Model-Based Reinforcement Learning (Day 1: Introduction)

Michael L. Littman

Rutgers University

Department of Computer Science

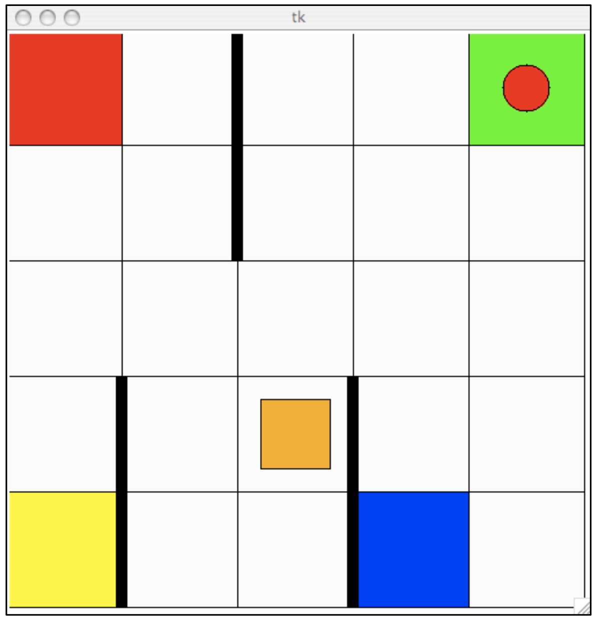
Rutgers Laboratory for Real-Life Reinforcement Learning

Plan

- Day 1: Introduction
  - RL
  - Q-learning
  - Convergence
  - Model-based RL
  - PAC-MDP
  - KWIK
- Day 2: Current Trends
  - Model-free RL & KWIK
  - Model/value approximation
  - Bayesian RL
  - UCT
  - Searchless planning

Start With Game...

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



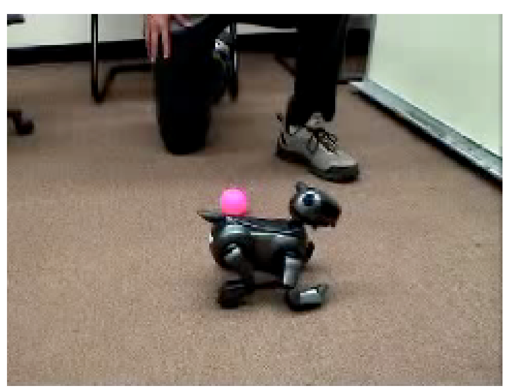
Find The Ball: Elements of RL

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

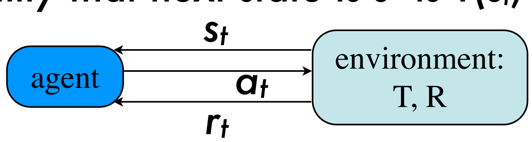
Three core issues in the dream RL system.

- <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 $0 \le \gamma \le 1$
- 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 $T(s_t, a_t, s')$

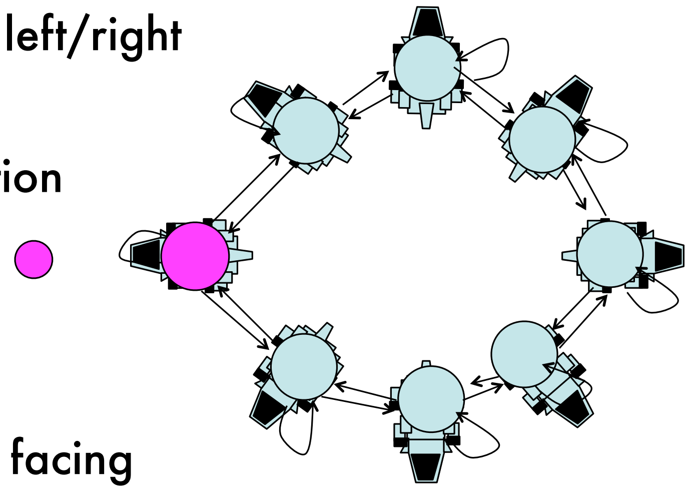


$$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q(s',a')$$

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

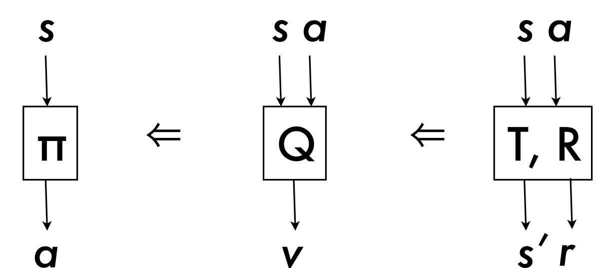
Find the Ball: MDP Version

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



Families of RL Approaches

policy search | value-function based | model based



More direct use, less direct learning

Search for action that maximizes value

Solve Bellman equations

More direct learning, less direct use

Q-learning

On experience $\langle s_t,a_t,r_t,s_{t+1}\rangle$:

$$\mathcal{Q}(s_t,a_t) \leftarrow \mathcal{Q}(s_t,a_t)$$
$$+ \alpha_t \left(r_t + \gamma \max_{a'} \mathcal{Q}(s_{t+1},a') - \mathcal{Q}(s_t,a_t)\right)$$

If:

- All $\langle s,a\rangle$ visited infinitely often.
- $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$

Then: $\mathcal{Q}(s,a) \rightarrow Q(s,a)$ (Watkins & Dayan 92).

