``

Background:

Knowledge bases: Declarative

Knowledge base = set of sentences (declarations) in a formal language

- Adding to the KB
  - Agent Tells the KB what it perceives: S_i
  - Inference: derive new statements from $KB + S_i$
- Using the KB
  - Agent Asks the KB what action to take
  - Declaration of the action to take leads to action

Procedural vs. Declarative Knowledge

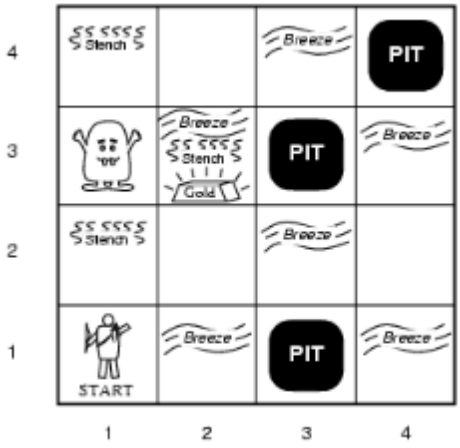
- Agents can be viewed at the knowledge level
  - Behavior is altered by adding knowledge
- Or at the implementation level
  - Procedural rules take KB statements as input
- The agent must be able to:
  - Represent states, actions, etc.
  - Incorporate new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions

Generic knowledge-based agent

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

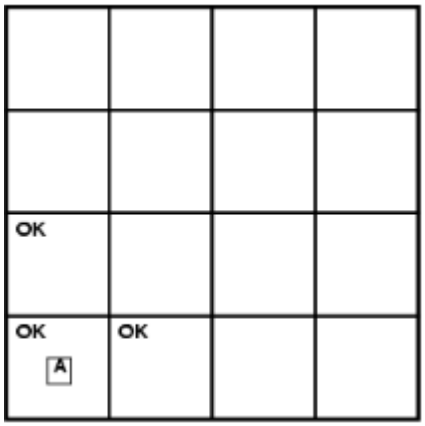
  TELL(KB, MAKE-PERCEPT-SENTENCE($percept, t$))
  $action \leftarrow$ ASK(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE($action, t$))
  $t \leftarrow t + 1$
  return *action*

Gold Miner Game description

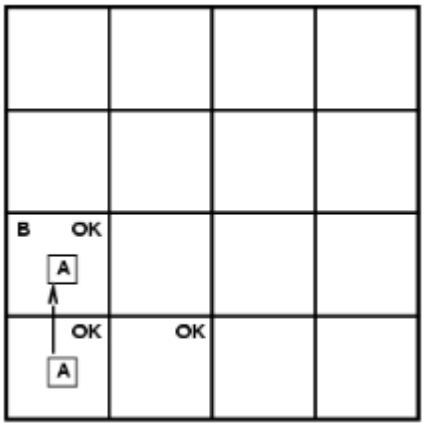


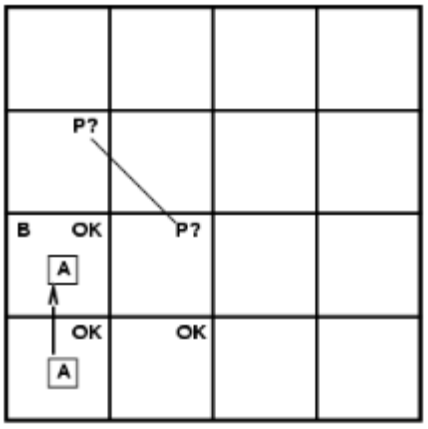
- Environment
  - Squares adjacent to wumpus have strench
  - Squares adjacent to pit have breeze
  - Shooting kills the monster if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square

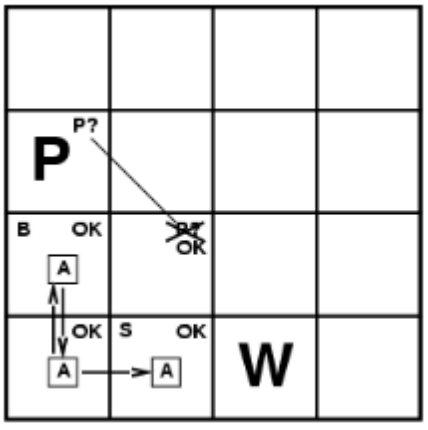
An example case exploration

- 1.
  - [1,1] is safe
  - agent starts in [1,1]
  - State detected: [None, None, None, None, None]
  - [1,2] is safe, [2,1] is safe
  - 

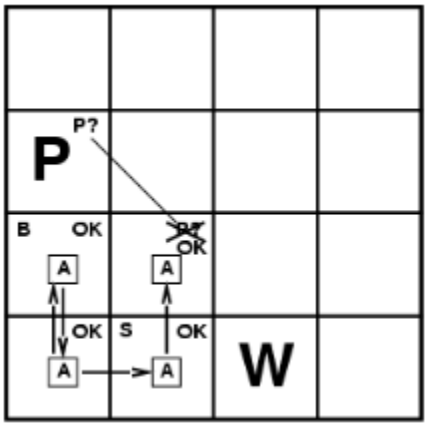
- 2.
  - Move to [1,2]
  - State detected: [Breeze, None, None, None, None]

