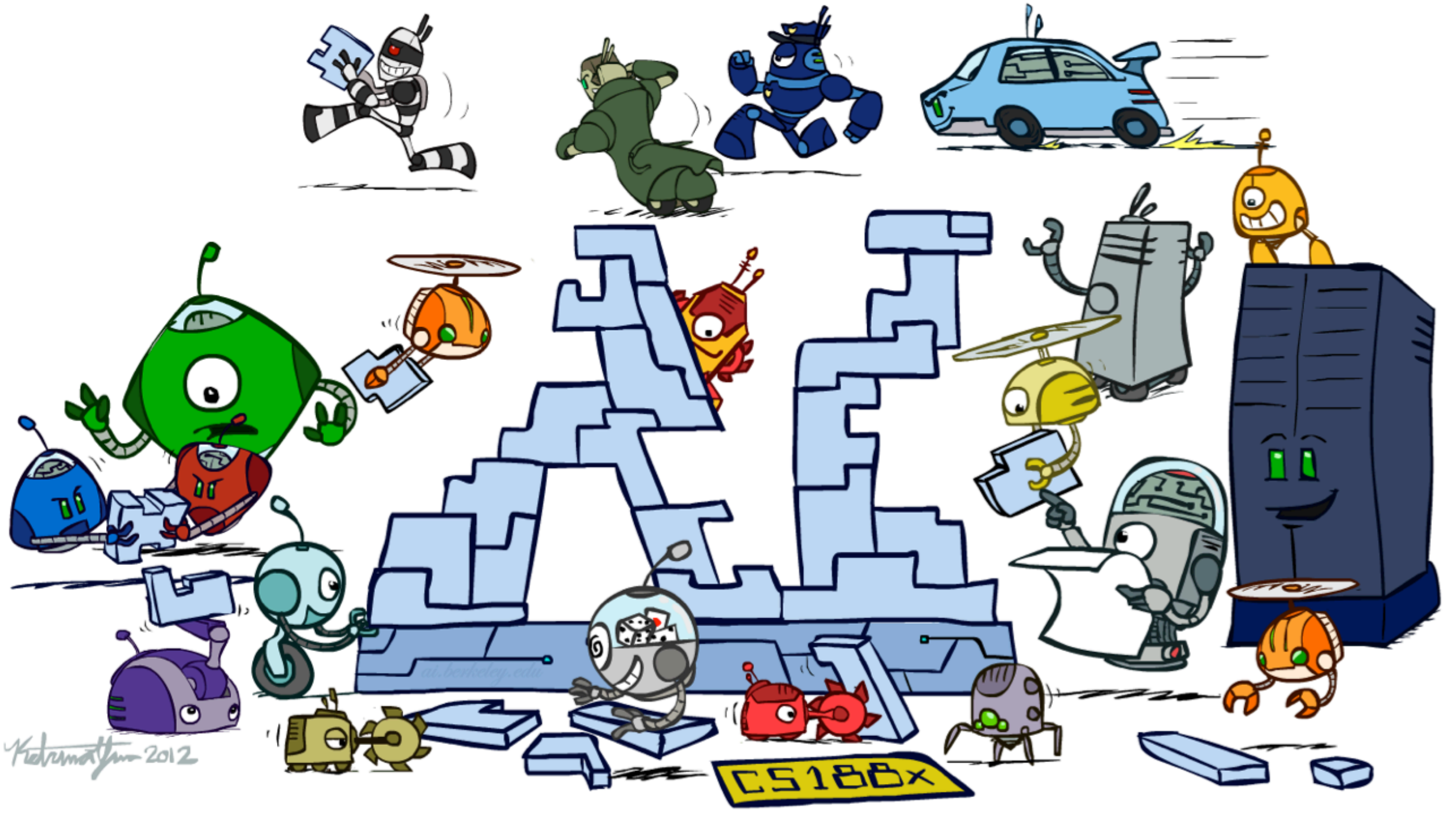
CSEP 573: Artificial Intelligence
Conclusion



Luke Zettlemoyer – 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.]

CourseTopics

- Search
  - Problem spaces
  - BFS, DFS, UCS, A* (tree and graph), local search
  - Completeness and Optimality
  - Heuristics: admissibility and consistency; pattern DBs
- 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
  - Naive Bayes
  - Perceptron
  - Neural Networks (not on exam)

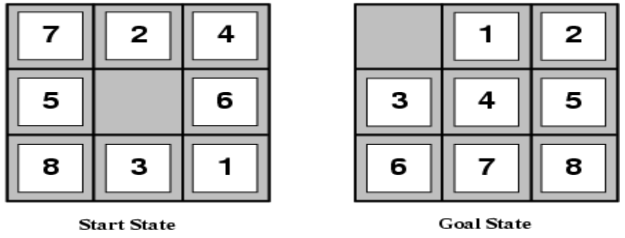
What is intelligence?

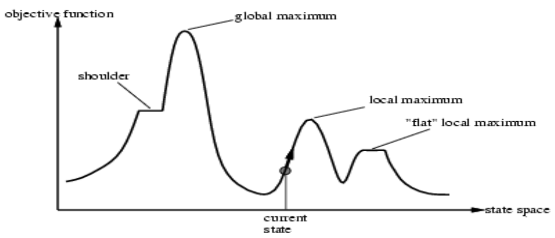
- (bounded) Rationality
  - Agent has a performance measure to optimize
  - Given its state of knowledge
  - Choose optimal action
  - With limited computational resources

