CS480 – Artificial Intelligence Fall 2015

Topic: Logical Agents
Chapter: 7
Date: 10/5

Mustafa Bilgic

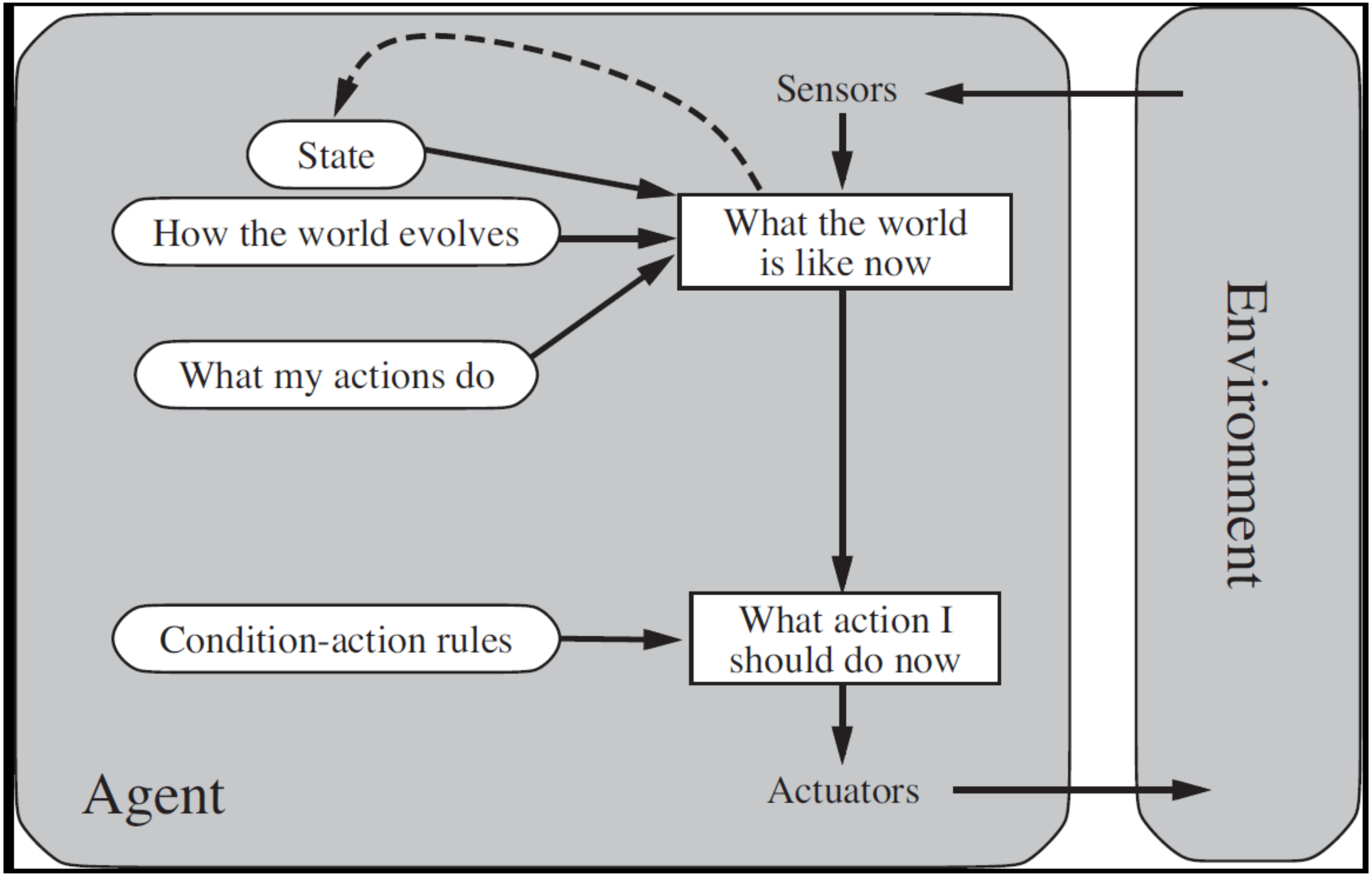
http://www.cs.iit.edu/~mbilgic

HUMANLY VS. RATIONALLY & THINKING VS. ACTING

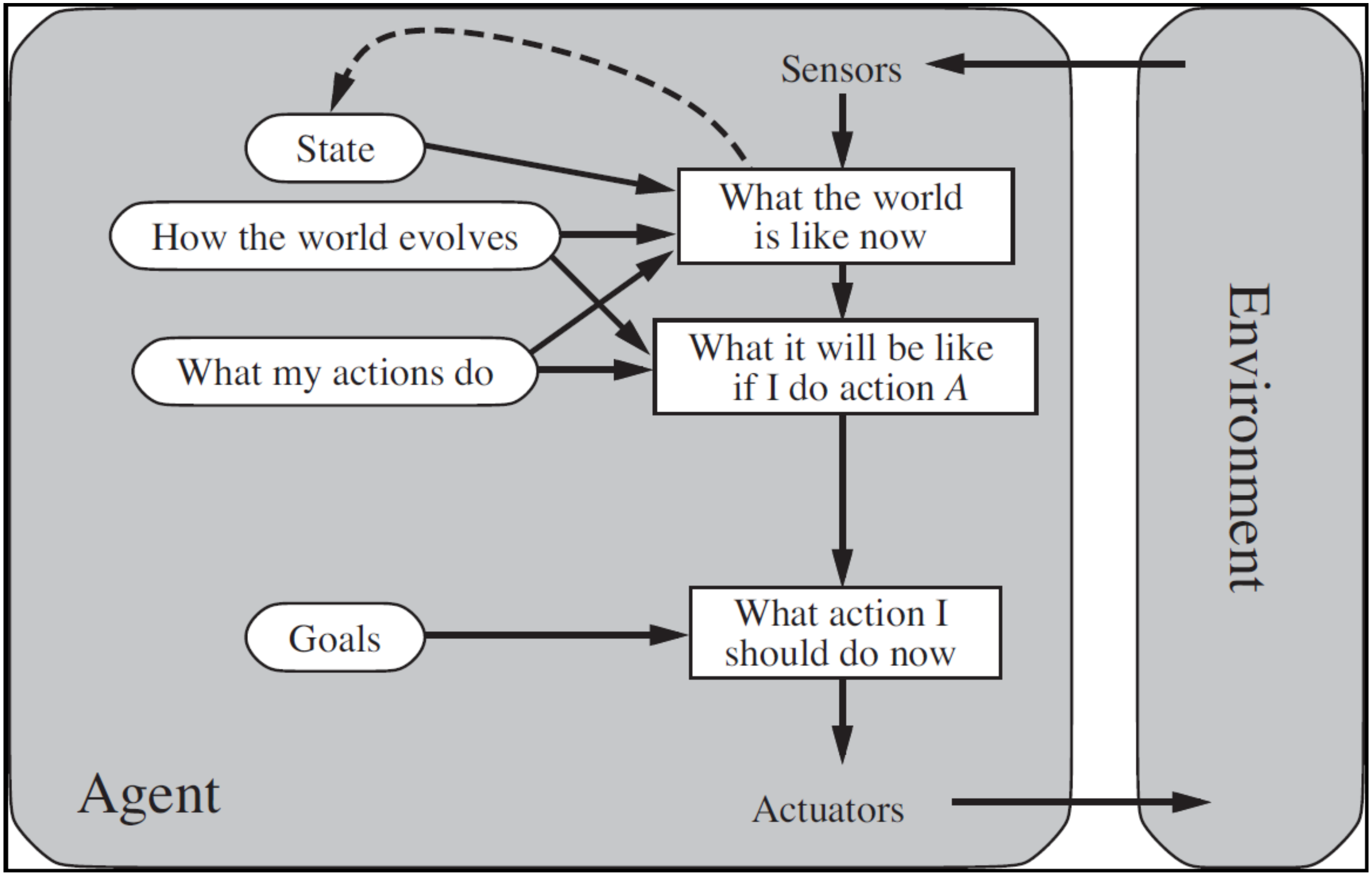
	Humanly	Rationally
Think	Thinking humanly	Thinking rationally
Act	Acting humanly	Acting rationally

