Model-Based Reinforcement Learning

Michael L. Littman

Rutgers University

Department of Computer Science

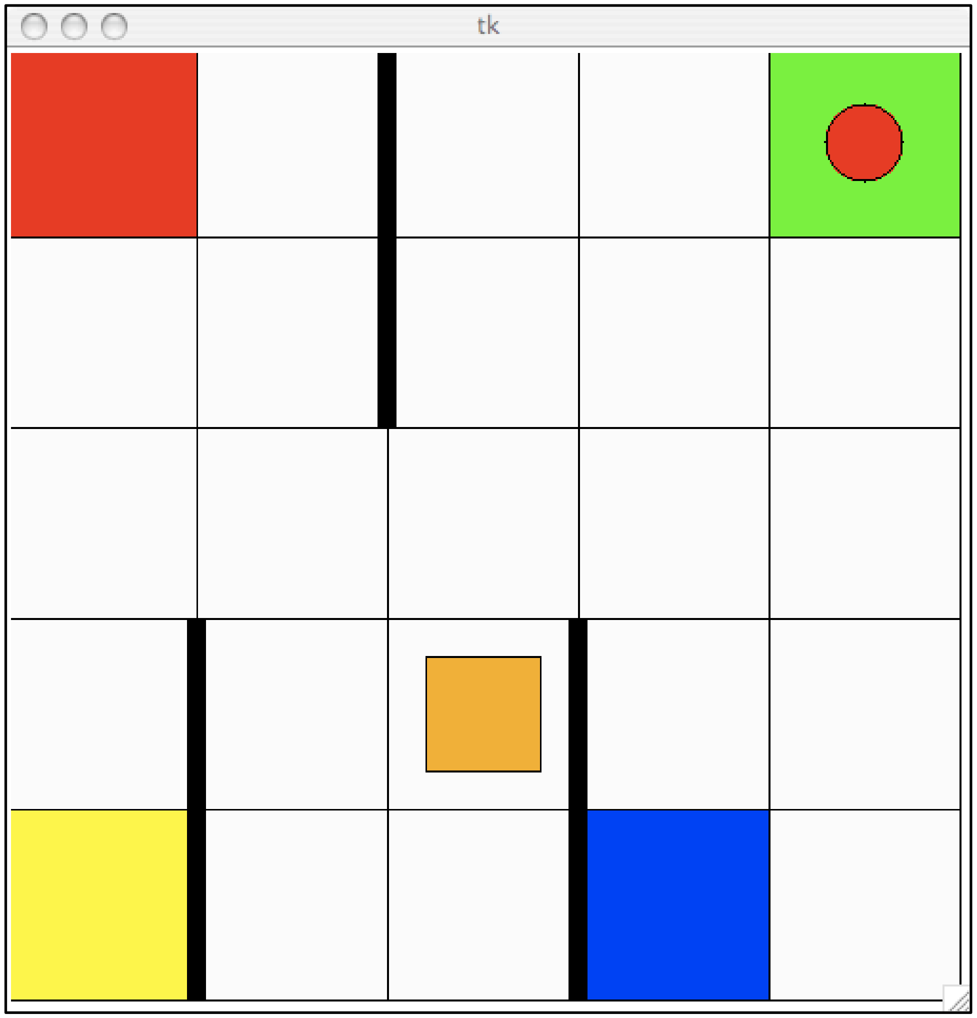
Rutgers Laboratory for Real-Life Reinforcement Learning

Topics

- Introduction
  - MDPs
  - Reinforcement Learning
- Model-based RL
- Efficient Exploration
  - PAC-MDP
  - KWIK
- Bayesian RL
  - Near-Bayesian
  - PAC-MDP
- Planning
  - Nesting approaches
  - UCT
- Model-based People?

Start With Game...

- up
- down
- left
- right
- A
- B



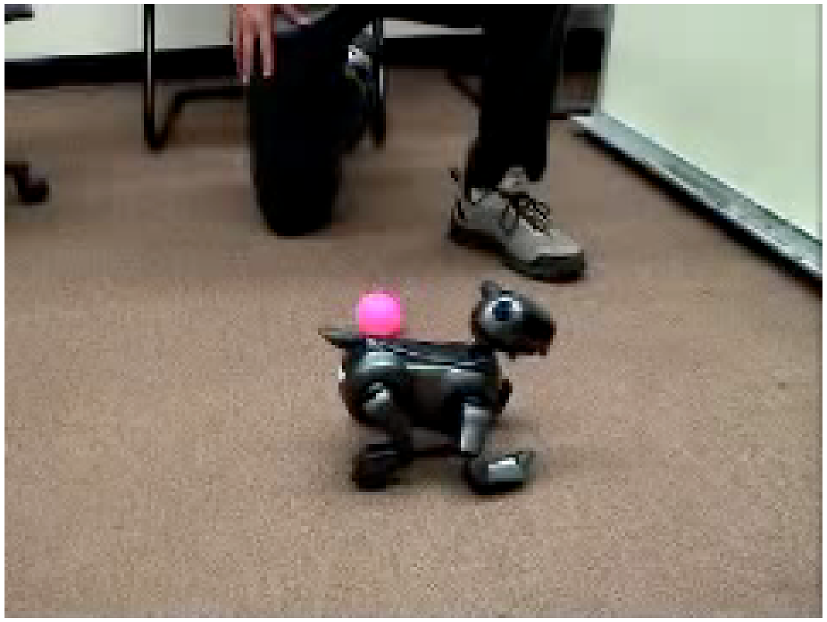
Find The Ball: Elements of RL

In reinforcement learning:

- agent interacts with its environment
- perceptions (state), actions, rewards [repeat]
- task is to choose actions to maximize rewards
- complete background knowledge unavailable

Learn:

- which way to turn
- to minimize time
- to see goal (ball)
- from camera input
- given experience.



Problem To Solve

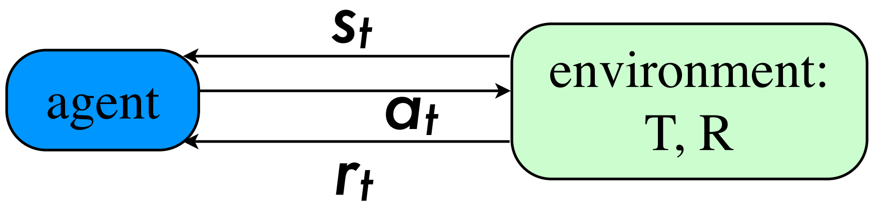
Three core issues in the dream RL system.

- <u>generalize experience</u>
  - use knowledge gained in similar situations
  - "learning"
- <u>sequential decisions</u>
  - deal properly with delayed gratification
  - "planning"
- <u>exploration/exploitation</u>
  - must strike a balance
  - unique to RL?

Markov Decision Processes

Model of sequential environments (Bellman 57)

- *n* states, *k* actions, discount $0 \leq \Upsilon \leq 1$
- step *t*, agent informed state is s_t, chooses a_t
- receives payoff r_t; expected value is $R(s_t, a_t)$
- probability that next state is s' is $T(s_t, a_t, s')$

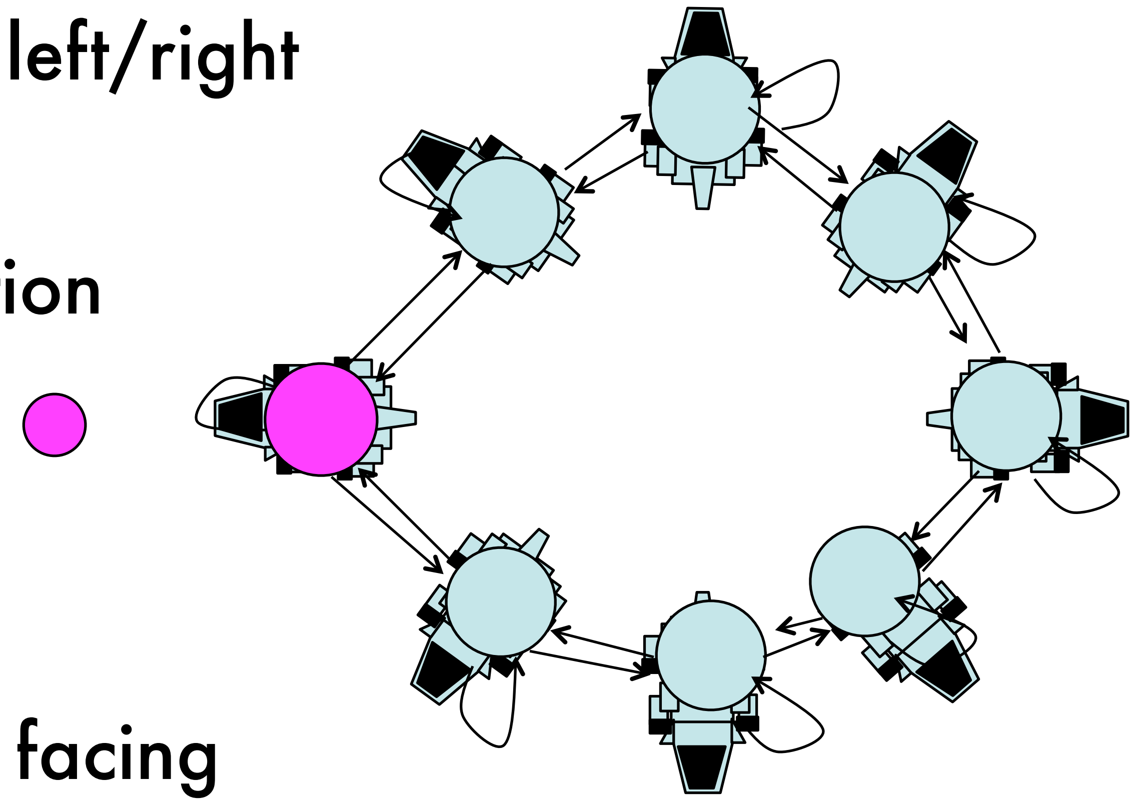


$$Q(s,a) = R(s,a) + \gamma \textstyle\sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$$

