Planning

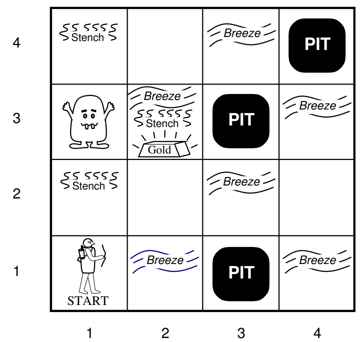
AI lecture (Yoonsuck Choe): Material from Russel and Norvig (2nd ed.)

- 7.2, 7.7: Wumpus world (an example domain)
- 10.3: Situation calculus
- 11: Planning

Planning

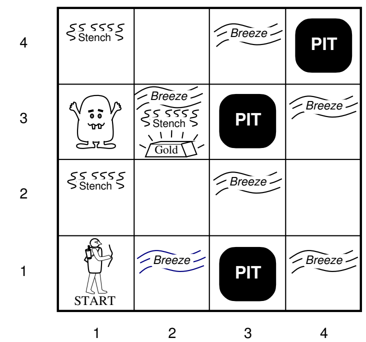
- The task of coming up with a sequence of actions that will achieve a goal is called **planning**.
- Simple approaches:
  - Search-based
  - Logic-based
- **Representation** of states and actions become important issues.

Example Domain: Wumpus World



- Want to get to the gold and grab it.
- Want to avoid pits and the “wumpus”.
- Clues: breeze near pits and stench near the wumpus.
- Other sensors: wall (bump), gold (glitter), kill (scream)
- Actions: move, grab, or shoot.

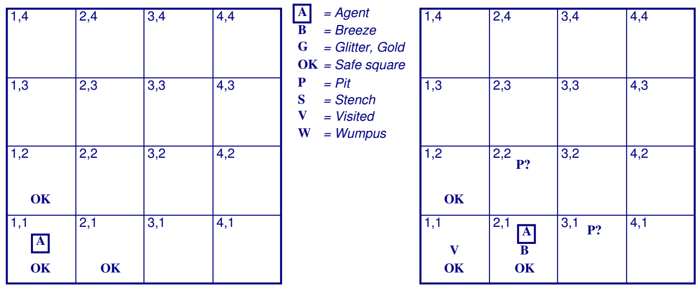
Wumpus World (WW)



Performance measure

- +1000: picking up gold
- -1000: fall in a pit, or get eaten by the wumpus
- -1: each action taken
- -10: each arrow used

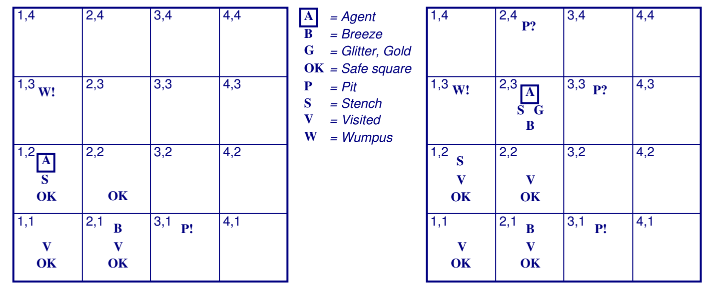
Evolution of Knowledge in WW



(a) (b)

- Move from [1,1] to [2,1].
- Based on the sensory data (breeze), we can mark [2,2] and [3,1] as potential pits, but not [1,1] since we came from there and we already know there's no pit there.

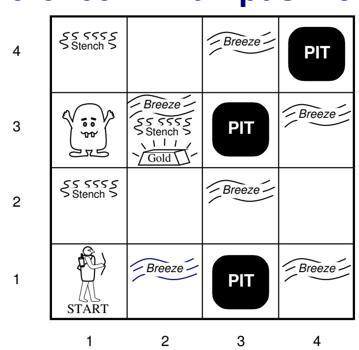
Evolution of Knowledge in WW



(a) (b)

