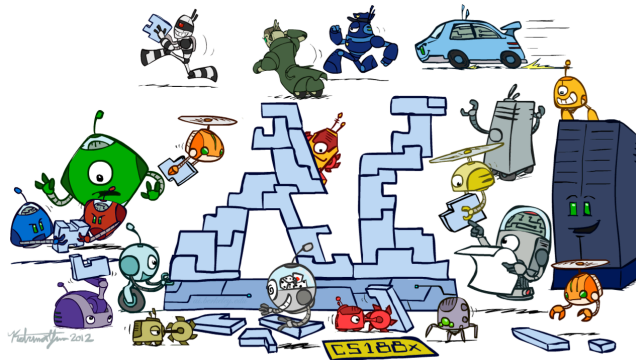


# CS 473: Artificial Intelligence

## Conclusion



Dan Weld – University of Washington

[Many of these slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at <http://ai.berkeley.edu>.]

## Exam Topics

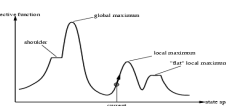
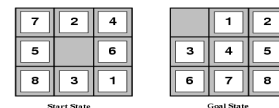
- **Search**
  - Problem spaces
  - BFS, DFS, UCS, A\* (tree and graph), local search
  - Completeness and Optimality
  - Heuristics: admissibility and consistency; pattern DBs
- **CSPs**
  - Constraint graphs, backtracking search
  - Forward checking, AC3 constraint propagation, ordering heuristics
- **Games**
  - Minimax, Alpha-beta pruning,
  - Expectimax
  - Evaluation Functions
- **MDPs**
  - Bellman equations
  - Value iteration, policy iteration
- **Reinforcement Learning**
  - Exploration vs Exploitation
  - Model-based vs. model-free
  - Q-learning
  - Linear value function approx.
- **Hidden Markov Models**
  - Markov chains, DBNs
  - Forward algorithm
  - Particle Filters
- **Bayesian Networks**
  - Basic definition, independence (d-sep)
  - Variable elimination
  - Sampling (rejection, importance)
- **Learning**
  - BN parameters with complete data
  - Search thru space of BN structures
  - Expectation maximization

# What is intelligence?

- (bounded) Rationality
  - Agent has a performance measure to optimize
  - Given its state of knowledge
  - Choose optimal action
  - With limited computational resources
- Human-like intelligence/behavior

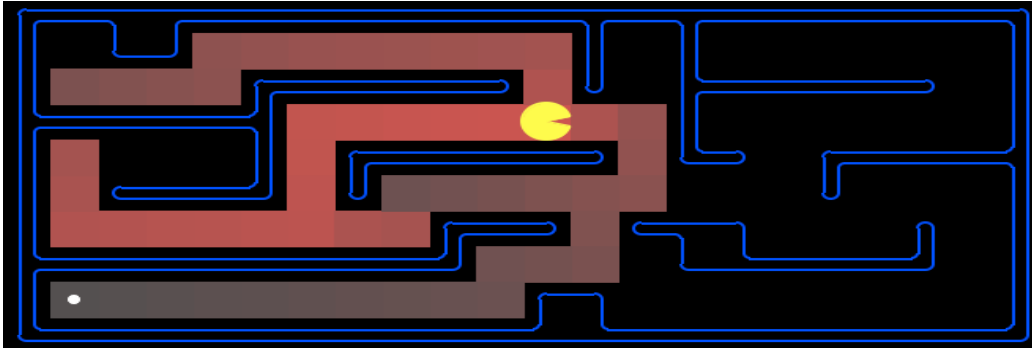
# Search in Discrete State Spaces

- Every discrete problem can be cast as a search problem.
  - states, actions, transitions, cost, goal-test
- Types
  - **uninformed systematic**: often slow
    - DFS, BFS, uniform-cost, iterative deepening
  - **Heuristic-guided**: better
    - Greedy best first, A\*
    - relaxation leads to heuristics
  - **Local**: fast, fewer guarantees; often local optimal
    - Hill climbing and variations
    - Simulated Annealing: global optimal
  - (Local) Beam Search

