when:
  - Contradiction (empty clause) is reached:
    - A ,¬ A → Ø
    - proves entailment.
  - No more new sentences can be derived
    - disproves it.

```
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 pair of clauses 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
```

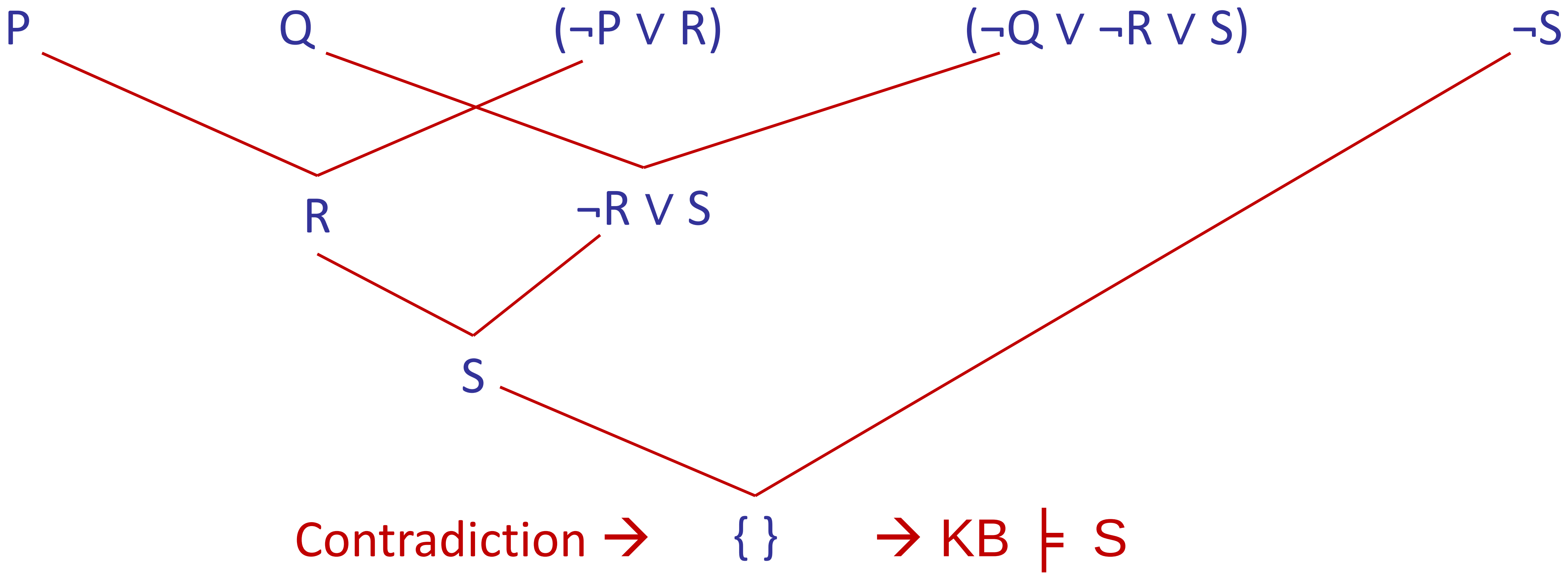
Example

- KB: $\{(P \land Q), (P \Rightarrow R), [(Q \land R) \Rightarrow S]\}$ Query/Theorem/Sentence: S

- Step 1. convert KB to CNF:
  - $(P \land Q) \equiv P \land Q$
  - $(P \Rightarrow R) \equiv (\neg P \lor R)$
  - $[(Q \land R) \Rightarrow S] \equiv (\neg Q \lor \neg R \lor S)$
  - KB = $P \quad Q \quad (\neg P \lor R) \quad (\neg Q \lor \neg R \lor S)$
- Step 2. Negate the theorem to prove it via refutation
  - S ---> $\neg S$
- Step 3. Run resolution on the set of clauses
  - $P \quad Q \quad (\neg P \lor R) \quad (\neg Q \lor \neg R \lor S) \quad \neg S$

Example

- KB: {(P ∧ Q) , (P ⇒ R), [(Q ∧ R) ⇒ S]}   Theorem/Sentence: S



Example

- **Knowledge Base**:
  1. The humidity is high or the sky is cloudy.
  2. If the sky is cloudy, then it will rain.
  3. If the humidity is high, then it is hot.
  4. It is not hot.
- **Query:** It will rain.