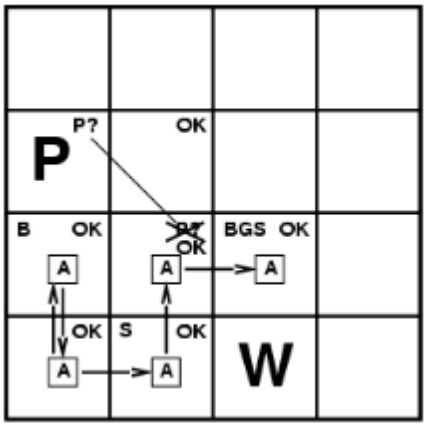
- 

- 3.
  - Possible Pit in [1,3] or [2,2]
  - 

- 4.
  - Move back to [1,1], then move to [2,1]
  - State detected: [None, None, None, Strench, None]
  - inference: **Monster is in [3,1]**
  - 

- 5.
  - Move to [2,2]
  - State detected: [None, None, None, None, None]
    - [2,3] is safe, [3,2] is safe

- 

- 6.
  - Move to [3,2]
    - State detected: [Breeze, Gold, None, Stench, None]
    - Grab gold-> player win
  - 

Logic Rules

Variables:

B : $Breeze$

G : $Gold$

S : $Strench$

P : Pit

M : $Monster$

x : $x\ coordinate\ on\ the\ board$

y : $y\ coordinate\ on\ the\ board$

$ArrowNumber$:

$$x\ ,\ where\ \ x \in \{0, 1\})$$

$Initial\ state\ in\ each\ cell$:

$$S(B, G, P, S, M) = (0, 0, 0, 0, 0)$$

$Initial\ state\ of\ player$:

$$Player(x, y) = (0, 0)$$

Initial state of player direction :

$$Facing(x),\ where\ \ x \in \{up, down, left, right\}$$

Rules:

$\forall x, y\ \ (x \in [0, 3] \land x \in \mathbb{Z}) \land (y \in [0, 3] \land y \in \mathbb{Z}) \leftrightarrow Valid(x, y)$

$\forall x, y\ \ Valid(x, y) \land (S(1) = 1) \leftrightarrow Breeze(x, y)$

$\forall x, y\ \ Valid(x, y) \land (S(2) = 1) \leftrightarrow Gold(x, y)$

$\forall x, y\ \ Valid(x, y) \land (S(3) = 1) \leftrightarrow Pit(x, y)$

$\forall x, y\ \ Valid(x, y) \land (S(4) = 1) \leftrightarrow Strench(x, y)$

$\forall x, y\ \ Valid(x, y) \land (S(5) = 1) \leftrightarrow Monster(x, y)$

$\neg Pit(0, 0) \land \neg Monster(0, 0) \rightarrow True$

$\forall x, y\ \ Valid(x, y) \land (Player(1) = x) \land (Player(2) = y) \leftrightarrow PlayerPosition(x, y)$

$\forall x, y\ \ (Valid(x - 1, y) \lor Valid(x + 1, y) \lor Valid(x, y - 1) \lor Valid(x, y + 1)) \leftrightarrow HasValidNeighbors(x, y)$

$\forall x, y\ \ Pit(x - 1, y) \lor Pit(x + 1, y) \lor Pit(x, y - 1) \lor Pit(x, y + 1) \leftrightarrow PitAtNeighbors(x, y)$

$\forall x, y\ \ Monster(x - 1, y) \lor Monster(x + 1, y) \lor Monster(x, y - 1) \lor Monster(x, y + 1) \leftrightarrow MonsterAtNeighbors(x, y)$

$\forall x, y\ \ Breeze(x - 1, y) \land Breeze(x + 1, y) \land Breeze(x, y - 1) \land Breeze(x, y + 1) \leftrightarrow BreezeAtAllNeighbors(x, y)$

$\forall x, y\ \ Strench(x - 1, y) \land Strench(x + 1, y) \land Strench(x, y - 1) \land Strench(x, y + 1) \leftrightarrow StrenchAtAllNeighbors(x, y)$

$\forall x, y\ \ Gold(x, y) \rightarrow Grab(gold)$

$$\forall x, y, k\ \ ((k < x) \land (k < y) \land (k \in \mathbb{Z})) \land Valid(x, y) \land PlayerPosition(x, y) \land$$
$$((Facing(up) \land Monster(x, y + k)) \lor (Facing(down) \land Monster(x, y - k)) \lor$$
$$(Facing(right) \land Monster(x + k, y)) \lor (Facing(left) \land Monster(x - k, y)))$$
$$\land (ArrowNumber = 1) \rightarrow Shoot(monster)$$

$\forall x, y\ \ Valid(x, y) \land Breeze(x, y) \land HasValidNeighbors(x, y) \rightarrow PitAtNeighbors(x, y)$

$\forall x, y\ \ Valid(x, y) \land Strench(x, y) \land HasValidNeighbors(x, y) \rightarrow MonsterAtNeighbors(x, y)$

$\forall x, y\ \ Valid(x, y) \land Pit(x, y) \land HasValidNeighbors(x, y) \rightarrow BreezeAtAllNeighbors(x, y)$

$\forall x, y\ \ Valid(x, y) \land Monster(x, y) \land HasValidNeighbors(x, y) \rightarrow StrenchAtAllNeighbors(x, y)$

$\forall x, y\ \ Valid(x, y) \land PlayerPosition(x, y) \land Monster(x, y) \land \neg Shoot(monster) \rightarrow Die(player)$

$\forall x, y\ \ Valid(x, y) \land PlayerPosition(x, y) \land Pit(x, y) \rightarrow Die(player)$

$\forall x, y\ \ Valid(x, y) \land PlayerPosition(x, y) \land Gold(x, y) \land Grab(gold) \rightarrow Win(player)$