CS5100 Foundations of Artificial Intelligence

Module 03 Lecture 06

Propositional Logic

Overview

- Knowledge-Based Agents
- The Wumpus World
- Logic
- Propositional Logic
- Propositional Theorem Proving
- Model Checking, Inference Rules, and Resolution

Human Agents

- Human Intelligence:
  achieved by **reasoning** on
  internal **representations of knowledge**

E.g., “I slipped on a street and fell”

- Was the street wet?
- Was there a banana peel or something slippery?

But not:

- Is it Tuesday?

Search ➔ Logic

- Search methods (BFS, DFS, A*, Minimax, … Local Search) use the Result function to predict outcomes of actions, *but cannot deduce anything new.*
- CSP allows for more efficient, domain-independent algorithms, *but cannot deduce anything new either.*
- For the next few lectures, our focus is on **logic** as a **representation** for **knowledge-based (KB)** agents
  - Can define new tasks
  - Can be told, or can learn new things
  - Can adapt to changes in environment

Knowledge-Based Agents

Agents that:

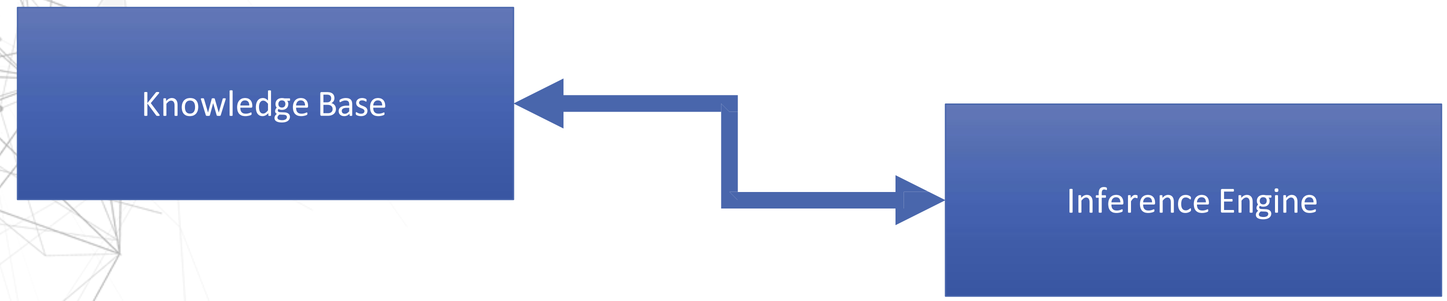
- Have a representation of the **knowledge** about their environment
- Use **inference** to derive new information from that knowledge combined with new perceptual inputs

Knowledge Base (Domain-specific)

- A set of **sentences** that describe facts about the world in some formal **knowledge representation language**

Inference Engine (Domain-independent)

- Procedures that use the representational language to **infer new facts** from known ones or answer a variety of KB queries. Inferences typically require search.



A simple knowledge-based agent

The agent program:

- Maintains a KB
- Tells the KB what it perceives
- Asks the KB what action it should take
    - ➔ Reasoning ➔ Chooses action
- Tells the KB what action was chosen, and takes the action

Declarative vs. **procedural** approaches to defining an agent

```
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
```

The Wumpus World

The **Wumpus World** is a cave consisting of rooms connected by passageways

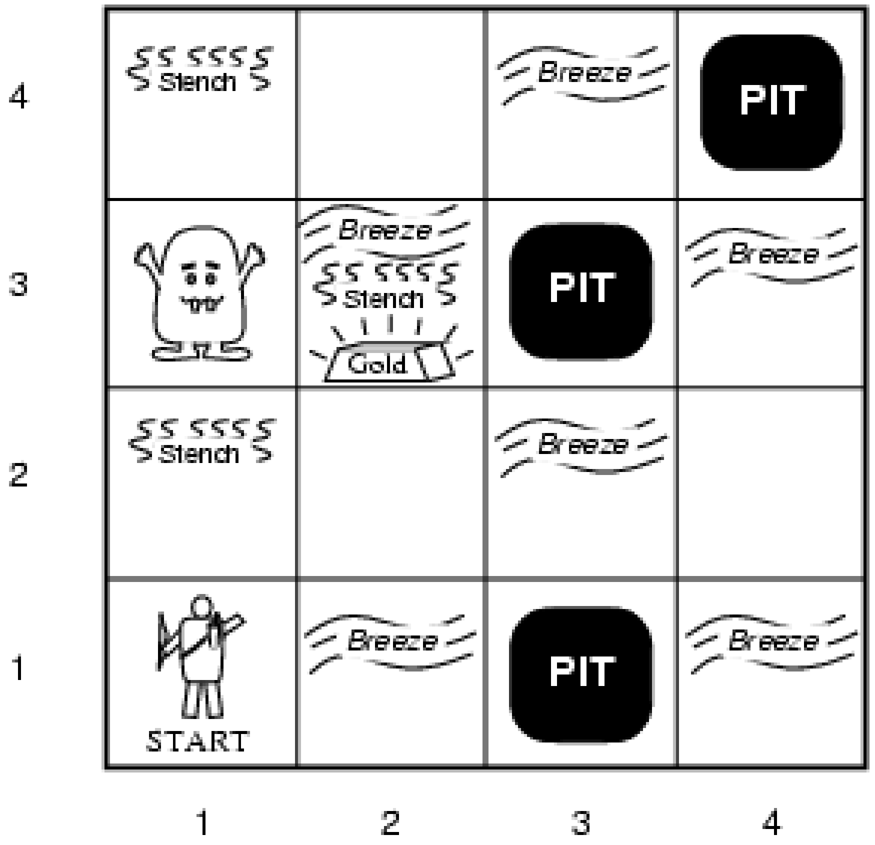
Goal for agent:

- Get the heap of gold
- Avoid the Wumpus (terrible beast)
- Avoid bottomless pits
- Can shoot wumpus, only 1 arrow

Sensors:

- Breeze ➔ you are close* to a pit
- Stench ➔ you are close* to the Wumpus
- Glitter ➔ in the same square as the gold

* You have to be in a directly adjacent square (top, bottom, left or right), to be 'close'



Wumpus World PEAS description

Performance measure

- Gold: +1000
  Death by wumpus/fall into pit: -1000
  -1 per action, -10 for use of arrow

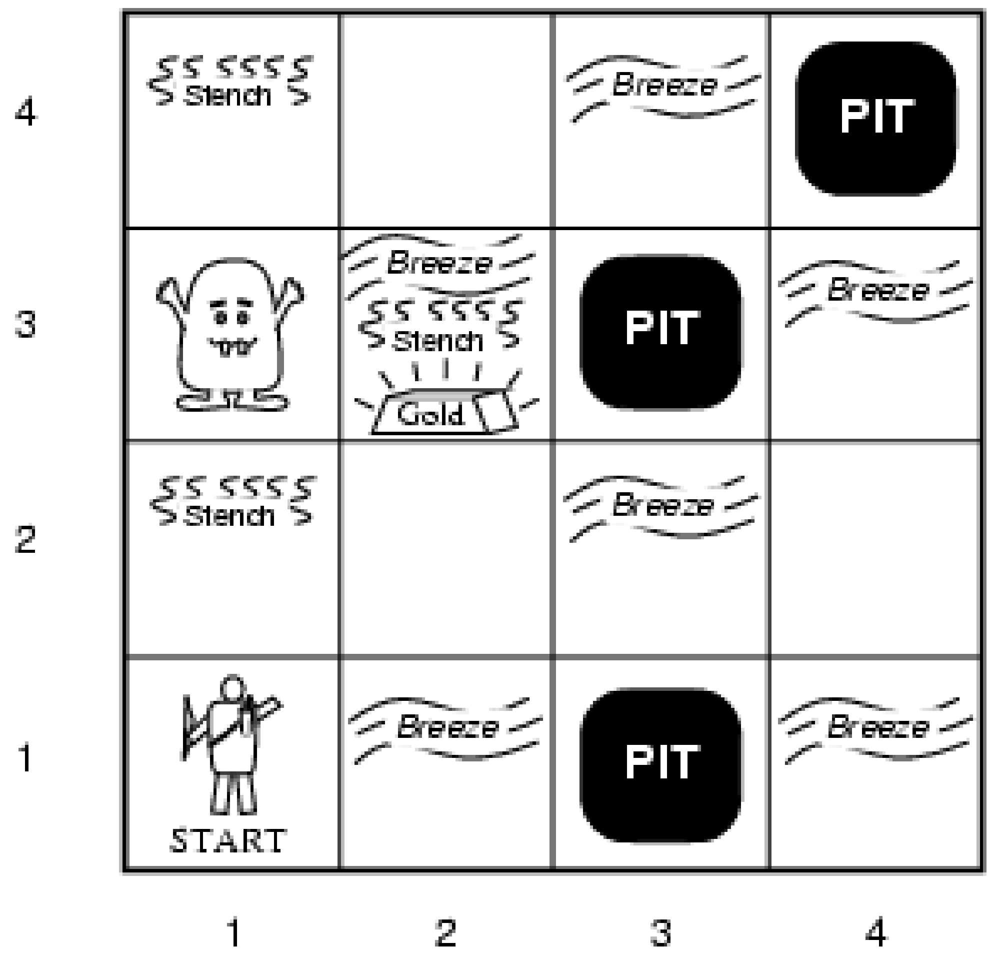
Game end: agent climbs out or dies

Environment: 4x4 grid

- Squares adjacent to wumpus are smelly
- Squares adjacent to pit are breezy
- Glitter if gold is in the same square
- Shooting kills wumpus if you are facing it
- Shooting uses up the only arrow
- Bump if agent walks into a wall
- Grabbing picks up gold if in same square
- Releasing drops the gold in same square
- Scream when wumpus dies

Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot

Sensors: Stench, Breeze, Glitter, Bump, Scream



Wumpus World characterization

Discrete?