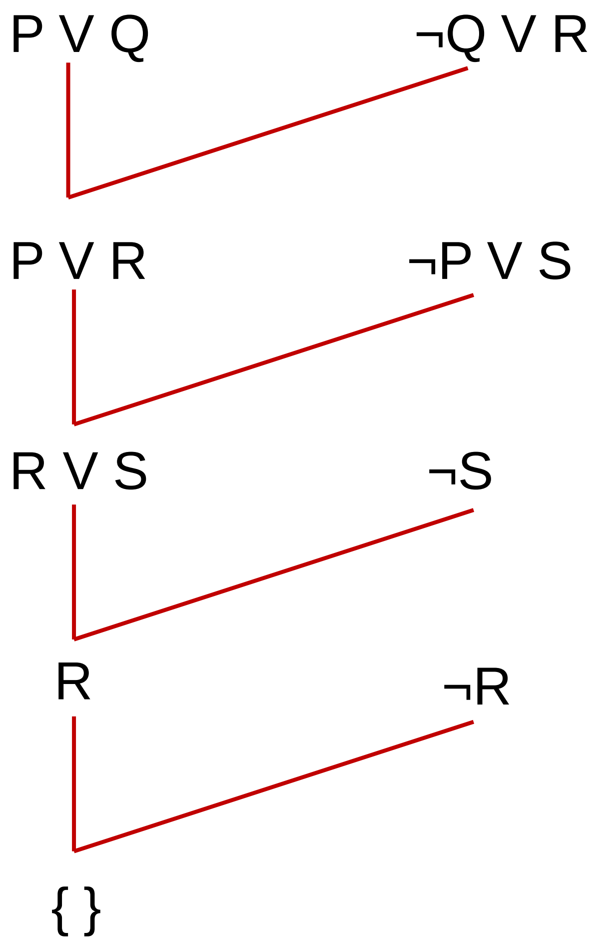
Formulate the Problem

- P: Humidity is high
- Q: Sky is cloudy
- R: It will rain
- S: It is hot

		CNF
1. The humidity is high or the sky is cloudy :	$P \vee Q$	$P \vee Q$
2. If the sky is cloudy, then it will rain :	$Q \Rightarrow R$	$(\neg Q \vee R)$
3. If the humidity is high, then it is hot :	$P \Rightarrow S$	$(\neg P \vee S)$
4. It is not hot :	$\neg S$	$\neg S$

