CMSC 471: Artificial Intelligence
Spring 2021

Propositional and First-Order Logic

Frank Ferraro – ferraro@umbc.edu

Many slides courtesy Tim Finin

Logic roadmap overview

- **Propositional logic**
  - Problems with propositional logic
- **First-order logic**
  - Properties, relations, functions, quantifiers, …
  - Terms, sentences, wffs, axioms, theories, proofs, …
  - Variations and extensions to first-order logic
- **Logical agents**
  - Reflex agents
  - Representing change: situation calculus, frame problem
  - Preferences on actions
  - Goal-based agents

Disclaimer

"Logic, like whiskey, loses its beneficial effect when taken in too large quantities."

- *Lord Dunsany*

Big Ideas

- **Logic**: great knowledge representation (KR) language for many AI problems
- **Propositional logic**: simple foundation and fine for many AI problems
- **First order logic (FOL):** more expressive as a KR language; needed for many AI problems
- **Variations** on classical FOL are common: horn logic, higher-order logic, modal logic, three-valued logic, probabilistic logic, fuzzy logic, etc.

AI Use Cases for Logic

Logic has many use cases even in a time dominated by deep learning, including these examples:

- Modeling and using knowledge
- Allowing agents to develop complex plans to achieve a goal and create optimal plans
- Defining and using semantic knowledge graphs such as schema.org and Wikidata
- Adding features to neural network systems

Knowledge-Based Agents: Big Idea

- Drawing reasonable conclusions from a set of data (observations, beliefs, etc.) seems key to intelligence
- Logic is a powerful and well-developed approach to this & highly regarded by people
- Logic is also a strong formal system that computers can use (cf. John McCarthy)
- We can solve some AI problems by representing them in logic and applying standard proof techniques to generate solutions

Inference in People

- People can do logical inference, but are not always very good at it
- Reasoning with negation and disjunction seems particularly difficult
- But, people seem to employ many kinds of reasoning strategies, most of which are neither *complete* nor *sound*

Question #1

Here is a simple puzzle

Don’t try to solve it -- listen to your intuition

Question #1

Here is a simple puzzle

Don’t try to solve it -- listen to your intuition

- A bat and ball cost $1.10
- The bat costs one dollar more than the ball
- How much does the ball cost?

Question #1

Here is a simple puzzle

Don’t try to solve it -- listen to your intuition

- A bat and ball cost $1.10
- The bat costs one dollar more than the ball
- How much does the ball cost?

The ball costs $0.05

Question #2

Try to determine, as quickly as you can, if the argument is logically valid. Does the conclusion follow the premises?

Question #2

Try to determine, as quickly as you can, if the argument is logically valid. Does the conclusion follow the premises?

- **All roses are flowers**
- **Some flowers fade quickly**
- **Therefore some roses fade quickly**

Question #2

Try to determine, as quickly as you can, if the argument is logically valid. Does the conclusion follow the premises?

- All roses are flowers
- Some flowers fade quickly
- Therefore some roses fade quickly

It is possible that there are no roses among the flowers that fade quickly

Question #3

It takes 5 machines 5 minutes to make 5 widgets

How long would it take 100 machines to make 100 widgets?

Question #3

It takes 5 machines 5 minutes to make 5 widgets

How long would it take 100 machines to make 100 widgets?

- **100 minutes or 5 minutes?**

Question #3

It takes 5 machines 5 minutes to make 5 widgets

How long would it take 100 machines to make 100 widgets?

- 100 minutes or 5 minutes?

5 minutes

Wason Selection Task

- I have a pack of cards; each has a *letter* written on one side and a *number* on the other
- I claim the following rule is true:

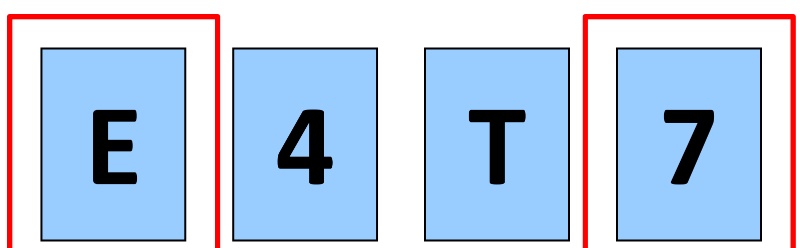
  If a card has a *vowel* on one side, then it has an *even number* on the other
- Which cards should you turn over in order to decide whether the rule is true or false?

E	**4**	**T**	**7**

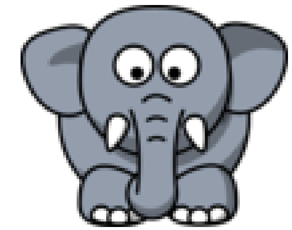
Wikipedia

Wason Selection Task

- Wason (1966) showed that people are bad at this task
- To disprove rule P=>Q, find a situation in which P is true but Q is false, i.e., show P^~Q
- To disprove **vowel => even**, find a card with a vowel and an odd number
- Thus, turn over the cards showing **vowels** and those showing **odd numbers**



Negation in Natural Language



- We often model the meaning of natural language sentences as a logic statements
- This maps these into equivalent statements
  - All elephants are gray
  - No elephant are not gray
- Double negation is common in informal language: th*at won't do you no good*
- But what does this mean: *we cannot underestimate the importance of logic*

Logic as a Methodology

Even if people don’t use formal logical reason-ing for solving a problem, logic might be a good approach for AI for a number of reasons

- Airplanes don’t need to flap their wings
- Logic may be a good implementation strategy
- Solution in a formal system can offer other benefits, e.g., letting us prove properties of the approach

- See neats vs. scruffies

Knowledge-based agents

- Knowledge-based agents have a knowledge base (KB) and an inference system
- KB: a set of representations of facts believed true
- Each individual representation is called a **sentence**
- Sentences are expressed in a **knowledge represent-ation language**
- The agent operates as follows:
  1. It **TELL**s the KB what it perceives
  2. It **ASK**s the KB what action it should perform
  3. It performs the chosen action

Architecture of a KB agent

- **Knowledge Level**
  - Most abstract: describe agent by what it knows
  - Ex: Autonomous vehicle knows Golden Gate Bridge connects San Francisco with the Marin County
- **Logical Level**
  - Level where knowledge is encoded into *sentences*
  - Ex: **links**(GoldenGateBridge, SanFran, MarinCounty)
- **Implementation Level**
  - Software representation of sentences, e.g.
    `(links goldengatebridge sanfran marincounty)`

Does your agent have complete knowledge?

- Closed world assumption (CWA): the lack of knowledge is assumed to mean it's false
- Open world assumption: no such assumption is made

Q: Why would we ever make a closed world assumption?

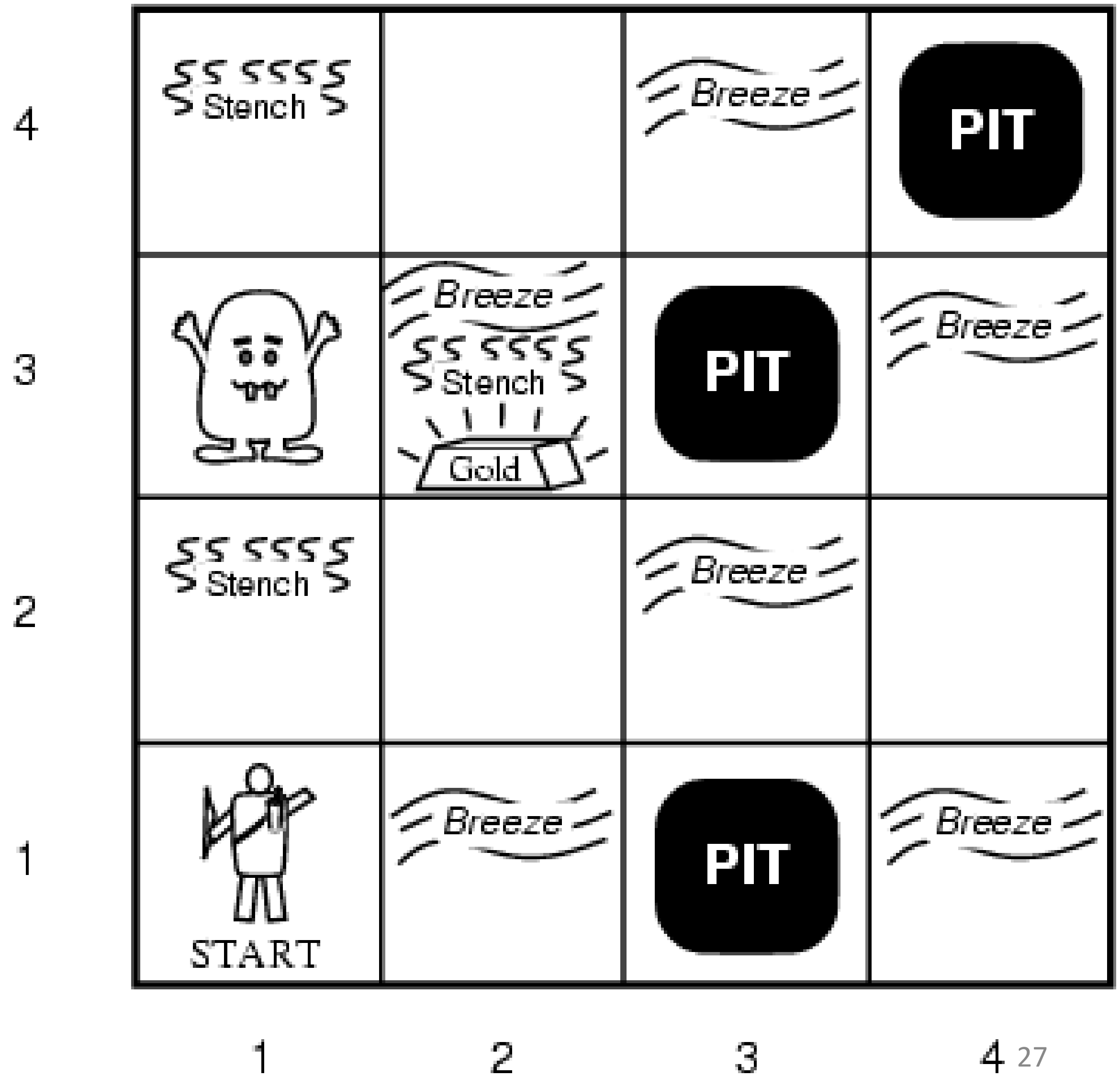
Wumpus World environment

- Based on Hunt the Wumpus computer game
- Agent explores cave of rooms connected by passageways
- Lurking in a room is the *Wumpus*, a beast that eats any agent that enters its room
- Some rooms have *bottomless pits* that trap any agent that wanders into the room
- Somewhere is a heap of gold in a room
- Goal: collect gold & exit w/o being eaten

AIMA's Wumpus World

The agent always starts in the field [1,1]

Agent's task is to find the gold, return to the field [1,1] and climb out of the cave



Agent in a Wumpus world: Percepts

- The agent perceives
  - **stench** in square containing Wumpus and in adjacent squares (not diagonally)
  - **breeze** in squares adjacent to a pit
  - **glitter** in the square where the gold is
  - **bump**, if it walks into a wall
  - Woeful **scream** everywhere in cave, if Wumpus killed
- Percepts given as five-tuple, e.g., if stench and breeze, but no glitter, bump or scream:

  [Stench, Breeze, None, None, None]
- Agent cannot perceive its location, e.g., (2,2)

Wumpus World Actions

- **go forward**
- **turn right** 90 degrees
- **turn left** 90 degrees
- **grab**: Pick up object in same square as agent
- **shoot**: Fire arrow in direction agent faces. It continues until it hits & kills Wumpus or hits outer wall. Agent has one arrow, so only first shoot action has effect
- **Climb:** leave cave, only effective in start square
- **die**: automatically and irretrievably happens if agent enters square with pit or living Wumpus

Wumpus World Goal

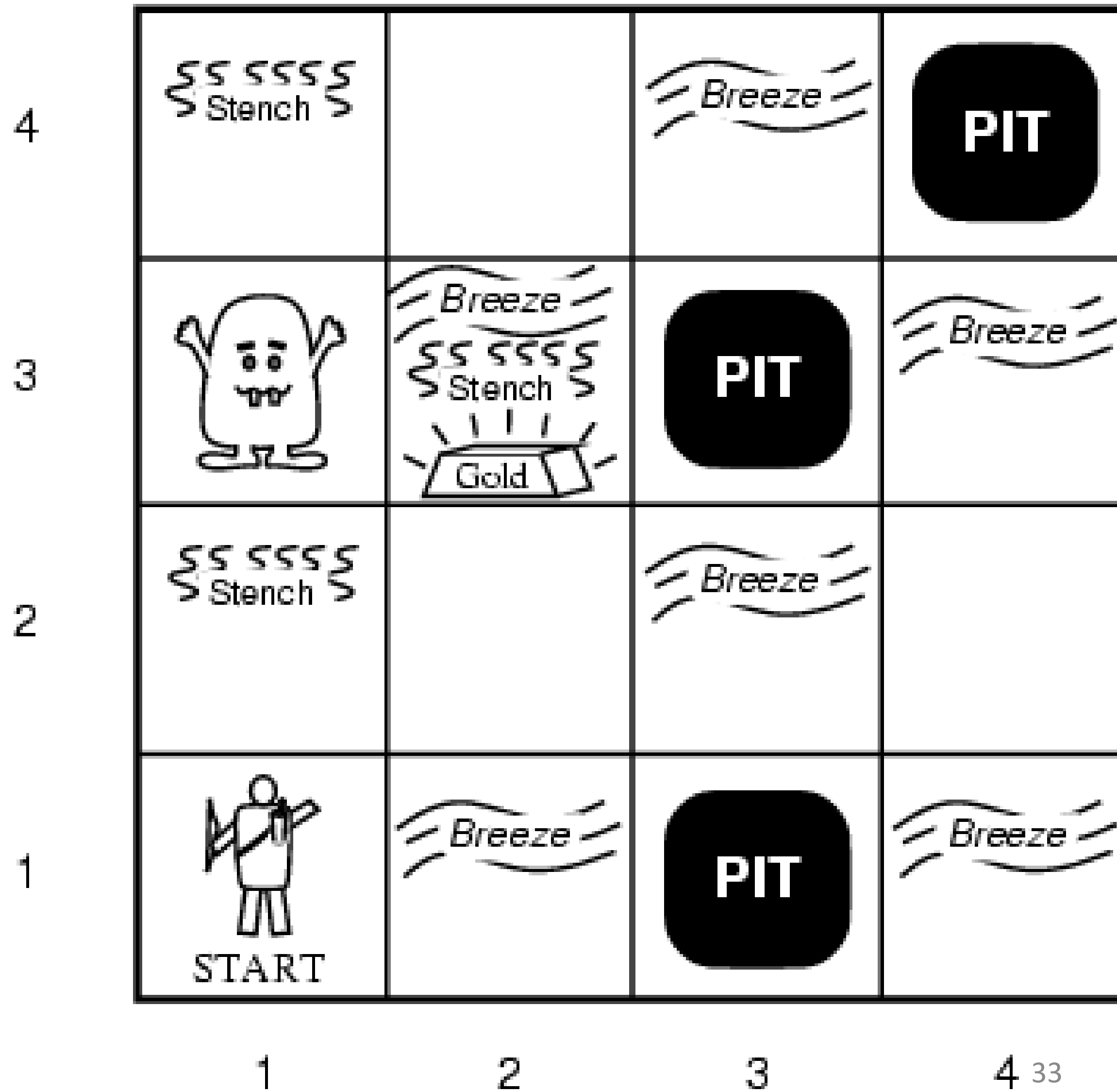
Agent’s goal is to find the gold and bring it back to the start square as quickly as possible, without getting killed