- Yes

Static?

- Yes – Wumpus, pits do not move around

Single-agent?

- Yes – Wumpus does not move

Fully Observable?

- No: Partial – only local perception

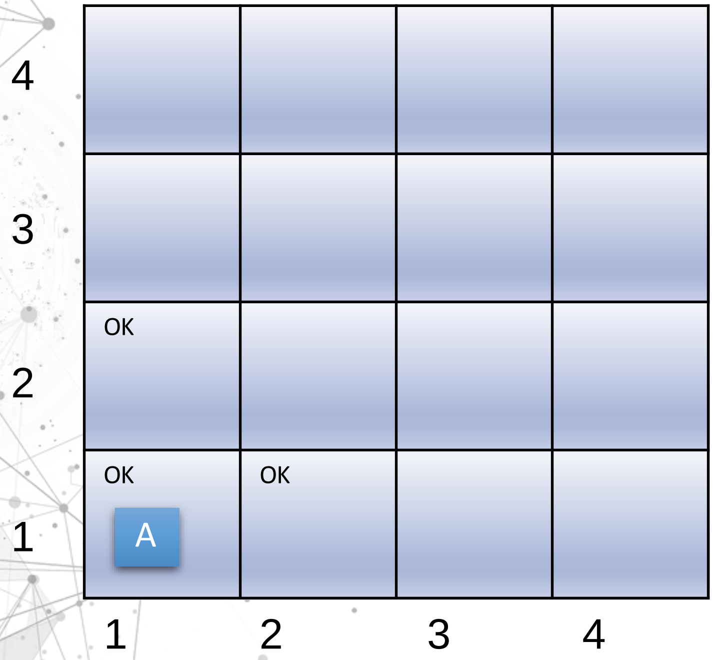
Deterministic?

- Yes – outcomes exactly specified

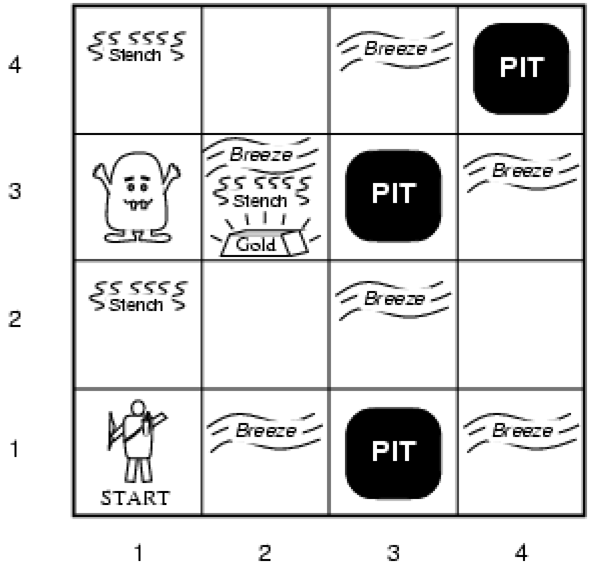
Episodic?

- No – sequential at the level of actions

Exploring the Wumpus World



A: Agent
B: Breeze
G: Gold
OK: safe square
P: Pit
S: Stench
V: Visited
W: Wumpus

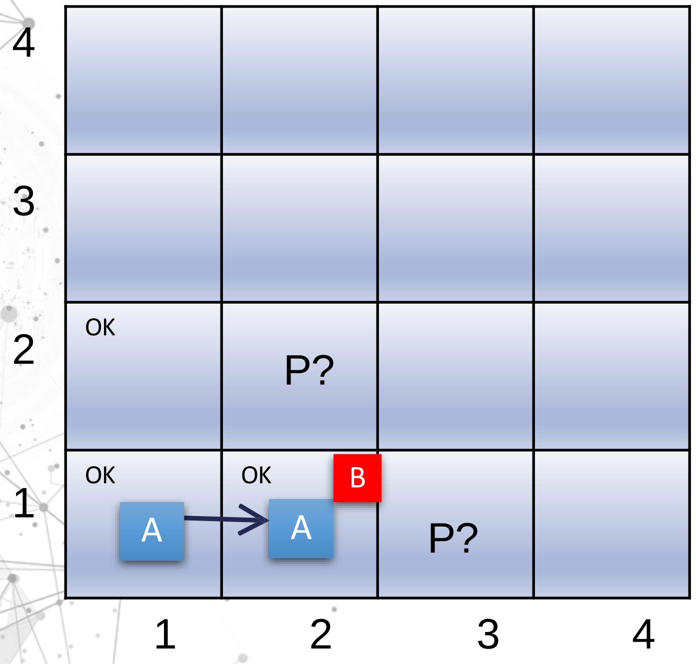


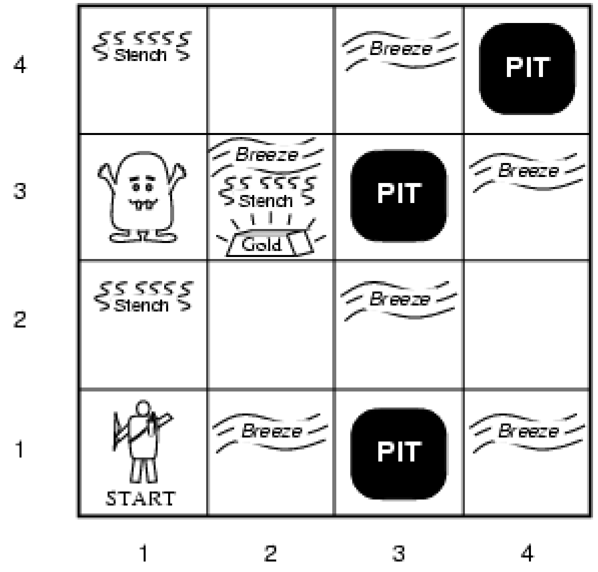
Agent Position: (1,1)
Agent still alive, so no pit or Wumpus in (1,1).

No stench ($\neg S_{11}$), No breeze ($\neg B_{11}$) ➔
no pit ($\neg P_{11}$) and
no Wumpus in (2,1) or (1,2) ($\neg W_{21}$, $\neg W_{12}$)

2 options: Agent moves to (2,1) or (1,2)

Exploring the Wumpus World





Agent Position: (2,1)

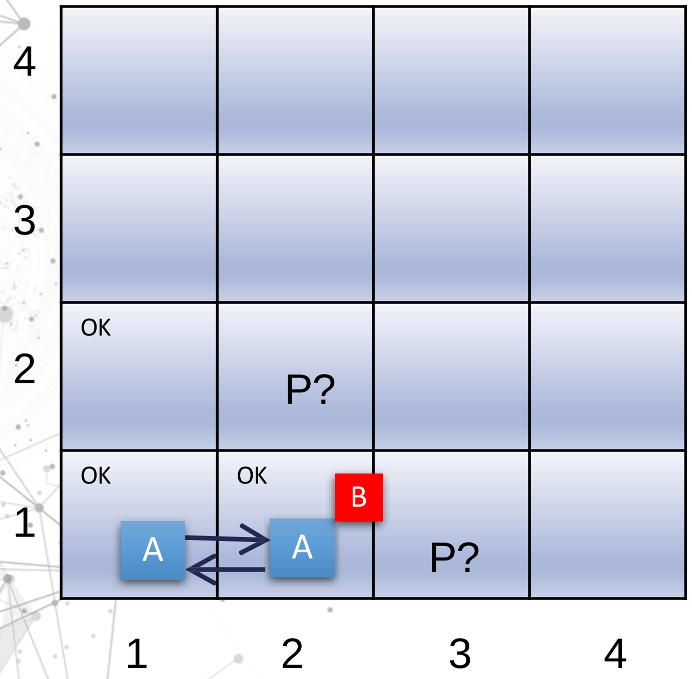
Breeze in (2,1) B_{21} ➔ pit in (1,1), (2,2) or (3,1)