- 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

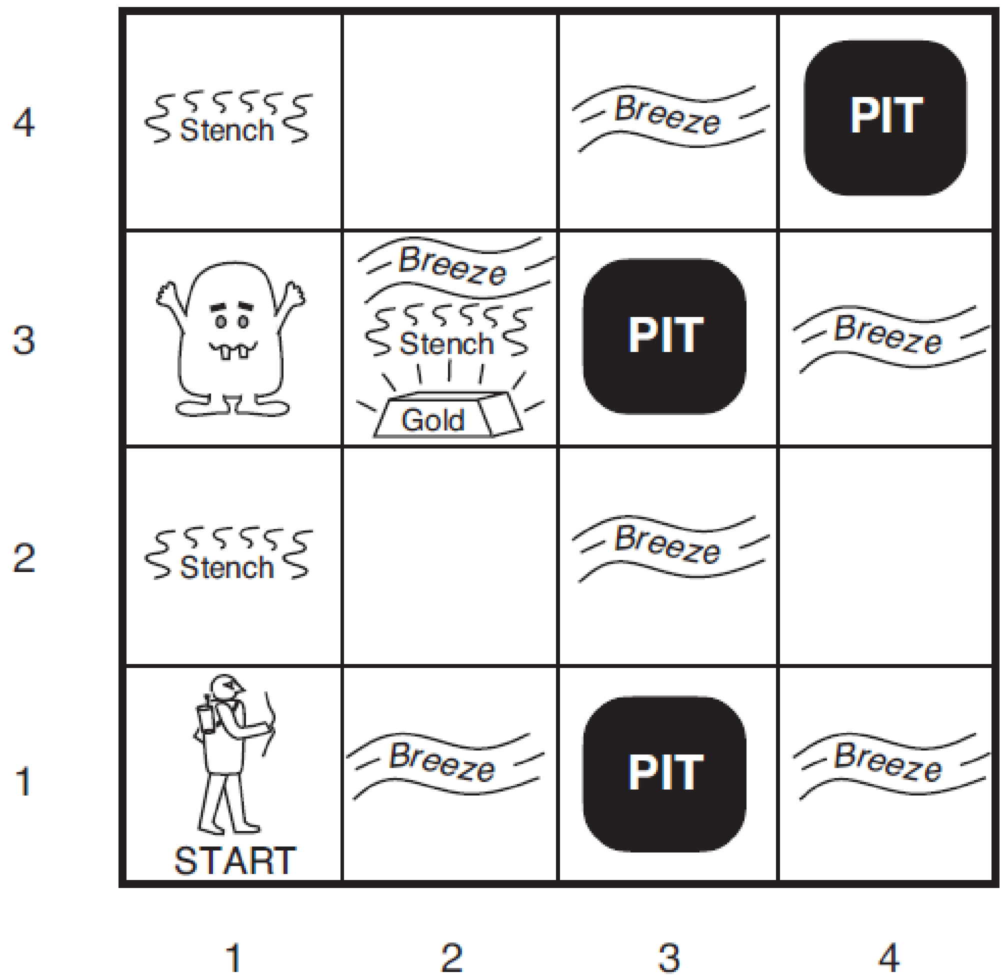
function KB-AGENT(*percept*) **returns** an *action*
  persistent: KB, a knowledge base
    t, a counter, initially 0, indicating time

  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

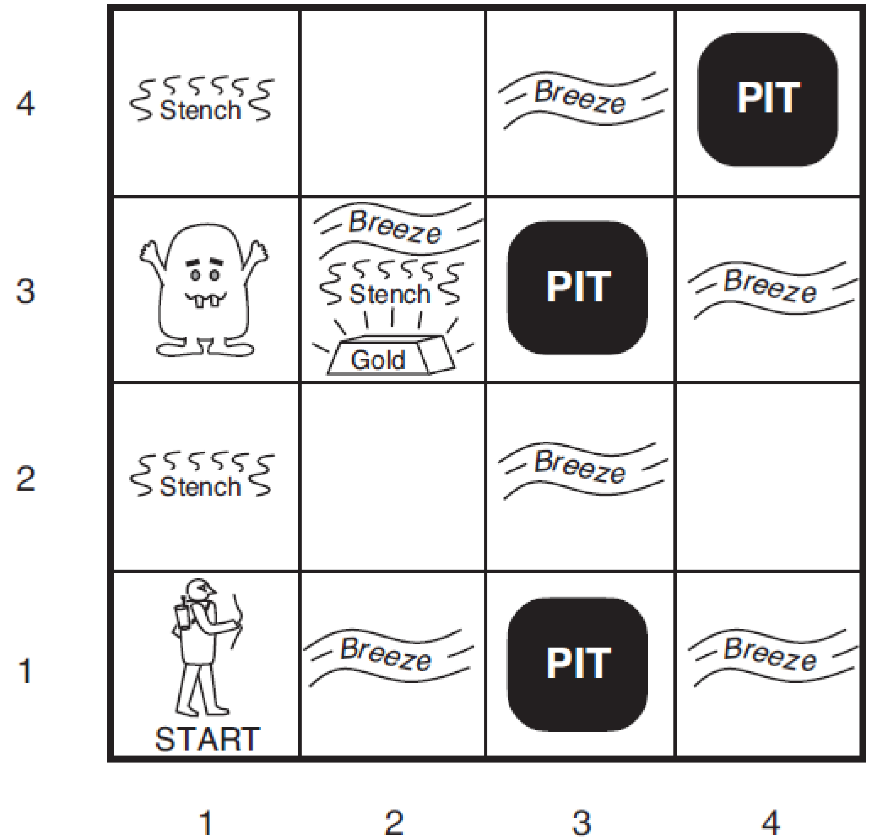
  Grabbing picks up gold if in same square

