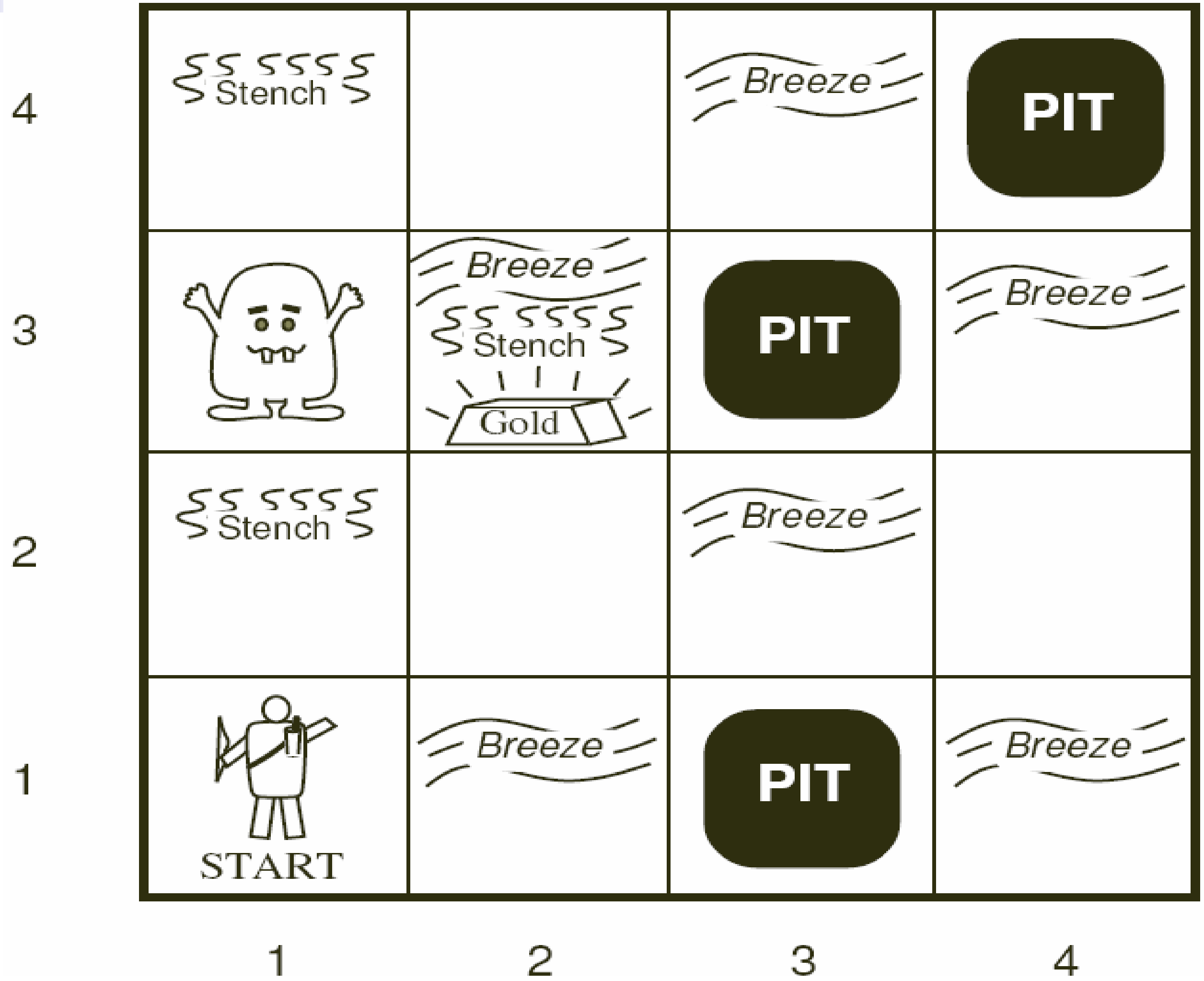
Situation Calculus

RN, Chapter 10.3

Logical Agents

- Reasoning [Ch 6]
- Propositional Logic [Ch 7]
- Predicate Calculus
  - Representation [Ch 8]
  - Inference [Ch 9]
- Implemented Systems [Ch 10]
- **Situation Calculus [Ch 10.3]**
- Planning [Ch 11]

Typical Wumpus World



Simple Reflex Agent

- Rep'n: At time = *t*, specify

  Percept([s,b,g,u,c], *t*)

  where s ∈ {Stench, –}, b ∈ {Breeze, –}, ...

