MUC, 2022

Knowledge-Base

Knowledge-based Agents

Propositional - First order logic

Dr. Bilal Hoteit

Table of contents

Introduction.

- What is logic?
  a. The term logic comes from the Greek word logos.
  b. From a philosophical perspective: Logic is a way of thinking clearly and basing your reasoning on objective facts. philosophers try to distinguish good reasoning from bad reasoning.
  c. From a mathematical perspective: A statement is a sentence or a mathematical expression that is either definitely true or definitely false.
  d. From a Sciences perspective: deals with logic as a tool or a language to represent knowledge and reason about it.

Introduction.

- Simple example?

  The car is not (red or green)
  - The car is not red and it's not green

  logical equivalence:
  - $\sim (A \lor B) = \sim A \land \sim B$

  Question – can we claim “car is not red”?

Introduction.

- Simple example?

Propositions: a *car is red* *(A)*
another *car is green* *(B)*

Knowledge: $\sim(A \lor B)$
Inference: $\sim(A \lor B)$
$\sim A \land \sim B$ *(logical equivalence)*
$\sim A$ *(and elimination)*

Answer: car is not red

Propositional logic: Syntax.

- The proposition symbols S1, S2 etc… are sentences

- If S is a sentence, $\neg$ S is a sentence (negation)
- If S1 and S2 are sentences, S1 $\wedge$ S2 is a sentence (conjunction)
- If S1 and S2 are sentences, S1 $\vee$ S2 is a sentence (disjunction)
- If S1 and S2 are sentences, S1 $\Rightarrow$ S2 is a sentence (implication)
- If S1 and S2 are sentences, S1 $\Leftrightarrow$ S2 is a sentence (biconditional)

Propositional logic: Semantics.

- Each model specifies true/false for each proposition symbol

E.g. S1 S2 S3

false true false

With these symbols, 8 possible models, can be enumerated automatically.

- Rules for evaluating truth with respect to a model m:

$\neg S$	is true iff	S is false	
$S_1 \wedge S_2$	is true iff	S_1 is true and	S_2 is true
$S_1 \vee S_2$	is true iff	S_1 is true or	S_2 is true
$S_1 \Rightarrow S_2$	is true iff	S_1 is false or	S_2 is true
$S_1 \Rightarrow S_2$	is false iff	S_1 is true and	S_2 is false
$S_1 \Leftrightarrow S_2$	is true iff	$S_1 \Rightarrow S_2$ is true and	$S_2 \Rightarrow S_1$ is true

- Simple recursive process evaluates an arbitrary sentence, e.g.,

$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \mathit{true} \wedge (\mathit{true} \vee \mathit{false}) = \mathit{true} \wedge \mathit{true} = \mathit{true}$

Propositional logic: Logical equivalences.

Two sentences are logically equivalent iff true in same models:

$\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{de Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Propositional logic: Truth tables .

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

Propositional logic: Validity of Arguments.

$$p \vee (q \vee r)$$

$$\sim r$$

$$\therefore p \vee q$$

Is it a valid argument?
Also called: Query

				premises		conclusion
p	q	r	$q \vee r$	$p \vee (q \vee r)$	$\sim r$	$p \vee q$
T	T	T	T	T	F	
T	T	F	T	T	T	T
T	F	T	T	T	F	
T	F	F	F	T	T	T
F	T	T	T	T	F	
F	T	F	T	T	T	T
F	F	T	T	T	F	

critical rows

In each situation where the premises are both true, the conclusion is also true, so the argument form is valid.

Propositional logic: Validity of Arguments.

$$p \to q \vee \sim r$$

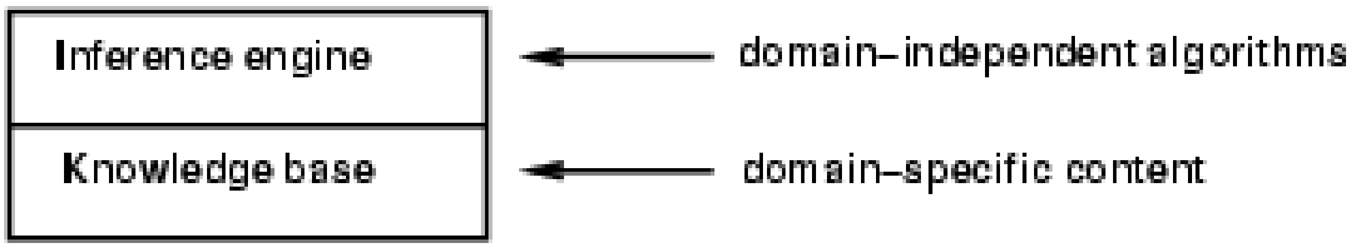
$$q \to p \wedge r$$

$$\therefore p \to r$$

Is it a valid argument?
Also called: Query

						premises		conclusion
p	q	r	$\sim r$	$q \vee \sim r$	$p \wedge r$	$p \to q \vee \sim r$	$q \to p \wedge r$	$p \to r$
T	T	T	F	T	T	T	T	T
T	T	F	T	T	F	T	F	F
T	F	T	F	F	T	F	T	T
T	F	F	T	T	F	T	T	F
F	T	T	F	T	F	T	F	T
F	T	F	T	T	F	T	F	T
F	F	T	F	F	F	T	T	T
F	F	F	T	T	F	T	T	T