- **Actuators:** Left turn, Right turn, Forward, Grab, Shoot

- **Sensors:** Stench, Breeze, Glitter, Bump, Scream

A Typical Wumpus World

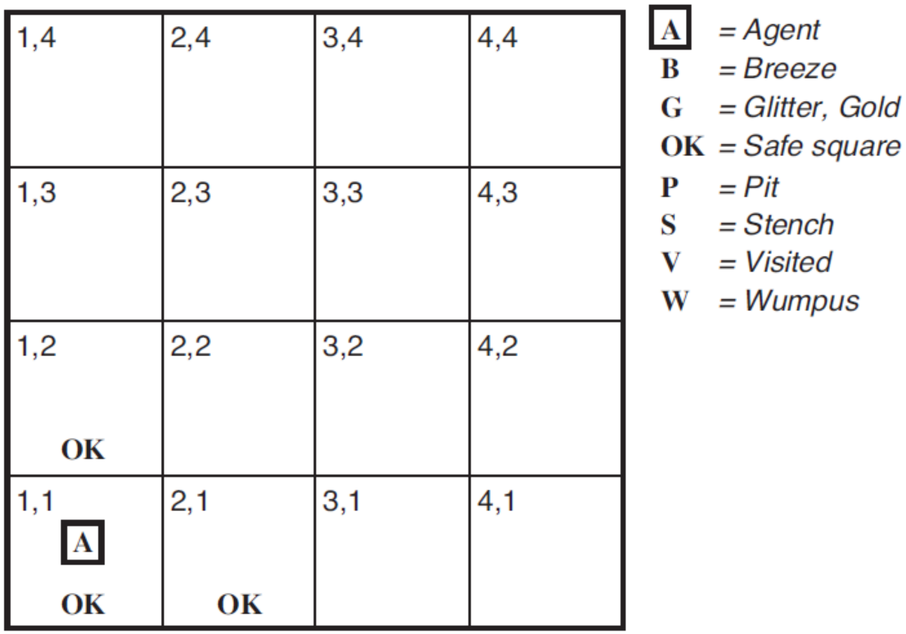
- Start at [1,1]
- The task is to find the gold, grab it, return to [1,1], and climb out



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



- 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 $\boxed{A}$ 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

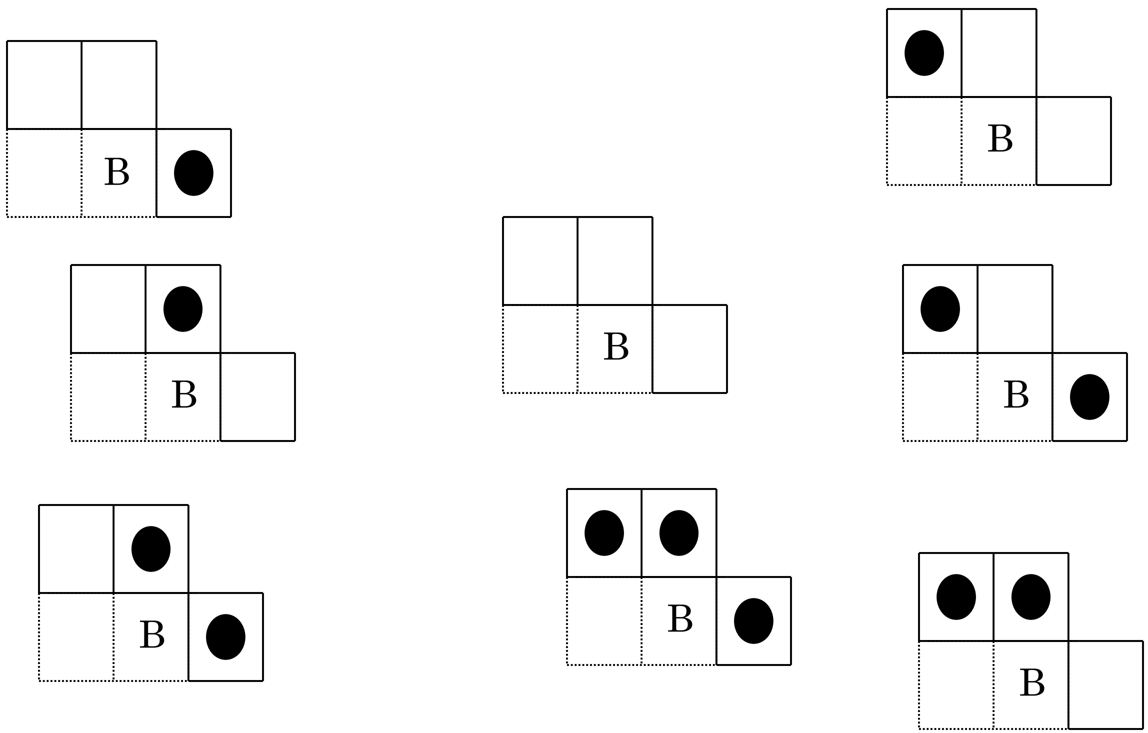
LOGIC

- 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]"?