- Human-like intelligence/behavior

Search in Discrete State Spaces

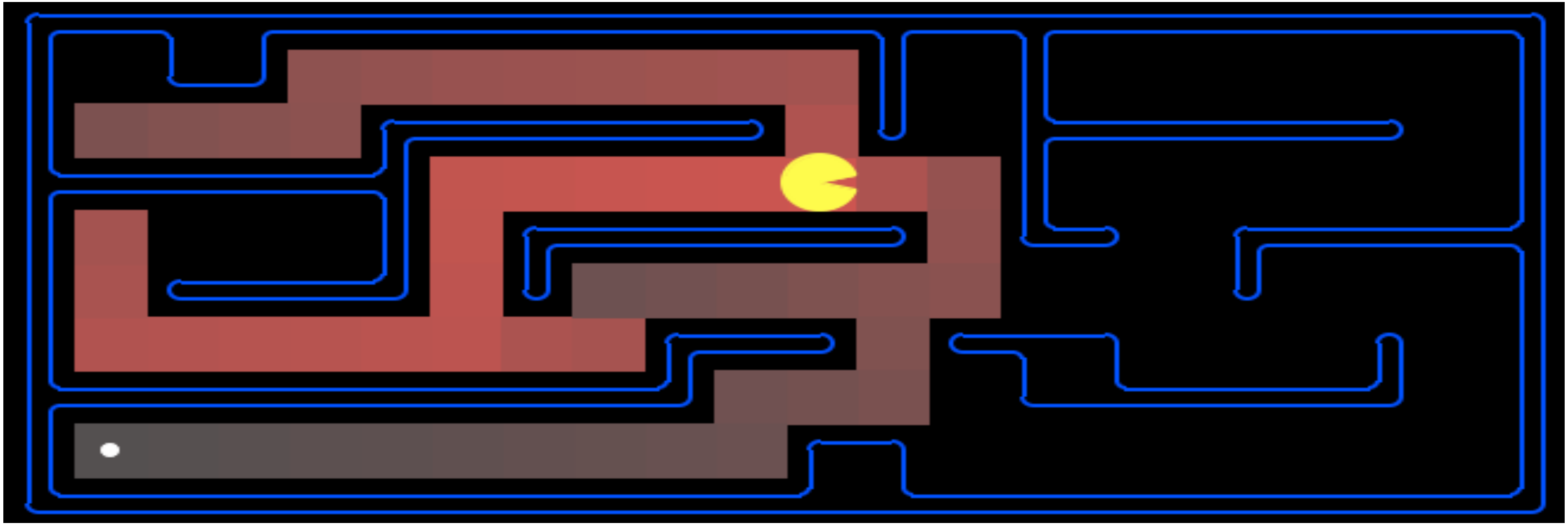
- Every discrete problem can be cast as a search problem.
  - states, actions, transitions, cost, goal-test
- Types
  - uninformed systematic: often slow
    - DFS, BFS, uniform-cost, iterative deepening
  - Heuristic-guided: better
    - Greedy best first, A*
    - relaxation leads to heuristics
  -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





Which Algorithm?

- A*, Manhattan Heuristic:

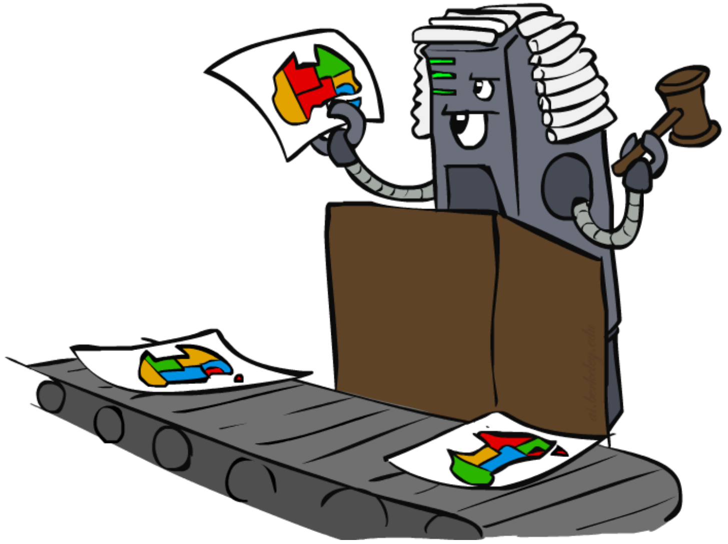


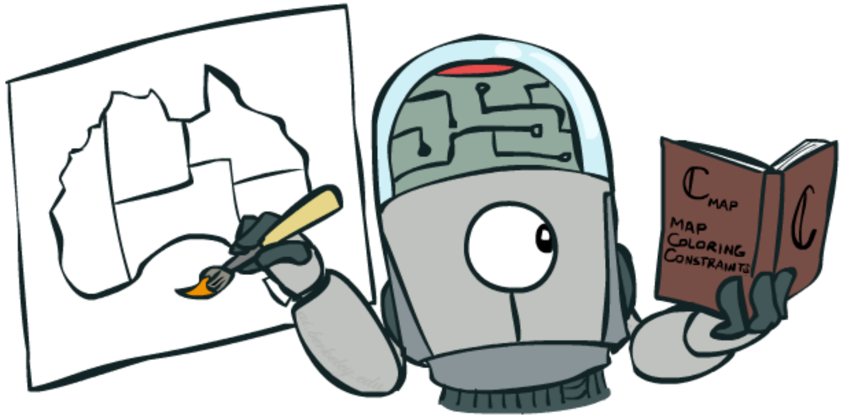
Constraint Satisfaction Problems

- Standard search problems:
  - State is a "black box": arbitrary data structure
  - Goal test can be any function over states
  - Successor function can also be anything