- 1,000 point reward for climbing out of cave with gold
- 1 point deducted for every action taken
- 10,000 point penalty for getting killed

AIMA's Wumpus World

The agent always starts in the field [1,1]

Agent's task is to find the gold, return to the field [1,1] and climb out of the cave



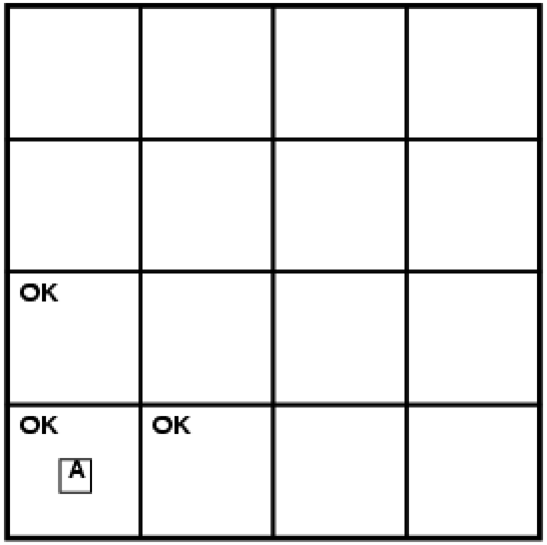
The Hunter’s first step

(a) (b)

Since agent is alive and perceives neither breeze nor stench at [1,1], it **knows** [1,1] and its neighbors are OK

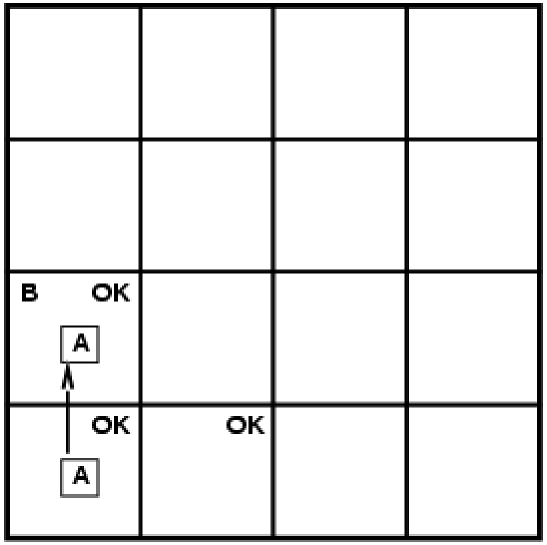
Moving to [2,1] is a **safe move** that reveals a breeze but no stench, **implying** that Wumpus isn’t adjacent but one or more pits are

Exploring a wumpus world



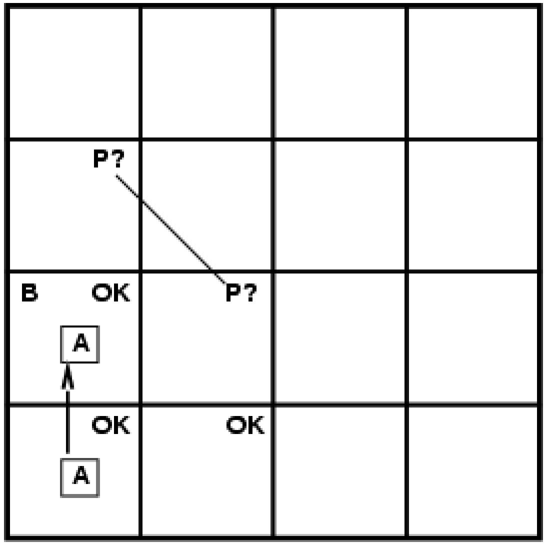
A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Exploring a wumpus world



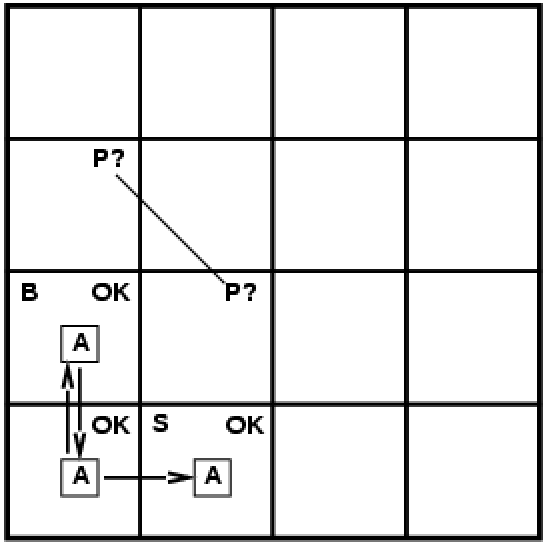
A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Exploring a wumpus world



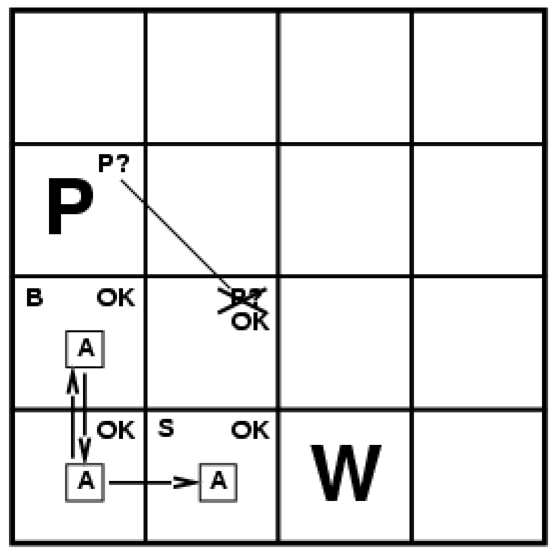
A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

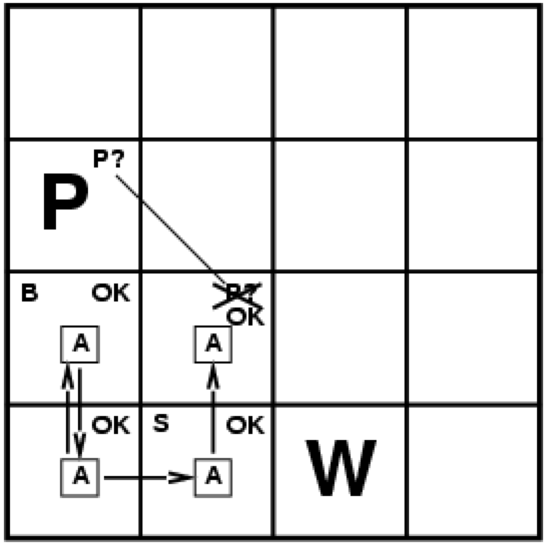
Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

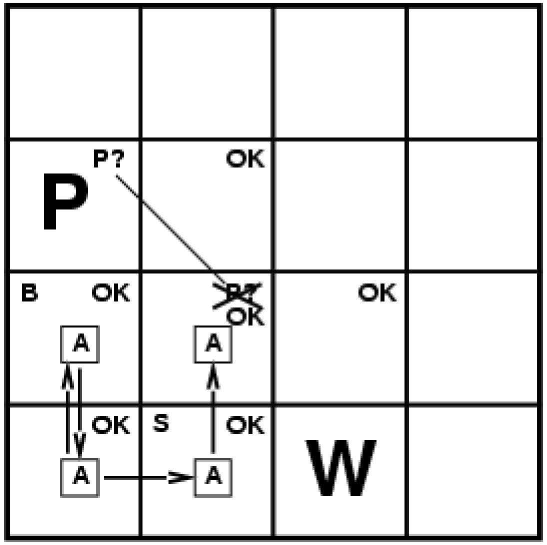
No stench in (1,2) => Wumpus not in (2,2)
No breeze in (2,1) => no pit in (2,2) => pit in (1,3)

Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

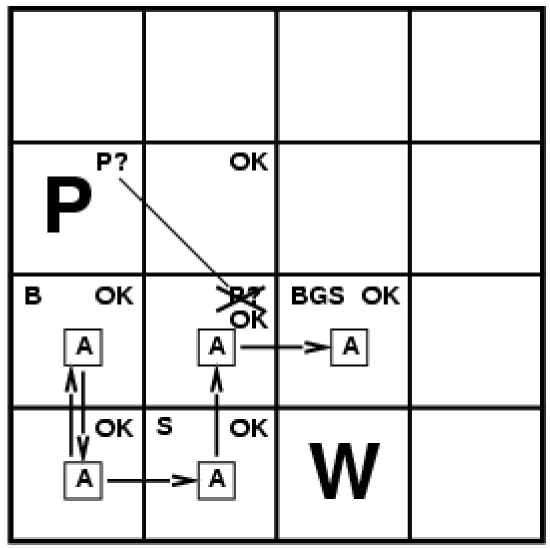
Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Going to (2,2) is the only “safe” move

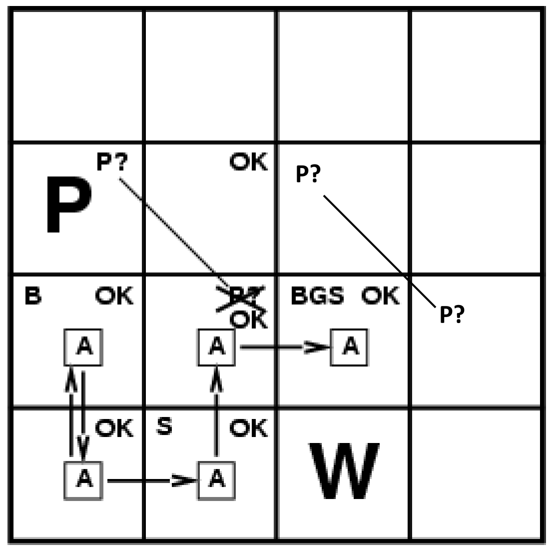
Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Going to (2,3) is a “safe” move

Exploring a wumpus world



A	agent
B	breeze
G	glitter
OK	safe cell
P	pit
S	stench
W	wumpus

Found gold! Now find way back to (1,1)

Logic in general

- **Logics** are formal languages for representing information so that conclusions can be drawn
- **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., the language of arithmetic

- $x+2 \geq y$ is a sentence; $x2+y > \{\}$ is not a sentence
- $x+2 \geq y$ is true iff the number $x+2$ is no less than the number y
- $x+2 \geq y$ is true in a world where $x = 7, y = 1$
- $x+2 \geq y$ is false in a world where $x = 0, y = 6$
- $x+1 > x$ is true for all numbers x

Extension vs. Intension

- Extension: direct enumeration/listing of values that can satisfy constraints

- Intension: constraint-satisfying values provided via formulas

Entailment

- Entailment: one thing follows from another

Entailment

- Entailment: one thing follows from another
- Written as: KB $\models \alpha$

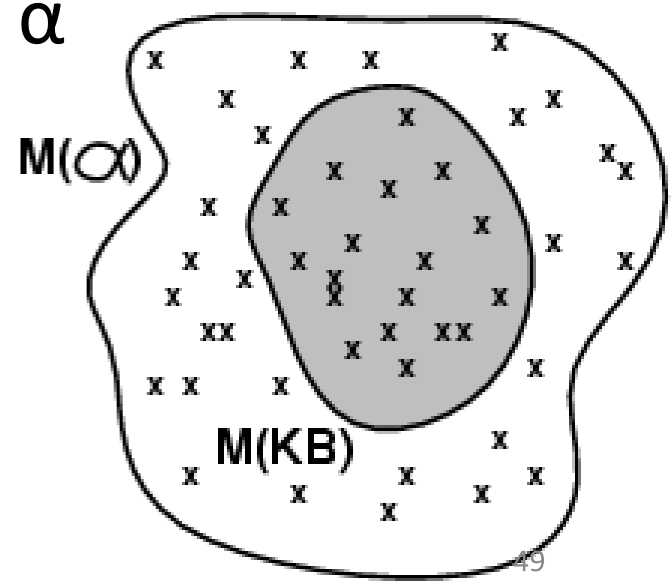
Entailment

- Entailment: one thing follows from another
- Written as: KB $\models$ α
- Knowledge base *KB* entails sentence α iff α is true in *all possible worlds* where *KB* is true
  - E.g., the KB containing "UMBC won" and "JHU won" entails "Either UMBC won or JHU won"
  - E.g., x+y = 4 entails 4 = x+y
  - Entailment is a relationship between (sets of) sentences (i.e., syntax) that is based on **semantics**