Logic agents

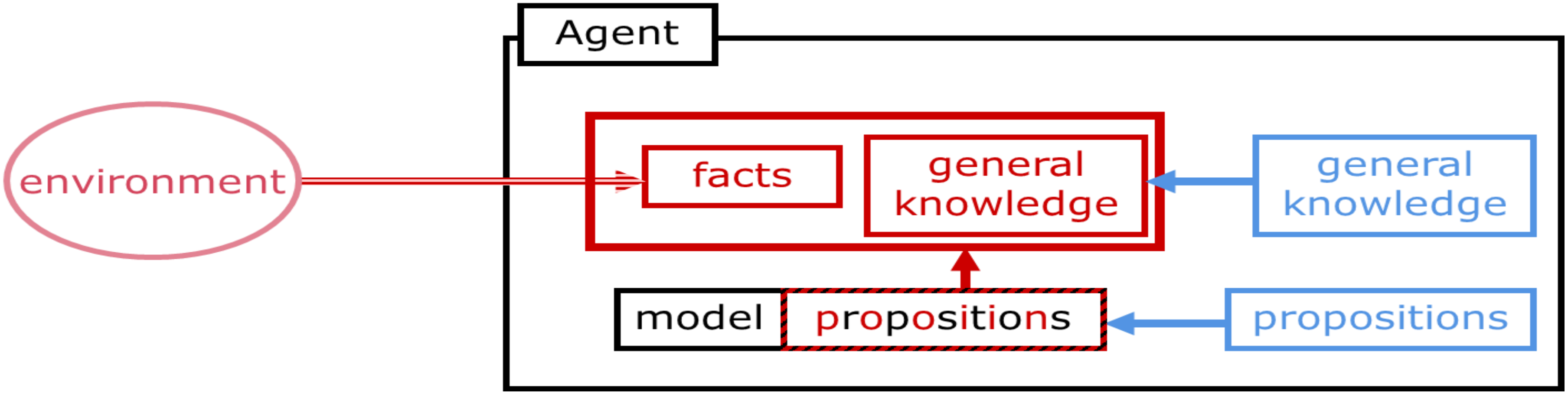


- knowledge-based agent consists of a knowledge base (KB) and inference engine (IE).

a. <u>propositions:</u> set of atomic statements that may be true or false
b. <u>general knowledge:</u> complex sentences describing conditions on environment
c. <u>facts:</u> data (from perceptions) about specific state of environment
d. <u>KB - knowledge base:</u> conjunction of general knowledge and facts
e. <u>model:</u> assignment of true/false values to the propositions

f. The Inference engine derives new sentences from the input and KB
g. The inference mechanism depends on representation in KB

Logic agents



- The agent operates as follows:
  a. It receives percepts from environment
  b. It computes what action it should perform (by IE and KB)
  c. It performs the chosen action (some actions are simply inserting inferred new facts into KB).

knowledge-based agent.

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

  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
  action ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
  TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
  $t \leftarrow t + 1$
  return *action*

- The agent must be able to:

  a. Represent states, actions, etc..
  b. Incorporate new percepts
  c. Update internal representations of the world
  d. Deduce hidden properties of the world

knowledge-based agent.

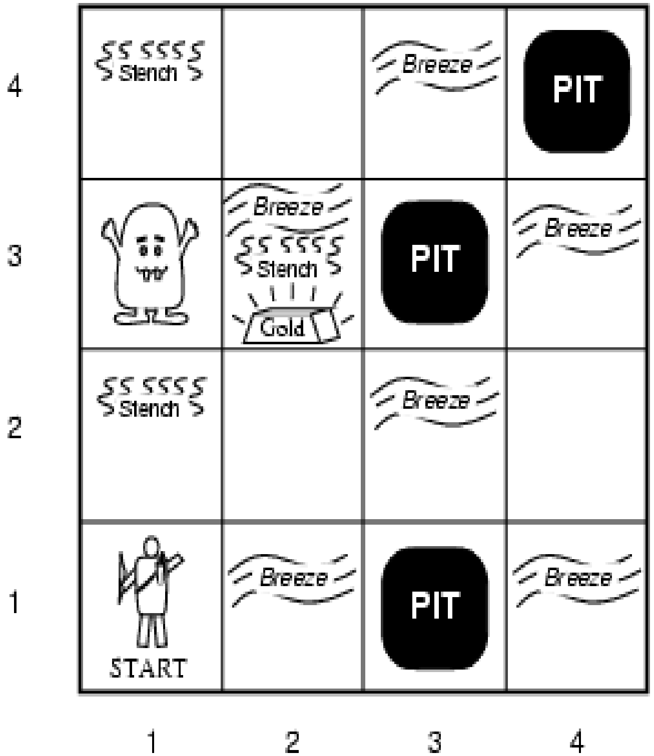
```
function KB-AGENT(percept) returns an action
    static: KB, a knowledge base
            t, a counter, initially 0, indicating time

    TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
    action ← ASK(KB, MAKE-ACTION-QUERY(t))
    TELL(KB, MAKE-ACTION-SENTENCE(action, t))
    t ← t + 1
    return action
```

- The agent must be able to:

    a. Represent states, actions, etc..
    b. Incorporate new percepts
    c. Update internal representations of the world
    d. Deduce hidden properties of the world

Wumpus World PEAS description.



