

CS480 – INTRODUCTION TO ARTIFICIAL INTELLIGENCE

TOPIC: LOGICAL AGENTS
CHAPTER: 7



Mustafa Bilgic



<http://www.cs.iit.edu/~mbilgic>



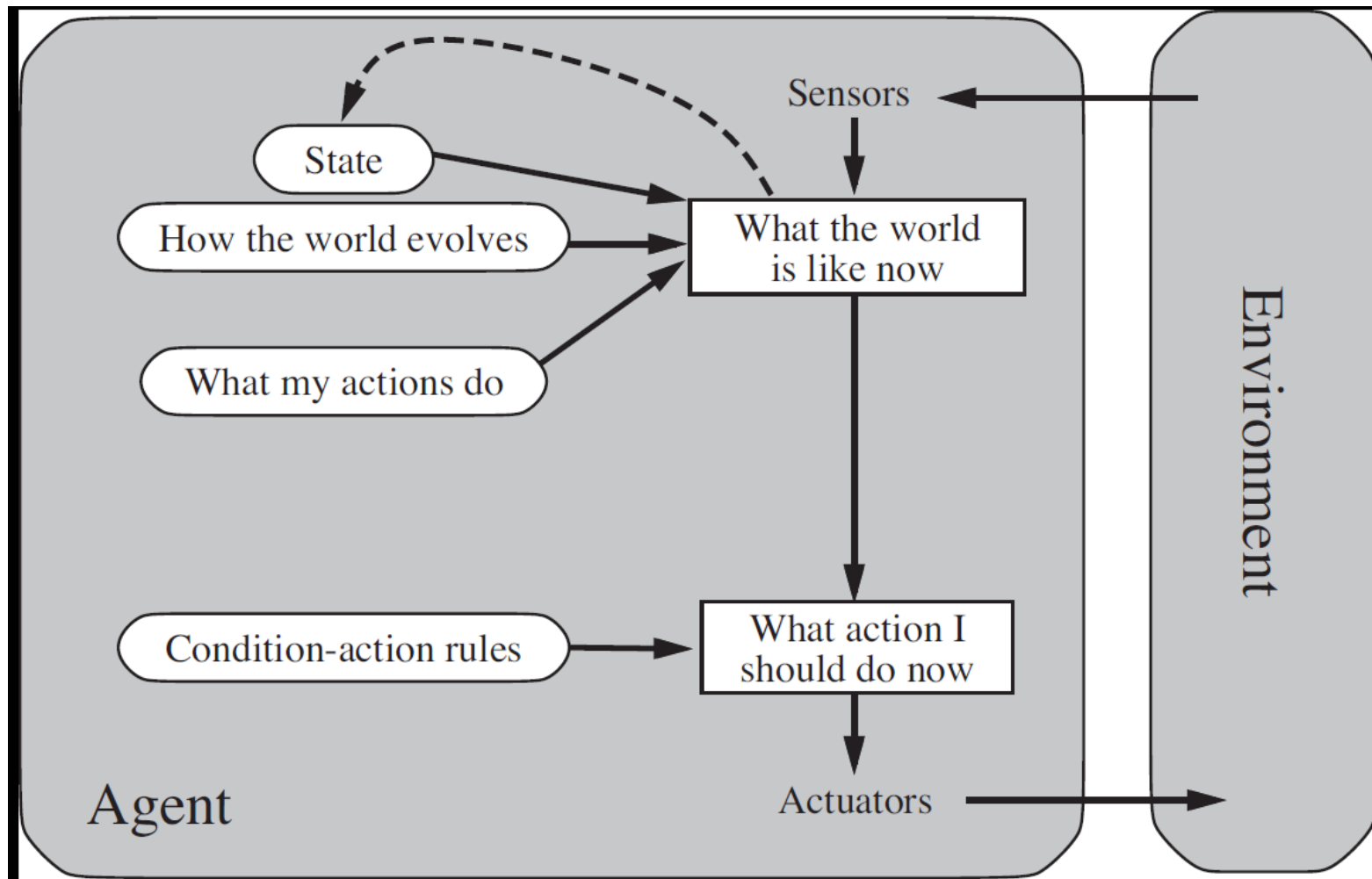
<https://twitter.com/bilgicm>

HUMANLY VS. RATIONALLY & THINKING VS. ACTING

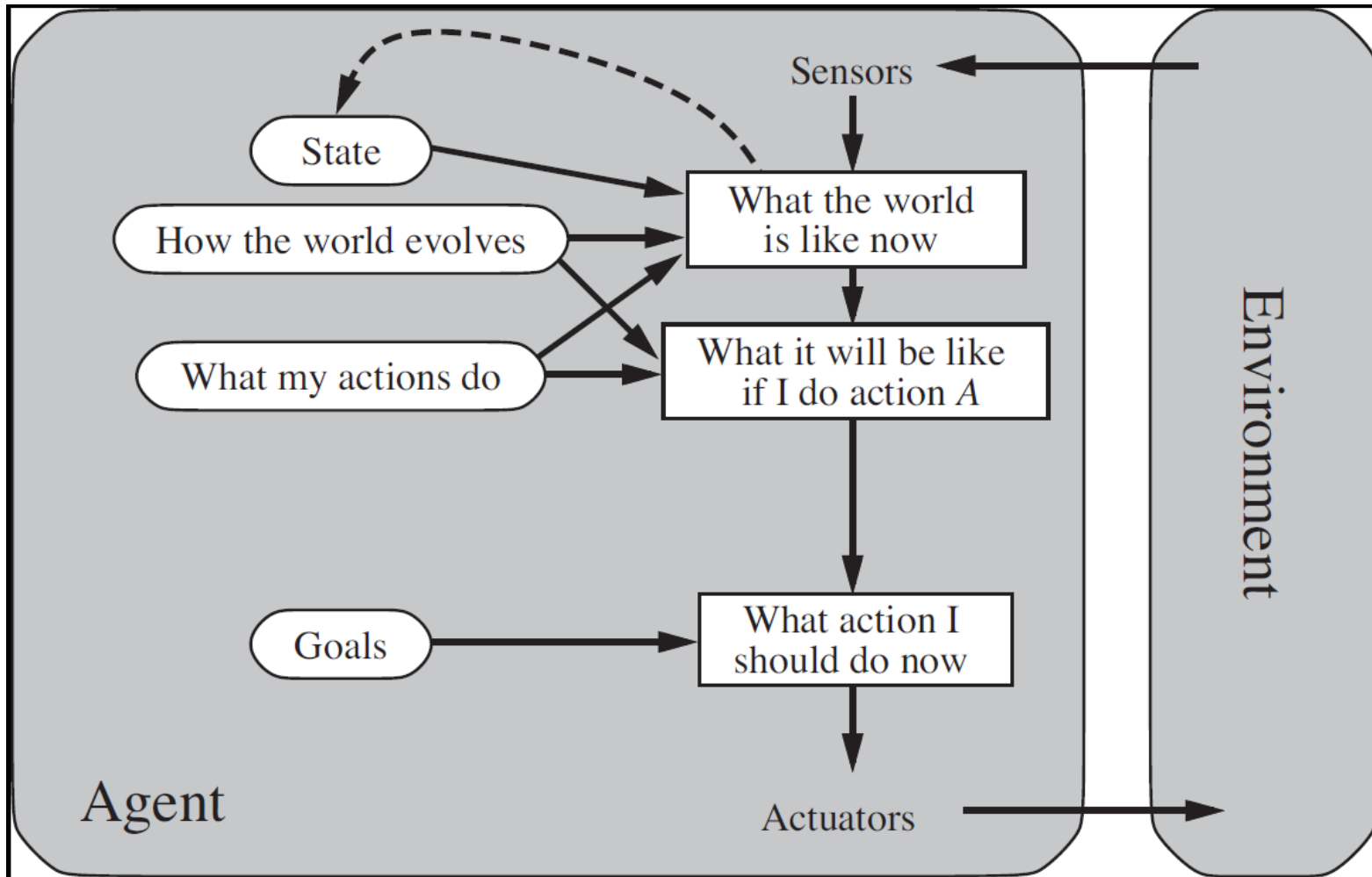
	Humanly	Rationally
Think	Thinking humanly	Thinking rationally
Act	Acting humanly	Acting rationally

- This class: acting rationally.
- Acting rationally requires thinking rationally.
- This lecture: thinking rationally.