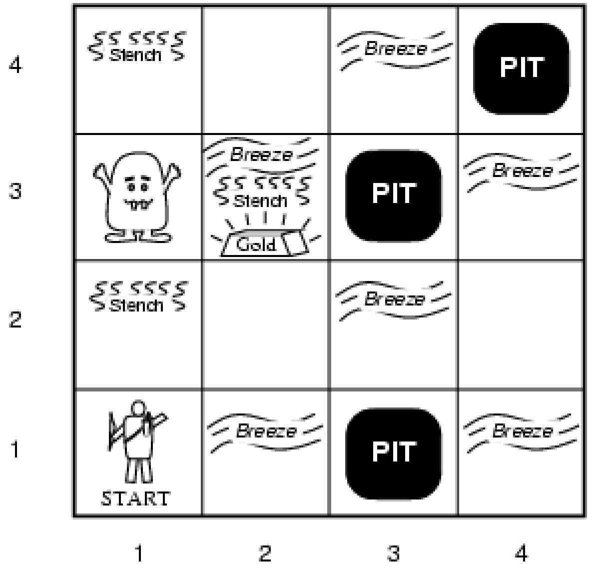
Player already ruled out (1,1)

Since there is danger in moving to (3,1) and (2,2), player moves back to (1,1)

A: Agent
B: Breeze
G: Gold
OK: safe square
P: Pit
S: Stench
V: Visited
W: Wumpus

Exploring the Wumpus World

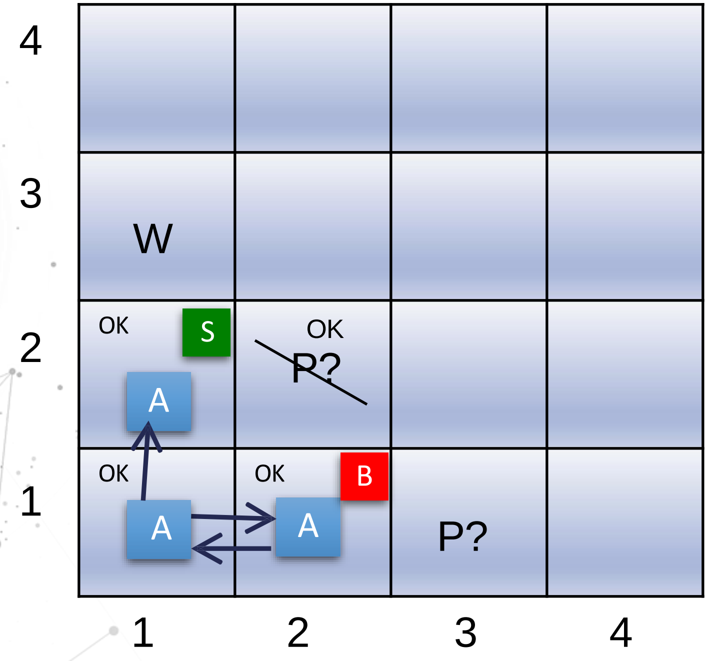


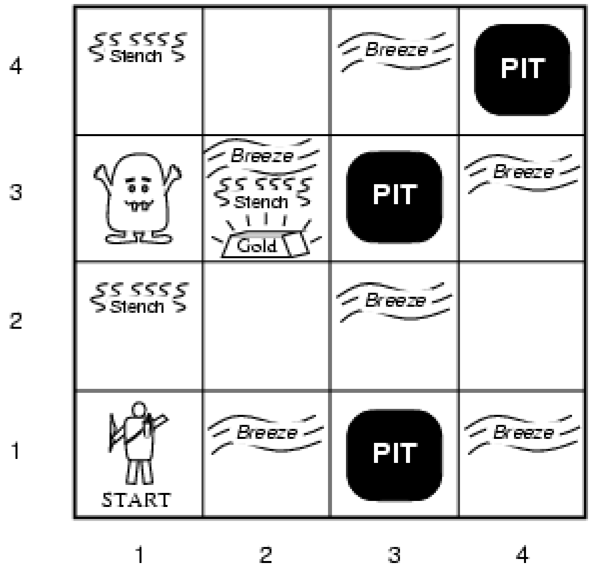


Agent Position: (1,1)
Move to Position (1,2) as that is the safest next place to explore

A: Agent
B: Breeze
G: Gold
OK: safe square
P: Pit
S: Stench
V: Visited
W: Wumpus

Exploring the Wumpus World





Agent Position: (1,2)
Not Breezy here. So ¬ P_{22}. And so P_{31}
Stench in this position S_{12} implies Wumpus in 1,3 or 2,2 or 1,1
We know that ¬W_{11} and ¬W_{22} Therefore W_{13}
Next available safe spot is (2,2,) ….
and so go on to collect gold and get out safely.

A: Agent
B: Breeze
G: Gold
OK: safe square
P: Pit
S: Stench
V: Visited
W: Wumpus

What we take away

- Agent draws conclusions based on available information
- Conclusions guaranteed to be correct if the available information is correct
- The agent updates its incomplete model with new information based on new percepts

Rest of this lecture:
How we build logical agents that represent info and draw conclusions

Logic

Logic is formal language that is used for representing information such that conclusions can be drawn from it

Syntax defines the sentences in the language

Semantics define the "meaning" of sentences;

- i.e., define truth of a sentence in a world

E.g., in the language of arithmetic

- Syntax: x+y = 5 is a well-formed sentence, but x2y= is not
- Semantics: x+y = 5 is true iff the numbers x + y add up to 5
- x+y = 5 is **true** in a **world** (a **model**) where x = 2, y = 3,
  but false in a world where x = 3 and y = 3

Models

Semantics defines truth of each sentence with respect to each **possible world** or **model**

Model: Mathematical Abstraction of the real world

We say *m is a model of a sentence α* (or *m satisfies α*) if α is true in (the model) *m*

M(α) is the set of all models of α, that is, all the models where α is true

M(x+y = 5) is …

… all the worlds, all the models, in which if you add x plus y, you get the answer 5.

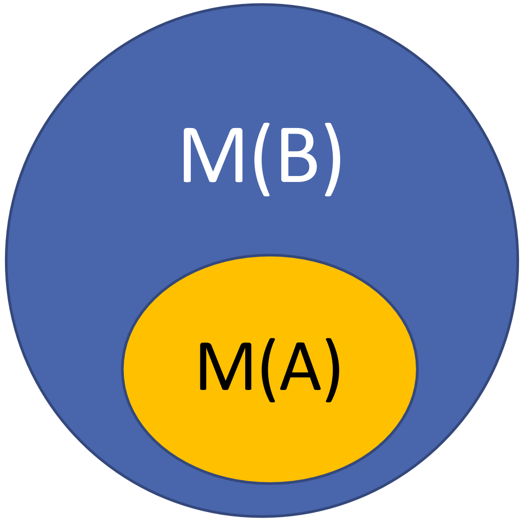
Entailment

Entailment reflects the idea that one sentence in the world follows logically from another

$A \mid= B$ (A entails B)

Definition: $A \mid= B$ if and only if $M(A) \subseteq M(B)$

- Under all interpretations in which A is true, B is true as well
- All models of A are models of B
- Whenever A is true, B is true as well; B follows from A



