Wumpus World

Scott A. DeLoach, PhD
Department Of Computing & Information Sciences
Kansas State University

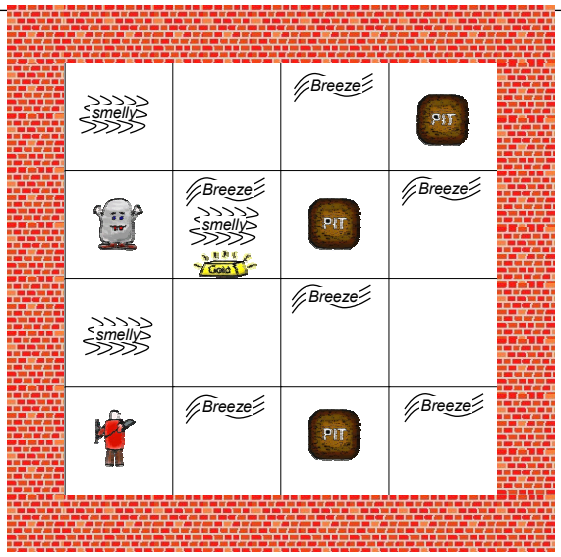
Wumpus World

- Agent is a treasure-hunter / adventurer
- Environment is a maze
  - Grid containing Walls, Pits, Wumpuses (Wumpi?), and Gold
- Rules
  - Must retrieve Gold
  - Agent dies if it enters a square containing a Pit or Wumpus
  - Agent has one arrow -- a Wumpus dies if shot

Wumpus World

- Percepts: breeze, glitter, smell, bump, scream
- Actions: turn left, turn right, forward, grab, release, shoot
- Goal: get gold back to start without entering pit or wumpus square
- Environment:
  - squares adjacent to wumpus are smelly
  - squares adjacent to pit are breezy
  - squares glitter only if gold is in the same square
  - shooting kills the wumpus if you are facing it
  - shoot uses up the only arrow
  - grabbing picks up the gold if in the same square
  - releasing drops the gold in the same square

Wumpus World



Wumpus World Environment

- Deterministic?
- Fully Accessible?
- Static?
- Discrete?

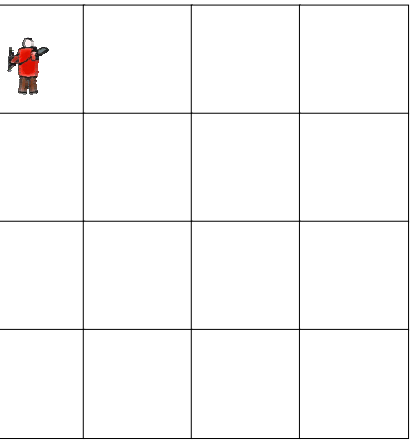
Wumpus World Environment

- Deterministic?
  - Yes, the outcomes are exactly specified
- Fully Accessible?
  - No, only local perceptions are available
- Static?
  - Yes, wumpus and pits do not move
- Discrete?
  - yes

Exploring Wumpus World

Start in cell [1,1]

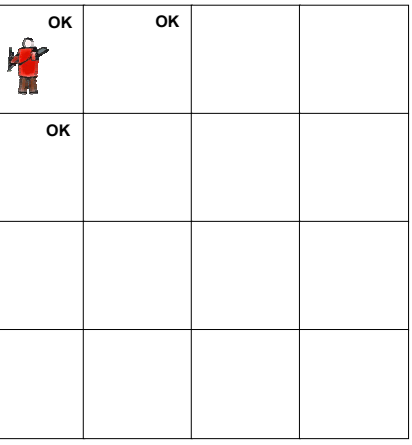
- have to look at percepts to determine what's in the environment



Exploring Wumpus World

Start in cell [1,1]

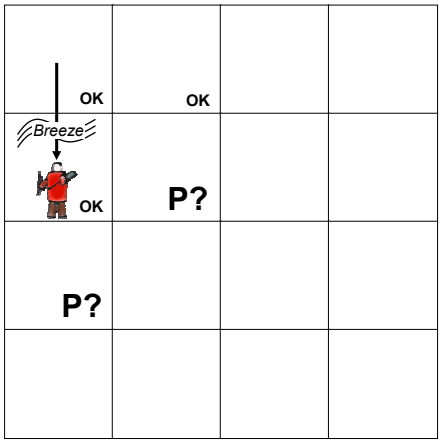
- no breeze or stench



Exploring Wumpus World

Move to cell [2,1]

