

Logical Agents

Logical agents for Wumpus World



We'll use the Wumpus World domain to explore three (non-exclusive) agent architectures:

① Reflex agents

Rules classify situations based on percepts and specify how to react to each possible situation

② Model-based agents

Construct an internal model of their world

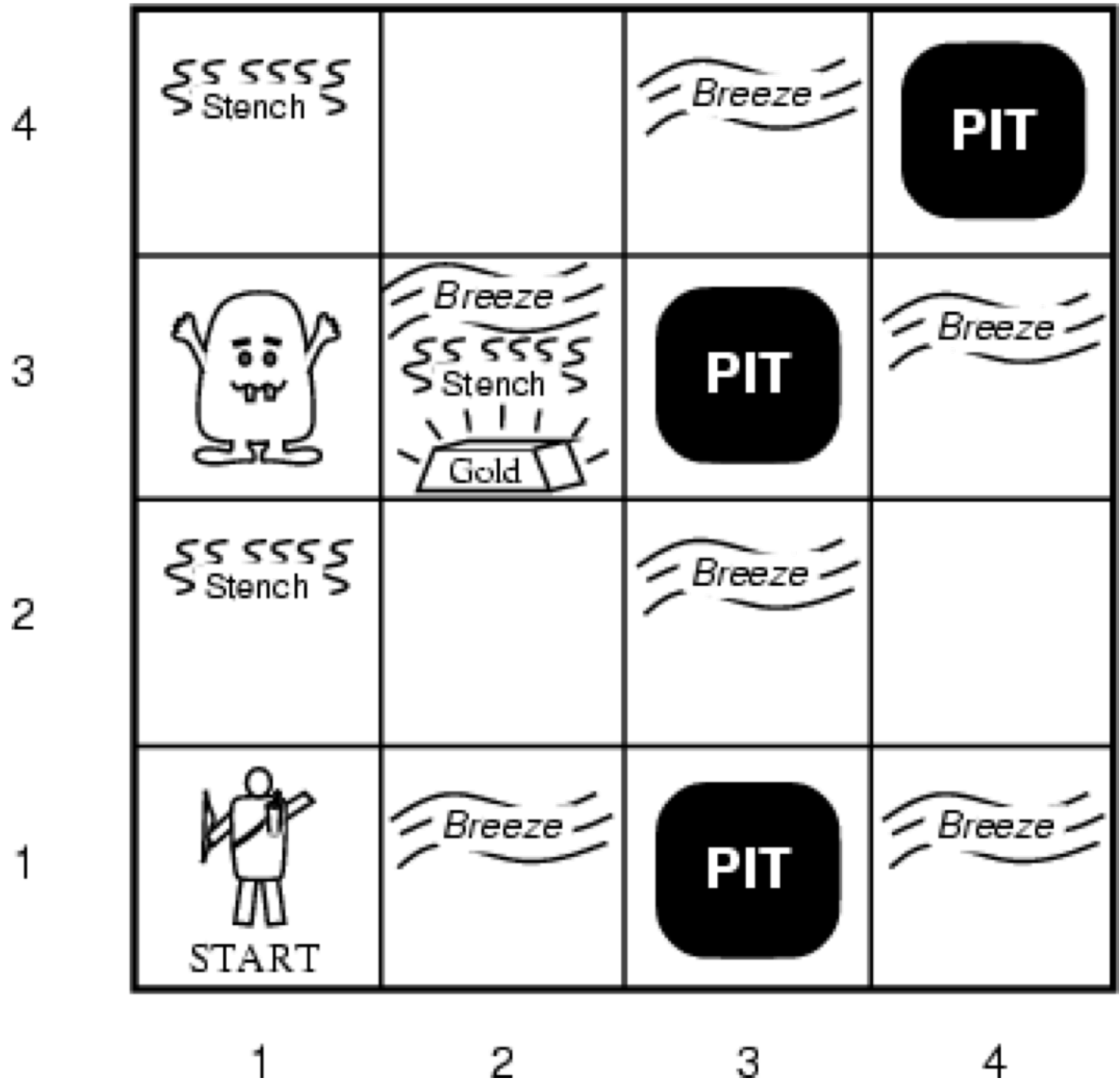
③ Goal-based agents

Form goals and try to achieve them

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



① simple reflex agent: if-then rules

- Rules to map percepts into observations:

$\forall b, g, u, c, t \text{ Percept}([\text{Stench}, b, g, u, c], t) \rightarrow \text{Stench}(t)$

$\forall s, g, u, c, t \text{ Percept}([s, \text{Breeze}, g, u, c], t) \rightarrow \text{Breeze}(t)$

$\forall s, b, u, c, t \text{ Percept}([s, b, \text{Glitter}, u, c], t) \rightarrow \text{AtGold}(t)$

- Rules to select action given observations:

$\forall t \text{ AtGold}(t) \rightarrow \text{Action}(\text{Grab}, t);$

- Difficulties:

- Consider Climb: No percept indicates agent should climb out; position & holding gold not part of percept sequence
- Loops: percepts repeated when you return to a square, causing same response (unless we maintain some internal model of the world)

① Model based agents

- Acquire and use a model of their environment
- Model must change over time
- Agent should, in general, remember past environments
- Agent must be able to predict how an action will/may change the environment

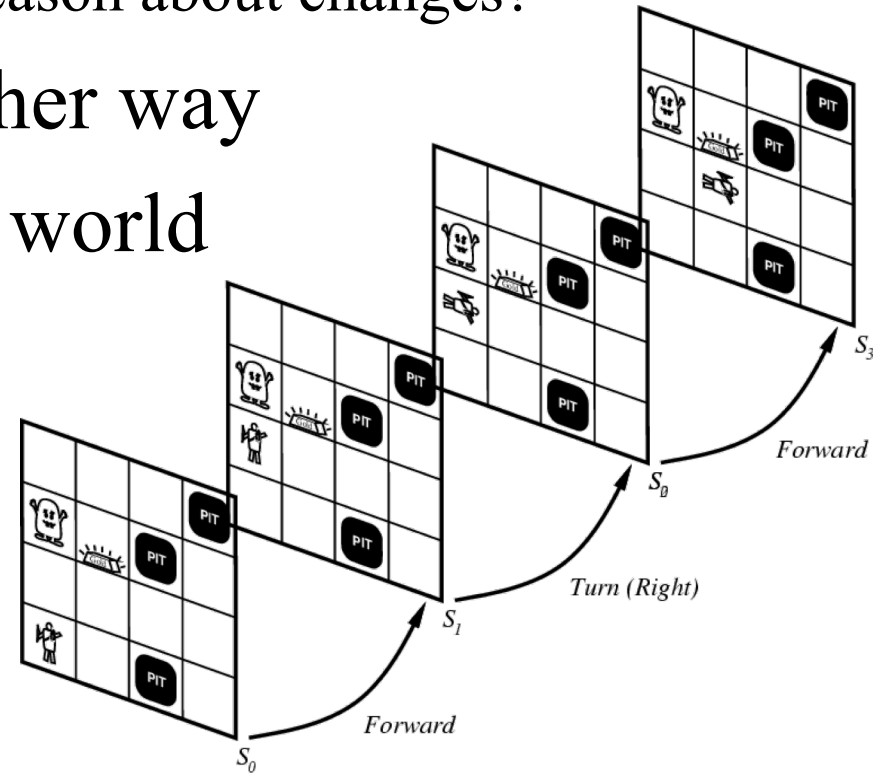
Representing change in Logic

- Representing changing world in logic is tricky
- One way: just change the KB
 - Add and delete sentences from KB to reflect changes
 - How do we remember past, or reason about changes?

- **Situation calculus** is another way

- **Situation**: snapshot of the world at some instant in time

- When the agent performs action A in situation S1, result is new situation S2



Situation calculus

- A **situation** is a snapshot of the world at an interval of time during which nothing changes
 - Need way to associate assertions with a situation
1. Add **situation variables** to **every** predicate
e.g., $\text{at}(\text{Agent}, L)$ becomes $\text{at}(\text{Agent}, L, s_0)$:
 $\text{at}(\text{Agent}, L)$ true in situation (i.e., state) s_0
 2. Add a special 2nd order predicate, **holds(f, s)**,
meaning “*f is true in situation s*”
e.g., $\text{holds}(\text{at}(\text{Agent}, L), s_0)$

Situation calculus

- Add new function, *result(a, s)*, mapping situation *s* to new situation as result of performing action *a*
 - i.e., *result(forward, s)* is a function returning next situation
- Example: The action agent-walks-to-location-*y* could be represented by

$$(\forall x)(\forall y)(\forall s) (\text{at}(\text{Agent}, L1, S) \wedge \neg \text{onbox}(S)) \\ \rightarrow \text{at}(\text{Agent}, L2, \text{result}(\text{walk}(L2), S))$$