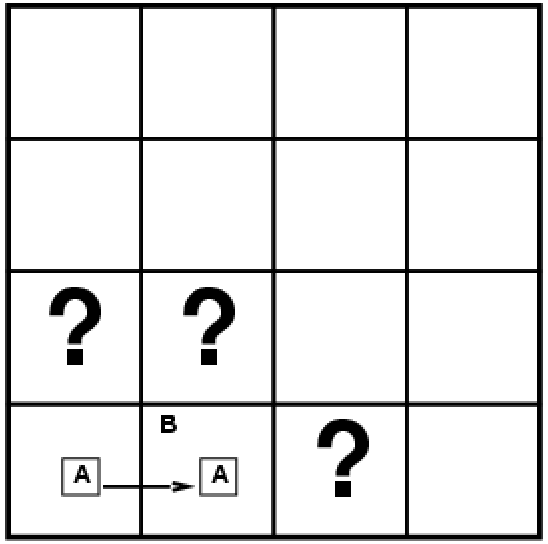
Math example: $x = 0$ entails $x*y = 0$

Entailment in the Wumpus World

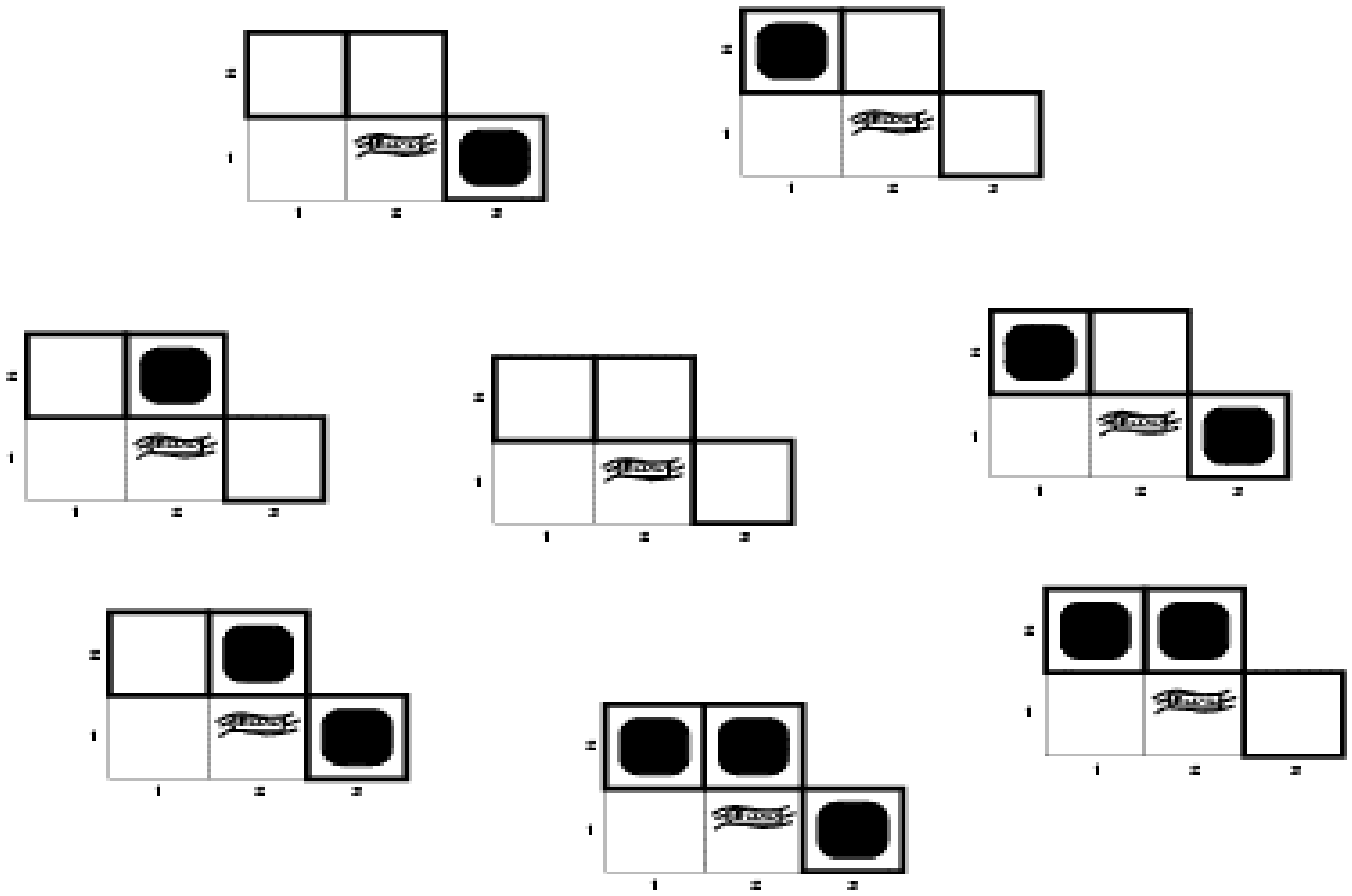
Situation after detecting nothing in [1,1], moving right, breeze in [2,1]

Consider possible models for *KB* assuming only pits
  Do any of the squares adjoining [1,1] and [2,1] contain pits?

3 Boolean choices means 2^3 = 8 possible models

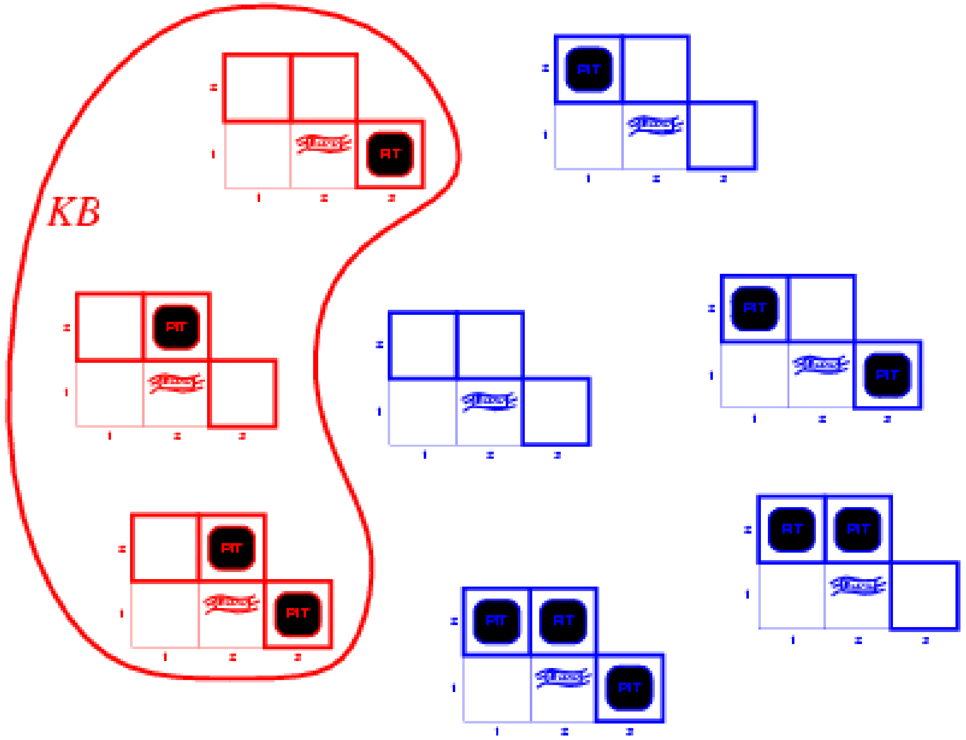


Wumpus models



Three models where KB is true

KB = wumpus-world rules + observations



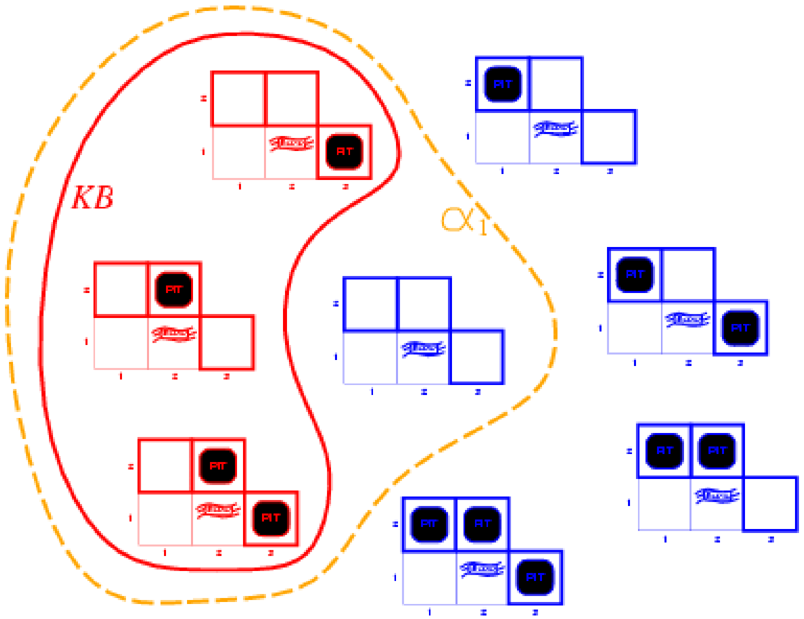
Wumpus models for α_1

KB = wumpus-world rules + observations

α_1 = "[1,2] is safe" (no pit there),

Remember the definition of A |= B: Whenever A is true, B is true as well

KB $\models \alpha_1$, proved by model checking (by enumerating all possible models)



