Adapted from AIMA

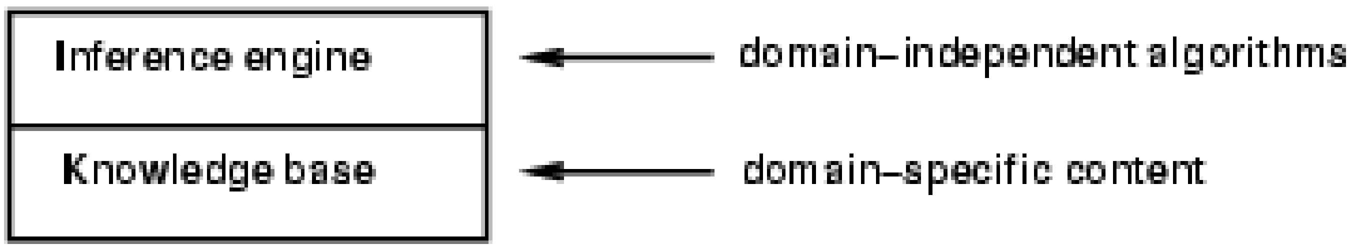
Logic
proof and truth
syntacs and semantics

Peter Antal
antal@mit.bme.hu

Outline

- Knowledge-based agents
- Wumpus world
- Logic in general
  - Syntacs
    - transformational grammars
  - Semantics
    - Truth, meaning, models and entailment
  - Inference
    - Model-based inference methods
    - Syntactic proof methods
- Propositional (Boolean) logic
- On proof and truth
- Inference rules and theorem proving
  - forward chaining
  - backward chaining
  - resolution

Knowledge bases



- Knowledge base = set of sentences in a formal language
- Declarative approach to building an agent (or other system):
  - `Tell` it what it needs to know
- Then it can `Ask` itself what to do - answers should follow from the KB
- Agents can be viewed at the knowledge level
  i.e., what they know, regardless of how implemented
- Or at the implementation level
  - i.e., data structures in KB and algorithms that manipulate them

