Logical Agent

Knowledge-based Agent

- Reusing knowledge or reasoning of knowledge
  - Provide unobserved information
  - New facts can be concluded under logical reasoning
- Reasoning is very important
  - Especially in partially observable problems
    - Results of actions may be unknown

Knowledge-based Agent

Central component

- Memory – **knowledge base**, or **KB**
- Set of representation of facts
- Each is called a **sentence**

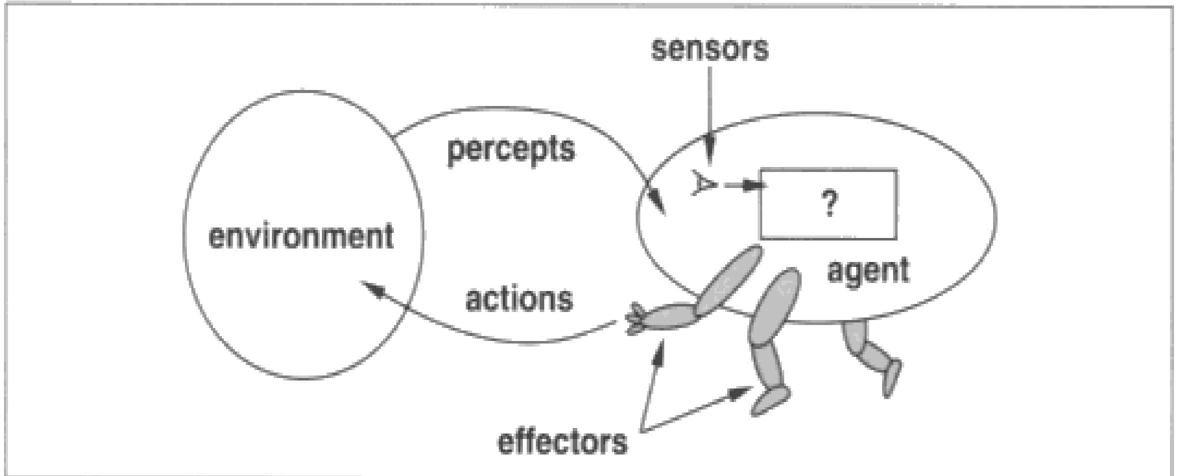


Figure 2.1 Agents interact with environments through sensors and effectors.

Design of Knowledge-based Agent

In partially observable problems

- Results of actions may be unknown
- To do rational action
  - Reasoning in knowledge base is necessary
- A formal language to express the knowledge
  - No ambiguity
  - Carry out reasoning

Logic → Logical agent

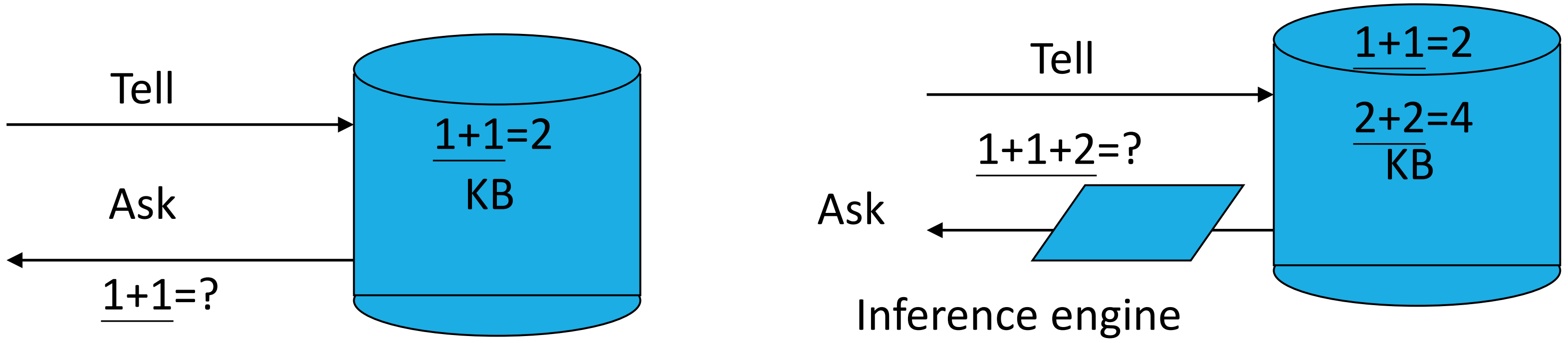
Input & Query KB

To work with KB

- Tell: add new sentences (percepts)
- Ask: query what sentences the KB contains

Main component of KB agent

- ***Inference engine*** (Reasoning)
- Answer query based on sentences in KB



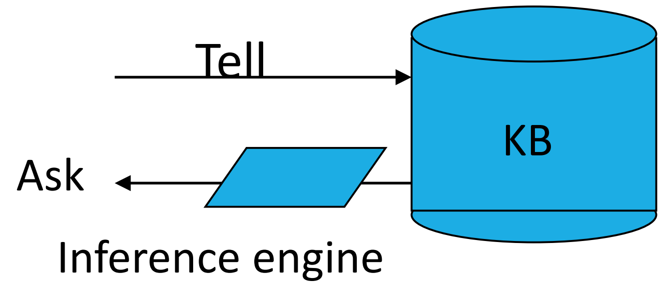
KB Agent

TELL KB what it perceives (Percepts)

ASK KB what action to perform (Actions)

