Introduction to Logical Agents

RN, Chapter 7-7.3

Logical Agents

- Reasoning [Ch 6]
  - Why represent world? (“state information”)
  - Wumpus World
  - Which formalism? (… logic)
- Propositional Logic [Ch 7]
- Predicate Calculus
  - Representation [Ch 8]
  - Inference [Ch 9]
- Implemented Systems [Ch 10]
- Applications [Ch 8.4,10]
- Planning [Ch 11]

The story so far. . .

- Simple situation:
  - agent has to deal with SINGLE goal
    (Eg, get to B; clean house; …)
  - agent (designer) has accurate model of world
  - agent can determine its state by sensing
  - agent's actions are deterministic
  - world does not change while agent is thinking
  - ...

⇒ Designer only needs one “program“
(Eg, heuristic function, …)

Agent does not need internal model of …

- world, state, task (goal)

Here: simple **Search-based Agents** are adequate

Many situations require more. . .

- World not always accessible

  ie, cannot simple “read off" state ... “perceptual aliasing"

  - **Reflex Agents**: Keep no state information

    ⇒ Can't solve such problems

  - **State Tracking Agents**:

    Maintain state as a single data structure

    In ambiguous situations: must use *set* of all consistent states

    [Eg... unknown start state, in Vacuum World]

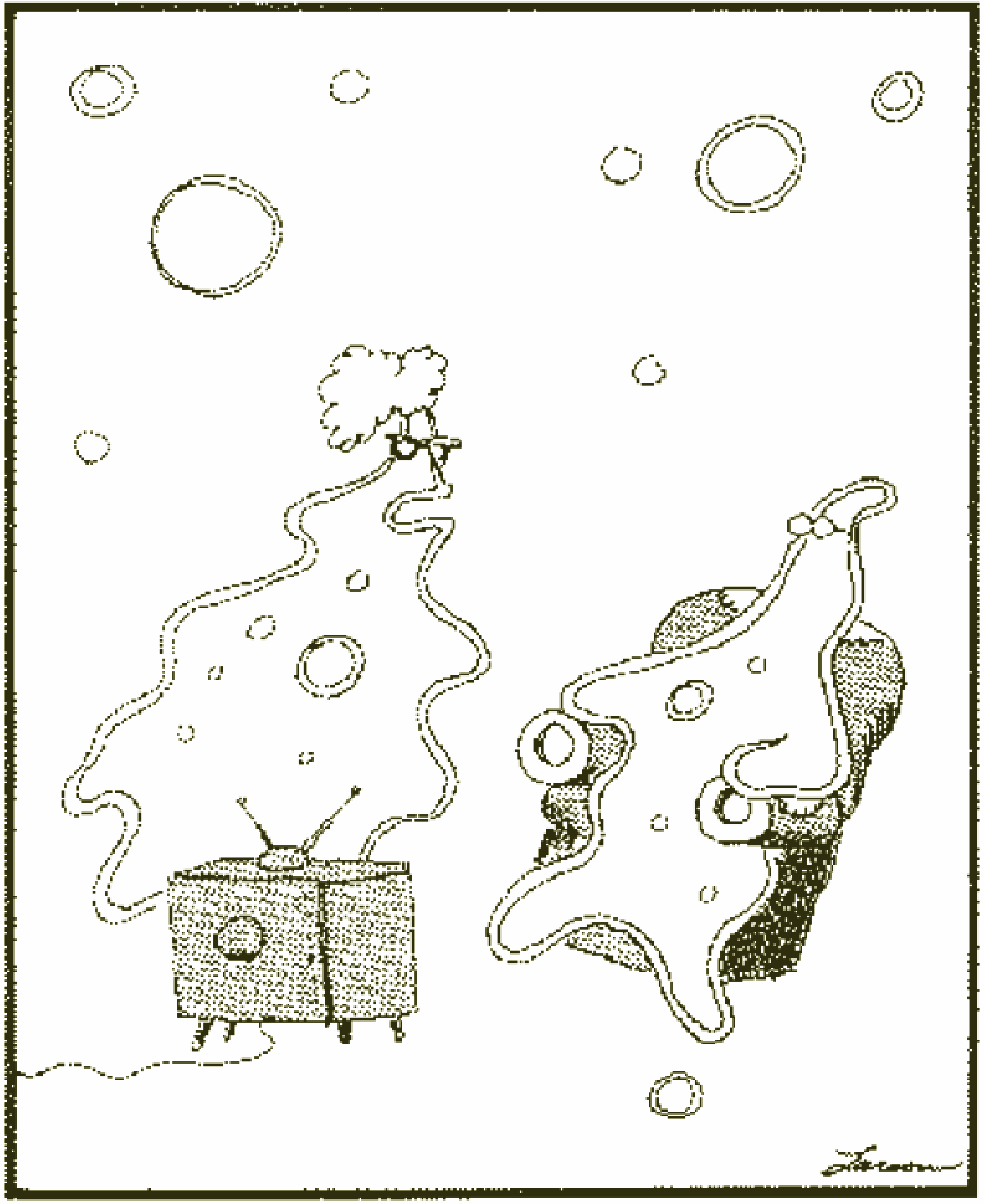
  ⇒ *Neither method scales up* . . .

  - ... Need (laconic) internal model to help determine best action

- Have diverse “goals”; diverse “worlds”
  - (TAXI: different destinations, different traffic patterns)

⇒ Need easy way to change agents

(Rather than “re-programming” each time)



Stimulus, response!
Stimulus, response!
Don’t you ever think?