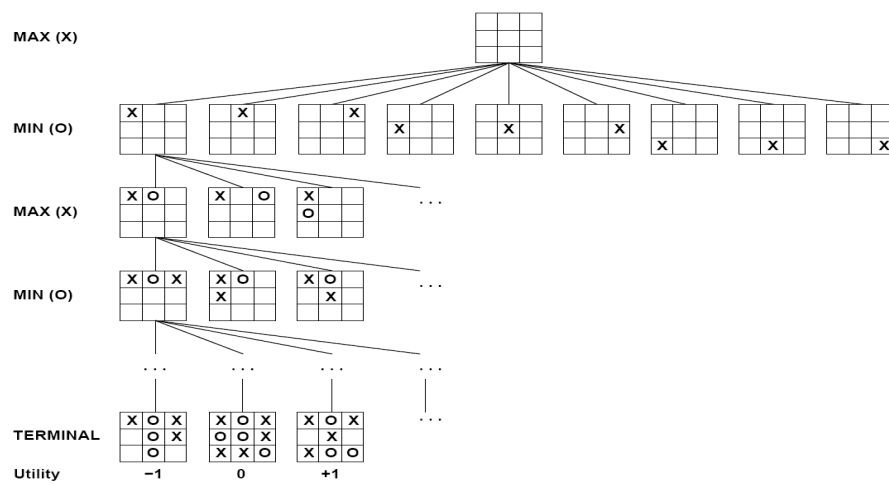


## Which Algorithm?

- A\*, Manhattan Heuristic:



## Adversarial Search



## Adversarial Search

- AND/OR search space (max, min)
- minimax objective function
- minimax algorithm (~dfs)
  - alpha-beta pruning
- Utility function for partial search
  - Learning utility functions by playing with itself
- Openings/Endgame databases



## Knowledge Representation and Reasoning

- **Representing: what agent knows**

Propositional logic  
Constraint networks  
HMMs  
Bayesian networks  
...

- **Reasoning: what agent can infer**

Search  
Dynamic programming  
Preprocessing to simplify

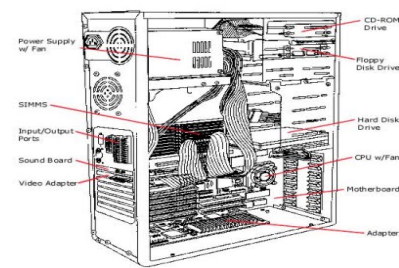
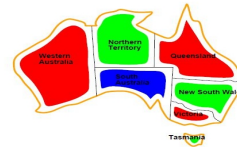
## Search+KR&R Example: CSP

### Representation

- Variables, Domains, Constraints

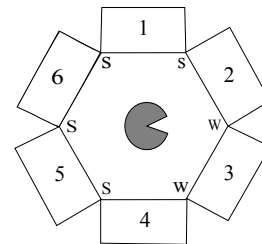
### Reasoning:

- Arc Consistency (k-Consistency)
- Solving
  - Backtracking search: partial var assignments
    - Heuristics: min remaining values, min conflicts
  - Local search: complete var assignments



## Trapped

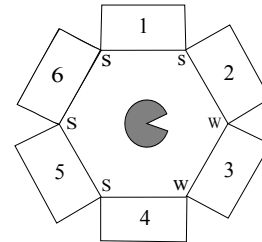
- Pacman is trapped! He is surrounded by mysterious corridors, each of which leads to either a pit (P), a ghost (G), or an exit (E). In order to escape, he needs to figure out which corridors, if any, lead to an exit and freedom, rather than the certain doom of a pit or a ghost. The one sign of what lies behind the corridors is the wind: a pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all. Unfortunately, Pacman cannot measure the strength of the breeze at a specific corridor. Instead, he can stand between two adjacent corridors and feel the max of the two breezes. For example, if he stands between a pit and an exit he will sense a strong (S) breeze, while if he stands between an exit and a ghost, he will sense a weak (W) breeze. The measurements for all intersections are shown in the figure below. Also, while the total number of exits might be zero, one, or more, Pacman knows that two neighboring squares will not both be exits.



Variables?

## Trapped

- Pacman is trapped! He is surrounded by mysterious corridors, each of which leads to either a pit (P), a ghost (G), or an exit (E). In order to escape, he needs to figure out which corridors, if any, lead to an exit and freedom, rather than the certain doom of a pit or a ghost. The one sign of what lies behind the corridors is the wind: a pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all. Unfortunately, Pacman cannot measure the strength of the breeze at a specific corridor. Instead, he can stand between two adjacent corridors and feel the max of the two breezes. For example, if he stands between a pit and an exit he will sense a strong (S) breeze, while if he stands between an exit and a ghost, he will sense a weak (W) breeze. The measurements for all intersections are shown in the figure below. Also, while the total number of exits might be zero, one, or more, Pacman knows that two neighboring squares will not both be exits.