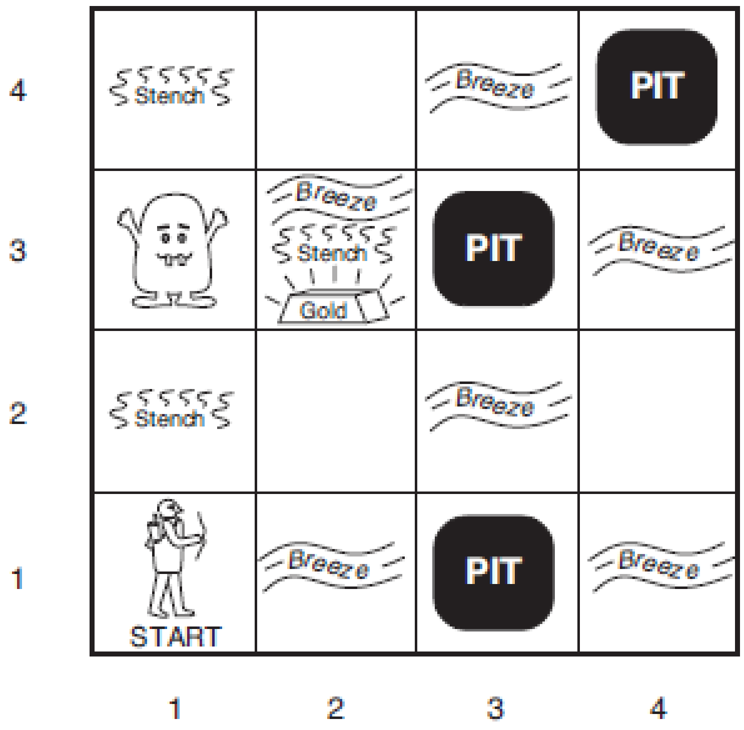
Inference engine do reasoning logically

- Concluding action from sentences in KB
- Action found is proved to be better

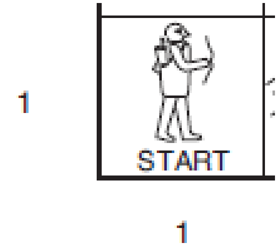


The WUMPUS World

Partially Observable Environment Example



Initial State



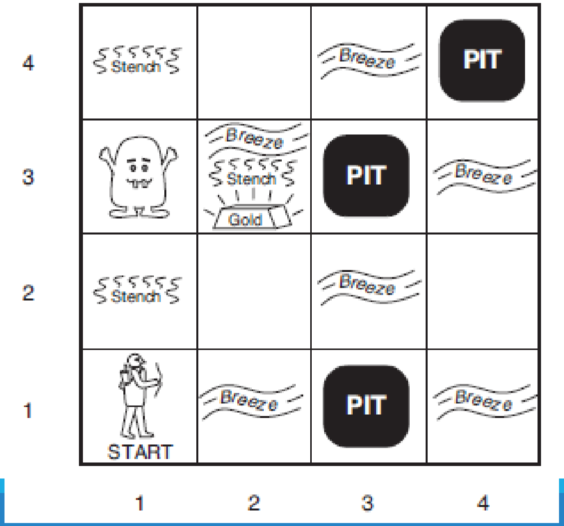
PEAS Description

To well -define the problem

- **P**erformance, **E**nvironment, **A**ctuators, **S**ensors

Performance measure

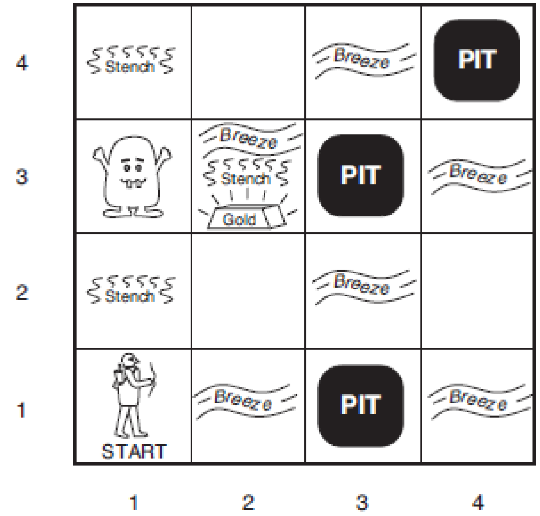
- +1000 for picking up the gold
- -1000 falling into a pit, or being eaten by the wumpus
- -1 for each action taken
- -10 for using up an arrow



PEAS Description

Environment

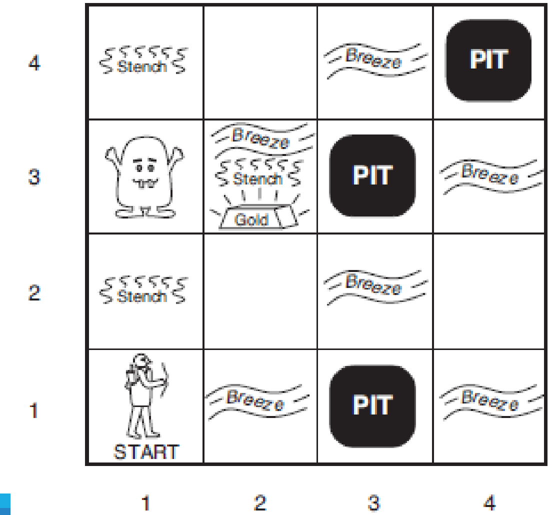
- 4x4 grid of room
  - Player does not know, need to explore
- Agent start in square [1,1]
  - facing to right
- Locations of gold and wumpus
  - Chosen randomly and uniformly
  - Except [1,1]
- Pit
  - Every square, except [1,1], may be a pit
  - With probability 0.2



PEAS Description

Actuators (Actions)

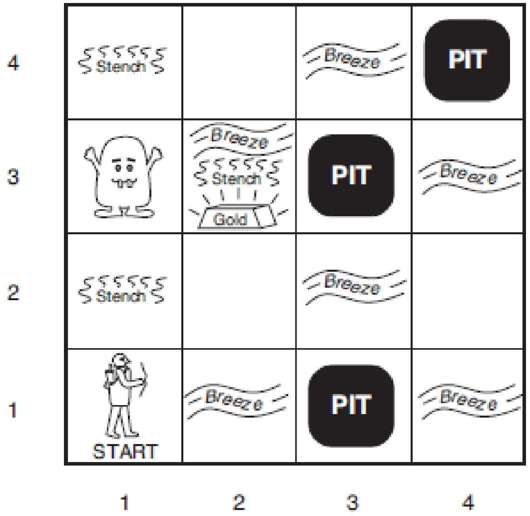
- Move forward, Turn left, Turn right
  - No effect in moving forward when there is wall
- Grab
  - Pick up an object in the same square
- Shoot
  - Fire an arrow in facing direction in straight line
  - Continues till hits wumpus or a wall
  - Can be used only once
- Dies if
  - Enter square with pit or living wumpus



PEAS Description

Sensors –5 percepts

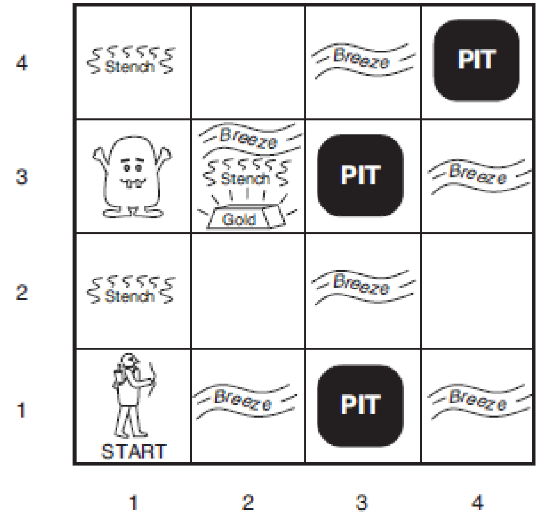
- Stench
  - In square with wumpus (alive or dead)
  - In squares directly adjacent to wumpus
- Breeze
  - In squares directly adjacent to a pit
- Glitter
  - In square containing gold
- Bump
  - Walks into a wall
- Scream
  - In all [x,y] when wumpus is killed