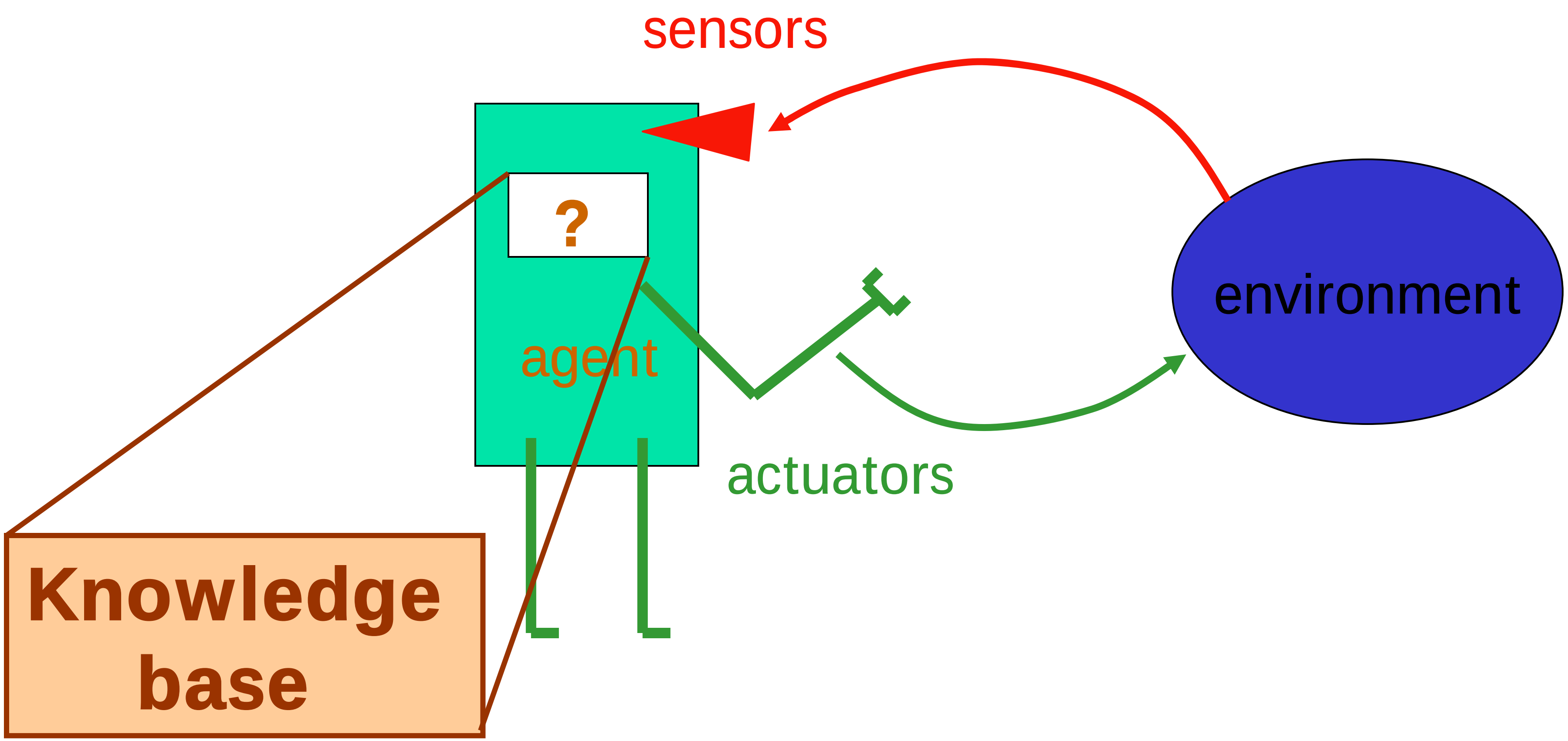
Knowledge-based Approach

- To be effective, agent may need to know
  - current state of the world
  - unseen properties of world
  - how world evolves
  - what it wants to achieve
  - what its actions do in various situations

⇒ Need to go beyond simple “Search-based Agents”

- Current Focus:
  - Deterministic
  - Discrete
  - World is Known (but state may not be)
  - Static (initially)
- Basic search techniques still used
  but perhaps wrt other “spaces"

Knowledge-Based Agent



Types of Knowledge

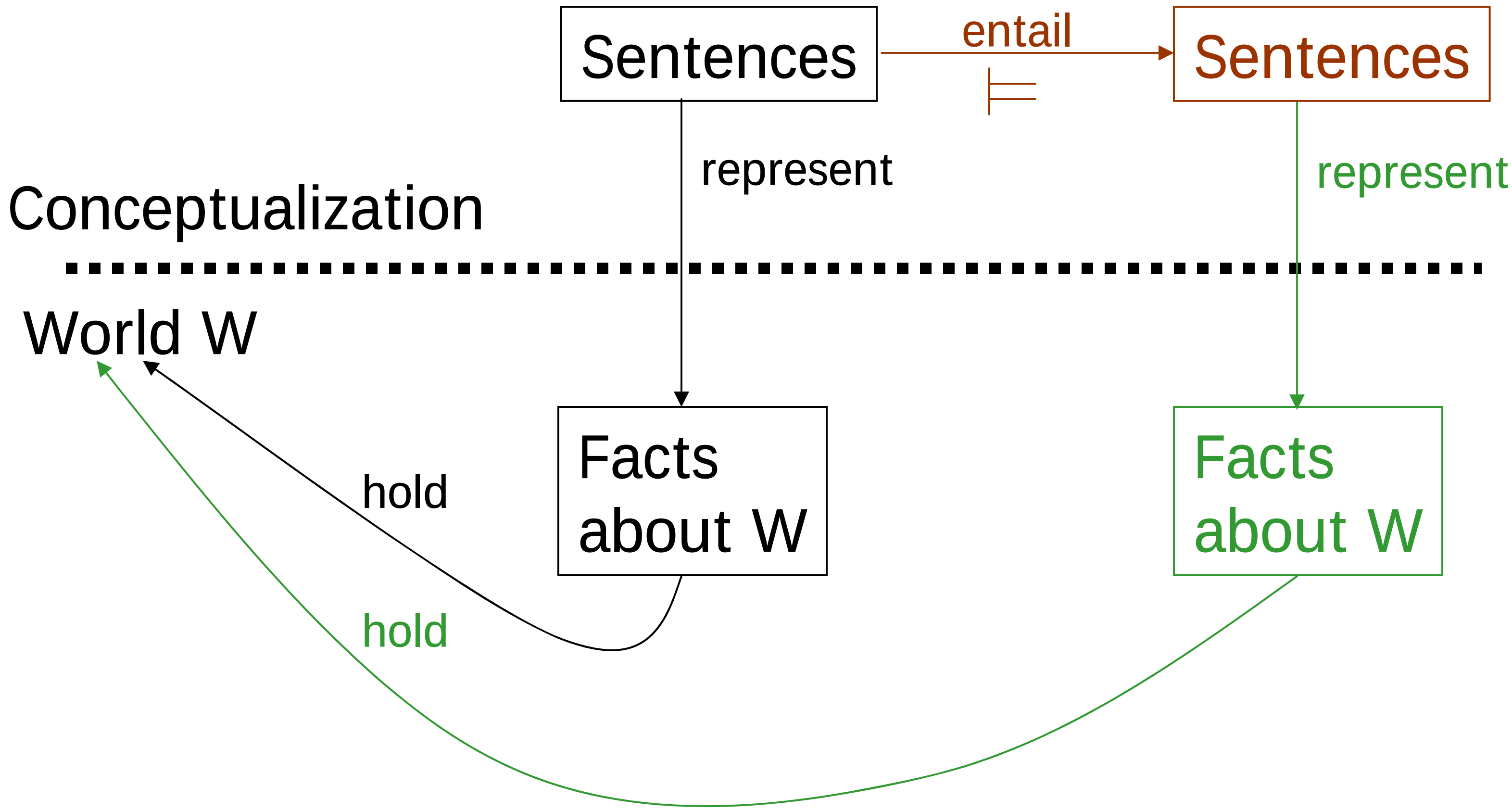
- Procedural
  - e.g. functions
  - Such knowledge can only be used in one way: By executing it
- Declarative
  - e.g. constraints
  - Such knowledge can be used to perform many different sorts of inferences

