MODULE 2

# LOGIC AGENTS

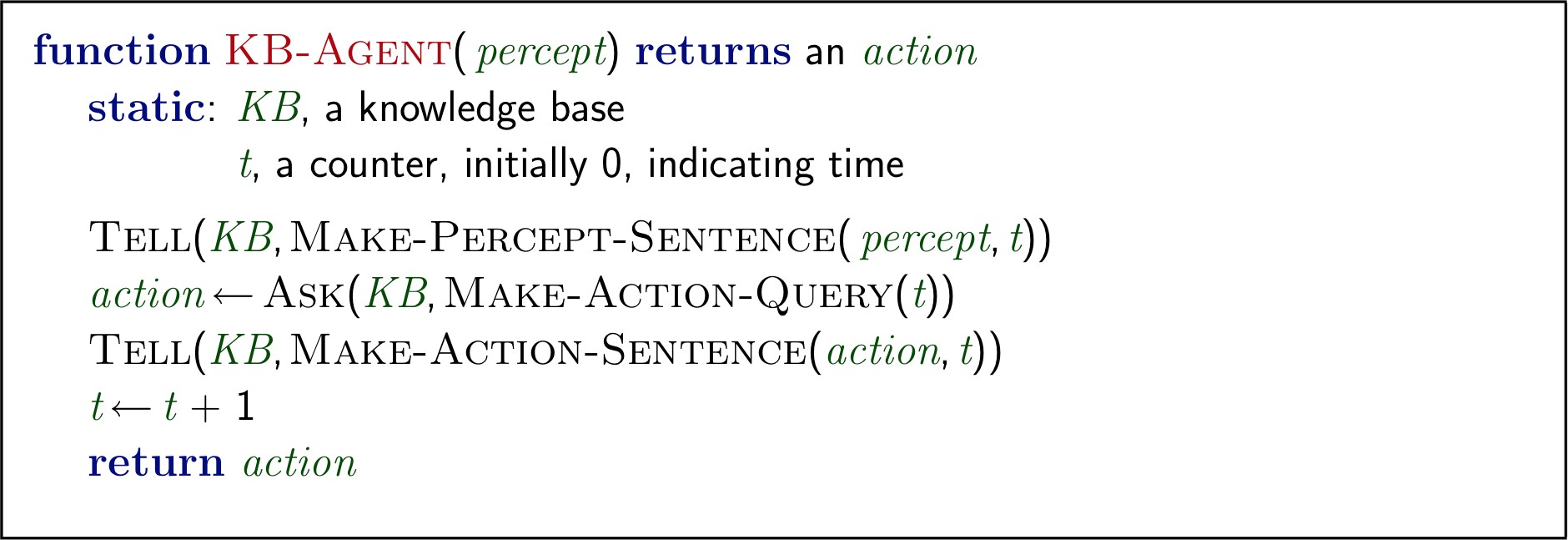
**Knowledge-based agents**

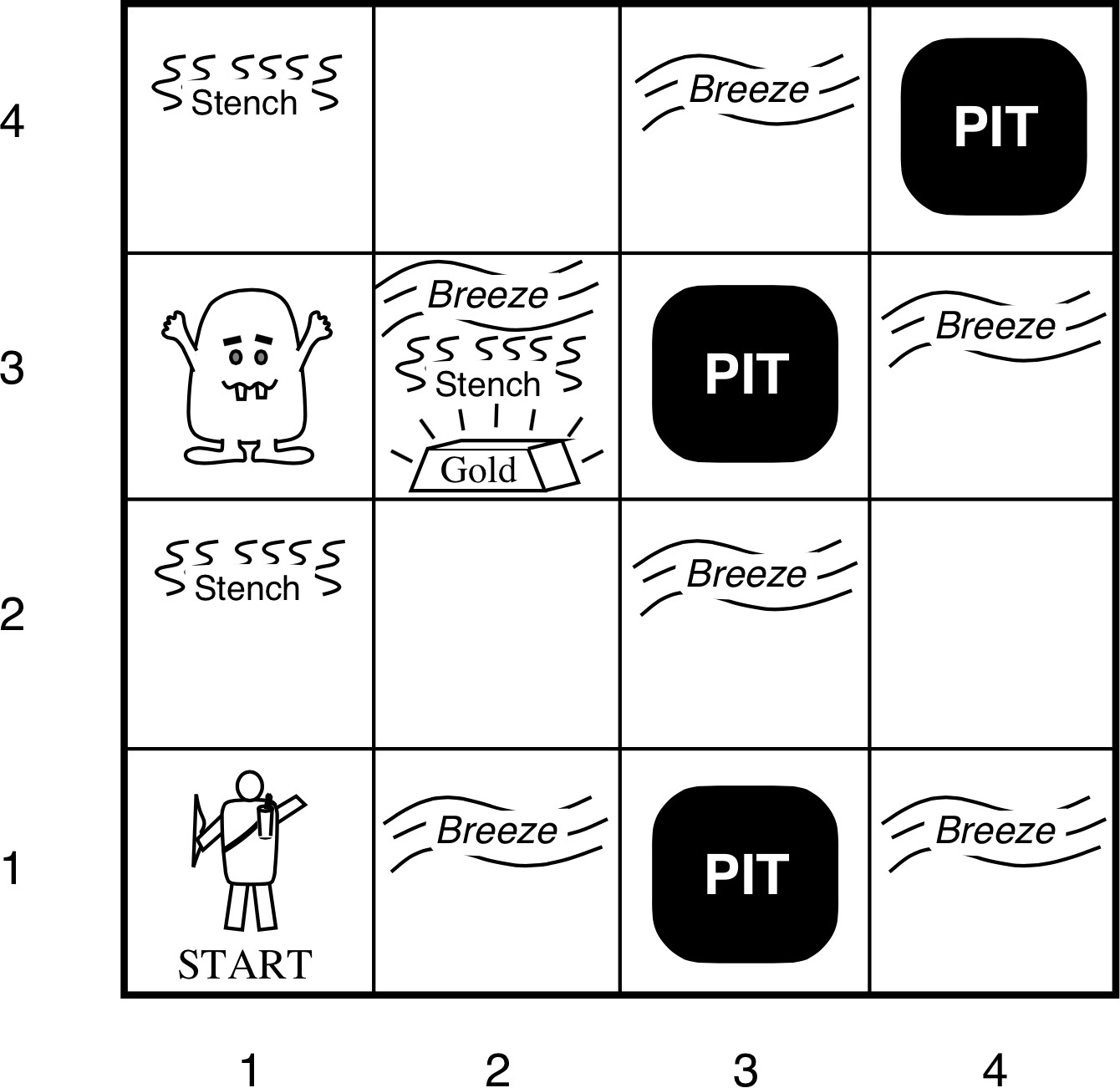
## Logical AI:

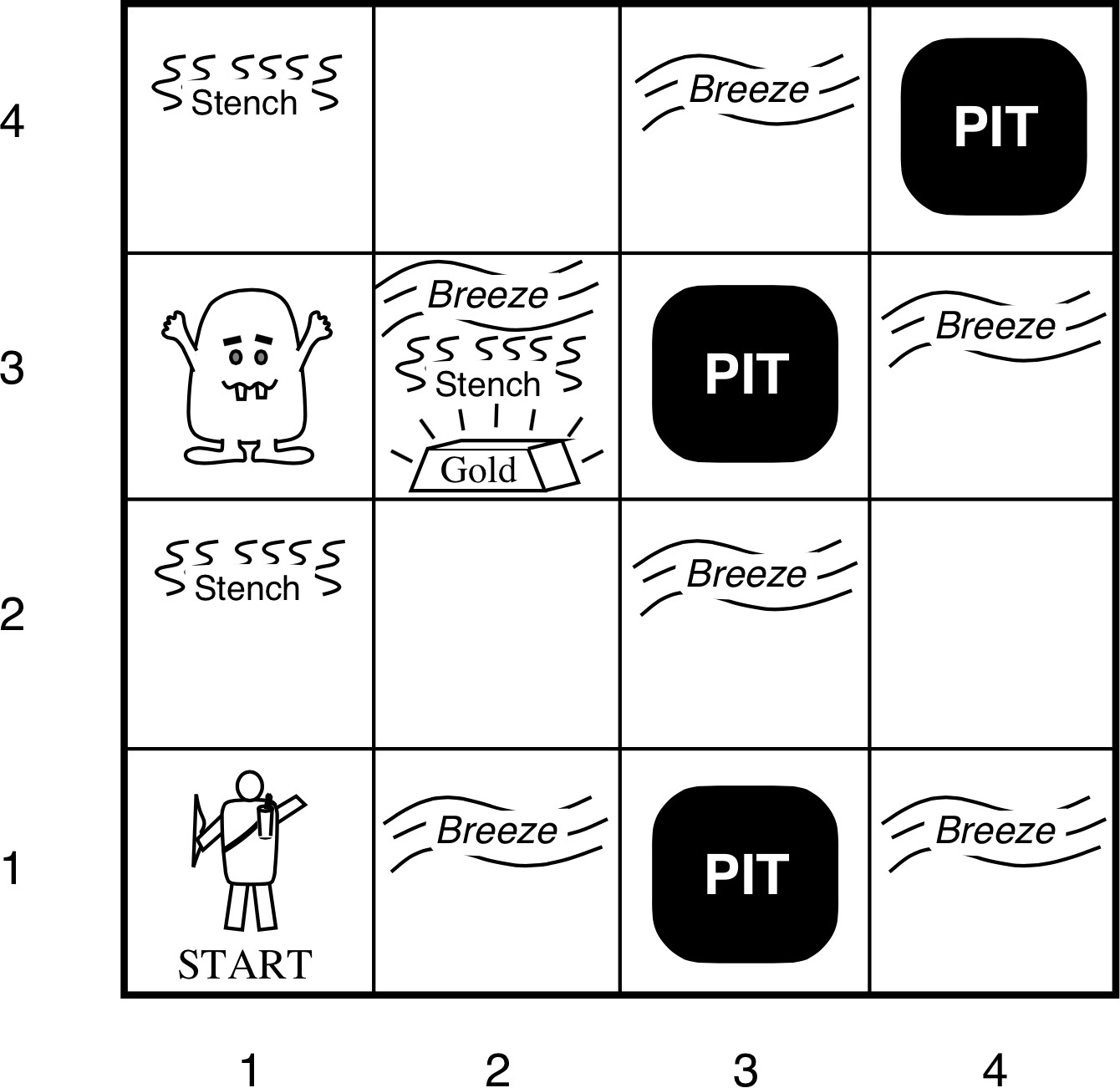
The idea is that an agent can represent knowledge of its world, its goals and the current situation by sentences in logic and decide what to do by inferring that a certain action or course of action is appropriate to achieve its goals.”

**Knowledge-based agents**

* Intelligent agents need **knowledge** about the world to choose good actions/decisions.
* Knowledge = *{***sentences***}* in a knowledge representation lan- guage (formal language).
* A sentence is an assertion about the world.
* A knowledge-based agent is composed of:
  1. Knowledge base: **domain-specific** content.
  2. Inference mechanism: **domain-independent** algorithms.
* The agent **must be able to**:
* Represent states, actions, etc.
* Incorporate new percepts
* Update internal representations of the world
* Deduce hidden properties of the world
* Deduce appropriate actions
* The agent **must be able to**:
* Represent states, actions, etc.
* Incorporate new percepts
* Update internal representations of the world
* Deduce hidden properties of the world
* Deduce appropriate actions
* **Declarative** approach to building an agent:
* Add new sentences: **Tell** it what it needs to know
* Query what is known: **Ask** itself what to do - answers should follow from the KB





* 4 X 4 grid of rooms
* Squares adjacent to Wumpus are

smelly and squares adjacent to pit are breezy

* Glitter iff gold is in the same square
* Shooting kills Wumpus if you are

facing it

* Wumpus emits a horrible scream when it is killed that can be heard anywhere
* Shooting uses up the only arrow
* Grabbing picks up gold if in same

square

* Releasing drops the gold in same square

**Wumpus World PEAS:**

* **Performance measure**: gold +1000, death (eaten or falling in a pit) -1000, -1 per action taken, -10 for using the arrow.