PEAS Description

Percept is expressed

- As a state of five elements
- Like in square [2,3]
  - Percept looks like
  - [***Stench, Breeze, Glitter, None, None***]



Restriction on Agent

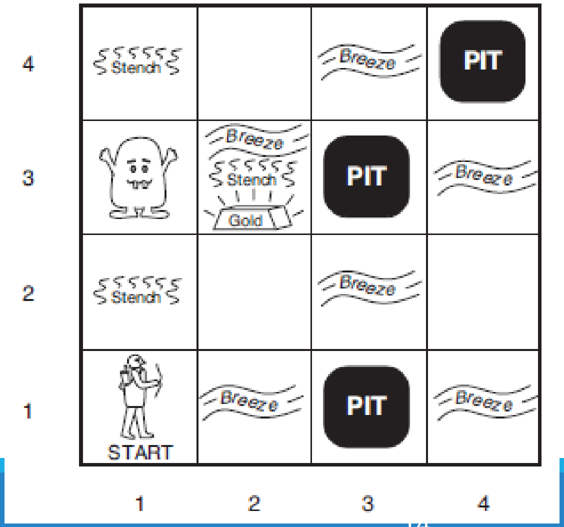
Can only perceive its own location

- Not location adjacent to itself

Partially observable environment

Actions are stochastic (nondeterministic)

- Moving forward → Do not know the result



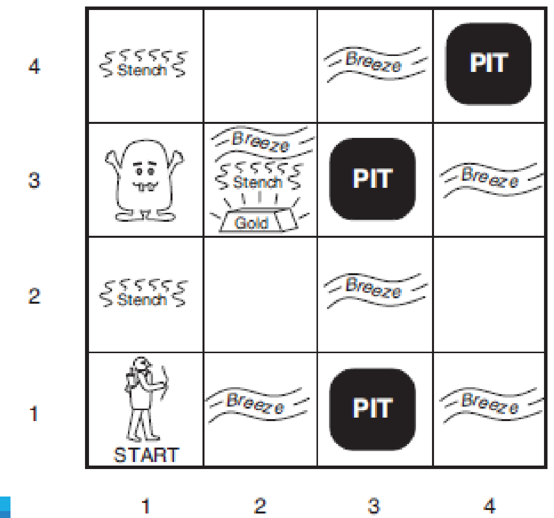
Partially Observable Environment Example

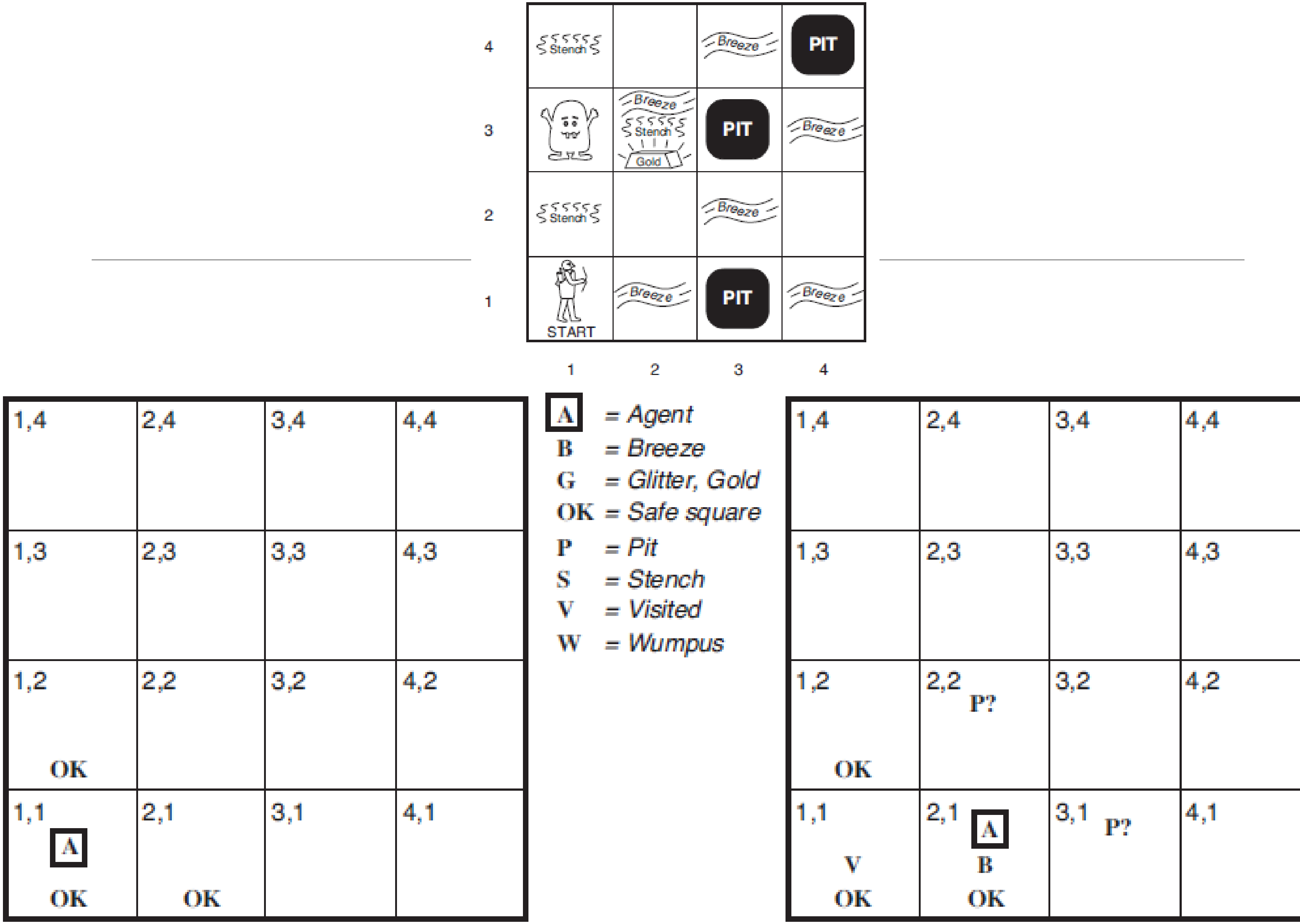
Most case

- Can retrieve gold safely

About 21% of the environments

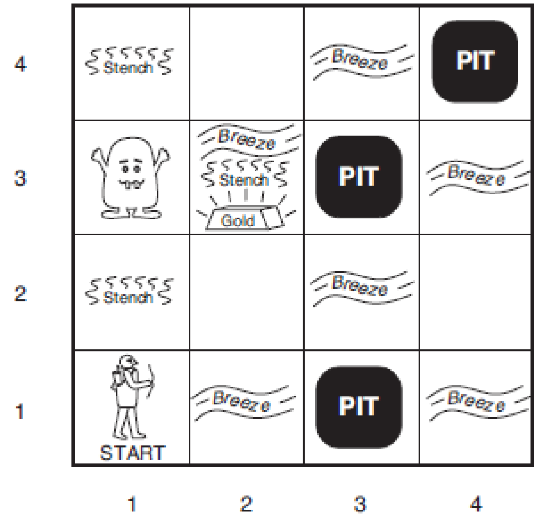
- No way to succeed
- Squares around the gold are pits
- The gold is in a square of pit