Model-based MDP Learner

On experience $\langle s_t,a_t,r_t,s_{t+1}\rangle$:

- $\mathcal{R}(s_t,a_t) \leftarrow \mathcal{R}(s_t,a_t) + \alpha_t(r_t - \mathcal{R}(s_t,a_t))$
- $\mathcal{T}(s_t,a_t,s_{t+1}) \leftarrow \mathcal{T}(s_t,a_t,s_{t+1}) + \alpha_t(1 - \mathcal{T}(s_t,a_t,s_{t+1}))$
- $\mathcal{T}(s_t,a_t,s') \leftarrow \mathcal{T}(s_t,a_t,s') + \alpha_t(0 - \mathcal{T}(s_t,a_t,s'))$
- $\mathcal{Q}(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{T}(s,a,s') \max_{a'} \mathcal{Q}(s',a')$

If:

- All $\langle s,a\rangle$ visited infinitely often.
- $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$

Then: $\mathcal{Q}(s,a) \rightarrow Q(s,a)$ (Littman 96).

PAC-MDP Reinforcement Learning

PAC: Probably approximately correct (Valiant 84)

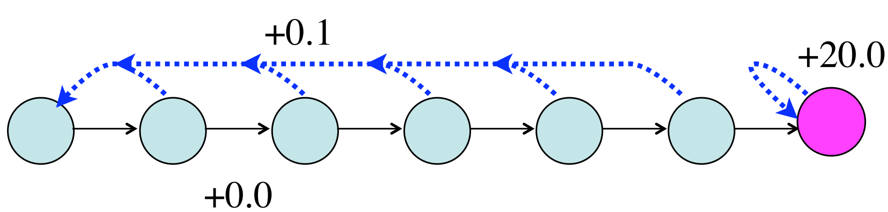
Extended to RL (Fiechter 95, Kakade 03, etc.).

- Given $\epsilon>0$, $\delta>0$, k actions, n states, γ.
- We say a strategy makes a <u>mistake</u> 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, $1/\epsilon$, $1/\delta$, $1/(1-\gamma)$.

Must balance <u>exploration</u> and <u>exploitation</u>!

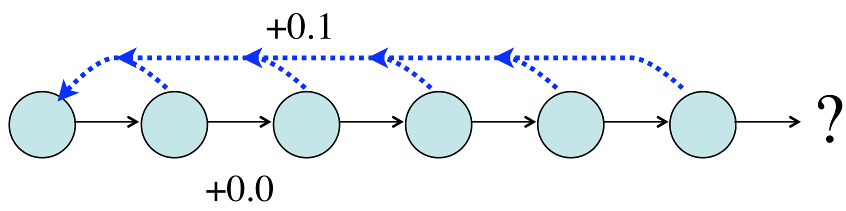
Q-learning Not PAC-MDP

- Family: initialization, exploration, α_t decay
- Combination lock



- Initialize low, random exploration (ϵ-greedy)
  - 2^n to find near-optimal reward. Keeps resetting.
  - Needs more external direction.

Model-based Can Be PAC-MDP



- Behavior differs depending on assumption

	truth: ? = low	truth: ? = high	
assume: ? = low	ignore ?, optimal	ignore ? suboptimal!	← No PAC-MDP guarantee
assume: ? = high	visit ?, explore	visit ?, optimal	← PAC-MDP if not too much exploration

Optimism Under Uncertainty

- Idea of exploration bonus well known.
- Shown to provide PAC-MDP guarantee (Kearns & Singh 02, Brafman & Tennenholtz 02).
- Key ideas:
  - Simulation lemma: Optimal for approximate model is near-optimal.
  - Explore or exploit lemma: If can't reach unknown states quickly, can achieve near-optimal reward.
- Extend to factored dynamics (Kearns & Koller 99) and metric spaces (Kakade et al. 03).

Model-free PAC-MDP

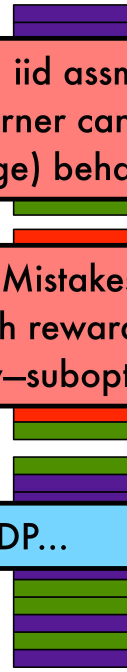
- Although not directly relevant, this problem was solved (Strehl, Li, Wiewiora, Langford, Littman 06).
- Modifies Q-learning to build rough model from recent experience.
- Total mistakes in learning ≈ $nk/((1-\gamma)^8\epsilon^4)$.
- Compare to model-based methods: mistakes in learning ≈ $n^2k/((1-\gamma)^6\epsilon^3)$. (Better in states, worse in horizon.)
- Lower bound, also (Li 09).

Generalization in PAC-MDP

- Can we draw on classical ML theory?
- Model learning is a supervised problem.
  - Given examples of *s*,*a* pairs, predict *s′*.
- Not just for table lookup anymore!
- Extend results to functions that generalize by defining the right learning problem...