The games ends either when the agent dies or comes out of the cave.

## Environment

* 4 X 4 grid of rooms
* Agent starts in square [1,1] facing to the right
* Locations of the gold, and Wumpus are chosen randomly with a uniform distribution from all squares except [1,1]
* Each square other than the start can be a pit with proba- bility of 0.2

## Actuators:

**–** Left turn, Right turn, Forward, Grab, Release, Shoot

## Sensors:

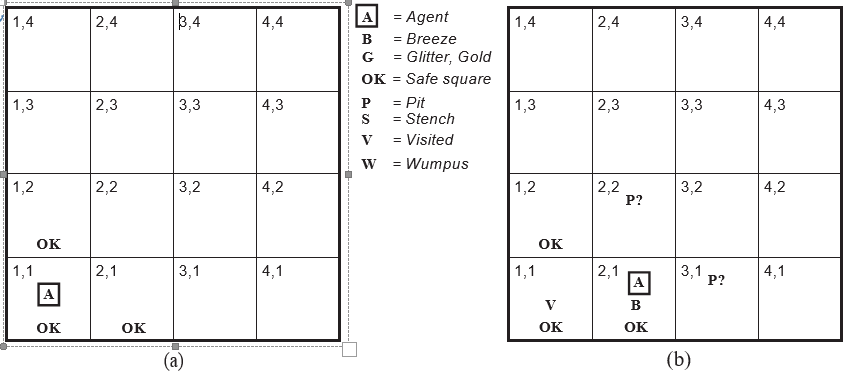
* Stench, Breeze, Glitter, Bump, Scream
* Represented as a 5-element list
* Example: [Stench, Breeze, None, None, None]

**Wumpus World properties:**

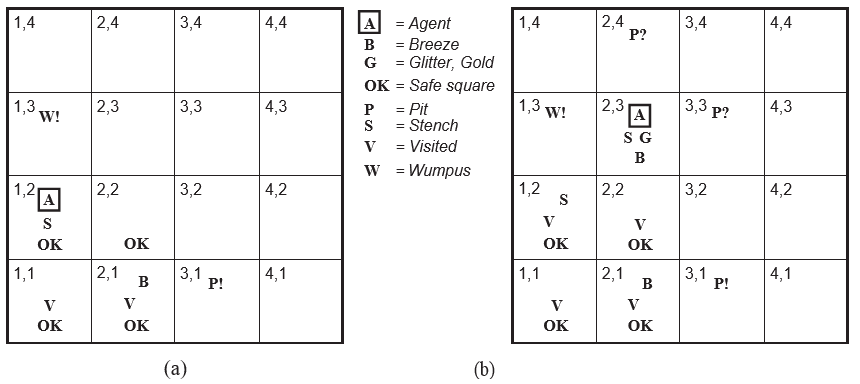
* Partially observable
* Static
* Discrete
* Single-agent
* Deterministic
* Sequential

**Exploring Wumpus World:**

Agent’s first steps:



Agent’s later steps:



**Logic:**

* **Knowledge base**: a set of sentences in a formal representation, logic
* **Logics**: are formal languages for representing knowledge to extract conclusions
* **Syntax**: defines well-formed sentences in the language
* **Semantic**: defines the truth or meaning of sentences in a world
* **Inference**: a procedure to derive a new sentence from other ones.
* **Logical entailment**: is a relationship between sentences. It means that a sentence follows logically from other sentences

*KB* |= *α*

**Propositional logic:**

* Propositional logic (PL) is the simplest logic.
* **Syntax of PL**: defines the allowable sentences or propositions.
* **Definition (Proposition)**: A proposition is a declarative statement that’s either True or False.
* **Atomic proposition**: single proposition symbol. Each symbol is a proposition. Notation: upper case letters and may contain subscripts.
* **Compound proposition**: constructed from atomic propositions using parentheses and logical connectives.

**Atomic proposition:**

## Examples of atomic propositions:

* 2+2=4 is a true proposition
* *W*1*,*3 is a proposition. It is true if there is a Wumpus in [1,3]
* “If there is a stench in [1,2] then there is a Wumpus in [1,3]” is a proposition
* “How are you?” or “Hello!” are not propositions. In general, statement that are questions, commands, or opinions are not propositions.

**Compound proposition:**

## Examples of compound/complex propositions:

Let *p*, *p*1, and *p*2 be propositions

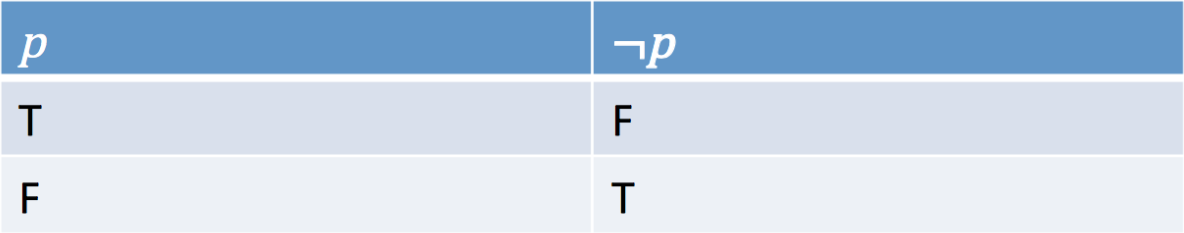
* **Negation** ¬*p* is also a proposition. We call a **literal** either an atomic proposition or its negation. E.g., *W*1*,*3 is a positive literal, and ¬*W*1*,*3 is a negative literal.
* **Conjunction** *p*1 ∧ *p*2. E.g., *W*1*,*3 ∧ *P*3*,*1
* **Disjunction** *p*1 ∨ *p*2 E.g., *W*1*,*3 ∨ *P*3*,*1
* **Implication** *p*1 → *p*2. E.g., *W*1*,*3 ∧ *P*3*,*1 → ¬*W*2*,*2
* **If and only if** *p*1 ↔ *p*2. E.g., *W*1*,*3 ↔ ¬*W*2*,*2

**Truth tables:**

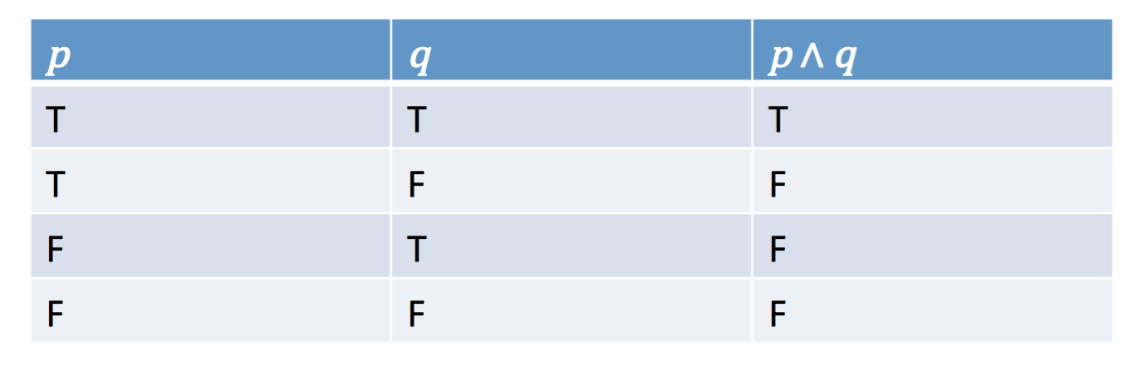
* The semantics define the rules to determine the truth of a sentence.
* Semantics can be specified by truth tables.
* Boolean values domain: T,F
* n-tuple: (*x*1*, x*2*, ..., xn*)
* Operator on n-tuples : *g*(*x*1 = *v*1*, x*2 = *v*2*, ..., xn* = *vn*)
* Definition: A truth table defines an operator *g* on n- tuples by specifying a boolean value for each tuple
* Number of rows in a truth table? *R* = 2*n*

**Building propositions:**

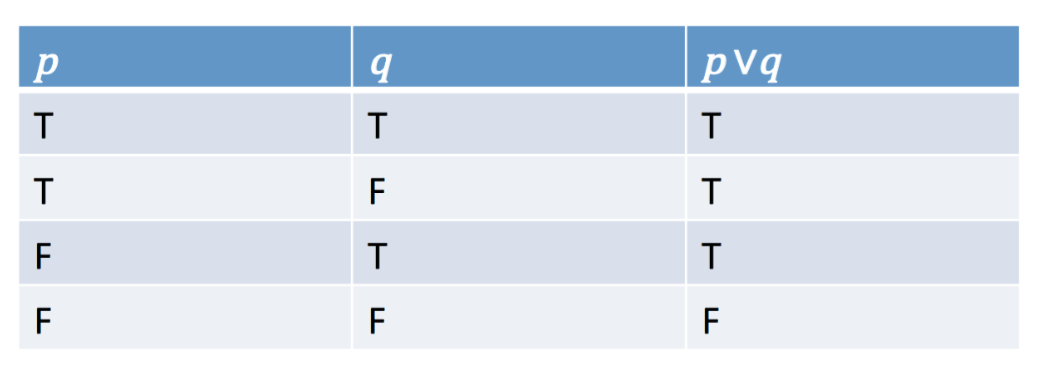
## Negation:



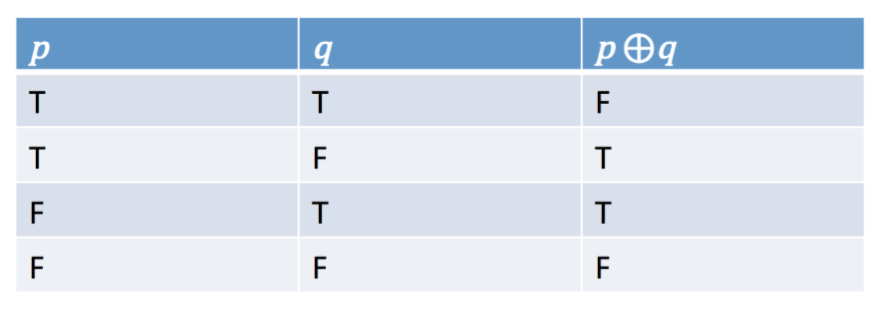
**Conjunction:**



## Disjunction:



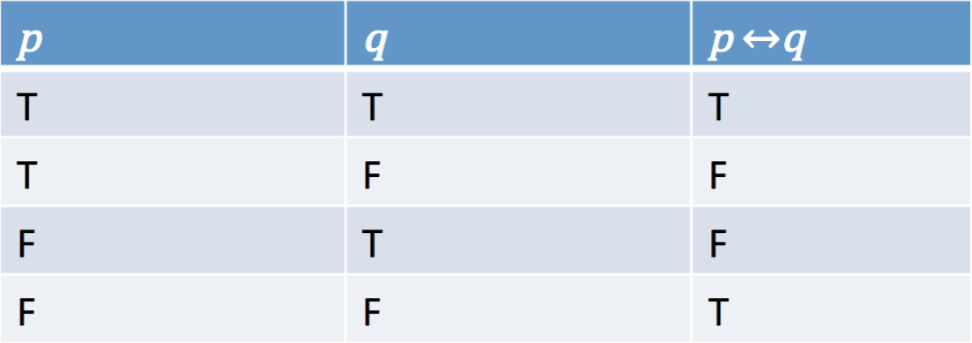
**Exclusive or:**



## Implication:

## 

**Biconditional or If and only if (IFF):**



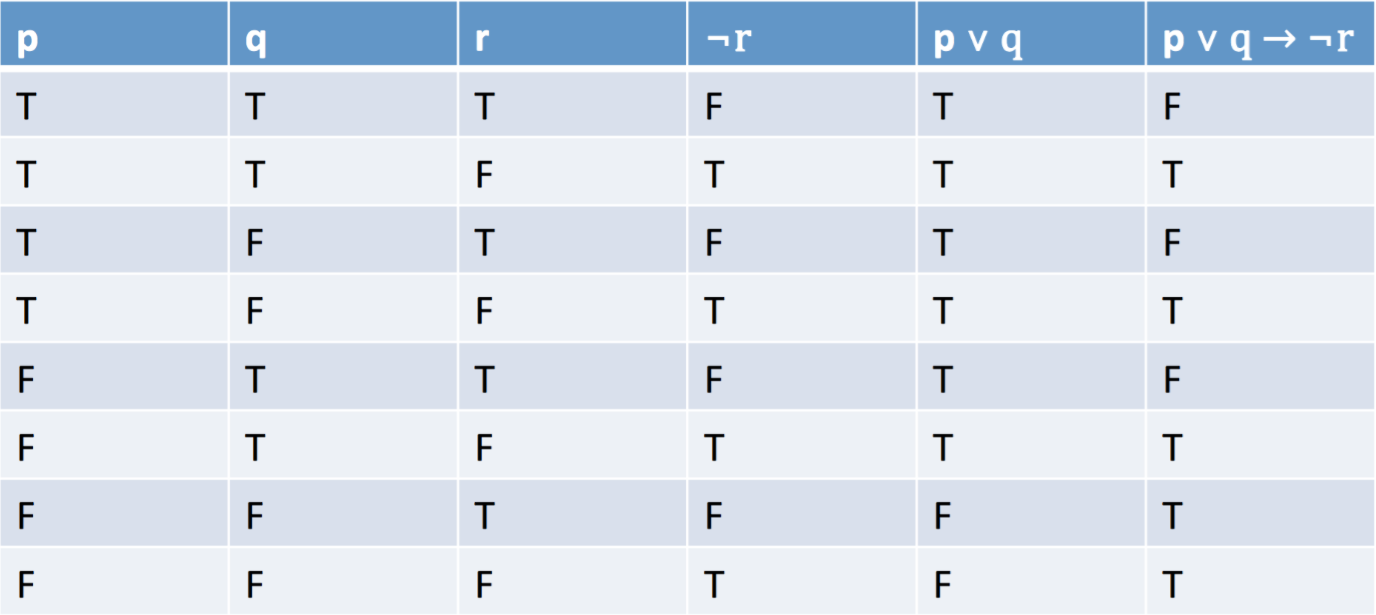
**Precedence of operators:**

* Just like arithmetic operators, there is an operator precedence when evaluating logical operators as follows:

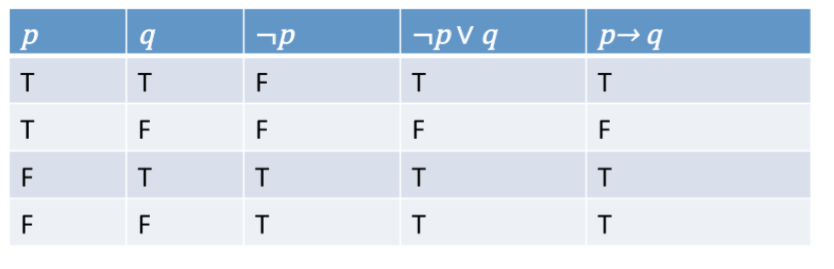
1. Expressions in parentheses are processed (inside to outside)
2. Negation
3. AND
4. OR
5. Implication
6. Biconditional
7. Left to right

* Use parentheses whenever you have any doubt!

**Building propositions:**



**Logical equivalence:**

* Two propositions *p* and *q* are logically equivalent if and only if the columns in the truth table giving their truth values agree.
* We write this as *p* ⇔ *q* or *p* ≡ *q*.

**Properties of operators:**

* Commutativity: *p* ∧ *q* = *q* ∧ *p p* ∨ *q* = *q* ∨ *p*
* Associativity: (*p* ∧ *q*) ∧ *r* = *p* ∧ (*q* ∧ *r*) (*p* ∨ *q*) ∨ *r* = *p* ∨ (*q* ∨ *r*)
* Identity element: *p* ∧ *True* = *p p* ∨ *True* = *True*
* ¬(¬*p*) = *p*
* *p* ∧ *p* = *p p* ∨ *p* = *p*
* Distributivity:

*p* ∧ (*q* ∨ *r*) = (*p* ∧ *q*) ∨ (*p* ∧ *r*)

*p* ∨ (*q* ∧ *r*) = (*p* ∨ *q*) ∧ (*p* ∨ *r*)

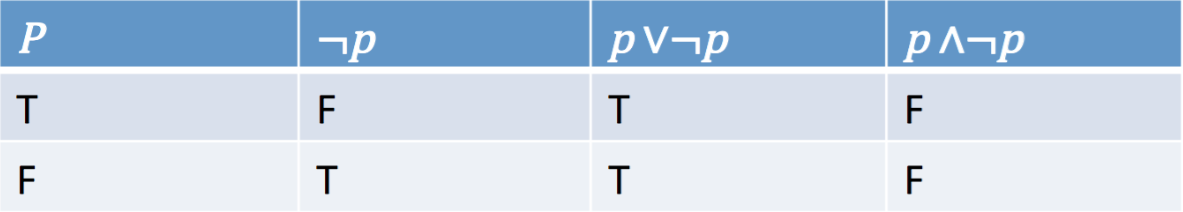
* *p* ∧ (¬*p*) = *False* and *p* ∨ (¬*p*) = *True*
* DeMorgan’s laws:

¬ (*p* ∧ *q*) = (¬*p*) ∨ (¬*q*)

¬ (*p* ∨ *q*) = (¬*p*) ∧ (¬*q*)

**Tautology and contradiction:**

* **Tautology** is a proposition which is always true
* **Contradiction** is a proposition which is always false
* **Contingency** is a proposition which is neither a tautology or a contradiction



**Contrapositive, inverse, etc.:**

* Given an **implication** *p* → *q*
* The **converse** is: *q* → *p*
* The **contrapositive** is: ¬*q* → ¬*p*
* The **inverse** is: ¬*p* → ¬*q*

**Example:** Hot is a sufficient condition for my going to the beach.

* The **implication** is:
* The **converse** is:
* The **contrapositive** is:
* The **inverse** is:

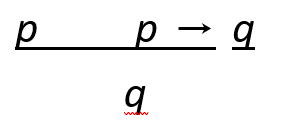
**Contrapositive, inverse, etc.**

* Given an **implication** *p* → *q*
* The **converse** is: *q* → *p*
* The **contrapositive** is: ¬*q* → ¬*p*
* The **inverse** is: ¬*p* → ¬*q*

**Example:** Hot is a sufficient condition for my going to the beach.

* The **implication** is:
* The **converse** is:
* The **contrapositive** is:
* The **inverse** is:

**Inference (Modus Ponens)**

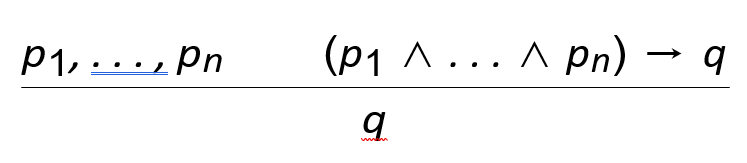


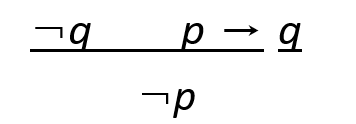


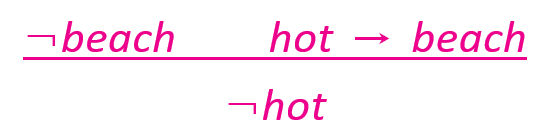
**Horn clauses**: a proposition of the form:

*p*1 ∧ *. . .* ∧ *pn* → *q*

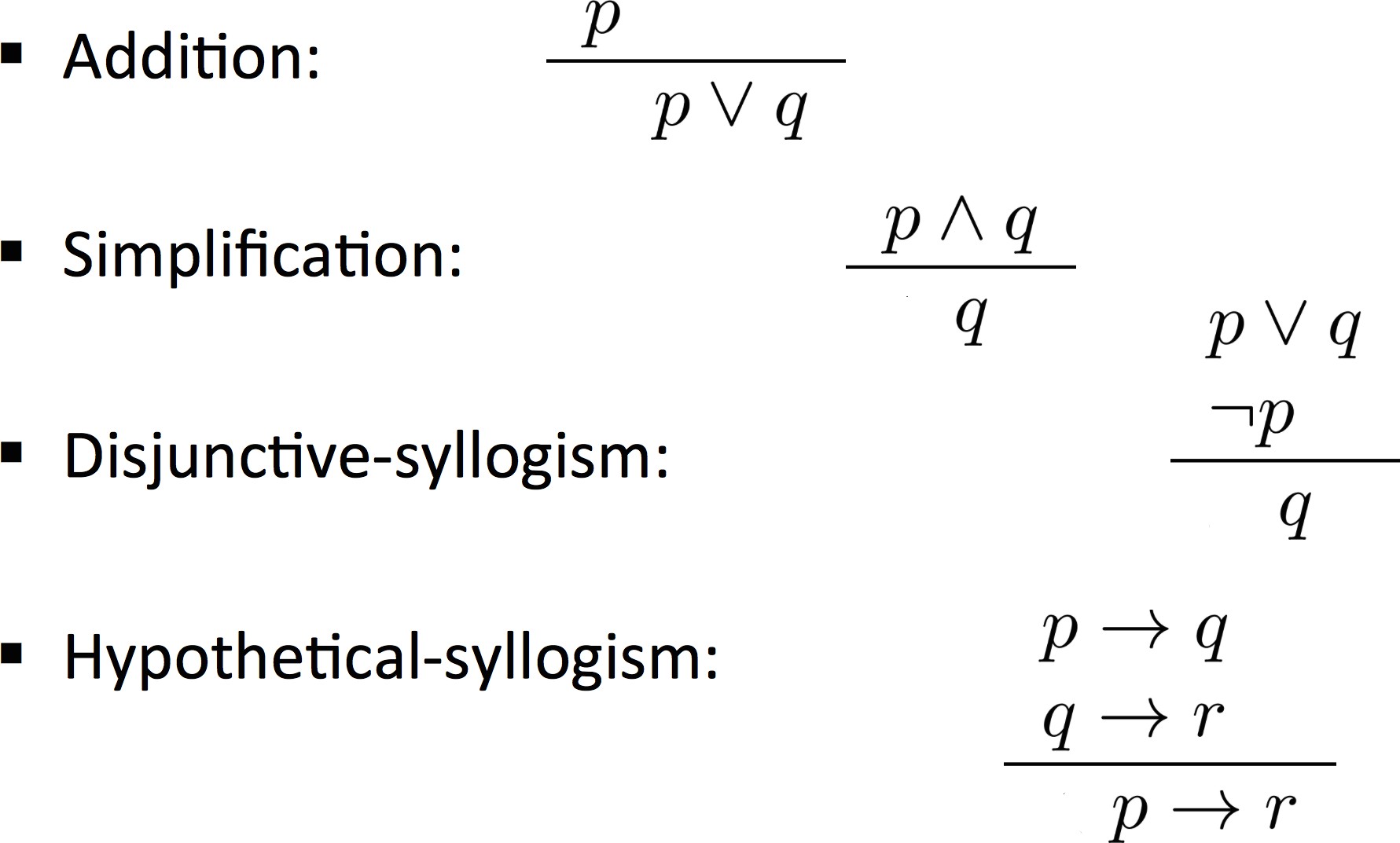
Modus Ponens deals with **Horn clauses**:





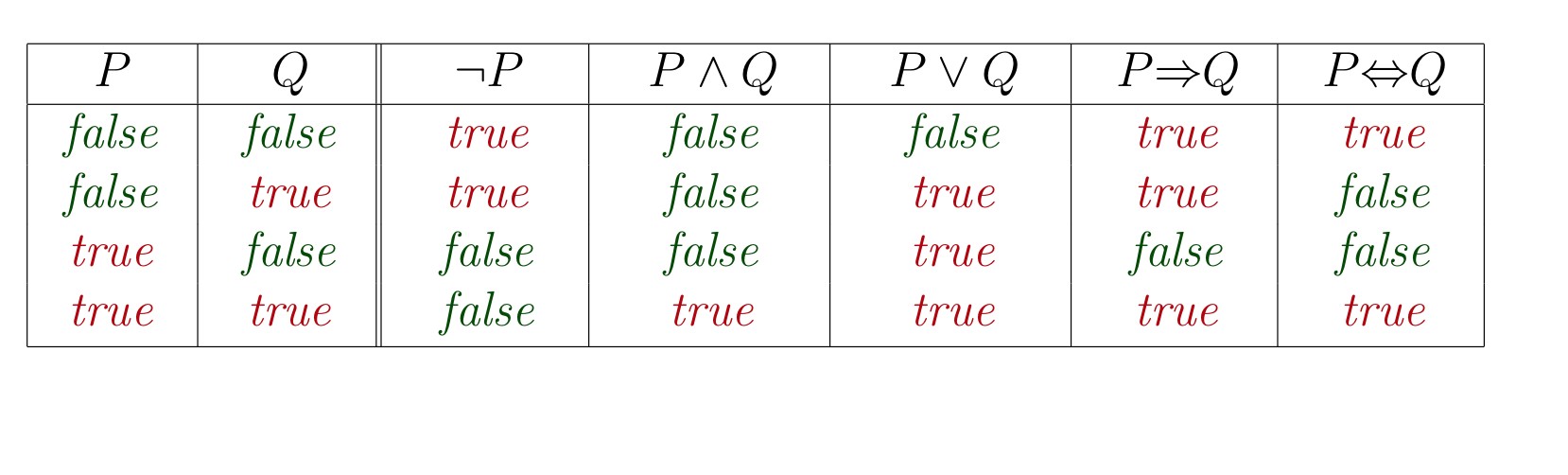


**Common Rules**



**Truth Tables for connectives**

## Summary:



**Wumpus world KB**

Let’s build the KB for the reduced Wumpus world.

|  |  |  |  |
| --- | --- | --- | --- |
| 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 **A**  **B OK** | 3,1 **P?** | 4,1 |

* Let *Pi,j* be true if there is a pit in [*i, j*]
* Let *Bi,j* be true if there is a breeze in [*i, j*]

¬*P*1*,*1

* “A square is breezy if and only if there is an adjacent pit”

*B*1*,*1 ⇔ *P*1*,*2 ∨ *P*2*,*1

*B*2*,*1 ⇔ *P*1*,*1 ∨ *P*2*,*2 ∨ *P*3*,*1

¬*B*1*,*1

*B*2*,*1

Let’s build the KB for the reduced Wumpus world.

|  |  |  |  |
| --- | --- | --- | --- |
| 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 **A**  **B OK** | 3,1 **P?** | 4,1 |

* Let *Pi,j* be true if there is a pit in [*i, j*]
* Let *Bi,j* be true if there is a breeze in [*i, j*] *R*1: ¬*P*1*,*1
* “A square is breezy if and only if there is an adjacent pit”

*R*2: *B*1*,*1 ⇔ *P*1*,*2 ∨ *P*2*,*1

*R*3: *B*2*,*1 ⇔ *P*1*,*1 ∨ *P*2*,*2 ∨ *P*3*,*1 *R*4: ¬*B*1*,*1

*R*5: *B*2*,*1

Let’s build the KB for the reduced Wumpus world.

|  |  |  |  |
| --- | --- | --- | --- |
| 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 **A**  **B OK** | 3,1 **P?** | 4,1 |

* Let *Pi,j* be true if there is a pit in [*i, j*]
* Let *Bi,j* be true if there is a breeze in [*i, j*] *R*1: ¬*P*1*,*1
* “A square is breezy if and only if there is an adjacent pit”

*R*2: *B*1*,*1 ⇔ *P*1*,*2 ∨ *P*2*,*1

*R*3: *B*2*,*1 ⇔ *P*1*,*1 ∨ *P*2*,*2 ∨ *P*3*,*1 *R*4: ¬*B*1*,*1

*R*5: *B*2*,*1

**Questions:** Based on this KB, is *KB* |= *P*1*,*2? Is *KB* |= *P*2*,*2?

**Entailment and Inference:**

* **Semantics**: Determine entailment by **Model Checking**, that is enumerate all models and show that the sentence *α* must hold in all models.

*KB* |= *α*

* **Syntax**: Determine entailment by **Theorem Proving**, that is apply rules of inference to KB to build a proof of *α without enumerating and checking all models*.

*KB* ► *α*

* But how are entailment and inference related?

**Soundness & Completeness**

* We want an inference algorithm that is:

1. **Sound**: does not infer false formulas, that is, derives only entailed sentences.

{*α*|*KB* ► *α*} ⊆ {*KB* |= *α*}

1. **Complete**: derives ALL entailed sentences.

{*α*|*KB* ► *α*} ⊇ {*KB* |= *α*}

**Validity & satisfiability**

* A sentence is valid (aka tautology) if it is true in all models, e.g., *True*, *p* ∨ ¬*p*, *p* ⇒ *p*, (*p* ∧ (*p* ⇒ *q*)) ⇒ *q*
* Validity is connected to inference via the Deduction Theorem:

*KB* |= *α IFF* (*KB* ⇒ *α*) is valid

* A sentence is satisfiable if it is true in some model e.g., *p* ∨ *q, r*
* A sentence is unsatisfiable if it is true in no models e.g., *p* ∧ ¬*p*
* Satisfiability is connected to inference via the following:

*KB* |= *α IFF* (*KB* ∧ ¬*α*) is unsatisfiable

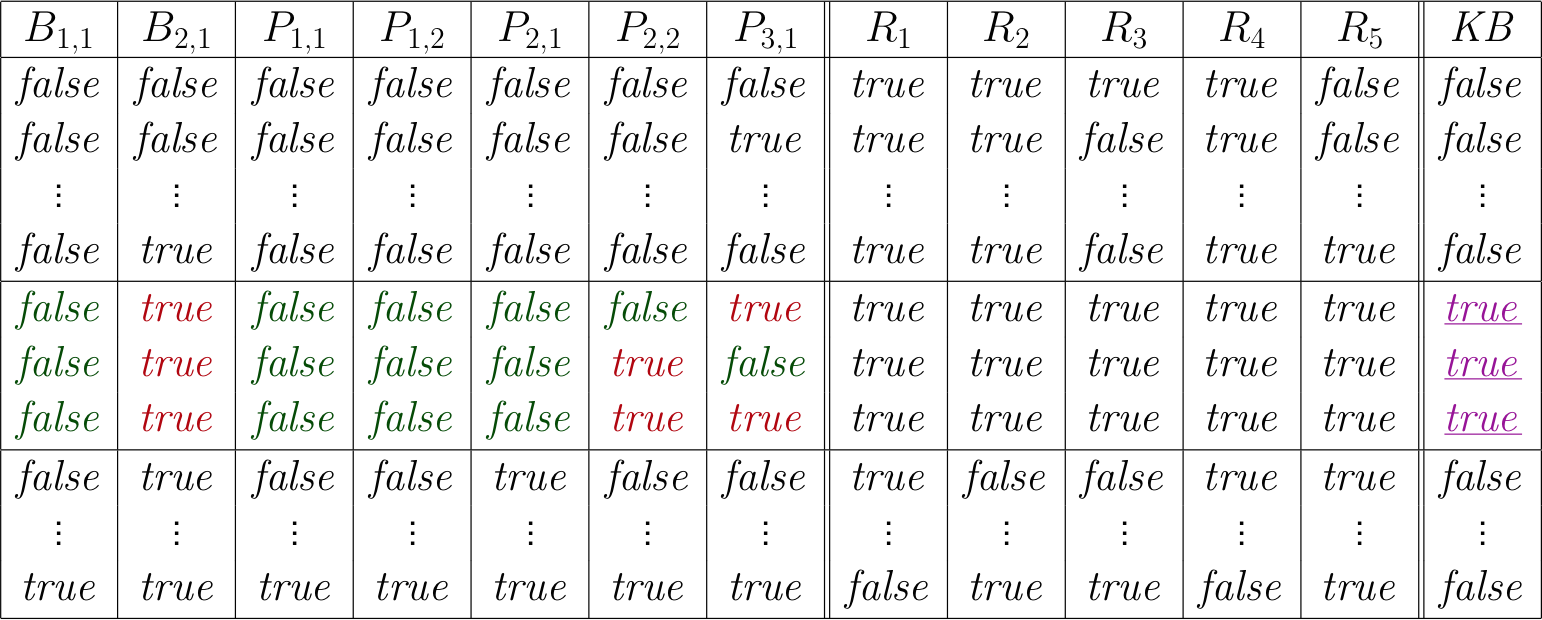
i.e., prove *α* by contradiction

**Determining entailment**

* Given a Knowledge Base (KB) (set of sentences in PL), given a query *α*, output whether KB entails *α*, noted: *KB* |= *α*
* We will see two ways of doing proofs in PL:
* **Model checking** enumerate the models (truth table enu- meration, exponential).
* **Application of inference rules** (proof checking/theorem proving): Syntactic derivations with rules like Modus Po- nens (Backward chaining and forward chaining). A proof is a sequence of inference rule applications.

**Model Checking**

* Truth Table for inference
* Model: assignment T/F to every propositional symbol.
* Check that *α* is true in every model in which KB is true.



**Inference Wumpus world**

*R*1: ¬*P*1*,*1

*R*2: *B*1*,*1 ⇔ *P*1*,*2 ∨ *P*2*,*1

*R*3: *B*2*,*1 ⇔ *P*1*,*1 ∨ *P*2*,*2

*R*4: ¬*B*1*,*1

*R*5: *B*2*,*1

**Inference as a search problem**

* **Initial state:** The initial KB
* **Actions:** all inference rules applied to all sentences that match the top of the inference rule
* **Results:** add the sentence in the bottom half of the inference rule
* **Goal:** a state containing the sentence we are trying to prove.

**Theorem proving:**

* Search for proofs is a more efficient way than enumerating models (We can ignore irrelevant information)
* Truth tables have an exponential number of models.
* The idea of inference is to repeat applying *inference rules* to the KB.
* Inference can be applied whenever suitable premises are found in the KB.
* Inference is sound. How about completeness?
* Two ways to ensure completeness:
* **Proof by resolution**: use powerful inference rules (resolu tion rule)
* **Forward or Backward chaining**: use of modus ponens on a restricted form of propositions (Horn clauses)
* Resolution: ONE single inference rule
* Invented by Robinson, 1965
* Resolution + Search = complete inference algorithm.

**Proof by Resolution:**

* Resolution & Wumpus world:

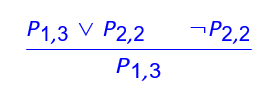
|  |  |  |  |
| --- | --- | --- | --- |
| 1,4 | 2,4 | 3,4 | 4,4 |
| 1,3 **W!** | 2,3 | 3,3 | 4,3 |
| 1,2 **A**  **S OK** | 2,2  **OK** | 3,2 | 4,2 |
| 1,1  **V OK** | 2,1 **B V OK** | 3,1 **P!** | 4,1 |

## Unit resolution:

## 

where *li* and *m* are complementary literals.

* **Example:**



* We call a **clause** a disjunction of literals.
* Unit resolution: Clause + Literal = New clause.

## Resolution inference rule (for CNF):

## 

where *li* and *mj* are complementary literals.