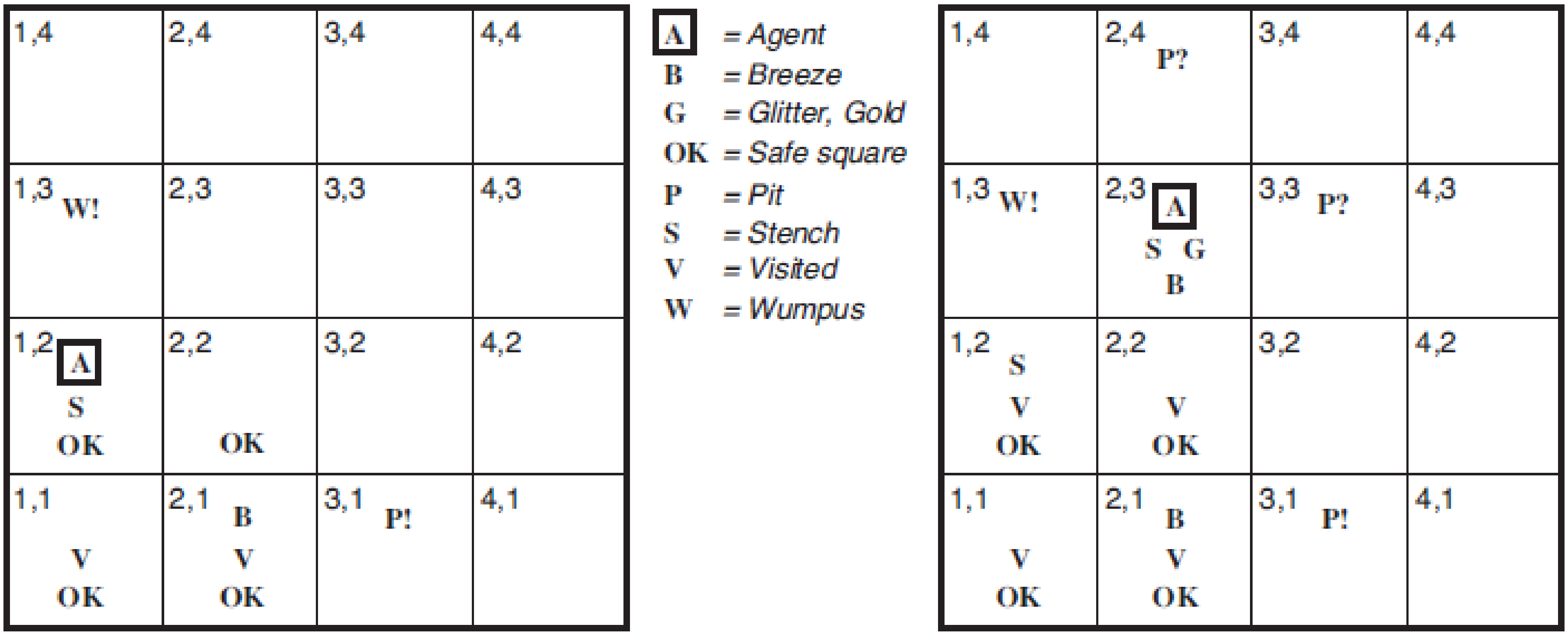


Stench
Breeze
PIT
Gold
START
1
2
3
4
A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus
1,4 2,4 3,4 4,4
1,3 2,3 3,3 4,3
1,2 2,2 3,2 4,2
1,1 2,1 3,1 4,1
A
OK
P?
V
B

(a) (b)



Stench
Breeze
PIT
Gold
START
1 2 3 4



A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus
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
2,4 P?
2,3 A S G B
3,3 P?
1,2 S V OK
2,2 V OK

(a) (b)

Propositional Logic

Propositional Logic

Method of Reasoning

Provides rules and techniques to determine whether an argument is valid

Example

- If x is an even integer, then $x + 1$ is an odd integer

A statement or a proposition

- Declarative sentence that is either true or false, not both

Proposition

Letters denote propositions

Proposition example

- p:2 is an even number (true)
- q: 3 is an odd number (true)
- r: A is a consonant (false)

NOT proposition example

- p: My cat is beautiful
- q: Are you in charge?

Proposition = a Boolean variable

Proposition and Negation

Truth value is assigned to a statement

- True is abbreviated to T or 1
- False is abbreviated to F or 0

Negation

- Negation of p, $\neg$ p
- Statement obtained by negating the statement p
- Example
  - p: A is a consonant
  - $\neg$ p: A is not a consonant

p	**$\neg$ p**
T	F
F	T

Conjunction

Let p and q be statements

- Conjunction of p and q, $p \wedge q$
- Statement formed by joining the two statements with 'and'
- $p \wedge q$ is true only if both p and q are true

p	**q**	**$p \wedge q$**
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction

Let p and q be statements

- Disjunction of p and q, $p \vee q$
- Statement formed by joining the two statements with ‘or’
- $p \vee q$ is true if at least one of p and q is true

p	**q**	**$p \vee q$**
T	T	T
T	F	T
F	T	T
F	F	F

Implication

Let p and q be statements

Implication or condition

- $p \Rightarrow q$

Read as

- If p then q
- p is sufficient for q
- q if p
- q whenever p

p	**q**	**$p \Rightarrow q$**
T	T	T
T	F	F
F	T	T
F	F	T

p is called hypothesis, q is called conclusion

Implication

Let p : Today is Sunday and q: I will wash the car

Implication, $p \Rightarrow q$

- If today is Sunday, then I will wash the car

Converse of implication, $q \Rightarrow p$

- If I wash the car, then today is Sunday

Inverse of implication, $\neg p \Rightarrow \neg q$

- If today is not Sunday, then I will not wash the car

Contrapositive of implication, $\neg q \Rightarrow \neg p$

- If I do not wash the car, then today is not Sunday

Biconditional

Let p and q be statements

Biimplication or biconditional

- $p \Leftrightarrow q$

Read as

- p if and only if q
- p is necessary and sufficient for q
- q if and only if p
- q when and only when p

p	**q**	**$p \Leftrightarrow q$**
T	T	T
T	F	F
F	T	F
F	F	T

Syntax for Propositional Logic

Syntax

- Logical constants: True and False
- Propositional symbols, such as p and q
- Logical connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$ and ()