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* mistak[...]
- 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 Learning

- "Knows What It Knows"
  - Like PAC, no mistakes.
  - Like mistake bound, no distribution assumption.
- Harder problem
  - PAC ≤ mistake bound ≤ KWIK
- Very well suited to model learning:
  - experience distribution changes during learning
    - distribution varies with behavior, which should change!
  - exploration driven by known/unknown distinction
    - don't want to be wrong and stop exploring too soon

KWIK Learn a Coin 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 $1-\delta$.

Some Things to KWIK Learn

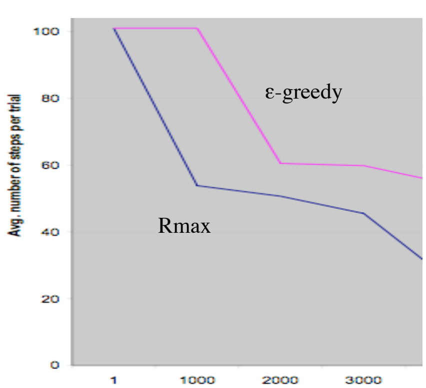
- coin probability
- an output vector, each component is KWIK learnable
  - multinomial probability (dice learning)
- a mapping from input partition to outputs where partition is known and mapping within each partition is KWIK learnable
  - That's a standard transition function (*s*,*a* to vector of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

RMAX and KWIK Learning

- RMAX (Brafman & Tennenholtz 02)
  - KWIK learn model ($T(s,a,\cdot)$ unknown m times).
  - For unknown parts, assume max possible reward ($\mathcal{Q}(s,a) = rmax/(1-\Upsilon)$).
  - Solve for $\mathcal{Q}$ and use resulting policy until something new becomes known.
- Total mistakes in learning ≈ $n^2k/((1-\Upsilon)^3\epsilon^3)$ (Strehl, Li, Wiewiora, Langford, Littman 06; Li 09).

RMAX Speeds Learning

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

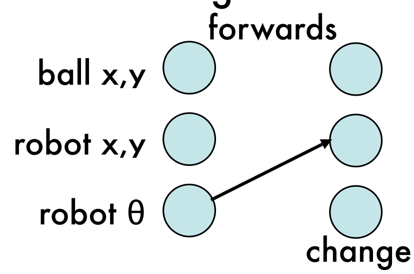


Details: Discretized 15x15 grid x 18 orientation (4050 states); 6 actions: forward, backward, turn L ,turn R, slide L, slide R.

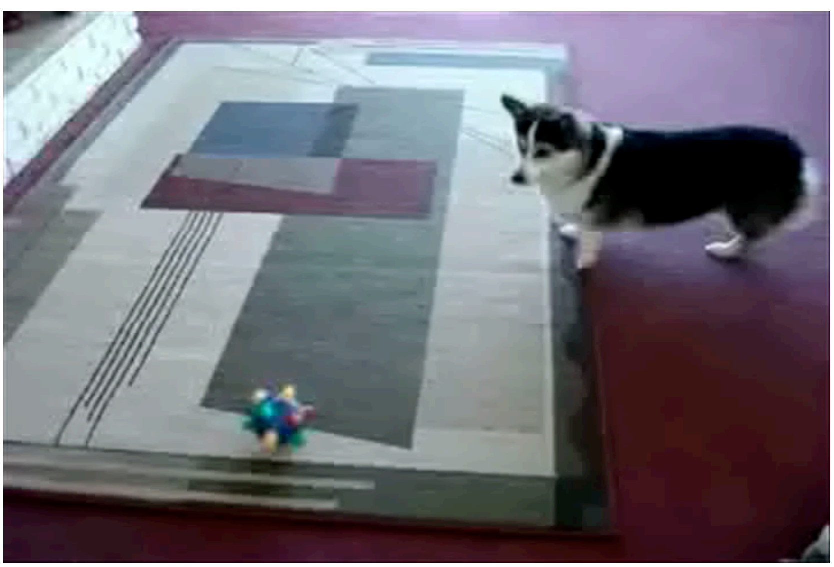
(Nouri)

Generalizing Transitions

- In MDPs, states are viewed as independent.
  - Transition knowledge doesn't transfer.
- Real-life action outcomes generalize.
  - Learn in one state, apply to others.
- Needed:
  - MDP variants that capture transition regularities.
    - Continuous MDPs
    - RAM-MDPs
    - Factored-state MDPs
    - Object oriented MDPs



Continuous Transition Model



(Nouri)

Relocatable Action Models

Decompose MDP transitions into state-independent outcomes (Sherstov, Stone 05).

$$T'(s,a,s') = \sum_{o \text{ s.t. } \eta(s,o)=s'} t(\kappa(s),a,o)$$

- $\kappa : S \rightarrow C$ is the *type function*. It maps each state to a type (or cluster or class) $c \in C$.
- $t : C \times A \rightarrow \Pr(O)$ is the *relocatable action model*. It captures the outcomes of different actions in a state-independent way by mapping a type and action to a probability distribution over possible outcomes.
- $\eta : S \times O \rightarrow S$ is the *next-state function*. It takes a state and an outcome and provides the next state that results.