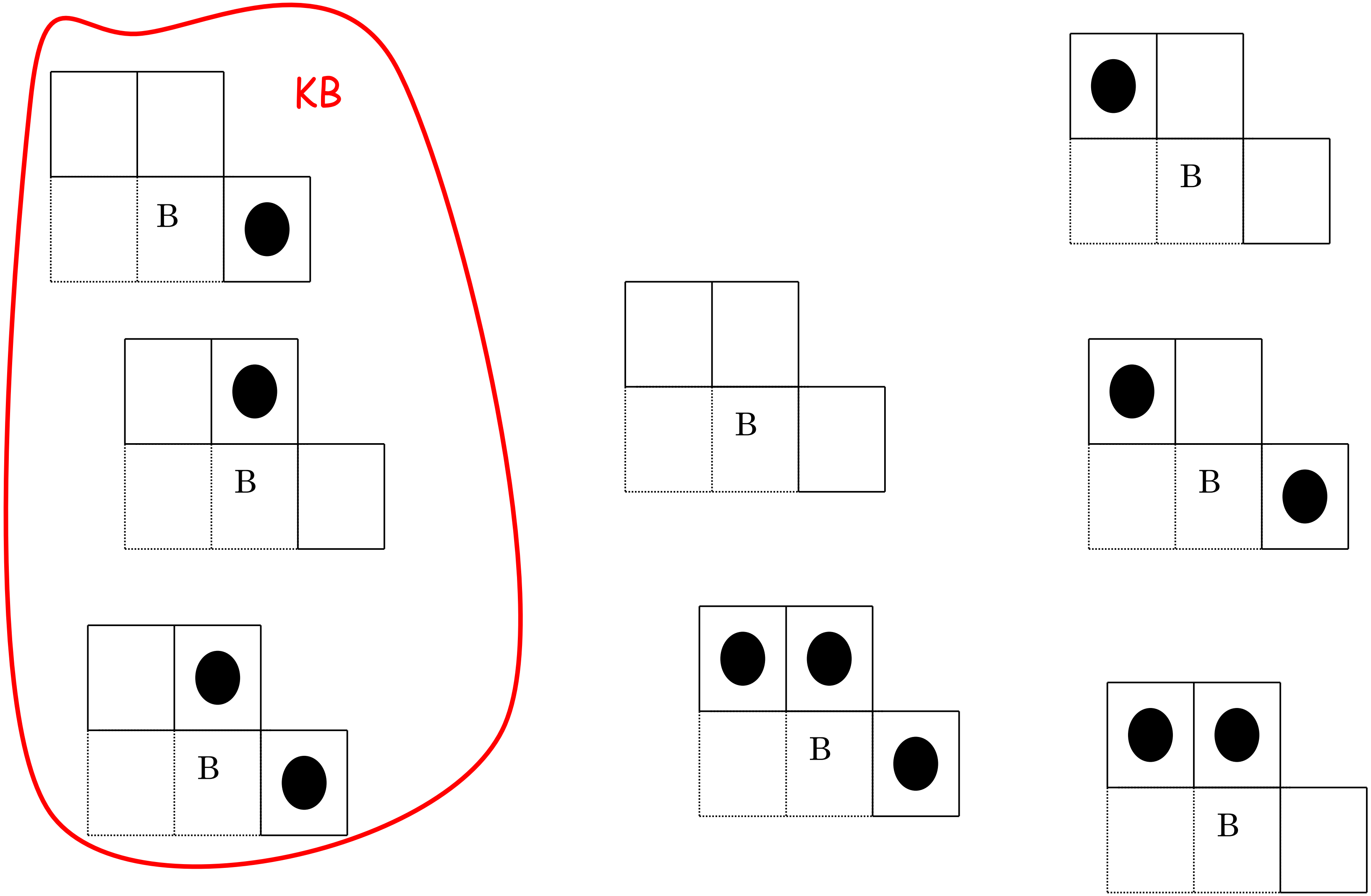
Logical Reasoning – Entailment

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

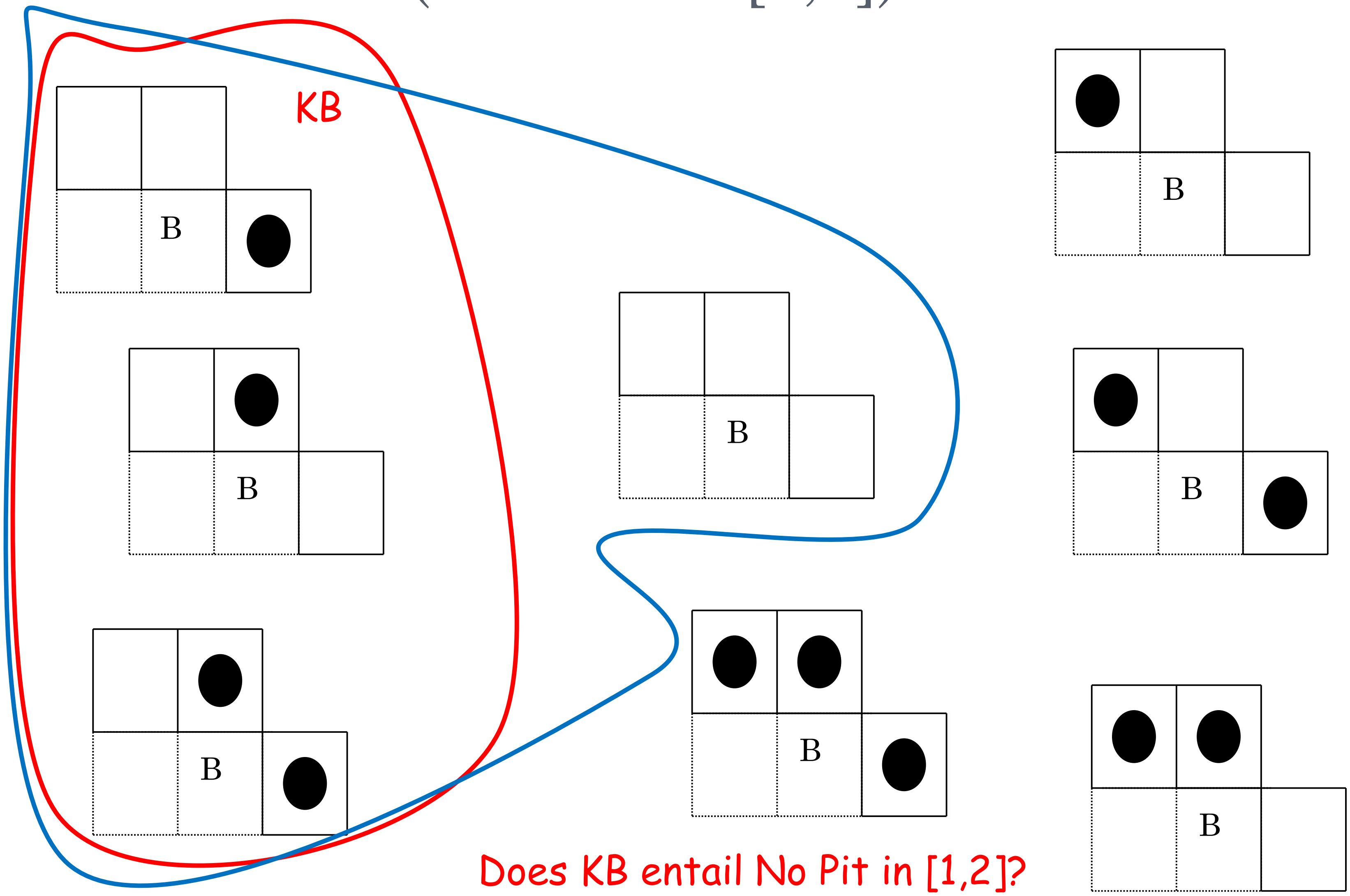
KB ⊨ A SENTENCE?