* Resolution **applies only to clauses**
* **Fact:** Every sentence in PL is logically equivalent to a con- junction of clauses.
* Conjunctive Normal Form (CNF): Conjunction of disjunction of literals:
* Example: (*A* ∨ ¬*B*) ∧ (*B* ∨ ¬*C* ∨ ¬*D*)
* Resolution inference rule (for CNF): sound and complete for propositional logic

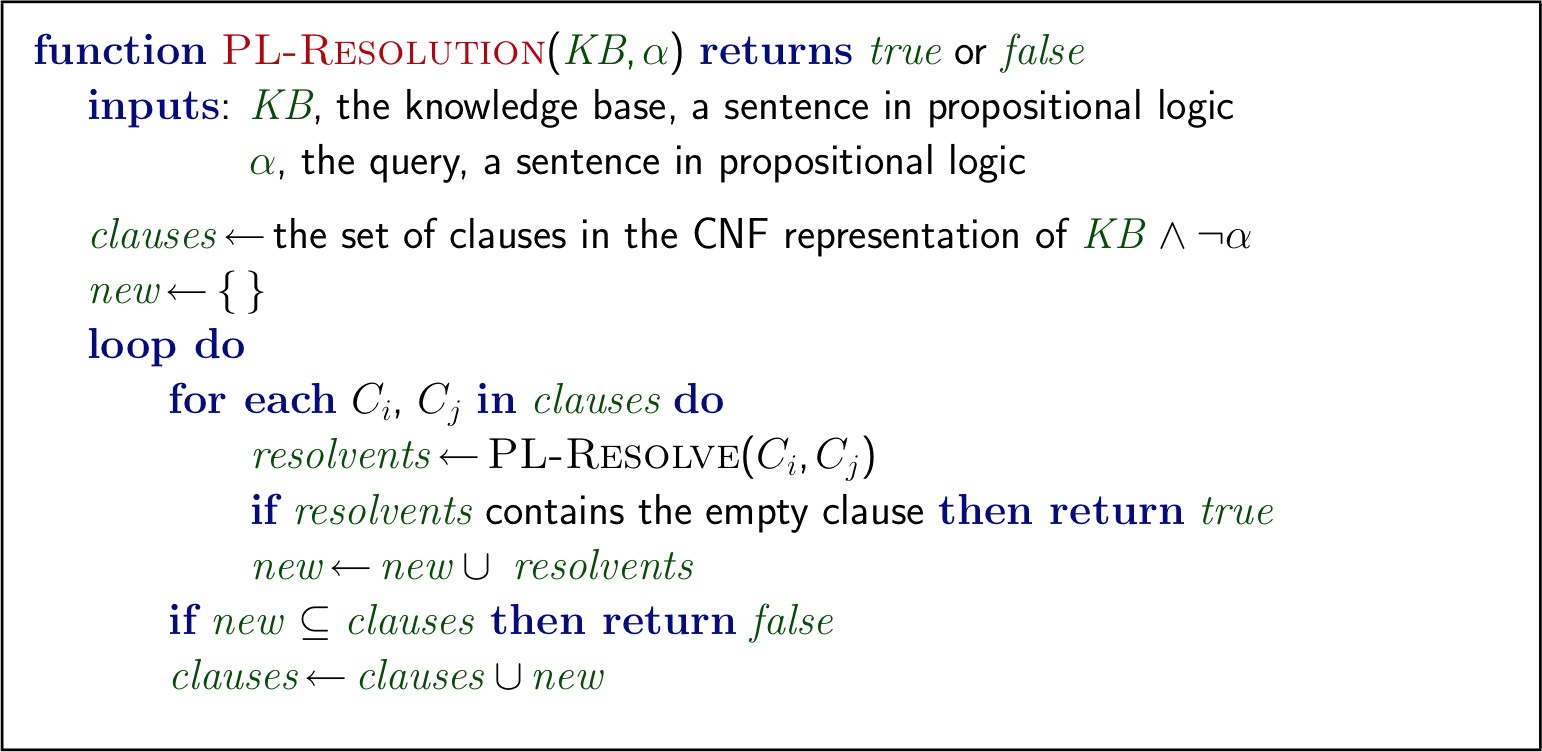
**Conversion to CNF**

*B*1*,*1 ⇔ (*P*1*,*2 ∨ *P*2*,*1)

1. Eliminate ⇔, replacing *α* ⇔ *β* with (*α* ⇒ *β*) ∧ (*β* ⇒ *α*). (*B*1*,*1 ⇒ (*P*1*,*2 ∨ *P*2*,*1)) ∧ ((*P*1*,*2 ∨ *P*2*,*1) ⇒ *B*1*,*1)

1. Eliminate ⇒, replacing *α* ⇒ *β* with ¬*α* ∨ *β*. (¬*B*1*,*1 ∨ *P*1*,*2 ∨ *P*2*,*1) ∧ (¬(*P*1*,*2 ∨ *P*2*,*1) ∨ *B*1*,*1)
2. Move ¬ inwards using de Morgans rules and double-negation: (¬*B*1*,*1 ∨ *P*1*,*2 ∨ *P*2*,*1) ∧ ((¬*P*1*,*2 ∧ ¬*P*2*,*1) ∨ *B*1*,*1)
3. Apply distributivity law (∨ over ∧) and flatten: (¬*B*1*,*1 ∨ *P*1*,*2 ∨ *P*2*,*1) ∧ (¬*P*1*,*2 ∨ *B*1*,*1) ∧ (¬*P*2*,*1 ∨ *B*1*,*1)

**Resolution algorithm:**

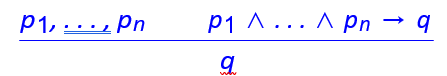


## Resolution example:

## 

**Forward/backward chaining:**

* KB = conjunction of Horn clauses
* Horn clauses: logic proposition of the form: *p*1 ∧ *. . .* ∧ *pn* → *q*
* Modus Ponens (for Horn Form): complete for Horn KBs

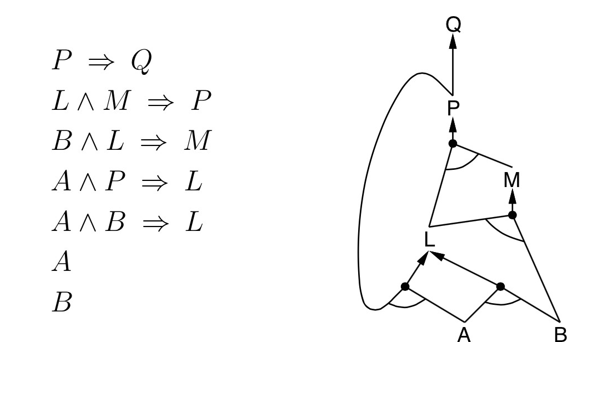


* Can be used with **forward chaining** or **backward chaining**.
* These algorithms are very natural and run in linear time

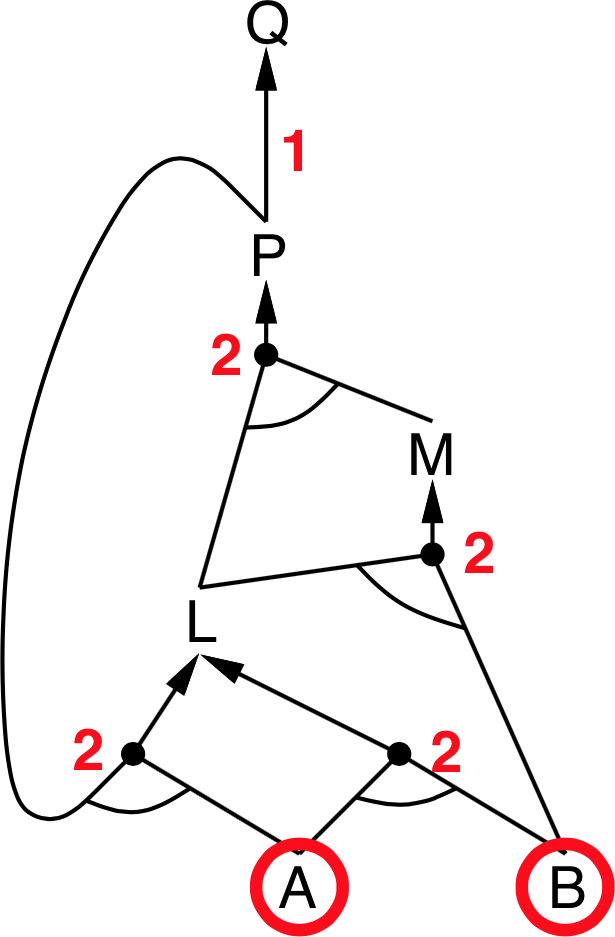
**Forward chaining:**

## Idea:

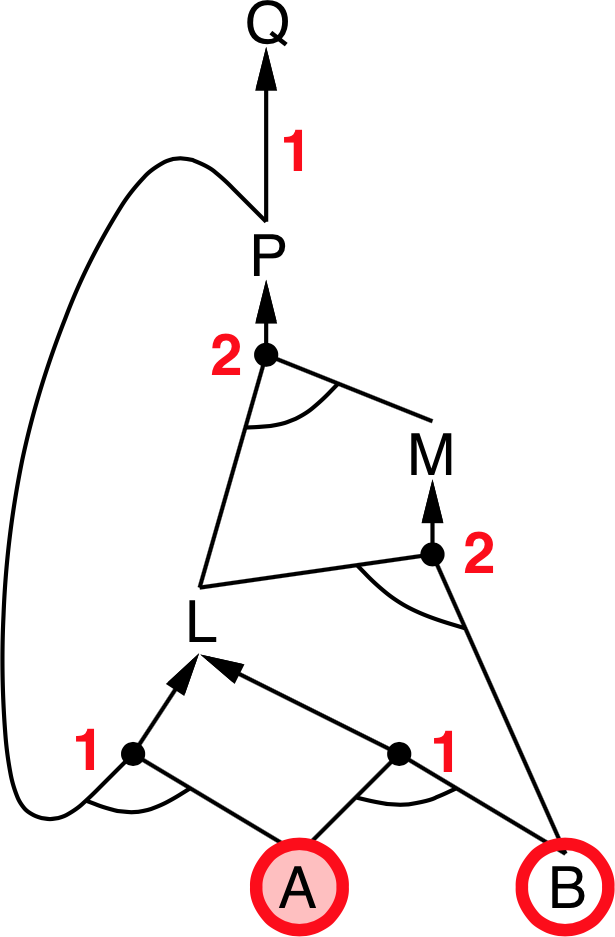
Fire any rule whose premises are satisfied in the KB, add its con- clusion to the KB, until query is found