Deducing hidden properties

- From the perceptual information we obtain in situations, we can **infer properties of locations**

$$\forall l, s \text{ at}(\text{Agent}, L, s) \wedge \text{Breeze}(s) \rightarrow \text{Breezy}(L)$$

$$\forall l, s \text{ at}(\text{Agent}, L, s) \wedge \text{Stench}(s) \rightarrow \text{Smelly}(L)$$

- Neither *Breezy* nor *Smelly* need situation arguments because pits and the Wumpus do not move around

Deducing hidden properties II

- Need rules relating aspects of single world state (as opposed to between states)
- Two main kinds of such rules:
 - **Causal rules** reflect assumed direction of causality
 - $(\forall L1, L2, S) \text{ at(Wumpus, L1, S)} \wedge \text{ adjacent(L1, L2)} \rightarrow \text{Smelly(L2)}$
 - $(\forall L1, L2, S) \text{ at(Pit, L1, S)} \wedge \text{ adjacent(L1, L2)} \rightarrow \text{Breezy(L2)}$
 - Systems with causal rules are **model-based reasoning systems**
 - **Diagnostic rules** infer presence of **hidden properties** directly from percept-derived information, e.g.
 - $(\forall L, S) \text{ at(Agent, L, S)} \wedge \text{ Breeze(S)} \rightarrow \text{Breezy(L)}$
 - $(\forall L, S) \text{ at(Agent, L, S)} \wedge \text{ Stench(S)} \rightarrow \text{Smelly(L)}$

Blocks world

The **blocks world** is a micro-world consisting of a table, a set of blocks and a robot hand

Some domain constraints:

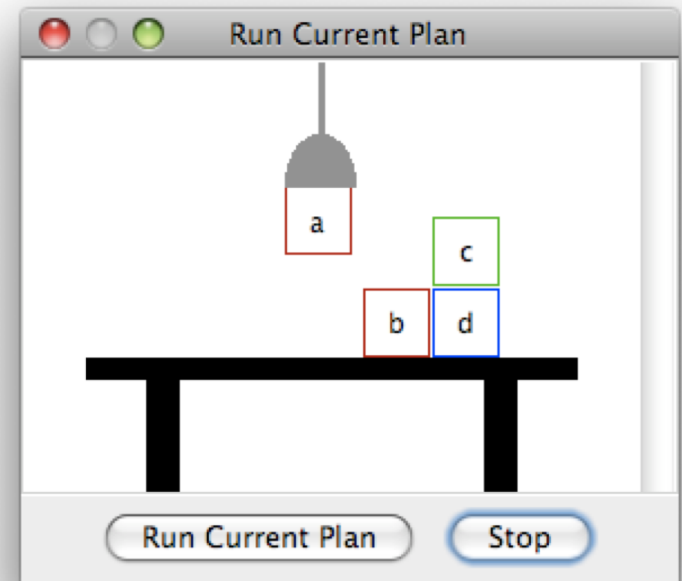
- Only one block can be on another block
- Any number of blocks can be on table
- The hand can only hold one block

Typical representation:

`ontable(b)` `ontable(d)`

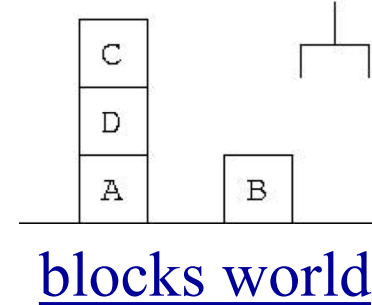
`on(c,d)` `holding(a)`

`clear(b)` `clear(c)`



Meant to be a simple model!