- 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

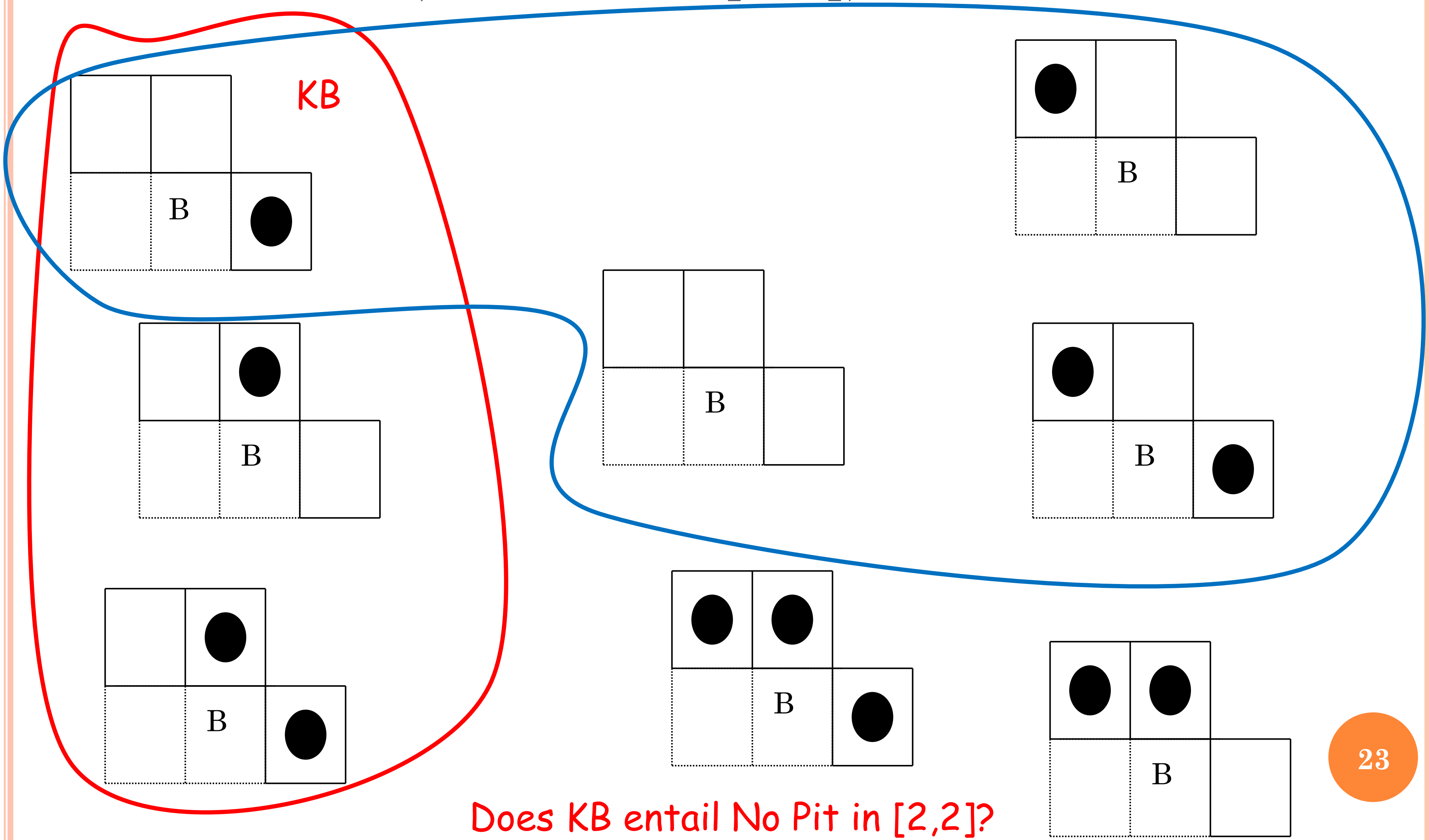
M(KB) In The Wumpus World



WHAT IS M(NO PIT IN [1,2])?



What is M(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.

    i is sound if whenever $KB \vdash_i \alpha$ *it is also true that* $KB \models \alpha$
- **Completeness** : the algorithm can derive any sentence that is entailed.

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

We'll Cover Two Types of Logic

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

Propositional Logic Language

- Propositional symbols
  - P, Q, R, …
  - 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**

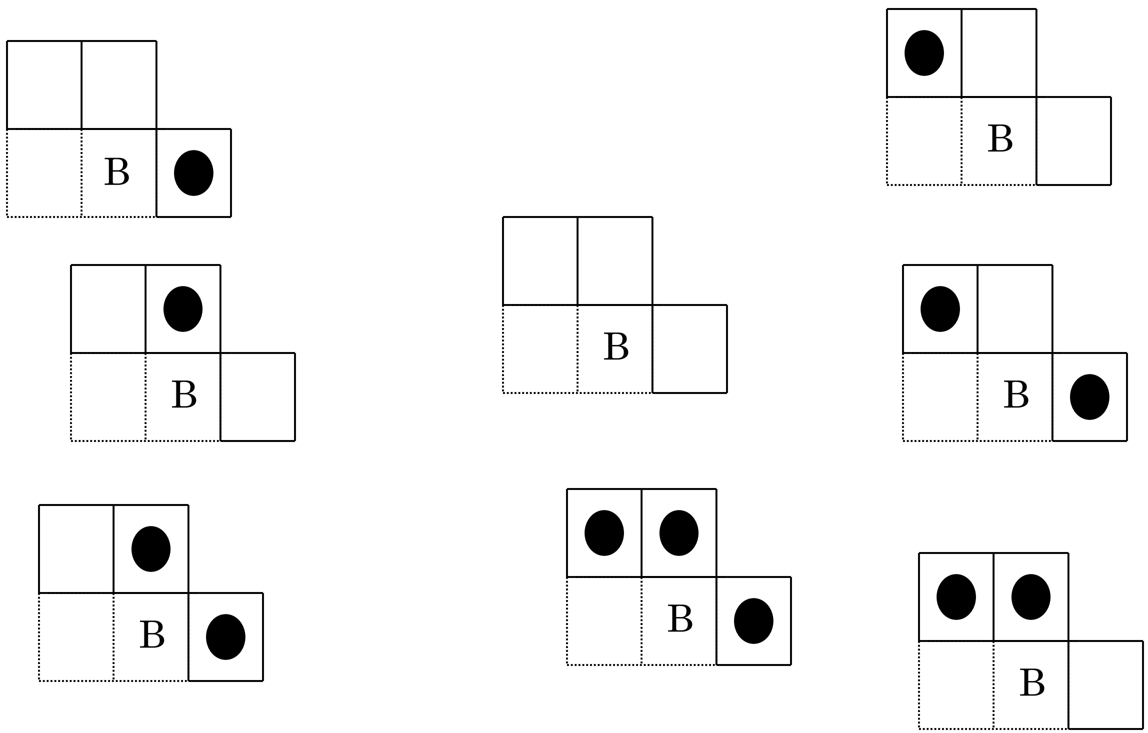
Syntax

- 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	¬P	P∧Q	P∨Q	P⇒Q	P⇔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	¬P	P∧Q	P∨Q	P⇒Q	P⇔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	¬P	P∧Q	P∨Q	P⇒Q	P⇔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 ⇒

- $P \Rightarrow Q$ 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
  - $\alpha \land \beta \equiv \beta \land \alpha$
  - $\alpha \lor \beta \equiv \beta \lor \alpha$
- Associativity
  - $((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$
  - $((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$
- 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: $B_{1,1} \Leftrightarrow P_{1,2} \lor P_{2,1}$
- R_3: $B_{2,1} \Leftrightarrow P_{1,1} \lor P_{2,2} \lor P_{3,1}$
- R_4: $\neg B_{1,1}$
- R_5: $B_{2,1}$

META-SOLUTION

- R6 = biconditional elimination from R2
- R7 = and-elimination from R6
- R8 = controposition 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, $P \vee \neg P$, $False \Rightarrow P$
- **Deduction theorem**:
  - $\alpha \models \beta$ if and only if the sentence $\alpha \Rightarrow \beta$ is valid
    - How would you prove it?
  - To check $KB \models \alpha$, we need to check if $KB \Rightarrow \alpha$ 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
  - $\alpha \models \beta$ iff the sentence $\neg(\alpha \Rightarrow \beta)$ is unstasfiable
  - $\alpha \models \beta$ iff the sentence $\alpha \wedge \neg\beta$ is unstasfiable

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 ¬β 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 ¬β
  - And then the agent arrives at a contradiction

Resolution

- Unit Resolution
  - Given
    1. $l_1 \vee \ldots \vee l_{i-1} \vee l_i \vee l_{i+1} \vee \ldots \vee l_k$
    2. $\boldsymbol{u}$, where $\boldsymbol{u}$ and $\boldsymbol{l}_i$ are complementary (one is the negation of the other)
  - Conclude
    - $l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k$
  - Example
    - From R1: $P \vee \neg Q$ and R2: Q conclude R3: ?