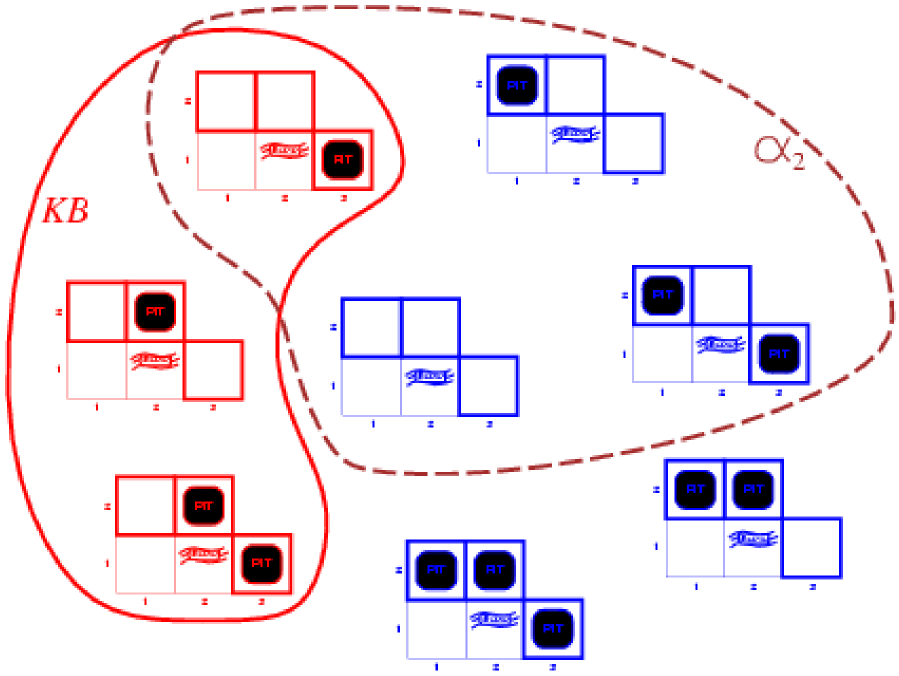
Wumpus models for α_2

KB = wumpus-world rules + observations

α_2 = "[2,2] is safe"; in some models where KB is true, α_2 is false

Question: Does KB $\models \alpha_2$?

No! Cannot conclude anything about pit in 2,2



Remember the definition of A |= B: Whenever A is true, B is true as well

Sound and Complete Inference

Soundness:

An inference algorithm that derives only entailed sentences
is called **sound** or **truth-preserving**

Model-checking is sound (when applicable)

Completeness:

An inference algorithm is **complete** if it can derive any sentence that is entailed

Propositional Logic

Propositional logic:
specific language for **symbolic reasoning**
defined by its syntax and semantics

Proposition:

a statement that can hold a *true* or *false* value

Are these examples of propositions:

- Seattle is a city
  - Yes
- Seattle is always sunny
  - Yes
- What color are the walls?
  - No

Syntax of Propositional Logic

Propositional symbols:

- Constants with fixed meanings: True, False
- Propositional symbols (typically uppercase;
  atomic, even when of the form $A_{1,2}$)

Examples of **atomic sentences** (single propositional symbols):

- P, Q, Raining, $W_{2,3}$

A set of 5 **connectives**, along with operator precedence rules

¬, ∧ , ∨ , ⇒ , ⇔

Complex sentences made using symbols, connectives and parentheses

E.g.: $\neg A$,$(A \land B)$, $(A \lor B)$, $(A \Rightarrow B)$, $(A \Leftrightarrow B)$,
$(A \lor (C \land B))$

BNF Grammar of Propositional Logic

Sentence ➔ Atomic Sentence | Complex Sentence

Atomic Sentence ➔ True | False | P | Q | R …

Complex Sentence ➔ (Sentence) | [Sentence] |
¬ Sentence |
Sentence ∧ Sentence |
Sentence ∨ Sentence |
Sentence ⇒ Sentence |
Sentence ⇔ Sentence

Operator Precedence: ¬, ∧ , ∨ , ⇒ , ⇔

Backus-Naur Form

Semantics of Propositional Logic ..1

Semantics:

rules to determine truth of a sentence with respect to a **model/possible world**

Semantics in propositional logic defined by:

1. Semantics of atomic sentences

   How we interpret propositional symbols and constants

2. Semantics of complex sentences:

   How we interpret the operators

Semantics of Propositional Logic ..2

A propositional symbol

- a statement about the world, either True or False

  e.g.: It is hot today

Propositional symbols are mapped to one of the two values:

- True is True in all models, False is False in all models.
- Other atomic propositions have to be assigned a value of True or False in the model.

  e.g. It is hot today -> False

For complex sentences, use semantics of operators.

Semantics of NOT (¬)

P	Q	¬P
True	True	False
True	False	False
False	True	True
False	False	True

Semantics of OR (∨) and AND (∧)

P	Q	P ∨ Q
True	True	True
True	False	True
False	True	True
False	False	False

OR: Disjunction
AND: Conjunction

P	Q	P ∧ Q
True	True	True
True	False	False
False	True	False
False	False	False

Semantics of IMPLIES (⇒)

P	Q	P ⇒ Q
True	True	True
True	False	False
False	True	True
False	False	True

- P ⇒ Q True unless P is True and Q is false in model
- No causation or relevance required between P and Q
- “5 is odd” ⇒ “Olympia Is the capital of Washington” is peculiar to say, but True!
- Important: Implication True if antecedent (P) is False!
  - “If P is True, I am claiming Q is true, else I’m making no claim”

Semantics of BICONDITIONAL (⇔)

P	Q	P ⇔ Q
True	True	True
True	False	False
False	True	False
False	False	True

- P ⇔ Q is true when both P ⇒ Q and Q ⇒ P are true
- In English: P if and only if Q (P iff Q)

- Wumpus World: A square is breezy *if a* neighboring square has a pit, and a square is breezy only *if* a neighboring square has a pit

$$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$$

Semantics of Composite Sentences

Can compute semantics of complex sentences using truth tables

P	Q	¬P	P ∧ Q	P ∨ Q	P ⇒ Q	P ⇔ Q
True	True	False	True	True	True	True
True	False	False	False	True	False	False
False	True	True	False	True	True	False
False	False	True	False	False	True	True

Examples

Given the propositions

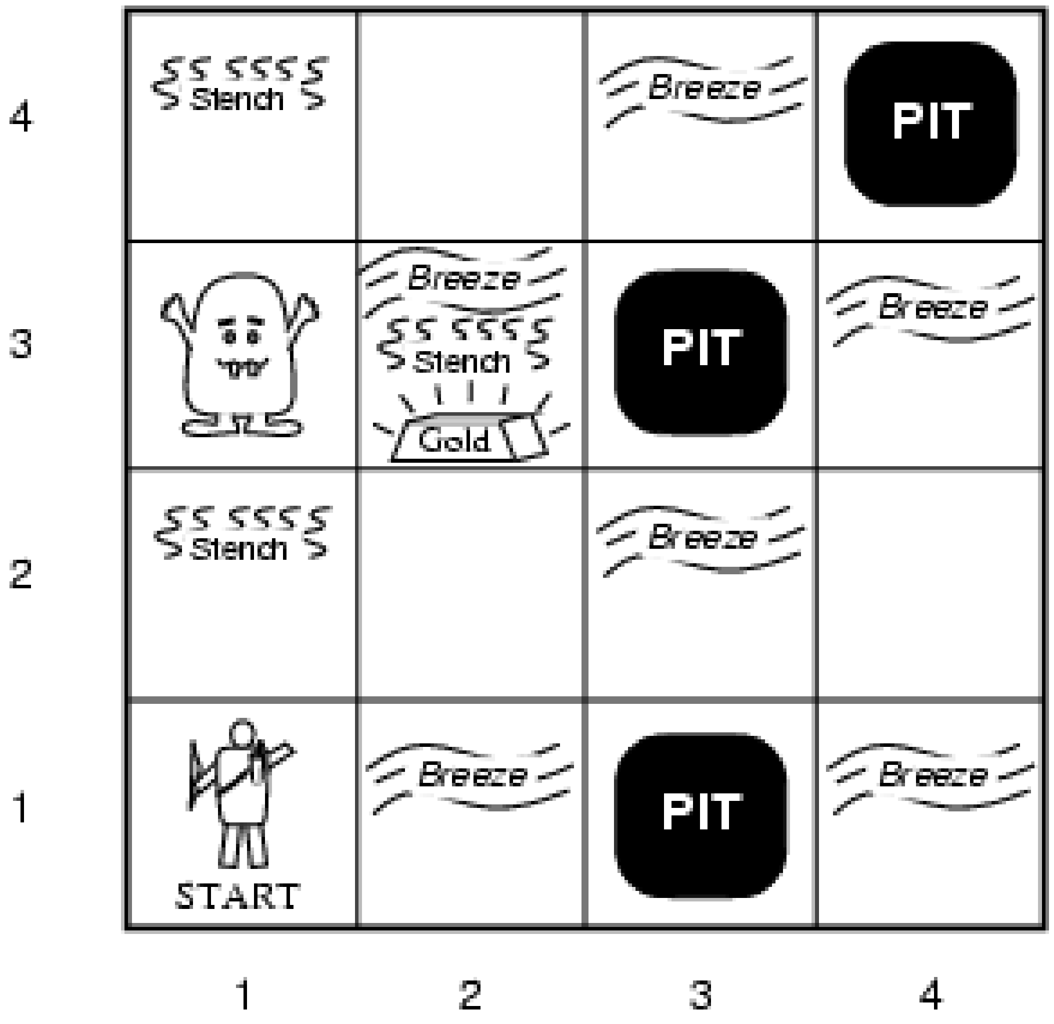
- S: sunny -- it is sunny today
- R: rainy -- it is raining today
- H: hiking – we will go on a hike today
- B: biking -- we will go biking today
- M: museum – we're going to the Seattle Art Museum