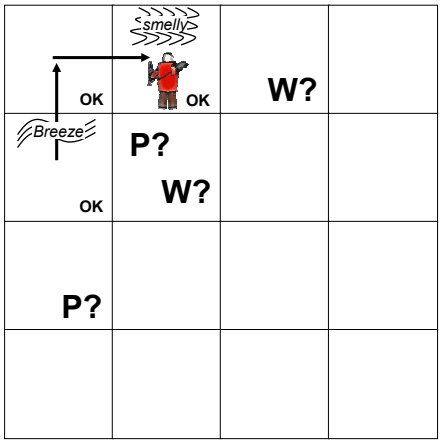
- perceive a breeze
- possible pit in [2,2] or [3,1]



Exploring Wumpus World

Move to cell [1,1] and then [1,2]

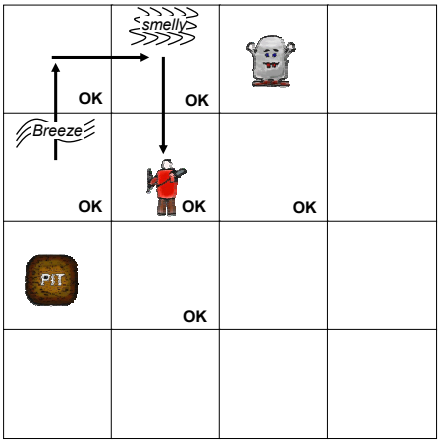
- perceive a smell
- possible wumpus in [2,2] or [1,3]



Exploring Wumpus World

Move to cell [2,2]

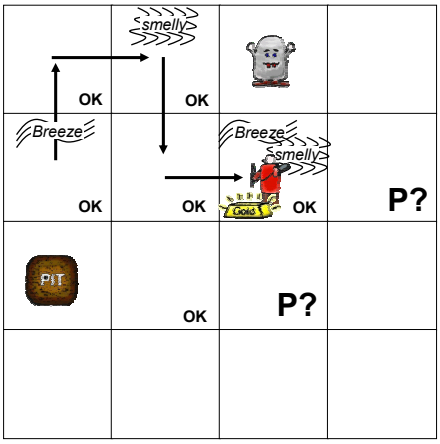
- **no pit or wumpus**
- therefore pit & wumpus locations are known
- no percepts
- [3,2] and [2,3] are ok ...



Exploring Wumpus World

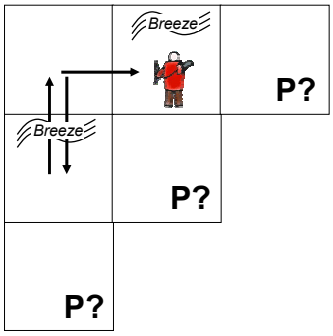
Move to cell [2,2]

- breeze, smell, & glitter
- grab the gold and head for home!



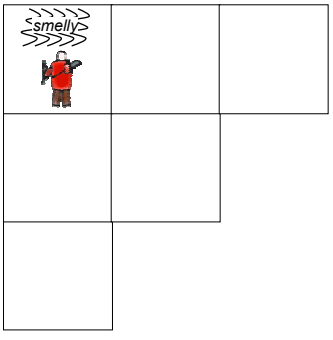
Other Tight Spots

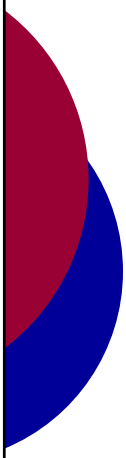
- Breeze in (1,2) and (2,1)
  - no safe action!
- Assuming pits uniformly distributed, (2,2) is most likely to have a pit



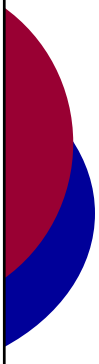
Other Tight Spots

- Smell in (1,1)
  - cannot move!
- Can use the strategy of coercion:
  - shoot straight ahead
  - if the wumpus was there he is now dead and it is safe
  - if the wumpus wasn't there it is safe





Implementation



Wumpus World Actions

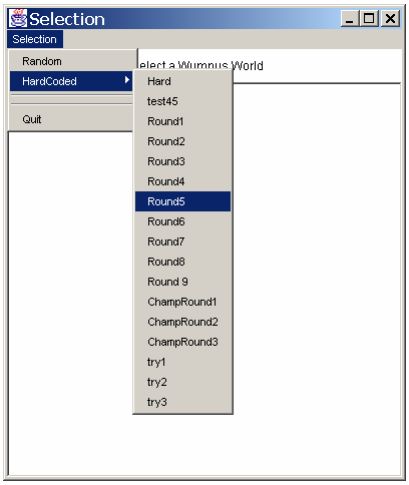
- Movement (forward, turn)
- Grab (the gold)
- Shoot (an arrow)
- Climb (exit maze)