- Performance measure
  a. gold +1000, death -1000
  b. -1 per step, -10 for using the arrow

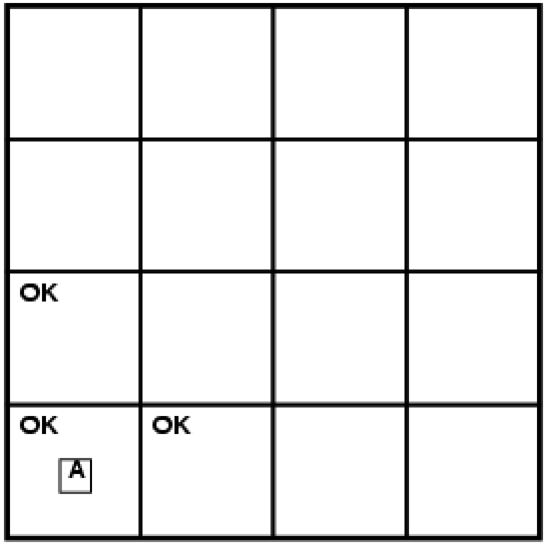
- Environment

  a. Squares adjacent to wumpus are smelly
  b. Squares adjacent to pit are breezy
  c. Glitter iff gold is in the same square
  d. Shooting kills wumpus if you are facing it
  e. Shooting uses up the only arrow
  f. Grabbing picks up gold if in same square
  g. Releasing drops the gold in same square

- Sensors: [Stench, Breeze, Glitter, Bump, Scream]

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

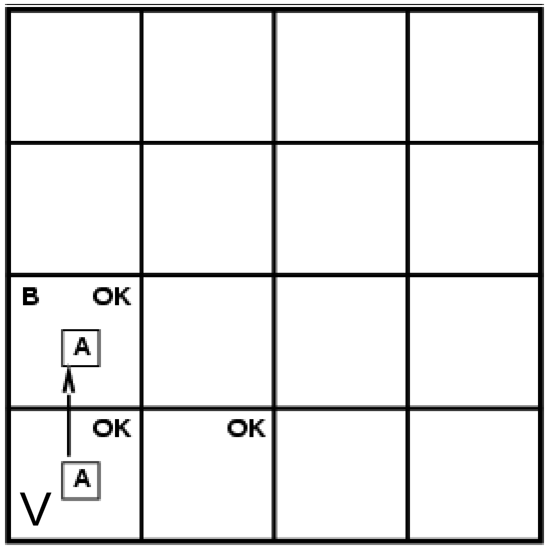
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [None, none, none, none, none]

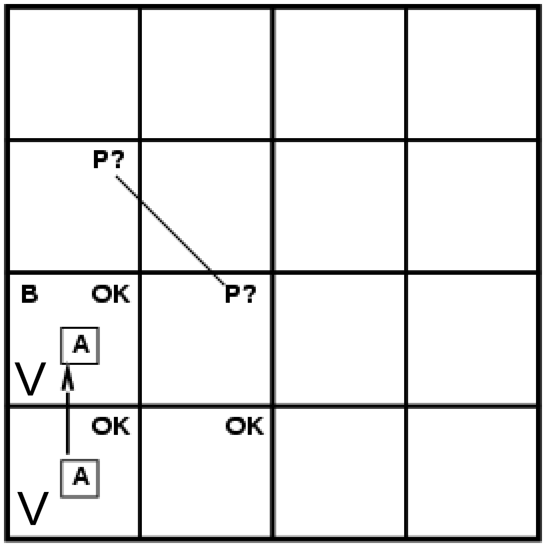
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [None, none, none, none, none]

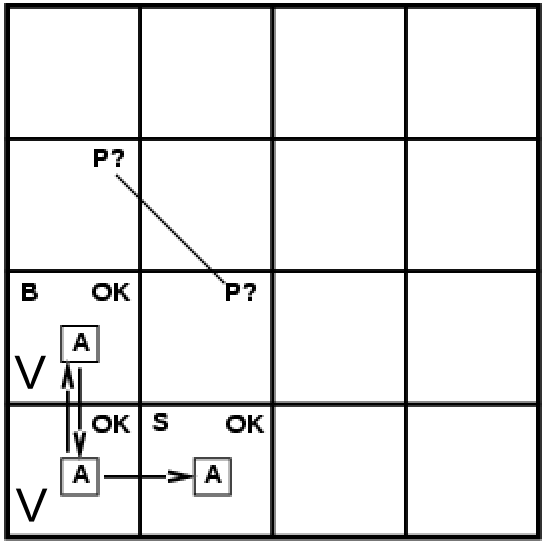
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [None, breeze, none, none, none]