A simple knowledge-based agent

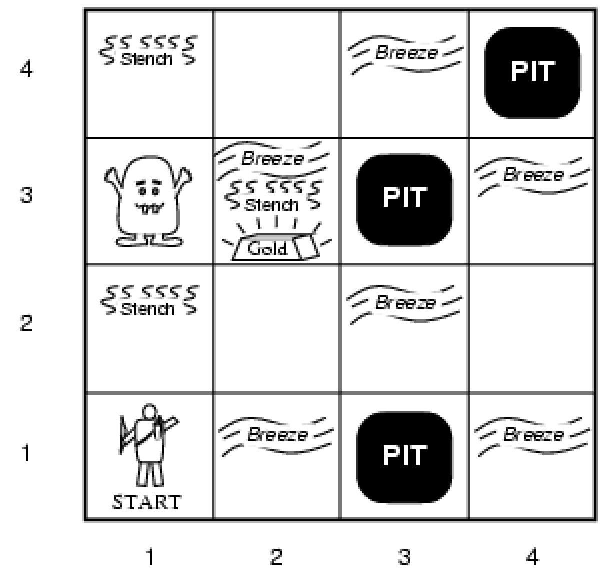
```
function KB-AGENT(percept) returns an action
    static: KB, a knowledge base
            t, a counter, initially 0, indicating time

    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:
  - Represent states, actions, etc.
  - Incorporate new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions

Wumpus World PEAS description

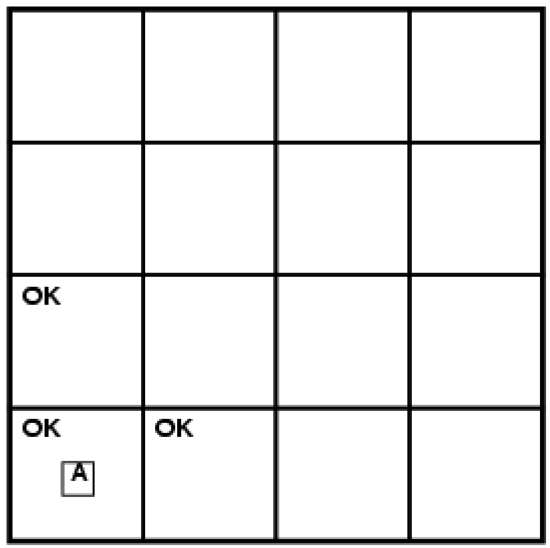
- Performance measure
  - gold +1000, death -1000
  - -1 per step, -10 for using the arrow
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square
  - Releasing drops the gold in same square
  - Sensors: Stench, Breeze, Glitter, Bump, Scream
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot



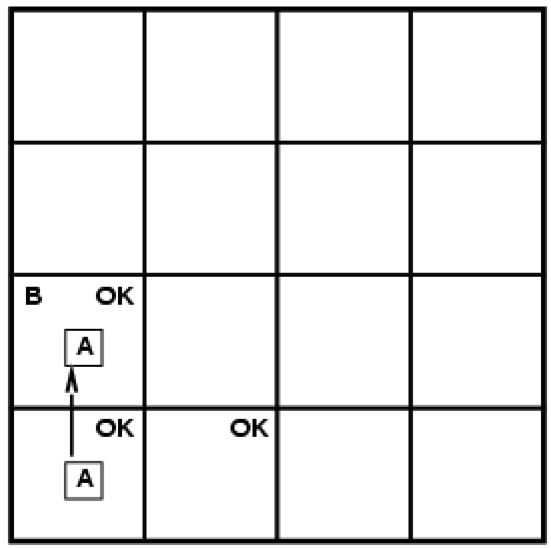
Wumpus world characterization

- Fully Observable No – only local perception
-
- Deterministic Yes – outcomes exactly specified
-
- Episodic No – sequential at the level of actions
-
- Static Yes – Wumpus and Pits do not move
-
- Discrete Yes
-
- Single-agent? Yes – Wumpus is essentially a natural feature
-

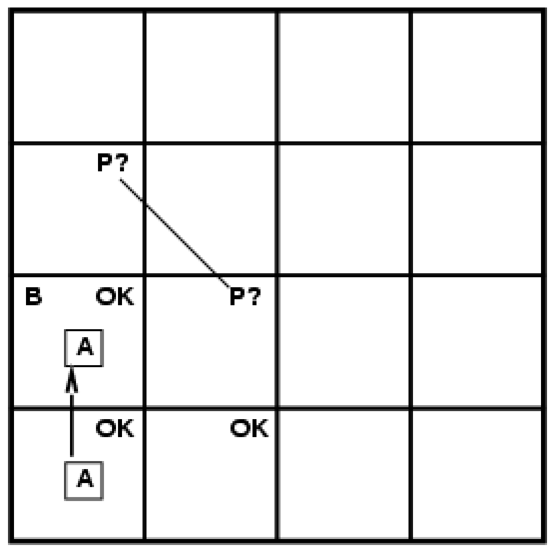
Exploring a wumpus world



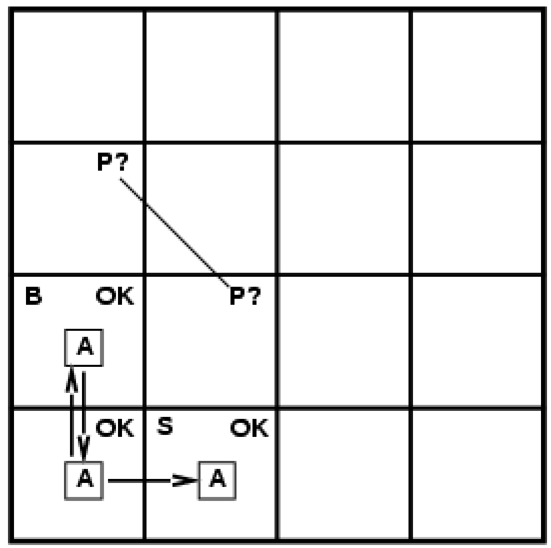
Exploring a wumpus world



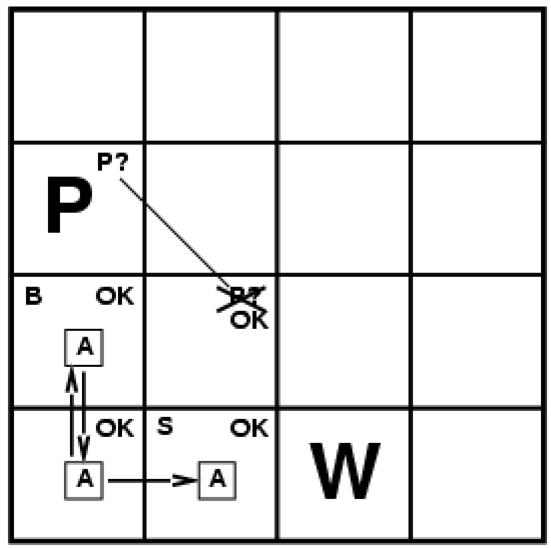
Exploring a wumpus world



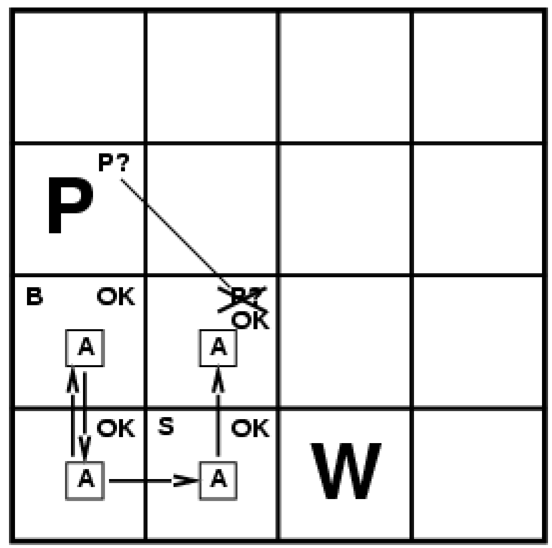
Exploring a wumpus world