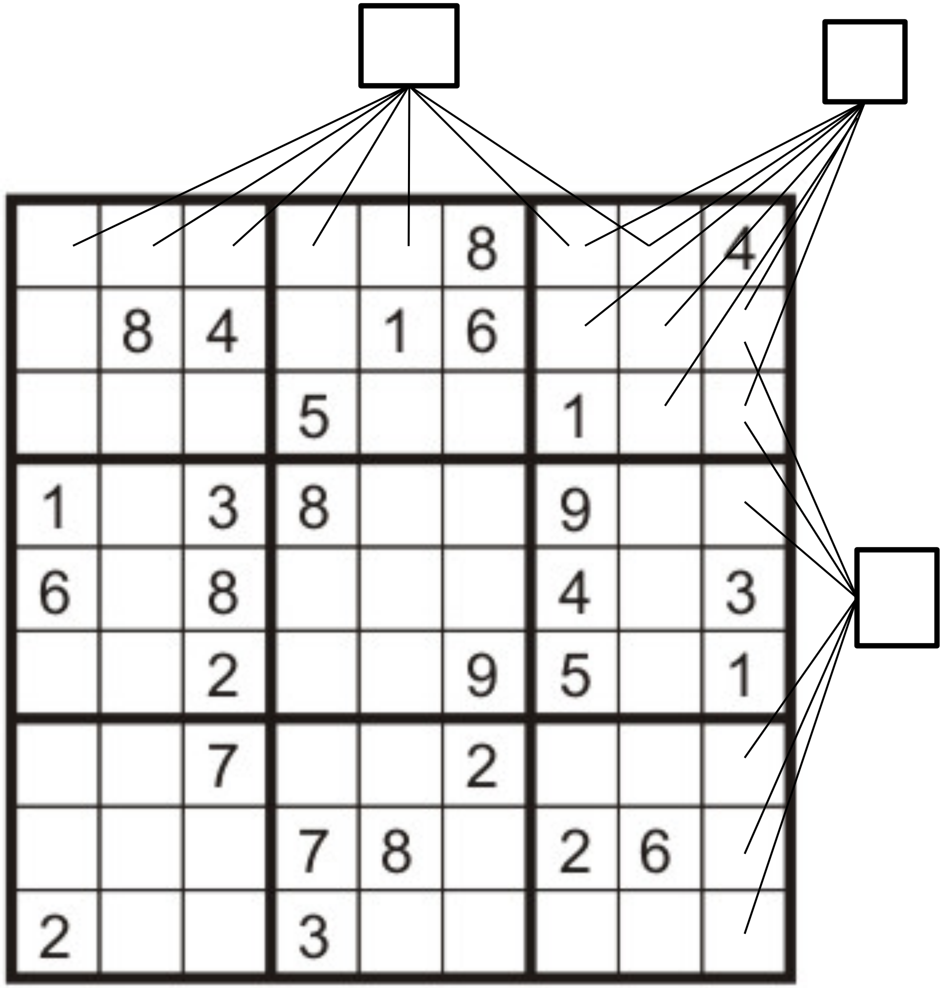
- Constraint satisfaction problems (CSPs):
  - A special subset of search problems
  - State is defined by variables X_i with values from a domain D (sometimes D depends on i)
  - Goal test is a set of constraints specifying allowable combinations of values for subsets of variables

- Making use of CSP formulation allows for optimized algorithms
  - Typical example of trading generality for utility (in this case, speed)





Example: Sudoku



- Variables:
  - Each (open) square
- Domains:
  - {1,2,…,9}
- Constraints:

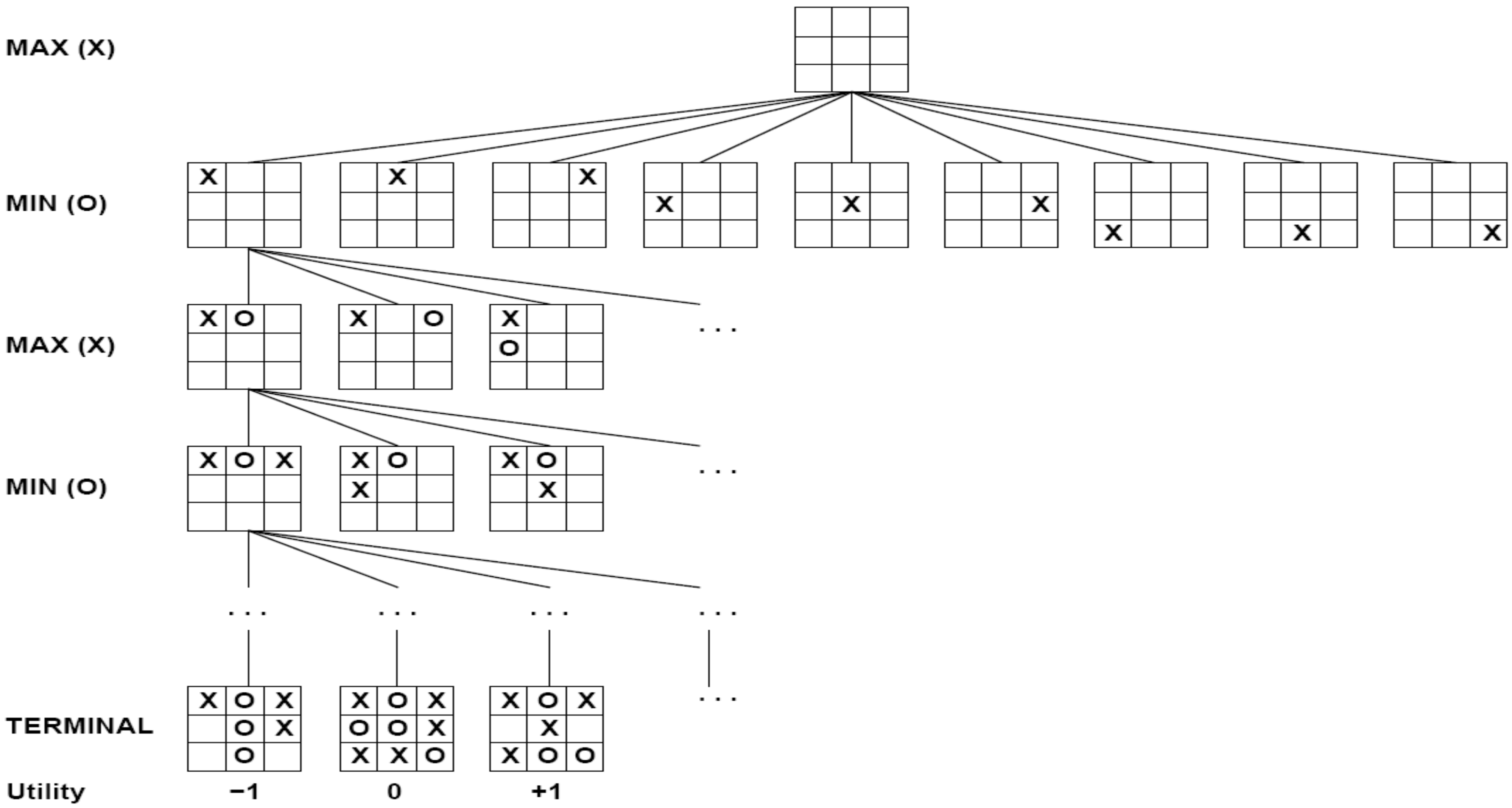
9-way alldiff for each column

9-way alldiff for each row

9-way alldiff for each region

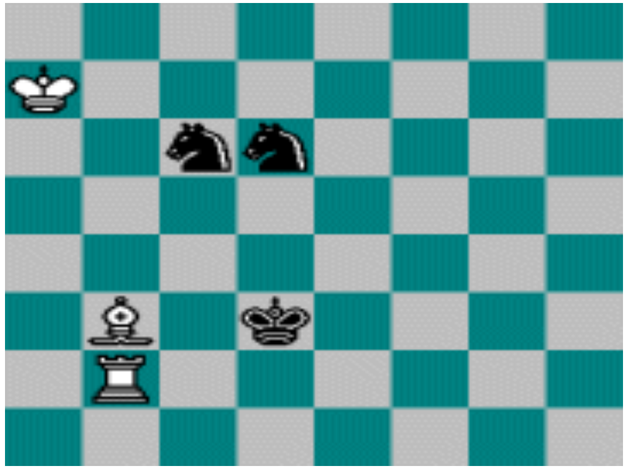
(or can have a bunch of pairwise inequality constraints)

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



Big News Today!

Google's AlphaGo wins second game against Go champion

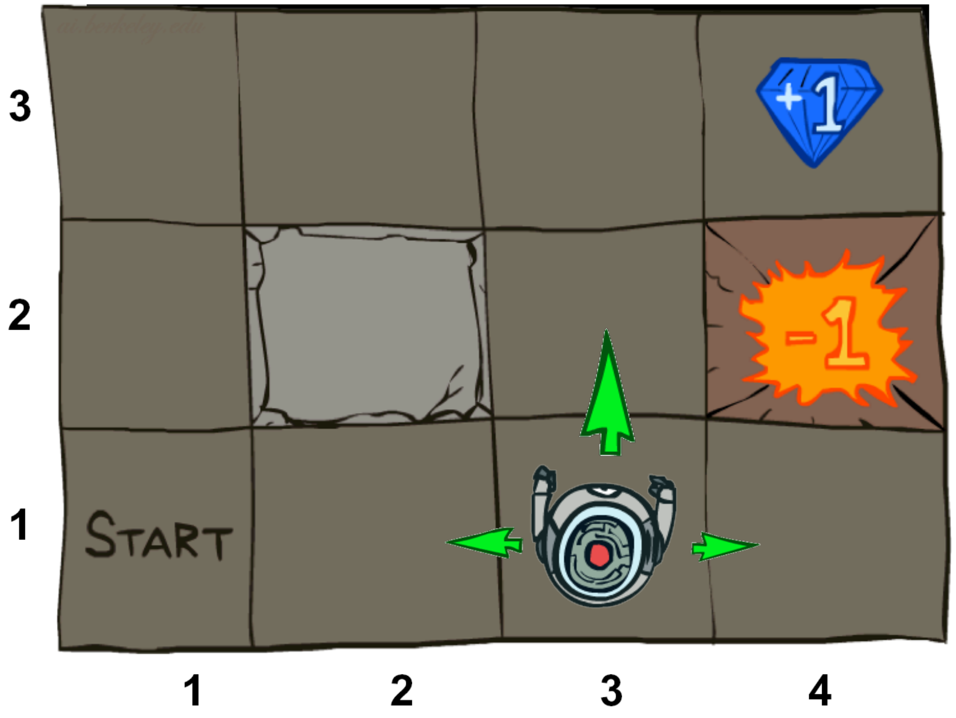
AI machine takes 2-0 lead against South Korea's Lee Sedol, putting its owners one victory away from $1m prize

AlphaGo beats world champion Lee Sedol in the second match of the Go tournament

Google's Go-playing machine has scored a second victory against the best human

Markov Decision Processes

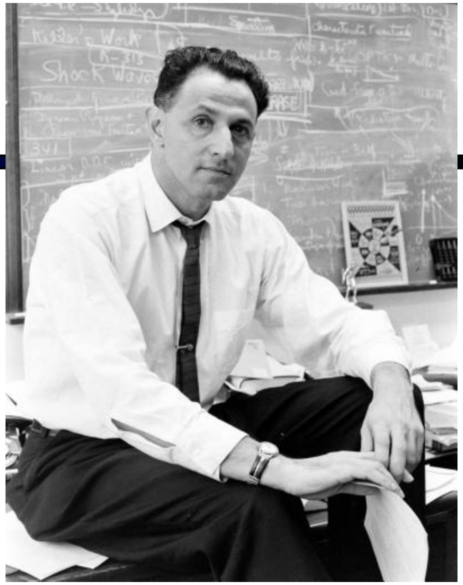
- An MDP is defined by:
  - A set of states $s \in S$
  - A set of actions $a \in A$
  - A transition function T(s, a, s')
    - Probability that a from s leads to s', i.e., P(s'| s, a)
    - Also called the model or the dynamics
  - A reward function R(s, a, s')
    - Sometimes just R(s) or R(s')
  - A start state
  - Maybe a terminal state