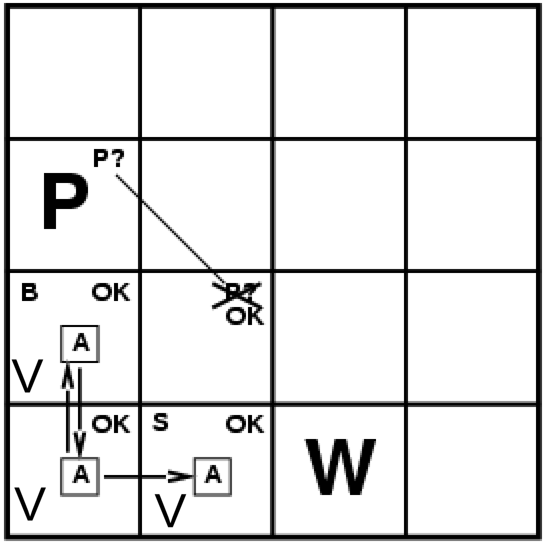
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [Stensh, None, none, none, none]

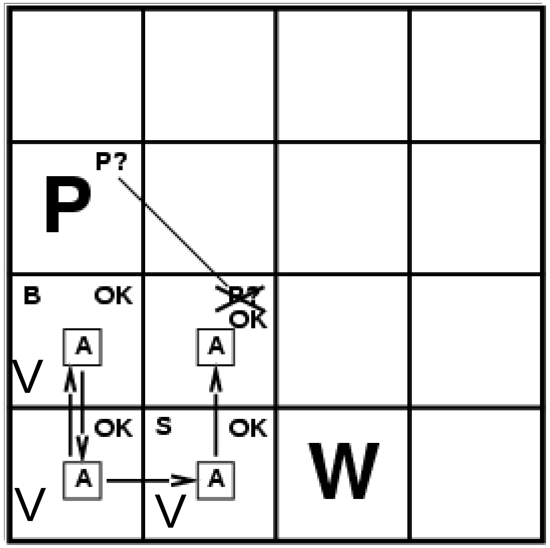
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [Stensh, None, none, none, none]

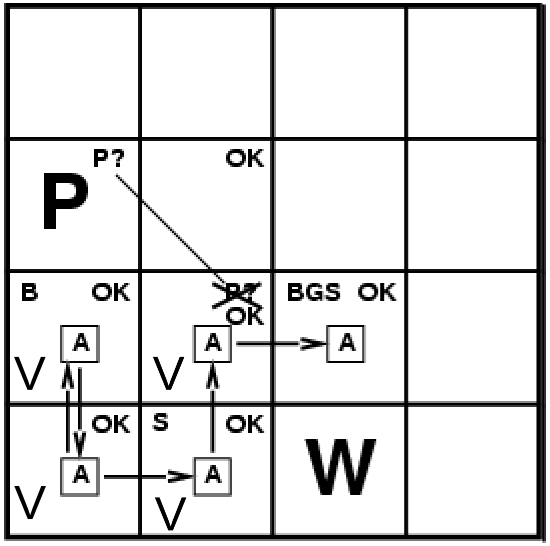
Wumpus World Simulation.



A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

- Sensors: [None, None, none, none, none] ?????

Wumpus World Simulation.

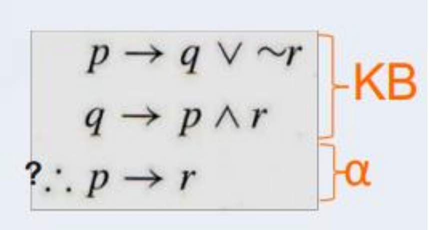


A : Agent
B: Breeze
G: Glitter, Gold
Ok: Safe square
P: Pit
S: Stensh
V: Visited
W: Wumpus

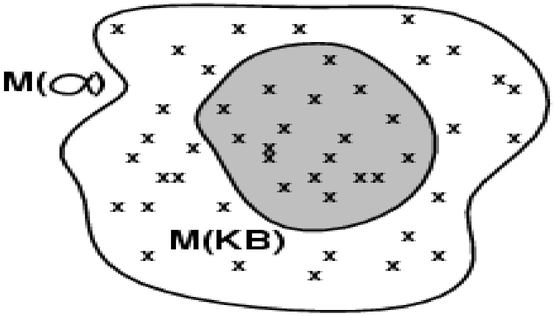
- Sensors: [None, None, none, none, none]

Reasoning prerequisites.

- Entailment
  - Entailment means that one thing follows from another:
    KB ╞ α

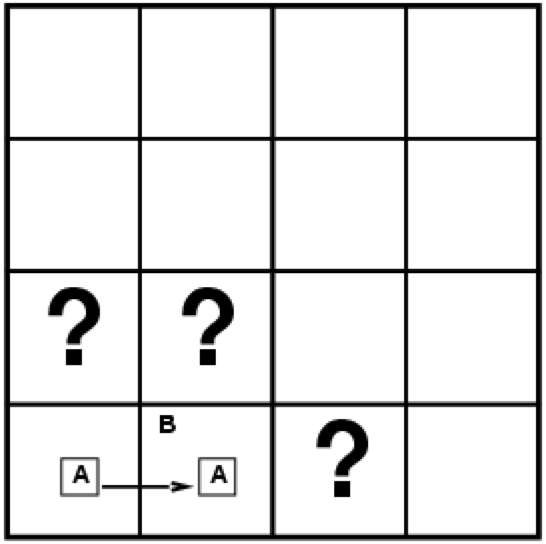


  - Knowledge base KB entails sentence α if and only if α is true in all worlds where KB is true
  - E.g., x+y = 4 entails 4 = x+y
  - Entailment is a relationship between sentences (i.e., syntax) that is based on semantics
  - We say m is a model of a sentence α if α is true in m
  - M(α) is the set of all models of α
  - Then KB ╞ α iff M(KB) $\subseteq$ M(α)



Reasoning Simulation (Entailment).