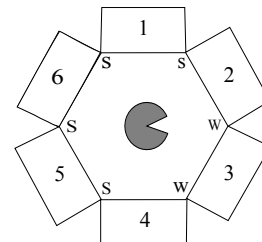
Variables?  $X_1, \dots, X_6$   
Domains {P, G, E}

12

## Trapped

- A pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all. Pacman feels the max of the two breezes.
- the total number of exits might be zero, one, or more,
- two neighboring squares will not both be exits.

Constraints?

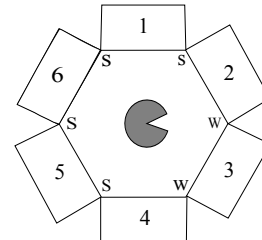


Variables?  $X_1, \dots, X_6$   
Domains {P, G, E}

13

# Trapped

- A pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all.
- Pacman feels the max of the two breezes.
- the total number of exits might be zero, one, or more,
- two neighboring squares will not both be exits.



Also!  $X_2 \neq P$   
 $X_3 \neq P$   
 $X_4 \neq P$

Constraints?

$X_1 = P$  or  $X_2 = P$        $X_2 = E$  or  $X_3 = E$   
 $X_3 = E$  or  $X_4 = E$        $X_4 = P$  or  $X_5 = P$   
 $X_5 = P$  or  $X_6 = P$        $X_6 = P$  or  $X_1 = P$

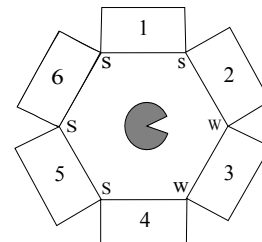
$X_i = E$  nand  $X_{i+1|7} = E$

$X_1$	P	G	E
$X_2$	P	G	E
$X_3$	P	G	E
$X_4$	P	G	E
$X_5$	P	G	E
$X_6$	P	G	E

14

# Trapped

- A pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all.
- Pacman feels the max of the two breezes.
- the total number of exits might be zero, one, or more,
- two neighboring squares will not both be exits.



Constraints?

$X_1 = P$  or  $X_2 = P$        $X_2 = E$  or  $X_3 = E$   
 $X_3 = E$  or  $X_4 = E$        $X_4 = P$  or  $X_5 = P$   
 $X_5 = P$  or  $X_6 = P$        $X_6 = P$  or  $X_1 = P$

$X_i = E$  nand  $X_{i+1|7} = E$

MRV heuristic?

Arc consistent?

$X_1$	P	G	E
$X_2$	P	G	E
$X_3$	P	G	E
$X_4$	P	G	E
$X_5$	P	G	E
$X_6$	P	G	E

15

## KR&R: Markov Decision Process

### Representation

- states, actions, probabilistic outcomes, rewards
- ~AND/OR Graph (sum, max)
- Generalization of expectimax

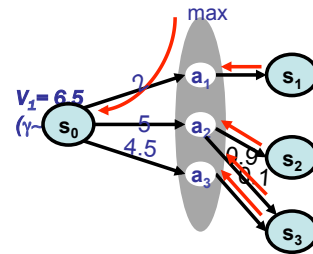
### Reasoning: $V^*(s)$

- Value Iteration: dynamicvalue space

$$V^*(s) = \max_a Q^*(s, a)$$

### Reinforcement Learning:

- Exp  $Q^*(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$
- Learn model or learn Q-function?



## KR&R: Markov Decision Process

### Representation

- states, actions, probabilistic outcomes, rewards

$$V^*(s) = \max_a Q^*(s, a)$$

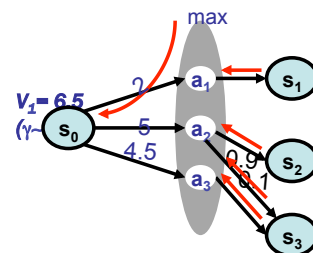
$$Q^*(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$$