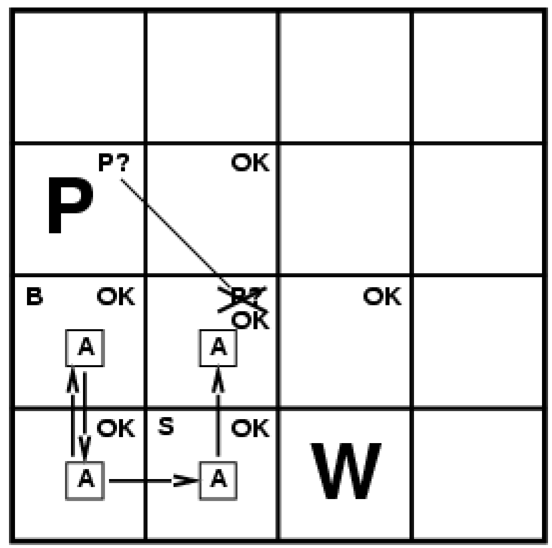
Exploring a wumpus world



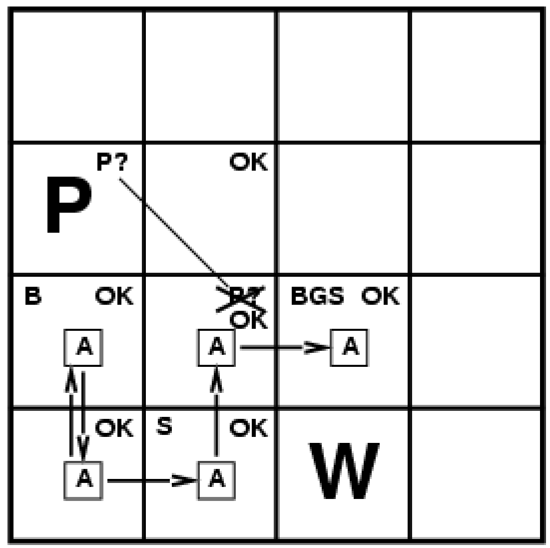
Exploring a wumpus world



Exploring a wumpus world



Exploring a wumpus world



Logic in general

- Logics are formal languages for representing information such 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 $x = 0, y = 6$
  -

Propositional logic

- Propositional logic is the simplest logic – illustrates basic ideas
-
- The proposition symbols P_1, P_2 etc are sentences
  - If S is a sentence, $\neg S$ is a sentence (negation)
  - If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction)
  - If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (disjunction)
  - If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (implication)
  - If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (biconditional)
- How can the “well-formed” sentences be defined?
  - ➔ Transformational grammars

Syntacs – Transformational grammars (TG)

- ‘Colourless green ideas sleep furiously’.
- N. Chomsky constructed finite formal machines – ‘**grammars**’.
- ‘Does the language contain this sentence?’ (intractable) ⇔ ‘Can the grammar create this sentence?’ (can be answered).
- TG are sometimes called *generative grammars.*

- *TG slides are adapted from* Berdnikova&Miretskiy

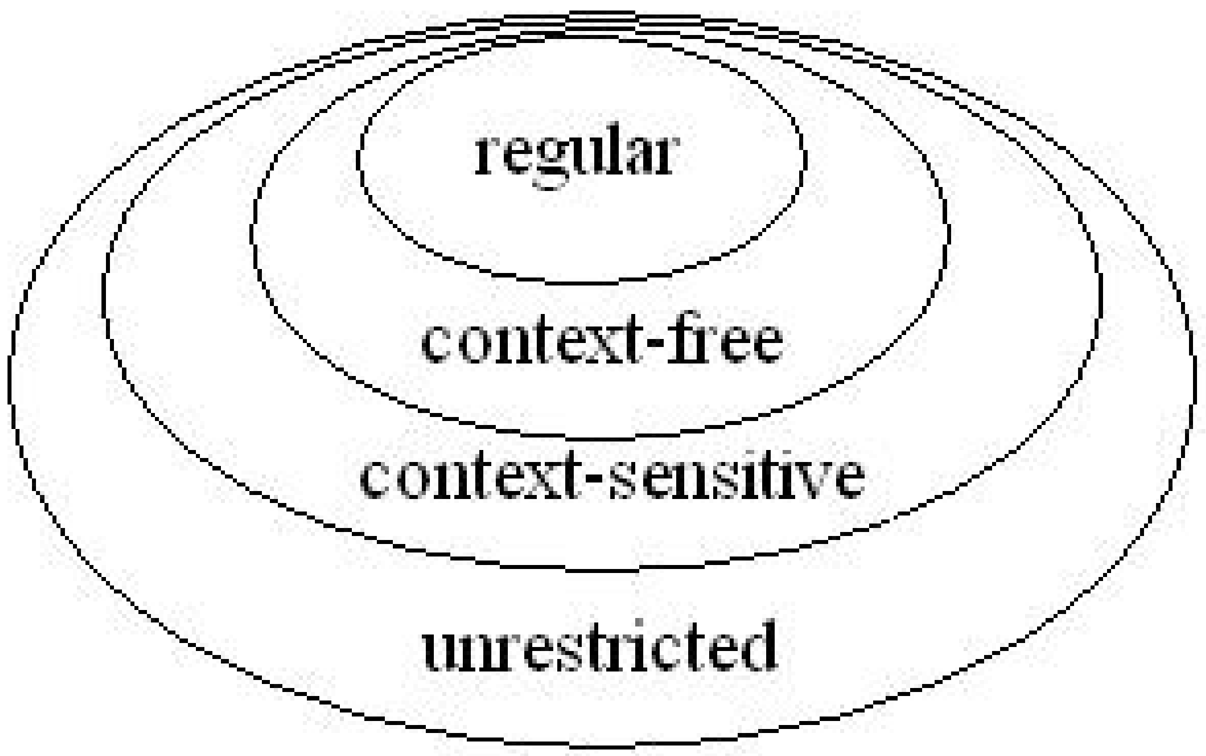
Transformational grammars