MODEL-BASED REFLEX AGENTS



MODEL-BASED & GOAL-BASED AGENT



KNOWLEDGE-BASED AGENTS

- **Knowledge base (KB): A set of sentences**
- **Sentence**
 - Expressed in a **knowledge representation language**
 - Represents some assertion about the world

INFERENCE

- Given a KB, we would like to
 - Generate new sentences and them to the KB
 - Ask what is known
- This is called **inference**
- Adding new sentences: **TELL**
- Querying KB: **ASK**

KB-AGENT

function KB-AGENT(*percept*) **returns** an *action*

persistent: *KB*, a knowledge base

t, a counter, initially 0, indicating time

TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))

action \leftarrow ASK(*KB*, MAKE-ACTION-QUERY(*t*))

TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))

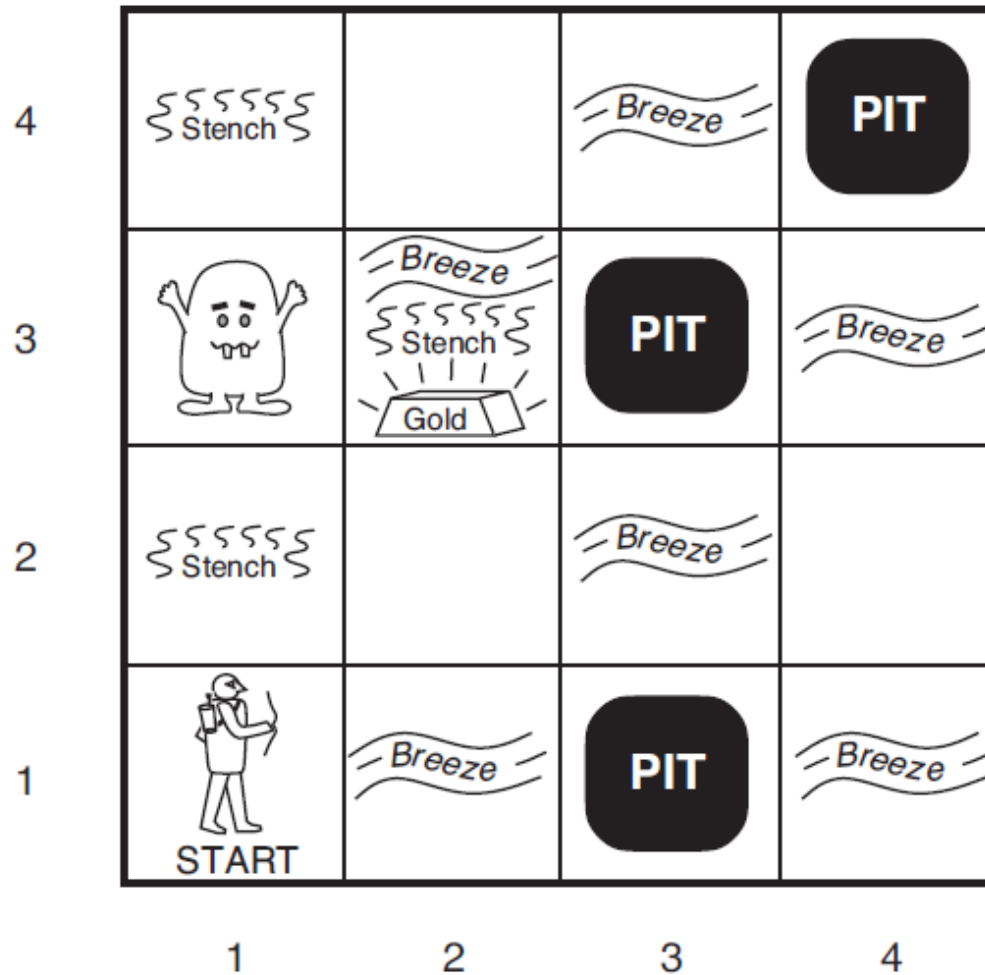
t \leftarrow *t* + 1

return *action*

RUNNING EXAMPLE – THE WUMPUS WORLD

- The Wumpus computer game
- The agent explores a cave consisting of rooms connected by passageways.
- Lurking somewhere in the cave is the Wumpus, a beast that eats any agent that enters its room.
- Some rooms contain bottomless pits that trap any agent that wanders into the room.
- Occasionally, there is a heap of gold in a room.
- The goal is to collect the gold and exit the cave without being eaten

A TYPICAL WUMPUS WORLD

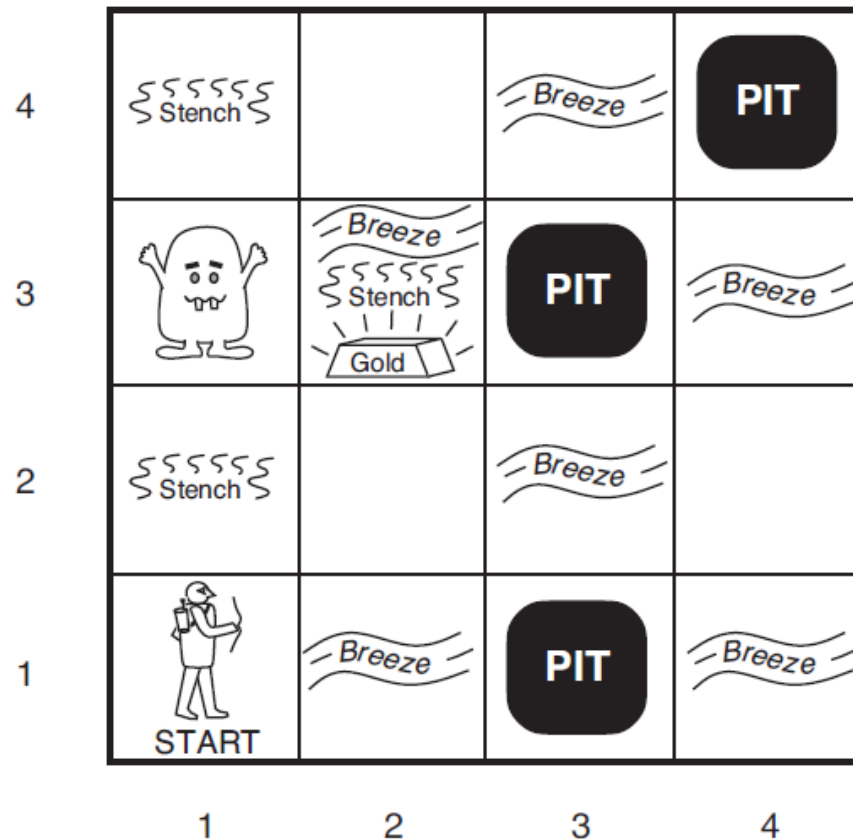


WUMPUS PEAS

- **Performance measure:**
gold +1000, death -1000, -1 per step, -10 use arrow
- **Environment:**
Squares adjacent to wumpus are smelly
Squares adjacent to pit are breezy
Glitter iff gold is in the same square
Bump iff move into a wall
Shooting kills wumpus if you are facing it
Woeful scream iff the wumpus is killed
Shooting uses up the only arrow
Grabbing picks up gold if in same square
- **Actuators:** Left turn, Right turn, Forward, Grab, Shoot
- **Sensors:** Stench, Breeze, Glitter, Bump, Scream

A TYPICAL WUMPUS WORLD

- Start at [1,1]
- The task is to find the gold, grab it, return to [1,1], and climb out



ENVIRONMENT CHARACTERISTICS

- Observable?
 - No, only local perception
- Deterministic?
 - Yes, outcome exactly specified
- Episodic or Sequential?
 - Sequential
- Static?
 - Yes, the Wumpus and pits do not move
- Discrete?
 - Yes
- Single-agent?
 - Yes, Wumpus acts as a feature

THE FIRST STEPS

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2	2,2	3,2	4,2
OK			
1,1	2,1	3,1	4,1
A OK	OK		

A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus

- Percept: {Stench?, Breeze?, Glitter?, Bump?, Scream?}
- The first percept is {None, None, None, None, None}

THE NEXT STEP

- Forward
 - The agent is in {2,1}.
- The percept is
 - {None, Breeze, None, None, None}
- What can we infer?