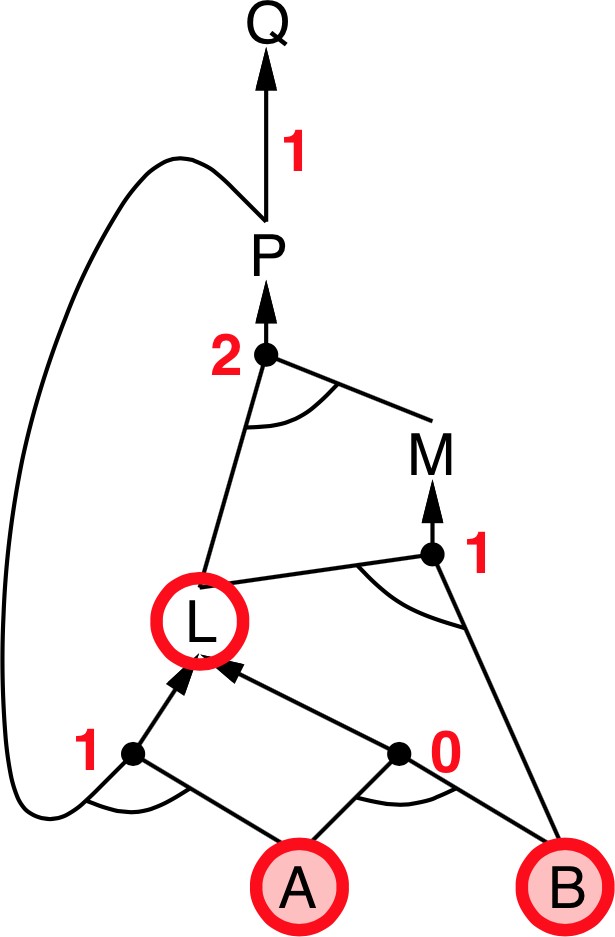
**Step 1:**



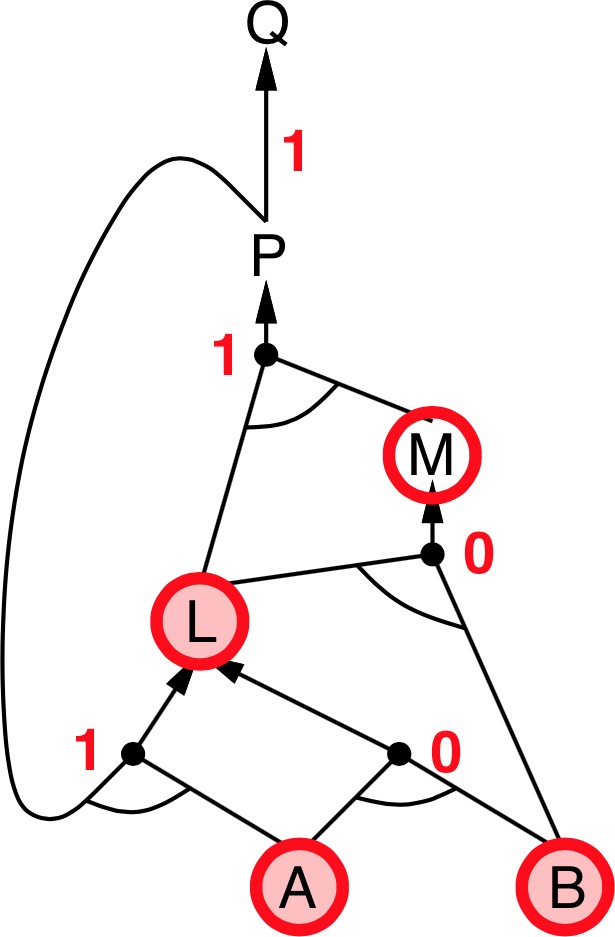
**Step 2:**



**Step 3:**



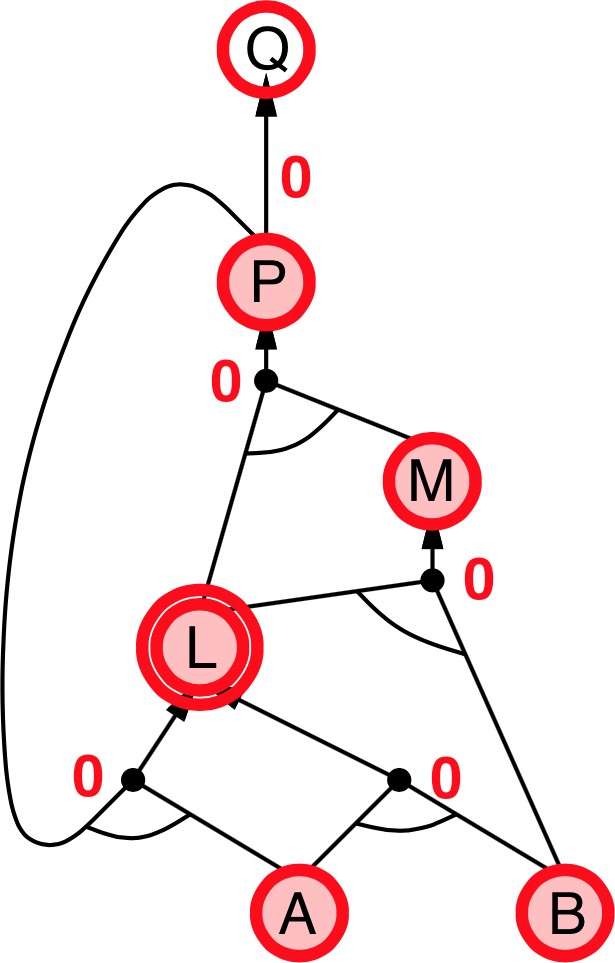
**Step 4:**



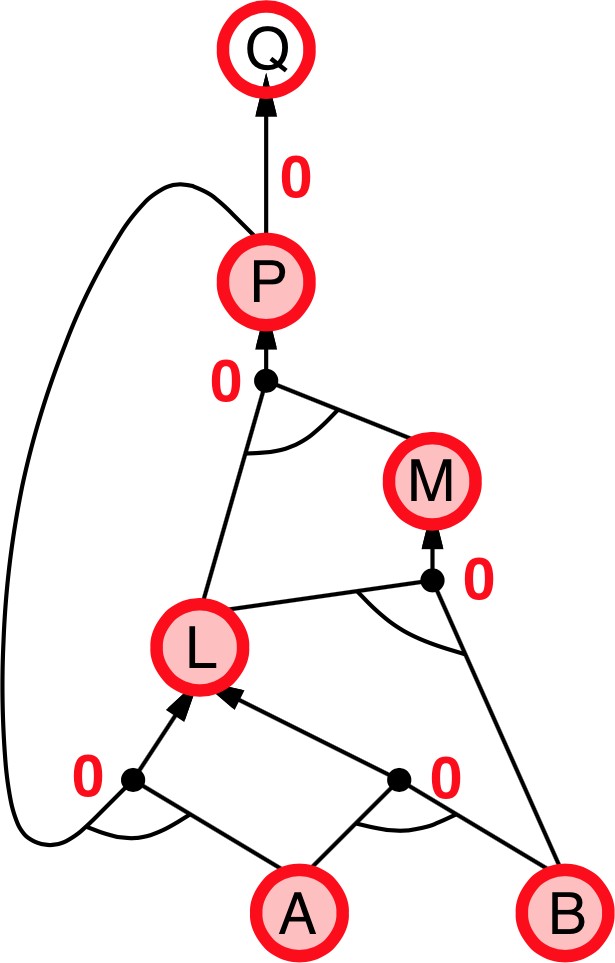
**Step 5:**



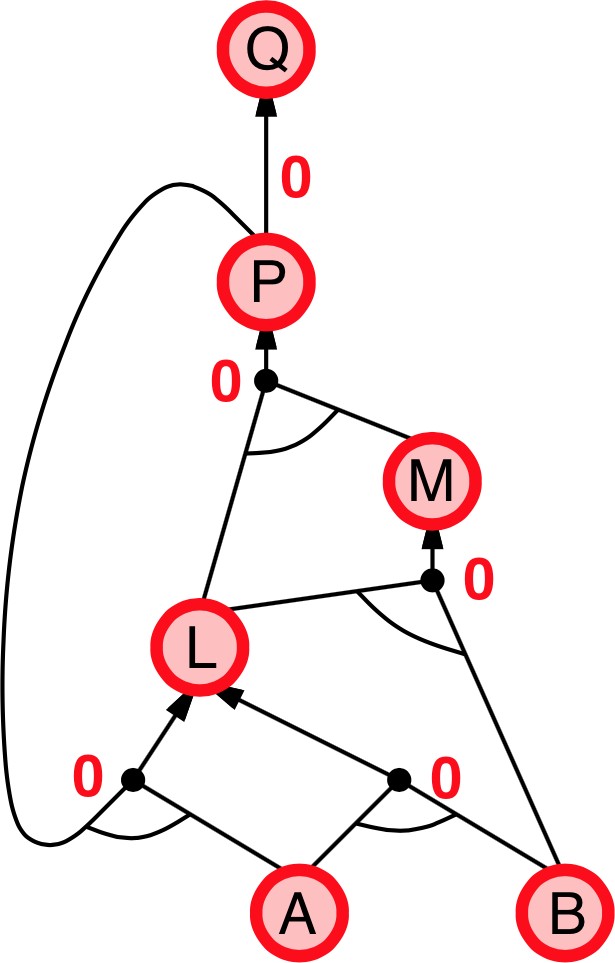
**Step 6:**



**Step 7:**



**Step 8:**



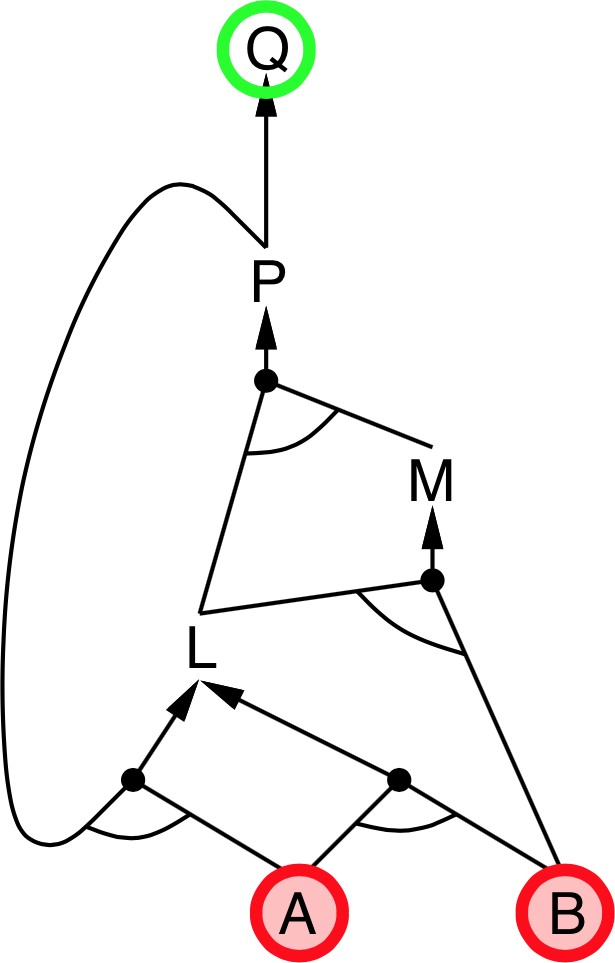
**Backward chaining:**

**Idea:** Works backwards from the query *q*

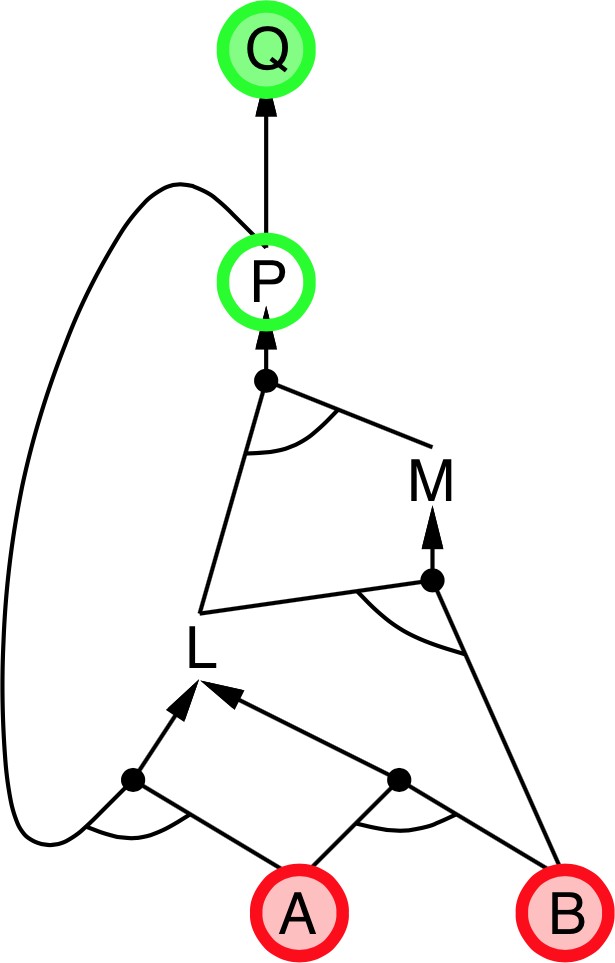
* to prove *q* by Backward Chaining:
  + Check if *q* is known already, or
  + Prove by Backward Chaining all premises of some rule con- cluding *q*
* Avoid loops: check if new subgoal is already on the goal stack
* Avoid repeated work: check if new subgoal
  + has already been proved true, or
  + has already failed