RESOLUTION

- Full Resolution
  - Given
    1. $l_1 \vee \ldots \vee l_{i-1} \vee \boldsymbol{l_i} \vee l_{i+1} \vee \ldots \vee l_k$
    2. $u_1 \vee \ldots \vee u_{j-1} \vee \boldsymbol{u_j} \vee u_{j+1} \vee \ldots \vee u_n$, where $\boldsymbol{u_j}$ and $\boldsymbol{l_i}$ are complementary (one is the negation of the other)
  - Conclude
    - $l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee u_1 \vee \ldots \vee u_{j-1} \vee u_{j+1} \vee \ldots \vee u_n$
  - 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:
  - $P \vee Q \vee \neg R$
  - $\neg Q \vee R$

CONJUCTIVE 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

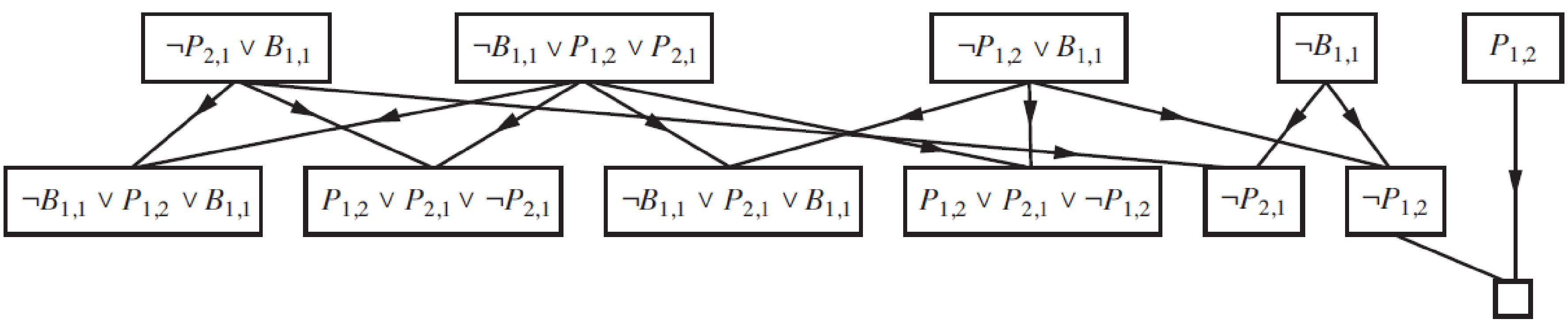
Sound and complete

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

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

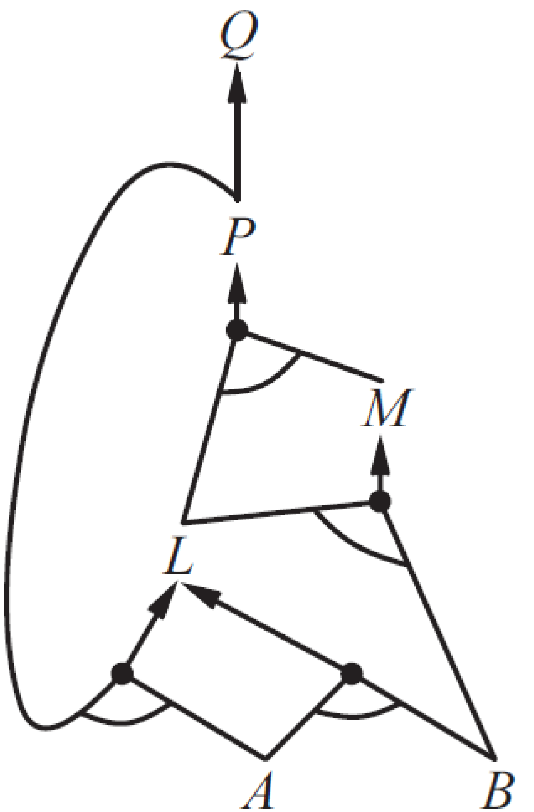
function PL-FC-ENTAILS?(KB, q) **returns** *true* or *false*
  inputs: KB, the knowledge base, a set of propositional definite clauses
      q, the query, a proposition symbol
  $count \leftarrow$ a table, where $count[c]$ is the number of symbols in c's premise
  $inferred \leftarrow$ a table, where $inferred[s]$ 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**
    $p \leftarrow$ POP($agenda$)
    if $p = q$ **then return** *true*
    if $inferred[p] = false$ **then**
      $inferred[p] \leftarrow true$
      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)

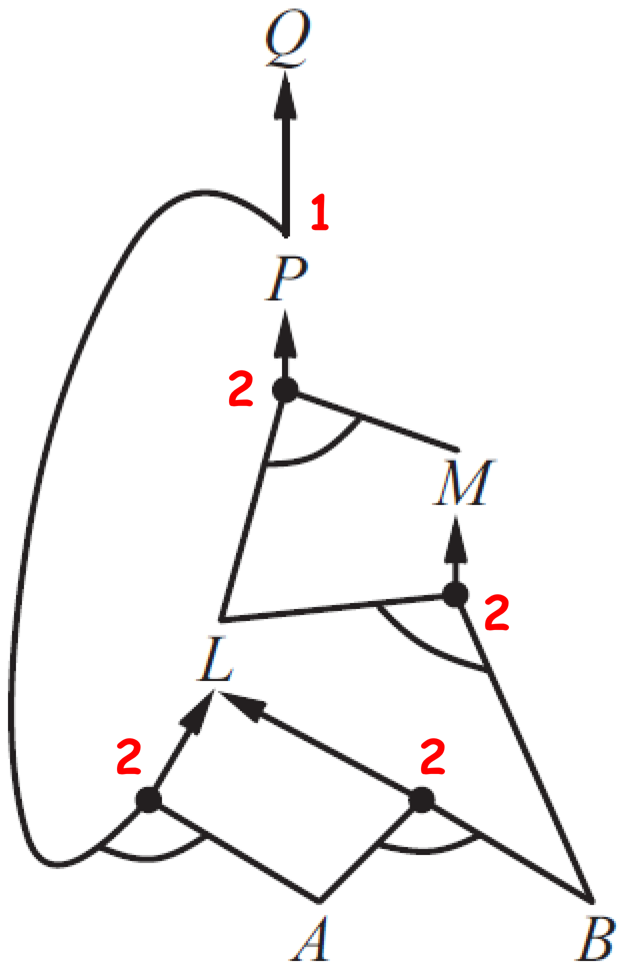


(b)