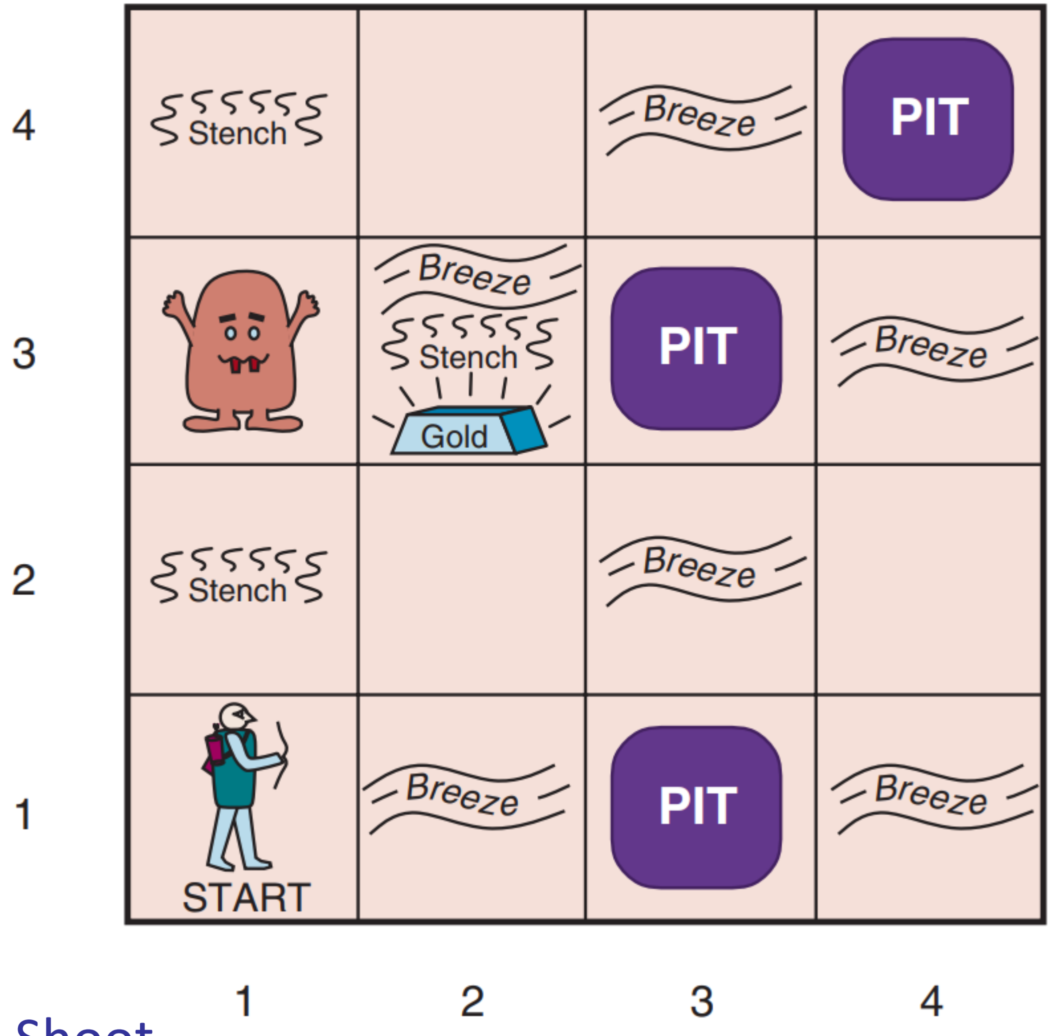
Resolution

- KB
  - P V Q
  - (¬Q V R)
  - (¬P V S)
  - ¬S
- Query :
  - ¬R



Wumpus World PEAS description

- Performance measure
  - gold +1000, death -1000
  - -1 per step, -10 for using the arrow
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square
  - Releasing drops the gold in same square
- Sensors: Stench, Breeze, Glitter, Bump, Scream
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot



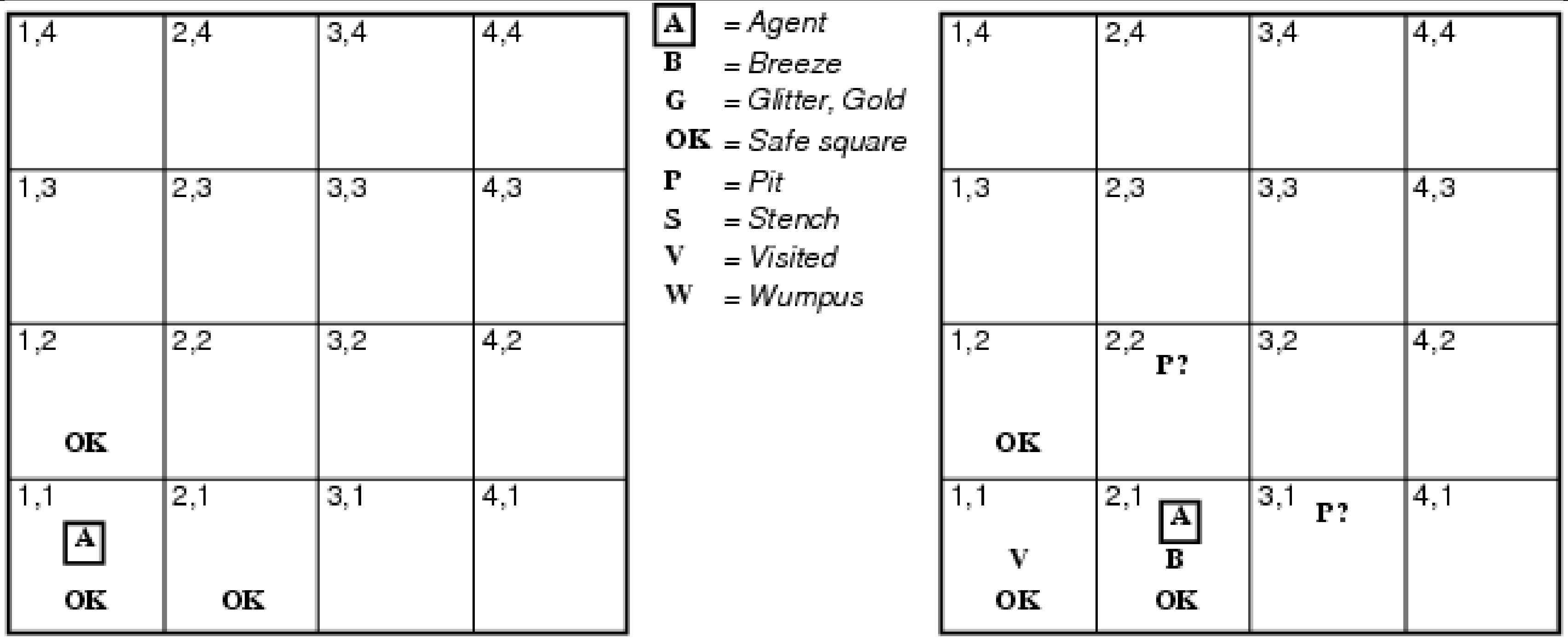
Wumpus world characterization

- Fully Observable
- Deterministic
- Episodic
- Static
- Discrete
- Single-agent?