- TG = ({*symbols*}, {*rewriting rules* $\alpha \rightarrow \beta$ – *productions*})
- {symbols} = {*nonterminal*} U {*terminal*}
- α contains at least one nonterminal, β – terminals and/or nonterminals.
- $S \rightarrow aS$, $S \rightarrow bS$, $S \rightarrow e$ ($S \rightarrow aS \mid bS \mid e$)
- *Derivation*: $S => aS => abS => abbS => abb$.
- Parse tree: root – start nonterminal S, leaves – the terminal symbols in the sequence, internal nodes are nonterminals.
- The children of an internal node are the productions of it.

The Chomsky hierarchy

- *W – nonterminal, a – terminal, α and γ – strings of nonterminals and/or terminals including the null string, β – the same not including the null string*.
- **regular grammars**:
  - $W \rightarrow aW$ or $W \rightarrow a$
- **context-free grammars**:
  - $W \rightarrow \beta$
- **context-sensitive grammars**:
  - $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$. $AB \rightarrow BA$
- **unrestricted (phase structure) grammars**:
  - $\alpha_1 W \alpha_2 \rightarrow \gamma$

The Chomsky hierarchy



Automata

- Each grammar has a corresponding **abstract computational device** – *automaton.*
- *Grammars*: generative models, *automata:* parsers that accept or reject a given sequence.
- – automata are often more easy to describe and understand than their equivalent grammars.

  – automata give a more concrete idea of how we might recognise a sequence using a formal grammar.