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 <div style="border: 1px solid black; display: inline-block; padding: 2px;">A</div> B OK	3,1 P?	4,1

THE NEXT STEPS

- Left turn, left turn,
forward, forward,
right turn, forward
 - At [1,2]
- The percept is
{Stench, None,
None, None, None}
- What can we infer?

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

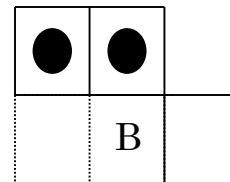
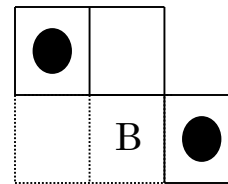
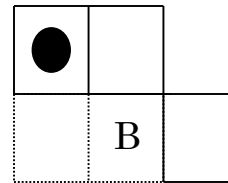
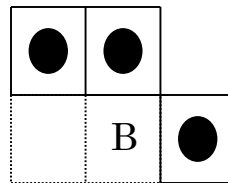
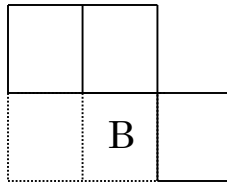
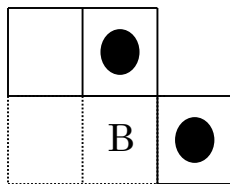
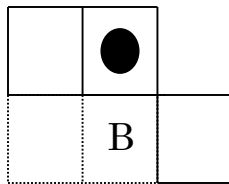
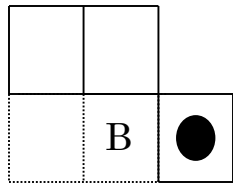
LOGIC

- A formal language
 - **Syntax** – what expressions are legal (well-formed)
 - **Semantics** – what legal expressions mean
 - in logic the truth of each sentence with respect to each possible world.
- E.g., the language of arithmetic
 - $x+y = 4$ is a sentence, $x4+y=$ is not a sentence
 - $x+y = 4$ is true in a world where $x=1$ and $y=3$
 - $x+y = 4$ is false in a world where $x=0$ and $y=6$

MODEL

- A **model** is a mathematical abstraction of a possible world
- Very much like arithmetic, logic expressions (sentences), contain variables
 - A possible world (a model), is an assignment of values to those variables
- If a sentence α is true in a model m , then we say m **satisfies** α , or m **is a model of** α
 - $M(\alpha)$: The set of all models of α
- Example:
 - $x+y = 4$, where x and y are non-negative integers, is a sentence
 - What are the possible worlds (models) in this domain?
 - Which model(s) satisfy this sentence, i.e., what is $M(x+y=4)$?

MODEL EXAMPLE – PIT? IN [3,1], [1,2], [2,2]



Why do we have eight models?
Which model(s) satisfy "No Pit in [1,2]"?

LOGICAL REASONING – ENTAILMENT

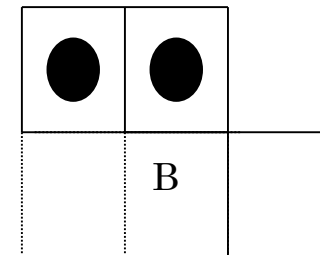
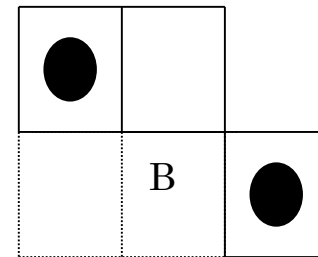
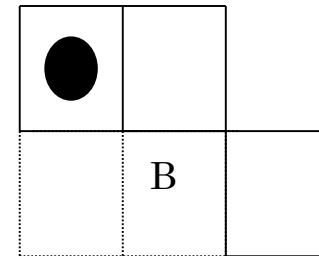
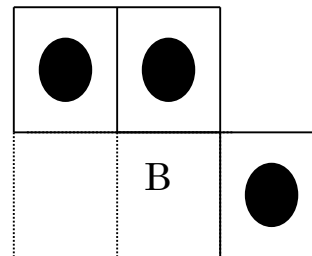
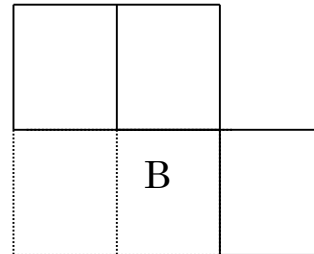
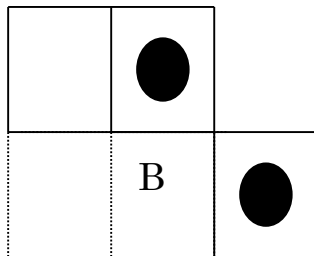
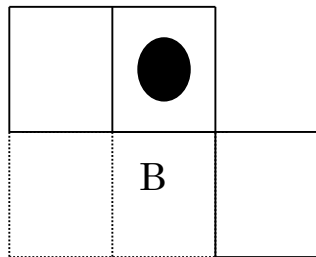
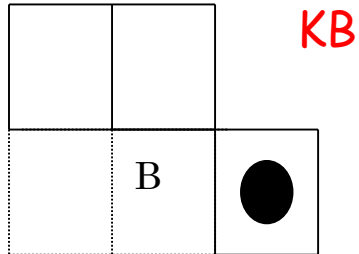
- We would like to both
 - Add new sentences
 - Ask queries
- Does a sentence β *follow logically* from another sentence α ? Does α **entail** β ?
 - $\alpha \models \beta$?
- $\alpha \models \beta$ if and only if, in every model in which α is true, β is also true.

$$M(\alpha) \subseteq M(\beta)$$

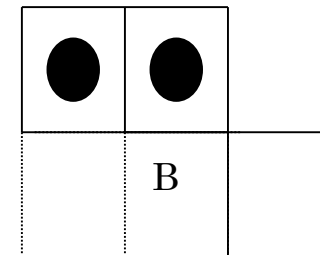
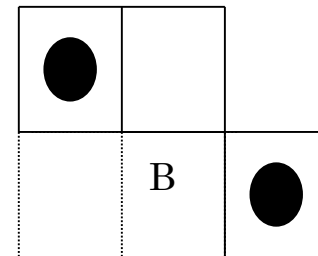
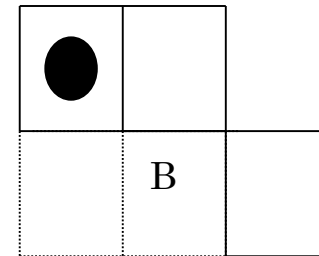
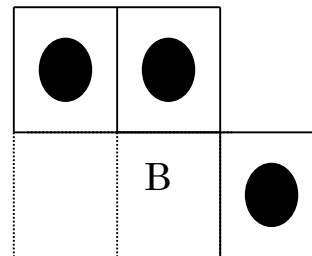
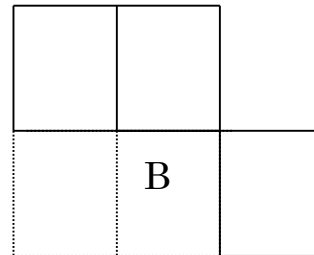
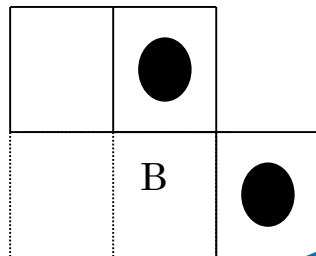
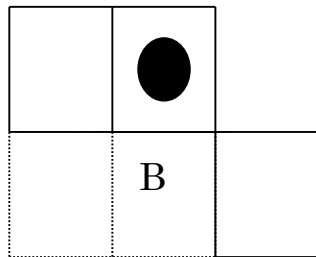
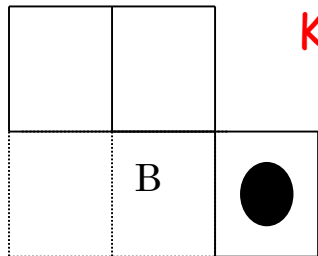
KB \models A SENTENCE?