Sentences in propositional logic

- True, and False
- Propositional symbol
- Wrapping "( )" around a sentence yields a sentence

Sentence for Propositional Logic

Sentence

- Formed by combining sentences with logical connectives
  - $\neg$: negation
  - $\wedge$: conjunction
  - $\vee$: disjunction
  - $\Rightarrow$: implication, $p \Rightarrow q$: if p then q
  - $\Leftrightarrow$: bidirectional

Antecedent / Premise

Conclusion / Consequent

Atomic sentence

- Sentence contains only one symbol or one constant
- p, True

Literal and Complex Sentence

Literal

- Atomic sentence or its negation
- p, ¬q

Complex sentence

- Sentence constructed from simpler sentences using logical connectors

$$
\begin{array}{rcl}
\mathit{Sentence} & \rightarrow & \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} & \rightarrow & \mathbf{True} \mid \mathbf{False} \mid \mathit{Symbol} \\
\mathit{Symbol} & \rightarrow & \mathbf{P} \mid \mathbf{Q} \mid \mathbf{R} \mid \ldots \\
\mathit{ComplexSentence} & \rightarrow & \neg\ \mathit{Sentence} \\
 & \mid & (\ \mathit{Sentence} \wedge \mathit{Sentence}\ ) \\
 & \mid & (\ \mathit{Sentence} \vee \mathit{Sentence}\ ) \\
 & \mid & (\ \mathit{Sentence} \Rightarrow \mathit{Sentence}\ ) \\
 & \mid & (\ \mathit{Sentence} \Leftrightarrow \mathit{Sentence}\ )
\end{array}
$$

Semantics / Interpretation

Sentence → {True, False}

Semantics of propositional logic

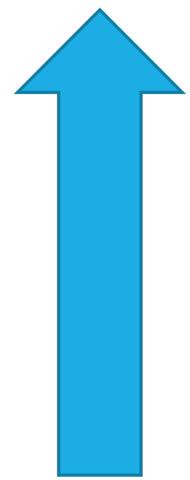
- Interpret truth values of symbols
  - Assign *True* or *False* to the logical symbols
  - Combination of truth values for the logical symbols
    - Models, e.g. m_1 = {P = false, Q = false}
  - Summarize the models
    - Truth table

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*

Precedence of Logical Connectives

Negation	$\neg$
Conjunction	$\wedge$
Disjunction	$\vee$
Implication	$\Rightarrow$
Bidirectional	$\Leftrightarrow$

Highest



Lowest

Example

Let A be the sentence $(\neg(p \vee q)) \Rightarrow (q \wedge p)$

Truth table for A

p	**q**	$(p \vee q)$	$(\neg(p \vee q))$	$(q \wedge p)$	A
T	T	T	F	T	T
T	F	T	F	F	T
F	T	T	F	F	T
F	F	F	T	F	F

Tautology and Contradiction

Tautology

- Sentence is always True
  - Any assignment to the logical symbols in the sentence
- $p \lor \neg p$
- $(p \Rightarrow q) \Leftrightarrow (\neg q \Rightarrow \neg p)$

Contradiction

- Sentence is always False
  - Any assignment to the logical symbols in the sentence
- $p \land \neg p$

Logically Imply and Logically Equivalent

Logically imply

- Implication is a tautology
- *A* logically implies *B*
  - $A \Rightarrow B$ is a tautology, i.e. $A \Rightarrow B$ is always true

Logically equivalent

- Bidirectional is a tautology
- *A* logically equivalent to *B*
  - $A \Leftrightarrow B$ is a tautology, i.e. $A \Leftrightarrow B$ is always true
  - $A \equiv B$

Inference Rules for Propositional Logic

Inference rule

- A rule capturing a certain pattern of inference
- To say β is derived / concluded from α
- Written as $\alpha \vdash \beta$ or $\dfrac{\alpha}{\beta}$

Modus Ponens or **Implication-Elimination**: (From an implication and the premise of the implication, you can infer the conclusion.)

$$\frac{\alpha \Rightarrow \beta, \qquad \alpha}{\beta}$$

And-Elimination: (From a conjunction, you can infer any of the conjuncts.)

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}{\alpha_i}$$

Inference Rules

Modus Ponens (Method of Affirming)

$$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

Modus Tollens (Method of Denying)

$$\frac{\alpha \Rightarrow \beta,\ \neg\,\beta}{\neg\,\alpha}$$

Inference Rules

Disjunctive Syllogisms

$$\frac{\alpha \vee \beta, \neg\alpha}{\beta}$$

Disjunctive Syllogisms

$$\frac{\alpha \vee \beta, \neg\beta}{\alpha}$$

Inference Rules

Disjunctive Addition

$$\frac{\alpha}{\alpha \vee \beta}$$

Disjunctive Addition

$$\frac{\beta}{\alpha \vee \beta}$$

Inference Rules

Conjunctive Simplification

$$\frac{\alpha \wedge \beta}{\alpha}$$

Conjunctive Simplification

$$\frac{\alpha \wedge \beta}{\beta}$$

Conjunctive Addition

$$\frac{\alpha, \beta}{\alpha \wedge \beta}$$

Inference Rules

Hypothetical Syllogism

$$\frac{\alpha \Rightarrow \beta,\ \beta \Rightarrow \gamma}{\alpha \Rightarrow \gamma}$$

Dilemma

$$\frac{\alpha \vee \beta,\ \alpha \Rightarrow \gamma,\ \beta \Rightarrow \gamma}{\gamma}$$

Inference Rules

$(\alpha \land \beta) \equiv (\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta) \equiv (\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha) \equiv \alpha$	double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$	De Morgan
$\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$	De Morgan
$(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Inference Rules

Absorption law

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

Idempotent law

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

Exercise

Verify the equivalences using truth tables

$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$

$\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$

$\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$

$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$

$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$

Proof of $(\neg p \wedge q) \Rightarrow (\neg(q \Rightarrow p))$

$(\neg p \wedge q) \Rightarrow (\neg(q \Rightarrow p))$

$\equiv \neg(\neg p \wedge q) \vee (\neg(q \Rightarrow p))$ by implication elimination