Forward Chaining

Queue

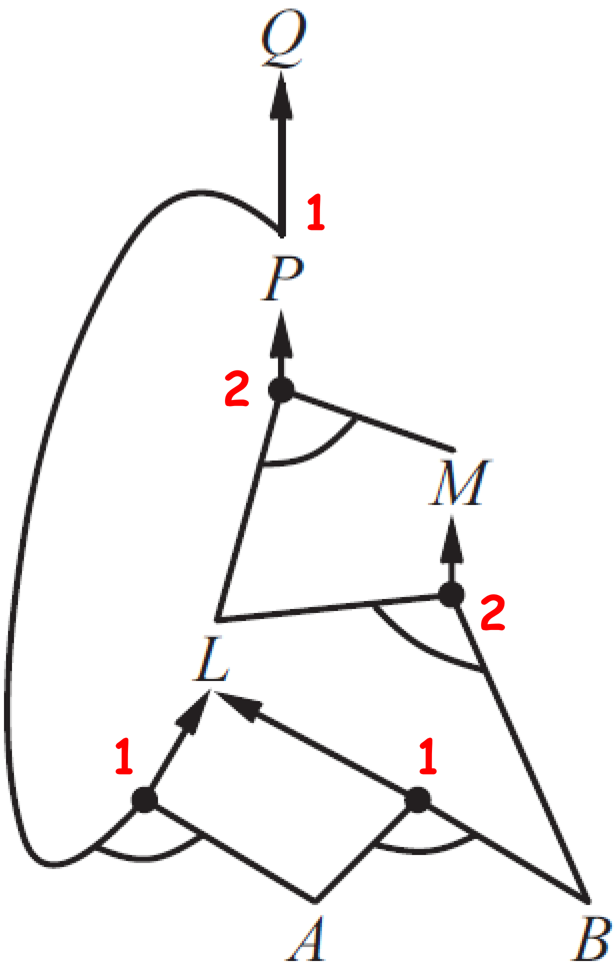
A
B



Forward Chaining – Pop A

Queue

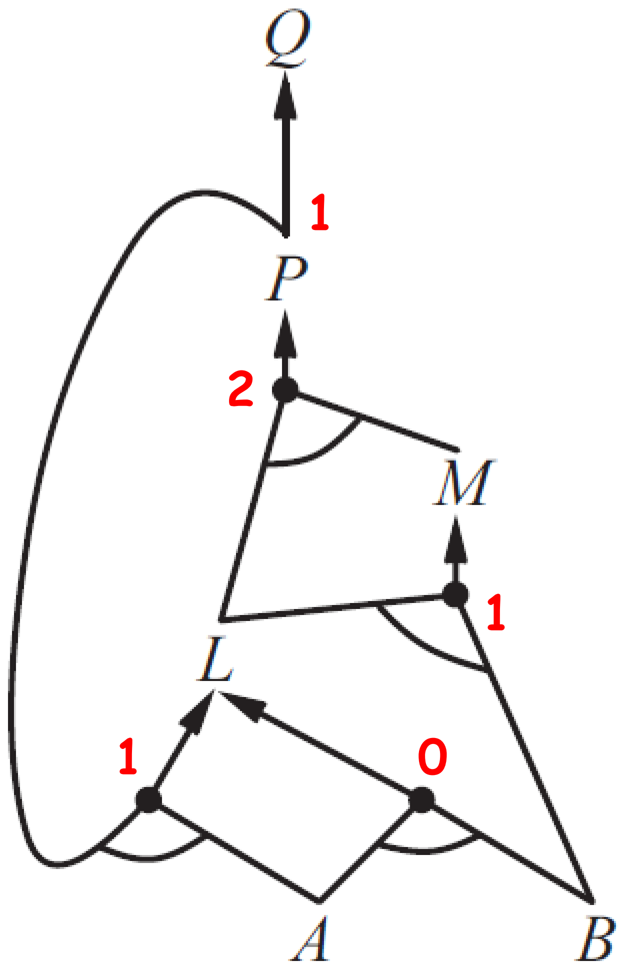
B



Forward Chaining – Pop B

Queue

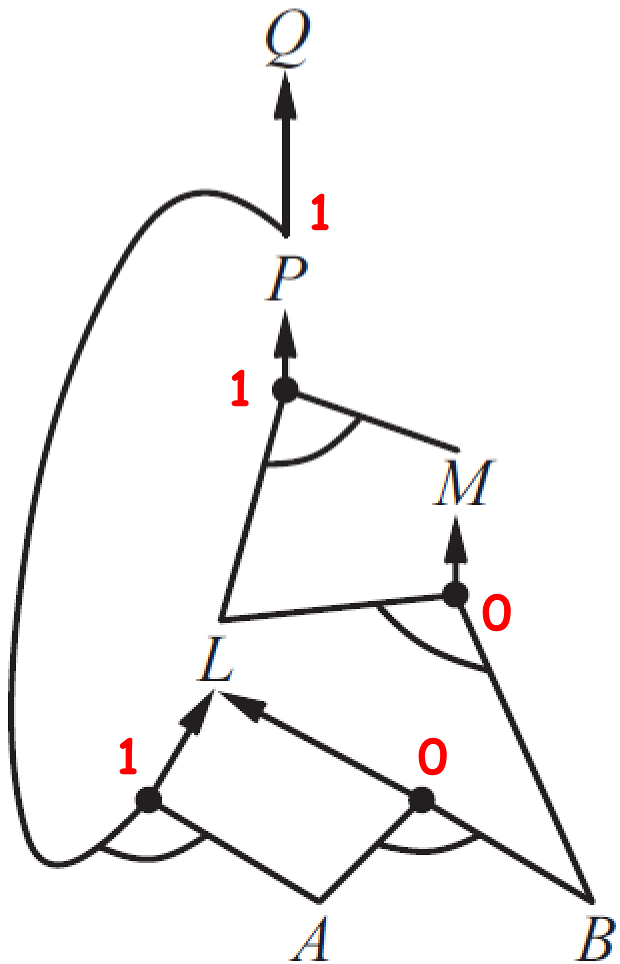
L



Forward Chaining – Pop L

Queue

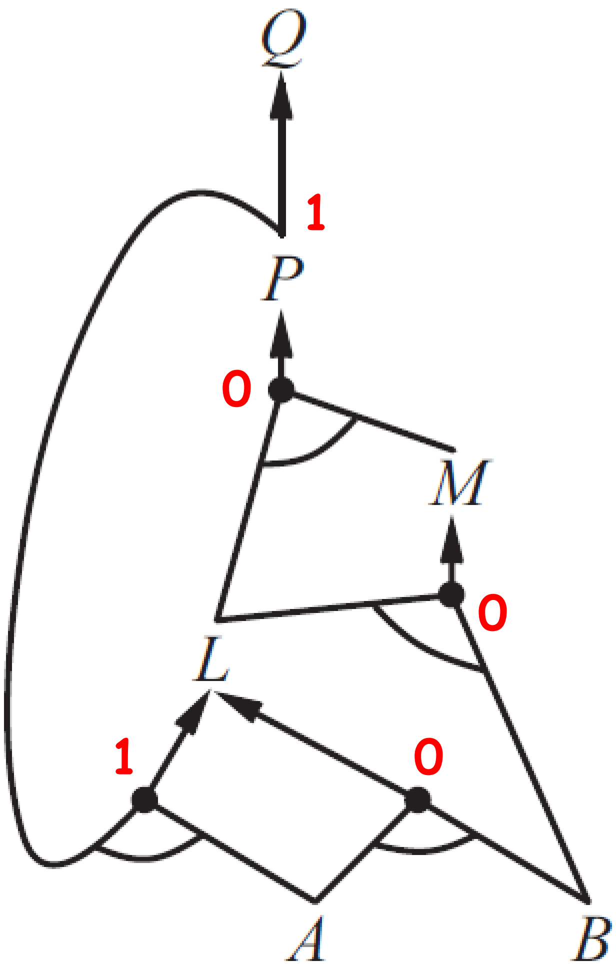
M



Forward Chaining – Pop M

Queue

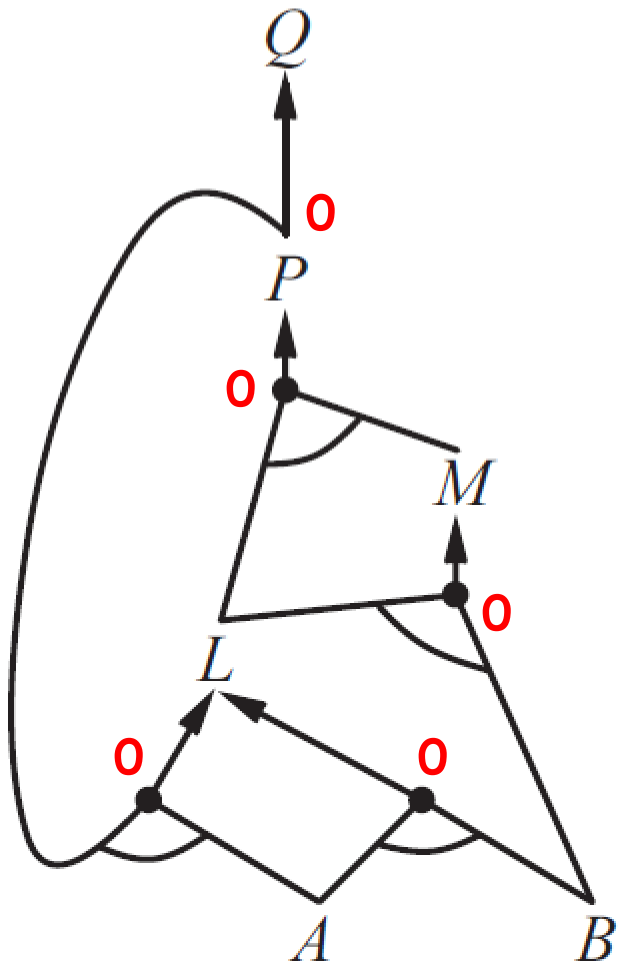
P



Forward Chaining – Pop P

Queue

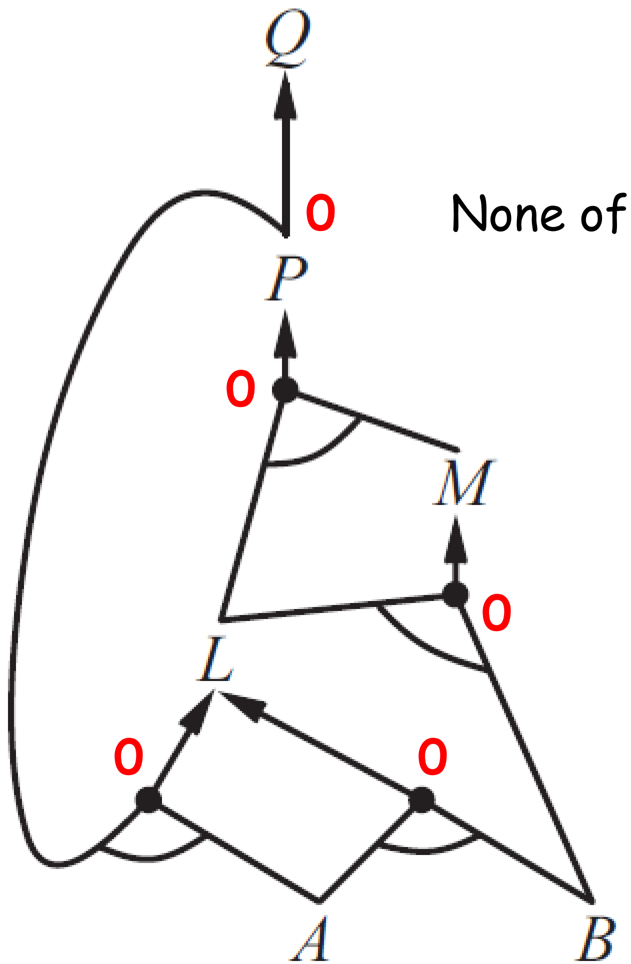