- The percepts, combined with the knowledge of how the world works, constitute the KB
- KB can be thought of a set of sentences *or* a giant sentence that asserts all the individual sentences
 - Thus, it makes perfect sense to ask whether KB is true or false *and* whether it entails another sentence
- KB is false in models that contradict what the agent knows

M(KB) IN THE WUMPUS WORLD

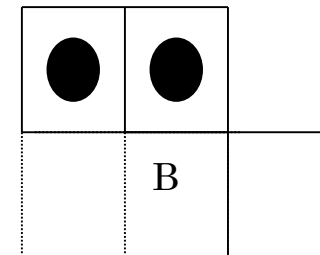
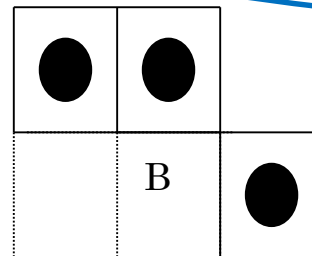
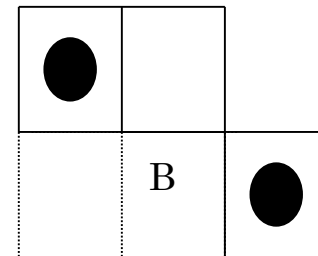
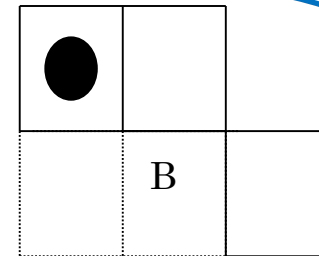
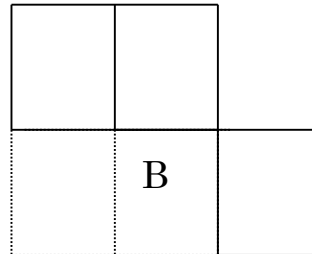
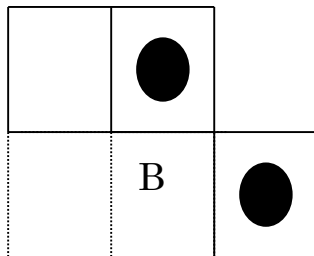
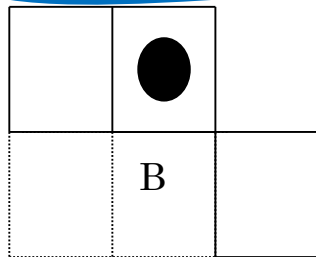
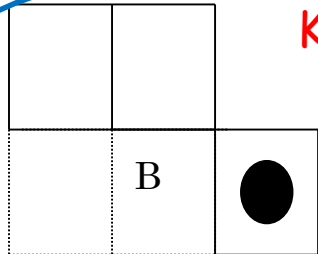


WHAT IS $M(\text{No PIT in } [1,2])$?



Does KB entail No Pit in $[1,2]$?

WHAT IS $M(\text{No PIT in } [2,2])$?



Does KB entail No Pit in $[2,2]$?

LOGICAL INFERENCE

- The notion of entailment can be used for logical inference
 - Model checking: enumerate all possible models and check whether α is true.
- If an algorithm derives only entailed sentences it is called **sound** or **truth preserving**
 - Otherwise it just makes things up.
- **Completeness** : the algorithm can derive any sentence that is entailed.

i is sound if whenever $KB \models_i \alpha$ it is also true that $KB \models \alpha$

i is complete if whenever $KB \models \alpha$ it is also true that $KB \models_i \alpha$

WE'LL COVER TWO TYPES OF LOGIC

- Propositional logic
 - $A \wedge B \Rightarrow C$
- First-order logic
 - $(\forall x)(\exists y) \text{Mother}(y,x)$

PROPOSITIONAL LOGIC LANGUAGE

- Propositional symbols
 - P, Q, R, \dots
 - True, False
- Connectives
 - $\neg, \wedge, \vee, \Rightarrow, \Leftarrow, \Leftrightarrow$

MORE ON THE CONNECTIVES

- \neg (negation)
- \wedge (and): $P \wedge Q$ is called a **conjunction** and P and Q are **conjuncts**
- \vee (or): $P \vee Q$ is called a **disjunction** and P and Q are **disjuncts**
- \Rightarrow (implies): $P \Rightarrow Q$ is called an **implication**. P is the **premise** or **antecedent** and Q is its **conclusion** or **consequent**
- \Leftrightarrow (if and only if): $P \Leftrightarrow Q$ is called a **biconditional**

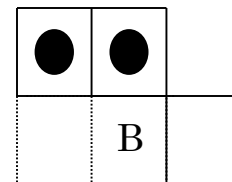
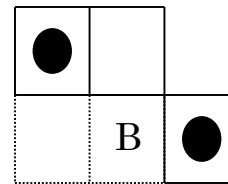
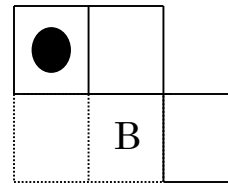
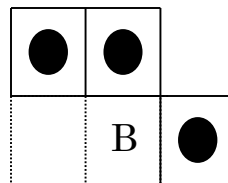
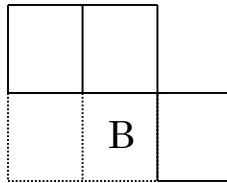
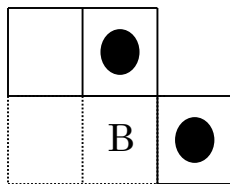
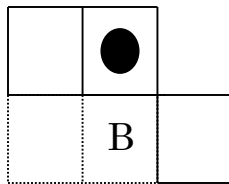
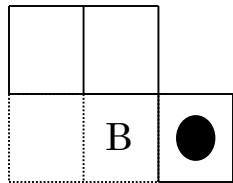
SYNTAX

- Sentence \rightarrow AtomicSentence | ComplexSentence
- AtomicSentence \rightarrow True | False | P | Q | R | ...
- ComplexSentence \rightarrow
 - \neg sentence
 - sentence \wedge sentence
 - sentence \vee sentence
 - sentence \Rightarrow sentence
 - sentence \Leftrightarrow sentence
- Operator precedence: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

SEMANTICS

- Semantics define the rules for determining the truth of a sentence with respect to a **model**
- A model fixes the truth value of for every proposition
- Going back to the Wumpus world

MODEL EXAMPLE – PIT? IN [3,1], [1,2], [2,2]



With n propositions, how many possible models are there?

SEMANTICS OF PROPOSITIONAL LOGIC

- It specifies how to determine the truth value of any sentence in a model m
- The truth value of *True* is *True*
- The truth value of *False* is *False*
- The truth value of each atomic sentence is given by m
- The truth value of every other sentence is obtained recursively by using truth tables



TRUTH RULES

- $\neg P$ is true iff P is false
- $P \wedge Q$ is true iff both P and Q are true
- $P \vee Q$ is true iff either P or Q or both are true
- $P \Rightarrow Q$ is true unless P is true and Q is false
- $P \Leftrightarrow Q$ is true iff P and Q have equal truth values;
i.e., iff both P and Q are true or iff both P and Q
are false

TRUTH TABLES

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T					
T	F					
F	T					
F	F					

NEGATION

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F				
T	F	F				
F	T	T				
F	F	T				

AND

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F	T			
T	F	F	F			
F	T	T	F			
F	F	T	F			

OR

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F	T	T		
T	F	F	F	T		
F	T	T	F	T		
F	F	T	F	F		

IMPLICATION

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F	T	T	T	
T	F	F	F	T	F	
F	T	T	F	T	T	
F	F	T	F	F	T	

BICONDITIONAL

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

POSSIBLE CONFUSION ABOUT \Rightarrow

- $P \Rightarrow Q$ is true unless P is true and Q is false
- If 5 is odd, then 7 is odd. T or F?
- If 5 is odd, then 10 is odd. T or F?
- If 5 is even, then 7 is even. T or F?
- If 5 is even, then 7 is odd. T or F?
- If 5 is even, then 10 is even. T or F?
- If 5 is even, then Chicago is the capital of US. T or F?
- **The key: Logic does not assume any causation or correlation between P and Q .**

If 5 is odd, then Prof. Bilgic is the best prof ever. T or F?