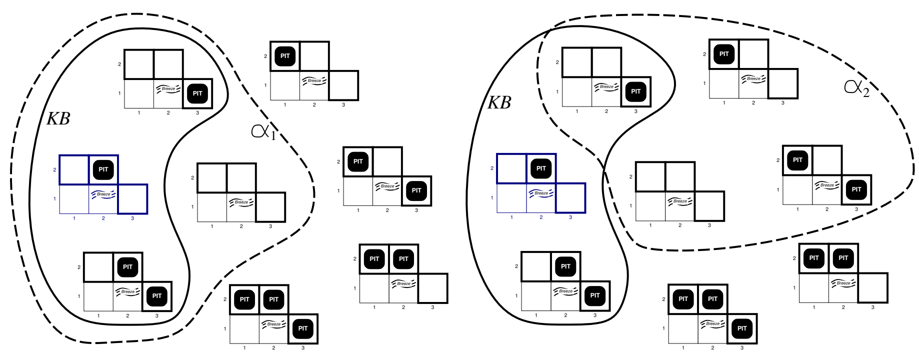
- Move back to [1,1] and then to [1,2]. At this point, the agent can infer that the wumpus is in [1,3]!
- Then move to [2,2] and then to [2,3] where the gold can be gound (glitter).

Inference in Wumpus World



- Knowledge Base: basic rules of the Wumpus World.
- Additional knowledge is added to the KB: facts you gather as you explore ([x,y] has stench, breeze, etc.)
- We can ask if a certain statement is a logical consequence of the KB: "There is a pit in [1,2]"

Inference in Wumpus World



KB: basic rules, plus [1,1] and [2,1] explored.

- α_1 = "There is no pit in [1,2]"
- α_2 = "There is no pit in [2,2]"
- Only α_1 follows from the KB.

Propositonal-logic-based Agent

- Query KB: Is there a Wumpus in [x,y]? Is there a pit in [x,y]?
- Add knowledge to KB (perceptual input): Breeze felt in [x,y], Stench detected in [x,y], etc.
- Decide which action to take (move where, etc.): Move to [x,y], grab gold, etc.

Note: here, there's only one goal, to grab the gold. Can we specify an arbitrary goal and derive a plan?

Problem: Propositions need to be explicit about location, e.g., $Breeze_{x,y}, Stench_{x,y}, \neg Wumpus_{x,y}$.

Situation Calculus

Make propositional-logic-based planner scalable.

- Situations: logical *terms* indicating a state.
  Example: In situation S_0 taking action a leads to situation S_1:

$$S_1 = Result(a, S_0).$$

- Fluents: *funcitons* and *predicates* that vary from one situation to the next.
  Example: $\neg Holding(Gold_1, S_0), Age(Wumpus)$

Other stuff: Atemporal/eternal predicates $Gold(Gold_1)$, empty actions $Result([], s) = s$, sequence of actions $Result([a|seq], s) = Result(seq, Result(a, s))$.

Situation Calculus: Tasks

- Projection:
  Deduce the outcome of a given sequence of actions
- Planning:
  Find a sequence of actions that achieves a desired effect.
  Example: Wumpus world

$$\text{Initial: } At(Agent, [1,1], S_0) \wedge At(G_1, [1,2], S_0), \ldots$$

$$\text{Goal: } \exists seq\ At(G_1, [1,1], Result(seq, S_0))$$

Describing Actions in Situation Calculus

Two axioms:

- Possibility axiom: when it is possible to execute an action

$$\text{Preconditions} \longrightarrow Poss(a, s)$$

- Effect axiom: What happens when a possible action is taken

$$Poss(a, s) \longrightarrow \text{Changes that result}$$

Wumpus World: Axioms

- Possibility axioms: Move, grab, release

$$At(Agent, x, s) \wedge Adjacent(x, y) \rightarrow Poss(Go(x, y(, s)$$

$$Gold(g) \wedge At(Agent, x, s) \wedge At(g, x, s) \rightarrow Poss(Grab(g), s)$$

$$Holding(g, s) \rightarrow Poss(Release(g), s)$$

- Effect axioms: Move, Grab, Release

$$Poss(Go(x, y), s) \rightarrow At(Agent, y, Result(Go(x, y), s))$$

$$Poss(Grab(g), s) \rightarrow Holding(g, Result(Grab(g), s))$$

$$Poss(Release(g), s) \rightarrow \neg Holding(g, Result(Release(g), s))$$

Frame Problem

- In the previous slide, we cannot deduce if the following can be proven (G_1 represents a particular lump of gold):

$$At(G_1, [1, 1], Result([Go([1, 1], [1, 2]), Grab(G_1), Go([1, 2], [1, 1])], S_0)$$