Models

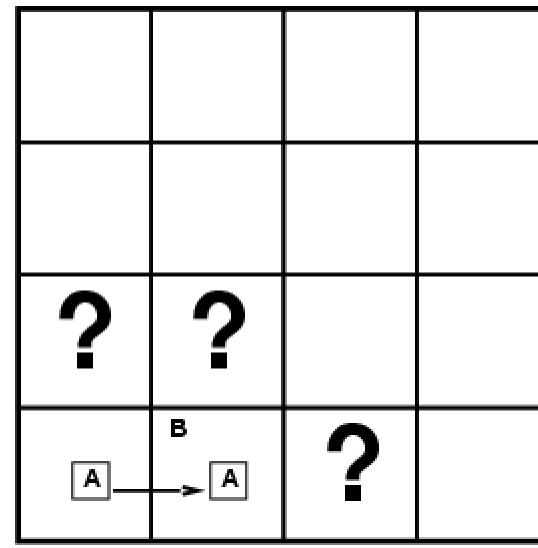
- Logicians talk of models: formally structured worlds w.r.t which truth can be evaluated
- *m* is a model of sentence α if α is true in *m*

  Lots of other things might or might not be true or might be unknown in *m*
- *M(α)* is the set of all models of α
- Then KB $\models$ α iff *M(KB)* $\subseteq$ *M*(α)
  - *KB* = UMBC and JHU won
  - α = UMBC won
  - Then KB $\models$ α



Entailment in the Wumpus World

- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]
- Possible models for *KB* assuming only pits and restricting cells to {(1,3)(2,1)(2,2)}
- Two observations: ~B11, B12
- Three propositional variables variables: P13, P21, P22
- $\Rightarrow$ 8 possible models

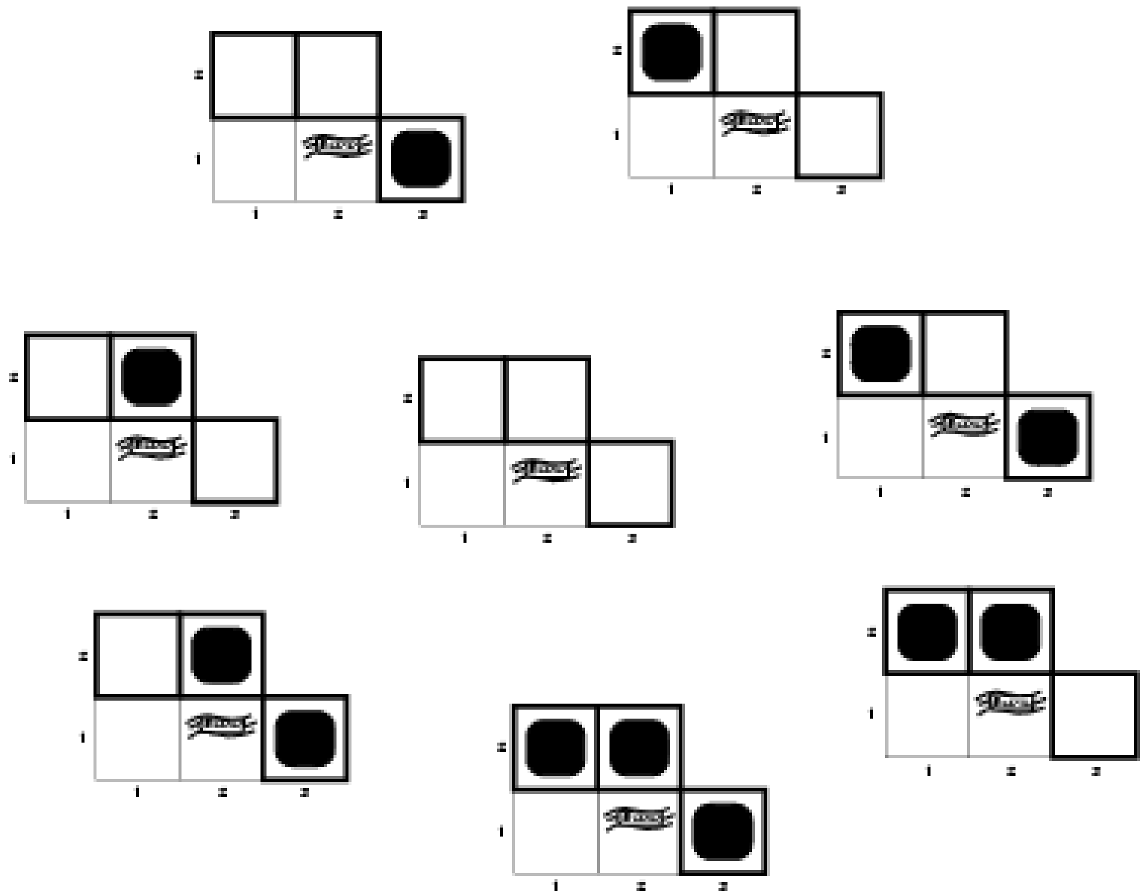


B11: breeze in (1,1)
P13: pit in (1,3)

Wumpus models

P13	P21	P22
F	F	F
F	F	T
F	T	F
F	T	T
T	F	F
T	F	T
T	T	F
T	T	T

Each row is a possible world



Wumpus World Rules (1)

- **If a cell has a pit, then a breeze is observable in every adjacent cell**
- In propositional calculus we can not have rules with variables (e.g., forall X…)

P11 => B21

P11 => B12

P21 => B11

P21 => B22 …

If a pit in (1,1) then a breeze in (2,1), …

these also follow

~B21 => ~P11

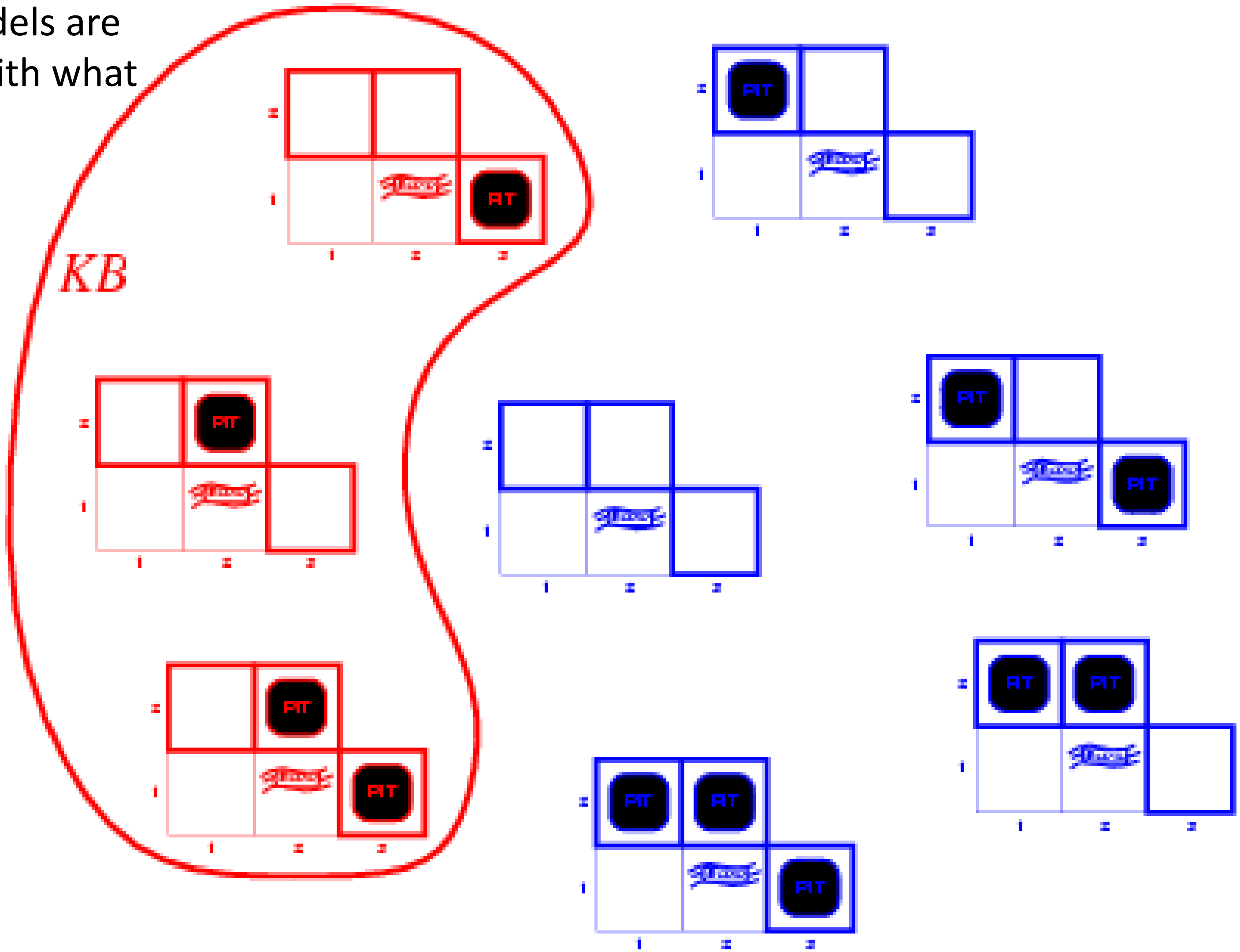
~B12 => ~P11

~B11 => ~P21

~B22 => ~P21

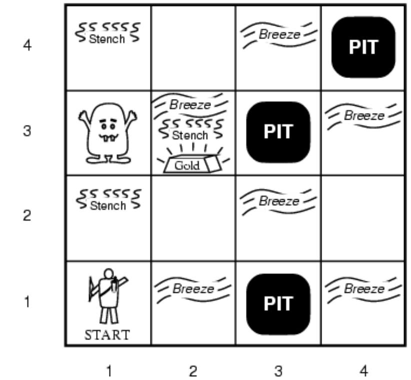
…

Only three of the possible models are consistent with what we know



KB = wumpus-world rules + observations

Wumpus World Rules (2)



- **Cell safe if it has neither a pit nor wumpus**

  $OK11 \Rightarrow \sim P11 \land \sim W11$

  $OK12 \Rightarrow \sim P12 \land \sim W12$ ...

OK11: (1,1) is safe
W11: Wumpus in (1,1)

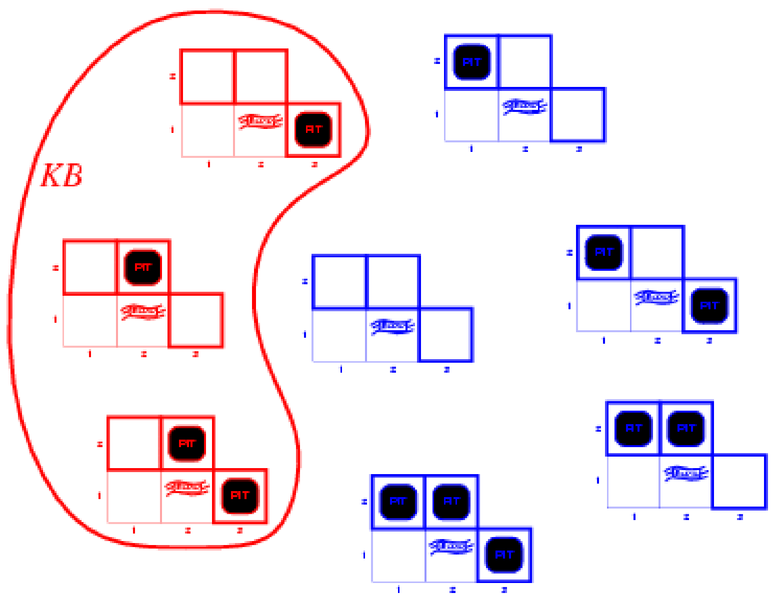
- From which we can derive

  $P11 \lor W11 \Rightarrow \sim OK11$

  $P11 \Rightarrow \sim OK11$

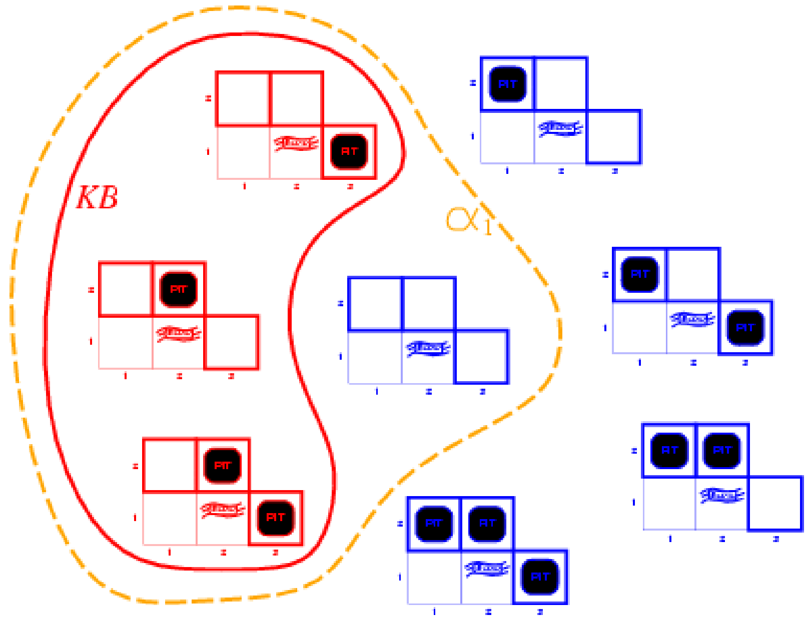
  $W11 \Rightarrow \sim OK11$ ...