- Optimal behavior is $a_t = \text{argmax}_a\, Q(s_t,a)$
- R, T unknown; some experimentation needed

Find the Ball: MDP Version

- Actions: rotate left/right
- States: orientation
- Reward: $+1$ for facing ball, 0 otherwise



Families of RL Approaches

policy search | value-function based | model based

$$\begin{tikzcd}
s \arrow[d] & & s \arrow[d] & a \arrow[dl] & & s \arrow[d] & a \arrow[dl] \\
\boxed{\pi} \arrow[d] & \Leftarrow & \boxed{Q} \arrow[d] & & \Leftarrow & \boxed{T, R} \arrow[d] \arrow[dr] & \\
a & & v & & & s' & r
\end{tikzcd}$$

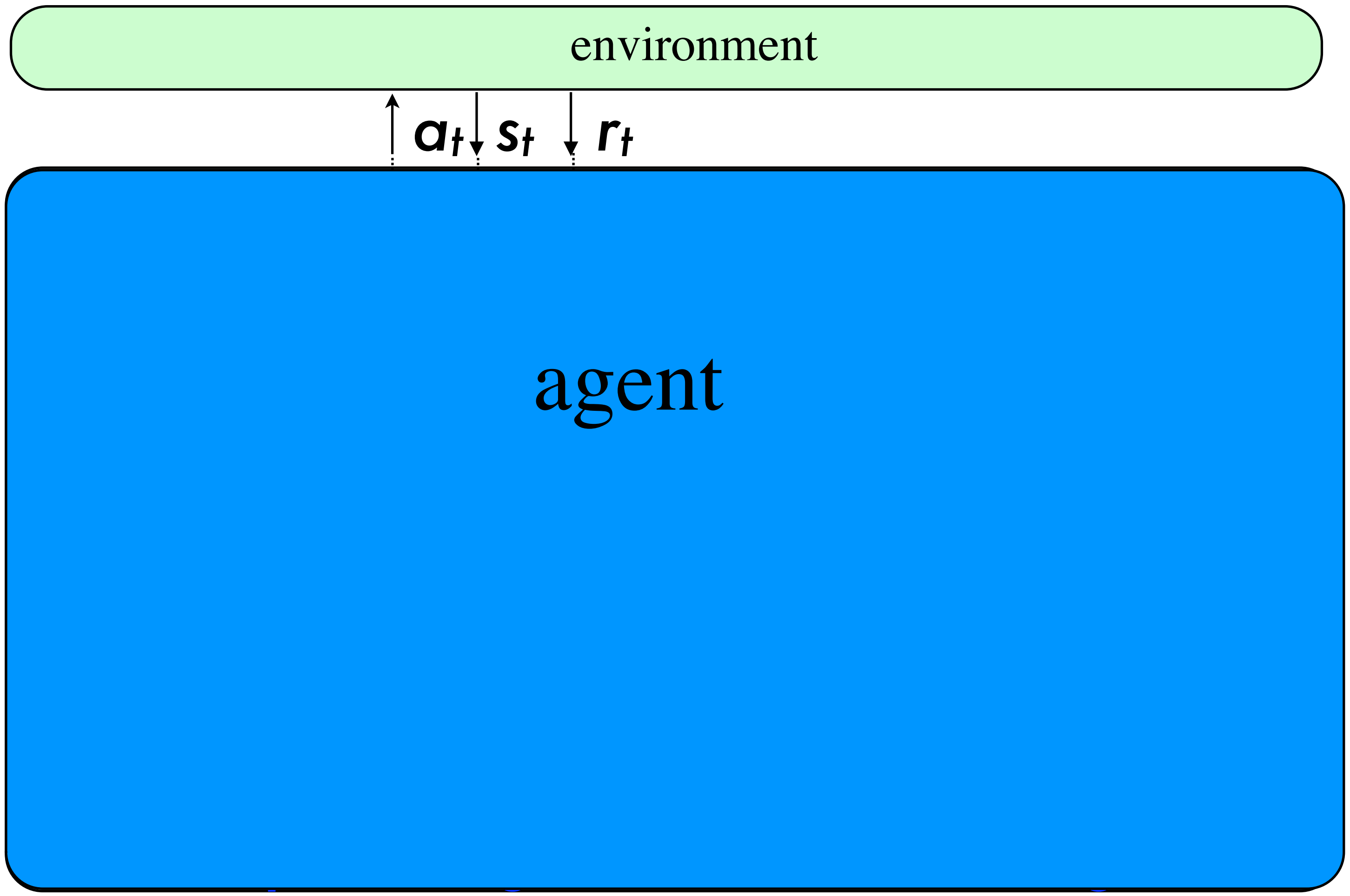
Search for action that maximizes value

Solve Bellman equations

More direct use, less direct learning

More direct learning, less direct use

Model-based RL Schematic



PAC-MDP Reinforcement Learning

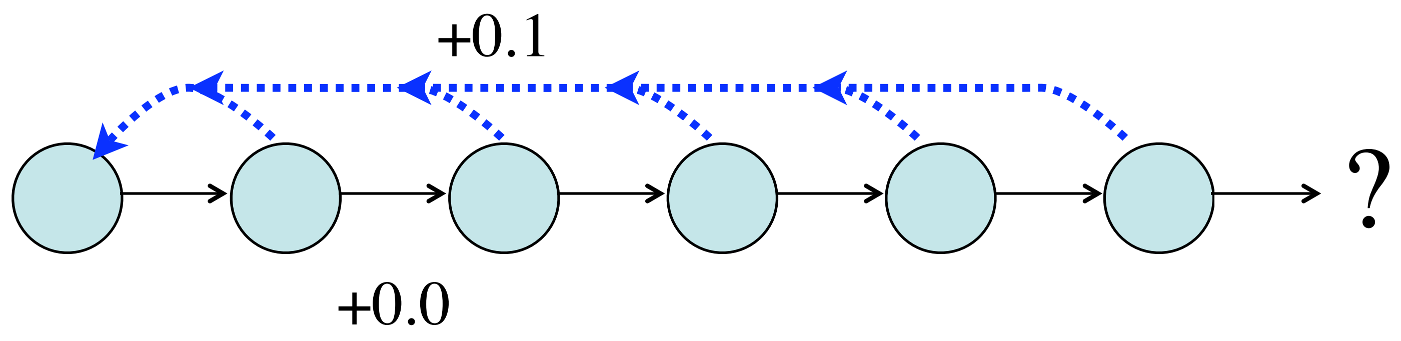
PAC: Probably approximately correct (Valiant 84)

Extended to RL (Fiechter 95, Kakade 03, etc.).

- Given $\epsilon>0$, $\delta>0$, k actions, n states, Υ.
- We say a strategy makes a mistake each timestep t s.t. $Q(s_t,a_t) < \max_a Q(s_t,a)-\epsilon$.
- Let m be a bound on the number of mistakes that holds with probability $1-\delta$.
- Want m poly in k, n, $1/\epsilon$, $1/\delta$, $1/(1-\Upsilon)$.

Must balance exploration and exploitation!

Model-based Can Be PAC-MDP



- Behavior differs depending on assumption

	truth: ? = low	truth: ? = high
assume: ? = low	ignore ?, optimal	ignore ? suboptimal!
assume: ? = high	visit ?, explore	visit ?, optimal

← No PAC-MDP guarantee

← PAC-MDP if not too much exploration

Model-driven Exploration

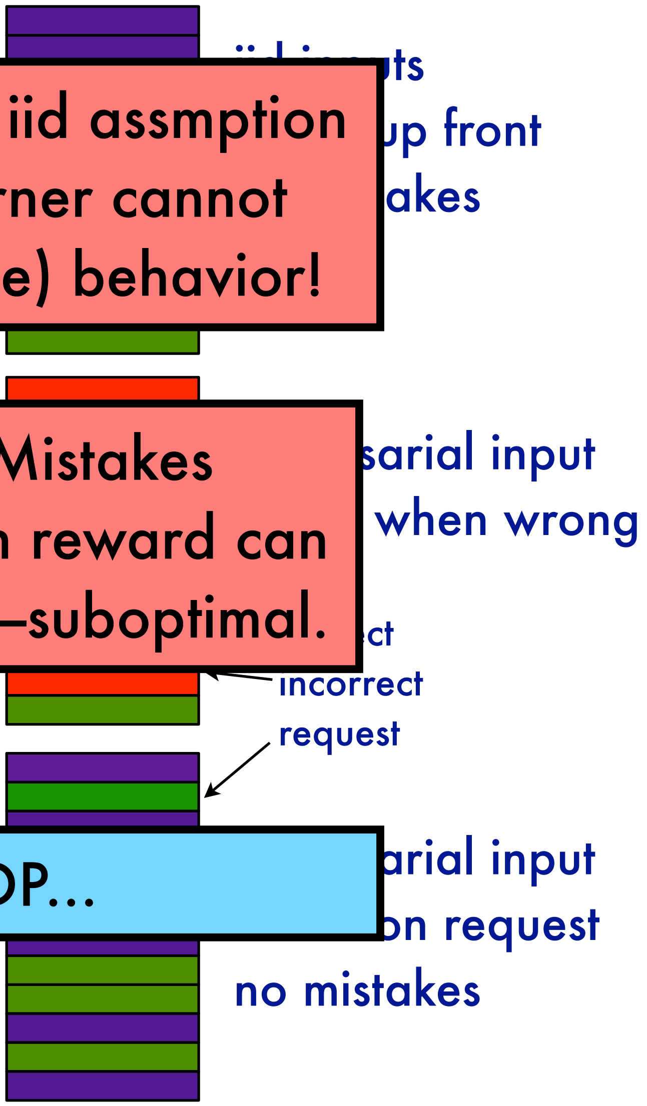
- A *model* is $\mathcal{R}(s,a)$ and $\mathcal{T}(s,a,s')$.
- Model-based approach:
  - learn a model of the environment (approximately, distinguishing known/unknown transitions).
  - augment model w/ bonus for unknown transitions.
  - plan behavior wrt the augmented model.
  - repeat

Key that learner "knows what it knows" (KWIK).