Logic

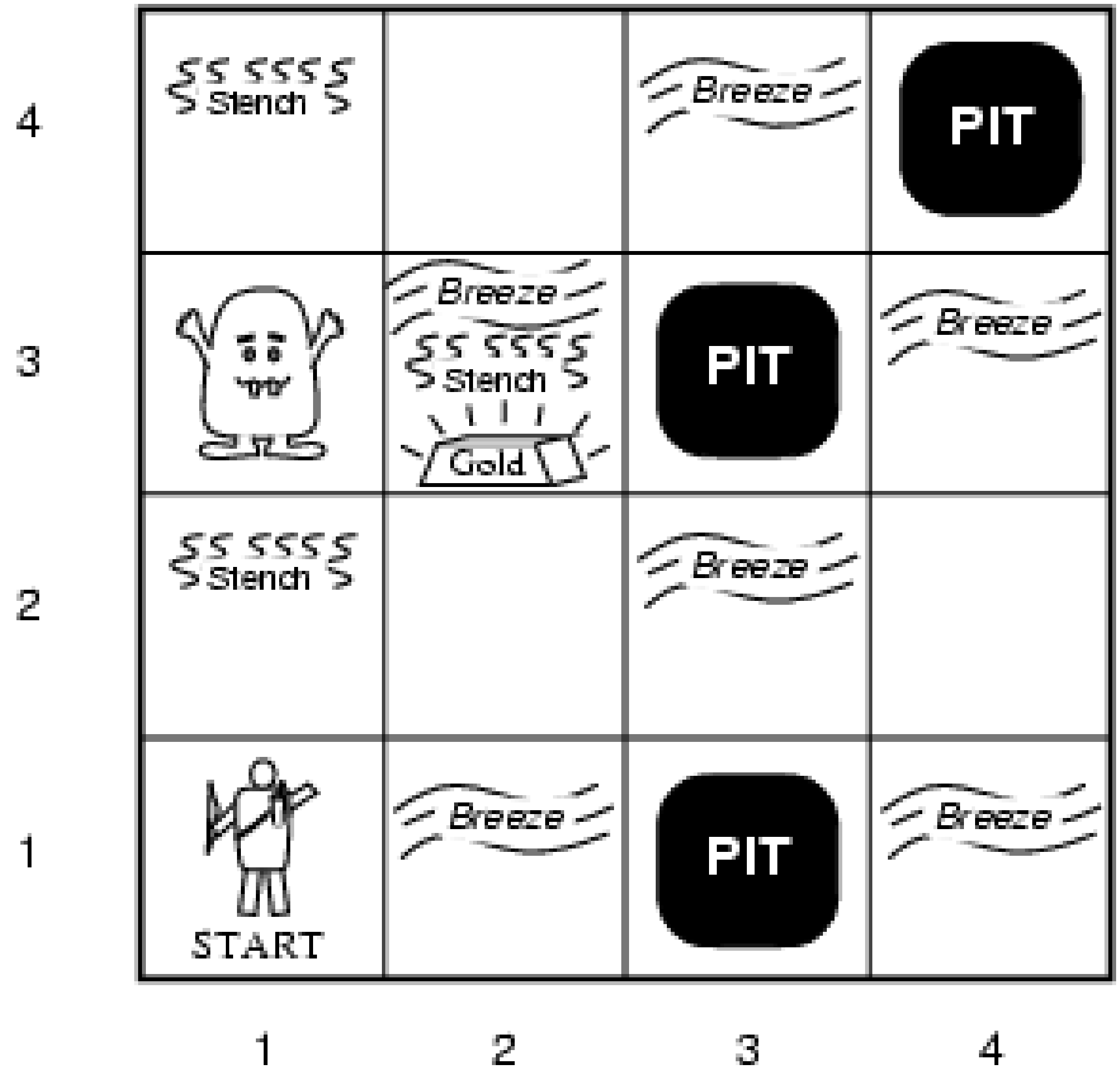
Logic is a declarative language to:

- Assert sentences representing facts that hold in a world W
  (these sentences are given the value true)
- Deduce the true/false values of sentences representing other aspects of W

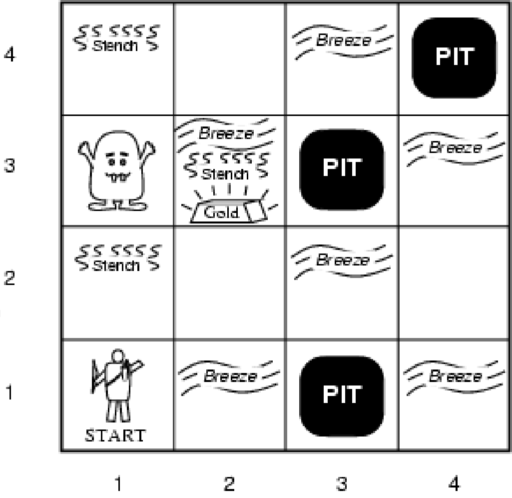
Connect World-Representation



Wumpus World!