On truth: entailment

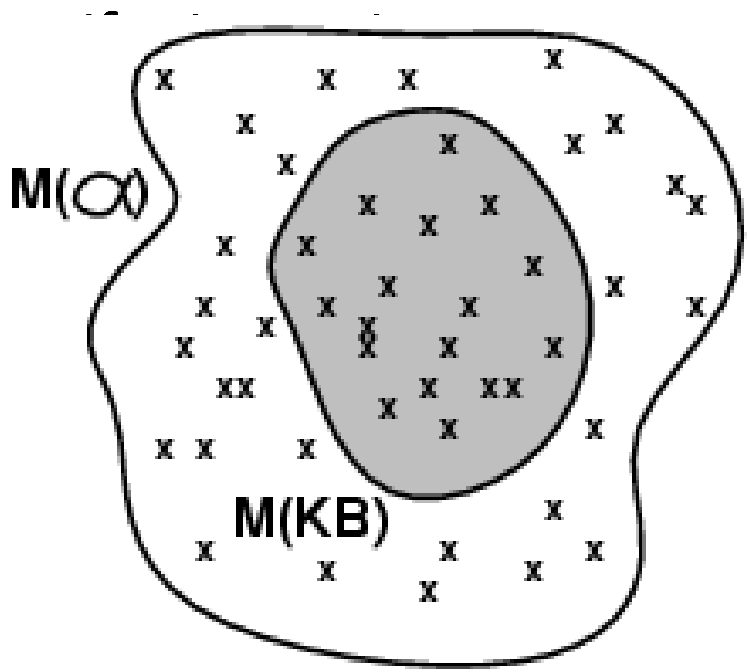
- Entailment means that one thing follows from another:
-

$$KB \models \alpha$$

- Knowledge base *KB* entails sentence α if and only if α is true in all worlds where *KB* is true
    - E.g., the KB containing "the Giants won" and "the Reds won" entails "Either the Giants won or the Reds won"
    -
    - E.g., x+y = 4 entails 4 = x+y
    -
    - Entailment is a relationship between sentences (i.e., syntax) that is based on semantics

Models

- Logicians typically think in terms of models, which are formally structured worlds with respect to which truth can be evaluated
- We say *m* is a model of a sentence
- *M(α)* is the set of all models of α
- Then KB $\models \alpha$ iff $M(KB) \subseteq M(\alpha)$
  - E.g. *KB* = Giants won and Reds won α = Giants won



Propositional logic: Semantics

Each model specifies true/false for each proposition symbol

E.g.	$P_{1,2}$	$P_{2,2}$	$P_{3,1}$
	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
i.e.,	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.,

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

Truth tables for connectives

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

Logical equivalence

- Two sentences are logically equivalent} iff true in same models: $\alpha \equiv ß$ iff $\alpha \models \beta$ and $\beta \models \alpha$