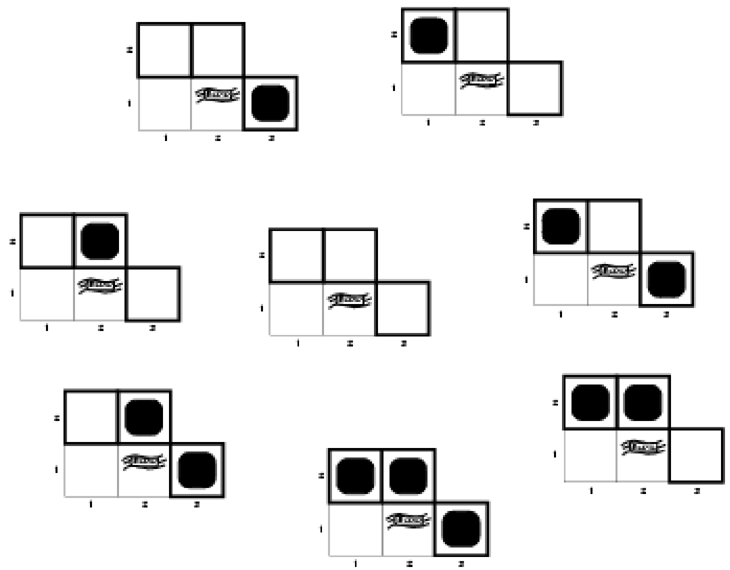
Situation after detecting nothing in [1,1], moving right, breeze in [2,1]



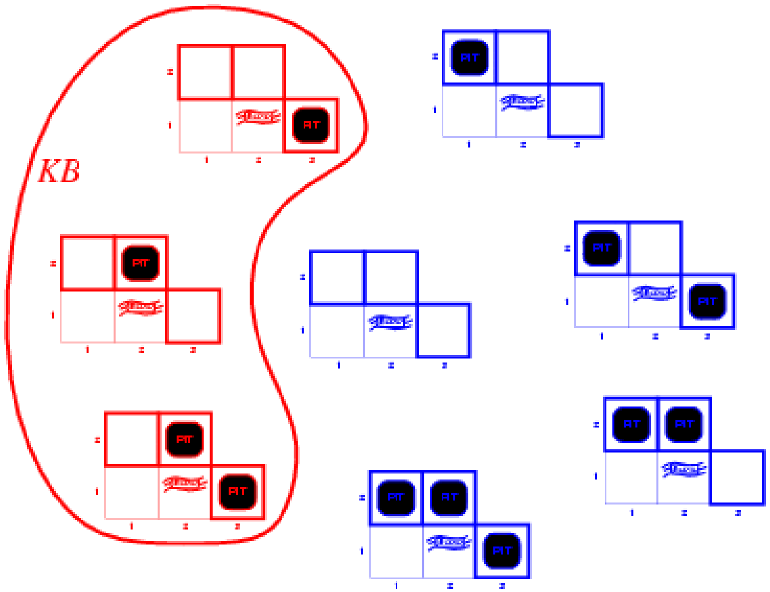
Consider possible models for *KB* assuming only pits

3 Boolean choices $\Rightarrow$ 8 possible models

Reasoning Simulation (Entailment).

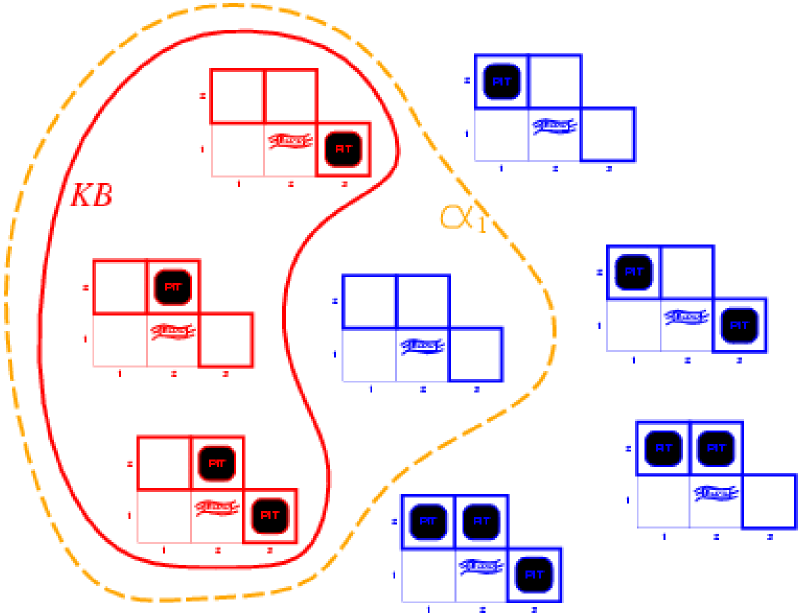


Reasoning Simulation (Entailment).