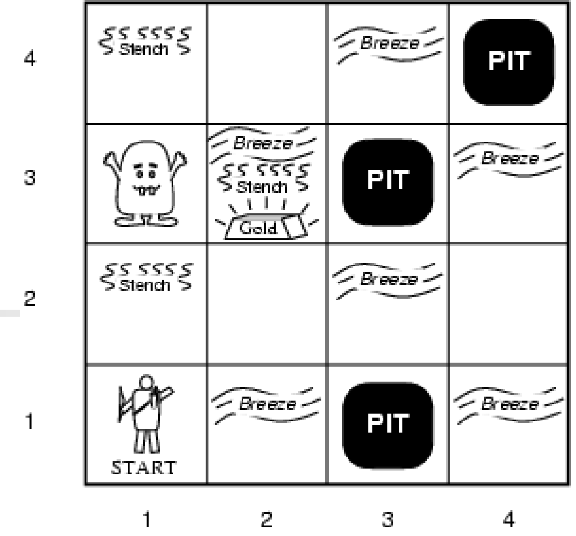


PEAS Description



- **Performance measure**
  - gold +1000, death -1000
  - -1 per step, -10 for using the arrow
- **Environment**
  - Squares adjacent to wumpus are stench-y
  - Squares adjacent to pit are breeze-y
  - 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
- **Actuators**:
  Left_turn, Right_turn, Forward, Grab, Release, Shoot
- **Sensors**: Stench, Breeze, Glitter, Bump, Scream

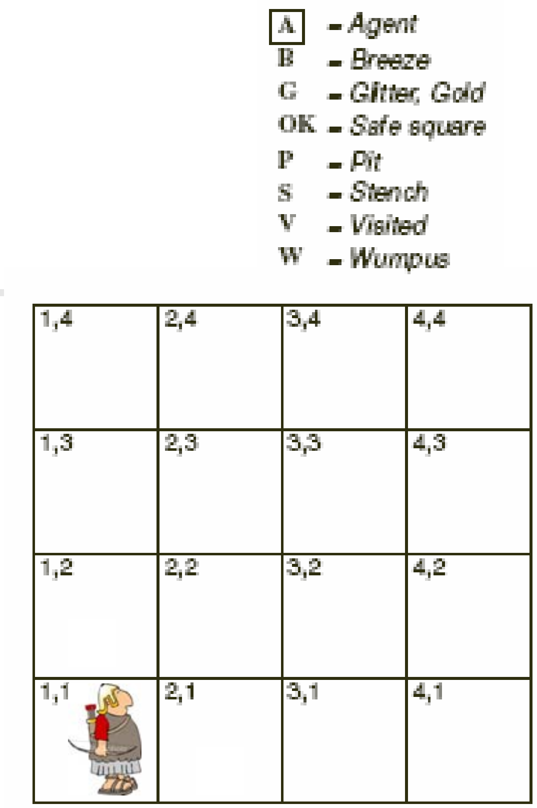
Characterizing Wumpus World



Is the world …

…deterministic?	**Yes** Outcomes specified exactly
… fully *accessible*?	**No** Only local perception
… static?	**Yes** Wumpus, Pits do not move
… discrete?	**Yes**

Acting + Reasoning in Wumpus World



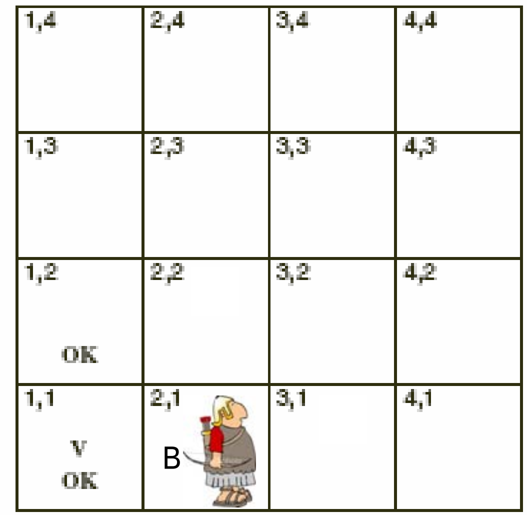
- Location: [1,1]
- Sense:

  [-Stench, -Breeze, -Glitter, -Bump, -Scream]
- Can Reason. . .
  - As -Stench, Wumpus $\notin$ { [1,2], [2,1] }
  - As -Breeze, Pit $\notin$ { [1,2], [2,1] }
- **Conclude**: [1,2] is “safe"; [2,1] is “safe"

  $\Rightarrow$ Action = Forward (to [2,1])

Acting + Reasoning, #2

A – Agent
B – Breeze
G – Glitter, Gold
OK – Safe square
P – Pit
S – Stench
V – Visited
W – Wumpus



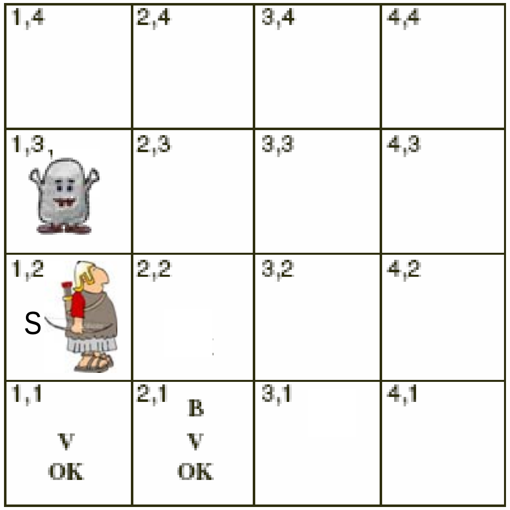
- Location: [2,1]
- Sense:
  [-Stench, +Breeze, -Glitter,
  -Bump, -Scream]
- Can Reason. . .
  - As -Stench, Wumpus $\notin$ { [1,1], [3,1], [2,2] }
  - As +Breeze, Pit $\in$ { [1,1], [3,1], [2,2] }

  Note Pit NOT in [1,1]: agent was there, did NOT fall in

$\Rightarrow$ Only GUARANTEED safe move is...