**Backward chaining example:**

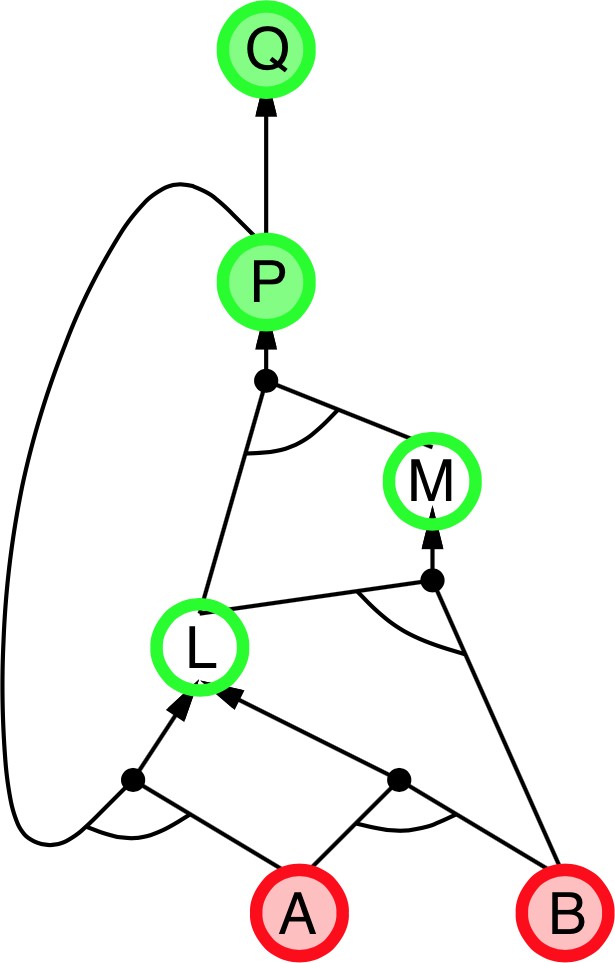
**Step 1:**



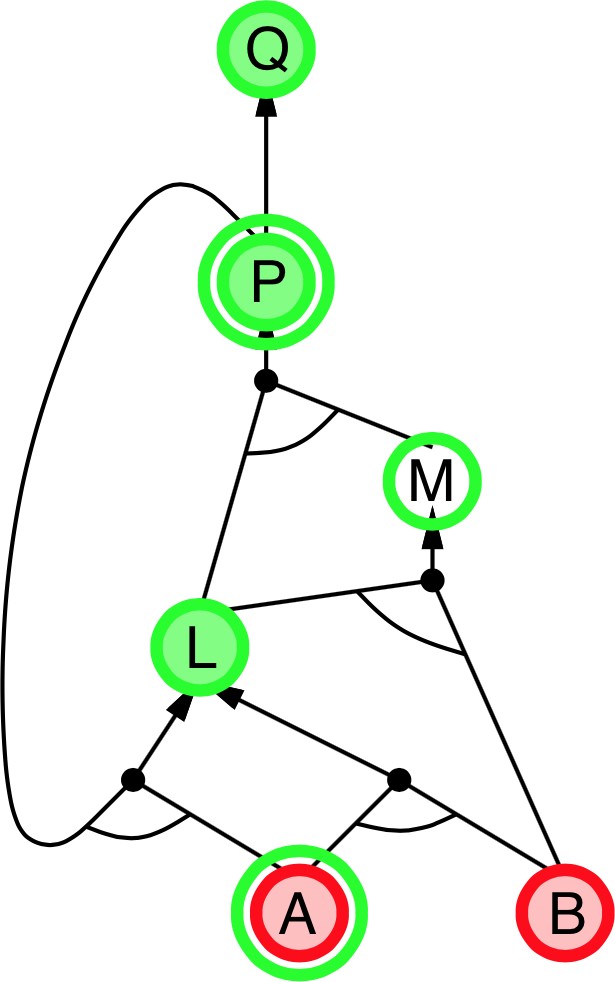
**Step 2:**



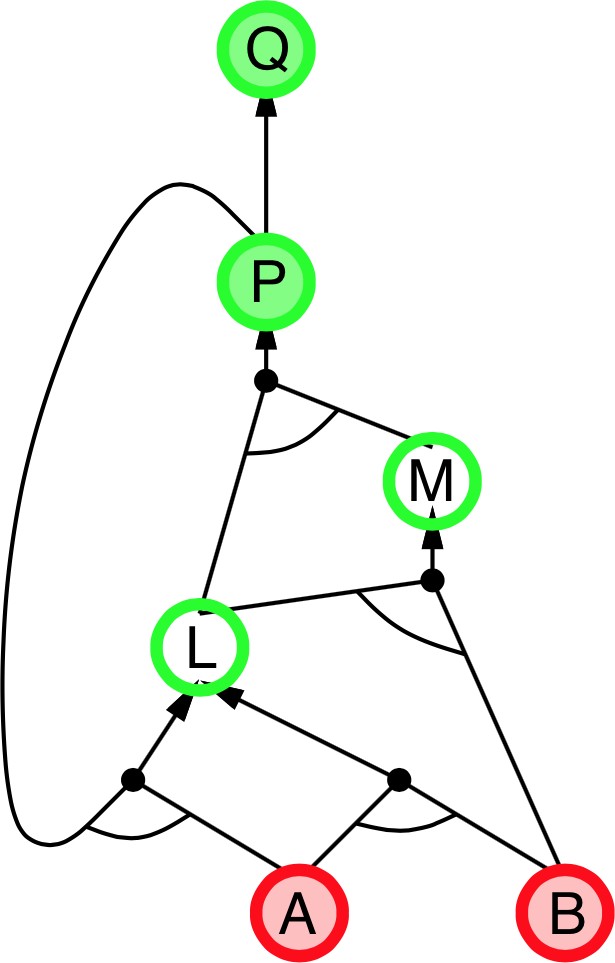
**Step 3:**



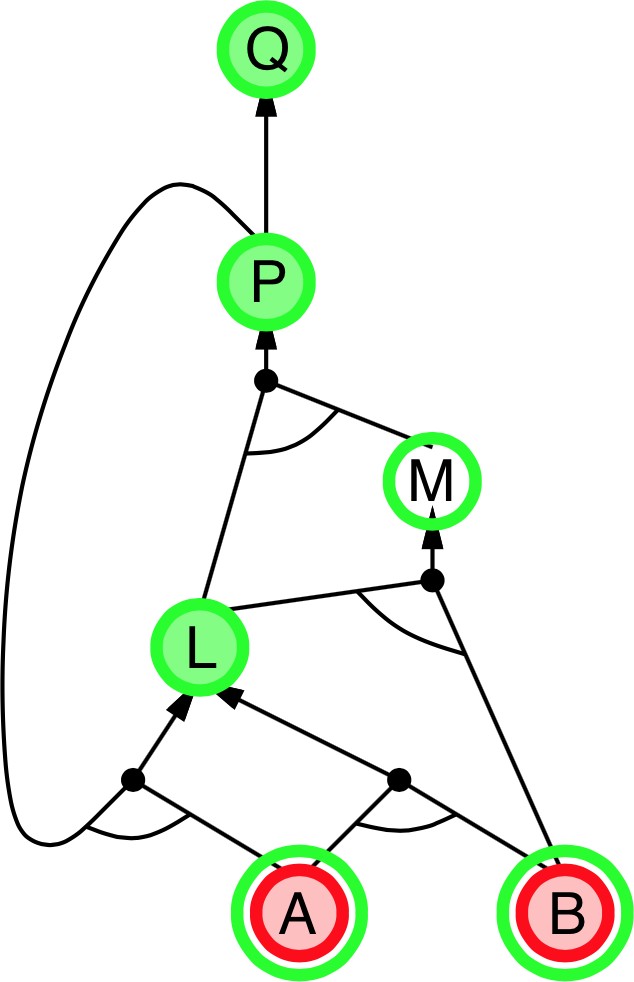
**Step 4:**



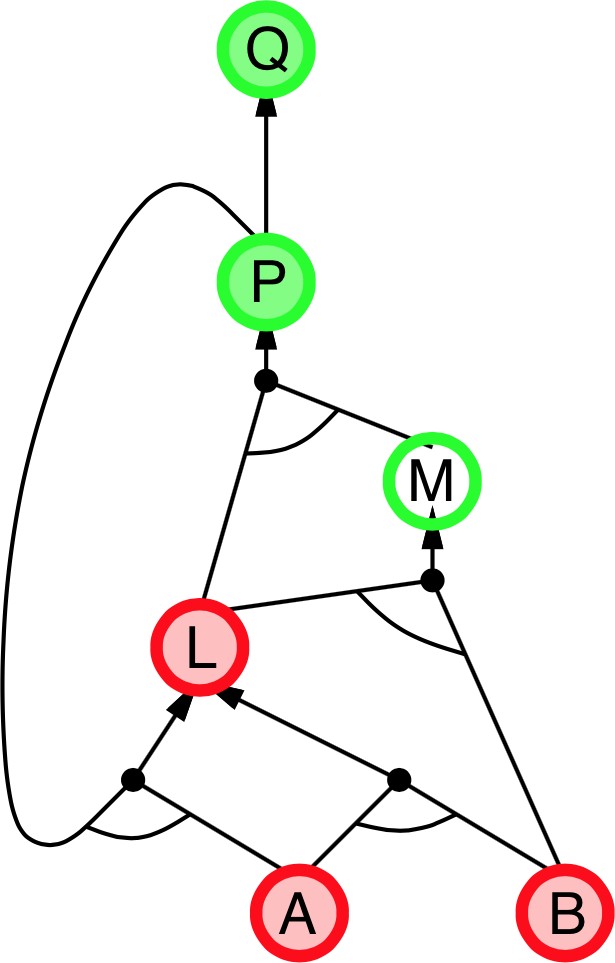
**Step 5:**



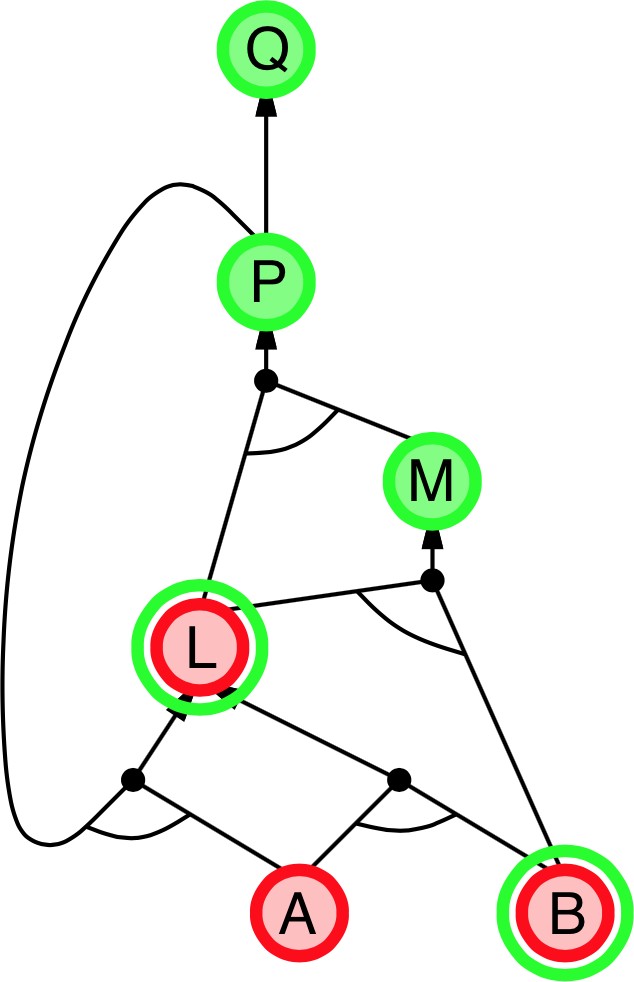
**Step 6:**



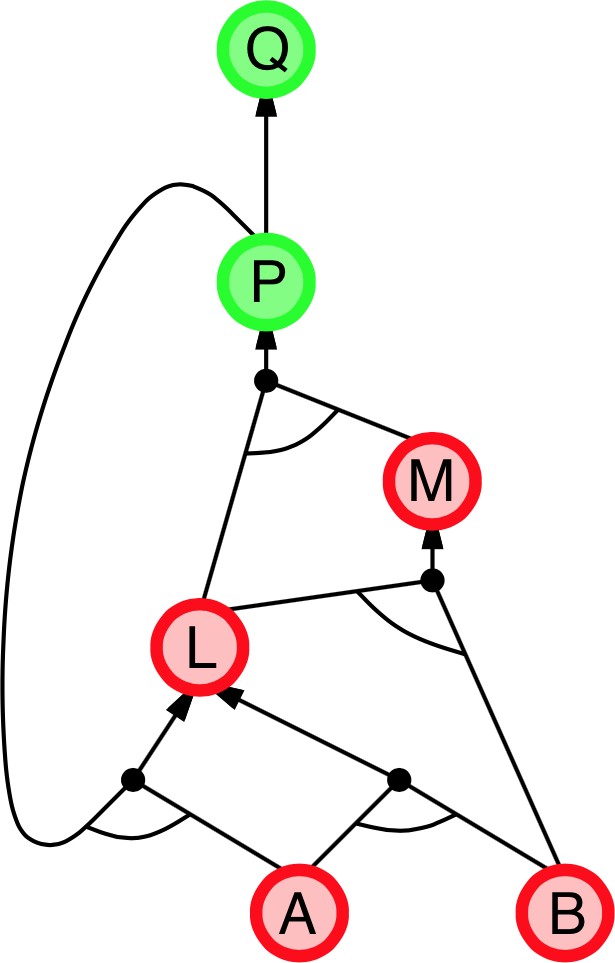
**Step 7:**



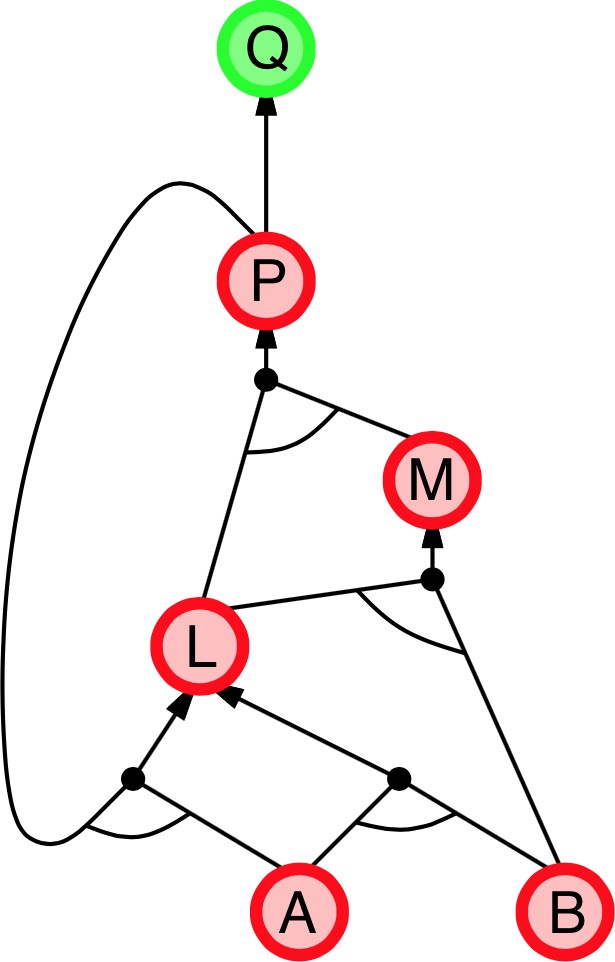
**Step 8:**



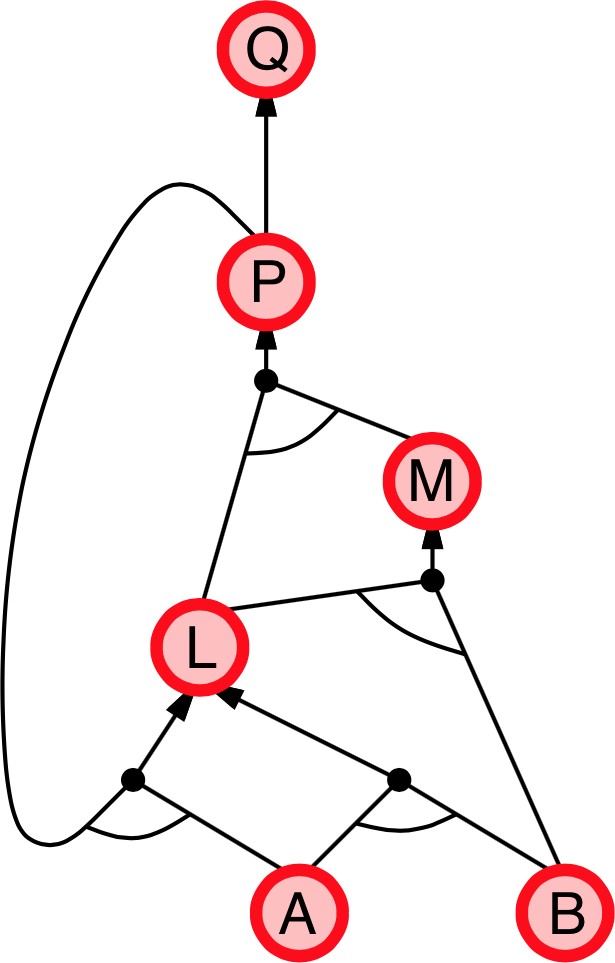
**Step 9:**



**Step 10:**



**Step 11:**



**Forward vs Backward:**

* Forward chaining:
* Data-driven, automatic, unconscious processing,
* May do lots of work that is irrelevant to the goal
* Backward chaining:

- Goal-driven, appropriate for problem-solving,

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

**Propositional Logic:**

* Propositional Logic (PL) is a formal language to describe the world around us.
* Logic can be used by an agent to model the world.
* Sentences in PL have a fixed **syntax**.
* With symbols and connectives we can form logical sentences:
  + Symbols or terms that can be either True or False or un- known.
  + Logical connectives

Example: *hot* ∧ *sunny* ⇒ *beach* ∨ *pool*

* Syntax and **Semantic** represent two important and distinct aspects in PL.
* Semantic: configurations/instantiations of the world.
* Modus Ponens inference rule:



* Example:



* Modus Ponens deals with Horn clauses
* Horn clauses: logic proposition of the form: *p*1 ∧ *. . .* ∧ *pn* → *q*
* **Inference:** we want an inference algorithm that is:
  1. sound (does not infer false formulas), and
  2. ideally, complete too (derives all true formulas).

## Inference in PL with horn clauses is sound and complete.

* Limits of PL?

1. PL is not expressive enough to describe all the world around us. It can’t express information about different object and the relation between objects.
2. PL is not compact. It can’t express a fact for a set of objects without enumerating all of them which is sometimes impossible.

* Example: We have a vacuum cleaner (Roomba) to clean a 10×10 squares in the classroom. Use PL to express information about the squares.