- No – only local perception
- Yes – outcomes exactly specified
- No – sequential at the level of actions
- Yes – Wumpus and Pits do not move
- Yes
- Yes – Wumpus is essentially a natural feature

Exploring the Wumpus World

The KB initially contains the rules of the environment.



(a) (b)

[1,1] : [*none,none,none,none,none*],
Move to safe cell e.g. [2,1]

[2,1] :[none,Breeze,none,none,none]
There is a pit in [2,2] or [3,1]
Return to [1,1] to try next safe cell

Exploring the Wumpus World

(a) (b)

[1,2] : [*Stench,none,none,none,none*]

Wumpus is in [1,3] or [2,2]

YET … not in [1,1]

Thus … not in [2,2] or stench would have been detected in [2,1]

Thus … wumpus is in [1,3]

Thus … [2,2] is safe because of lack of breeze in [1,2]

Thus … pit in [3,1]

Move to next safe cell [2,2]

Exploring the Wumpus World

(a) (b)

[2,2] : [*none,none,none,none,none*]
Move to next safe cell [2,3]

Exploring the Wumpus World

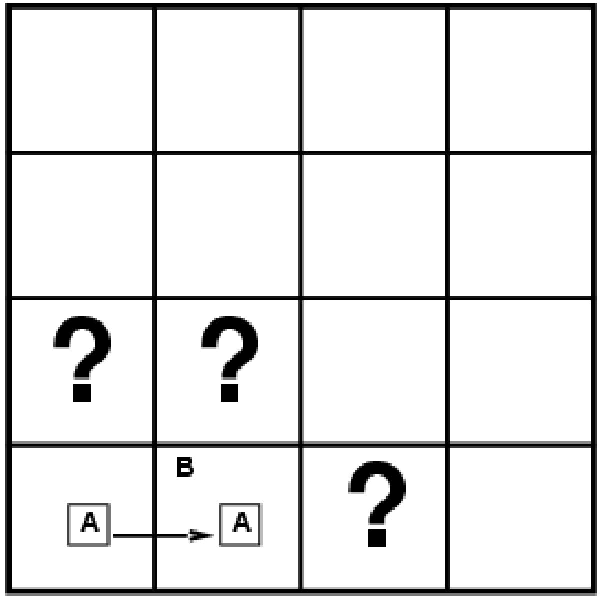
(a) (b)

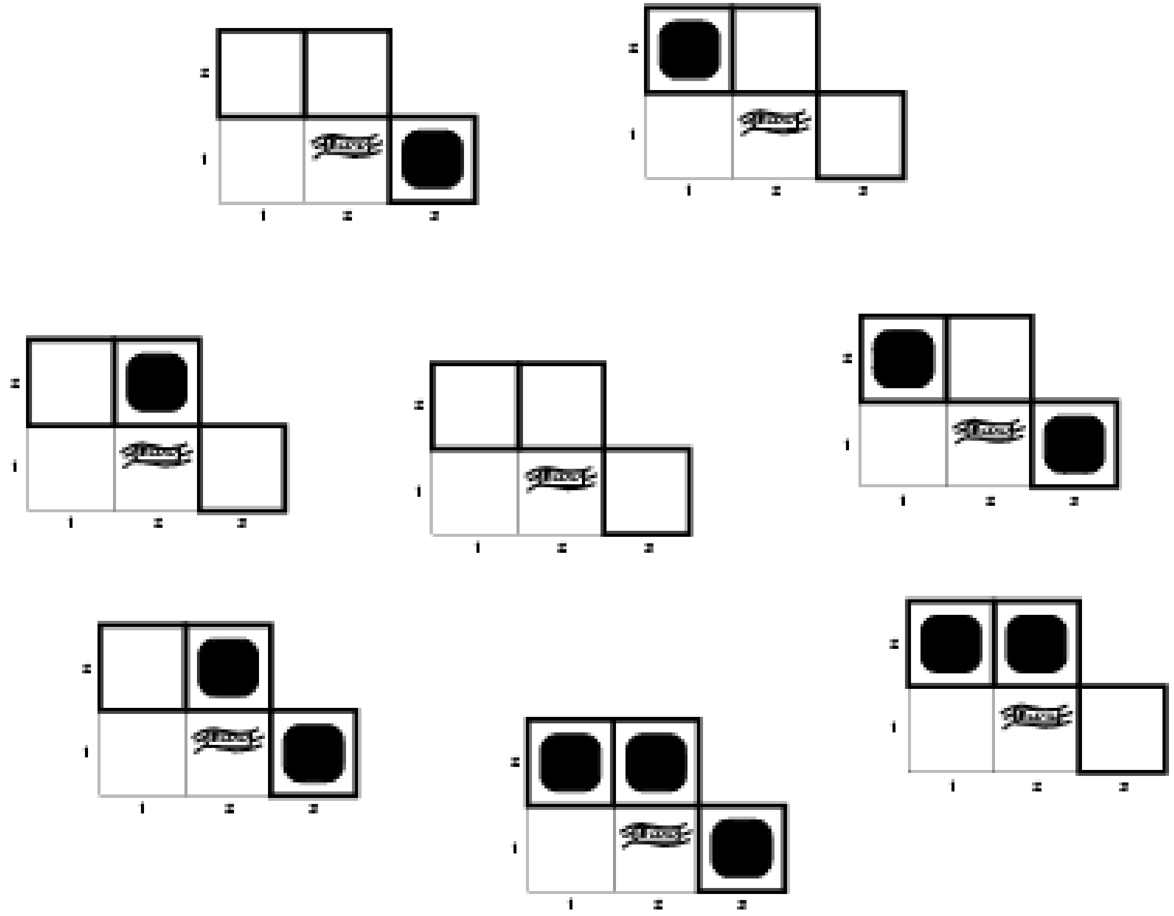
[2,3] : [*Stench,Breeze,Glitter,none,none*]