What learning setting is appropriate?

3 Models for Learning Models

- PAC: Inputs drawn fr distribution. Observ inputs. For future inp from the distribution,
- Mistake bound: Inpu online. For each, pr If mistake, observe l more than *m* mistake
- KWIK: Inputs presented online. For each, can predict output or say "I don't know" a label. No mistakes, but can say "I don't know" *m* times.



Not PAC-MDP. iid assmption implies that learner cannot improve (change) behavior!

Not PAC-MDP. Mistakes mean that a high reward can be assumed low—suboptimal.

Can be PAC-MDP...

KWIK-Rmax Proof

- Provides PAC-MDP guarantee in flat MDPs (Kearns & Singh 02, Brafman & Tennenholtz 02).
- Key ideas:
  - Simulation lemma: Optimal actions for approximate model near-optimal in real model.
  - Explore or exploit lemma: If can't reach unknown states quickly, can achieve near-optimal reward.
- Unflat: factored dynamics (Kearns & Koller 99), metric spaces (Kakade et al. 03), KWIK (Li 09).

Time to learn depends on KWIK bound.

KWIK Learn a Probability

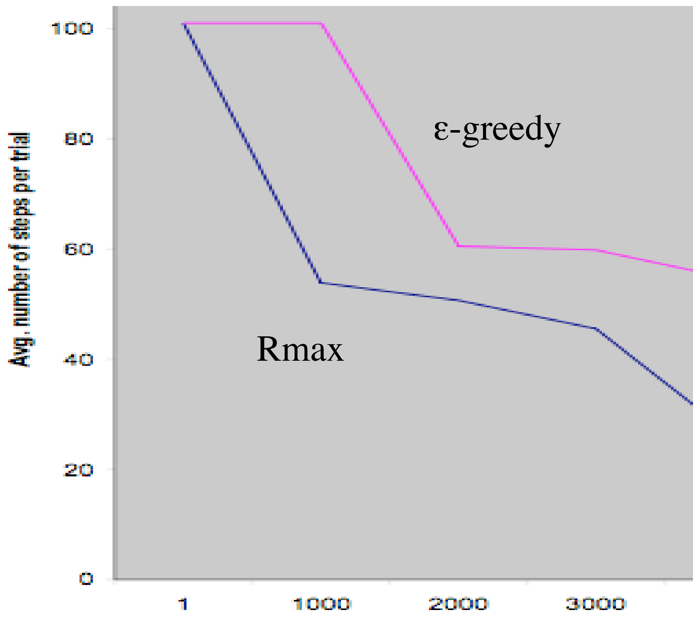
- Given *m* trials, *x* successes, $\hat{p} = x/m$
- Hoeffding bound:
  - Probability of an empirical estimate of a random variable in the range [*a*,*b*] based on *m* samples being more than ϵ away from the true value is bounded by $\exp\left(-\frac{2m\epsilon^2}{(b-a)^2}\right)$
- So, can KWIK learn a transition probability:
  - say "I don't know" until *m* is big enough so that $\hat{p}$ is ϵ-accurate with probability 1−δ.

Other Things to KWIK Learn

- coin probability
- vector of outputs, each KWIK learnable
  - multinomial probability (dice learning)
- mapping from input partition to outputs, partitions known, mappings KWIK learnable
  - That's a standard transition function (*s*,*a* to vector of coins) (Li, Littman, Walsh 08).
- Also, union of two KWIK learnable classes.

RMAX Speeds Learning

Task: Exit room using bird's-eye state representation.

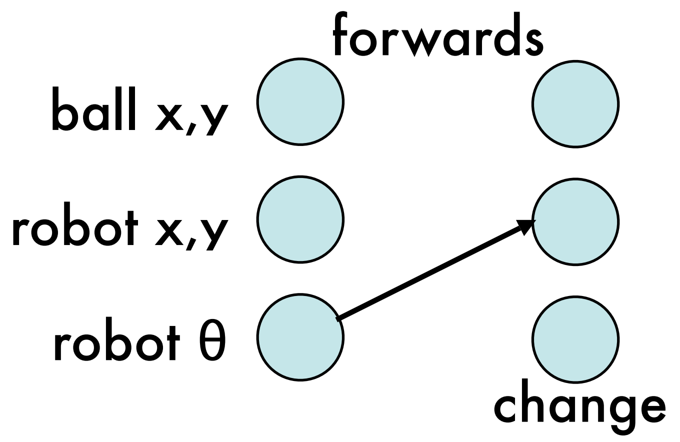


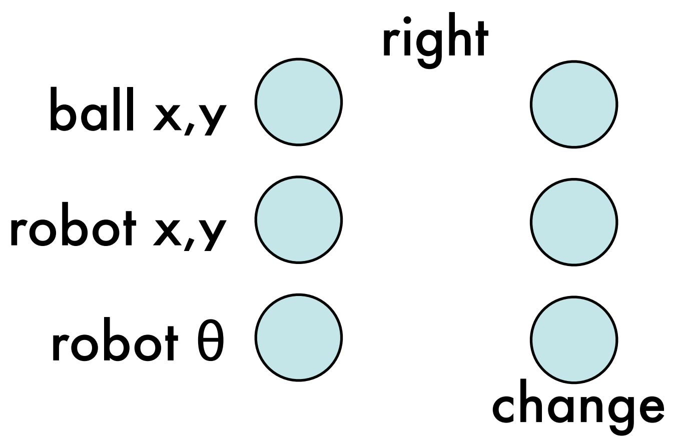
Details: Discretized 15x15 grid x 18 orientation (4050 states); 6 actions: forward, backward, turn L ,turn R, slide L, slide R.

(Nouri)

Generalizing Transitions

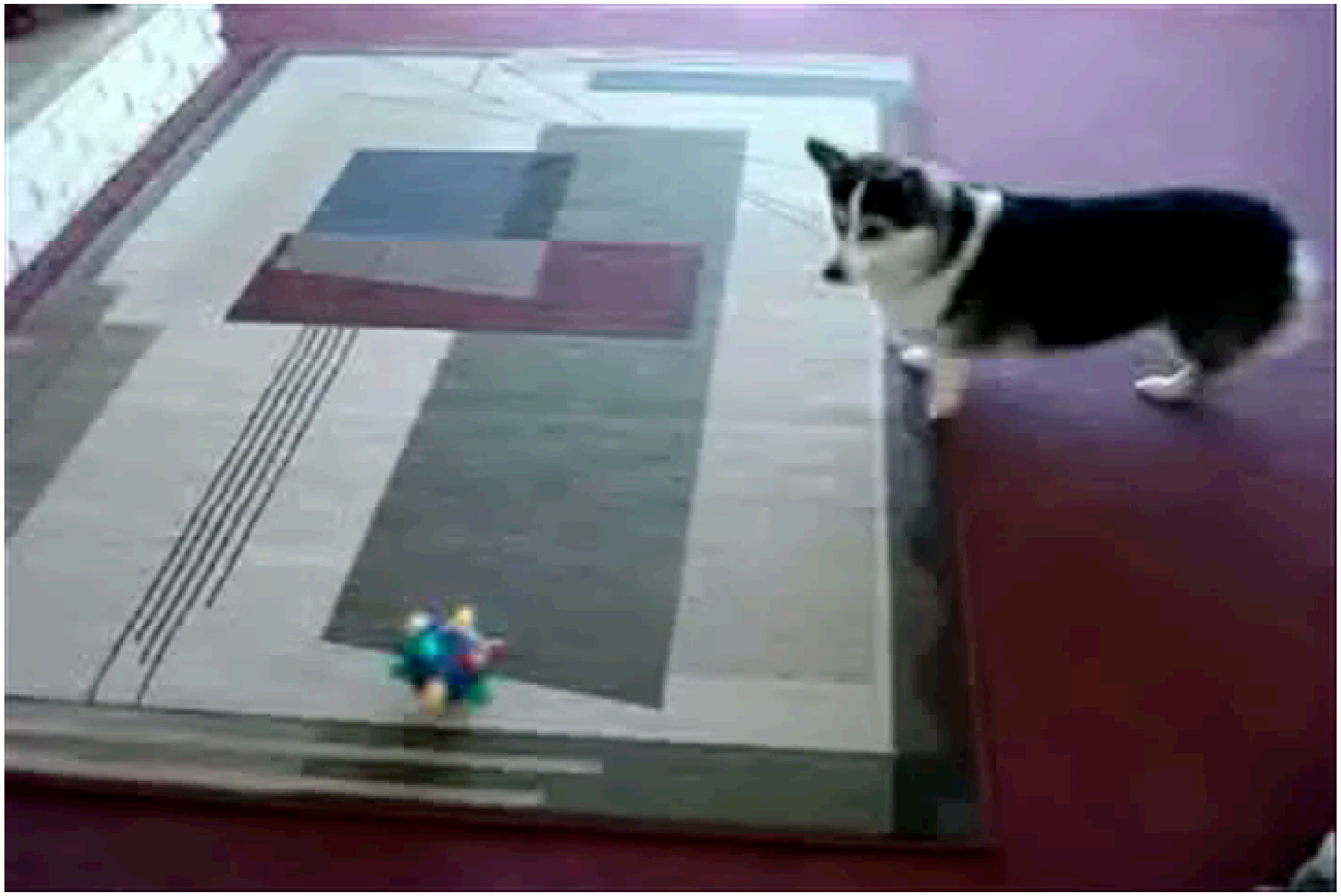
- Flat MDPs, states viewed as independent.
  - Transition knowledge doesn't transfer.
- DBN representation shares structure.
  - Learn components independently, KWIK!