- It is because the effect axioms say only *what should change*, but not *what does not change when actions are taken.*
- Initial solution: *Frame axioms*

$$At(o, x, s) \wedge (o \neq Agent) \wedge \neg Holding(o, s)$$
$$\rightarrow At(o, x, Result(Go(y, z), s)).$$

This says moving does not affect the gold when it is not held. Problem is that you need $O(AF)$ such axioms for all $(action, fluent)$ pair (A: num of actions, F: num of fluent predicates).

Two Frame Problems

- Representational frame problem:
  Explained in the previous slide
- Inferential frame problem:
  Projection of results of a t-step sequence of actions in time $O(Et)$ (E is the number of effects, typically much less than F, the number of fluent predicates), rather than $O(Ft)$ or $O(AEt)$.

Solving the Representational Frame Problem

- Consider how each fluent predicate evolves over time:
  Successor-state axioms Action is possible $\rightarrow$
  (Fluent is true in result state $\leftrightarrow$ Action's effect made it true
  $\vee$
  It was true before and action left it alone).
- Example:

$$Poss(a, s) \rightarrow$$
$$(At(Agent, y, Result(a, s)) \leftrightarrow a = Go(x, y)$$
$$\vee (At(Agent, y, s) \wedge a \neq Go(y, z))).$$

- Remaining issues: implicit effect (moving while holding something moves that something as well) – ramification problem. Can solve by using a more general succesor-state axiom.

Solving the Inferential Frame Problem

- Given a t-step plan p ($S_t = Result(p, S_0)$), decide which fluents are true in S_t.
- We need to consider each of the F frame axiom of each time step t.
- Axioms have an average size of AE/F, we have an $O(AEt)$ inferential work. Most of the work is done copying unchanged fluents from time step to time step.
- Solutions: use fluent calculus rather than situation calculus, or make the process more efficient.

Solving the Inferential Frame Problem

- Typical frame axiom: $Poss(a, s) \rightarrow$
  $$F_i(Result(a, s)) \leftrightarrow (a = A_1 \vee a = A_2 \ldots) \vee (F_i(s) \wedge (a \neq A_3) \wedge (a \neq A_4) \ldots)$$
- Several actions that make the fluent true and several that make the fluent false: Formalize using the predicate $PosEffect(a, F_i)$ and $NegEffect(a, F_i)$.
  $$Poss(a, s) \rightarrow F_i(Result(a, s)) \leftrightarrow PosEffect(a, F_i) \vee [F_i(s) \wedge \neg NegEffect(a, F_i)]$$
  $PosEffect(A_1, F_i), PosEffect(A_1, F_i)$
  $NegEffect(A_3, F_i), NegEffect(A_4, F_i)$
  * This can be done efficiently: get current action, and fetch its effects, then update those fluents $O(Et)$.

Other Formalisms

- Event calculus: Fluents hold at diffetent time points, not situations. Reasoning is done over time.
- Other constructs: generalized events (spatiotemporal), process, intervals, etc.
- Formal theory of belief: propositional attitude, reification, etc.

Truth Maintenance Systems

New facts inferred from the KB can turn out to be incorrect.

- Let's say P was derived in the KB and later it was found that $\neg P$.
- Adding $\neg P$ to the KB will invalidate the entire KB, so P should be removed ($Retract(KB, P)$).
- Care needs to be taken since other facts in the KB may have been derived from P, etc.
- Truth maintenance systems are designed to handle these complications.

Planning Approaches

- State-space search: forward or backward.
- Heuristic search: subgoal independence assumption.
- Partial-order planning: utilize problem decomposition. Can place two actions into a plan without specifying the order. Several different total order plans can be constructed from partial order plans.