Thus… pick up gold

Thus… pit in [3,3] or [2,4]

Entailment in the wumpus world





- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]
- Consider possible models for *KB* assuming only pits
- 3 Boolean choices $\Rightarrow$ 8 possible models

Atomic proposition variable and rules for Wumpus world

- Let $\mathbf{P}_{i,j}$ be true if there is a Pit in the room [i, j].
- Let $\mathbf{B}_{i,j}$ be true if agent perceives breeze in [i, j], (dead or alive).
- Let $\mathbf{W}_{i,j}$ be true if there is wumpus in the square[i, j].
- Let $\mathbf{S}_{i,j}$ be true if agent perceives stench in the square [i, j].
- Let $\mathbf{V}_{i,j}$ be true if that square[i, j] is visited.
- Let $\mathbf{G}_{i,j}$ be true if there is gold (and glitter) in the square [i, j].
- Let $\mathbf{OK}_{i,j}$ be true if the room is safe.

(R1) $\neg S_{11} \rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{21}$

(R2) $\neg S_{21} \rightarrow \neg W_{11} \wedge \neg W_{21} \wedge \neg W_{22} \wedge \neg W_{31}$

(R3) $\neg S_{12} \rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{22} \wedge \neg W_{13}$

(R4) $S_{12} \rightarrow W_{13} \vee W_{12} \vee W_{22} \vee W_{11}$

Representation of Knowledge for Wumpus world

- Following is the Simple KB for wumpus world when an agent moves from room [1, 1], to room [2,1]:

$\neg W_{11}$	$\neg S_{11}$	$\neg P_{11}$	$\neg B_{11}$	$\neg G_{11}$	V_{11}	OK_{11}
$\neg W_{12}$	----	$\neg P_{12}$	------	----	$\neg V_{12}$	OK_{12}
$\neg W_{21}$	$\neg S_{21}$	$\neg P_{21}$	B_{21}	$\neg G_{21}$	V_{21}	OK_{21}

Wumpus world Entailment by Model-Checking

- There is no pit in [1,1]:

  R1 : $\neg P_{1,1}$

- Rule:

  R2 : $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

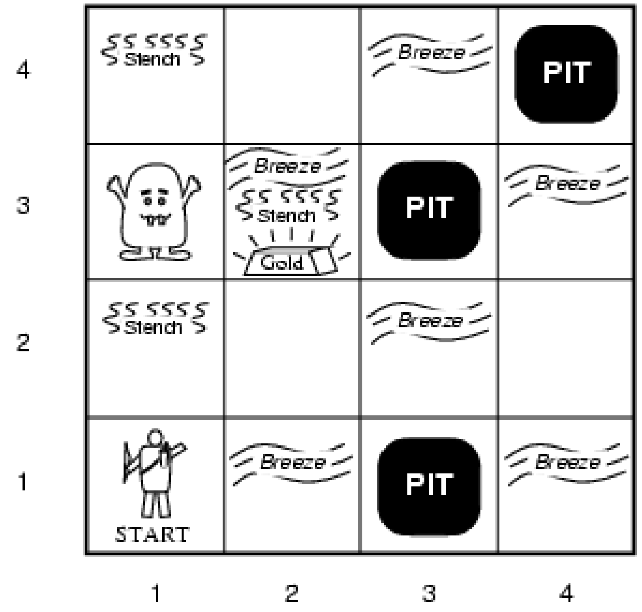
  R3 : $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

- Percepts for the first two squares visited in the specific world the agent:

  R4 : $\neg B_{1,1}$

  R5 : $B_{2,1}$

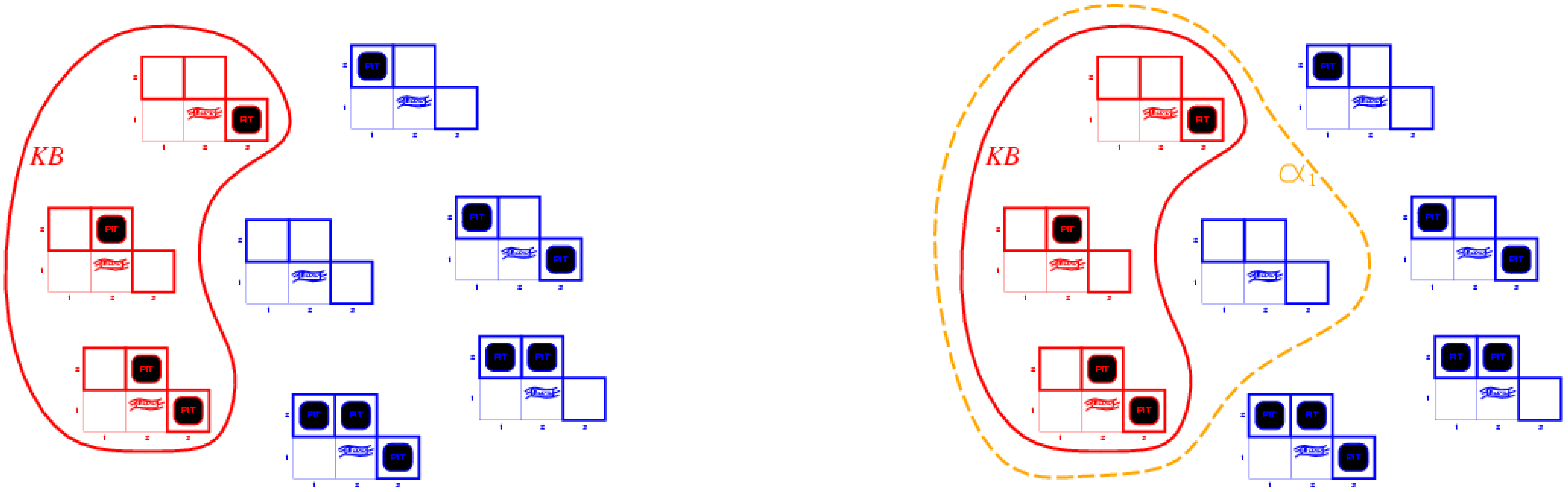
- $\neg P_{1,2}$ entailed by our KB?
- symbols are $B_{1,1}$, $B_{2,1}$, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, and $P_{3,1}$



Truth table constructed for the Knowledge Base

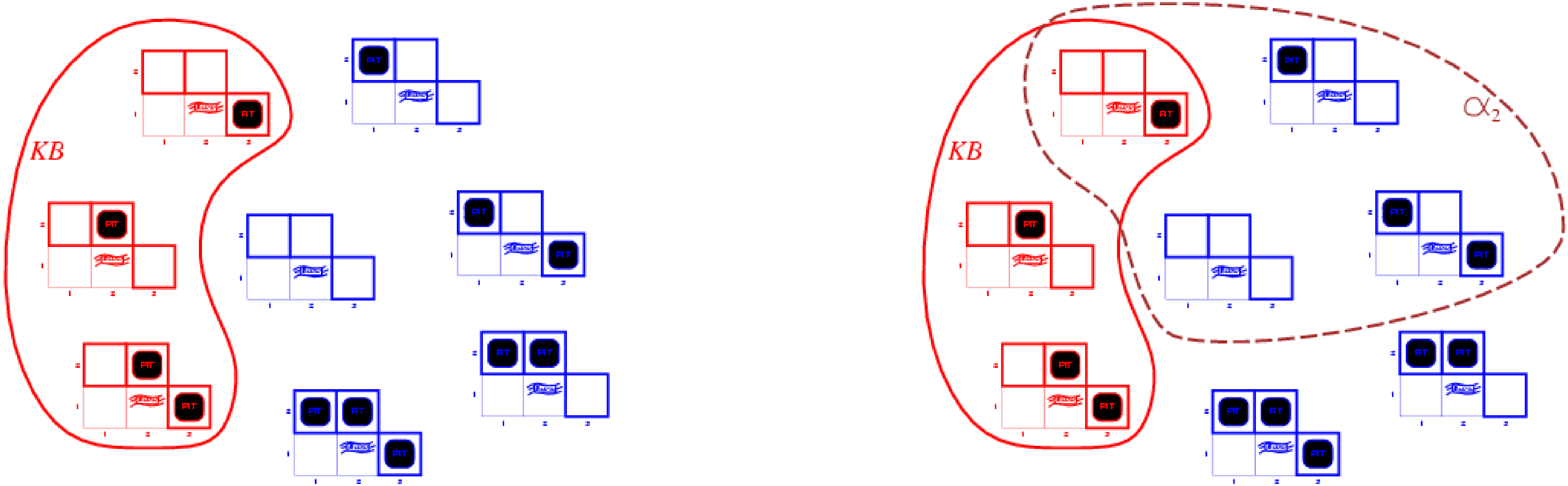
$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
$\vdots$	$\vdots$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