How do you say the following in Propositional Logic?

1. It's not rainy today and we will go on a hike today $\neg R \wedge H$
2. If we don't go biking, we will go on a hike $\neg B \Rightarrow H$

Wumpus World KB (subset)

- No pit [1,1]
  R1: $\neg P_{1,1}$
- A square is breezy iff pit in neighboring square
  R2: $(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}))$
  R3: $(B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1}))$
  …
- Breeze precepts:
  R4: $\neg B_{1,1}$
  R5: $B_{2,1}$



Logical Equivalences

Sentences A and B are **logically equivalent** if each of them entails the other

- i.e. they are true under exactly the same interpretations

$A \equiv B$ iff $A \models B$ and $B \models A$

Validity

A sentence (or set of sentences) is **valid** if

it is true under all interpretations

E.g.: $P \vee \neg P$

Valid sentences also known as **tautologies** (necessarily true)

- **Contradictions** are always false e.g. $P \wedge \neg P$

Satisfiability

A sentence (or set of sentences) is **satisfiable** if it is true in *some* model

Solving Logical Inference Problems

Three Approaches:

- Model Checking Approach
- Inference Rules Approach
- Resolution – Refutation Approach

Model Checking Approach

Problem: KB $\models \alpha$?

- We need to check all the possible interpretations for which the KB is true (models of KB), if α is true for each of them

Truth Table

- Enumerates the truth values of sentences for all possible interpretations

Remember: The entailment model in the Wumpus World used model checking

Model Checking Approach Example 1

KB		α
A	B	A ∧ B
F	F	F
F	T	F
T	F	F
T	T	T

- Problem: KB |= α?
  - A, B entails A∧B?
- Solution
  - Generate table for all possible interpretations
  - Check whether α is true whenever KB is true

A,B, Entails A∧B

Model Checking Approach Example 2

		KB		α
A	B	C	A ∧ C	B ∧ C
F	F	F	F	F
F	F	T	F	F
F	T	F	F	F
F	T	T	F	T
T	F	F	F	F
T	F	T	T	F
T	T	F	F	F
T	T	T	T	T

- Problem: KB |= α ?
  - KB = A ∧ C, C
  - α = B ∧ C

A ∧ C, C does not entail B∧C

Model Checking
Pros and Cons

- The Model Checking approach is sound and complete for Propositional Logic
- But: Search Space for truth tables is exponential.
    - 3 variables – 2^3 = 8
    - 10 variables – 2^10 = 1024
    - KB is true only in a small set of interpretations

- Need to be more efficient

Inference Rules for Logic

Apply **inference rules** to derive a **proof Modus Ponens**

$$\frac{A \Rightarrow B,\ A}{B}$$

Premise

Conclusion

If the premise is true, conclusion is also true

E.g. from (Wumpus Ahead ⇒ Shoot), Wumpus-Ahead infer Shoot

Inference Rules for Logic

And Elimination

$$\frac{A \wedge B}{A}$$

From a conjunction, any of the conjuncts can be inferred

Bi-Conditional Elimination

$$\frac{A \Leftrightarrow B}{(A \Rightarrow B) \wedge (B \Rightarrow A)}$$

Double Negation Elimination

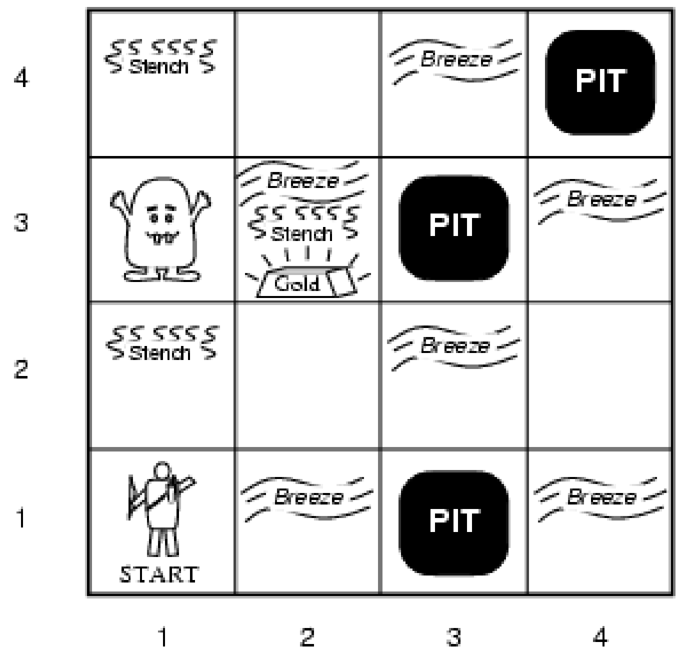
$$\frac{\neg\neg A}{A}$$

Logical equivalences

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

Wumpus World KB (subset) REPEAT

- No pit [1,1]
  R1: $\neg P_{1,1}$
- A square is breezy iff pit in neighboring square
  R2: $(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}))$
  R3: $(B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1}))$ ...
- Breeze precepts:
  R4: $\neg B_{1,1}$
  R5: $B_{2,1}$



Example
Proof by Inference Rules

Given KB, prove $\neg P_{12}$

Start with R2: $(B_{11} \Leftrightarrow (P_{12} \vee P_{21}))$ [Why? Because it contains P_{12}]

1. Bi-conditional elimination:

   R6: $(B_{11} \Rightarrow (P_{12} \vee P_{21})) \wedge ((P_{12} \vee P_{21}) \Rightarrow B_{11})$
2. AND-elimination: R7: $((P_{12} \vee P_{21}) \Rightarrow B_{11})$
3. Contrapositives: R8: $(\neg B_{11} \Rightarrow \neg (P_{12} \vee P_{21}))$
4. Apply Modus Ponens to R8 with percept R4 $(\neg B_{11})$ to get R9: $\neg (P_{12} \vee P_{21})$
5. De Morgan's rule: R10: $\neg P_{12} \wedge \neg P_{21}$

That is, neither P_{12} nor P_{21} contain a pit.