Representing change



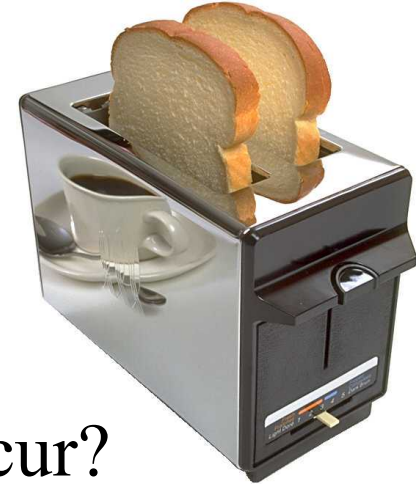
- **Frame axioms** encode what's **not** changed by an action
- E.g., moving a clear block to the table doesn't change the location of any other blocks

$\text{On}(x, z, s) \wedge \text{Clear}(x, s) \rightarrow$

$\text{On}(x, \text{table}, \text{Result}(\text{Move}(x, \text{table}), s)) \wedge$
 $\neg \text{On}(x, z, \text{Result}(\text{Move}(x, \text{table}), s))$

$\text{On}(y, z, s) \wedge y \neq x \rightarrow \text{On}(y, z, \text{Result}(\text{Move}(x, \text{table}), s))$

- Proliferation of frame axioms becomes very cumbersome in complex domains
 - What about color, size, shape, ownership, etc.



Qualification problem

- How can you characterize every effect of an action, or every exception that might occur?
- Putting my bread into the toaster, & pushing the button, it will become toasted after two minutes, unless...
 - The toaster is broken, or...
 - The power is out, or...
 - I blow a fuse, or...
 - A neutron bomb explodes nearby and fries all electrical components, or...
 - A meteor strikes the earth, and the world we know it ceases to exist, or...

Ramification problem

Nearly impossible to characterize every side effect of every action, at every level of detail



When I put my bread into the toaster, and push the button, the bread will become toasted after two minutes, and...

- The crumbs that fall off the bread onto the bottom of the toaster over tray will also become toasted, and...
- Some of those crumbs will become burnt, and...
- The outside molecules of the bread will become “toasted,” and...
- The inside molecules of the bread will remain more “breadlike,” and...
- The toasting process will release a small amount of humidity into the air because of evaporation, and...
- The heating elements will become a tiny fraction more likely to burn out the next time I use the toaster, and...
- The electricity meter in the house will move up slightly, and...

Knowledge engineering!



- Modeling the *right* conditions and the *right* effects at the *right* level of abstraction is difficult
- Knowledge engineering (creating & maintaining KBs for intelligent reasoning) is field unto itself
- We hope automated knowledge acquisition and machine learning tools can fill the gap:
 - Intelligent systems should **learn** about conditions and effects, just like we do!
 - Intelligent systems should learn when to pay attention to, or reason about, certain aspects of processes, depending on context. (metacognition?)

③ Goal-based agents

- Goal based agents model their goals
- And how their actions move them closer or farther than their goals

Preferences among actions

- Problem: how to decide which of several actions is best
- E.g., in choosing between forward and grab, axioms describing when it is OK to move to a square would have to mention glitter
- Not modular!
- We can solve this problem by separating **facts about actions** from **facts about goals**
- This way our agent can be reprogrammed just by asking it to achieve different goals

Preferences among actions

- First step: describe desirability of actions independent of each other
- We can use a simple scale: actions can be *Great, Good, Medium, Risky* or *Deadly*
- Agent should always do best action it can find:

$$(\forall a,s) \text{Great}(a,s) \rightarrow \text{Action}(a,s)$$

$$(\forall a,s) \text{Good}(a,s) \wedge \neg(\exists b) \text{Great}(b,s) \rightarrow \text{Action}(a,s)$$

$$(\forall a,s) \text{Medium}(a,s) \wedge (\neg(\exists b) \text{Great}(b,s) \vee \text{Good}(b,s)) \\ \rightarrow \text{Action}(a,s)$$

...

Preferences among actions

Until it finds gold, basic agent strategy can be:

- **Great actions:** picking up gold when found; climbing out of cave with the gold
- **Good actions:** moving to square that's OK and hasn't been visited yet
- **Medium actions:** moving to a square that is OK and has already been visited
- **Risky actions:** moving to square that's not known to be deadly or OK
- **Deadly actions:** moving into a square that is known to have pit or Wumpus

Achieving one goal uncovers another

- Once gold is found, we must change strategies, requiring new set of action values
- We could encode this as a rule:
 - $(\forall s) \text{ Holding}(\text{Gold}, s) \rightarrow \text{GoalLocation}([1, 1], s)$
- We must decide how the agent will work out a sequence of actions to accomplish the goal
- Three possible approaches:
 - **Inference**: good versus wasteful solutions
 - **Search**: a problem with operators and set of states
 - **Planning**: to be discussed later

Coming up next

- Logical inference
- Knowledge representation
- Planning