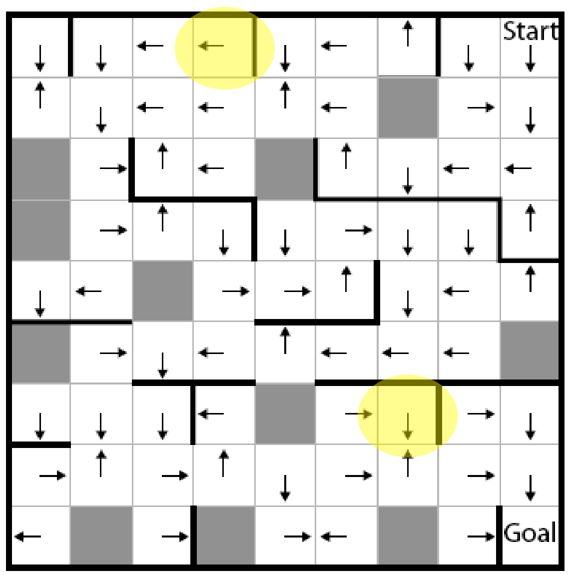
RAM Example

- η: the geometry
- *t*: actions effects
- κ: the local walls

Example: .8 in intended direction, .1 at right angles, unless wall blocks motion.

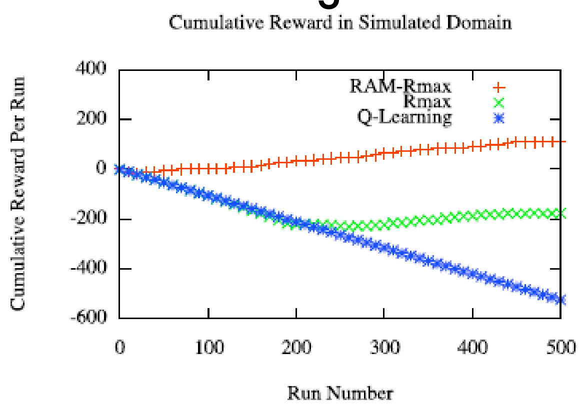
Action "go N" in a state with a walls to the N&E will go W wp .1, not move wp .9 (.8 N + .1 E).

(Leffler, Edmunds, Littman 07)



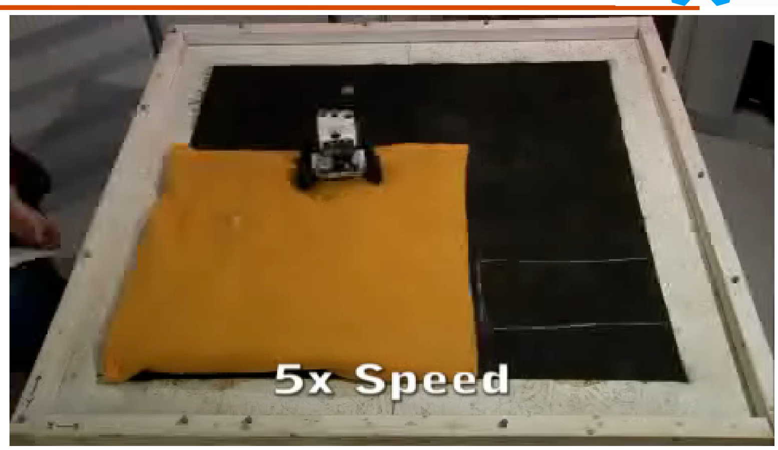
Speeds Up Learning

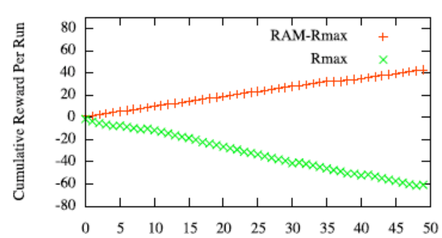
- Cumulative reward is larger for RAM-RMAX than RMAX or Q-learning.
- KWIK bound depends on classes, not states.
- (It also has more background knowledge.)



Robotic Example

- States: position and orientation
- Goal: Reach box as quickly as possible
- Types: sand, wood
- Actions: L, R, F





RAM Learning #2

QuickTime™ and a
decompressor
are needed to see this picture.

(Leffler, Mansley, Edmunds)

Factored-state MDPs

- Generalizing MDP states via DBN factoring of transition function (Boutilier et al. 99).
- 2^n states, k actions
- Blends planning-type state representations with Bayes net probability distributions
- R, T unknown; some experimentation needed
- KWIK learnable: Just a different partition of the input space.

Factored-state MDP

State is a cross product.

- Example: Taxi (Dietterich 98).

Primitive actions:

- N,S,E,W, pickup, dropoff.

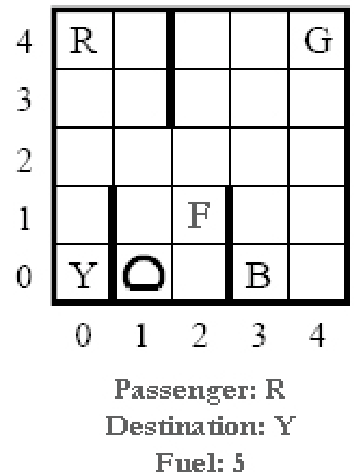
Passenger is at R, Y, G, B.

Destination is R, Y, G, B.

Reward for successful delivery.

Approx. 500 states, but related.