- No pit [1,1]

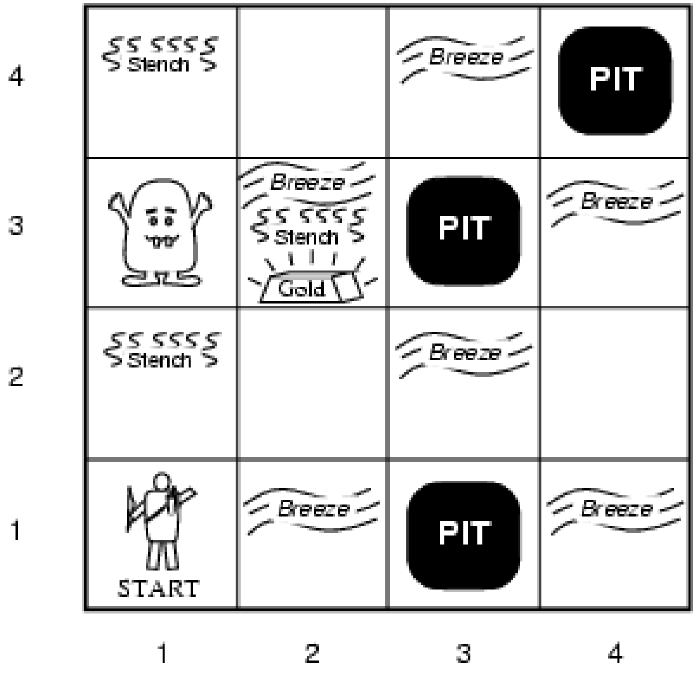
  R1: $\neg P_{1,1}$
- A square is breezy iff pit in neighboring square

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

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

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

  R5: $B_{2,1}$



Resolution in Wumpus World

Say: There is a pit at 2,1 or 2,3 or 1,2 or 3,2

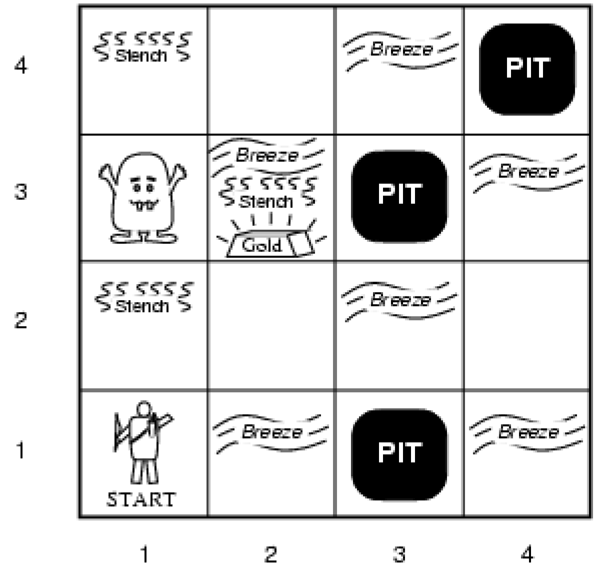
- $P_{21} \lor P_{23} \lor P_{12} \lor P_{32}$

And: There is no pit at 2,1

- $\neg P_{21}$

Therefore (by resolution) the pit must be at 2,3 or 1,2 or 3,2

- $P_{23} \lor P_{12} \lor P_{32}$



Inference Rules for Logic: Resolution

Resolution takes two clauses (disjunction of literals) and produces a new clause containing all the literals of the two original clauses except the two complementary literals

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

But it applies only to clauses!
So convert to Conjunctive Normal Form (CNF) – coming up !

The resulting clause should contain only one copy of each literal.

$$\frac{(A \vee B),\ (A \vee \neg B)}{(A \vee A)} \quad = \quad \frac{(A \vee B),\ (A \vee \neg B)}{A}$$

Proof using Resolution

To prove P from KB:

1. Convert KB and P into CNF
2. To prove P, prove KB $\wedge \neg$P leads to contradiction (empty clause)
3. Specifically, apply resolution on pairs of clauses, adding new clauses produced, until:

- No new clauses can be added, (KB does not entail P) or
- **The empty clause is derived (KB does entail P).**

Resolution Example

When the agent is in 1,1, there is no breeze, so there can be no pits in neighboring squares

- KB: $(B_{11} \Leftrightarrow (P_{12} \vee P_{21}))$; $\neg B_{11}$
- Prove: $\neg P_{12}$.

Conversion to CNF

Conjunctive Normal Form (CNF): conjunction (AND) of clauses, where each clause is a disjunction (OR) of literals

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

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 ($\land$ over $\lor$) and flatten, to get:

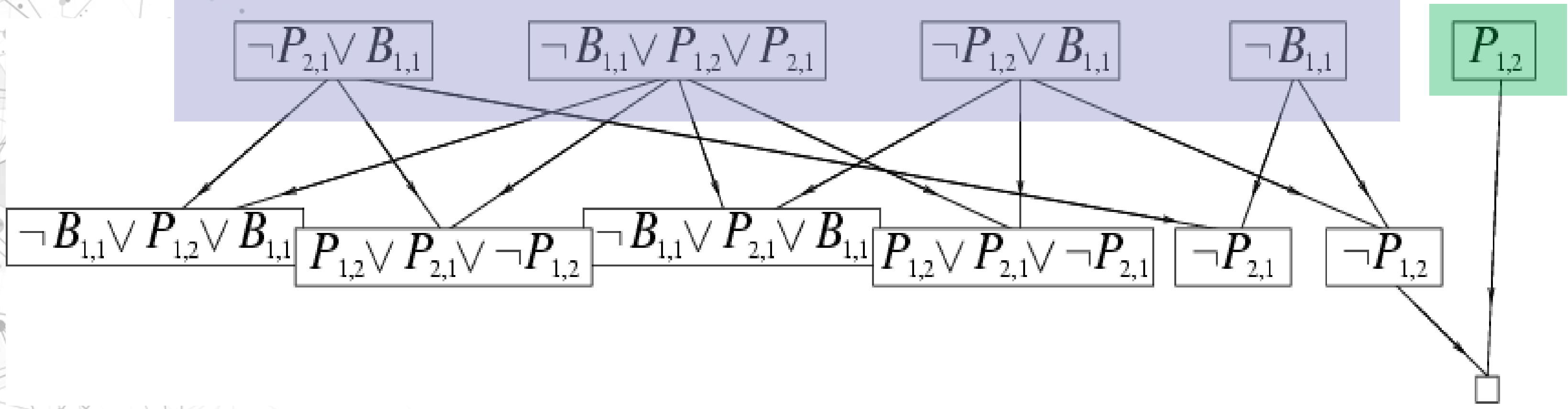
   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1})$

Resolution example

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

$\alpha = \neg P_{1,2}$

CNF of KB: $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1}) \wedge \neg B_{1,1}$



Resolution algorithm

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

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

Horn Clauses & Definite Clauses .. 1

- Restrict form of sentences to make inference more efficient
- **Definite clause:** disjunction of literals, of which exactly one is positive
- **Horn clause:** disjunction of literals, of which at most one literal is positive
  (at most one = 0 or 1)

- All definite clauses are Horn clauses.
- *Goal clauses* with no positive literals are also Horn clauses.
- Horn clauses closed under resolution:
  resolving 2 Horn clauses results in a Horn clause

Horn Clauses & Definite Clauses ..2

- Definite clauses can be written as an implication.
  - Premise (body) is a conjunction of literals
  - Conclusion (tail) is a single positive literal

  e.g. $A \wedge B \wedge C \Rightarrow D$

- Facts, such as $B_{1,1}$ can be written as implications too, but simpler to write as literals.

- Inference can be done over Horn clauses with **forward chaining** or **backward chaining.**
- These algorithms that decide entailment run in linear time!

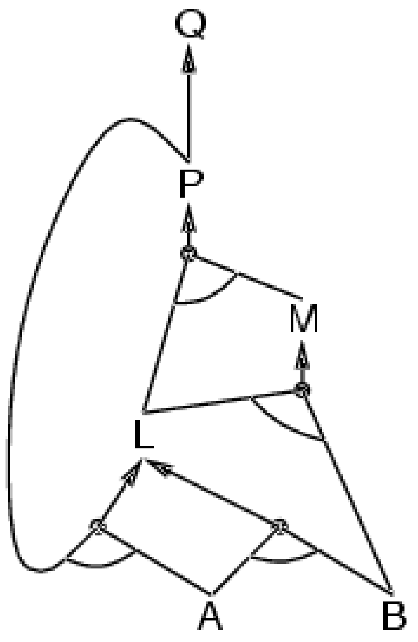
Forward chaining

Idea: fire any rule whose premises are satisfied in the *KB*,

- add its conclusion to the *KB*, until query is found

Graph written as an AND-OR graph

$$P \Rightarrow Q$$
$$L \land M \Rightarrow P$$
$$B \land L \Rightarrow M$$
$$A \land P \Rightarrow L$$
$$A \land B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining algorithm

Forward chaining is sound (applications of Modus Ponens)
and complete.