Action = "Return to [1,1]"
(Turn_Left, Turn_Left, Forward)

Acting + Reasoning, #3



- Location: [1,2]
- Sense:
  [+Stench, +Breeze,
  -Glitter, -Bump, -Scream]
- Can Reason. . .
  - As +Stench, Wumpus $\in$ { [1,1], [1,3], [2,2] }
  - As -Breeze, Pit $\notin$ { [1,1], [1,3], [2,2] }

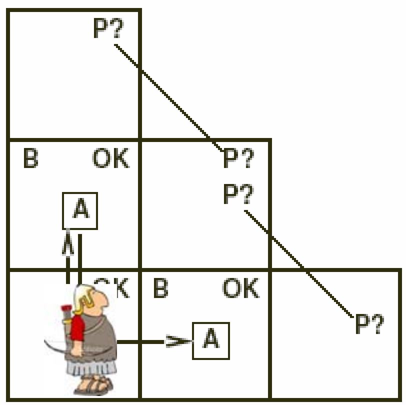
  Note Wumpus NOT in [1,1]: agent was there, not eaten
  Wumpus NOT in [2,2]: else +Stench in [2,1]

  $\Rightarrow$ Wumpus is in [1,3] !

  $\Rightarrow$ Only unvisited adjacent OK square = [2,2]

$\Rightarrow$ Action = “Go to [2,2]" (Turn Right, Forward)

Other Interesting Situations

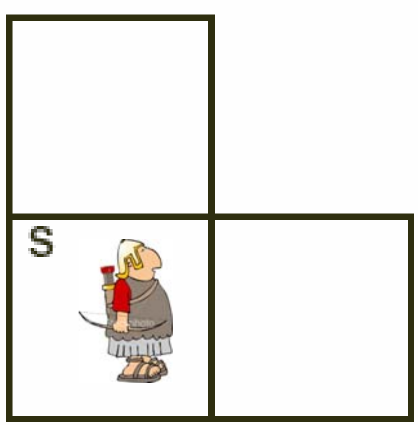


- Breeze in [1, 2], [2, 1]

  ⇒ no safe actions

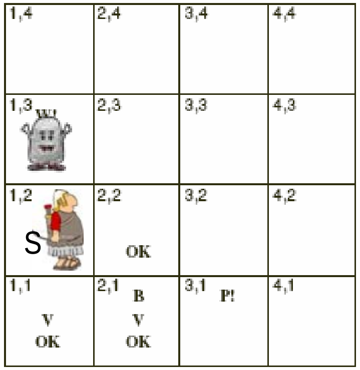
⇒ Assuming pits uniformly distributed...

Pit most likely in [2, 2]



- Smell in [1, 1]
  - ⇒ cannot move
- Can use coercion strategy:
  - shoot straight ahead
  - Wumpus was [1,2] ⇒ dead ⇒ safe
  - Wumpus wasn't there ⇒ in [2,1] ... safe

Challenges



- Need to encode what is known / observed
  + Partial information
    - Don't know where Wumpus is exactly, but constrained to {… }
  + Obtained at different times, from different locations
- … use it to reach appropriate conclusions
  - Only correct conclusions (sound)
  - All correct conclusions (complete)

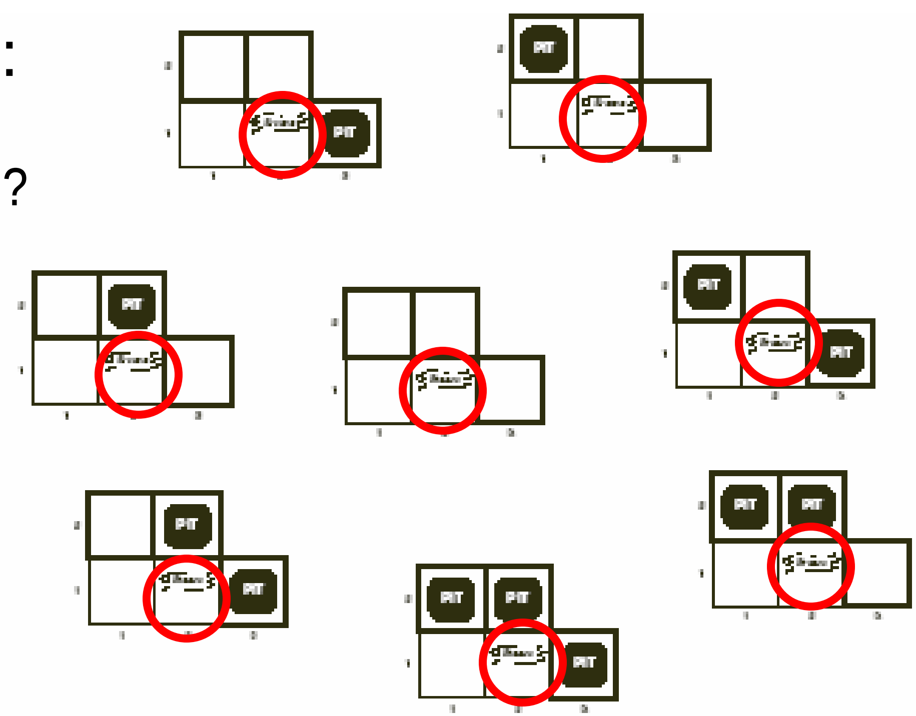
As +Stench @ [1,2],
  ⇒ Wumpus ∈ { [1,1], [1,3], [2,2] }
As agent was in [1,1], but not eaten
  ⇒ Wumpus NOT in [1,1]
As -Stench in [2,1]
  ⇒ Wumpus NOT in [2,2]
⇒ Wumpus is in [1,3] !

- Requires "LOGIC"
  To define what answers SHOULD be returned
  … Models!