- state = (taxi loc, pass. loc, dest)

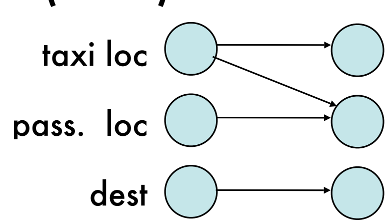


Compact Model

- Abstraction: Use a factored (DBN) model.

independence relations for "pickup"

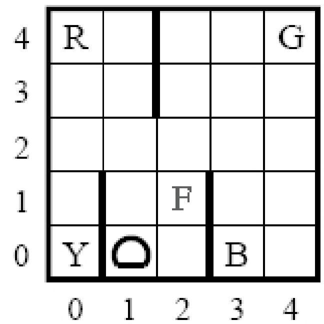
(further, can use context-specific independence, Boutilier et al. 95)



- Model generalizes because transitions for multiple states share structure/parameters.
- If graph known, KWIK learnable: composition of output vector and input partition.

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)

Model-Based Reinforcement Learning (Day 2: Other Stuff)

Michael L. Littman

Rutgers University

Department of Computer Science

Rutgers Laboratory for Real-Life Reinforcement Learning

Plan

- Day 1: Introduction
  - RL
  - Q-learning
  - Convergence
  - Model-based RL
  - PAC-MDP
  - KWIK
- Day 2: Current Trends
  - Bayesian RL
  - Model/value approximation
  - UCT
  - Searchless planning

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

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

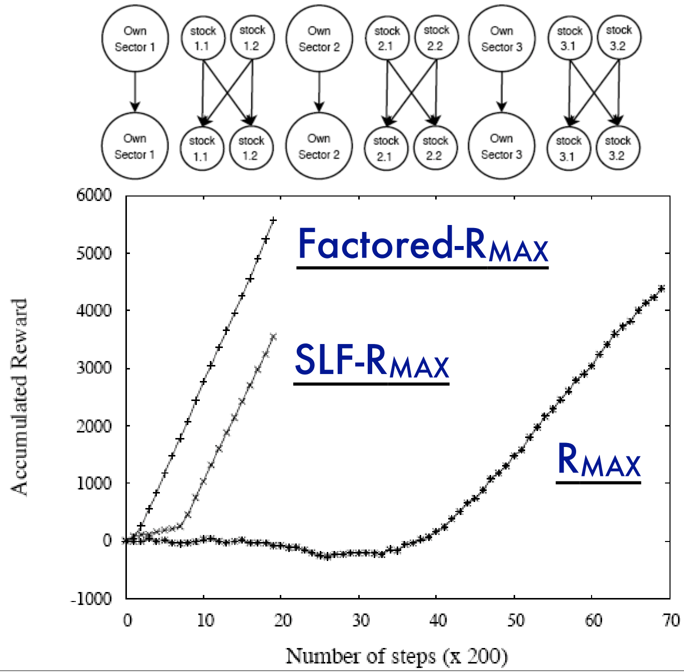
Artificial Stock Example

Discovers the structure and exploits it much faster than R_{MAX} can learn the MDP.

Factored-R_{MAX}: Knows DBNs

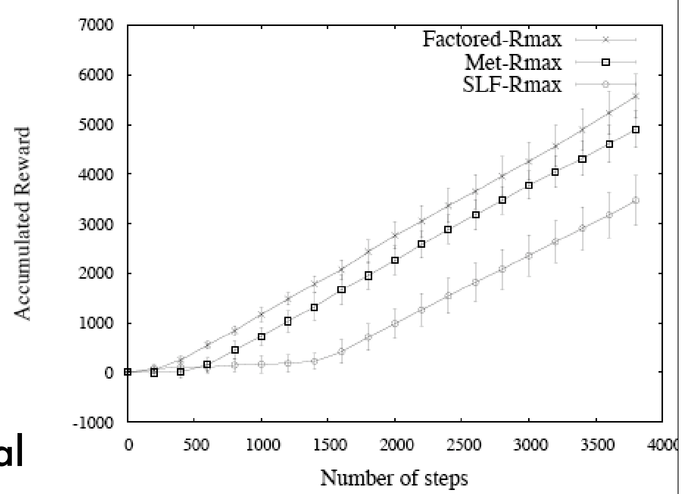
SLF-R_{MAX}: Knows size of parent sets

R_{MAX}: It's an MDP



Improved Bounds

- SLF-R_{MAX} runs in roughly $k^2 \log k$
- MET-R_{MAX} faster, like $k \log k$
- Bounds weak, but suggest a better algorithm!
- Also selected visual pattern for terrain learning.



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

Grid World Demo (expt2)

- Unknown: What's a wall?
- (Purpose: What doesn't KWIK do?)

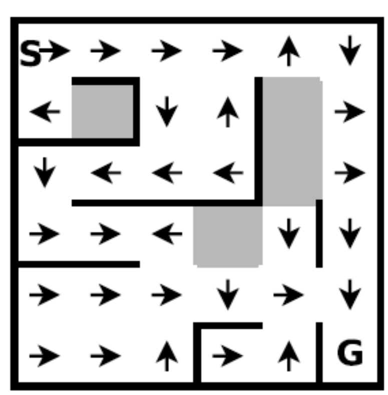
People Learn to Learn

- expt1: Always + (no mistakes)
- expt2: Always one shape (one mistake).
- expt3: Always some feature (two mistakes).