KB = wumpus-world rules + observations

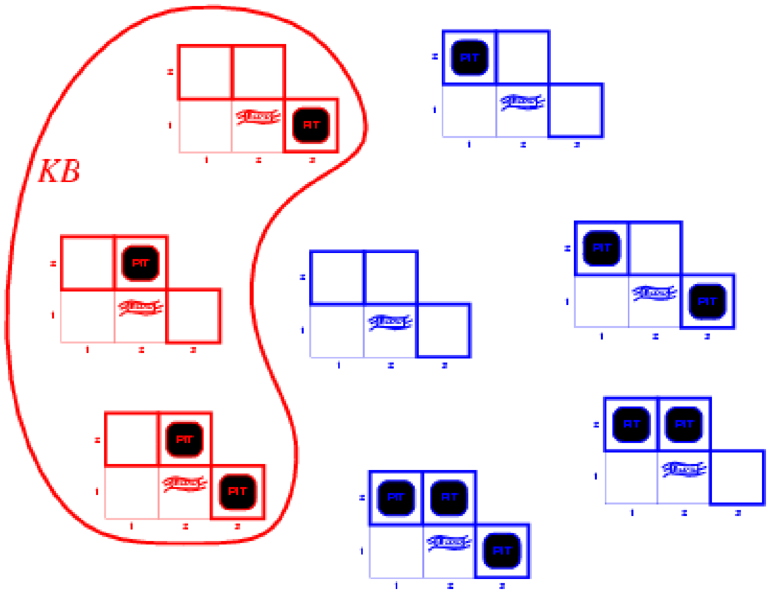
Reasoning Simulation (Entailment).



KB = wumpus-world rules + observations

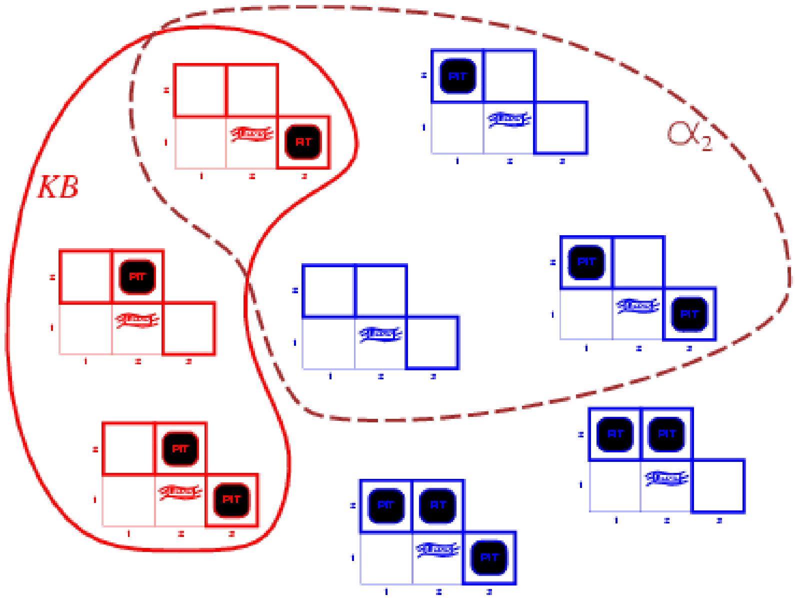
α_1 = "[1,2] is safe", *KB* $\models \alpha_1$, proved by model checking

Reasoning Simulation (Entailment).



KB = wumpus-world rules + observations

Reasoning Simulation (Entailment).



KB = wumpus-world rules + observations

α_2 = "[2,2] is safe", $KB \models \alpha_2$

KB Simulation.

- $P_{x,y}$ is true if there's a pit in $[x, y]$
- $W_{x,y}$ is true if there is a Wumpus in $[x, y]$
- $B_{x,y}$ is true if the agent perceives a breeze in $[x, y]$
- $S_{x,y}$ is true if the agent perceives a stench in $[x, y]$

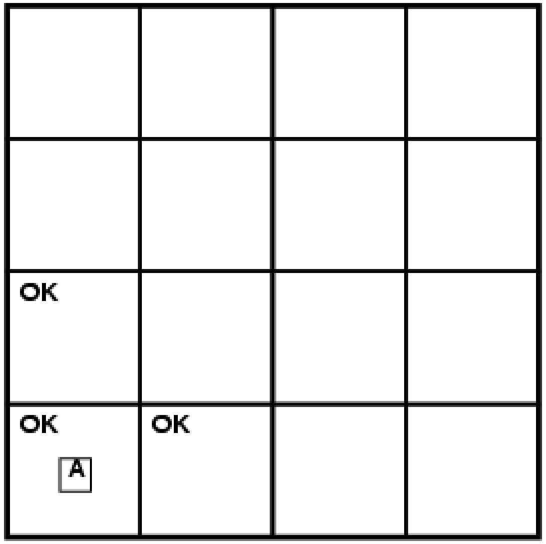
For the Wumpus world in general.

- $R_1 : \neg P_{1,1}$
- $R_2 : B_{1,1} \leftrightarrow (P_{1,2} \vee P_{2,1})$
- $R_3 : B_{2,1} \leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

Now, after visiting [1,1]; [1,2] and [2,1]

- $R_4 : \neg B_{1,1}$
- $R_5 : B_{2,1}$

$KB = R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$



KB Simulation.