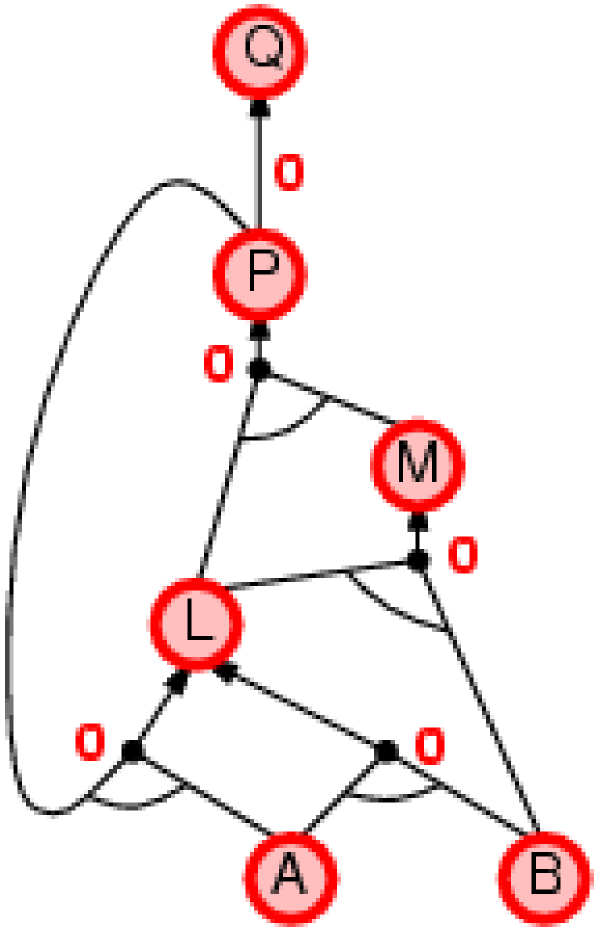
Example of data - driven reasoning; need some control over irrelevant consequences

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                   inferred, a table, indexed by symbol, each entry initially false
                   agenda, a list of symbols, initially the symbols known to be true

  while agenda is not empty do
      p ← POP(agenda)
      unless inferred[p] do
          inferred[p] ← true
          for each Horn clause c in whose premise p appears do
              decrement count[c]
              if count[c] = 0 then do
                  if HEAD[c] = q then return true
                  PUSH(HEAD[c], agenda)
  return false
```

Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining

Idea: work backwards from the query *q*:

to prove *q* by backward chaining (BC),

check if *q* is known already, or

prove by BC all premises of some rule concluding *q*

(creates subgoals)

Avoid loops: check if new subgoal is already on the goal stack

Avoid repeated work: check if new subgoal

1. has already been proved true, or
2. has already failed

Form of goal-directed reasoning. Runs in linear time or less.

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

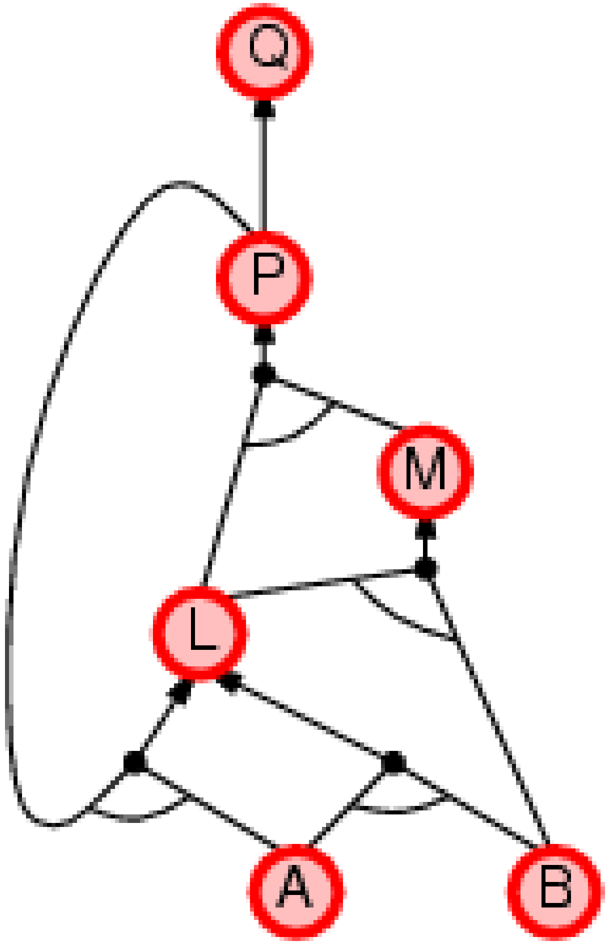
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

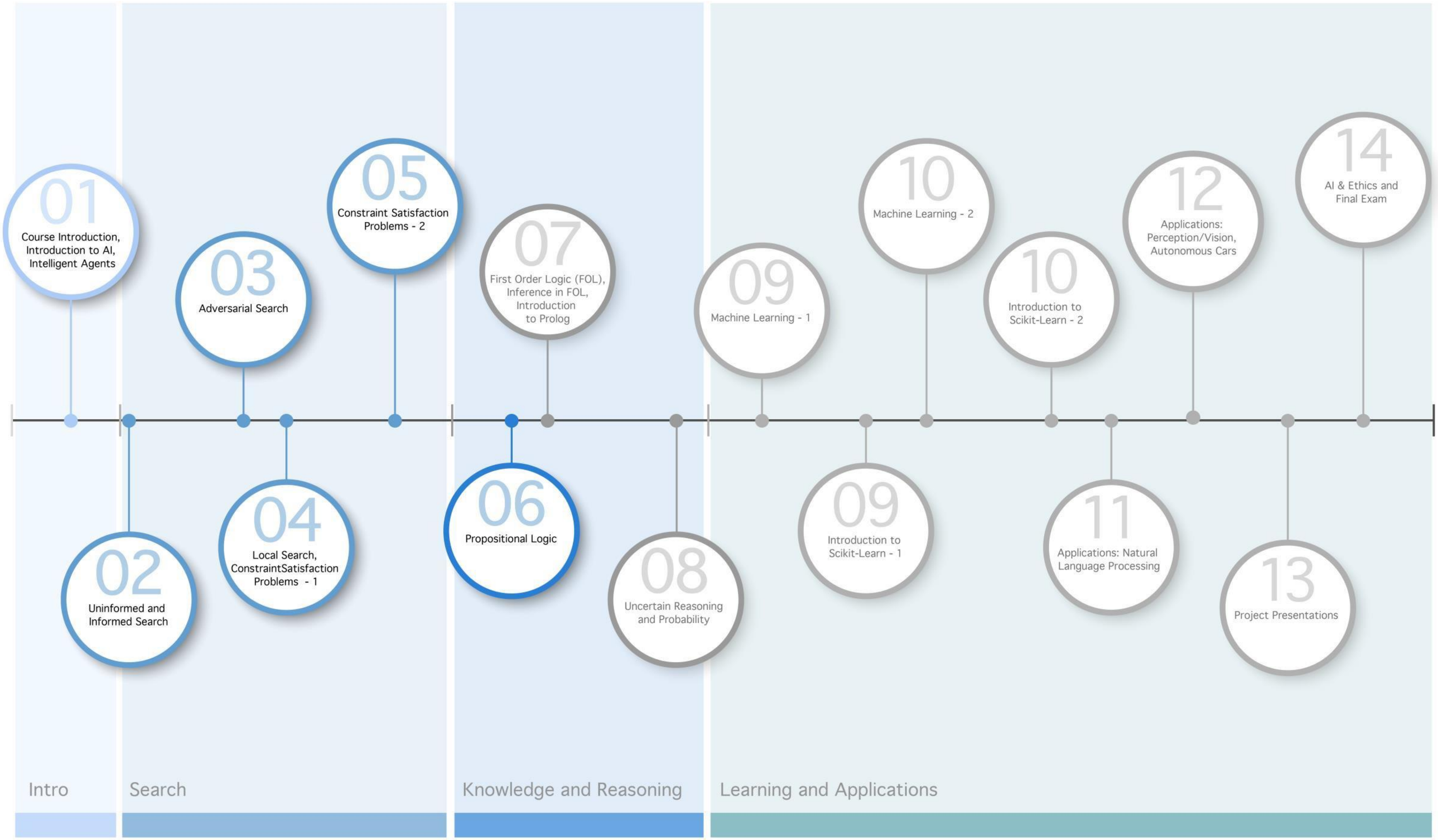


Limitations of Propositional Logic

How would you say:
“All the days in August in Seattle are warm” ?

Or
“At least one student in this course likes cooking and dislikes exams” ?

For that, we turn to First Order Logic


01
Course Introduction, Introduction to AI, Intelligent Agents
02
Uninformed and Informed Search
03
Adversarial Search
04
Local Search, ConstraintSatisfaction Problems - 1
05
Constraint Satisfaction Problems - 2
06
Propositional Logic
07
First Order Logic (FOL), Inference in FOL, Introduction to Prolog
08
Uncertain Reasoning and Probability
09
Machine Learning - 1
09
Introduction to Scikit-Learn - 1
10
Machine Learning - 2
10
Introduction to Scikit-Learn - 2
11
Applications: Natural Language Processing
12
Applications: Perception/Vision, Autonomous Cars
13
Project Presentations
14
AI & Ethics and Final Exam
Intro
Search
Knowledge and Reasoning
Learning and Applications