  - Eg Tell(KB, Percept([Stench, –,Glitter, –, –], 3))

- Connect percepts directly to actions:

  ∀ s, b, u, c, t Percept([s, b, Glitter, u, c], t)
  ⇒ Action(Grab; t)

- Or, more indirectly:

  ∀ s, b, u, c, t Percept([s, b, Glitter, u, c], t) ⇒ AtGold(t)

  ∀ t AtGold(t) ⇒ Action(Grab, t)

- Q1: Which is more flexible?
- Q2: Limitations of reflex approach?

Problem with Reflex Agents

Q: When to Climb?

A: @ [1,1], and have Gold

. . . but . . .

agent cannot *sense* { being @ [1,1] ; having Gold }

Also. . . May be @ [2,3] when hunting for gold

Then reach same [2,3] when returning

As SAME percepts both times
& reflex agent can ONLY
use percept to decide on action
agent must take SAME actions both times.

⇒ ∞-loop!

⇒ Agent needs
INTERNAL MODEL of state/world

Tracking a Changing World

- Consider FINDING KEYS
  ... when "Keys are in pocket"

  ... agent could keep percept history, & replay it

  Better: just store this info!

FACT:

Any decision based on past/present percepts can be based on current "world state" (... which is updated each action)

So... perhaps KB should keep ONLY info about current situation

Single-Time Knowledge Base

Time 0: Initial configuration

```
At(Agent, [1,1])
Facing(Agent, North)
Smell(No)
...
```

Time 1: Then take a step `Action = Forward`
NOW in new situation:

Remove false statements
remove `At(Agent, [1,1])`, ...

Add in statement that are now true:
add `At(Agent, [1,2])`, ...

$\Rightarrow$ use revised KB:

```
At(Agent, [1,2])
Facing(Agent, North)
Smell(Yes)
...
```

Time 2: Turn to the right `Actions = TurnRight`

$\Rightarrow$ use revised KB:

```
At(Agent, [1,2])
Facing(Agent, East)
Smell(Yes)
...
```

Problems with Single-Time KBs

but . . . may need to reason about MANY times

Eg: "Was there a stench in [1,2] and [2,3]?"

- Need to
  Maintain info from previous states
  . . . labeled with state

- Kinda like "time stamp"
  . . . but "time" is not relevant
  better to record SEQUENCE of ACTIONS!

Compare: | Having GOLD at time 4 |

with

Having GOLD after *Going forward*, then *Turning right*, then *Going forward*, then *Grabbing*

$\Rightarrow$ Sequence of actions $\approx$ "plan"!

Situation Calculus

- Tag each "changable predicate" with "time":

Eg: At(Agent, [1,1], S_0)

At(Agent, [1,2], Result(Forward, S_0))

At(Agent, [1,3], Result(Forward, Result(Forward, S_0)))

...

Notice: all stay around!

- Only "label" predicates that can change.

As Pit doesn't move, At(Pit, $\langle 3,2 \rangle$) sufficient

Similarly, just 2+2=4, ...

- "Time" only wrt *actions*

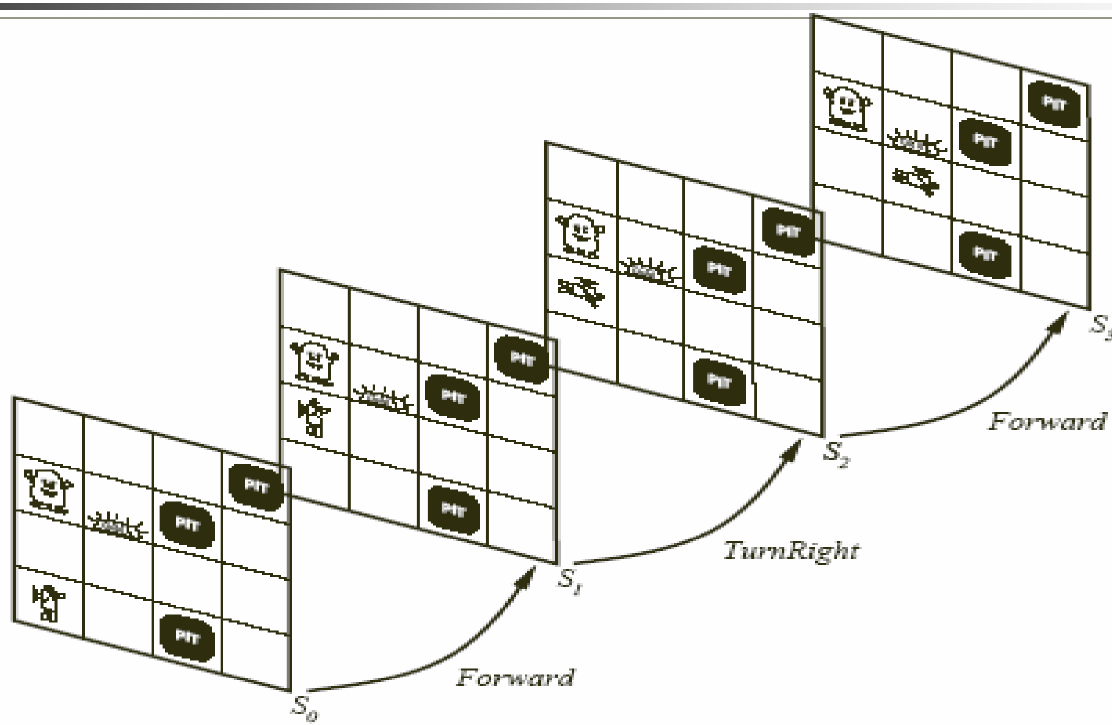
Snapshot of SITUATION...