WUMPUS WORLD SENTENCES

- $P_{x,y}$ is true if there is a pit in $[x,y]$
- $W_{x,y}$ is true if there is a wumpus in $[x,y]$, dead or alive
- $B_{x,y}$ is true if the agent perceives a breeze in $[x,y]$
- $S_{x,y}$ is true if the agent perceives a stench in $[x,y]$
- “A square is breezy if and only if there is an adjacent pit”
 - $B_{11} \Leftrightarrow P_{12} \vee P_{21}$
 - $B_{21} \Leftrightarrow ???$
- “A square is stenchy if and only if there is an adjacent wumpus”
- A few others for Glitter, Bump, Scream, wumpus being dead or alive, etc.

THE FIRST TWO STEPS

- $R_1: \neg P_{1,1}$
- $R_2: B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$
- $R_3: B_{2,1} \Leftrightarrow P_{1,1} \vee P_{2,2} \vee P_{3,1}$
- $R_4: \neg B_{1,1}$
- $R_5: B_{2,1}$
- Does $KB \models \neg P_{1,2}$?

How can we check if KB entails $\neg P_{1,2}$?

ALGORITHMS

1. Model checking
2. Logical equivalence rules
3. Proof-by-contradiction
 - **Resolution**
4. Forward chaining
5. Backward chaining

MODEL CHECKING

- For all models of KB (i.e., for all possible worlds where KB is True), $\neg P_{1,2}$ has to be True.
- We have the following symbols:
 - $B_{1,1}, B_{2,1}, P_{1,1}, P_{1,2}, P_{2,1}, P_{2,2}, P_{3,1}$
- How many possible worlds?

MODEL CHECKING

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
T	T	T	T	T	T	T						
...												
F	T	F	F	T	F	F						
F	T	F	F	F	T	T						
F	T	F	F	F	T	F						
F	T	F	F	F	F	T						
F	T	F	F	F	F	F						
...												
F	F	F	F	F	F	F						

LOGICAL EQUIVALENCE

○ Commutativity

- $\alpha \wedge \beta \equiv \beta \wedge \alpha$
- $\alpha \vee \beta \equiv \beta \vee \alpha$

○ Associativity

- $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$
- $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$

○ Double negation elimination

- $\neg(\neg\alpha) \equiv \alpha$

○ Contraposition

- $\alpha \Rightarrow \beta \equiv \neg \beta \Rightarrow \neg \alpha$

LOGICAL EQUIVALENCE CONT.

- Implication elimination

- $\alpha \Rightarrow \beta \equiv \neg\alpha \vee \beta$

- Biconditional elimination

- $\alpha \Leftrightarrow \beta \equiv (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

- De Morgan

- $\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$

- $\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$

- Distributivity \wedge of over \vee

- $\alpha \wedge (\beta \vee \gamma) \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$

- Distributivity of \vee over \wedge

- $\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$

INFERENCE RULES

○ Modus Ponens

- Given
 - $\alpha \Rightarrow \beta$
 - α
- Conclude
 - β

○ And-Elimination

- Given
 - $\alpha \wedge \beta$
- Conclude
 - α
 - β

PROVE $\neg P_{1,2}$ FROM R_1 THROUGH R_5

- $R_1: \neg P_{1,1}$
- $R_2: B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$
- $R_3: B_{2,1} \Leftrightarrow P_{1,1} \vee P_{2,2} \vee P_{3,1}$
- $R_4: \neg B_{1,1}$
- $R_5: B_{2,1}$

META-SOLUTION

- R6 = biconditional elimination from R2
- R7 = and-elimination from R6
- R8 = contraposition of R7
- R9 = modes ponens from R8 and R4
- R10 = de morgan of R9

Note: We haven't used R1, R3, and R5. We used only R2 and R4.
That is, we haven't touched the symbols $B_{2,1}$, $P_{1,1}$, $P_{2,2}$, and $P_{3,1}$ at all.
Vanilla model checking would include these symbols.

SEARCHING FOR PROOFS

- **Initial state:** the initial knowledge base
- **Actions:** apply one of the inference rules
- **Result:** add the derived sentence
- **Goal:** the sentence we are trying to prove

VALID

- A sentence is **valid** if it is true in *all* models
 - E.g, True, $P \vee \neg P$, $\text{False} \Rightarrow P$
- **Deduction theorem:**
 - $\alpha \models \beta$ if and only if the sentence $\alpha \Rightarrow \beta$ is valid
 - How would you prove it?
 - To check $\text{KB} \models \alpha$, we need to check if $\text{KB} \Rightarrow \alpha$ is valid

SATISFIABILITY

- A sentence is **satisfiable** if it is true in, or satisfied by, *some* model
- How can you check satisfiability?
 - By enumerating all possible models until one is found
- Checking for satisfiability, SAT, is NP-complete

SATISFIABILITY AND VALIDITY

- Satisfiability and validity
 - A sentence α is valid iff $\neg\alpha$ is unsatisfiable
- Deduction theorem revisited
 - $\alpha \models \beta$ iff the sentence $\alpha \Rightarrow \beta$ is valid
 - $\alpha \not\models \beta$ iff the sentence $\neg(\alpha \Rightarrow \beta)$ is unsatisfiable
 - $\alpha \models \beta$ iff the sentence $\alpha \wedge \neg\beta$ is unsatisfiable

ALGORITHMS

1. Model checking
2. Logical equivalence rules
3. Proof-by-contradiction
 - **Resolution**
4. Forward chaining
5. Backward chaining

PROOF BY CONTRADICTION

- To prove β entails from α , check the unsatisfiability of $\alpha \wedge \neg\beta$
 - Assume α is True
 - Assume β is False (i.e., assume $\neg\beta$ is True)
 - Show that this leads to a contradiction (i.e., it leads to False)
- To prove β entails from KB, check the unsatisfiability of $KB \wedge \neg\beta$
 - The agent already knows KB
 - The agent now assumes it knows $\neg\beta$
 - And then the agent arrives at a contradiction

RESOLUTION

○ Unit Resolution

- Given

1. $l_1 \vee \dots \vee l_{i-1} \vee l_i \vee l_{i+1} \vee \dots \vee l_k$

2. u , where u and l_i are complementary (one is the negation of the other)

- Conclude

- $l_1 \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k$

- Example

- From R1: $P \vee \neg Q$ and R2: Q conclude R3: ?

RESOLUTION

○ Full Resolution

- Given

1. $l_1 \vee \dots \vee l_{i-1} \vee \mathbf{l_i} \vee l_{i+1} \vee \dots \vee l_k$

2. $u_1 \vee \dots \vee u_{j-1} \vee \mathbf{u_j} \vee u_{j+1} \vee \dots \vee u_n$, where $\mathbf{u_j}$ and $\mathbf{l_i}$ are complementary (one is the negation of the other)

- Conclude

- $l_1 \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k \vee u_1 \vee \dots \vee u_{j-1} \vee u_{j+1} \vee \dots \vee u_n$

- Example

- From R1: $P \vee Q$ and R2: $\neg Q \vee \neg S$ conclude R3=?

A FEW EXAMPLES OF RESOLUTION

○ Given:

- $P \vee Q$
- $\neg P$

○ Given:

- $P \vee Q \vee R$
- $\neg Q \vee S$

○ Given:

- $P \vee Q \vee R$
- $\neg Q \vee R$

○ Given:

- P
- $\neg P$

○ Given:

- $P \vee Q$
- $\neg P \vee \neg Q$

○ Given:

- $P \vee Q \vee \neg R$
- $\neg Q \vee R$

CONJUNCTIVE NORMAL FORM

- The resolution applies to clauses, i.e., disjunctions (\vee) of literals
- Every sentence of propositional logic can be converted into **conjunctive normal form** (CNF)
- CNF is conjunction (\wedge) of disjunctions (\vee)
- Are the following in CNF form?
 - $P \Rightarrow Q$?
 - $P \vee Q$?
 - $P \wedge Q$?
 - $(P \vee Q) \wedge (R \vee S)$?
 - $(P \wedge Q) \vee (R \wedge S)$?