### Reasoning: $V^*(s)$

- Expectimax
- Value Iteration: dynamic programming

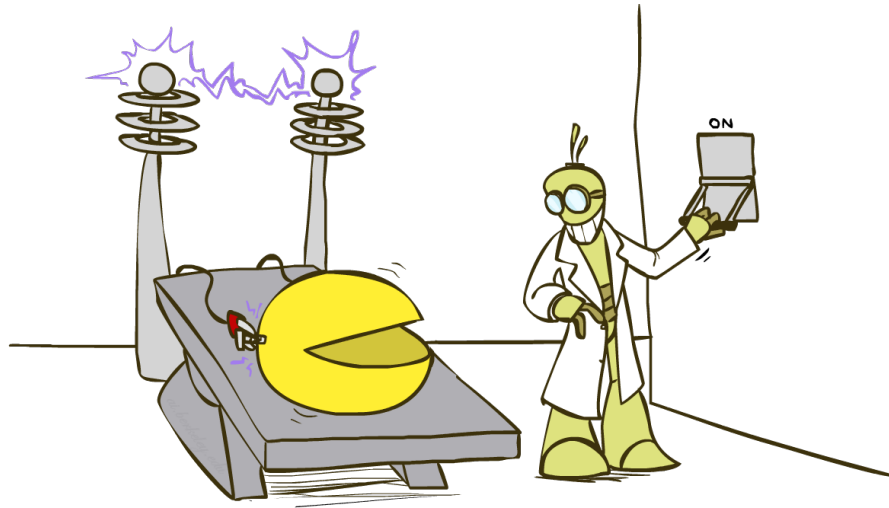
### Reinforcement Learning:

- Exploration / exploitation
- Learn model or learn Q-function?





## Pac-Man Beyond the Game!



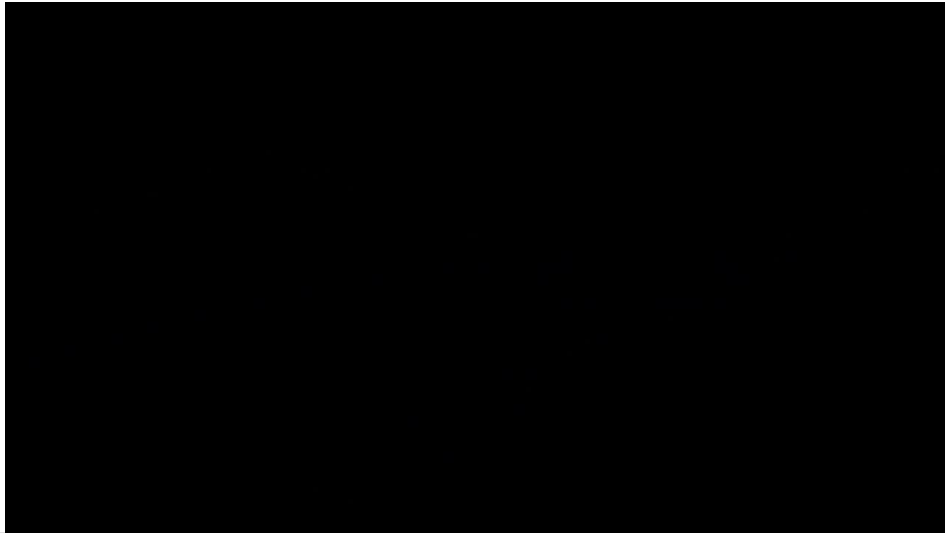
## Pacman: Beyond Simulation?



Students at Colorado University: <http://pacman.elstonj.com>

## Pacman: Beyond Simulation!

[VIDEO: Roomba Pacman.mp4]



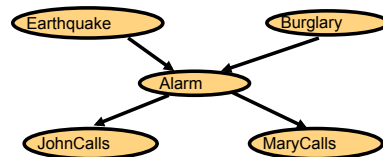
## KR&R: Probability

### ■ Representation: Bayesian Networks

- encode probability distributions compactly
  - by exploiting conditional independences

### ■ Reasoning

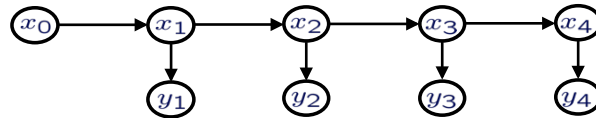
- Exact inference: var elimination
- Approx inference: sampling based methods
  - rejection sampling, likelihood weighting, MCMC/Gibbs