- Last maze is always +, but people perform differently depending on their experience.
- Transfer learning in RL (Taylor & Stone 07, e.g.).
- KWIK can learn any of these classes, but if all are possible, devolves to worst case.

Playing the Odds

- Standard PAC-MDP algorithms can't say:
  - I know you told me all states independent,
  - but every wall I've seen has been painful.
  - Can I just walk around now, please?
- Rmax vs. RAM-Rmax
  - Rmax: states independent
  - RAM-Rmax: Types known
- What if states "cluster"?
  - new state likely to be familiar



Bayesian Perspective

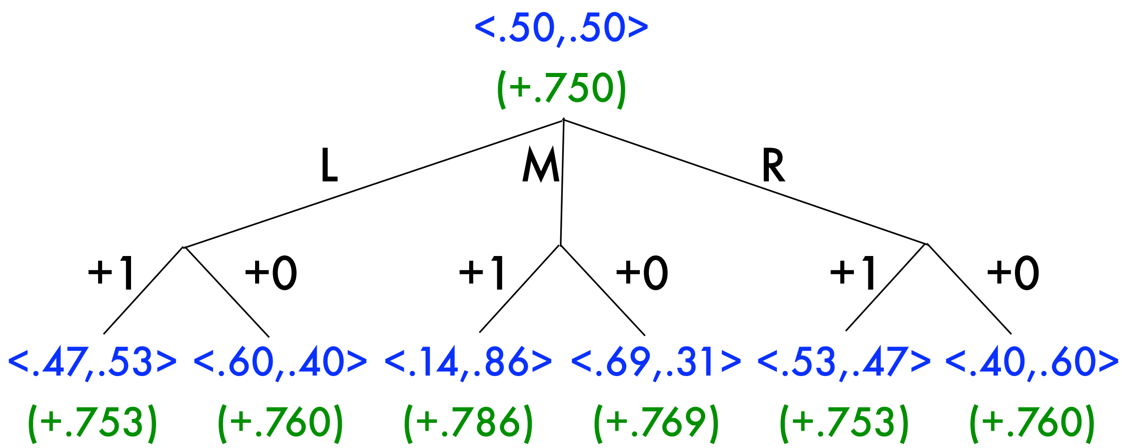
- Start with a prior over models.
- Maintain a posterior across tasks.
- Now, we can talk about more/less likely models instead of just *possible* models.
- How can we use the Bayesian view in exploration?

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 will change the posterior.
  - Can choose actions that truly maximize expected reward: No artificial distinction between exploring and exploiting or learning and acting.
- Hideously intractable except in some special cases (bandits, short horizons).

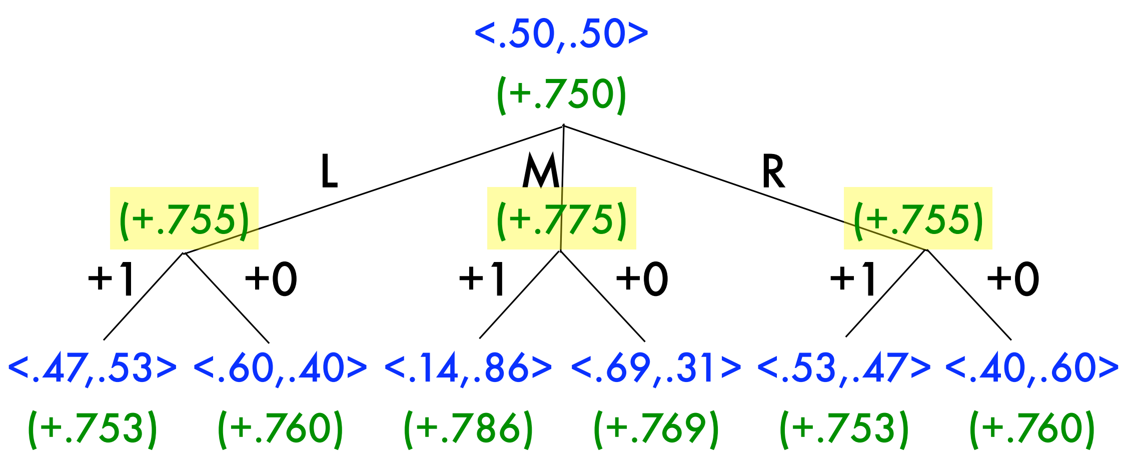
Concrete Example

- 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)