- MDPs are non-deterministic search problems
  - One way to solve them is with expectimax search
  - We'll have new tools soon

[Demo – gridworld manual intro (L8D1)]

The Bellman Equations

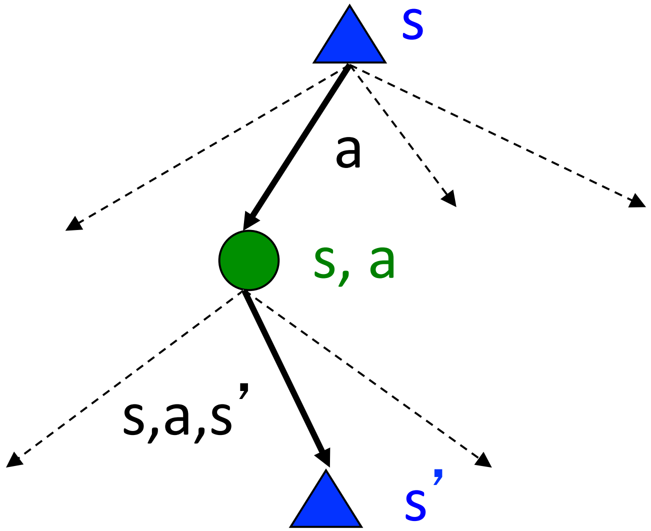


(1920-1984)

- Definition of "optimal utility" via expectimax recurrence gives a simple one-step lookahead relationship amongst optimal utility values

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

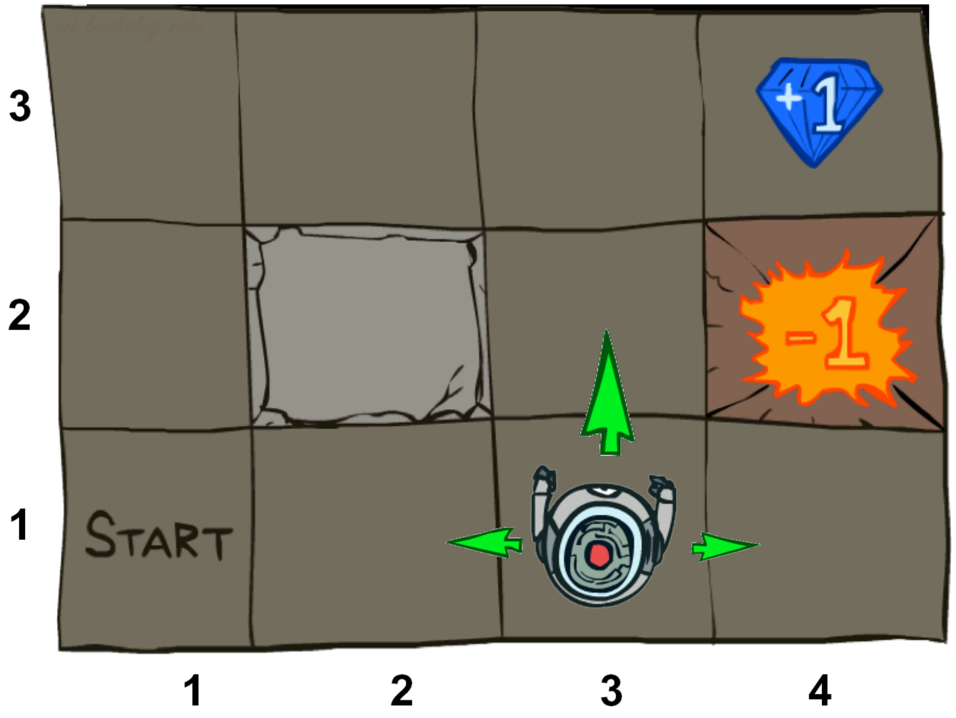
$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$