## KR&R: Hidden Markov Models

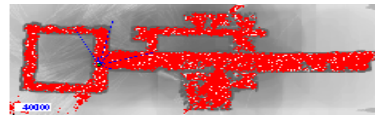
- **Representation**

- Spl form of BN
- Sequence model
- One hidden state, one observation



- **Reasoning/Search**

- most likely state sequence: Viterbi algorithm
- marginal prob of one state: forward-backward



## Learning Bayes Networks

- **Learning Structure of Bayesian Networks**

- Search thru space of BN structures

- **Learning Parameters for a Bayesian Network**

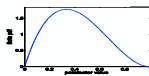
- Fully observable variables
  - Maximum Likelihood (ML), MAP & Bayesian estimation
  - Example: Naïve Bayes for text classification
- Hidden variables
  - Expectation Maximization (EM)

## Bayesian Learning

Use Bayes rule:

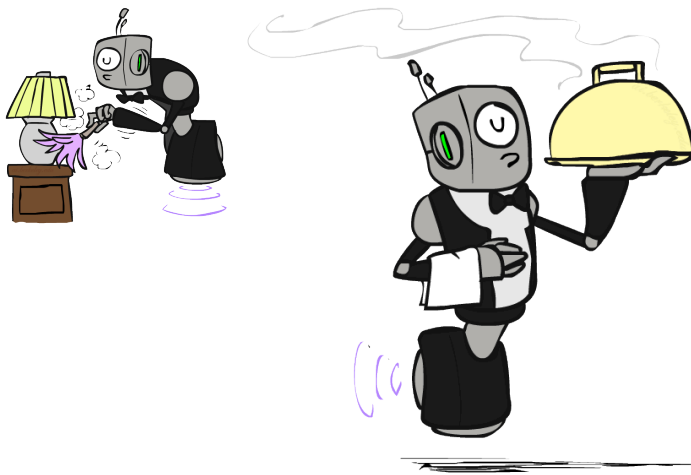
Diagram illustrating the components of Bayes' rule:

- Data Likelihood** (red text) points to  $P(X|Y)$  in the numerator.
- Prior** (red text) points to  $P(Y)$  in the numerator.
- Posterior** (red text) points to  $P(Y|X)$  in the numerator.
- Normalization** (red text) points to  $P(X)$  in the denominator.

$$P(Y|X) = \frac{P(X|Y) P(Y)}{P(X)}$$


Or equivalently:  $P(Y|X) \propto P(X|Y) P(Y)$

## Personal Robotics



## PR2 (autonomous)

[VIDEO: Spile\_200x.mp4]

[Maitin-Shepard, Cusumano-Towner, Lei, Abbeel, 2010]



## Autonomous tying of a knot for previously unseen situations

[VIDEO: knots\_apprentice.mp4]

[Schulman, Ho, Lee, Abbeel, 2013]



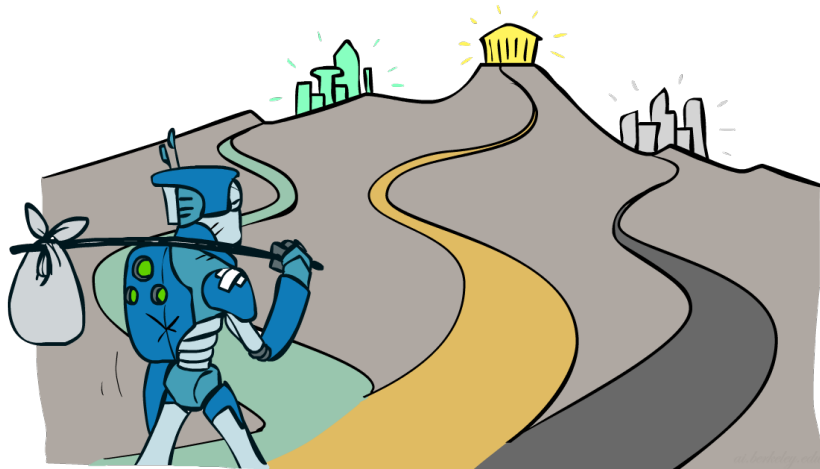
## Experiment: Suturing

[VIDEO: suturing-short-sped-up.mp4]

[Schulman, Gupta, Venkatesan,  
Tayson-Frederick, Abbeel, 2013]



## Where to Go Next?



## That's It!

- Help us out with some course evaluations
- Have a great summer, and always maximize your expected utilities!