CONVERSION TO CNF

1. Eliminate \Leftrightarrow
2. Eliminate \Rightarrow
3. Move \neg inwards
4. Distribute \vee over \wedge

EXAMPLE CONVERSION

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

RESOLUTION ALGORITHM

- To prove $KB \models \alpha$, prove unsatisfiability of $KB \wedge \neg\alpha$
- First convert $KB \wedge \neg\alpha$ into CNF
- Then apply resolution until
 - No new clauses can be added
 - KB does not entail α
 - Empty clause is generated, i.e., contradiction
 - KB entails α

THE RESOLUTION ALGORITHM

Sound and complete

function PL-RESOLUTION(KB, α) **returns** *true* or *false*

inputs: KB , the knowledge base, a sentence in propositional logic
 α , the query, a sentence in propositional logic

$clauses \leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg\alpha$

$new \leftarrow \{ \}$

loop do

for each pair of clauses C_i, C_j **in** $clauses$ **do**

$resolvents \leftarrow$ PL-RESOLVE(C_i, C_j)

if $resolvents$ contains the empty clause **then return** *true*

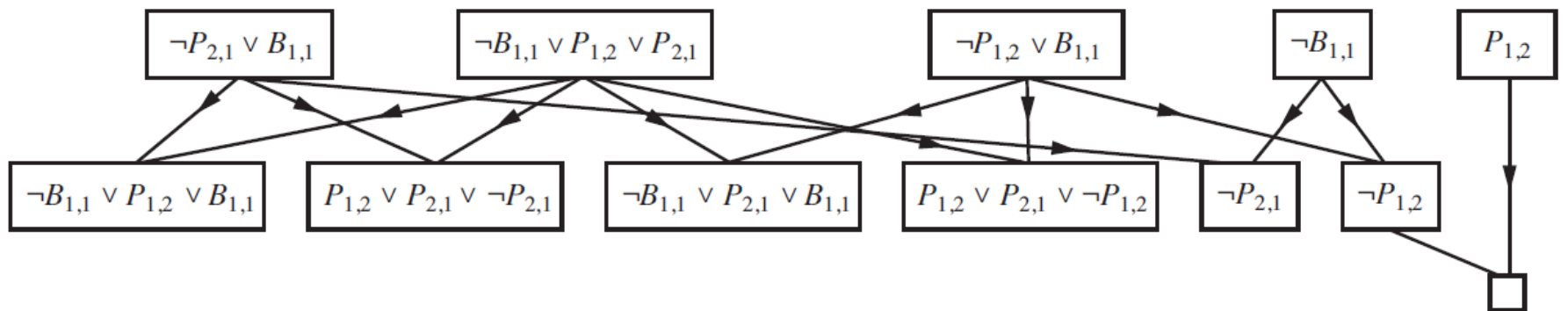
$new \leftarrow new \cup resolvents$

if $new \subseteq clauses$ **then return** *false*

$clauses \leftarrow clauses \cup new$

AN APPLICATION OF RESOLUTION

- No Breeze in [1,1]
- Prove that there is no pit in neighboring squares
- KB is: $(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$
- Prove: $\neg P_{1,2}$



A MORE RESTRICTED REPRESENTATION: HORN CLAUSES

- A **Horn clause** is a *disjunction* of literals of which at most one is positive
- Horn clauses are closed under resolution; if you resolve two Horn clauses, you get another Horn clause.
- All Horn clauses can be written as an implication whose premise is a conjunction of positive literals and whose conclusion is a single positive literal
- The premise is called the **body** and the conclusion is called the **head**
- A sentence consisting of a single positive literal is called a **fact**. It can be also written in an implication form. *How?*
- Sentences with no positive literals are called **goal clauses**. *How can we write them in implication form?*

INFERENCE WITH HORN CLAUSES

- Entailment can be decided in time that is linear in the size of the KB!
- 1. **Forward chaining**
- 2. **Backward chaining**

FORWARD CHAINING

- $KB \models \alpha$?
- Start with the facts
- If all the premises of an implication are known, then add its conclusion to the known facts
- Continue this process until α is added or no further inferences can be made
- This is an instance of **data-driven** reasoning

FORWARD CHAINING

Sound and complete

Linear

function PL-FC-ENTAILS?(KB, q) **returns** *true* or *false*

inputs: KB , the knowledge base, a set of propositional definite clauses

q , the query, a proposition symbol

$count \leftarrow$ a table, where $count[c]$ is the number of symbols in c 's premise

$inferred \leftarrow$ a table, where $inferred[s]$ is initially *false* for all symbols

$agenda \leftarrow$ a queue of symbols, initially symbols known to be true in KB

while $agenda$ is not empty **do**

$p \leftarrow \text{POP}(agenda)$

if $p = q$ **then return** *true*

if $inferred[p] = \text{false}$ **then**

$inferred[p] \leftarrow \text{true}$

for each clause c in KB where p is in c .PREMISE **do**

decrement $count[c]$

if $count[c] = 0$ **then** add c .CONCLUSION to $agenda$

return *false*

FORWARD CHAINING EXAMPLE

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

$$B \wedge L \Rightarrow M$$

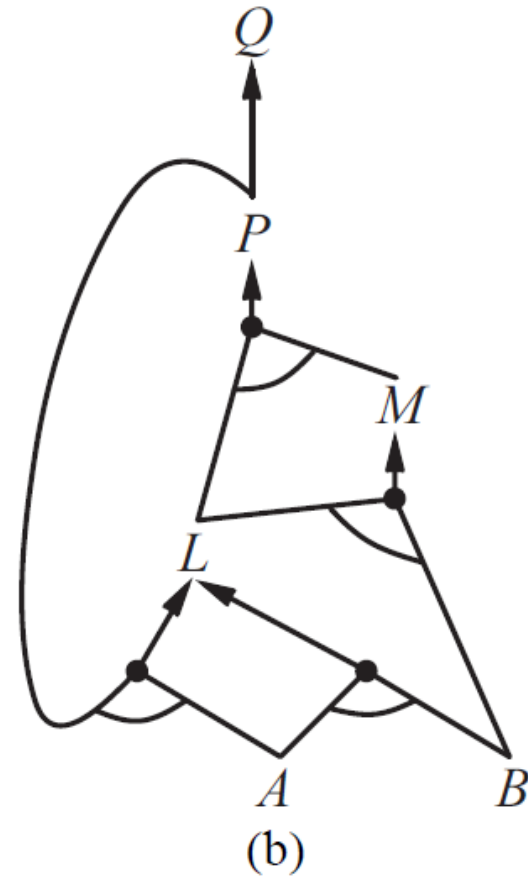
$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B

(a)