I want to find whether my KB says there's no pit in [1,2]

That is, does $KB \models \neg P_{1,2}$?

We say that $\neg P_{1,2}$ is a sentence α

Main goal: decide whether $KB \models \alpha$

α can be a much more complex query

To solve it:

- enumerate the models
- for each model, check that:
- if it is true in α is has to be true in KB

In the Wumpus world: 7 relevant symbols:
$B_{1,1}, B_{2,1}, P_{1,1}, P_{1,2}, P_{2,1}, P_{2,2}, P_{3,1}$
$2^7 = 128$ models. Only 3 are true

Reasoning methods.

- Inference method means checking if a sentence is true in the knowledge-based, this means the sentence is entailed in the knowledge-based.

- Different method exists to check the entailment:

  - Model checking: enumerate all the models in which KB is true to check if the sentence is true.

  - Resolution algorithm: it is used a method of contradiction. To proof KB entails new sentence (KB |= α), we show that Kb and the negation of the sentence: (KB ∧ ¬α) is un-satisfiable. (Convert (KB ∧ ¬α) to CNF).

  - Forward and backward chaining algorithm: transforming the KB to horn clause, then check the premises of the clause, if it is known then the algorithm adds the conclusion of the clause to the set of facts.

  - Advanced algorithms:
    a. DPLL algorithm: Improved Model checking by backtracking.
    b. GSAT, WalkSAT And SATPlan : it uses local search.

Inference by Model checking.

- Truth tables for inference:

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
$false$	$false$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$false$	$false$	$false$	$false$	$false$	$true$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$false$	$true$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$true$	$false$	$false$	$false$	$false$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$false$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$true$	$false$	$false$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$true$	$true$	$true$	$true$	$true$	$true$	$true$	$false$	$false$

Inference by Model checking.

Let $\alpha = A \vee B$ and $KB = (A \vee C) \wedge (B \vee \neg C)$
Is it the case that: $KB \models \alpha$
Check all possible models,
- α must be true wherever KB is true

- Truth tables for inference:

A	B	C	$A \vee C$	$B \vee \neg C$	KB	α
False	*False*	*False*	*False*	*True*	*False*	*False*
False	*False*	*True*	*True*	*False*	*False*	*False*
False	*True*	*False*	*False*	*True*	*False*	*True*
False	*True*	*True*	*True*	*True*	*True*	*True*
True	*False*	*False*	*True*	*True*	*True*	*True*
True	*False*	*True*	*True*	*False*	*False*	*True*
True	*True*	*False*	*True*	*True*	*True*	*True*
True	*True*	*True*	*True*	*True*	*True*	*True*

Inference by Model checking.