Less to learn, faster to behave well.

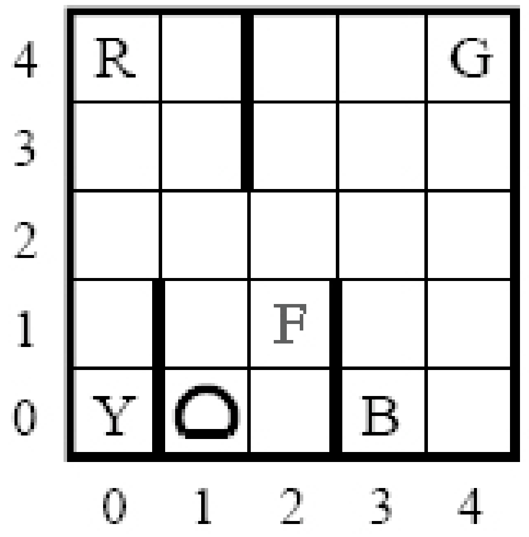
Continuous-state DBN



(Nouri)

World of Objects

- Objects in taxi:
  - taxi (location)
  - passenger (location/in taxi)
  - walls (location)
  - destination (location)



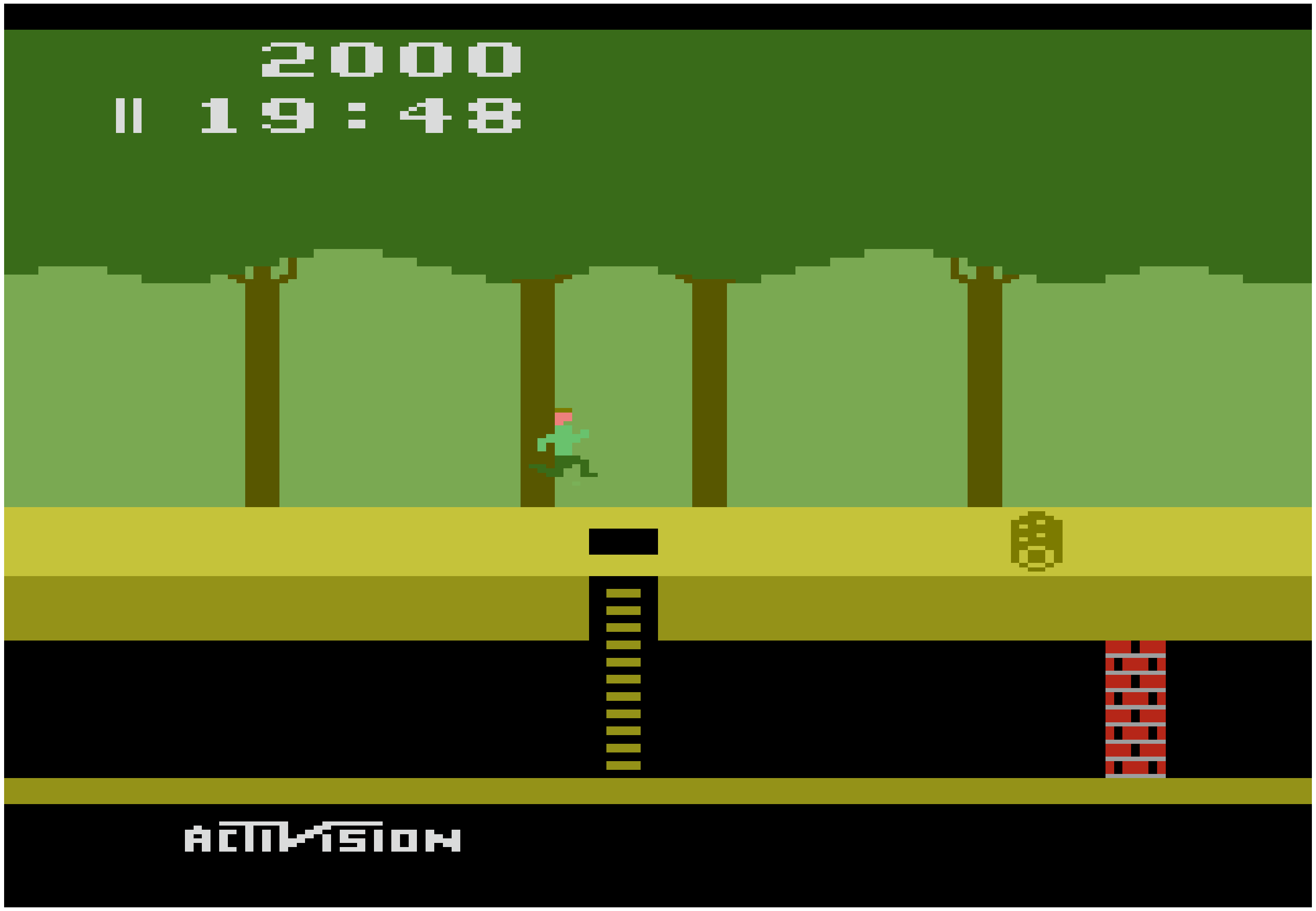
- Not *states* or state *features*, instead try *objects* and object *attributes*.
- Model: What happens when objects interact?
- More "human like" exploration.

Comparing Taxi Results

- North, not touchN(taxi,wall) → taxi.y++
- Drop, pass.in, touch(taxi, dest) → ¬pass.in
- KWIK bound: poly in types (exp in condition)
- Taxi: How long until optimal behavior?

Exploration style	Algorithm	# of steps
ϵ greedy	Q-learning	47157
count on states	Flat Rmax	4151
count on features	Factored Rmax	1839
count on interaction	Objects	143
whatever people do	People	50

Pitfall!



A childhood dream fulfilled... (Diuk, Cohen)

Structure Learning in DBNs

- Unknown structure fundamentally different.
- How can you keep statistics if you don't know what they depend on?
- Can be solved using a technique for a simpler "hidden bit" problem:
  - *n*-bit input, one bit (unknown) controls output
  - one output distribution if bit is on, another if off
  - Find DBN structure by same idea: one parent set controls output...

Hidden-Bit Problem

Assume the simpler deterministic setting.

Output is copy or flip of one input.

• 0110 → 0	1101 → 1	0000 → 1
• 1101 → 1	0011 → 0	1111 → 0
• 1000 → 1	1110 → 0	1100 → 1

Is it 0, 1, or “I don’t know”?

If noisy, can’t predict with each bit position separately, don’t know which to trust. Can learn about all 2^n bit patterns separately, but that’s too much.

Hidden-bit Problem via KWIK

- Can observe predictions to figure out which of *k* "adaptive meteorologists" to trust (Strehl, Diuk, Littman 07; Diuk et al. 09).
- Solvable with bound of $O\left(\frac{k}{\epsilon^2}\ln\frac{k}{\delta}\right)+\sum_{i=1}^{k}\zeta_i\left(\frac{\epsilon}{8},\frac{\delta}{k+1}\right)$
- By considering all *k*-size parent sets, get a structure-learning algorithm with a KWIK bound of

$$\kappa = O\left(\frac{n^{D+3}AD}{\epsilon^3(1-\gamma)^6}\ln\frac{nA}{\delta}\ln\frac{1}{\epsilon(1-\gamma)}\right)$$

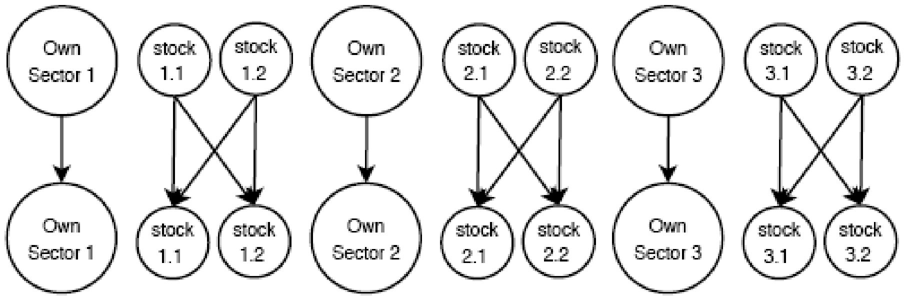
Artificial Stock Example

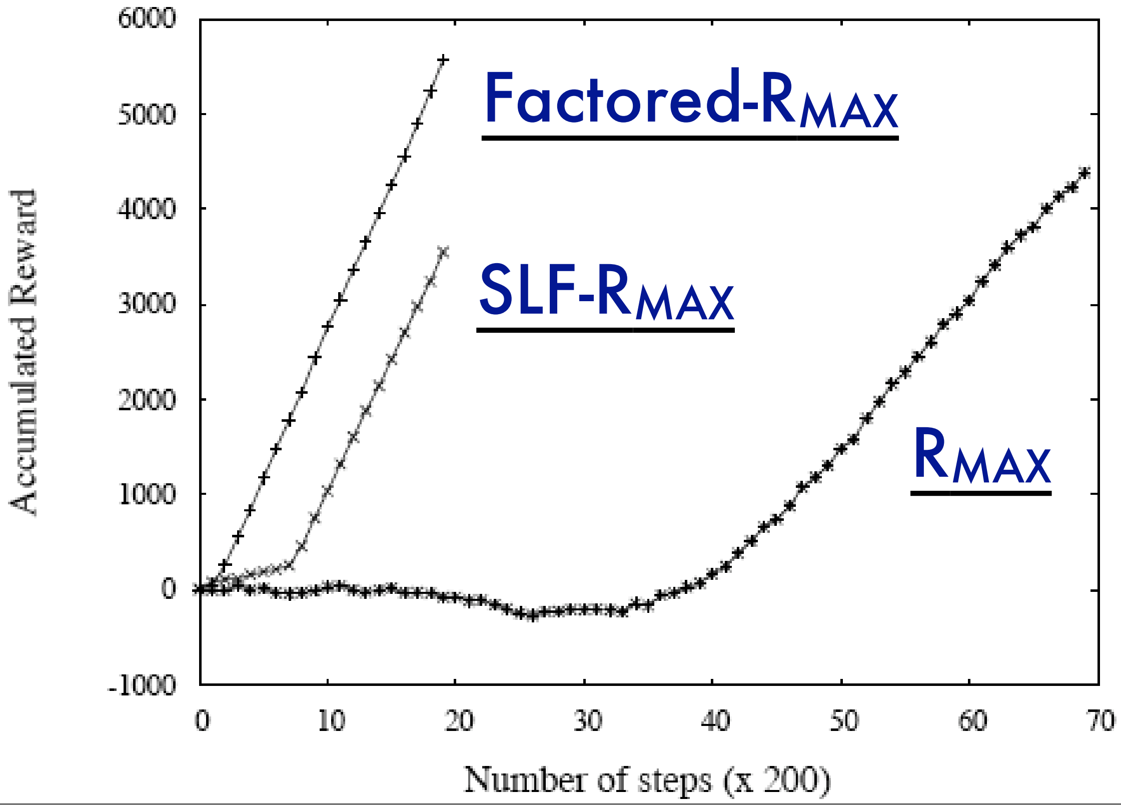
Discovers the structure and exploits it much faster than RMAX can learn the MDP.