$\equiv(\neg\neg p \vee \neg q) \vee (\neg(q \Rightarrow p))$ by DeMorgan's law

$\equiv(p \vee \neg q) \vee (\neg(q \Rightarrow p))$ by double negation's law

$\equiv(p \vee \neg q) \vee (\neg(\neg q \vee p))$ by implication elimination

$\equiv(p \vee \neg q) \vee (\neg\neg q \wedge \neg p)$ by DeMorgan's law

$\equiv(p \vee \neg q) \vee (q \wedge \neg p)$ by double negation's law

$\equiv p \vee (\neg q \vee (q \wedge \neg p))$ by associativity of $\vee$

Proof of $(\neg p \wedge q) \Rightarrow (\neg(q \Rightarrow p))$

$\equiv p \vee ((\neg q \vee q) \wedge (\neg q \vee \neg p))$ by distributivity

$\equiv p \vee (T \wedge (\neg q \vee \neg p))$ by $\neg \alpha \vee \alpha \equiv T$

$\equiv p \vee (\neg q \vee \neg p)$ by $T \wedge \alpha \equiv \alpha$

$\equiv (p \vee \neg q) \vee \neg p$ by associativity of $\vee$

$\equiv (\neg q \vee p) \vee \neg p$ by commutativity of $\vee$

$\equiv \neg q \vee (p \vee \neg p)$ by associativity of $\vee$

$\equiv \neg q \vee T$ by $\neg \alpha \vee \alpha \equiv T$

$\equiv T$ by $\alpha \vee T \equiv T$

Proof of $(p \wedge \neg q) \vee q \Leftrightarrow p \vee q$

$(p \wedge \neg q) \vee q$	Left-Hand Statement
$\equiv q \vee (p \wedge \neg q)$	by commutativity of $\vee$
$\equiv (q \vee p) \wedge (q \vee \neg q)$	by distributivity
$\equiv (q \vee p) \wedge T$	by $\neg\alpha \vee \alpha \equiv T$
$\equiv q \vee p$	by $\alpha \wedge T \equiv \alpha$
$\equiv p \vee q$	by commutativity of $\vee$

Exercise

Verify the following as a tautology using

- Truth table
- Logic rules

$p \Rightarrow p \vee q$

$Big \vee Dumb \vee (Big \Rightarrow Dumb)$

$(Smoke \Rightarrow Fire) \Rightarrow ((Smoke \wedge Heat) \Rightarrow Fire)$

Exercise

Verify by

- Truth table
- Logic rules

$(p \Rightarrow q) \Leftrightarrow (\neg q \Rightarrow \neg p)$

$((Smoke \land Heat) \Rightarrow Fire) \Leftrightarrow ((Smoke \Rightarrow Fire) \lor (Heat \Rightarrow Fire))$

Knowledge base under Propositional Logic

Simple Knowledge Base

Wumpus world

- Pits and breezes
- Each square needs one proposition

For each i, j:

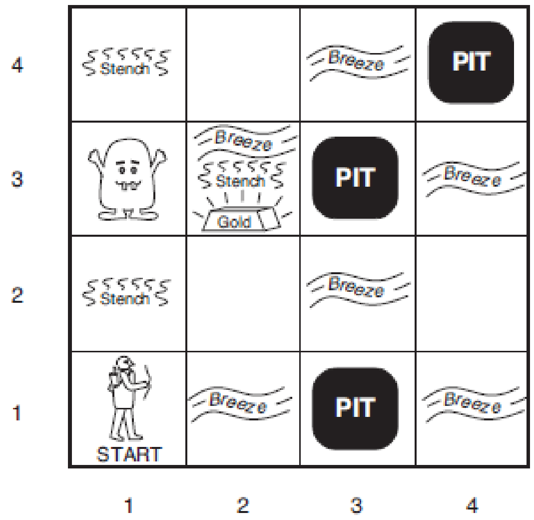
- $P_{i,j}$ = true if there is pit in [i, j]
- $B_{i,j}$ = true if there is breeze in [i, j]

Proposition are stored in knowledge base

- As sentences (rules)

No pits in [1,1]

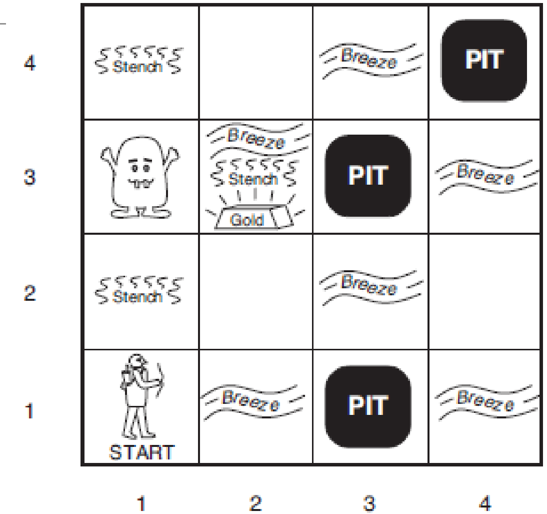
- R1 : $\neg P_{1,1}$



Simple Knowledge Base

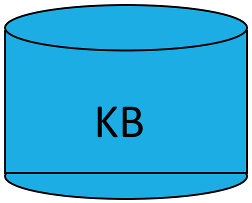
Square is breezy if and only if

- Neighboring square has pit
- R2: $B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
- R3: $B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$
- All squares must be stated

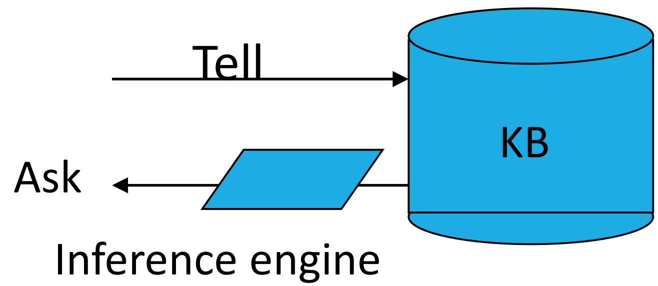


Breeze percepts from agent during runtime

- R4 : $\neg B_{1,1}$
- R5 : $B_{2,1}$



Inference



ASK KB (R1 to R5) a query, [1, 2] is pit?

- i.e. $P_{1,2}$ = true?