Wumpus models



- *KB* = wumpus-world rules + observations
- α_1 = "[1,2] is safe",
- *KB* $\models \alpha_1$, proved by model checking
  - In every model in which KB is true, α_1 is also true.

Wumpus models



- *KB* = wumpus-world rules + observations
- α_2 = "[2,2] is safe",
- *KB* $\not\models \alpha_2$
  - The agent *cannot* conclude that there is no pit in [2,2] (Nor can it conclude that there is a pit in [2,2])

Inference by Theorem Proving

- There is no pit in [1,1]:

  $R1 : \neg P_{1,1}$
- Rule:

  $R2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

  $R3 : B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$
- Percepts:

  $R4 : \neg B_{1,1}$

  $R5 : B_{2,1}$
- $\neg P_{1,2}$ entailed by our KB?

- Biconditional elimination to R2:

  $R6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
- And-Elimination to R6:

  $R7 : ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
- Logical equivalence for contrapositives:

  $R8 : (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \lor P_{2,1}))$
- Modus Ponens with R8 and the percept R4 (i.e., $\neg B_{1,1}$):

  $R9 : \neg(P_{1,2} \lor P_{2,1})$
- De Morgan's rule to R9:

  $R10 : \neg P_{1,2} \land \neg P_{2,1}$
- That is, neither [1,2] nor [2,1] contains a pit

Resolution in Wumpus World

$R11 : \neg B_{1,2}$

$R12 : B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$

- By the same process that led to R10 earlier, derive the absence of pits in [2,2] and [1,3] (remember $R1 : \neg P_{1,1}$ is in KB) :

$R13 : \neg P_{2,2}$

$R14 : \neg P_{1,3}$

- Apply biconditional elimination to $R3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ followed by Modus Ponens with R5

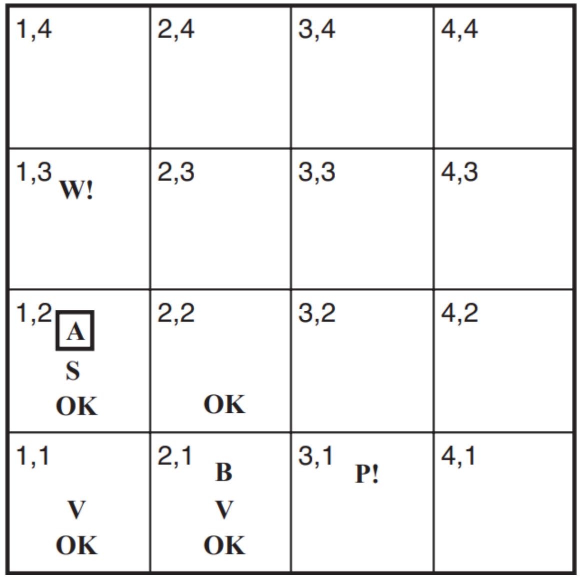
$R15 : P_{1,1} \vee P_{2,2} \vee P_{3,1}$

- Resolution: literal $\neg P_{2,2}$ in R13 resolves with the literal $P_{2,2}$ in R15

$R16 : P_{1,1} \vee P_{3,1}$

- Resolution: literal $\neg P_{1,1}$ in R1 resolves with the literal $P_{1,1}$ in R16

$R17 : P_{3,1}$



Inference by Theorem Proving

- There is no pit in [1,1]:
  $R1 : \neg P_{1,1}$
- Rule:
  $R2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
  $R3 : B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$
- Percepts:
  $R4 : \neg B_{1,1}$
  $R5 : B_{2,1}$
- $\neg P_{1,2}$ entailed by our KB?
- Biconditional elimination to R2:
  $R6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
- And-Elimination to R6:
  $R7 : ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
- Logical equivalence for contrapositives:
  $R8 : (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \lor P_{2,1}))$
- Modus Ponens with R8 and the percept R4 (i.e., $\neg B_{1,1}$):
  $R9 : \neg(P_{1,2} \lor P_{2,1})$
- De Morgan's rule to R9:
  $R10 : \neg P_{1,2} \land \neg P_{2,1}$
- That is, neither [1,2] nor [2,1] contains a pit