Factored-RMAX: Knows DBNs

SLF-RMAX: Knows size of parent sets

RMAX: It's an MDP





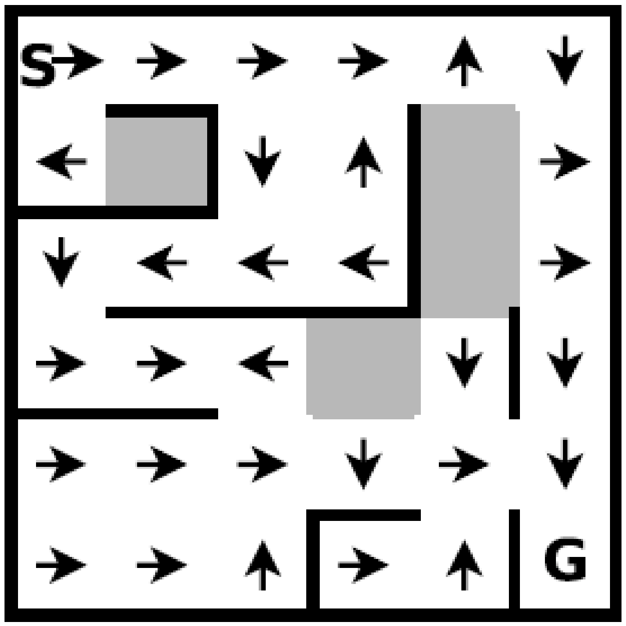
Many Learnable Problems

Many hypothesis classes KWIK learnable:

- coin flip probability
- Dynamic Bayes net probabilities given graph
- *k* Dynamic Bayes net
- *k* Meteorologist problem
- *k*-CNF
- *k*-depth decision tree
- unions of KWIK-learnable classes
- *k* feature linear function

Beyond Worst Case

- KWIK learns specific hypothesis class.
  - If too broad, learning too slow.
  - If too narrow, learning fails.
- Bayesian perspective:
  - Start with a prior over models.
  - Maintain a posterior.
  - Optimize for *probable* instead of just *possible*.
  - Similar states have similar dynamics. Probably.
- Bayesian view can drive exploration.

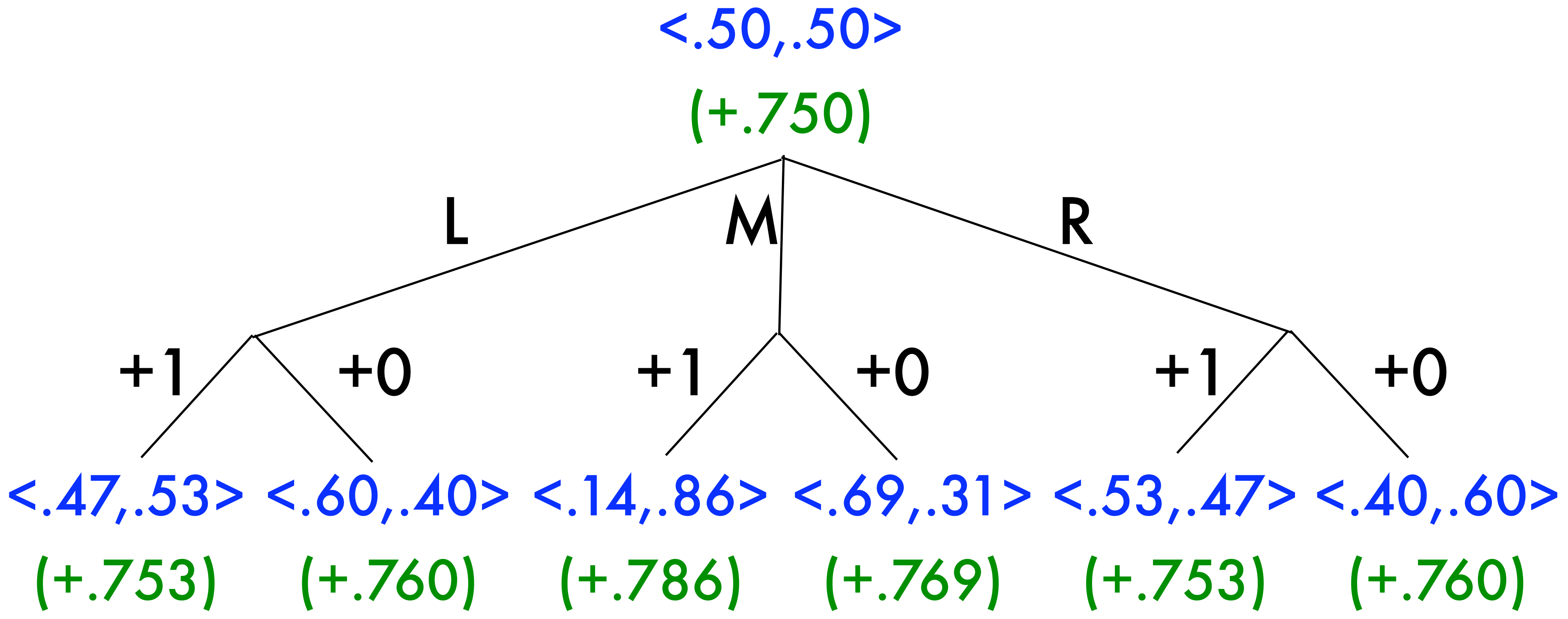


Bayes Optimal Exploration

- With a Bayesian representation of models, we can plan in the space of *posteriors*.
  - Can use posterior to evaluate the likelihood of any possible outcome of an action.
  - Can model how that outcome changes posterior.
  - Can choose actions that truly maximize expected reward: No artificial distinction between exploring and exploiting or learning and acting!
- Hideously intractable except in special cases (bandits, short horizons).

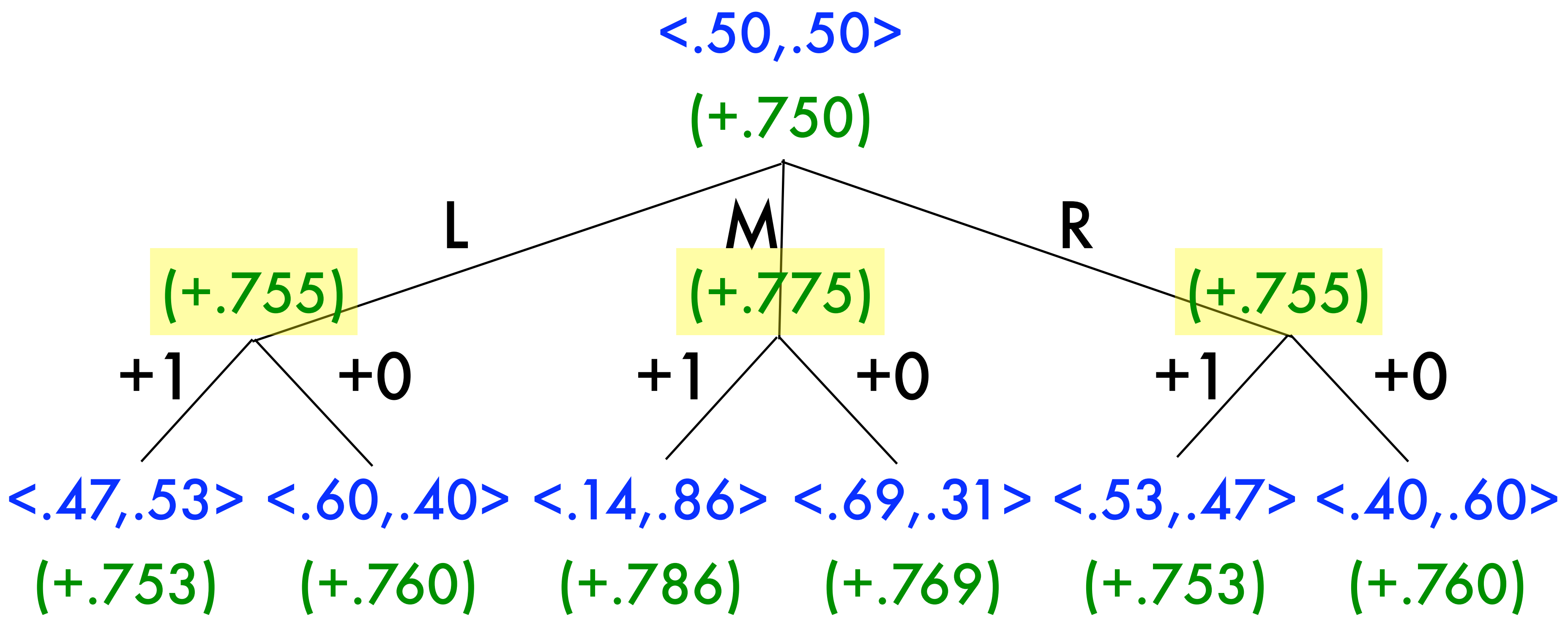
Concrete Example

- MDP has one state, 3 actions (bandit)
  - X: {.7 .1 .8}, Y: {.8 .6 .7}, $\gamma = 0.8$
  - Prior: <.50,.50> (1/2 X, 1/2 Y)