Inference engine performs, truth table is constructed

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Inference by Truth Table

Large KB contains many variables

- Need huge amount of memory
  - If there are n variables
  - Totally 2^n rows (models) in truth table
- Time required is also not short
  - Construct the truth table
- Simple rules for inference are preferred

Inference Rules (Revision)

Modus Ponens (Method of Affirming)

$$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

Modus Tollens (Method of Denying)

$$\frac{\alpha \Rightarrow \beta,\ \neg \beta}{\neg \alpha}$$

Inference Rules (Revision)

Conjunctive Simplification

$$\frac{\alpha \wedge \beta}{\alpha}$$

Conjunctive Simplification

$$\frac{\alpha \wedge \beta}{\beta}$$

Conjunctive Addition

$$\frac{\alpha, \beta}{\alpha \wedge \beta}$$

Inference Rules (Revision)

$(\alpha \land \beta) \equiv (\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta) \equiv (\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha) \equiv \alpha$	double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$	De Morgan
$\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$	De Morgan
$(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Rules for Inference

Start with R1 to R5, prove $\neg P_{1,2}$

Apply biconditional elimination to R2

$$R_6 : \quad (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}) \, .$$

Then we apply And-Elimination to R_6 to obtain

$$R_7 : \quad ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}) \, .$$

Logical equivalence for contrapositives gives

$$R_8 : \quad (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})) \, .$$

Now we can apply Modus Ponens with R_8 and the percept R_4 (i.e., $\neg B_{1,1}$), to obtain

$$R_9 : \quad \neg(P_{1,2} \vee P_{2,1}) \, .$$

Finally, we apply De Morgan's rule, giving the conclusion

$$R_{10} : \quad \neg P_{1,2} \wedge \neg P_{2,1} \, .$$

That is, neither [1,2] nor [2,1] contains a pit.

R1 : $\neg P_{1,1}$
R2: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
R3: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
R4 : $\neg B_{1,1}$
R5 : $B_{2,1}$

Rules for Inference

Preceding deviation – called a *proof*

- Sequence of applications of inference rules
- To find the goal sentence

Add new rules to KB

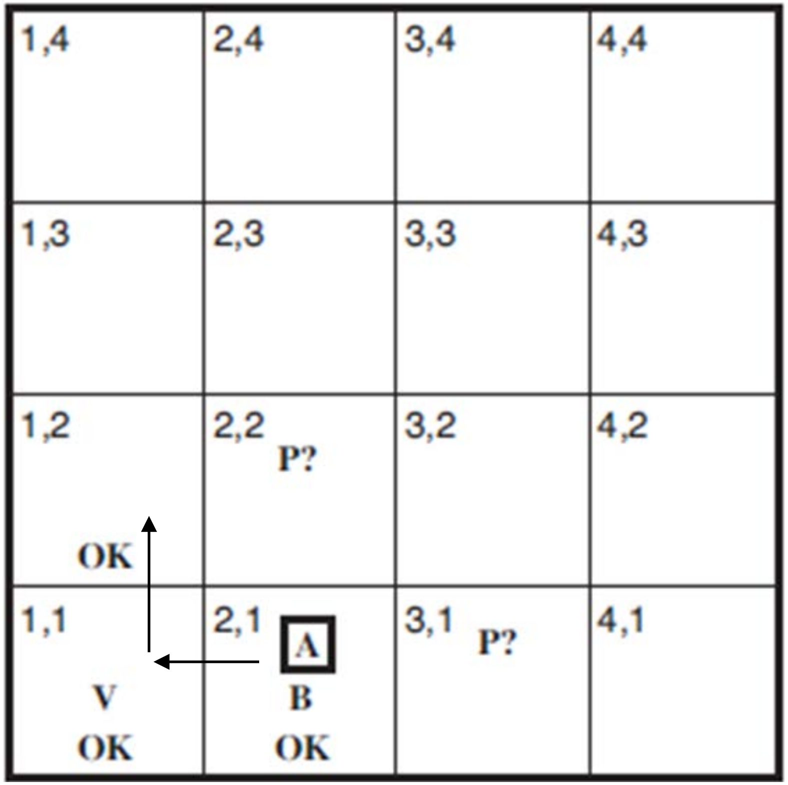
A complete inference algorithm

- Derive all true conclusions from a set of premises

Resolution

Agent

- Returns from [2,1] to [1,1]
- Goes to [1,2]
  - Stench ($S_{1,2}$)
  - No breeze ($\neg B_{1,2}$)
  - TELLed to KB



Resolution

$R_1: \neg P_{1,1}$
$R2: B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
$R3: B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$
$R4: \neg B_{1,1}$
$R5: B_{2,1}$
$R6: (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land$
$((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
$R7: ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
$R8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \lor P_{2,1}))$
$R9: \neg(P_{1,2} \lor P_{2,1})$
$R10: \neg P_{1,2} \land \neg P_{2,1}$

$$R_{11}: \quad \neg B_{1,2}\,.$$
$$R_{12}: \quad B_{1,2} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{1,3})$$

Similarly to getting R10, we know

$$R_{13}: \quad \neg P_{2,2}$$
$$R_{14}: \quad \neg P_{1,3}$$

Apply biconditional elimination to R3, then M.P. with R5, then

$$R_{15}: \quad P_{1,1} \lor P_{2,2} \lor P_{3,1}$$

Apply resolution rule: $\neg P_{2,2}$ in R13 resolves with $P_{2,2}$ in R15

$$R_{16}: \quad P_{1,1} \lor P_{3,1}$$

We know $\neg P_{1,1}$, [1,1] is not pit:

$$R_{17}: \quad P_{3,1}$$

Resolution

Unit resolution

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k}$$

- where ℓ_i and m are complementary literals
  - $m = \neg \ell_i$
- Left hand side: clause
  - A disjunction of literals
- Right hand side: unit clause

For full resolution rule

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n}$$

Resolution

Two more examples

$$\frac{\ell_1 \vee \ell_2, \qquad \neg \ell_2 \vee \ell_3}{\ell_1 \vee \ell_3}$$

$$\frac{P_{1,1} \vee P_{3,1}, \qquad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}} \; .$$

Factoring – removal of multiple copies

- e.g. resolve (A ∨ B) with (A ∨ ¬B)
- Generate (A ∨ A) = A

Conjunctive Normal Form

Resolution rule has a weak point

- Can only be applied to disjunctions of literals $\ell_1 \vee \cdots \vee \ell_k$
- Most sentences are conjunctive
- Sentences are transformed in CNF
  - Expressed as a conjunction of disjunctions of literals

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

Conversion Procedure

We illustrate the procedure by converting R_2, the sentence $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$, into CNF. The steps are as follows:

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}) .$$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$:

   $$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1}) .$$

3. CNF requires $\neg$ to appear only in literals, so we "move $\neg$ inwards" by repeated application of the following equivalences from Figure 7.11:

   $\neg(\neg\alpha) \equiv \alpha$ (double-negation elimination)
   $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ (de Morgan)
   $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ (de Morgan)

   In the example, we require just one application of the last rule:

   $$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1}) .$$

4. Now we have a sentence containing nested $\wedge$ and $\vee$ operators applied to literals. We apply the distributivity law from Figure 7.11, distributing $\vee$ over $\wedge$ wherever possible.

   $$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1}) .$$

The original sentence is now in CNF, as a conjunction of three clauses. It is much harder to read, but it can be used as input to a resolution procedure.