Wumpus World Percepts

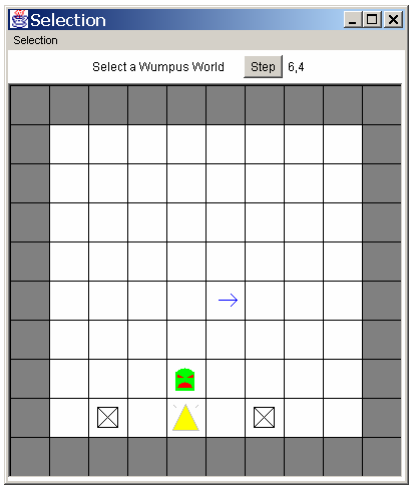
- isBump
- isGlitter (Gold in this square)
- isBreeze (Pit adjacent, not diagonal)
- isStench (Wumpus or DeadWumpus adjacent)
- isScream (Wumpus has become DeadWumpus)

Executing Wumpus World

Selecting a Saved Grid

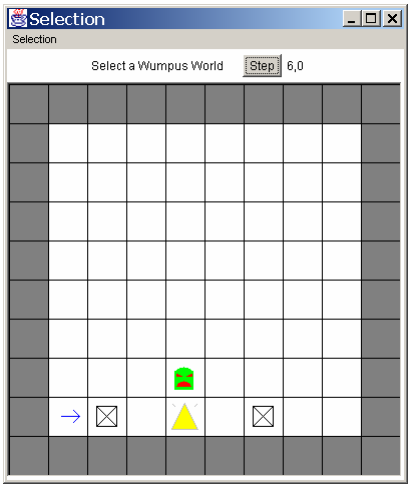


Starting on the Grid

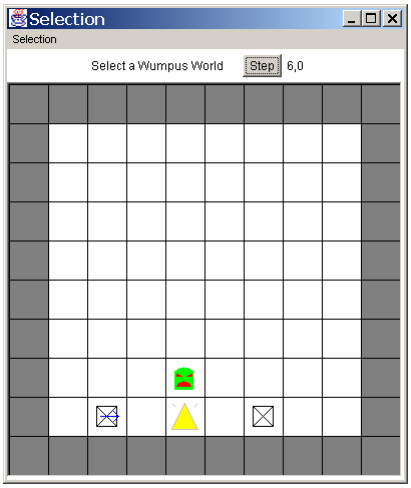


Note: Must "resize" window to get the grid to appear!

Searching



A Dead Wumpus Hunter



Wumpus World Performance Measure

- -1 point per move
- -10,000 points for dying
- 1000 points for exiting maze with gold

Wumpus Notes

- Three general types of mazes
  - Gold can be directly retrieved
  - Gold can be retrieved if a conclusion is reached
    - Wumpus location is determined - Wumpus is killed
    - Pit locations and safe path must be guessed
  - Gold cannot be retrieved
    - Agent should climb out after determining this
    - positive score impossible -- sometimes life isn't fair

Execution

- Run ksu.cis.wumpus.WumpusWorld
- Agent code is in ksu.cis.wumpus.WumpusHunterAgent
  - public WumpusHunterAgent(int xSize, int ySize)
    - Initializes size of grid and can be used for anything you want
  - public Action execute(Percept perceptArg)
    - The "logic" of the agent that can be changed to anything you want

Wumpus Hunter Initialization

```
package ksu.cis.wumpus;

/* fill in the methods of this class ... you may add or delete any methods
   algorithms, or data structures you need. */

import java.io.*;
class WumpusHunterAgent implements AgentProgram {
    // constants
    private final int right= 0;
    private final int left = 1;
    private final int up   = 2;
    private final int down = 3;

    // hidden attributes.  xLoc and yLoc are for the agent to "know" where it is
    private int xSize, ySize, xLoc, yLoc;
    private int gridMemory[][][];

    public WumpusHunterAgent(int xSize, int ySize) {
        this.xSize = xSize;
        this.ySize = ySize;
        xLoc = xSize;
        yLoc = ySize;
    }
}
```

Wumpus Hunter Agent

```
public Action execute(Percept perceptArg)
{
    WumpusPercept percept = (WumpusPercept) perceptArg;
    if (percept.isGlitter)
        return new AnAction("grab");
    if (percept.isBump)
        return new AnAction("turn", AIMA.randomChoice("right", "left"));
    if (percept.isBreeze)
        if (AIMA.random() < .60)
            return new AnAction("turn", AIMA.randomChoice("right", "left"));
    if (percept.isStench)
        if (AIMA.random() < .08)
            return new AnAction("shoot");
        else
            if (AIMA.random() < .3)
                return new AnAction("forward");
            else
                return new AnAction("turn", AIMA.randomChoice("right", "left"));
        if (AIMA.random() < .8)
        return new AnAction("forward");
    else
        return new AnAction("turn", AIMA.randomChoice("right", "left"));
    }
```