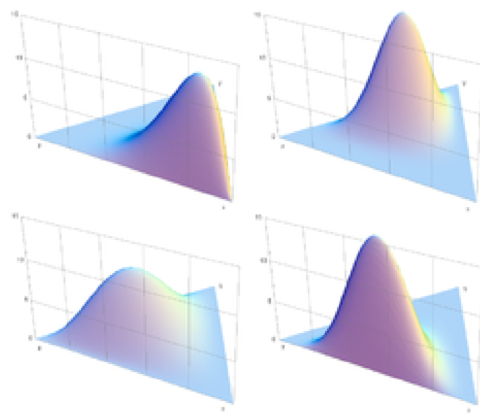
Concrete Example

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



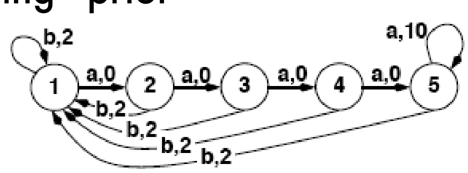
Representing Posteriors

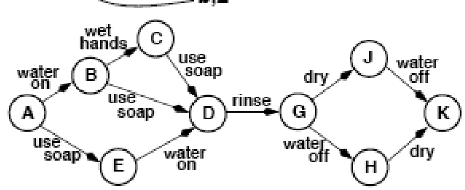
- T: $s,a \rightarrow$ multinomial over states
- If independent for each s,a: Dirichlet!
- Keep counts for each observed outcome.
- Can recover uncertainty in overall estimate.
- Unlike example, distribution over an infinite set.



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.





Near Bayes Optimal Behavior

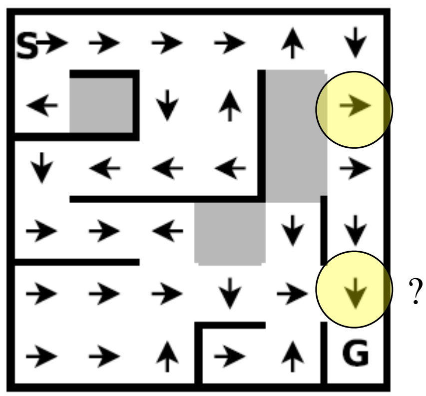
- Recall PAC-MDP, whp makes few mistakes.
- Near Bayesian: mistakes are actions taken with values far from Bayes optimal.
- Bayesian Exploration Bonus (Kolter & Ng 09) keeps mean of posterior and adds $1/n$ bonus to actions taken n times.
  - BEB is computationally simple.
  - BEB is Near Bayesian.
  - 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)
- Not clear which is "right".
- Who gets near optimal reward?
  - PAC-MDP: Future self
  - Near Bayesian: Current self
- Human behavior somewhere 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.
- <u>Idea</u>: 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

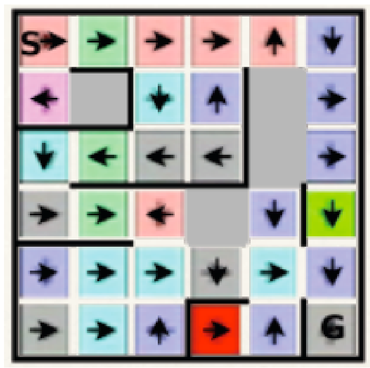
BOSS: Algorithmic Approach

- Optimism under uncertainty, not Bayes optimal
  - Sample models from the posterior.
  - Stitch together into a meta-MDP.
  - Solve to find optimal behavior: <u>b</u>est <u>o</u>f <u>s</u>ampled <u>s</u>et
  - 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 Maze

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

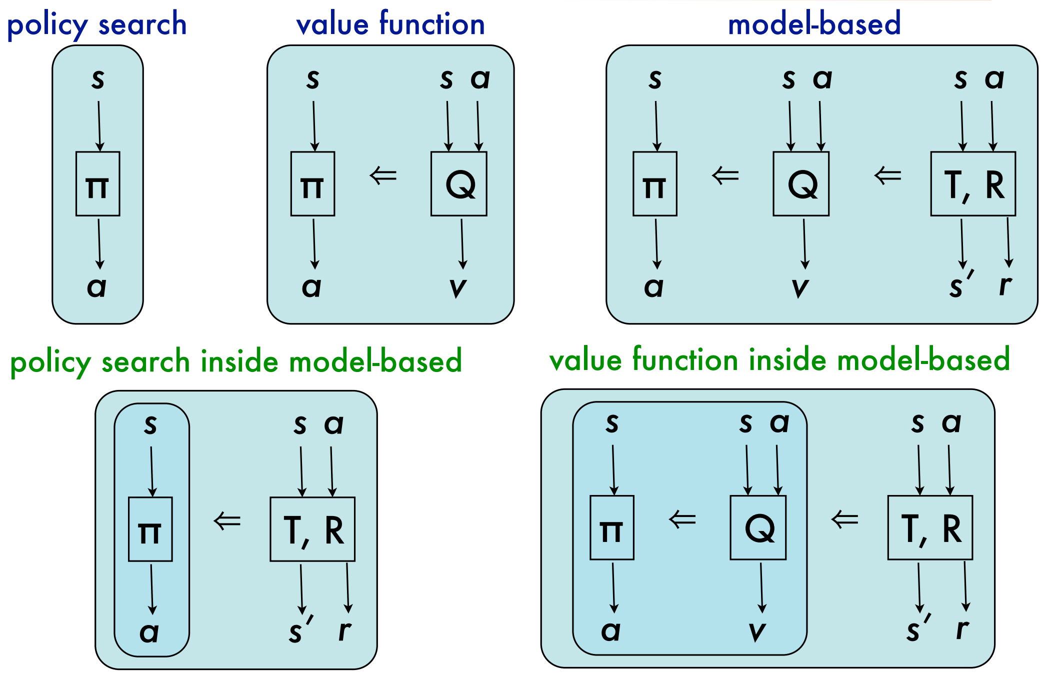


- Fewer than states
- Fewer than types
- Some types grouped
- Rare states nonsense

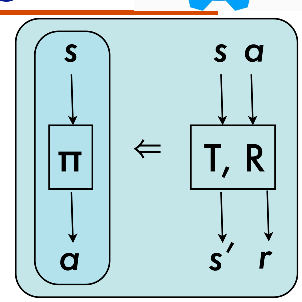
Computation Matters

- 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 intractible models!
  - Something needs to be done or the models are useless. (Not as focused on guarantees today.)