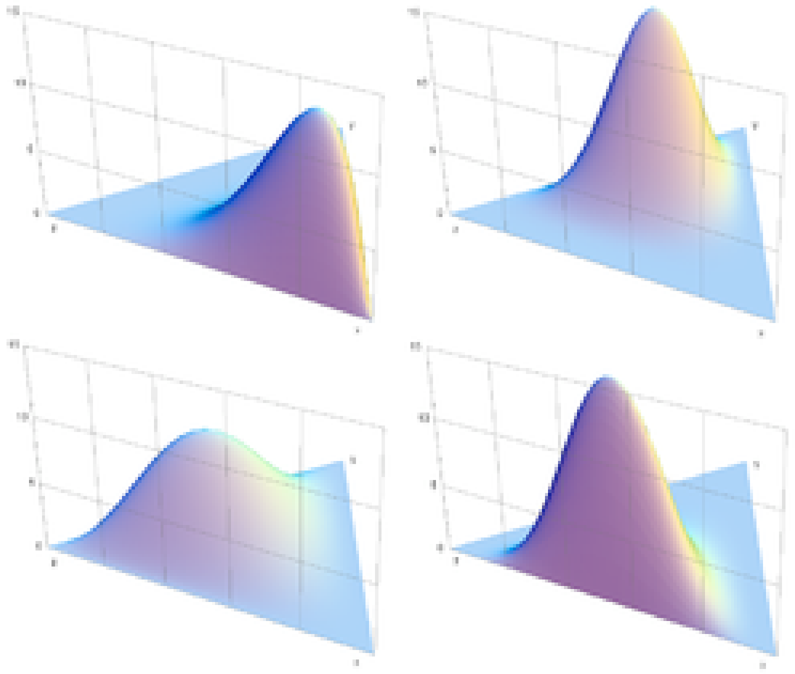
Concrete Example

- MDP has one state, 3 actions (bandit)
  - X: {.7 .1 .8}, Y: {.8 .6 .7}, γ = 0.8
  - Prior: <.50,.50> (1/2 X, 1/2 Y)



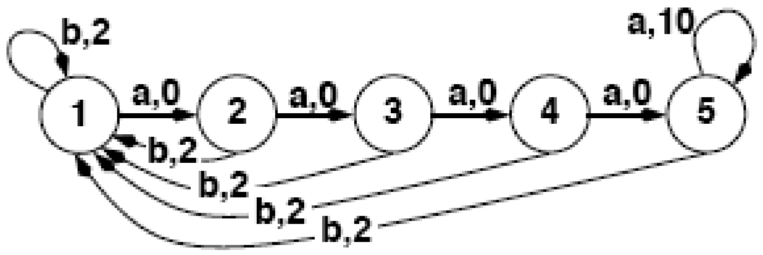
Representing Posteriors

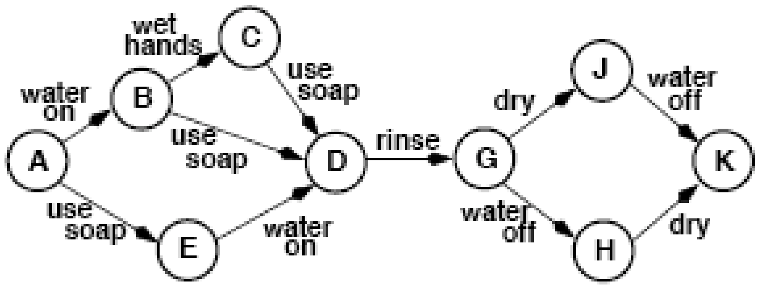
- T: *s*,*a*→multinomial over states
- If independent for each *s*,*a*: Dirichlet!
- Keep counts for each observed outcome.
- Can recover uncertainty in overall estimate.
- Unlike example, distribution over an infinite set.



Bayes Optimal Plans

- Many attempts (Duff & Barto 97; Dearden et al. 99)
- State of the art, BEETLE (Poupart et al. 06).
  - Latest ideas from solving continuous POMDPs
  - α functions are multivariate polynomials + PBVI
  - Can exploit "parameter tying" prior.
  - Near optimal plan in "combination lock".
  - Less optimal in bigger problem.
  - Planner outputs exploration scheme.



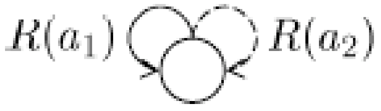


Near Bayes Optimal Behavior

- Recall PAC-MDP, whp makes few mistakes.
- Near Bayesian: mistakes are actions taken with values far from Bayes optimal.
- Bayesian Exploration Bonus (Kolter & Ng 09) keeps mean of posterior and adds $1/n$ bonus to actions taken n times.
  - BEB is computationally simple.
  - BEB is Near Bayesian.
  - BEB is not PAC-MDP, though...

Bayes Optimal Not PAC-MDP

- Examples where Bayes optimal does not find near optimal actions (Kolter & Ng 09; Li 09)



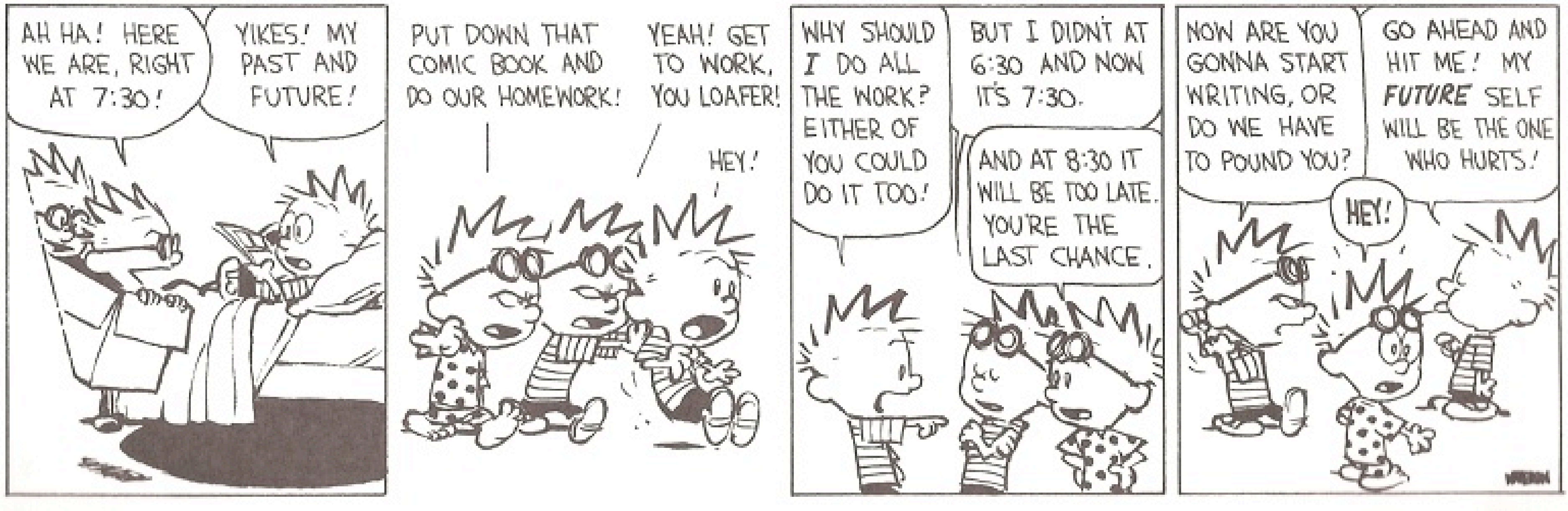
$a_1 \longrightarrow$ +1/2 with certainty

$a_2 \dashrightarrow$ +0 wp p

+1/2+ε(1-γ) wp 1-p

- Bayes optimal approach chooses a_1 forever even though not ε-optimal (for $p>2\varepsilon$).
- So, even if the two models are equally likely, Bayes optimal doesn't bother learning about the better option!

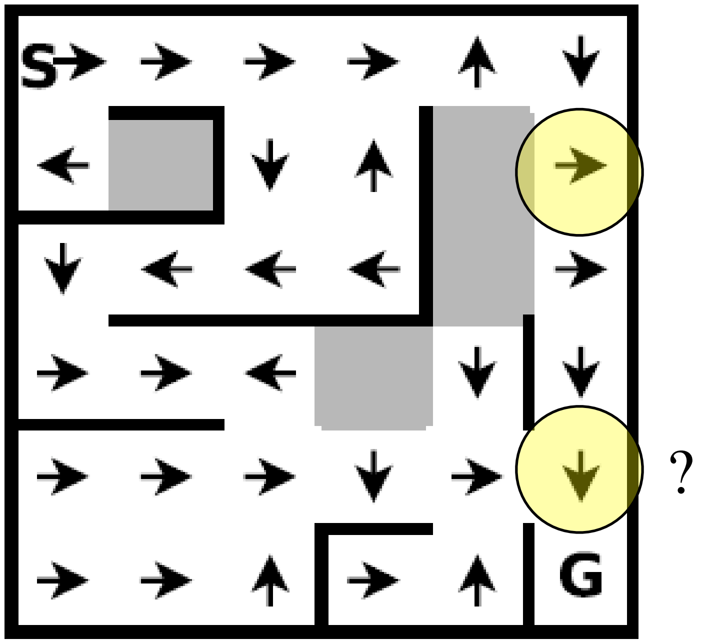
Inherent Conflict...