-
-

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

Truthtable method: an example

„Adam, Betty, and Chris played and a window got broken.
Adam says: ‘Betty made, Chris is innocent.'
Betty says: ‘If Adam is guilty, then Chris too'.
Chris says: ‘I am innocent; someone else did it'.“

1, Consistency?
2, Who lies?
3, Who is guilty?

Truthtable method: formalization

Propositional symbols:

A: Adam is not guilty (innocent).

B: Betty is not guilty (innocent).

C: Chris is not guilty (innocent).

Statements:

SA: $\neg B \wedge C$

SB: $\neg A \rightarrow \neg C$

SC: $C \wedge (\neg B \vee \neg A)$

A	B	C	SA	SB	SC	SA ^ SB ^ SC
F	F	F	F	T	F	F
F	F	T	T	F	T	F
F	T	F	F	T	F	F
F	T	T	F	F	T	F
T	F	F	F	T	F	F
T	F	T	T	T	T	T (1)(3)
T	T	F	F	T	F	F
T	T	T	F	T	F	F (2)

(1) There is a combination that all of them tells the truth.
(2) If they are not guilty, then Adam and Betty lied.
(3) If they told the truth, then Betty is guilty.

Propositional symbols:

A: Adam is not guilty (innocent).
B: Betty is not guilty (innocent).
C: Chris is not guilty (innocent).

Statements:

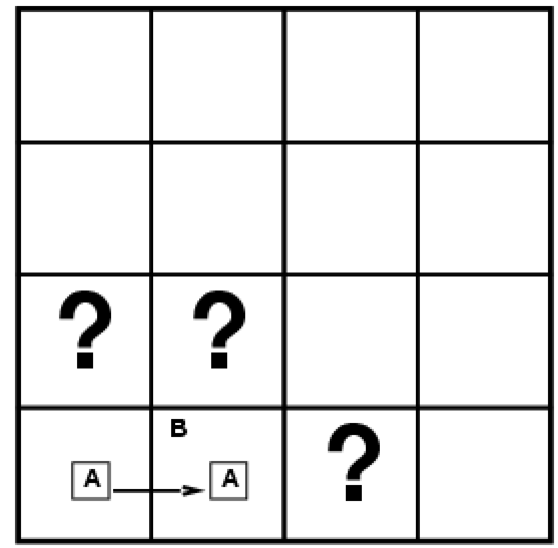
SA: $\neg B \wedge C$
SB: $\neg A \rightarrow \neg C$
SC: $C \wedge (\neg B \vee \neg A)$

Entailment in the wumpus world

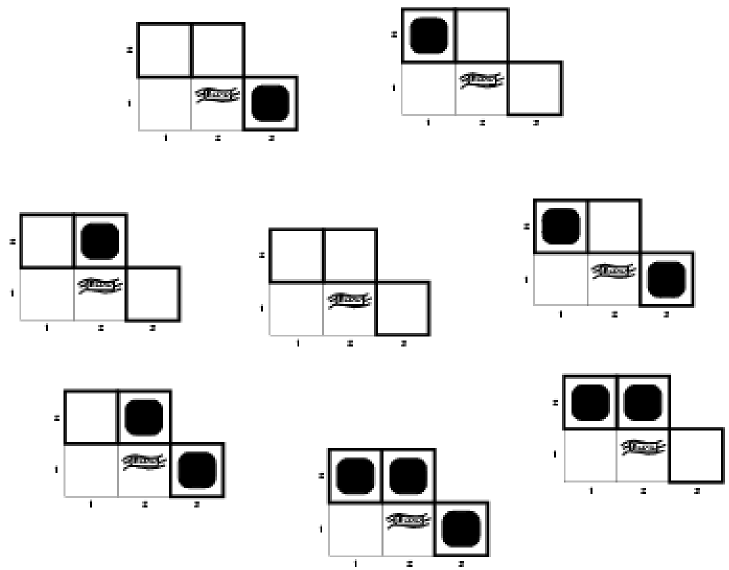
Situation after detecting nothing in [1,1], moving right, breeze in [2,1]