* The proposition *square*1 *is clean* expresses information about square1 being clean. How can one write this in a less heavy

way?

* How can we express that all squares in the room are clean?

*square*1 *is clean* ∧ *square*2 *is clean* ∧ *. . .* ∧ *square*100 *is clean*

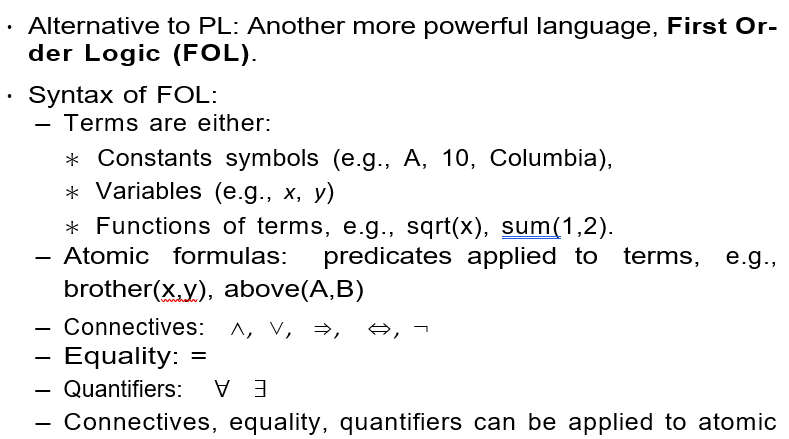
* How can we express that some squares in the room are clean?

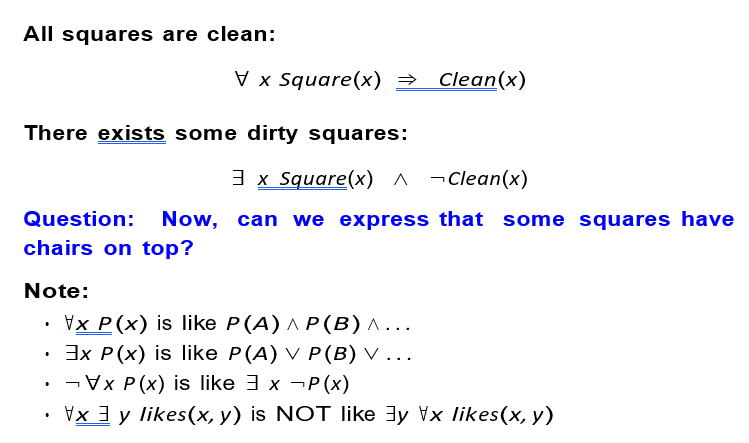
*square*1 *is clean* ∨ *square*2 *is clean* ∨ *. . .* ∨ *square*100 *is clean*

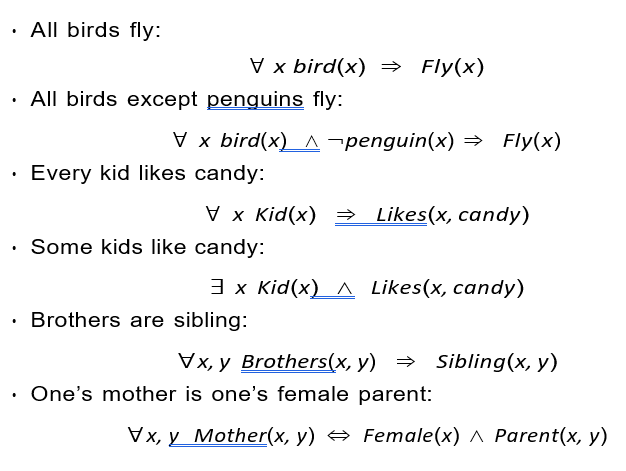
* How can we express that some squares have chairs on them?

*square*1 *has chair* ∨ *square*2 *has chair* ∨ *. . .* ∨ *square*100 *has chair*

**First Order Logic:**

 formulas to create sentences in FOL.





**Inference:** While a bit more complicated than PL, there are procedures to do inference with a knowledge base of FOL formulas (Further optional reading: book chapter 8, 9).

**Natural language**: The expressiveness of FOL suggests that it is possible to automate the conversion between natural language and logical expressions. This is very valuable for different applications, such as personal assistants (Siri), question/answering systems, and communicating with computers in general.

**Summary:**

* Logical agents apply inference to a knowledge base to derive new information and make decisions
* Basic concepts of logic:
  + **Syntax**: formal structure of sentences
  + **Semantics**: truth of sentences wrt models
  + **Entailment**: necessary truth of one sentence given another
  + **Inference**: deriving sentences from other sentences
  + **Soundness**: derivations produce only entailed sentences
  + **Completeness**: derivations can produce all entailed sen- tences
* Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
* Forward, backward chaining are linear in time, complete for Horn clauses Resolution is complete for propositional logic.
* Building logical agents was a main research trend in AI before the mid-nineties
* Logic is used in AI to represent the environment of the agent and reason about that environment
* Pros and cons of logical agents:
* Do not handle uncertainty, probability does
* Rule-based and do not use data, ML does
* It is hard to model every aspect of the world

+ Intelligibility of models: models are encoded explicitly