AH HA! HERE WE ARE, RIGHT AT 7:30!
YIKES! MY PAST AND FUTURE!
PUT DOWN THAT COMIC BOOK AND DO OUR HOMEWORK!
YEAH! GET TO WORK, YOU LOAFER!
HEY!
WHY SHOULD I DO ALL THE WORK? EITHER OF YOU COULD DO IT TOO!
BUT I DIDN'T AT 6:30 AND NOW IT'S 7:30.
AND AT 8:30 IT WILL BE TOO LATE. YOU'RE THE LAST CHANCE.
NOW ARE YOU GONNA START WRITING, OR DO WE HAVE TO POUND YOU?
HEY!
GO AHEAD AND HIT ME! MY FUTURE SELF WILL BE THE ONE WHO HURTS!

- PAC-MDP: Future self gets near-optimal reward.
- Near Bayesian: Current self gets near-optimal reward.
- Human behavior in between? (Hyperbolic discounting.)

PAC-MDP with Bayesian Priors

- With a prior that all similar colored squares are the same, we can bound the chance generalization will lead to sub-optimality.
- Idea: Don't worry about it if it's small!



X: {.7 .1 .8}, Y: {.8 .6 .7}

ϵ=0.0001, δ=0.05

<.99,.01>

R is near optimal whp

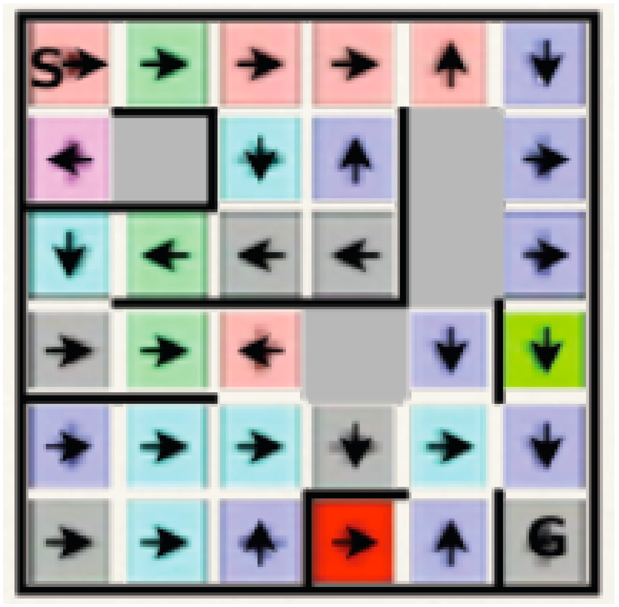
BOSS: Algorithmic Approach

- Optimism under uncertainty, not Bayes optimal
  - Sample models from the posterior.
  - Stitch together into a meta-MDP.
  - Solve to find optimal behavior: best of sampled set
  - Act accordingly until something new learned.
- If set big, near optimality whp (Asmuth et al. 09)
- Several ideas appear to be viable here

$$O\left(\frac{SAB}{\epsilon(1-\gamma)^2}\ln\frac{1}{\delta}\ln\frac{1}{\epsilon(1-\gamma)}\right)$$

BOSS in Structured Maze

- To learn in maze:
  - Chinese Restaurant Process prior
  - Finds (empirical) clusters
  - Outperforms Rmax, 1-cluster RAM-Rmax



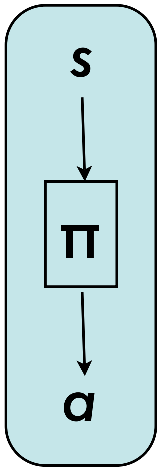
- Fewer than states
- Fewer than types
- Some types grouped
- Rare states nonsense

Computation Matters

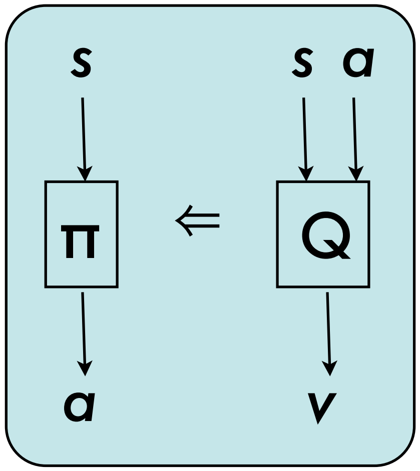
- Learning/exploration can be made efficient
  - model-based RL
  - PAC-MDP for studying efficient learning
  - KWIK for acquiring transition model
- Planning "just" a computational problem.
  - But, with powerful generalization, can quickly learn accurate yet intractable models!
  - Something needs to be done or the models are useless.

“Nesting” RL Approaches

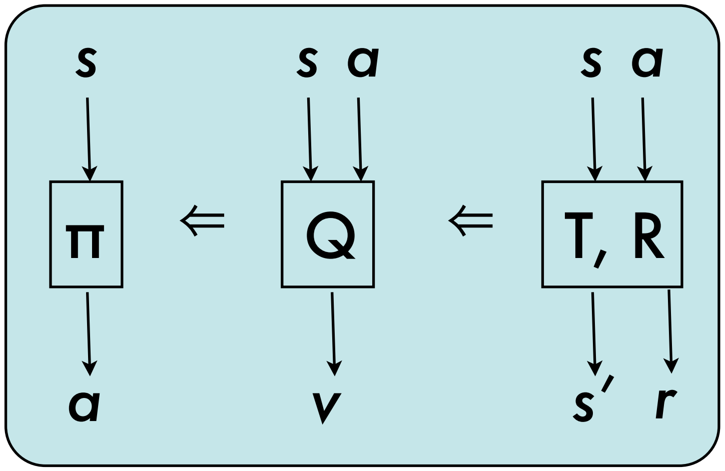
policy search



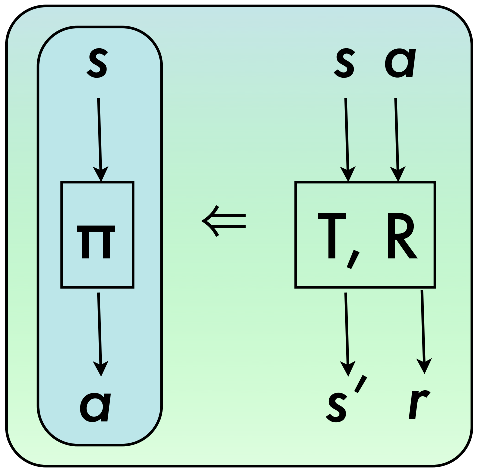
value function



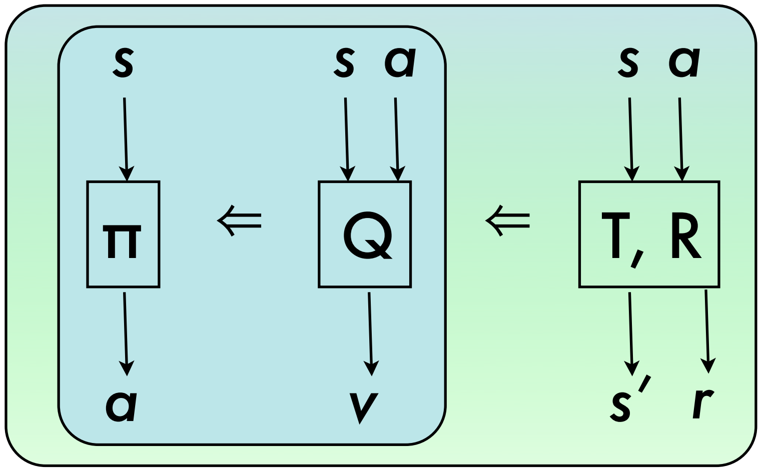
model-based



policy search inside model-based



value function inside model-based



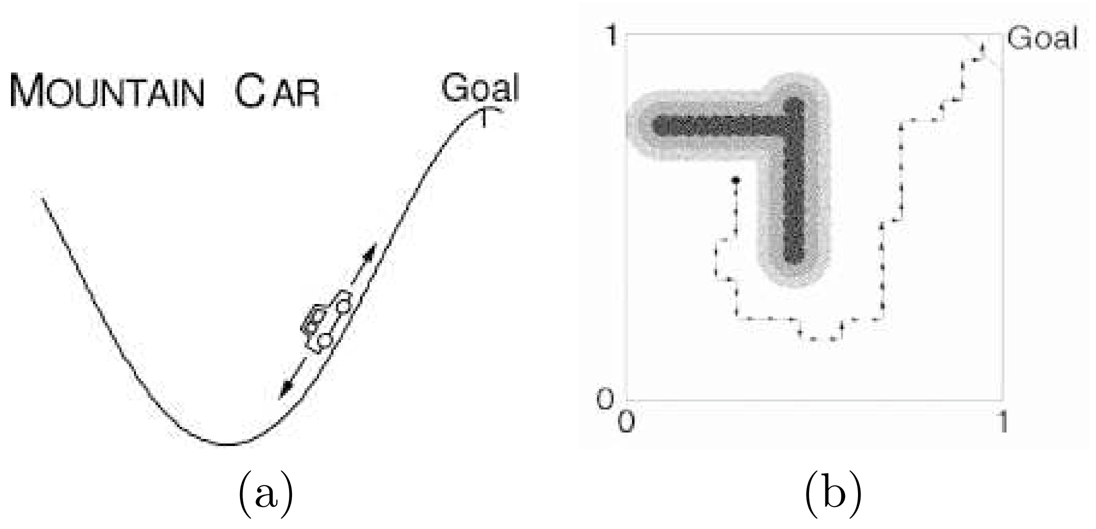
Function Approximation

A natural NIPS/model-based RL connection.