Consider possible models for *KB* assuming only pits

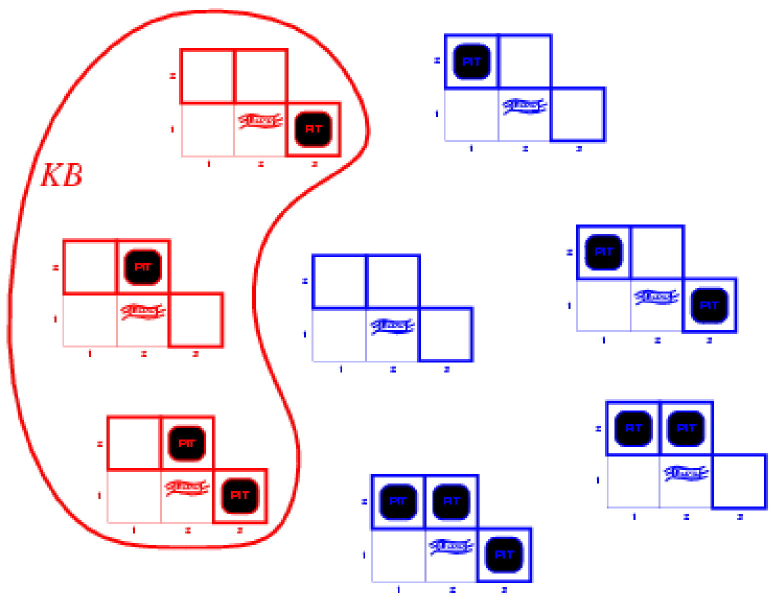
3 Boolean choices $\Rightarrow$ 8 possible models



Wumpus models

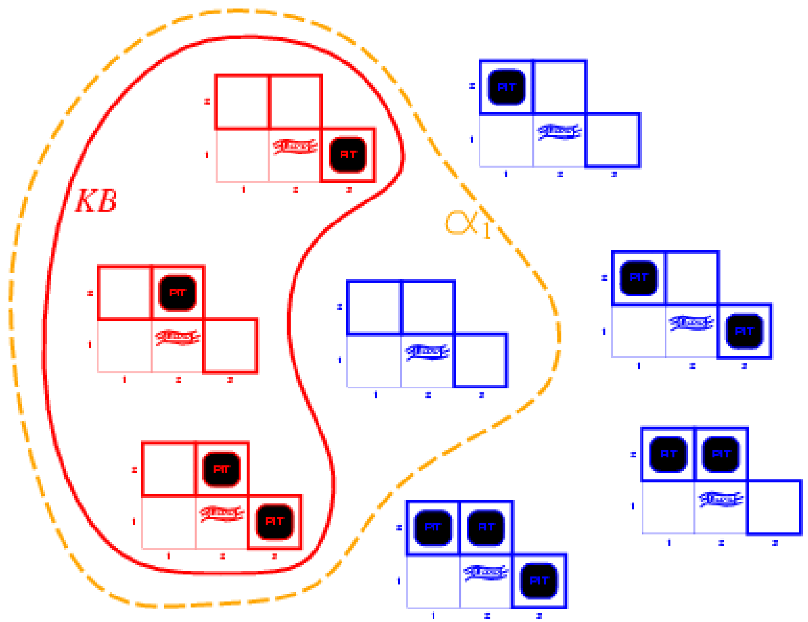


Wumpus models



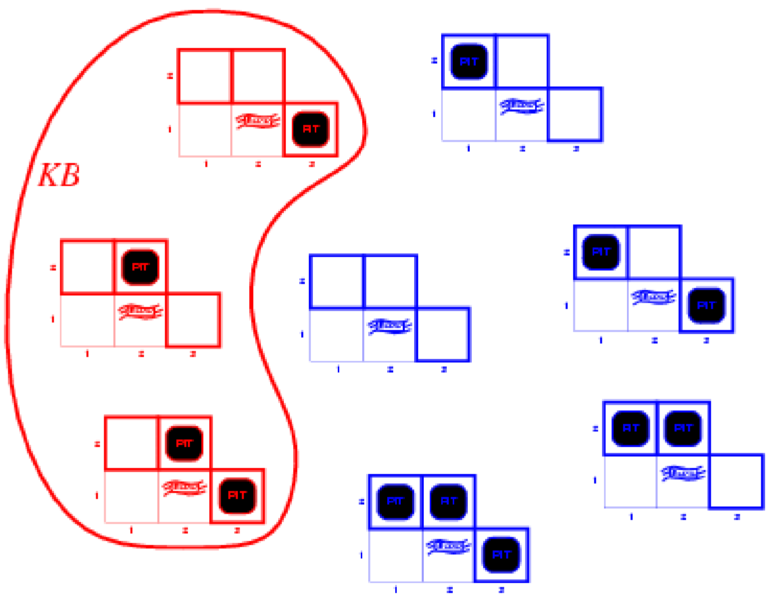
- KB = wumpus-world rules + observations
-

Wumpus models



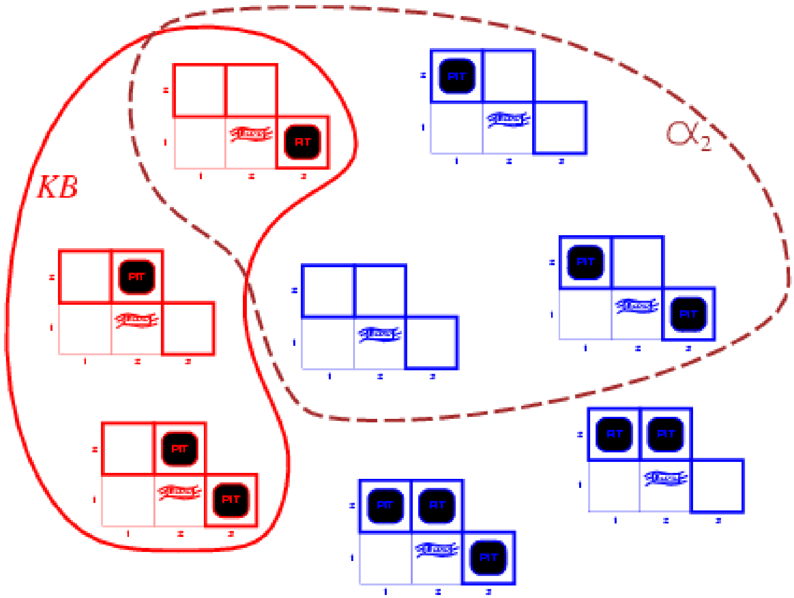
- KB = wumpus-world rules + observations
- α_1 = "[1,2] is safe", $KB \models \alpha_1$, proved by model checking

Wumpus models



- KB = wumpus-world rules + observations

Wumpus models



- KB = wumpus-world rules + observations
- α_2 = "[2,2] is safe", $KB \models \alpha_2$
-

Validity and satisfiability

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

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

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

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

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

Wumpus world sentences

Let $P_{i,j}$ be true if there is a pit in [i, j].
Let $B_{i,j}$ be true if there is a breeze in [i, j].

$\neg P_{1,1}$
$\neg B_{1,1}$
$B_{2,1}$

- "Pits cause breezes in adjacent squares"
- $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
  $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

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 enumeration

- 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))
```