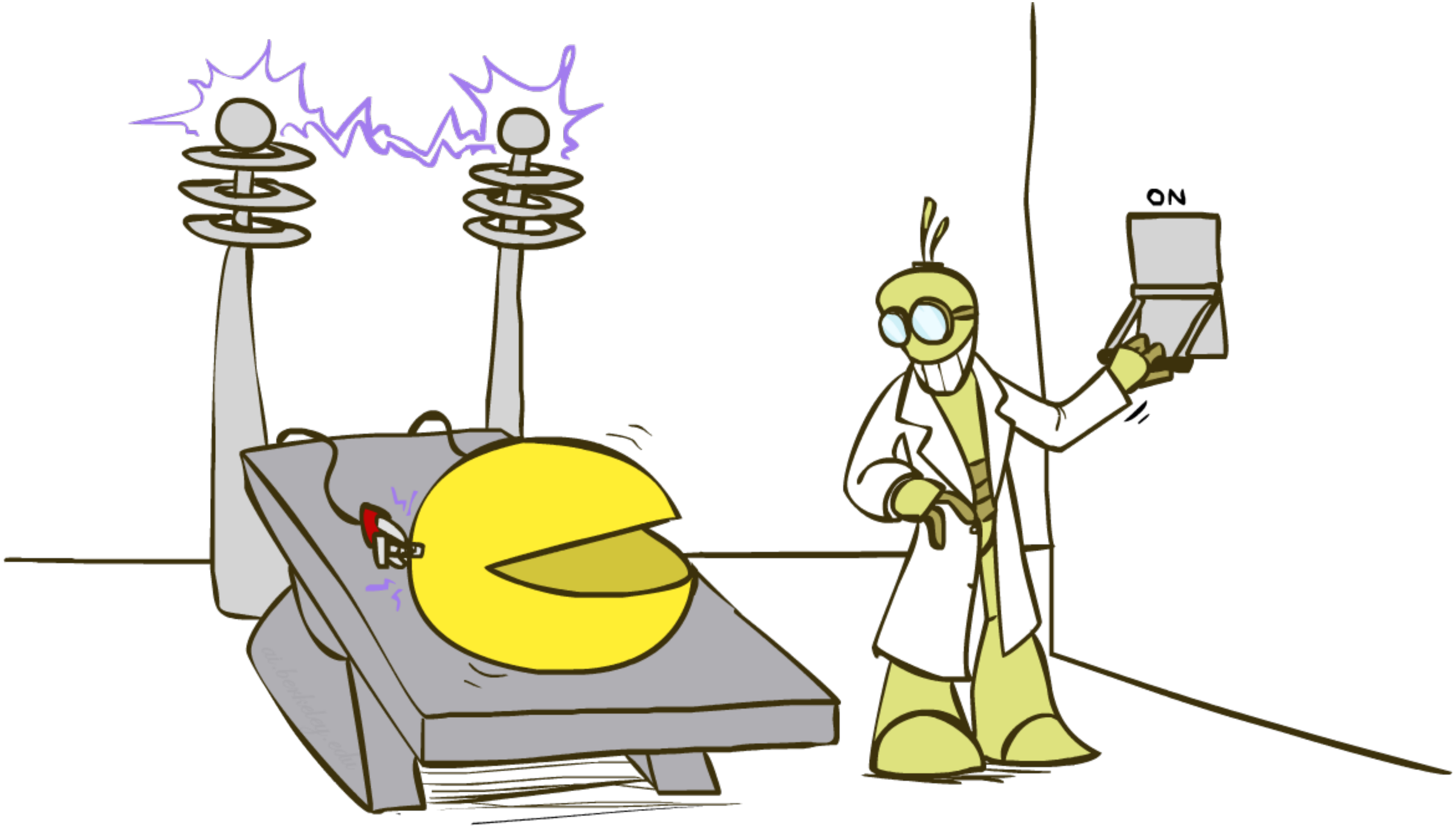
- These are the Bellman equations, and they characterize optimal values in a way we'll use over and over

Partially Observable Markov Decision Processes

- An MDP is defined by:
  - A set of states $s \in S$
  - A set of actions $a \in A$
  - A set of observation $o \in O$
  - A transition function T(s, a, s')
    - Probability that a from s leads to s', i.e., P(s' | s, a)
    - Also called the dynamics
  - A observation function O(s, a, o)
    - Probability of observing o, i.e., P(o | s, a)
    - T and O together are often called the *model*
  - A reward function R(s, a, s')
    - Sometimes just R(s) or R(s')
  - A start state
  - Maybe a terminal state

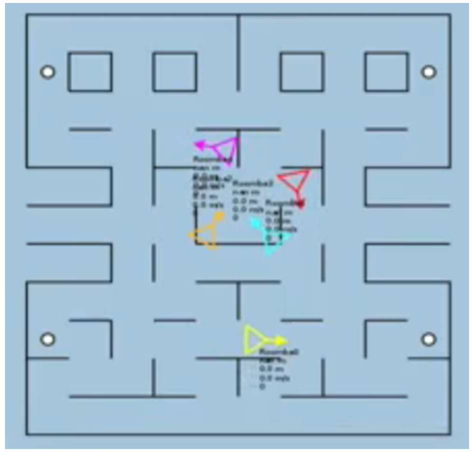


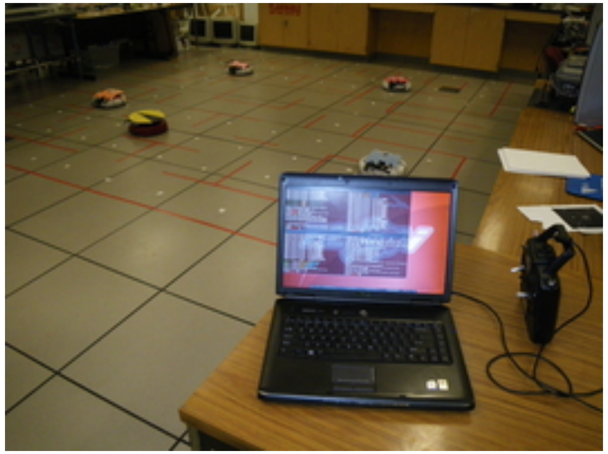
Pac-Man Beyond the Game!



Pacman: Beyond Simulation?

ROOMBA PACMAN





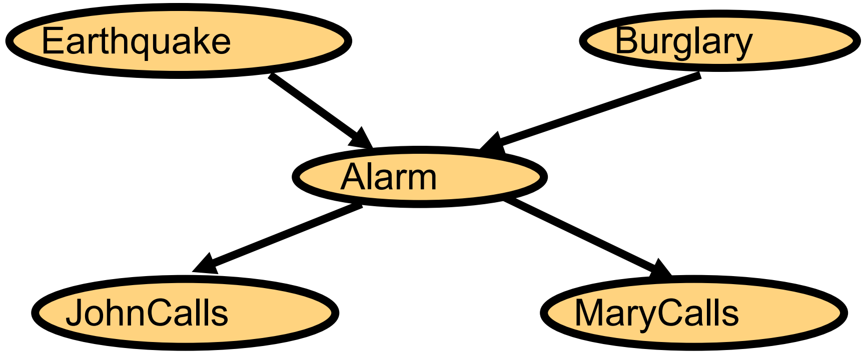
Students at Colorado University: http://pacman.elstonj.com

[VIDEO: Roomba Pacman.mp4]

Pacman: Beyond Simulation!