World represented as

SERIES OF SITUATIONS

connected by actions

Updating State, Based on Action



$$S_1 = \texttt{Result(Forward, } S_0 \texttt{)}$$

$$S_2 = \texttt{Result(TurnRight, } S_1 \texttt{)}$$
$$= \texttt{Result(TurnRight, Result(Forward, } S_0\texttt{))}$$

$$S_3 = \texttt{Result(Forward, } S_2 \texttt{)}$$
$$= \texttt{Result(Forward, Result(TurnRight, Result(Forward, } S_0 \texttt{)))}$$

$$\texttt{Result: Action} \times \texttt{State} \mapsto \texttt{State}$$

Computing Location

- Can compute location:

$$\forall i, j, s\ \texttt{At}(\texttt{Ag}, [\,i, j\,], s) \quad \wedge \quad \texttt{Facing}(\texttt{Ag}, \texttt{North}, s) \quad \Rightarrow$$
$$\texttt{At}(\texttt{Ag}, [\,i, j+1\,], \texttt{Result}(\texttt{Forward}, s))$$

$$\forall i, j, s\ \texttt{At}(\texttt{Ag}, [\,i, j\,], s) \quad \wedge \quad \texttt{Facing}(\texttt{Ag}, \texttt{East}, s) \quad \Rightarrow$$
$$\texttt{At}(\texttt{Ag}, [\,i-1, j\,], \texttt{Result}(\texttt{Forward}, s))$$

...

- Can compute orientation:

$\forall i, j, s$ Facing(Ag, North, s) $\Rightarrow$
Facing(Ag, East, Result(TurnRight, s))

...

Interpreting Percepts

- Extract Individual Percepts

$$\forall\, b, g, u, c, t\ \ \texttt{Percept([Stench, b, g, u, c], t)} \Rightarrow \texttt{Stench(t)}$$
$$\forall\, b, u, c, s, t\ \ \texttt{Percept([s, Breeze, g, u, c], t)} \Rightarrow \texttt{Breeze(t)}$$

- Combine with State to make Inferences about Locations

$$\forall\, \ell, s\ \ \texttt{At}(Agent, \ell, s) \land \texttt{Stench}(s) \Rightarrow \texttt{Smelly}(\ell)$$
$$\forall\, \ell, s\ \ \texttt{At}(Agent, \ell, s) \land \texttt{Breeze}(s) \Rightarrow \texttt{Breezy}(\ell)$$

- Combine with Causal Rules to Infer Locations of Wumpus, Pits

$$\forall\, \ell\ \ \texttt{Breezy}(\ell) \Leftrightarrow [\exists\, x\ \ \texttt{PitAt}(x) \land \texttt{Adj}(\ell, x)]$$
$$\forall\, \ell\ \ \texttt{Stench}(\ell) \Leftrightarrow [\exists\, x\ \ \texttt{WumpusAt}(x) \land \texttt{Adj}(\ell, x)]$$
$$\forall\, \ell\ \ \texttt{OK}(\ell) \Leftrightarrow (\neg\texttt{WumpusAt}(\ell) \land \neg\texttt{PitAt}(\ell))$$

Deducing Hidden Properties

- Squares are breezy near a pit:
  - Diagnostic rule — infer cause from effect

    $\forall \ell \; \texttt{Breezy}(\ell) \Rightarrow \exists x \; \texttt{Pit}(x) \wedge \texttt{Adj}(\ell, x)$
  - Causal rule — infer effect from cause

    $\forall \ell, x \; \texttt{Pit}(x) \wedge \texttt{Adj}(\ell, x) \Rightarrow \texttt{Breezy}(\ell)$
  - Neither is complete. . .