“Nesting” RL Approaches



Example: Autonomous Flight

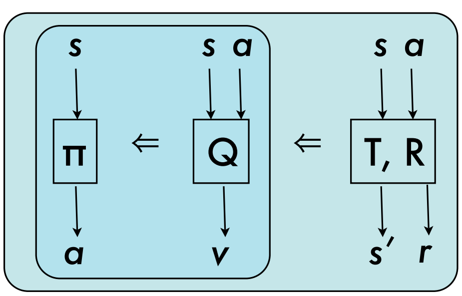


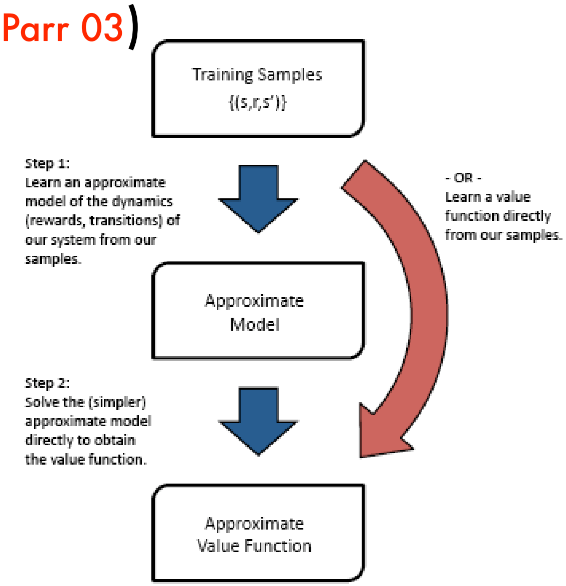
- Outer approach: Model-based RL.
  - Experts parameterize model space
  - Parameters learned quickly from expert demonstration (no exploration needed)
- Resulting model very high dimensional (S,A)
- Inner approach: Policy-search RL.
  - Experts parameterize space of policies
  - Offline search finds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

Tricks and Treats

Linear Models

- Linear value function approaches: LSTD/LSPI (Boyan 99; Lagoudakis & Parr 03)

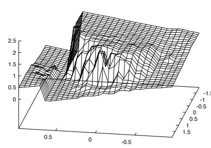


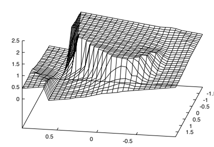


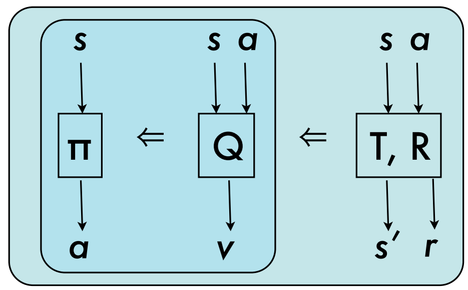
- Give the same result! (Parr et al. 08)

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)

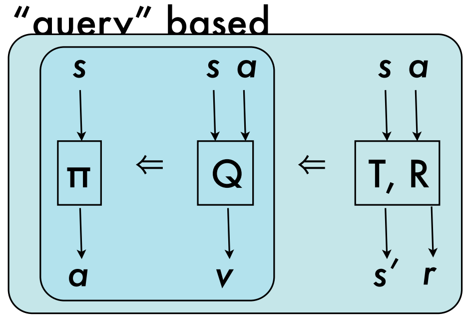






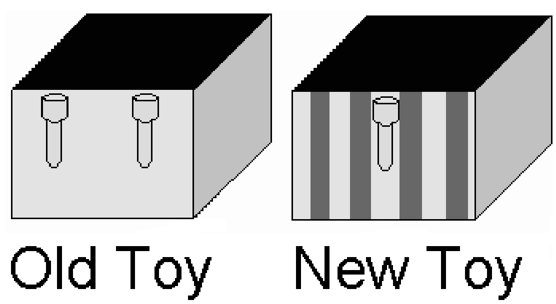
UCT: Upper Conf. in Trees

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



Do Kids Explore?

- Statistics of play sensitive to confounding
- Show kid 2-lever toy (Schulz/Bonawitz 07).
  - Demonstrate both. Kid becomes interested in new toy.
  - Demonstrate them together. Kids stays interested in old toy.
- Experiment design intractable. KWIK-like heuristic?



Do People Explore? (xkcd)

Wrap Up

- Introduction
  - Q-learning
  - MDPs
  - Model-based RL
  - PAC-MDP
  - KWIK
- Current topics
  - Bayesian RL
  - “Recursive” RL
  - Function approximation
  - Human exploration