Resolution in Wumpus World

R11 : $\neg B_{1,2}$

R12 : $B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$

- By the same process that led to R10 earlier, derive the absence of pits in [2,2] and [1,3] (remember R1 : $\neg P_{1,1}$ is in KB) :

R13 : $\neg P_{2,2}$

R14 : $\neg P_{1,3}$

- Apply biconditional elimination to R3 : $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ followed by Modus Ponens with R5

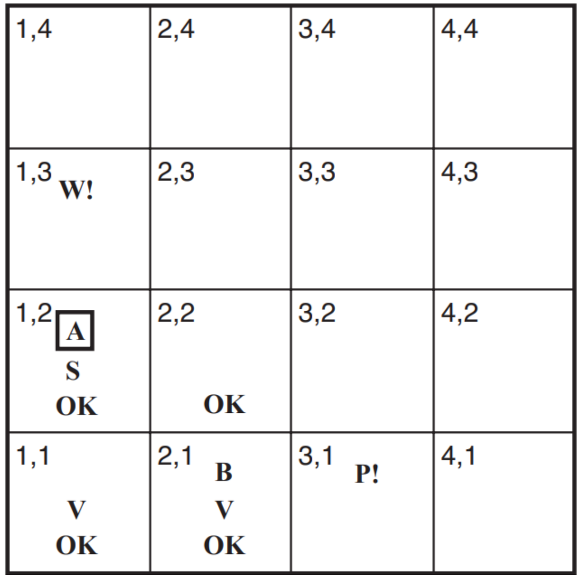
R15 : $P_{1,1} \vee P_{2,2} \vee P_{3,1}$

- Resolution: literal $\neg P_{2,2}$ in R13 resolves with the literal $P_{2,2}$ in R15

R16 : $P_{1,1} \vee P_{3,1}$

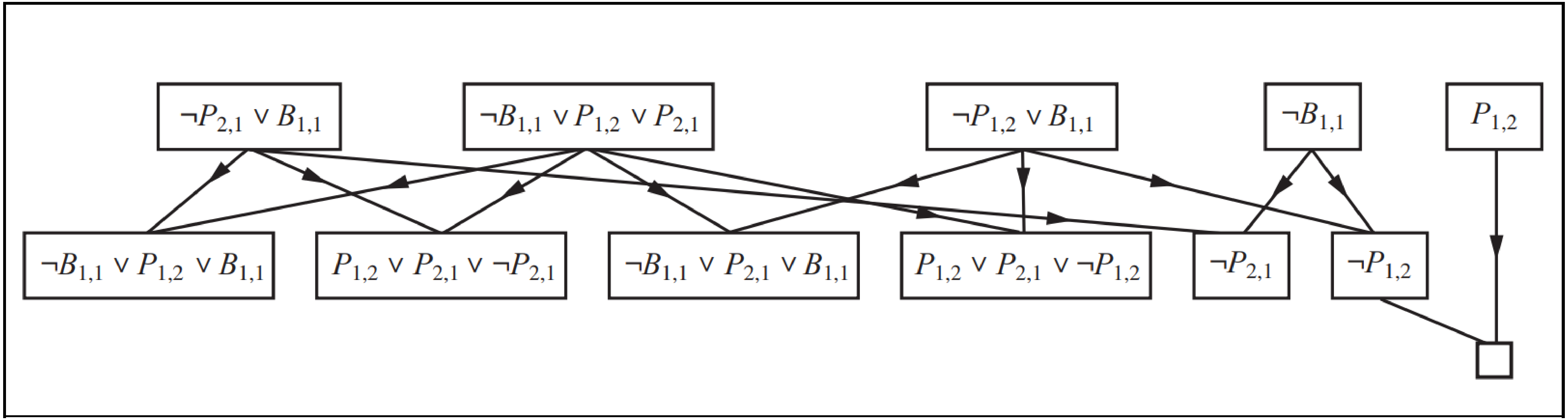
- Resolution: literal $\neg P_{1,1}$ in R1 resolves with the literal $P_{1,1}$ in R16

R17 : $P_{3,1}$



Wumpus World example

- *KB* = R4 $\wedge$ R5 = $(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$ $\alpha = \neg P_{1,2}$



Efficient propositional inference

Two families of efficient algorithms for propositional inference:

- Complete backtracking search algorithms
  - DPLL algorithm (Davis, Putnam, Logemann, Loveland)
- Incomplete local search algorithms
  - `WalkSAT` algorithm

Limitation

- Propositional logic has limited expressive power.
- We cannot represent relations like ALL, some, or none with propositional logic. Example:
  - **All the animals are intelligent.**
  - **Some apples are sweet.**
- We cannot describe statements in terms of their properties or logical relationships.