Q
L



Forward Chaining – Pop Q

Queue

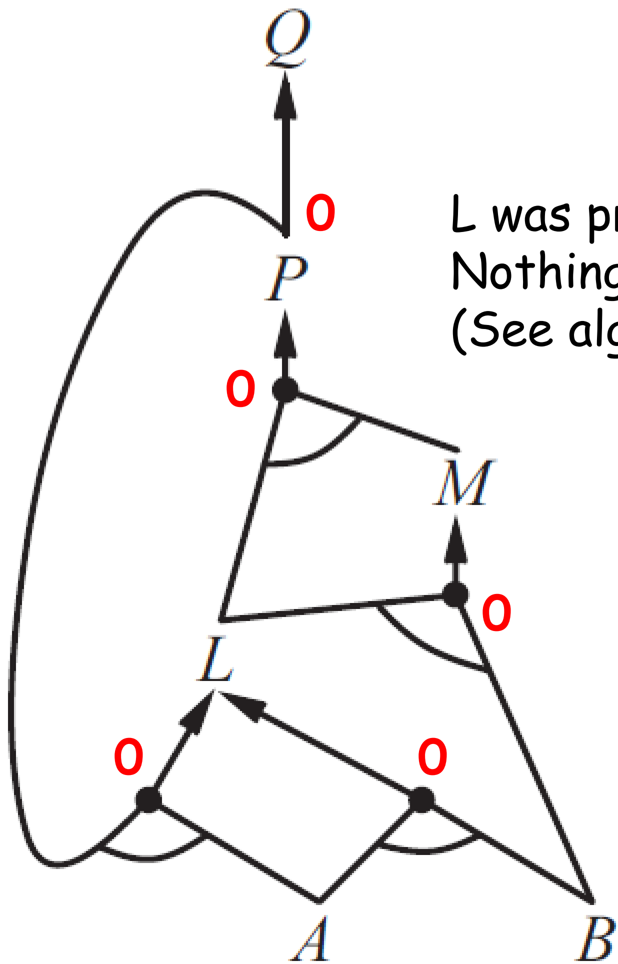
L



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

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

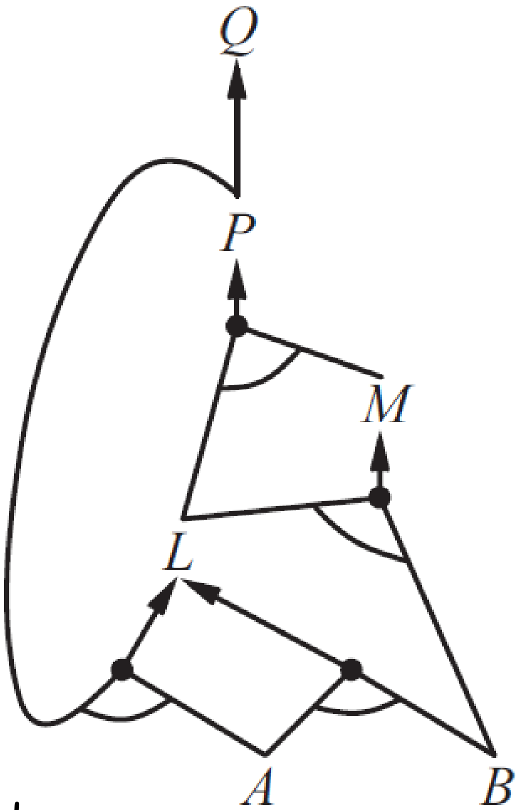
$A \wedge B \Rightarrow L$

A

B

(a)

- To prove Q, prove P
- To prove P, prove L and M
- To prove L, prove …
- …
- Continue until facts are reached.



(b)

NEXT

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