Possible Worlds ≡ Models

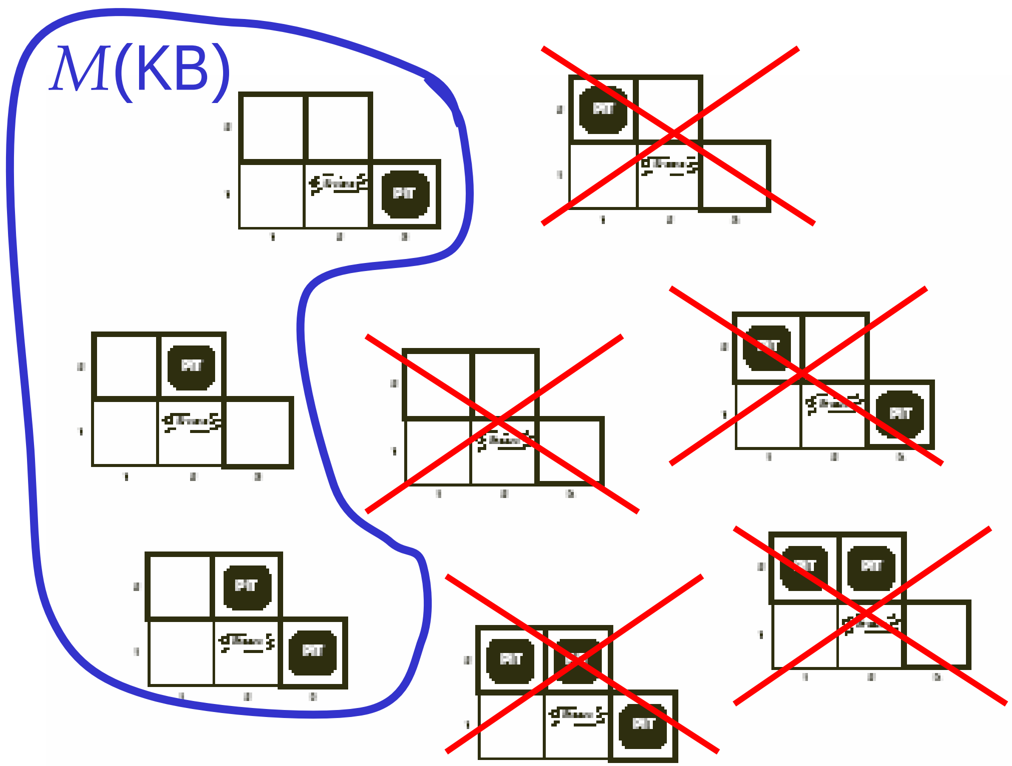
Initially. . .

- "PIT" in *any* subset of 4 x 4 grid
  "BREEZE" in any subset of 4 x 4 grid
  ...
  $\Rightarrow$ so $(2^{|\{P,B,W,G,S,...\}|})^{16}$ possible worlds
- FOCUS: on 8 of those worlds:
  - ∃ a BREEZE in [2, 1]
  - ∃? PIT in [1, 2], [2, 2], [3, 1] ?



Possible Worlds (II)

- KB ≡
  - Facts about WumpusWorld
  - Nothing in [1,1]
  - Breeze in [2,1]

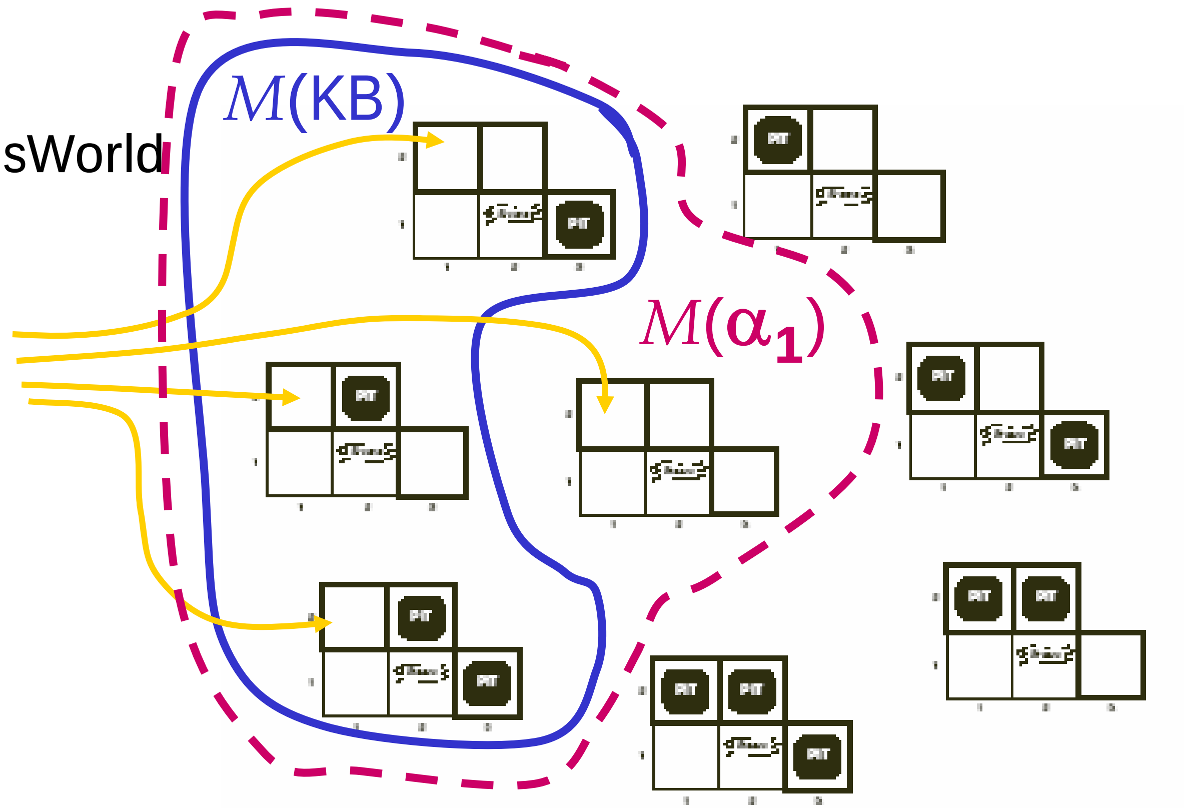


If you believe KB,
then real world ∈ M(KB)

- A world does NOT match KB iff
  - ∃ PIT not next to a BREEZE v ∃ BREEZE not next to a PIT
- 3 remaining possible worlds: M(KB)