    Eg, the causal rule doesn't say whether squares far away from pits can be breezy

  $\Rightarrow$ Definition for Breezy predicate:

    $\forall \ell \; \texttt{Breezy}(\ell) \Leftrightarrow [\exists x \; \texttt{PitAt}(x) \wedge \texttt{Adj}(\ell, x)]$

- Squares are stenchy near the wumpus:

  $\forall \ell \; \texttt{Stench}(\ell) \Leftrightarrow [\exists x \; \texttt{WumpusAt}(x) \wedge \texttt{Adj}(\ell, x)]$

Using Information

- After concluding

$$\neg\texttt{Stench}([1,1]) \quad \neg\texttt{Stench}([1,2]) \quad \texttt{Stench}([2,1])$$

$$\begin{array}{lcl} \forall \ell\ \texttt{Adj}([1,1],\ell) & \Leftrightarrow & (\ell=[2,1] \vee \ell=[1,2]) \\ \forall \ell\ \texttt{Adj}([2,1],\ell) & \Leftrightarrow & (\ell=[1,1] \vee \ell=[2,2] \vee \ell=[3,1]) \\ \forall \ell\ \texttt{Adj}([1,2],\ell) & \Leftrightarrow & (\ell=[1,1] \vee \ell=[2,2] \vee \ell=[1,3]) \end{array}$$

- Can derive

$\neg[\exists x\ \texttt{WumpusAt}(x) \wedge \texttt{Adj}([1,1],x)]$

$\forall x\ \texttt{Adj}([1,1],x) \Rightarrow \neg\texttt{WumpusAt}(x)$

$\neg\texttt{WumpusAt}([1,2]), \quad \neg\texttt{WumpusAt}([2,1])$

$\neg[\exists x\ \texttt{WumpusAt}(x) \wedge \texttt{Adj}([1,2],x)]$

$\neg\texttt{WumpusAt}([1,1]),\ \neg\texttt{WumpusAt}([2,2]),\ \neg\texttt{WumpusAt}([3,1])$

$[\exists x\ \texttt{WumpusAt}(x) \wedge \texttt{Adj}([2,1],x)]$

$\texttt{WumpusAt}([1,1]) \vee \texttt{WumpusAt}([2,2]) \vee \texttt{WumpusAt}([1,3])$

$\ldots \Rightarrow \quad \texttt{WumpusAt}([1,3])$

Connecting Inferences to Actions

- Rate Each Action

  $\forall$ r_1, s WumpusAt(r_1) & LocationAhead(Agent, s) = r_1
  $\Rightarrow$ Deadly(Forward, s)
  $\forall$ r_1, s   OK(r_1 s) & LocationAhead(Agent,s) = r_1 & $\neg$Visited(r_1 ,s)
  $\Rightarrow$ Good(Forward, s)
  $\forall$ r_1, s   Gold(r_1,s) $\Rightarrow$ Great(Grab, s)

- Choose Best Action

  $\forall$ a, s Great(a, s) $\Rightarrow$ Action(a, s)
  $\forall$ a, s Good(a, s) & ($\neg\exists$ b Great(b, s)) $\Rightarrow$ Action(a, s)

- Now, for each situation S,

  Ask(KB, $\exists$a Action(a, S))

. . . find a s.t. KB $\models$ Action(a, S)

Propagating Information

Effect Actions: *If agent Grabs, when in room with gold, he will be holding gold.*

$$\forall \ell, s \; \texttt{Glitters}(\ell) \wedge \texttt{At}(\texttt{Agent}, \ell, s) \Rightarrow \texttt{AtGold}(s)$$

$$\forall \ell, s \; \texttt{AtGold}(s) \Rightarrow \texttt{Holding}(\texttt{Gold}, \texttt{Result}(\texttt{Grab}, s))$$

So, if $\texttt{Glitters}([3,2])$, $\texttt{At}(\texttt{Agent}, [3,2], S_6)$,

then $\texttt{Holding}(\texttt{Gold}, S_7)$

where $S_7 = \texttt{Result}(\texttt{Grab}, S_6)$

What about NEXT situation?

eg, $S_8 = \texttt{Result}(\texttt{Turn\_Right}, S_7)$?

Want to derive

$\texttt{Holding}(\texttt{Gold}, \texttt{Result}(\texttt{Turn\_Right}, S_7))$,

This requires . . .