Wumpus models



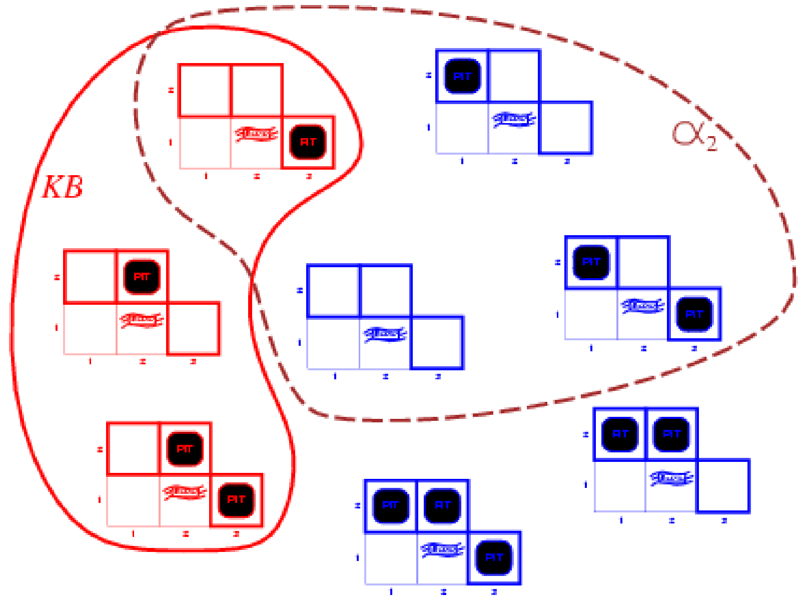
- *KB* = wumpus-world rules + observations

Wumpus models



- *KB* = wumpus-world rules + observations
- α_1 = “[1,2] is safe”
- *Since all models include* α_1
- *KB* $\models \alpha_1$, proved by **model checking**

Is (2,2) Safe?



- KB = wumpus-world rules + observations
- α_2 = "[2,2] is safe"
- Since some models don't include α_2, $KB \models \alpha_2$
- We cannot prove OK22; it might be true or false

Inference, Soundness, Completeness

- *KB* $\vdash_i$ α = sentence α can be derived from *KB* by procedure *i*
- **Soundness:** *i* is sound if whenever *KB* $\vdash_i$ α, it is also true that *KB* $\models$ α
- **Completeness:** *i* is complete if whenever *KB* $\models$ α, it is also true that *KB* $\vdash_i$ α

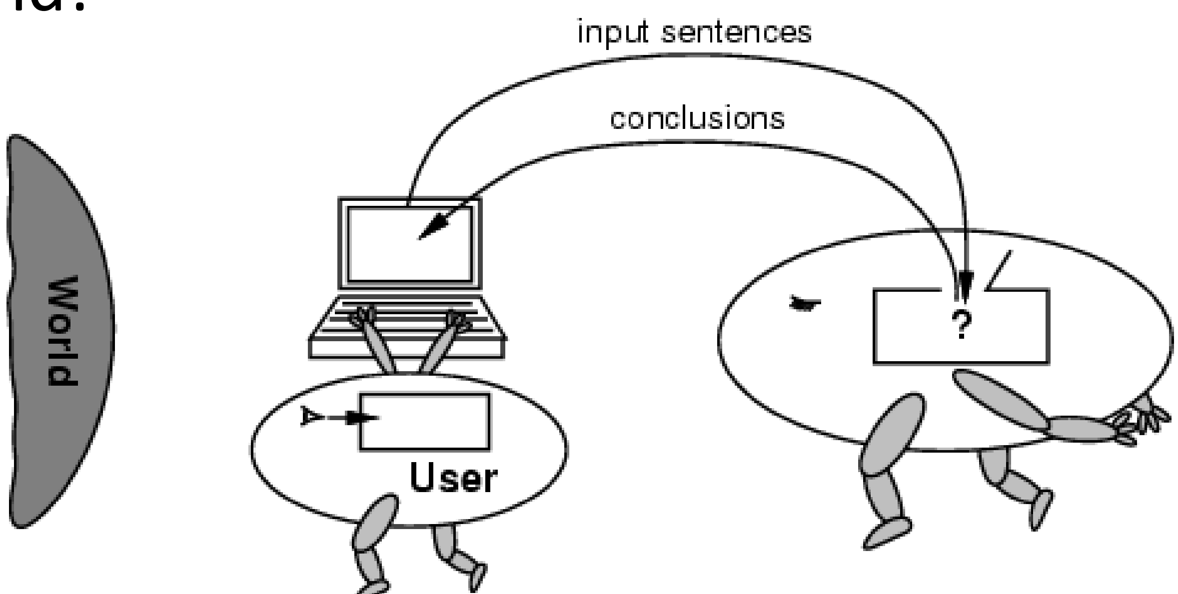
- Preview: **first-order logic** is expressive enough to say almost anything of interest and has a **sound** and **complete** inference procedure

Soundness and completeness

- A *sound* inference method derives only entailed sentences
- Analogous to the property of *completeness* in search, a *complete* inference method can derive any sentence that is entailed

No independent access to the world

- Reasoning agents often gets knowledge about facts of the world as a sequence of logical sentences and must draw conclusions only from them w/o independent access to world
- Thus, it is very important that the agents' reasoning is sound!



Summary

- Intelligent agents need knowledge about world for good decisions
- Agent’s knowledge stored in a knowledge base (KB) as **sentences** in a knowledge representation (KR) language
- Knowledge-based agents needs a **KB** & **inference mechanism**. They store sentences in KB, infer new sentences & use them to **deduce** which actions to take
- A **representation language** defined by its syntax & semantics, which specify structure of sentences & how they relate to facts of the world
- **Interpretation** of a sentence is fact to which it refers. If fact is part of the actual world, then the sentence is true

Propositional logic syntax

- Users specify
  - Set of propositional symbols (e.g., P, Q) whose values can be **True** or **False**
  - What each *means*, e.g.: P: "*It's hot*", Q: "*It's humid*"
- A sentence (well formed formula) is defined as:
  - Any symbol is a sentence
  - If S is a sentence, then $\neg$**S** is a sentence
  - If S is a sentence, then **(S)** is a sentence
  - If S and T are sentences, then so are **(S $\vee$ T), (S $\wedge$ T), (S $\rightarrow$ T)**, and **(S $\leftrightarrow$ T)**
  - A finite number of applications of the rules

Examples of PL sentences

- **Q**
  "It's humid"
- **$Q \rightarrow P$**
  "If it's humid, then it's hot"
- **$(P \wedge Q) \rightarrow R$**
  "If it's hot and it's humid, then it's raining"
- We're free to choose better symbols, e.g.:
  Hot for "It's hot"
  Humid for "It's humid"
  Raining for "It's raining"

Some terms

- Given the truth values of all symbols in a sentence, it can be ***evaluated*** to determine its **truth value** (True or False)
- We consider a **Knowledge Base** (KB) to be a set of sentences that are all True
- A **model** for a KB is a **possible world** – an assignment of truth values to propositional symbols that makes each KB sentence true

More terms

- A **valid sentence** or **tautology:** one that's **True** under all interpretations, no matter what the world is actually like or what the semantics is. Example: "It's raining or it's not raining" $(P \vee \neg P)$

- An **inconsistent sentence** or **contradiction:** a sentence that's **False** under all interpretations. The world is never like what it describes, as in "It's raining and it's not raining." $(P \wedge \neg P)$

Truth tables

Used to define meaning of logical connectives

Truth tables for the five logical connectives

P	$\neg P$
True	False
True	False
False	True
False	True

"not"

Truth tables

Used to define meaning of logical connectives

Truth tables for the five logical connectives

P	Q	$\neg P$	$P \wedge Q$
True	True	False	True
True	False	False	False
False	False	True	False
False	True	True	False

"and"

Truth tables

Used to define meaning of logical connectives

Truth tables for the five logical connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$
True	True	False	True	True
True	False	False	False	True
False	False	True	False	False
False	True	True	False	True

(inclusive)
"or"

Truth tables

Used to define meaning of logical connectives

Truth tables for the five logical connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \to Q$
True	True	False	True	True	True
True	False	False	False	True	False
False	False	True	False	False	True
False	True	True	False	True	True

implication
of q from p

Truth tables

Used to define meaning of logical connectives

Truth tables for the five logical connectives

P	Q	$\neg P$	$P \land Q$	$P \lor Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
True	True	False	True	True	True	True
True	False	False	False	True	False	False
False	False	True	False	False	True	True
False	True	True	False	True	True	False

Bidirectional implication (aka, equivalence)
$(P \rightarrow Q) \land (Q \rightarrow P)$

Distribution of Negation

$\neg$ $\neg$

P	Q	$\neg P$	$P \lor Q$	$\neg P \land \neg Q$	$P \land Q$	$\neg P \lor \neg Q$
True	True	False	True	False	True	False
True	False	False	True	False	False	True
False	False	True	False	True	False	True
False	True	True	True	False	False	True

Examples

- What's the truth table of

$$\neg P \lor Q$$

P	Q	$\neg P$	$P \lor Q$
True	True	False	True
True	False	False	True
False	False	True	False
False	True	True	True

Examples

- What's the truth table of

$$\neg \boldsymbol{P} \vee \boldsymbol{Q}$$

$\boldsymbol{P}$	$\boldsymbol{Q}$	$\neg\boldsymbol{P}$	$\boldsymbol{P} \vee \boldsymbol{Q}$	$\neg\boldsymbol{P} \vee \boldsymbol{Q}$
True	True	False	True	True
True	False	False	True	False
False	False	True	False	True
False	True	True	True	True

Examples

- What's the truth table of

$$\neg \boldsymbol{P} \lor \boldsymbol{Q}$$

$\boldsymbol{P}$	$\boldsymbol{Q}$	$\neg\boldsymbol{P}$	$\boldsymbol{P} \lor \boldsymbol{Q}$	$\neg\boldsymbol{P} \lor \boldsymbol{Q}$	$\boldsymbol{P} \to \boldsymbol{Q}$
True	True	False	True	True	True
True	False	False	True	False	False
False	False	True	False	True	True
False	True	True	True	True	True

- What's the truth table of

$$(\boldsymbol{P} \ \lor \boldsymbol{Q}) \land \neg\ \boldsymbol{Q}) \to \boldsymbol{P}?$$

(Work it out on your own)

The implies connective: $P \rightarrow Q$

$\rightarrow$ is a *logical connective*

- $P \rightarrow Q$ is a **logical sentence** and has a truth value, i.e., is either **True** or **False**
- If the sentence is in a KB, it can be used by a rule (*Modus Ponens*) to infer that Q is True if P is True in the KB
- Given a KB where P=True and Q=True, we can derive/infer/prove that $P \rightarrow Q$ is True
- Note: $P \rightarrow Q$ is equivalent to $\sim P \vee Q$

$P \to Q$

When is *$P \to Q$* true? Check all that apply

- ❑ P=Q=true
- ❑ P=Q=false
- ❑ P=true, Q=false
- ❑ P=false, Q=true

$P \rightarrow Q$

When is *P→Q* true? Check all that apply

- ☑ P=Q=true
- ☑ P=Q=false
- ☐ P=true, Q=false
- ☑ P=false, Q=true

- We can get this from the truth table for →
- Note: in FOL it's much harder to prove that a conditional true, e.g., prime(x) → odd(x)

Knowledge Bases (KBs)

- **Literal**: a Boolean variable
- **Clause**: a disjunction of literals
  - If $l_1, \ldots, l_N$ are literals, then $l_1 \vee \cdots \vee \ l_N$ is a clause
  - Clauses don't need to contain *all* literals
- If a literal only appears with one polarity in any clauses it appears in (either as l_i or $\neg l_i$, but not both), then it's a **pure literal**

Knowledge Bases (KBs)

- A conjunction of **definite clauses**
- **Definite clause** (aka Strict Horn clause): a *body* implies a *head*
  - Form: $a_1 \wedge a_2 \wedge \cdots \wedge a_M \to h$
  - Body: $a_1 \wedge a_2 \wedge \cdots \wedge a_M$
  - Head: h
- If the body is empty, then the head is a *fact*

Q: Is
$A \vee B \vee \neg C$
a definite clause?

Knowledge Bases (KBs)

- A conjunction of **definite clauses**
- **Definite clause** (aka Strict Horn Clause): a *body* implies a *head*
  - Form: $a_1 \land a_2 \land \cdots \land a_M \rightarrow h$
  - Body: $a_1 \land a_2 \land \cdots \land a_M$
  - Head: h
- If the body is empty, then the head is a *fact*

Q: Is
$A \lor B \lor \neg C$
a definite clause?

A: No. Can you turn it into one?

Models for a KB

- KB: [P∨Q, P→R, Q→R]
- What are the sentences?

  s1: P∨Q
  s2: P→R
  s3: Q→R
- What are the propositional variables?

  P, Q, R
- What are the candidate models?
  1) Consider all **eight** possible assignments of T|F to P, Q, R
  2) Check if each sentence is consistent with the model

P	Q	R	s1	s2	s3
F	F	F	x	✓	✓
F	F	T	x	✓	✓
F	T	F	✓	✓	x
F	T	T	✓	✓	✓
T	F	F	✓	x	✓
T	F	T	✓	✓	✓
T	T	F	✓	x	x
T	T	T	✓	✓	✓

Here **x** means the model makes the sentence False and ✓means it doesn't make it False

Models for a KB

- KB: [P∨Q, P→R, Q→R]
- What are the sentences?

  s1: P∨Q
  s2: P→R
  s3: Q→R
- What are the propositional variables?

  P, Q, R
- What are the candidate models?

  1) Consider all possible assignments of T|F to P, Q, R
  2) Check truth tables for consistency, eliminating any row that does not make every KB sentence true

P	Q	R	s1	s2	s3
~~F~~	~~F~~	~~F~~	~~x~~	~~✓~~	~~✓~~
~~F~~	~~F~~	~~T~~	~~x~~	~~✓~~	~~✓~~
~~F~~	~~T~~	~~F~~	~~✓~~	~~✓~~	~~x~~
F	T	T	✓	✓	✓
~~T~~	~~F~~	~~F~~	~~✓~~	~~x~~	~~✓~~
T	F	T	✓	✓	✓
~~T~~	~~T~~	~~F~~	~~✓~~	~~x~~	~~x~~
T	T	T	✓	✓	✓

- Only 3 models are consistent with KB
- R true in all of them
- Therefore R is true and can be added to the KB

A simple example

The KB

P

Q ∨ ¬ R

The KB has 2 sentences.