Possible Worlds (III)

- KB ≡
  - Facts about WumpusWorld
  - Nothing in [1,1]
  - Breeze in [2,1]
- α_1 ≡ No pit in [1,2]



- $M(\text{KB}) \subseteq M(\alpha_1)$

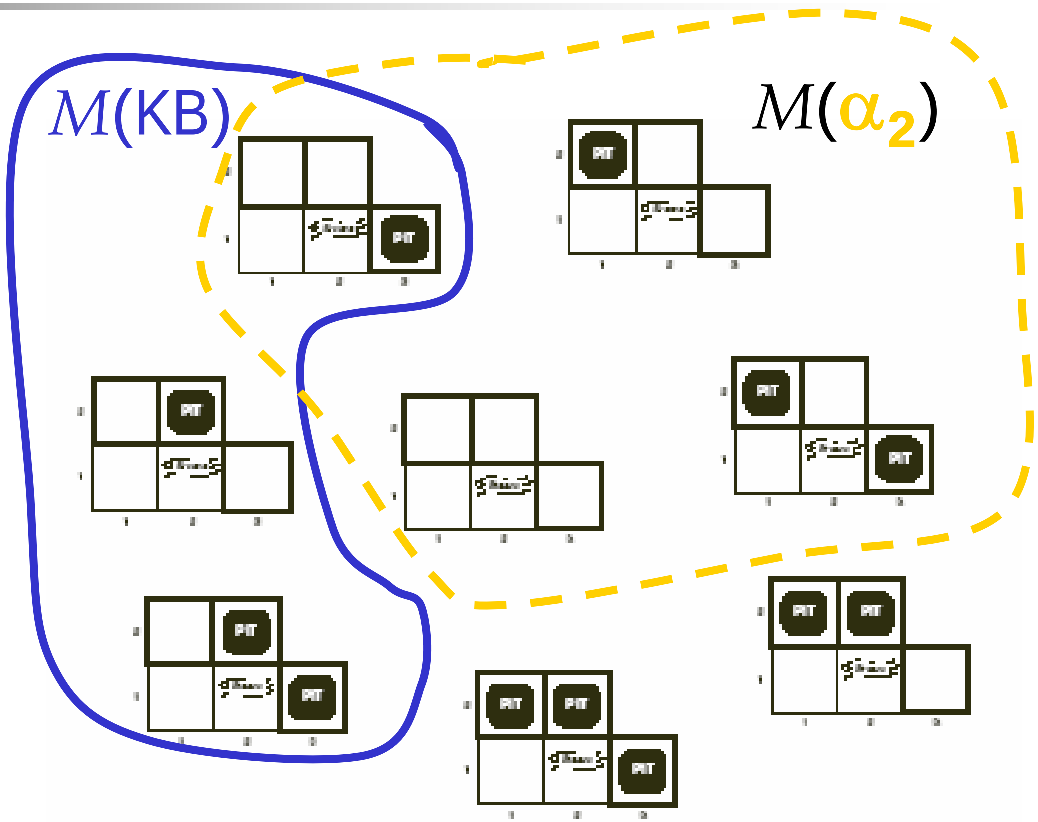
$\Rightarrow$ α_1 holds whenever KB holds!

$\Rightarrow$ KB $\models \alpha_1$

Possible Worlds (IV)

- KB ≡
  - Facts about WumpusWorld
  - Nothing in [1,1]
  - Breeze in [2,1]
- α_2 ≡ No pit in [2,2]

- $M(\text{KB}) \not\subseteq M(\alpha_2)$



$\Rightarrow$ α_2 does NOT have to holds whenever KB holds!

$\Rightarrow$ $\text{KB} \not\models \alpha_2$

Semantics: What HAS to hold

- Given *KB*, does α have to hold?

  $KB \models^? \alpha$

- Enumerate ALL possible worlds (models)
- Use information in KB to RULE out "impossible" worlds
  Let $\mathcal{M}$(KB) = remaining models
- $KB \models^? \alpha$ whenever

  α holds in ALL models of KB – $\mathcal{M}(KB) \subseteq \mathcal{M}(\alpha)$

- Why?
  - Only $\mathcal{M}$(KB) are still possible
  - α holds in each of $\mathcal{M}$(KB)

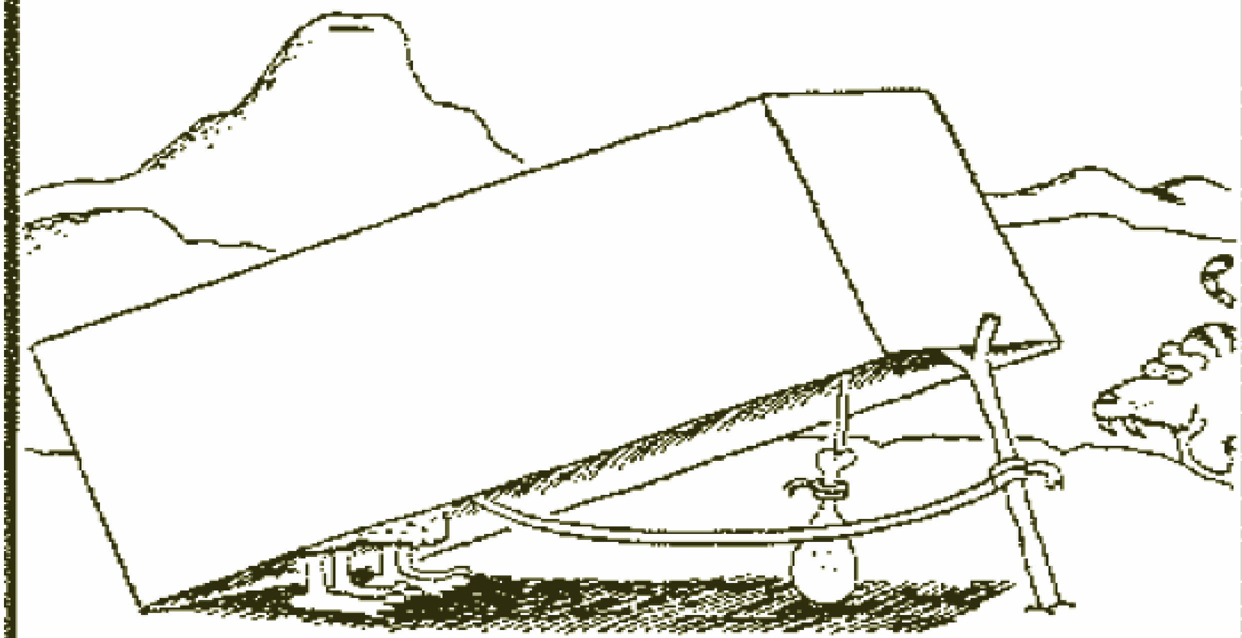
  $\Rightarrow$ So α must hold!