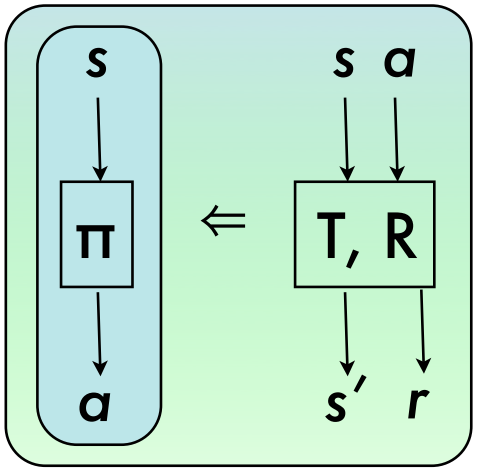
Use your favorite regression algorithm for T.

Use T as a simulator and run your favorite RL.

(Moore, Atkeson & Schaal 95; Jong & Stone 06)



Example: Autonomous Flight

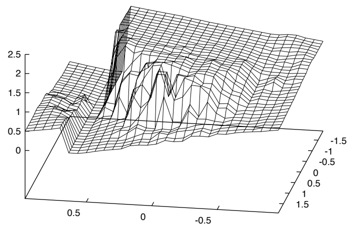


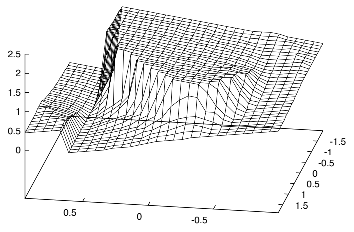
- Outer approach: Model-based RL.
  - Experts parameterize model space.
  - Parameters learned quickly from expert demonstration (no exploration needed).
- Resulting model very high dimensional (S,A)
- Inner approach: Policy-search RL.
  - Experts parameterize space of policies.
  - Offline search finds excellent policy on model.
  - Methodology robust to error in model.
- Learns amazing stunts (Ng et al. 03).

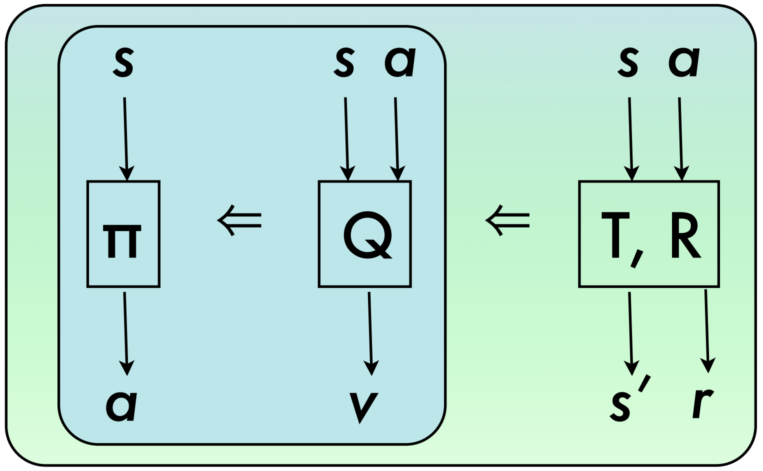
Tricks and Treats

Fitted Value Iteration

- Represent value function via anchor points and local smoothing (Gordon 95)
- Some guarantees if points densely sampled (Chow & Tsitsiklis 91)
- Combined with KWIK learning of model (Brunskill et al. 08)

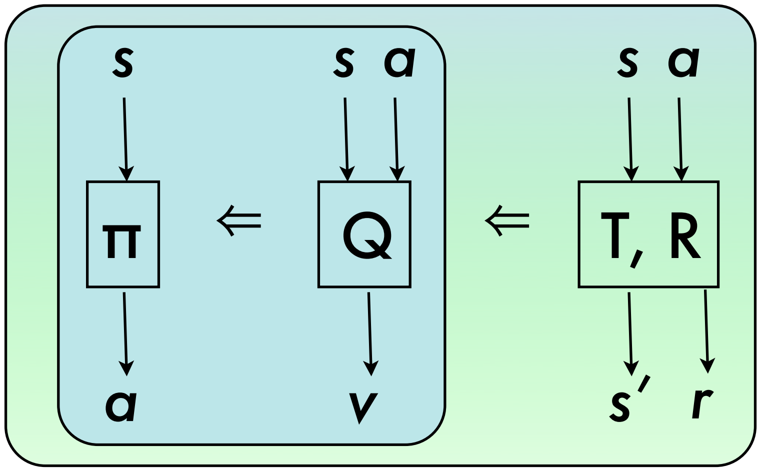






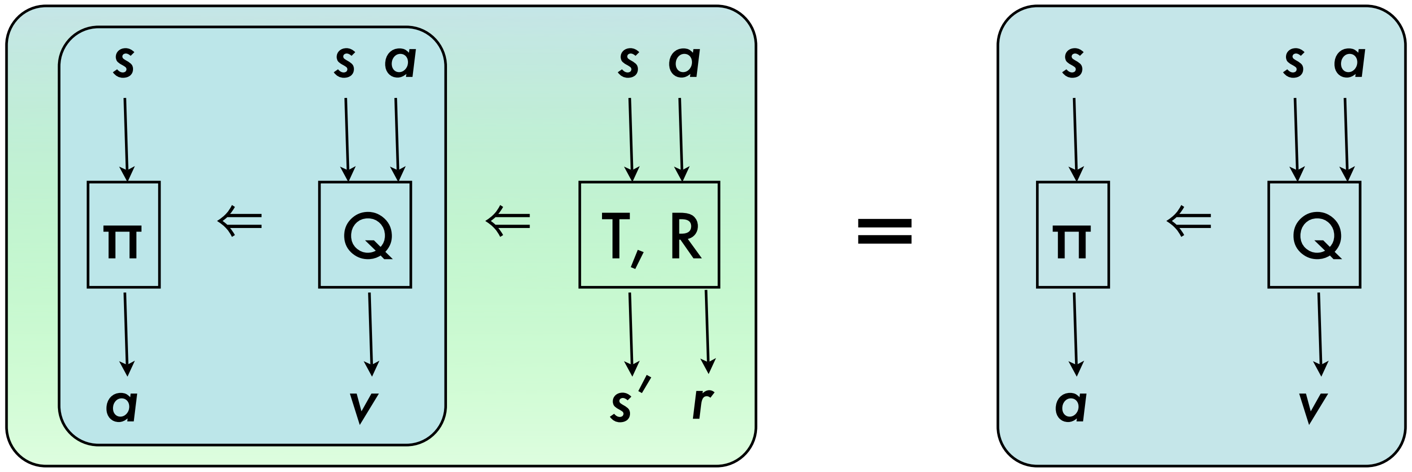
UCT: Upper Conf. in Trees

- Narrow, deep game-tree search via bandits (Kocsis & Szepsvári 06)
- Huge win in Go (Gelly & Wang 06; Gelly & Silver 07)
- Good fit w/ learned model.
  - Just needs to be able to simulate transitions.
  - KWIK-like methods are also "query" based.
- Not much work using it in RL setting.



Linear Models

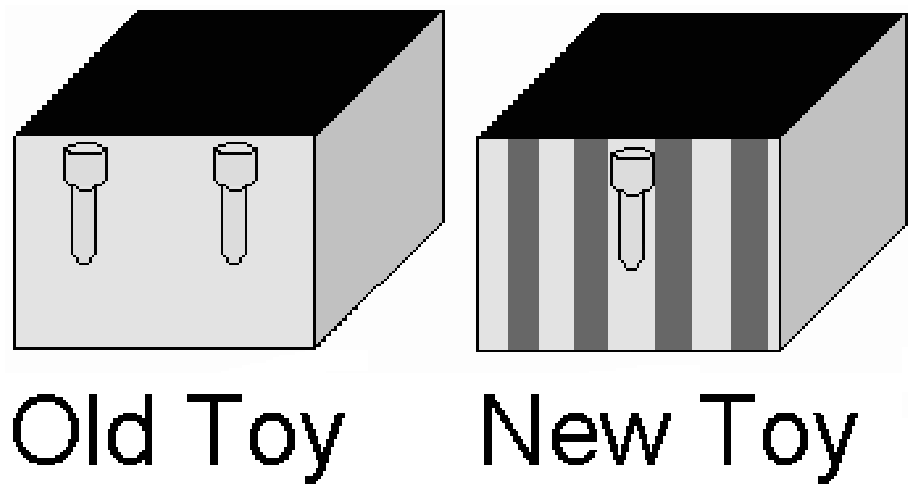
- Linear value function approaches: LSTD/LSPI (Boyan 99; Lagoudakis & Parr 03, Parr et al. 08).
- Method 1: Learn linear dynamics model from sample. Solve to get (linear) value function.
- Method 2: Learn value function directly.



Do Kids Explore Models?

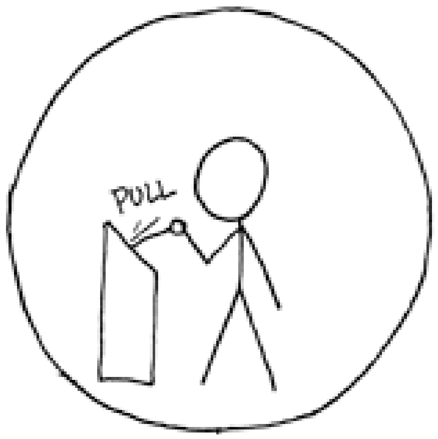
- Statistics of play sensitive to confounding
- Show kid 2-lever toy (Schulz/Bonawitz 07).
  - Demonstrate levers separately. Kid more interested in new toy.
  - Demonstrate them together. Kid stays interested in old toy.

- Experiment design intractable. KWIK-like heuristic?



Do People Explore Models?

xkcd



Wrap-Up

- Introduction
  - MDPs
  - Reinforcement Learning
- Model-based RL
- Efficient Exploration
  - PAC-MDP
  - KWIK
- Bayesian RL
  - Near-Bayesian
  - PAC-MDP
- Planning
  - Nesting approaches
  - UCT
- Model-based People?