The KB has 3 variables.

Models for the KB

P	Q	R	KB
T	T	F	T
T	T	T	T
T	F	F	T
T	F	T	F
F	T	F	F
F	T	T	F
F	F	T	F
F	F	F	F

The KB has 3 models. Each model has a value for every variable in the KB such every sentence evaluates to true.

Another simple example

The KB

$P \land Q$

$R \land \neg P$

The KB has 2 sentences.

The KB has 3 variables.

Models for the KB

P	Q	R

The KB has no models. There is no assignment of True or False to every variable that makes every sentence in the KB true

Finite CSP to Logic

- Let X be a variable with domain $\{a_1, a_2, \ldots, a_D\}$

Finite CSP to Logic

- Let X be a variable with domain $\{a_1, a_2, \ldots, a_D\}$
- Replace X with D different **indicator variables**
  - X_1 is true iff $X = \mathrm{a}_1$
  - X_2 is true iff $X = \mathrm{a}_2$
  - …
  - X_D is true iff $X = \mathrm{a}_D$
- Add pairwise constraints. For $i < j$:

Finite CSP to Logic

- Let X be a variable with domain $\{a_1, a_2, \dots, a_D\}$
- Replace X with D different **indicator variables**
  - X_1 is true iff $X = \mathrm{a}_1$
  - X_2 is true iff $X = \mathrm{a}_2$
  - …
  - X_D is true iff $X = \mathrm{a}_D$
- Add pairwise constraints. For $i < j$:
  - $\neg X_i \vee \neg X_j$
- At least one must be "on"

Finite CSP to Logic

- Let X be a variable with domain $\{a_1, a_2, \dots, a_D\}$
- Replace X with D different **indicator variables**
  - X_1 is true iff $X = \mathrm{a}_1$
  - X_2 is true iff $X = \mathrm{a}_2$
  - …
  - X_D is true iff $X = \mathrm{a}_D$
- Add pairwise constraints. For $i < j$:
  - $\neg X_i \vee \neg X_j$
- At least one must be "on"
  - $X_1 \vee X_2 \vee \cdots \vee X_D$

Reasoning With Propositional Logic

- There are many ways to approach reasoning with propositional logic
- We'll look at one, resolution refutation, that can be extended to first order logic
- Later, we will look other approaches that are special to propositional logic

Reasoning / Inference

- **Logical inference** creates new sentences that logically follow from a set of sentences (KB)
- It can also detect if a KB is inconsistent, i.e., has sentences that entail a **contradiction**
- An inference rule is **sound** if every sentence X it produces from a KB logically follows from the KB
  - i.e., inference rule creates no contradictions
- An inference rule is **complete** if it can produce every expression that logically follows from (is entailed by) the KB
  - Note analogy to complete search algorithms

Sound rules of inference

Examples of sound rules of inference

Each can be shown to be sound using a truth table

RULE	PREMISE	CONCLUSION
Modus Ponens	$A, A \rightarrow B$	B
And Introduction	A, B	$A \wedge B$
And Elimination	$A \wedge B$	A
Double Negation	$\neg\neg A$	A
Unit Resolution	$A \vee B, \neg B$	A
Resolution	$A \vee B, \neg B \vee C$	$A \vee C$

Resolution

- **Resolution** is a valid inference rule producing a new clause implied by two clauses containing *complementary literals*

  Literal: atomic symbol or its negation, i.e., P, ~P

- Amazingly, this is the **only** interference rule needed to build a sound & complete theorem prover
  - Based on proof by contradiction, usually called resolution refutation
- The resolution rule was discovered by Alan Robinson (CS, U. of Syracuse) in the mid 1960s

Resolution

- A KB is a set of sentences all of which are true, i.e., a conjunction of sentences
- To use resolution, put KB into *conjunctive normal form* (CNF)
  - Each sentence is a disjunction of one or more literals (positive or negative atoms)
- Every KB can be put into CNF, it's just a matter of rewriting its sentences using standard tautologies, e.g.: $P \rightarrow Q \equiv \sim P \vee Q$

Resolution Example

Tautologies

$(A \rightarrow B) \leftrightarrow (\sim A \vee B)$

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

- KB: $[P \rightarrow Q\ ,\ Q \rightarrow R \wedge S]$
- KB: $[P \rightarrow Q\ ,\ Q \rightarrow R,\ Q \rightarrow S\ ]$
- KB in CNF: $[\sim P \vee Q\ ,\ \sim Q \vee R\ ,\ \sim Q \vee S]$
- Resolve KB[0] and KB[1] producing:

  $\sim P \vee R$ *(i.e., $P \rightarrow R$)*
- Resolve KB[0] and KB[2] producing:

  $\sim P \vee S$ *(i.e., $P \rightarrow S$)*
- New KB: $[\sim P \vee Q\ ,\ \sim Q \vee R,\ \sim Q \vee S,\ \sim P \vee R,\ \sim P \vee S]$

Proving it’s raining with rules

- A **proof** is a sequence of sentences, where each is a premise (i.e., a given) or is derived from earlier sentences in the proof by an inference rule
- Last sentence is the **theorem** (also called goal or query) that we want to prove
- The *weather problem* using traditional reasoning

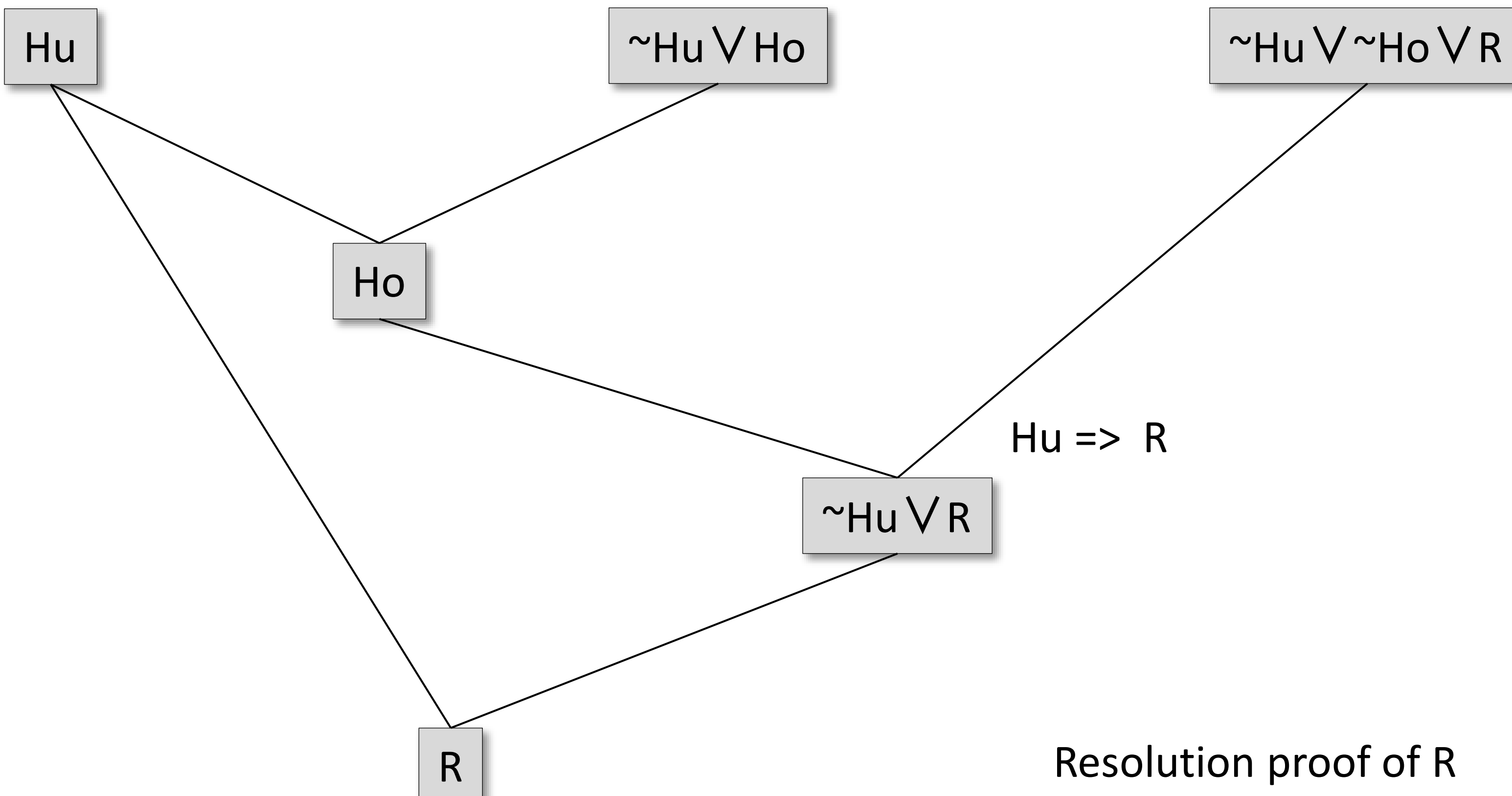
1 Hu	premise	“It's humid”
2 $Hu \rightarrow Ho$	premise	“If it's humid, it's hot”
3 Ho	modus ponens(1,2)	“It's hot”
4 $(Ho \wedge Hu) \rightarrow R$	premise	“If it's hot & humid, it's raining”
5 $Ho \wedge Hu$	and introduction(1,3)	“It's hot and humid”
6 R	modus ponens(4,5)	“It's raining”

Proving it's raining with resolution

Hu

Hu => Ho
~Hu ∨ Ho

Hu ∧ Ho => R
~(Hu ∧ Ho) ∨ R
~Hu ∨ ~Ho ∨ R



Resolution proof of R

A simple proof procedure

This procedure generates new sentences in a KB

1. Convert all sentences in the KB to CNF[1]
2. Find all pairs of sentences in KB with complementary literals[2] that have not yet been resolved
3. If there are no pairs stop else resolve each pair, adding the result to the KB and go to 2

- Is it sound?
- Is it complete?
- Will it always terminate?

1: a KB in conjunctive normal form is a set of disjunctive sentences

2: a literal is a variable or its negation

Resolution refutation

1. Add negation of goal to the KB
2. Convert all sentences in KB to CNF
3. Find all pairs of sentences in KB with complementary literals that have not yet been resolved
4. If there are no pairs stop else resolve each pair, adding the result to the KB and go to 2

- If we derived an empty clause (i.e., a contradiction) then the conclusion follows from the KB
- If we did not, the conclusion cannot be proved from the KB

Propositional logic: pro and con

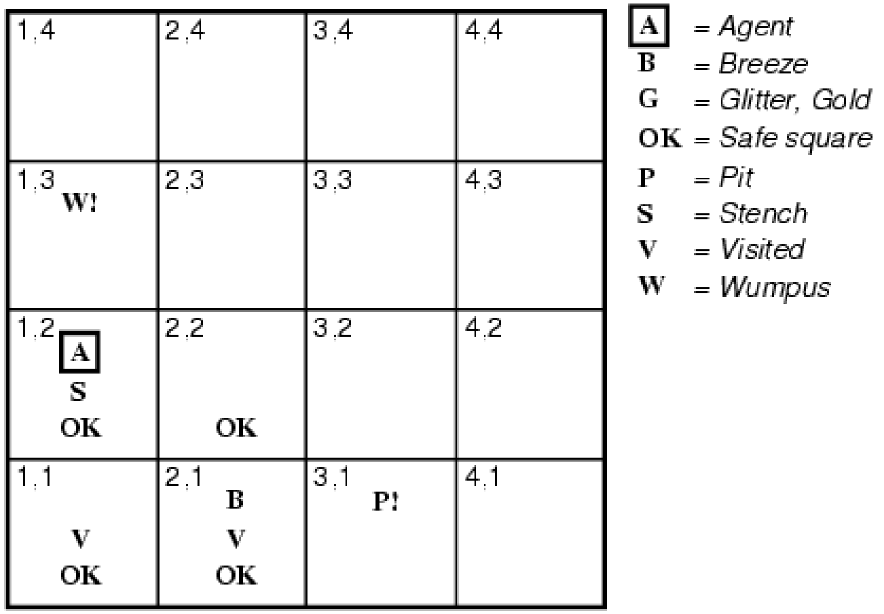
- **Advantages**
  - Simple KR language good for many problems
  - Lays foundation for higher logics (e.g., FOL)
  - Reasoning is decidable, though NP complete; efficient techniques exist for many problems
- **Disadvantages**
  - Not expressive enough for most problems
  - Even when it is, it can very “un-concise”

PL is a weak KR language

- Hard to identify *individuals* (e.g., Mary, 3)
- Can't directly represent properties of individuals or relations between them (e.g., "Bill age 24")
- Generalizations, patterns, regularities hard to represent (e.g., "all triangles have 3 sides")
- First-Order Logic (FOL) represents this information via **relations**, **variables** & **quantifier**s, e.g.,
  - *John loves Mary: loves(John, Mary)*
  - *Every elephant is gray:* $\forall$ x (elephant(x) → gray(x))
  - *There is a black swan:* $\exists$ x (swan(X) ^ black(X))

Hunt the Wumpus domain

- Some atomic propositions:
  A12 = agent is in call (1,2)
  S12 = There's a stench in cell (1,2)
  B34 = There's a breeze in cell (3,4)
  W22 = Wumpus is in cell (2,2)
  V11 = We've visited cell (1,1)
  OK11 = cell (1,1) is safe
  …



- Some rules:
  $\neg S22 \rightarrow \neg W12 \wedge \neg W23 \wedge \neg W32 \wedge \neg W21$
  $S22 \rightarrow W12 \vee W23 \vee W32 \vee W21$
  $B22 \rightarrow P12 \vee P23 \vee P32 \vee P21$
  $W22 \rightarrow S12 \wedge S23 \wedge S32 \wedge W21$
  $W22 \rightarrow \neg W11 \wedge \neg W21 \wedge \ldots \neg W44$
  $A22 \rightarrow V22$
  $A22 \rightarrow \neg W11 \wedge \neg W21 \wedge \ldots \neg W44$
  $V22 \rightarrow OK22$