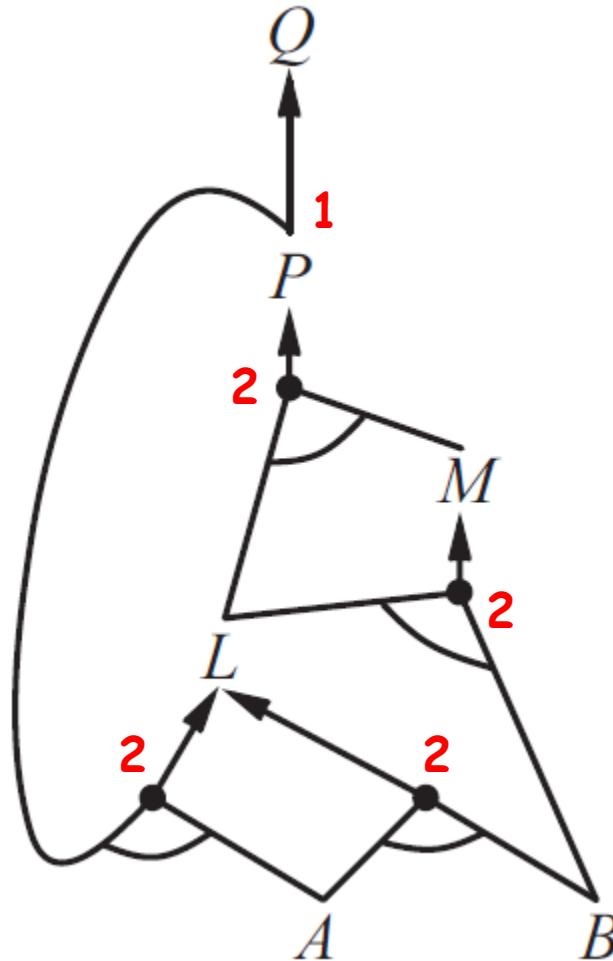


FORWARD CHAINING

Queue

A

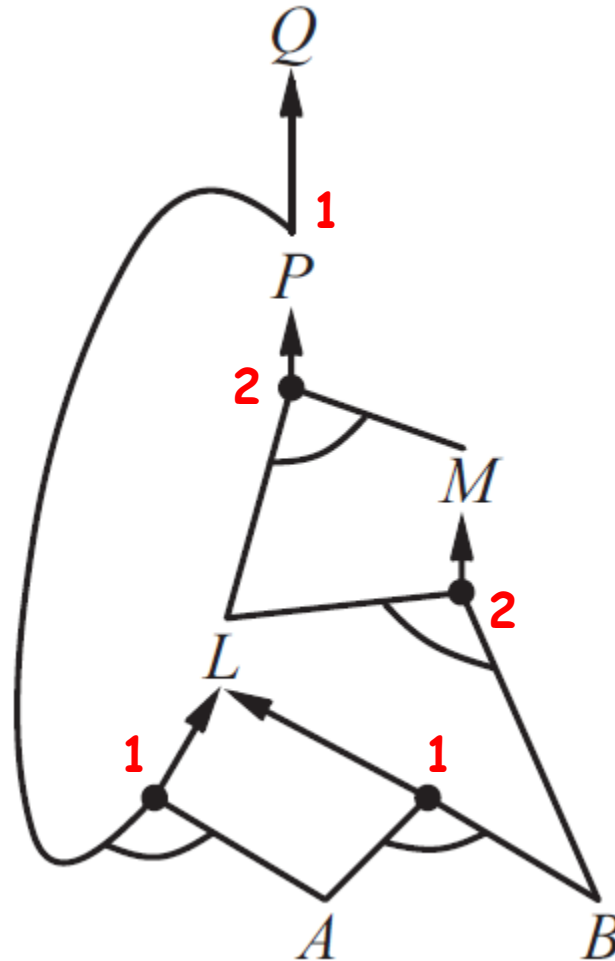
B



FORWARD CHAINING – POP A

Queue

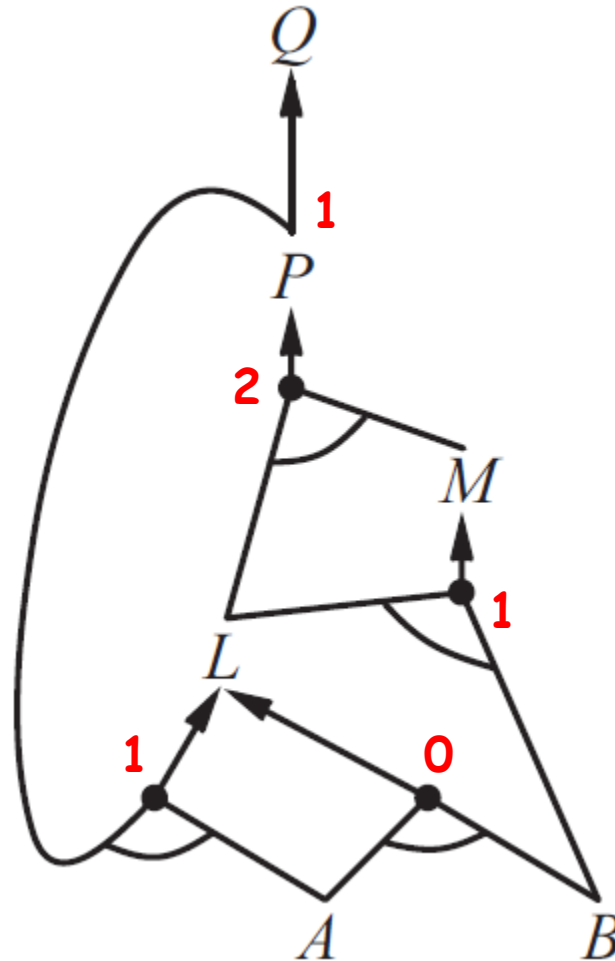
B



FORWARD CHAINING – POP B

Queue

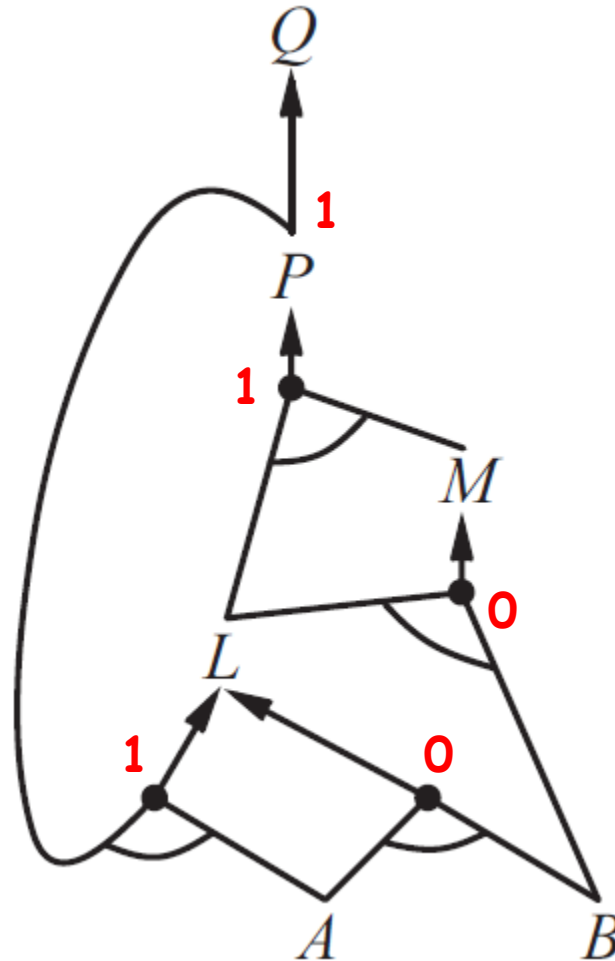
L



FORWARD CHAINING – POP L

Queue

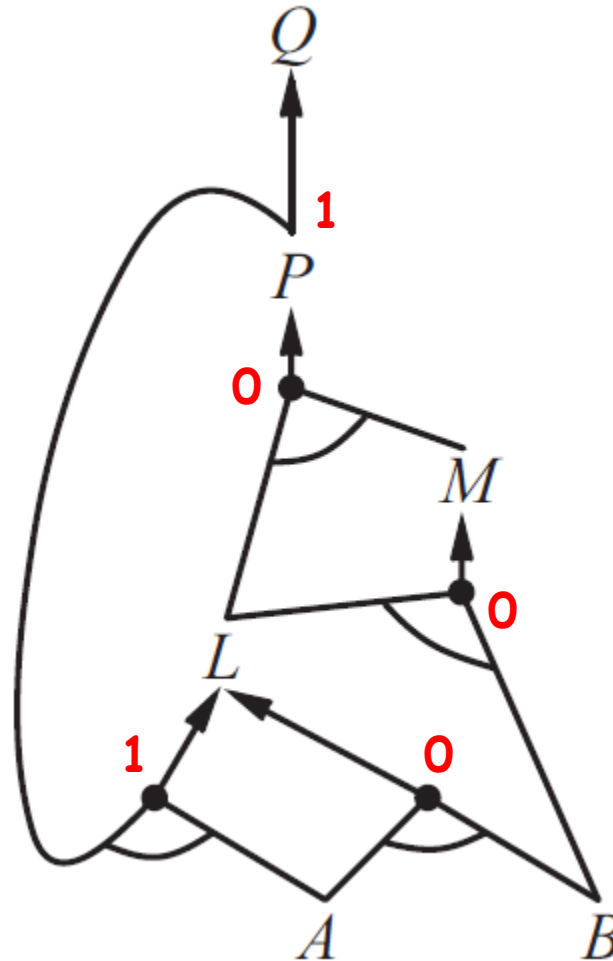
M



FORWARD CHAINING – POP M

Queue

P

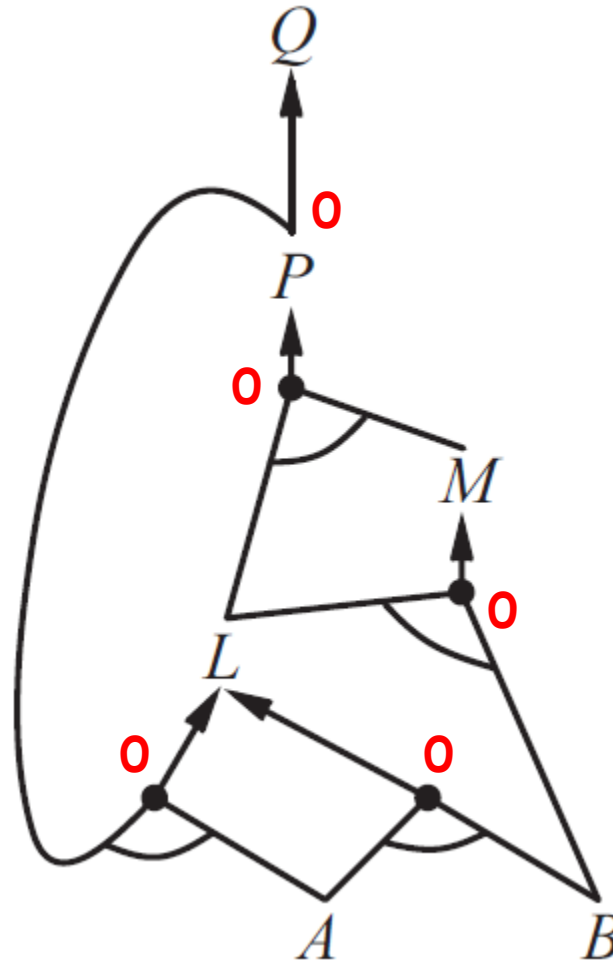


FORWARD CHAINING – POP P

Queue

Q

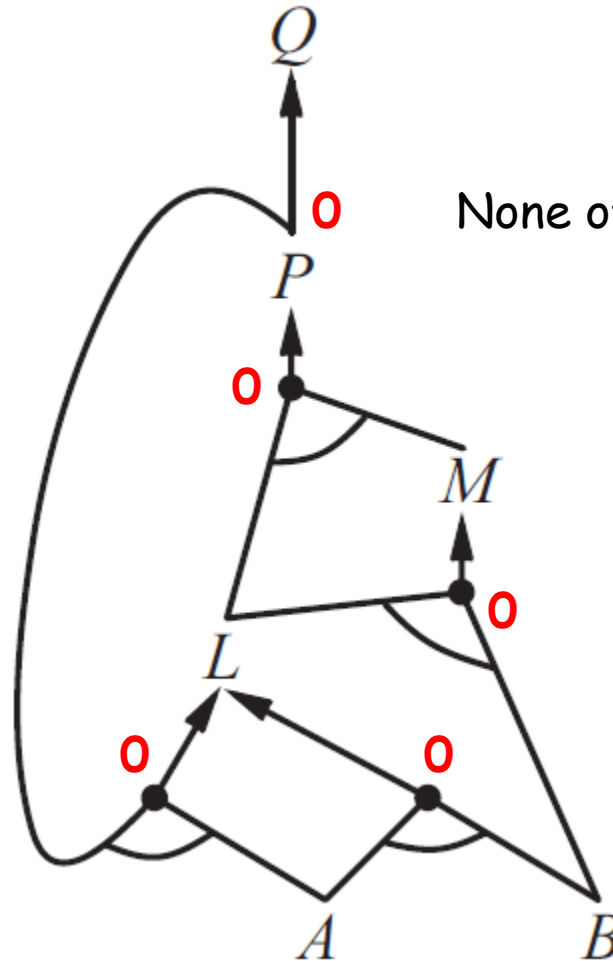
L



FORWARD CHAINING – POP Q

Queue

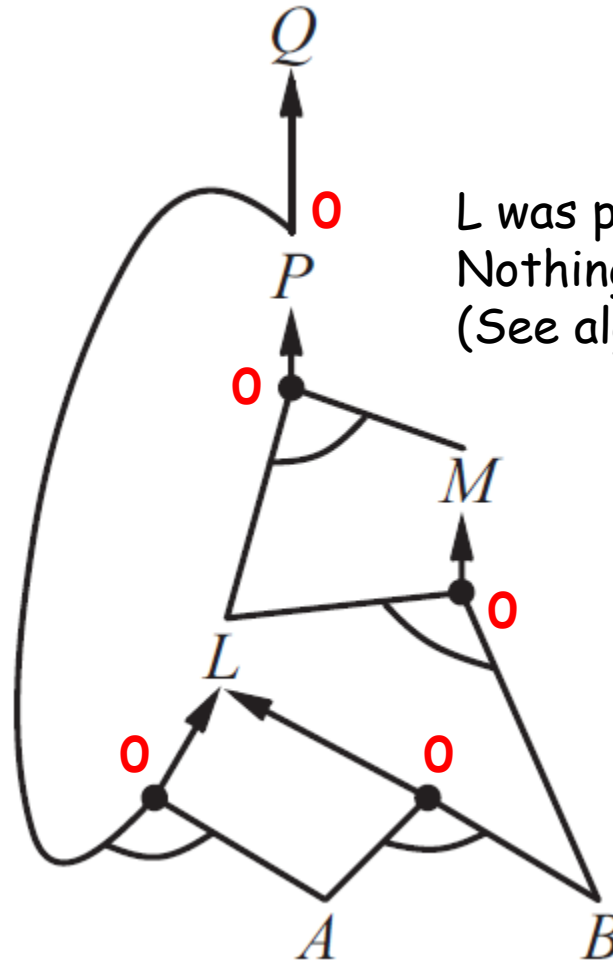
L



None of the counters change.

FORWARD CHAINING – POP L

Queue



L was processed before.
Nothing is done.
(See algorithm 7.15)

BACKWARD CHAINING

- Motivation: Need goal-directed reasoning to avoid getting overwhelmed with irrelevant consequences