Resolution Algorithm

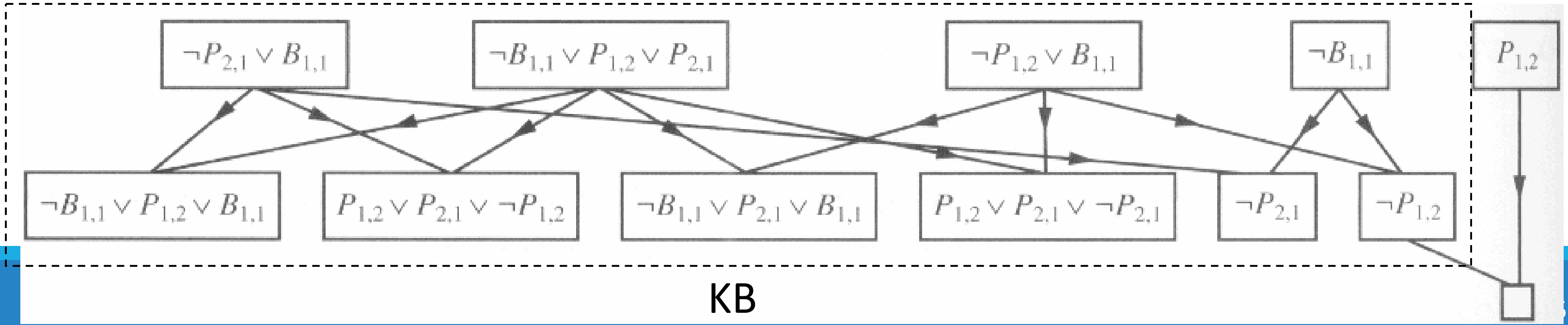
Inference of resolution

- "proof by contradiction", or refutation
  - Show (KB ∧ ¬α) is unsatisfiable
- Prove α, assume ¬α
  - (KB ∧ ¬α) = True → ¬α = True
  - (KB ∧ ¬α) = False → α = True

Steps

- (KB ∧ ¬α) are converted into CNF
- Apply resolution rules to this CNF

$\neg\alpha = P_{1,2}$



Forward and Backward Chaining

Many practical situations, resolution is not needed

Real-world KB only has Horn clauses

Horn clauses

- Disjunction of literals of which at most one is positive
- e.g. $\neg L_{1,1} \vee \neg Breeze \vee B_{1,1}$ is, $\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}$ is not

Horn clause

Implication

- Premise (left hand side) = conjunction of positive literals
- Conclusion (right hand side) = a single positive literal
- e.g. $(L_{1,1} \wedge Breeze) \Rightarrow B_{1,1}$ $\neg L_{1,1} \vee \neg Breeze \vee B_{1,1}$
- $(A \wedge B) \Rightarrow \neg C$
  - No positive literals

Inference

- Forward and backward chaining

Reasoning

- Work in time linear to size of KB
- Cheap for many propositional KB in practice

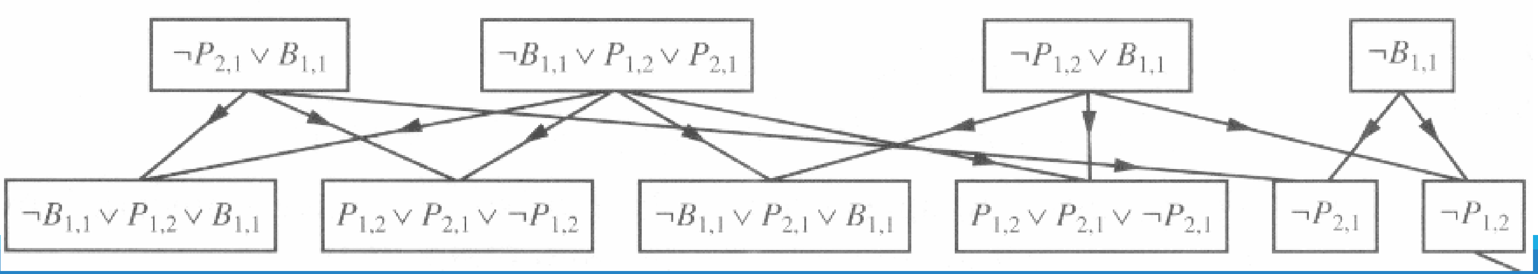
Forward Chaining

- Produce new information
  - Based on set of known facts and clauses
- New information is added to KB
  - Continue produce another set of information
- e.g. $(L_{2,1} \wedge \text{Breeze}) \Rightarrow B_{2,1}$

  $L_{2,1}$

  Breeze

  $B_{2,1}$ is added



Forward Chaining

Prove a proposition q

- Continue producing new facts until
  - Proposition q is added (i.e. result is found)
  - No new facts can be generated
  - Runs in linear time

Data-driven inference

- When new data comes
  - Inference procedure (forward-chaining) is activated

Backward Chaining

Opposite of forward chaining

Query q is asked

- If known (exist in KB), then finished
- Otherwise, finds all implications that conclude q
  - Try to prove all premises in matched implications
  - Every premise is then another query q

Prolog matching and unification

Runs in linear time or ***fewer*** than linear time

- Only relevant clauses about q are matched and used
- Forward-checking randomly selects any clause in KB

q.

$q \leftarrow \ldots$

Agents based on Propositional Logic

For every [x,y], handle pits and wumpuses,

- Rule for breeze $B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$
- Rule for stench $S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$
- Rules for wumpus
  - At least one wumpus: $W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$
  - At most one wumpus
    - For any two squares, one of them must be wumpus-free
    - e.g. $\neg W_{1,1} \vee \neg W_{1,2}$
    - With n squares, (n-1)n/2 sentences
    - 4 x 4 world, n = 16, 120 sentences
      - Each square has many percepts S, B, W, P, …
      - At least 64 distinct symbols

Location & Orientation

Every square

- 4 different orientation for moving forward
  - Up, Down, Left, Right
- For every action, rules are increased to 4 times
  - Too many rules
  - Greatly affect efficiency

$$L_{x,y} \wedge FacingRight \wedge Forward \Rightarrow L_{x+1,y}$$

Agents based on Propositional Logic

Still works in small domain (4 x 4)

Main problem

- Too many ***distinct*** propositions to handle

Weakness of Propositional Logic

- Lack of expressiveness
  - Similar variable must be listed out

Another powerful device

- First-order logic