If there's no stench in cell 2,2 then the Wumpus isn't in cell 21, 23 32 or 21

Hunt the Wumpus domain

- Eight symbols for each cell, i.e.: A11, B11, G11, OK11, P11, S11, V11, W11
- Lack of variables requires giving similar rules for each cell!
- Ten rules (I think) for each

$A11 \rightarrow \ldots$

$V11 \rightarrow \ldots$

$P11 \rightarrow \ldots$

$\neg P11 \rightarrow \ldots$

$W11 \rightarrow \ldots$

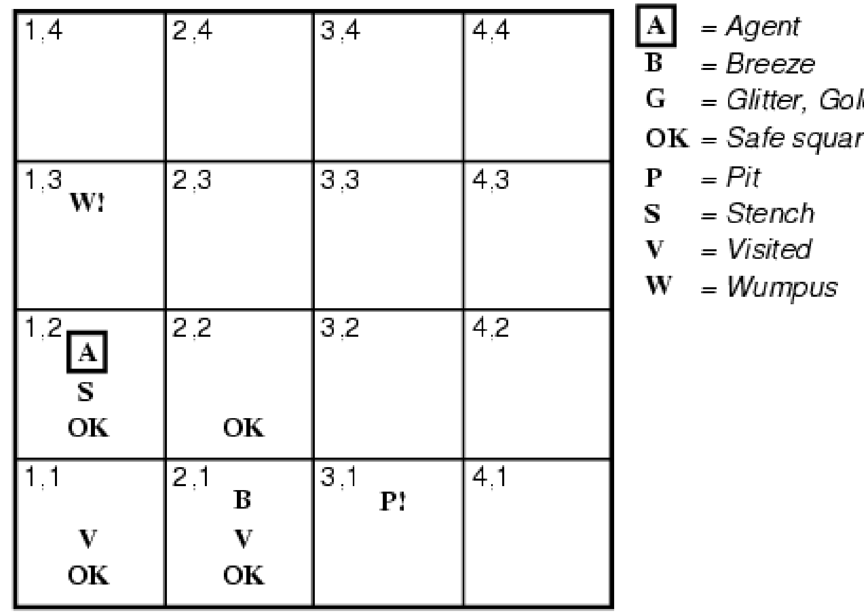
$\neg W11 \rightarrow \ldots$

$S11 \rightarrow \ldots$

$\neg S11 \rightarrow \ldots$

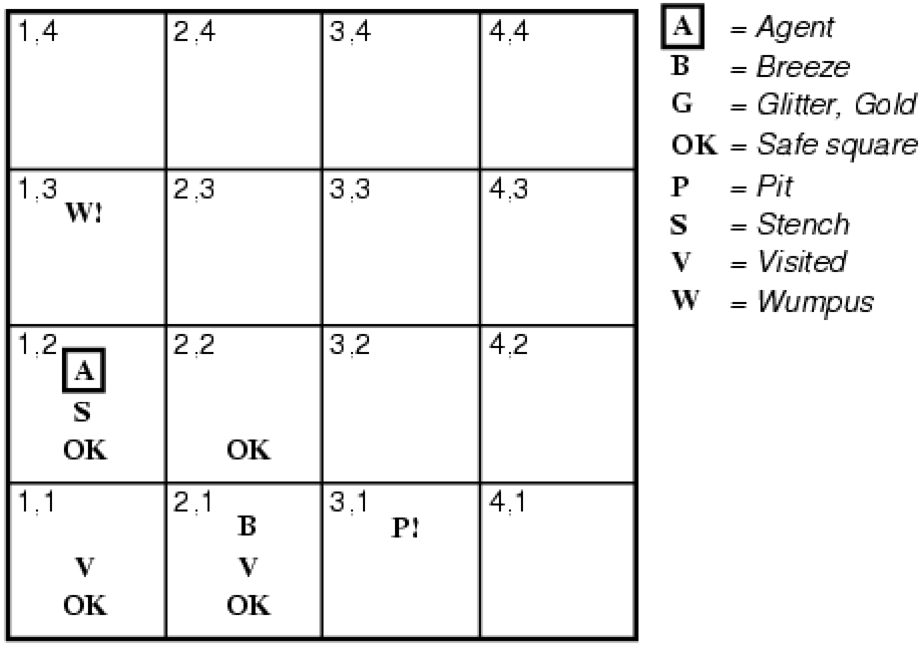
$B11 \rightarrow \ldots$

$\neg B11 \rightarrow \ldots$



- 8 symbols for 16 cells => 128 symbols
- 2^{128} possible models ☹
- Must do better than brute force

After third move



- We can prove that the Wumpus is in (1,3) using these four rules
- See R&N section 7.5

(R1) $\neg S11 \rightarrow \neg W11 \wedge \neg W12 \wedge \neg W21$

(R2) $\neg S21 \rightarrow \neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$

(R3) $\neg S12 \rightarrow \neg W11 \wedge \neg W12 \wedge \neg W22 \wedge \neg W13$

(R4) $S12 \rightarrow W13 \vee W12 \vee W22 \vee W11$

Proving W13: Wumpus is in cell 1,3

Apply **MP** with $\neg$S11 and R1:

$\neg$ W11 $\wedge$ $\neg$ W12 $\wedge$ $\neg$ W21

Apply **AE**, yielding three sentences:

$\neg$ W11, $\neg$ W12, $\neg$ W21

Apply **MP** to ~S21 and R2, then apply **AE**:

$\neg$ W22, $\neg$ W21, $\neg$ W31

Apply **MP** to S12 and R4 to obtain:

W13 $\vee$ W12 $\vee$ W22 $\vee$ W11

Apply **UR** on (W13 $\vee$ W12 $\vee$ W22 $\vee$ W11) and $\neg$W11:

W13 $\vee$ W12 $\vee$ W22

Apply **UR** with (W13 $\vee$ W12 $\vee$ W22) and $\neg$W22:

W13 $\vee$ W12

Apply **UR** with (W13 $\vee$ W12) and $\neg$W12:

W13

QED

(R1) $\neg$S11 $\rightarrow$ $\neg$W11 $\wedge$ $\neg$ W12 $\wedge$ $\neg$ W21

(R2) $\neg$ S21 $\rightarrow$ $\neg$W11 $\wedge$ $\neg$ W21 $\wedge$ $\neg$ W22 $\wedge$ $\neg$ W31

(R3) $\neg$ S12 $\rightarrow$ $\neg$W11 $\wedge$ $\neg$ W12 $\wedge$ $\neg$ W22 $\wedge$ $\neg$ W13

(R4) S12 $\rightarrow$ W13 $\vee$ W12 $\vee$ W22 $\vee$ W11

Rule Abbreviation
MP: modes ponens
AE: and elimination
R: unit resolution

Propositional Wumpus problems

- Lack of variables prevents general rules, e.g.:
  - $\forall$ x, y V(x,y) → OK(x,y)
  - $\forall$ x, y S(x,y) → W(x-1,y) $\vee$ W(x+1,y) …
- Change of KB over time difficult to represent
  - In classical logic; a fact is true or false for all time
  - A standard technique is to index dynamic facts with the time when they're true
    - A(1, 1, 0) *# agent was in cell 1,1 at time 0*
    - A(2, 1, 1) *# agent was in cell 2,1 at time 1*
  - Thus we have a separate KB for every time point

Monotonicity

- **Monotonic logic**: adding knowledge does not make previously provable items non-provable.
  - Definite clauses are monotonic
- **Non-monotonic logic**: previously-made conclusions (inferences) can be made invalid by the addition of new knowledge
- **Default rule**: knowledge that should be used, unless overridden

*duction

- **Abduction**: Given an observation, make assumptions that may explain it
- **Deduction**:
- **Induction**:

*duction

- **Abduction**: Given an observation, make assumptions that may explain it
- **Deduction**: Determine what must follow from a base of knowledge
- **Induction**:

*duction

- **Abduction**: Given an observation, make assumptions that may explain it
- **Deduction**: Determine what must follow from a base of knowledge
- **Induction**: infer generalities from specific examples

*duction

- **Abduction**: Given an observation, make assumptions that may explain it
- **Deduction**: Determine what must follow from a base of knowledge
- **Induction**: infer generalities from specific examples

Q: What could be some use cases for all of these?

Propositional logic summary

- **Inference**: process of deriving new sentences from old
  - **Sound** inference derives true conclusions given true premises
  - **Complete** inference derives all true conclusions from premises
- Different logics make different **commitments** about what the world is made of and the kind of beliefs we can have
- **Propositional logic** commits only to existence of facts that may or may not be the case in the world being represented
  - Simple syntax & semantics illustrates the process of inference
  - It can become impractical, even for very small worlds

First Order Logic Overview

- First Order logic (FOL) is a powerful knowledge representation (KR) system
- It’s used in AI systems in various ways, e.g.
  - To directly represent and reason about concepts and objects
  - To formally specify the meaning of other KR systems
  - To provide features that are useful in neural network deep learning systems

First-order logic

- First-order logic (FOL) models the world in terms of
  - **Objects,** which are things with individual identities
  - **Properties** of objects that distinguish them from others
  - **Relations** that hold among sets of objects
  - **Functions,** a subset of relations where there is only one "value" for any given "input"
- Examples:
  - Objects: students, lectures, companies, cars ...
  - Relations: brother-of, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...
  - Properties: blue, oval, even, large, ...
  - Functions: father-of, best-friend, more-than ...

User provides

- **Constant symbols** representing individuals in world
  - BarackObama, Green, John, 3, “John Smith”
- **Predicate symbols,** map individuals to truth values
  - greater(5,3)
  - green(Grass)
  - color(Grass, Green)
  - hasBrother(John, Robert)
- **Function symbols,** map individuals to individuals
  - father_of(SashaObama) = BarackObama
  - color_of(Sky) = Blue

What do these mean?

- User should also indicate what these mean in a way that humans will understand
  - i.e., map to their own internal representations
- May be done via a combination of
  - Choosing good names for a formal terms, e.g. calling a concept HumanBeing instead of Q5
  - Comments in the definition `#human being`
  - Descriptions and examples in documentation
  - Reference to other representations , e.g., sameAs /m/0dgw95 in Freebase and Person in schema.org
  - Giving examples (Donald Trump) and non-examples (Luke Skywalker)

FOL Provides

- **Variable symbols**
  - E.g., x, y, foo
- **Connectives**
  - Same as propositional logic: not ($\neg$), and ($\wedge$), or ($\vee$), implies ($\rightarrow$), iff ($\leftrightarrow$)
- **Quantifiers**
  - Universal $\forall$**x** or **(Ax)**
  - Existential $\exists$**x** or **(Ex)**

Sentences: built from terms and atoms

- **term** (denoting an individual): constant or variable symbol, or n-place function of n terms, e.g.:
  - Constants: john, umbc
  - Variables: X, Y, Z
  - Functions: mother_of(john), phone(mother(x))
- **Ground terms** have no variables in them
  - **Ground:** john, father_of(father_of(john))
  - **Not Ground:** father_of(X)
- Syntax may vary: e.g., maybe variables must start with a "?" of a capital letter

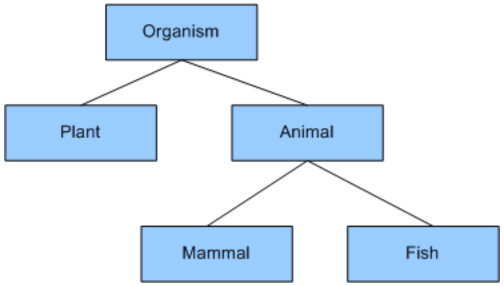
Sentences: built from terms and atoms

- **atomic sentences** (which are either true or false) are n-place predicates of n terms, e.g.:
  - green(kermit)
  - between(philadelphia, baltimore, dc)
  - loves(X, mother(X))
- **complex sentences** formed from atomic ones connected by the standard logical connectives with quantifiers if there are variables, e.g.:
  - loves(mary, john) $\vee$ loves(mary, bill)
  - $\forall$x loves(mary, x)

What do atomic sentences mean?

- Unary predicates typically encode a **type**
  - muppet(Kermit): kermit is a kind of muppet
  - green(kermit): kermit is a kind of green thing
  - integer(X): x is a kind of integer
- Non-unary predicates typically encode relations or properties
  - Loves(john, mary)
  - Greater_than(2, 1)
  - Between(newYork, philadelphia, baltimore)
  - hasName(john, “John Smith”)

Ontology



- Designing a logic representation is like designing a model in an object-oriented language
- **Ontology:** a "formal naming and definition of the types, properties and relations of entities for a domain of discourse"
- E.g.: schema.org ontology used to put semantic data on Web pages to help search engines
  - Here's the semantic markup Google sees on our 471 class site

Sentences: built from terms and atoms

- **quantified sentences** adds quantifiers $\forall$ and $\exists$
  - $\forall$x loves(x, mother(x))
  - $\exists$x number(x) $\wedge$ greater(x, 100), prime(x)
- **well-formed formula (wff):** a sentence with no *free* variables or where all variables are *bound* by a universal or existential *quantifier*
  - In **($\forall$x)P(x, y)** x is bound & y is free so it's not a wff

Quantifiers: ∀ and ∃

- **Universal quantification**
  - $(\forall x)P(X)$ means P holds for **all** values of X in the domain associated with variable[1]
  - E.g., $(\forall X)$ dolphin(X) $\rightarrow$ mammal(X)
- **Existential quantification**
  - $(\exists x)P(X)$ means P holds for **some** value of X in domain associated with variable
  - E.g., $(\exists \mathbf{X})$ mammal(X) $\wedge$ lays_eggs(X)
  - This lets us make statements about an object without identifying it

[1] a variable's domain is often not explicitly stated and is assumed by the context

Universal Quantifier: $\forall$

- Universal quantifiers typically used with *implies* to form *rules*:

  Logic: $(\forall X)\ student(X) \rightarrow smart(X)$

  Means: All students are smart

- Universal quantification *rarely* used without implies:

  Logic: $(\forall X)\ student(X) \wedge smart(X)$

  Means: Everything is a student and is smart

Existential Quantifier: $\exists$

- Existential quantifiers usually used with **and** to specify a list of properties about an individual

  Logic: $(\exists X)\ student(X) \wedge smart(X)$

  Meaning: There is a student who is smart

- Common mistake: represent this in FOL as:

  Logic: $(\exists X)\ student(X) \rightarrow smart(X)$

  Meaning: ?