- Main idea:
 - Work backwards from query α
 - To prove α :
 - Check if α is known already
 - Prove by backward chaining all premises of some rule concluding α
- This is an instance of **goal-driven** reasoning



BACKWARD CHAINING EXAMPLE

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

$$B \wedge L \Rightarrow M$$

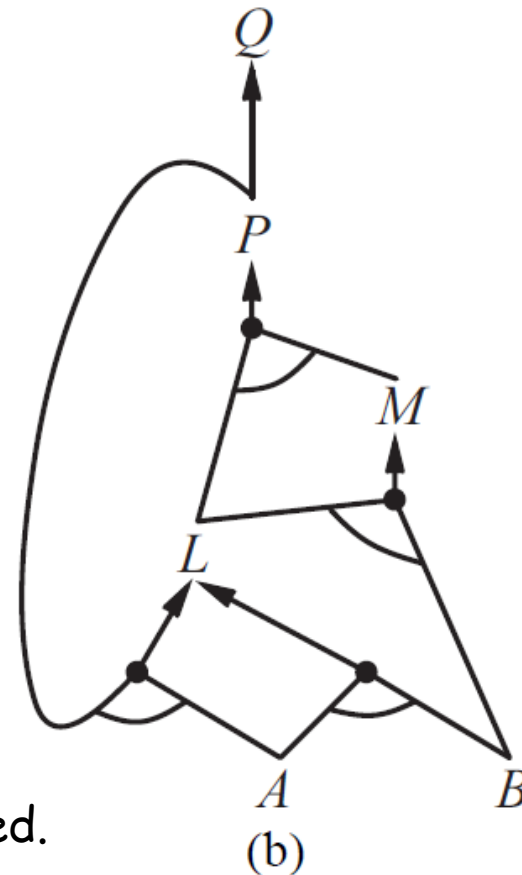
$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B

- (a)
- To prove Q , prove P
 - To prove P , prove L and M
 - To prove L , prove ...
 - ...
 - Continue until facts are reached.



NEXT

- So far we discussed propositional logic
 - Syntax
 - Semantics
 - Entailment
 - Rules of inference
 - Resolution
 - Forward chaining
 - Backward chaining
- Next
 - First-order logic