- Suggests an algorithm: "Model Checking" (later)

Logic/Knowledge-Based Approach

- Represent knowledge as
  declarative statements
- Use inference / reasoning mechanism to
  - derive “new" (implicit) information
  - make decisions
- **Key Problem**: Need to express partial knowledge about current state
- **Solution**: Use intensional representation based on formal logic.
  - logical language (propositional / first-order)
  - combined w/ logical inference mechanism

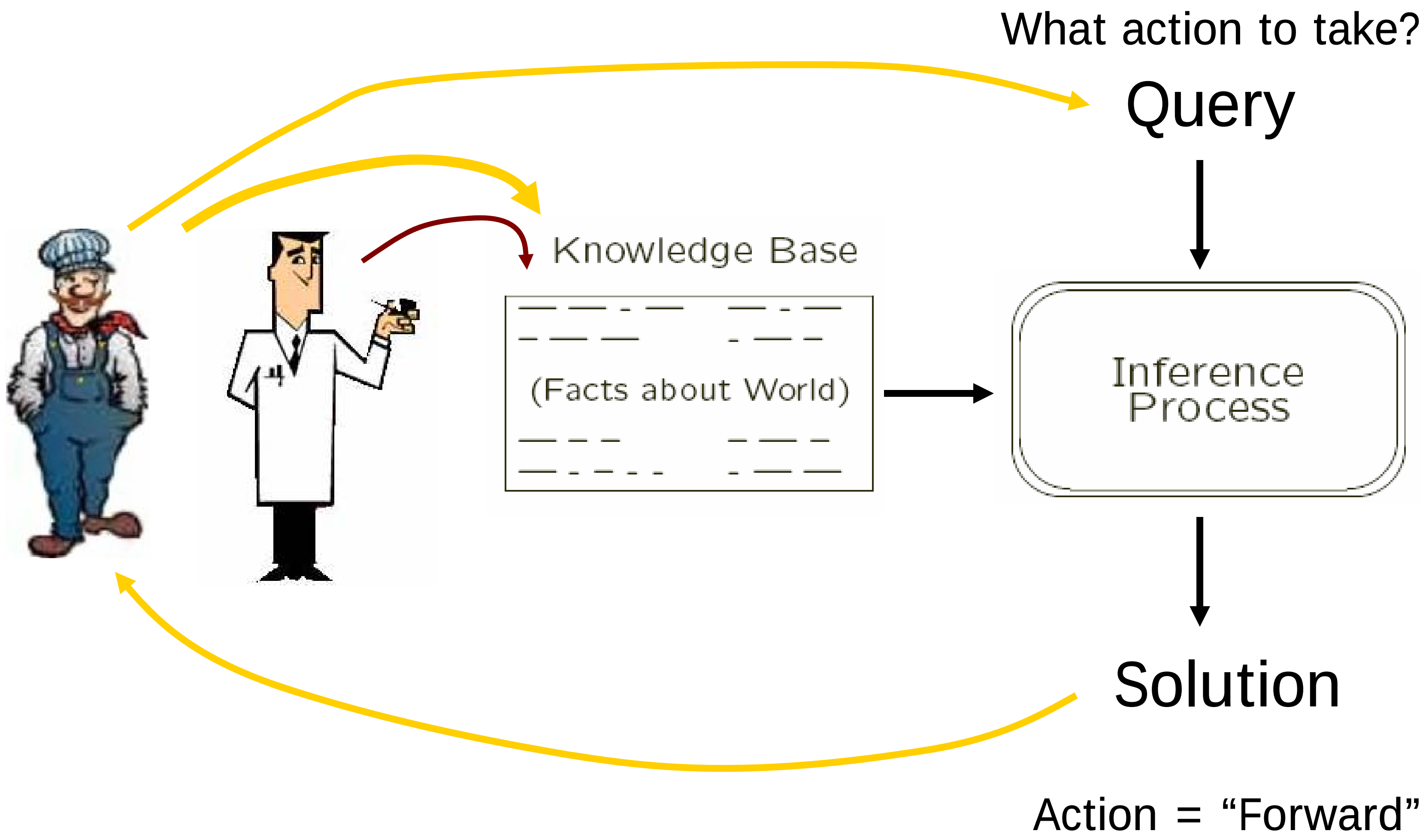
- Close to human thought? -- ??
  - . . . but appears reasonable strategy for machines



Shhh, Zog!
Here come one now!

Larson

"Knowledge-Based" System



"Reasoning" Agents

Knowledge Base (KB) abstract data type

- *Tell(KB, Fact)* records *Fact* into *KB*
- *Ask(KB, Query)* asks whether *Query* is true wrt *KB*

May return information "action" specifying when *Query* is true

```
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
```

Representing World

- Preferably:
  - expressive, concise
  - unambiguous, independent of context
  - have an effective procedure to derive implied (implicit) information
- Not easy meeting these goals . . .

  [Halting Problem; Godel's Incompleteness. . .]

  ... propositional / first-order logic meet some
- Must be able to handle **incompleteness / uncertainty**

- Contrast with
  - programming languages
  - natural language
  - ...

Advantages of Logic-Based Approach

- Simply
  - Store "truths"
  - Ask about other (possible) truths
- Information is
  - Modular
    - Easy to build
    - Easy to modify / extend / debug
    - Capable of Explanation
  - Runnable ... to find other truths
  - Declarative:
    - Use/reuse same information to
    - Laconic
    - Introspective
    - ...

Use/reuse same information to:

- Design circuit
- Simulate circuit
- Explain Circuit
- Diagnose Circuit