Frame Axioms

- $\forall a, x, s\ \texttt{Holding}(x, s) \ \wedge\ (a \neq \texttt{Release})$
  $\Rightarrow \quad \texttt{Holding}(x, \texttt{Result}(a, s)\ )$

  $\forall a, s\ \neg\texttt{Holding}(\texttt{Gold}, s) \ \wedge\ (a \neq \texttt{Grab} \vee \neg\texttt{AtGold}(s)$

  $\Rightarrow \quad \neg\texttt{Holding}(\ \texttt{Gold}, \texttt{Result}(a,\ s)\ )$

Gen'l:

true afterwards ⇔ [an action made it true ∨ (true already & no action made it false)]

Here: $\forall a, s\ \texttt{Holding}(\texttt{Gold}, \texttt{Result}(a, s)) \quad \Leftrightarrow$
$[\ (a = \texttt{Grab} \wedge \texttt{AtGold}(s)\ ) \quad \vee$
$(\ \texttt{Holding}(\texttt{Gold}, s) \ \wedge\ a \neq\ \texttt{Release}\ )\ ]$

Similarly: $\forall\ a, d, p, s\ \texttt{At}(\ p,\ \ell,\ \texttt{Result}(a, s)\ ) \quad \Leftrightarrow$
$[\ (a = \texttt{Forward} \ \wedge\ \ell = LocAhead(p, s) \ \wedge\ \neg\texttt{Wall}(\ell))$
$\vee\ (\texttt{At}(p, \ell, s) \ \wedge\ a \neq Forward)\ ]$

- "Successor State Axiom"

  Lists all ways predicate can become true/false

Frame, and Related, Problems

- Representational Frame Problem
  - Encoding what doesn't change, as actions take place
  - Solved via “success-state axioms”
- Inferential Frame Problem
  - … deal with long sequences of actions, ..
- Qualification Problem
  - dealing with all qualifications
  - … gold brick is not slippery, not screwed to table, …
- Ramification
  - When picking up the gold brick, also pick up the associated dust . . .

Goal-Based Agent

- These rules sufficient to FIND gold
  Then what?
- Need to change strategies:
  - Was "Find gold"
  - Now: "Get out!"

  $\forall s\ Holding(Gold, s) \Rightarrow GoalLocation([1, 1], s)$

  Need to incorporate… How?

How to Plan?

- Planning agents seek
  plan ≡ sequence of actions
  that achieve agent's goals.
- Inference: Let logical reasoning system perform search:
  Ask(KB, $\exists\, a_1, a_2, a_3, a_4, a_5, t$
  $t = Result(a_5; Result(a_4; Result(a_3; Result(a_2; Result(a_1; S_0)))))$
  & Holding(Agent; Gold; t) & At(Agent;Outside; t))
- Problematic, as
  - not easy to heuristically guide reasoning system. . .
  - What if > 5 actions required?
  - ...
- Search: View actions as operations on KB,
  Goal = “KB includes Holding(Agent, Gold, t), At(Agent, Outside, t))"
- Planning: Special purpose reasoning systems...

Logical Agents

- React to what it perceives
- Extract abstract descriptions of current state from percepts
- Maintain internal model of relevant aspects of world
  ... even those not directly observable
- Express and use info about desirability of actions in circumstances
- Use goals in conjunction with knowledge
  about actions to construct plans
- As all domain-specific knowledge is encoded as logical formulae, agent is completely generic!

Logic, Uncertainty, and Utility

Advantages of Logic-Based Agents

- High-level language for tracking environments.
- Permits modular decomposition of state representation.

Limitations of Simple Logic-Based Agents

- Cannot track stochastic environments.
- Cannot represent and reason with utilities – can't make tradeoffs

Limitations of Situation Calculus

- Situation Calculus works well for Wumpus World
  But…
- “Discrete Actions”
  Can't handle continuous actions
  - Flow of Electrons
  - Control of factory
  - . . .
- Action at an “instant”
  What if actions have duration?
- One action at a time
  - What if multiple agents?
  - What if world changes spontaneously?

Time and First-Order Logic

- Representing & reasoning
  with dynamic / changing world
  is not strong point of first-order logic
- Work on different logics:
  Eg dynamic logic / nonmonotonic logic
- Nonmon: long struggle
  Yale shooting problem:
  - Actions:
    load gun / point gun / wait 5 seconds / fire gun
  - Question:
    Is target dead? (was gun loaded when fired)
  - > 100 research papers since 1986; still not fully resolved
- First-order Logic better at “static” information