Existential Quantifier: ∃

- Existential quantifiers usually used with **and** to specify a list of properties about an individual

  Logic: $(\exists X)\ student(X) \wedge smart(X)$

  Meaning: There is a student who is smart

- Common mistake: represent this in FOL as:

  Logic: $(\exists X)\ student(X) \rightarrow smart(X)$

  $P \rightarrow Q = {\sim}P \vee Q$

  $\exists X\ student(X) \rightarrow smart(X) = \exists X\ {\sim}student(X) \vee smart(X)$

  Meaning: There's something that is either not a student or is smart

Quantifier Scope

- FOL sentences have structure, like programs
- In particular, variables in a sentence have a **scope**
- Suppose we want to say “everyone who is alive loves someone”

  $(\forall X)\ alive(X) \rightarrow (\exists\ Y)\ loves(X, Y)$
- Here’s how we scope the variables

$$(\forall X)\ alive(X) \rightarrow (\exists Y)\ loves(X, Y)$$

Scope of x
Scope of y

Quantifier Scope

- **Switching order of universal quantifiers *does not* change the meaning**
  - $(\forall X)(\forall Y)P(X,Y) \leftrightarrow (\forall Y)(\forall X)\ P(X,Y)$
  - Dogs hate cats (i.e., all dogs hate all cats)
- **You can switch order of existential quantifiers**
  - $(\exists X)(\exists Y)P(X,Y) \leftrightarrow (\exists Y)(\exists X)\ P(X,Y)$
  - A cat killed a dog
- **Switching order of universal and existential quantifiers *does* change meaning:**
  - Everyone likes someone: $(\forall X)(\exists Y)$ likes(X,Y)
  - Someone is liked by everyone: $(\exists Y)(\forall X)$ likes(X,Y)

Procedural example 1
(Illustrative only!)

```
def verify1():
    # Everyone likes someone: (∀x)(∃y) likes(x,y)
    for p1 in people():
        foundLike = False
        for p2 in people():
            if likes(p1, p2):
                foundLike = True
                break
        if not foundLike:
            print(p1, 'does not like anyone ☹')
            return False
    return True
```

Every person has at least one individual that they like.

Procedural example 2 (Illustrative only!)

```
def verify2():
    # Someone is liked by everyone: (∃y)( ∀x) likes(x,y)
    for p2 in people():
        foundHater = False
        for p1 in people():
            if not likes(p1, p2):
                foundHater = True
                break
        if not foundHater
            print(p2, 'is liked by everyone ☺')
            return True
    return False
```

There is a person who is liked by every person in the universe.

Connections between $\forall$ and $\exists$

- We can relate sentences involving $\forall$ and $\exists$ using extensions to **De Morgan's laws**:
  1. $(\forall x)\ P(x) \leftrightarrow \neg(\exists x)\ \neg\ P(x)$
  2. $\neg(\forall x)\ P(x) \leftrightarrow (\exists x)\ \neg P(x)$
  3. $(\exists\ x)\ P(x) \leftrightarrow \neg\ (\forall\ x)\ \neg P(x)$
  4. $\neg(\exists x)\ P(x) \leftrightarrow (\forall x)\ \neg P(x)$
- Examples
  1. All dogs don't like cats $\leftrightarrow$ No dog likes cats
  2. Not all dogs bark $\leftrightarrow$ There is a dog that doesn't bark
  3. All dogs sleep $\leftrightarrow$ There is no dog that doesn't sleep
  4. There is a dog that talks $\leftrightarrow$ Not all dogs can't talk

Notational differences

- **Different symbols** for *and, or, not, implies, ...*
  - $\forall \ \exists \ \Rightarrow \ \Leftrightarrow \ \wedge \ \vee \ \neg \ \bullet \ \supset$
  - p v (q ^ r)
  - p + (q * r)
- **Prolog**

  cat(X) :- furry(X), meows (X), has(X, claws)
- **Lispy notations**

```
(forall ?x (implies (and (furry ?x)
                         (meows ?x)
                         (has ?x claws))
                    (cat ?x)))
```

Translating English to FOL

Every gardener likes the sun

$\forall x\ gardener(x) \rightarrow likes(x,Sun)$

All purple mushrooms are poisonous

$\forall x\ (mushroom(x) \wedge purple(x)) \rightarrow poisonous(x)$

No purple mushroom is poisonous (two ways)

$\neg\exists x\ purple(x) \wedge mushroom(x) \wedge poisonous(x)$

$\forall x\ \ (mushroom(x) \wedge purple(x)) \rightarrow \neg poisonous(x)$

English to FOL: Counting

Use = predicate to identify different individuals

- **There are <u>at least</u> two purple mushrooms**
  $\exists x\ \exists y\ \text{mushroom}(x) \land \text{purple}(x) \land \text{mushroom}(y) \land \text{purple}(y) \land \boldsymbol{\neg(x=y)}$
- **There are <u>exactly</u> two purple mushrooms**
  $\exists x\ \exists y\ \text{mushroom}(x) \land \text{purple}(x) \land \text{mushroom}(y) \land \text{purple}(y) \land \boldsymbol{\neg(x=y)} \land \forall z\ (\text{mushroom}(z) \land \text{purple}(z)) \rightarrow \boldsymbol{((x=z) \lor (y=z))}$

Saying there are 802 different Pokemon will be hard!

Translating English to FOL

What do these mean?

- **You can fool some of the people all of the time**

- **You can fool all of the people some of the time**

Translating English to FOL

What do these mean?

Both English statements are ambiguous

- **You can fool some of the people all of the time**
  - There is a nonempty subset of people so easily fooled that you can fool that subset every time*
  - For any given time, there is a non-empty subset at that time that you can fool
- **You can fool all of the people some of the time**
  - There are one or more times when it's possible to fool everyone*
  - Everybody can be fooled at some point in time

* Most common interpretation, I think

Some terms we will need

- **person(x):** True iff x is a person
- **time(t):** True iff t is a point in time
- **canFool(x, t):** True iff x can be fooled at time t

Note: *iff* = *if and only if* = $\leftrightarrow$

Translating English to FOL

You can fool some of the people all of the time

There is a nonempty group of people so easily fooled that you can fool that group every time*

≡ There's (at least) one person you can fool every time

$\exists x\ \forall t\ \ person(x) \land time(t) \rightarrow canFool(x, t)$

For any given time, there is a non-empty group at that time that you can fool

≡ For every time, there's a person at that time that you can fool

$\forall t\ \exists x\ \ person(x) \land time(t) \rightarrow canFool(x, t)$

* Most common interpretation, I think

Translating English to FOL

You can fool all of the people some of the time

There's at least one time when you can fool everyone*

$\exists t\ \forall x\ time(t) \land person(x) \rightarrow canFool(x, t)$

Everybody can be fooled at some point in time

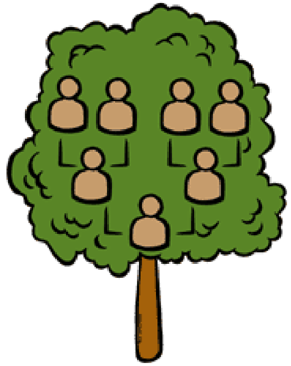
$\forall x\ \exists t\ person(x) \land time(t) \rightarrow canFool(x, t)$

* Most common interpretation, I think

Representation Design

- Many options for representing even a simple fact, e.g., something's color as red, green or blue, e.g.:
  - green(kermit)
  - color(kermit, green)
  - hasProperty(kermit, color, green)
- Choice can influence how easy it is to use
- Last option of representing properties & relations as triples used by modern knowledge graphs
  - Easy to ask: What color is Kermit? What are Kermit's properties?, What green things are there? Tell me everything you know, …

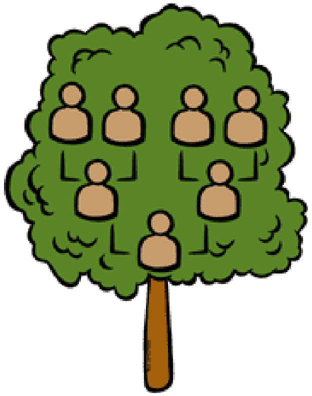
Simple genealogy KB in FOL



Design a knowledge base using FOL that

- Has facts of immediate family relations, e.g., spouses, parents, etc.
- Defines of more complex relations (ancestors, relatives)
- Detect conflicts, e.g., you are your own parent
- Infers relations, e.g., grandparent from parent
- Answers queries about relationships between people

How do we approach this?



- Design an initial ontology of types, e.g.
  - e.g., person, man, woman, male, female
- Extend ontology by defining relations, e.g.
  - spouse, has_child, has_parent
- Add general constraints to relations, e.g.
  - spouse(X,Y) => ~ X = Y
  - spouse(X,Y) => person(X), person(Y)
- Add FOL sentences for inference, e.g.
  - spouse(X,Y) ⇔ spouse(Y,X)
  - man(X) ⇔ person(X) ∧ male(X)

Example: A simple genealogy KB by FOL

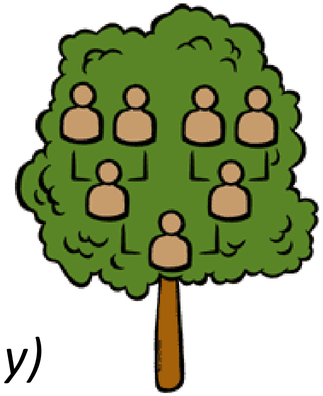
Predicates:

- parent(x, y), child(x, y), father(x, y), daughter(x, y), etc.
- spouse(x, y), husband(x, y), wife(x,y)
- ancestor(x, y), descendant(x, y)
- male(x), female(y)
- relative(x, y)

Facts:

- husband(Joe, Mary), son(Fred, Joe)
- spouse(John, Nancy), male(John), son(Mark, Nancy)
- father(Jack, Nancy), daughter(Linda, Jack)
- daughter(Liz, Linda)
- etc.

Example Axioms



$(\forall x,y)$ parent(x, y) $\leftrightarrow$ child (y, x)

$(\forall x,y)$ father(x, y) $\leftrightarrow$ parent(x, y) $\wedge$ male(x) *;similar for mother(x, y)*

$(\forall x,y)$ daughter(x, y) $\leftrightarrow$ child(x, y) $\wedge$ female(x) *;similar for son(x, y)*

$(\forall x,y)$ husband(x, y) $\leftrightarrow$ spouse(x, y) $\wedge$ male(x) *;similar for wife(x, y)*

$(\forall x,y)$ spouse(x, y) $\leftrightarrow$ spouse(y, x) *;spouse relation is symmetric*

$(\forall x,y)$ parent(x, y) $\rightarrow$ ancestor(x, y)

$(\forall x,y)(\exists z)$ parent(x, z) $\wedge$ ancestor(z, y) $\rightarrow$ ancestor(x, y)

$(\forall x,y)$ descendant(x, y) $\leftrightarrow$ ancestor(y, x)

$(\forall x,y)(\exists z)$ ancestor(z, x) $\wedge$ ancestor(z, y) $\rightarrow$ relative(x, y)

$(\forall x,y)$ spouse(x, y) $\rightarrow$ relative(x, y) ;related by marriage

$(\forall x,y)(\exists z)$ relative(z, x) $\wedge$ relative(z, y) $\rightarrow$ relative(x, y) *;transitive*

$(\forall x,y)$ relative(x, y) $\leftrightarrow$ relative(y, x) *;symmetric*

Axioms, definitions and theorems

- **Axioms**: facts and rules that capture (important) facts & concepts in a domain; axioms are used to prove **theorems**
  - Mathematicians dislike unnecessary (dependent) axioms, i.e. ones that can be derived from others
  - Dependent axioms can make reasoning faster, however
  - Choosing a good set of axioms is a design problem
- A **definition** of a predicate is of the form "p(X) $\leftrightarrow$ …" and can be decomposed into two parts
  - **Necessary** description: "p(x) $\rightarrow$ …"
  - **Sufficient** description "p(x) $\leftarrow$ …"
  - Some concepts have definitions (e.g., triangle) and some don't (e.g., person)

More on definitions

Example: define father(x, y) by parent(x, y) and male(x)

- **parent(x, y)** is a necessary (but not sufficient) description of father(x, y)

  father(x, y) $\rightarrow$ parent(x, y)

- **parent(x, y) ^ male(x) ^ age(x, 35)** is a sufficient (but not necessary) description of father(x, y):

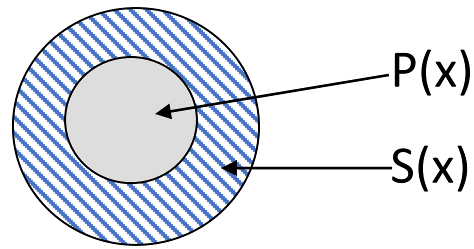
  father(x, y) $\leftarrow$ parent(x, y) ^ male(x) ^ age(x, 35)

- **parent(x, y) ^ male(x)** is a necessary and sufficient description of father(x, y)

  parent(x, y) ^ male(x) $\leftrightarrow$ father(x, y)

More on definitions

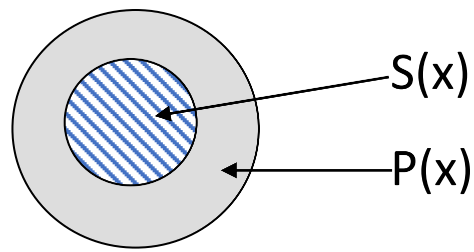
S(x) is a necessary condition of P(x)



 # all Ps are Ss

(∀x) P(x) => S(x)

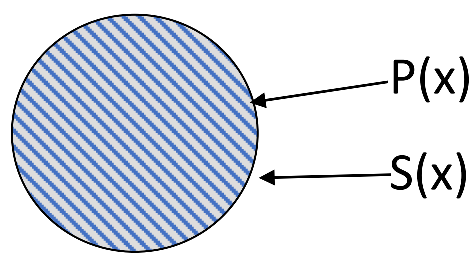
S(x) is a sufficient condition of P(x)