- For *n* symbols, time complexity is $O(2^n)$, space complexity is $O(n)$
-

On proof

- $KB \vdash_i \alpha$ = sentence α can be derived from KB by procedure i
- Inference methods divide into (roughly) two kinds:
  - Application of inference rules
    - Legitimate (sound) generation of new sentences from old
    - Proof = a sequence of inference rule applications
      Can use inference rules as operators in a standard search algorithm
      - E.g. Modus Ponens, Modus Tollens, resolution
      - Typically require transformation of sentences into a normal form, e.g. into Conjunctive Normal Form (CNF)
  - Model checking
    - truth table enumeration (always exponential in n)
    - improved backtracking, e.g., Davis--Putnam-Logemann-Loveland
    - heuristic search in model space (sound but incomplete)
      e.g., min-conflicts-like hill-climbing algorithms

On truth and proof

- Soundness: *i* is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$
-
- Completeness: *i* is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$
-
- Preview: we will define a logic (first-order logic) which is expressive enough to say almost anything of interest, and for which there exists a sound and complete inference procedure.
-
- That is, the procedure will answer any question whose answer follows from what is known by the *KB*.
-

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - syntax: formal structure of sentences
  - semantics: truth of sentences wrt models
  - entailment: necessary truth of one sentence given another
  - inference: deriving sentences from other sentences
  - soundness: derivations produce only entailed sentences
  - completeness: derivations can produce all entailed sentences
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Propositional logic lacks expressive power

- Suggested reading:
  - A.Tarski:Truth and Proof, 1969
    - http://people.scs.carleton.ca/~bertossi/logic/material/tarski.pdf
  - Interview with Douglas R. Hofstadter
    - http://www.americanscientist.org/bookshelf/pub/douglas-r-hofstadter
  - D.R.Hofstadter: Gödel, Escher, Bach, 1979