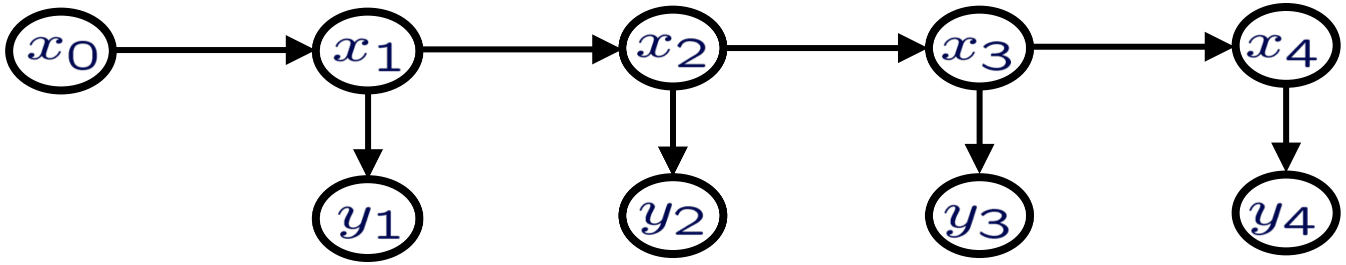
KR&R: Probability

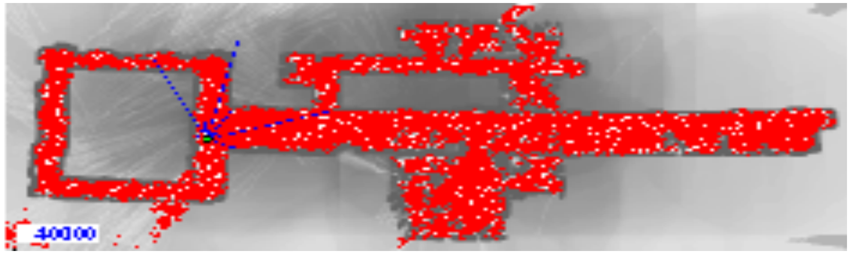
- Representation: Bayesian Networks
  - encode probability distributions compactly
    - by exploiting conditional independences
- Reasoning
  - Exact inference: var elimination
  - Approx inference: sampling based methods
    - rejection sampling, likelihood weighting, MCMC/Gibbs



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

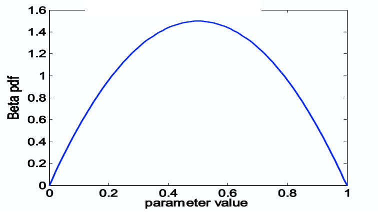
- We focused on Naïve Bayes and Perceptron, but you could also:
- Learn Structure of Bayesian Networks
  - Search thru space of BN structures
- Learn Parameters for a Bayesian Network
  - Fully observable variables
    - Maximum Likelihood (ML), MAP & Bayesian estimation
    - Example: Naïve Bayes for text classification
  - Hidden variables
    - Expectation Maximization (EM)

Bayesian Learning

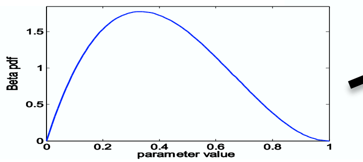
Use Bayes rule:

Data Likelihood

Prior



Posterior

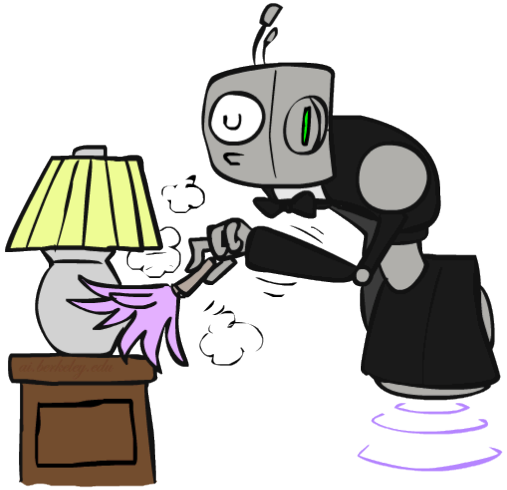


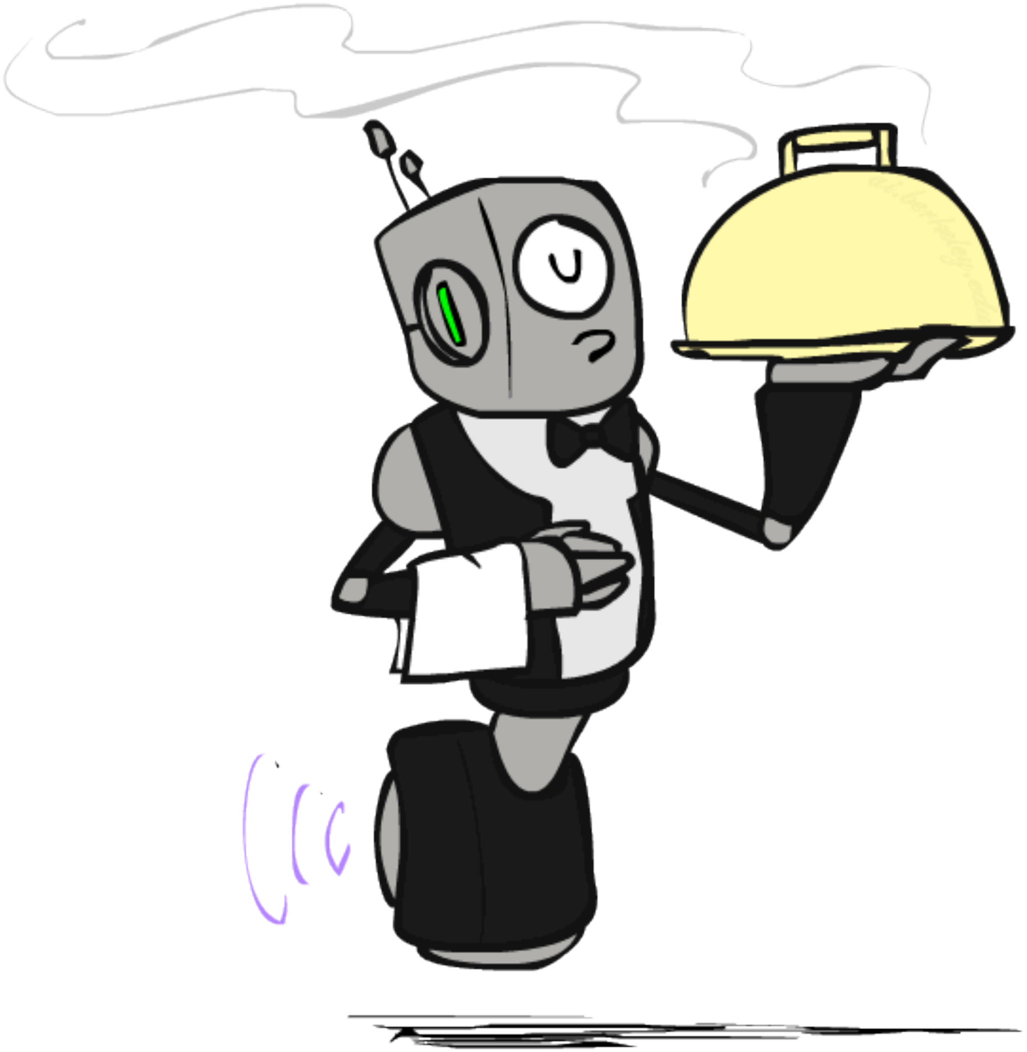
$$P(Y \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid Y)\, P(Y)}{P(\mathbf{X})}$$