 # all Ps are Ss

(∀x) P(x) <= S(x)

S(x) is a necessary and sufficient condition of P(x)



 # all Ps are Ss

all Ss are Ps

(∀x) P(x) <=> S(x)

Higher-order logic

- FOL only lets us quantify over variables, and **variables can only range over objects**
- HOL allows us to quantify over relations, e.g.
  "two functions are equal iff they produce the same value for all arguments"

  $\forall f \; \forall g \; (f = g) \leftrightarrow (\forall x \; f(x) = g(x))$
- E.g.: (quantify over predicates)

  $\forall r \; \text{transitive}(r) \rightarrow (\forall xyz) \; r(x,y) \wedge r(y,z) \rightarrow r(x,z))$
- More expressive, but reasoning is undecide-able, in general

Expressing uniqueness

syntactic sugar

- Often want to say that there is a single, unique object that satisfies a condition
- There exists a unique x such that king(x) is true
  - $\exists$x king(x) $\wedge$ $\forall$y (king(y) $\rightarrow$ x=y)
  - $\exists$x king(x) $\wedge$ $\neg\exists$y (king(y) $\wedge$ x$\neq$y)
  - $\exists$! x king(x)
- Every country has exactly one ruler
  - $\forall$c country(c) $\rightarrow$ $\exists$! r ruler(c,r)
- Iota operator: ι x P(x) means "the unique x such that p(x) is true"
  - The unique ruler of Freedonia is dead
  - dead(ι x ruler(freedonia,x))

Examples of FOL in use

- Semantics of W3C's Semantic Web stack (RDF, RDFS, OWL) is defined in FOL
- OWL Full is equivalent to FOL
- Other OWL profiles support a subset of FOL and are more efficient
- The semantics of schema.org is only defined in natural language text
- Wikidata's knowledge graph (and Google's) has a richer schema

FOL Summary

- First order logic (FOL) introduces predicates, functions and quantifiers
- More expressive, but reasoning more complex
  - Reasoning in propositional logic is NP hard, FOL is semi-decidable
- Common AI knowledge representation language
  - Other KR languages (e.g., OWL) are often defined by mapping them to FOL
- FOL variables range over objects
  - HOL variables range over functions, predicates or sentences

Logical Inference: Overview

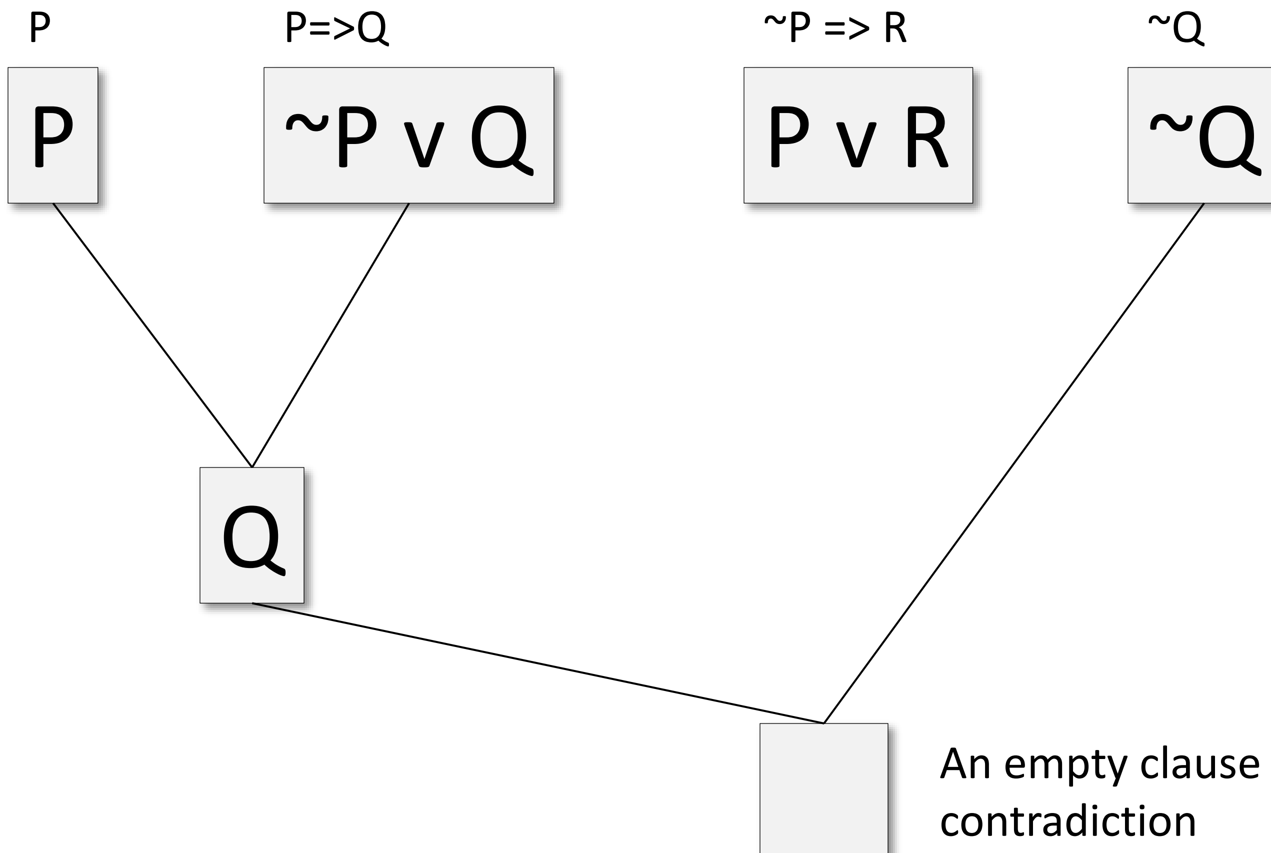
- Model checking for propositional logic
- Rule based reasoning in first-order logic
  - Inference rules and generalized modes ponens
  - Forward chaining
  - Backward chaining
- Resolution-based reasoning in first-order logic
  - Clausal form
  - Unification
  - Resolution as search

From Satisfiability to Proof

- To see if a satisfiable KB entails sentence S, see if $\underline{KB \wedge \neg S}$ is satisfiable
  - If it is not, then the KB entails S
  - If it is, then the KB does not entail S
  - This is a refutation proof
- Consider the KB with (P, P=>Q, ~P=>R)
  - Does the KB it entail Q? R?

Does the KB entail Q?

KB
P
P=>Q
~P=>R

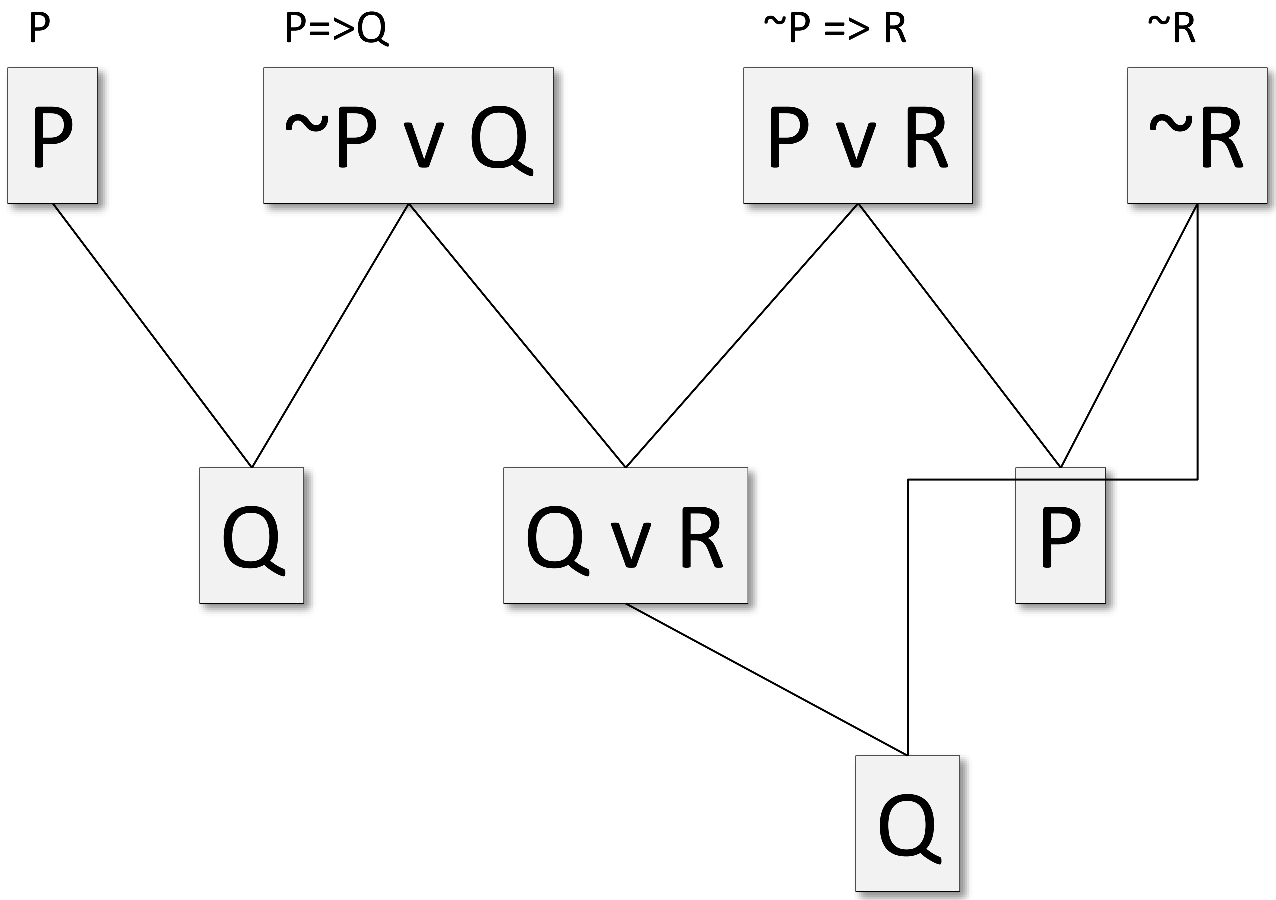


An empty clause represents a contradiction

We assume that every sentence in the KB is true. Adding ~Q to the KB yields a contradiction, so ~Q must be false, so **Q must be true**.

Does the KB entail R?

KB
P
P=>Q
~P=>R



Adding ~R to KB does not produce a contradiction after drawing all possible conclusions, so it could be False, so **KB doesn't entail R**. (but we also can't say KB entails not R).

Propositional logic model checking

- Given KB, does a sentence S hold?
  - All the variables in S must be in the KB
  - A candidate model is just an assignment of T|F to every variable in the KB
- Basically generate and test:
  - Consider candidate models M for the KB
  - If $\forall$M S is true, then S is **provably true**
  - If $\forall$M $\neg$S, then S is **provably false**
  - Otherwise ($\exists$M1 S $\wedge$ $\exists$M2 $\neg$S): S is **satisfiable** but neither provably true or provably false

Efficient PL model checking (1)

Davis-Putnam algorithm (DPLL) is generate-and-test model checking with several optimizations:

- *Early termination:* short-circuiting of disjunction or conjunction sentences
- *Pure symbol heuristic*: symbols appearing only negated or un-negated must be FALSE/TRUE respectively

  e.g., in [(A$\vee\neg$B), ($\neg$B$\vee\neg$C), (C$\vee$A)] A & B are pure, C impure.
  Make pure symbol literal true: if there's a model for S, making pure symbol true is also a model
- *Unit clause heuristic*: Symbols in a clause by itself can immediately be set to TRUE or FALSE

Using the AIMA Code

```
python> python
Python ...
>>> from logic import *
>>> expr('P & P==>Q & ~P==>R')
((P & (P >> Q)) & (~P >> R))

>>> dpll_satisfiable(expr('P & P==>Q & ~P==>R'))
{R: True, P: True, Q: True}

>>> dpll_satisfiable(expr('P & P==>Q & ~P==>R & ~R'))
{R: False, P: True, Q: True}

>>> dpll_satisfiable(expr('P & P==>Q & ~P==>R & ~Q'))
False

>>>
```

expr parses a string, and returns a logical expression

dpll_satisfiable returns a model if satisfiable else False

The KB entails Q but does not entail R

Efficient PL model checking (2)

- WalkSAT: a local search for satisfiability: Pick a symbol to flip (toggle TRUE/FALSE), either using min-conflicts *or* choosing randomly
- …or use *any* local or global search algorithm
- Many model checking algorithms & systems:
  - E.g.: MiniSat: minimalistic, open-source SAT solver developed to help researchers & developers use SAT"
  - E.g.: International SAT Competition (2002…2020): identify new challenging **benchmarks** to promote new **solvers** for Boolean SAT"

AIMA KB Class

```
>>> kb1 = PropKB()
>>> kb1.clauses
[]
>>> kb1.tell(expr('P==>Q & ~P==>R'))
>>> kb1.clauses
[(Q | ~P), (R | P)]
>>> kb1.ask(expr('Q'))
False
>>> kb1.tell(expr('P'))
>>> kb1.clauses
[(Q | ~P), (R | P), P]
>>> kb1.ask(expr('Q'))
{}
>>> kb1.retract(expr('P'))
>>> kb1.clauses
[(Q | ~P), (R | P)]
>>> kb1.ask(expr('Q'))
False
```

PropKB is a subclass

A sentence is converted to CNF and the clauses added

The KB does not entail Q

After adding P the KB does entail Q

Retracting P removes it and the KB no longer entails Q

Logic Summary

- **Propositional logic**
  - Problems with propositional logic
- **First-order logic**
  - Properties, relations, functions, quantifiers, …
  - Terms, sentences, wffs, axioms, theories, proofs, …
  - Variations and extensions to first-order logic
- **Logical agents**
  - Reflex agents
  - Representing change: situation calculus, frame problem
  - Preferences on actions
  - Goal-based agents