- Algorithm: Depth-first enumeration of all models is sound and complete

```
function TT-ENTAILS?(KB, α) returns true or false
    symbols ← a list of the proposition symbols in KB and α
    return TT-CHECK-ALL(KB, α, symbols, [])
---------------------------------------------------------------
function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
    if EMPTY?(symbols) then
        if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
        else return true
    else do
        P ← FIRST(symbols); rest ← REST(symbols)
        return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
                   TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

Inference by Resolution.

- Prerequisites:

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

Inference by Resolution.

➢ Prerequisites:

Many logical arguments are based on a rule which is known as modus ponens or rule of detachment. Assume that p is true and that $p \rightarrow q$ is true. Then you can conclude q.

Formally:
p
$p \rightarrow q$
Then q is satisfied.

here are some examples involving this rule:
p : It is September.
$p \rightarrow q$ (In September, Houston gets a cool -front.)

q : Houston will get a cool-front then ?????

Thus, Houston will get a cool-front this month. TRUE

Inference by Resolution.

➢ Prerequisites: Validity and satisfiability

a. A sentence is valid if it is true in all models,
e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

b. Validity is connected to inference via the Deduction Theorem:
$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

a. A sentence is satisfiable if it is true in some model
e.g., $A \vee B$, C

b. A sentence is unsatisfiable if it is true in no models
e.g., $A \wedge \neg A$

c. Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable

Inference by Resolution.

- Proof by contradiction. e.g: (KB ∧ ¬α) is un-satisfiable.

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

    clauses ← the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
    new ← { }
    loop do
        for each C_i, C_j **in** *clauses* **do**
            resolvents ← PL-RESOLVE(C_i, C_j)
            if *resolvents* contains the empty clause **then return** *true*
            new ← *new* ∪ *resolvents*
        if *new* ⊆ *clauses* **then return** *false*
        clauses ← *clauses* ∪ *new*

Inference by Resolution.

- Proof by contradiction. e.g.: (KB ∧ ¬α) is un-satisfiable.

Suppose we have a knowledge base in this form:

$$\frac{(A \vee \cancel{B}) \wedge (A \vee \neg \cancel{B})}{A}$$

By resolving (A v B) and (A v ¬ B), we obtain (A v A) which is simply A.

Notice that this rule applies only when a knowledge base in form of conjunctions of disjunctions of literals.

Inference by Resolution Simulation

Example

KB = $(P \rightarrow Q) \rightarrow Q$
$(P \rightarrow P) \rightarrow R$
$(R \rightarrow S) \rightarrow \neg(S \rightarrow Q)$

$\alpha = R$

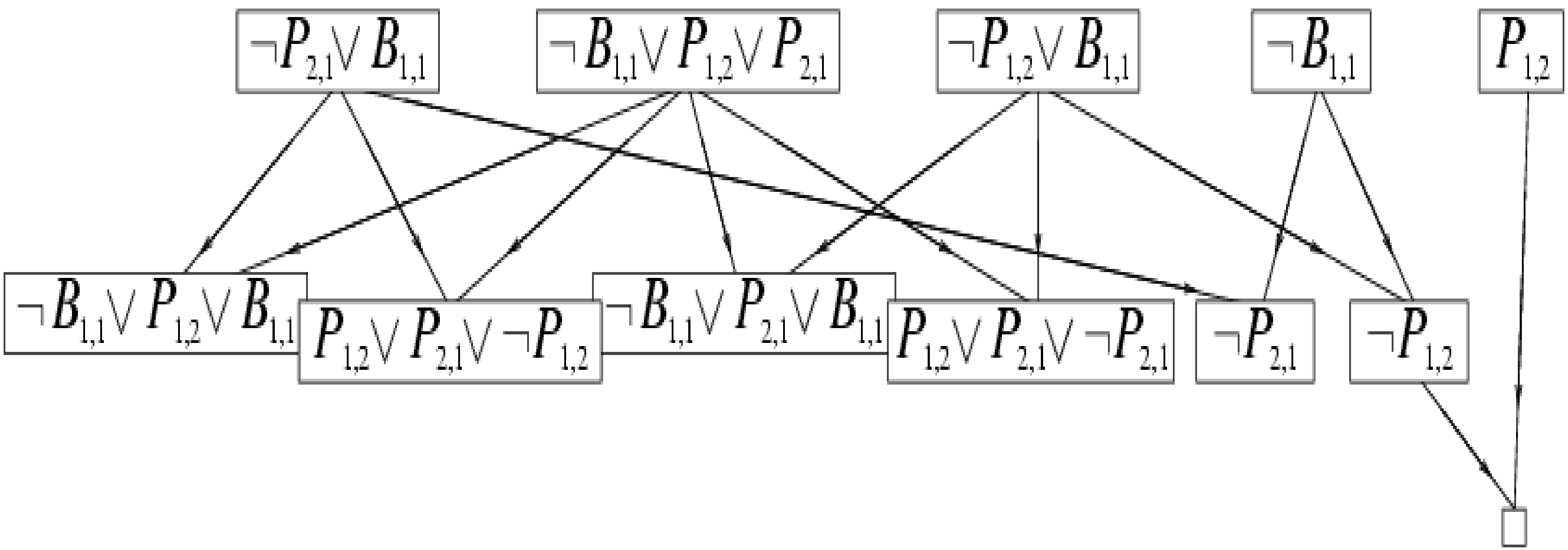
Does KB entails α ($KB \models \alpha$)

Contradiction!

1.	$P \lor Q$	
2.	$P \lor R$	
3.	$\neg P \lor R$	
4.	$R \lor S$	
5.	$R \lor \neg Q$	
6.	$\neg S \lor \neg Q$	
7.	$\neg R$	neg
8.	S	4,7
9.	$\neg Q$	6,8
10.	P	1,9
11.	R	3,10
12.	.	7,11

Inference by Resolution Simulation (2)

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



Inference by Resolution.

- Convert to CNF. (prerequisites)

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

- Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)$.
  $(B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$

- Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \lor \beta$.
  $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg(P_{1,2} \lor P_{2,1}) \lor B_{1,1})$

- Move $\neg$ inwards using de Morgan's rules and double-negation:
  $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land ((\neg P_{1,2} \land \neg P_{2,1}) \lor B_{1,1})$

- Apply distributivity law ($\land$ over $\lor$) and flatten

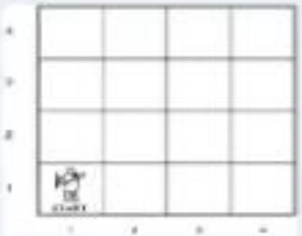
  $(\neg B1,1 \lor P1,2 \lor P2,1) \land (\neg P1,2 \lor B1,1) \land (\neg P2,1 \lor B1,1)$

Inference by Resolution Exercise

Exercise 2

$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}))$ Breeze in [1,1] iff there is a it is [1,2] or [2.1].

$\neg B_{1,1}$ There is on Breeze in [1,1]



$\alpha = P_{1,2}$ Pit in [1,2]?

Does KB entails α ($KB \models \alpha$)

Algorithm works using **proof by contradiction.**

To show $KB \models \alpha$ we show that $KB \wedge \neg\alpha$ is not satisfiable

Apply resolution to $KB \wedge \neg\alpha$ in CNF

and Resolve pairs with complementary literals

$$\frac{l_1 \vee \ldots \vee l_k, \quad m_1 \vee \ldots \vee m_n}{l_1 \vee \ldots l_{i-1} \vee l_{i+1} \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \ldots \vee m_n}$$

if l_i and m_j are complimentary literals

and add new clauses

until

- there are no new clauses to be added
- two clauses resolve to the *empty* class, which means $KB \models \alpha$

Thank you for your attention!

Questions?