Normalization

Or equivalently: $P(Y \mid \mathbf{X}) \propto P(\mathbf{X} \mid Y)\, P(Y)$

Personal Robotics





PR2 (autonomous)

[VIDEO: 5pile_200x.mp4]

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

Autonomous tying of a knot for previously unseen situations

[VIDEO: knots_apprentice.mp4]

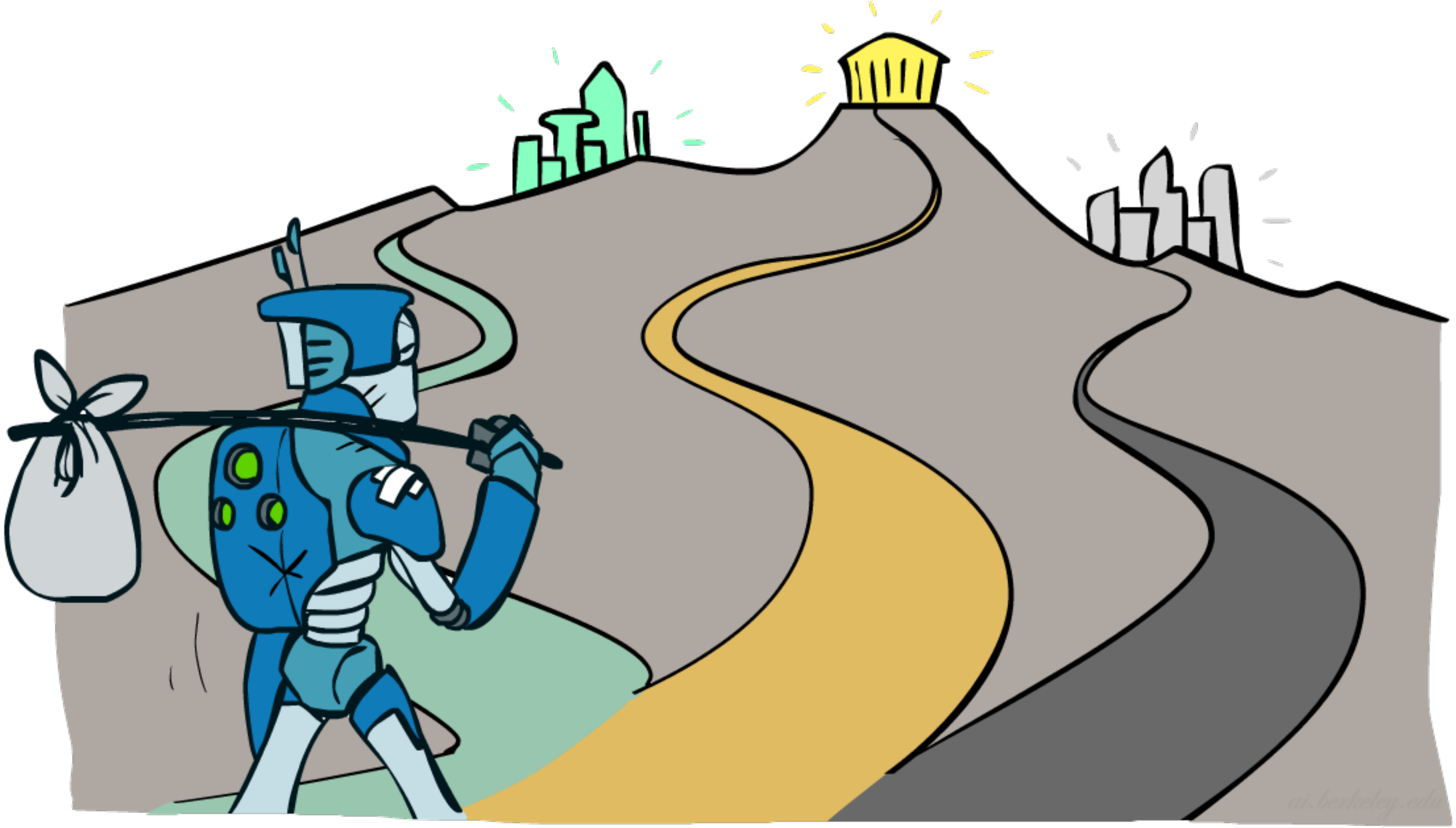
[Schulman, Ho, Lee, Abbeel, 2013]

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

Experiment: Suturing

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

Where to Go Next?



That’s It!

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