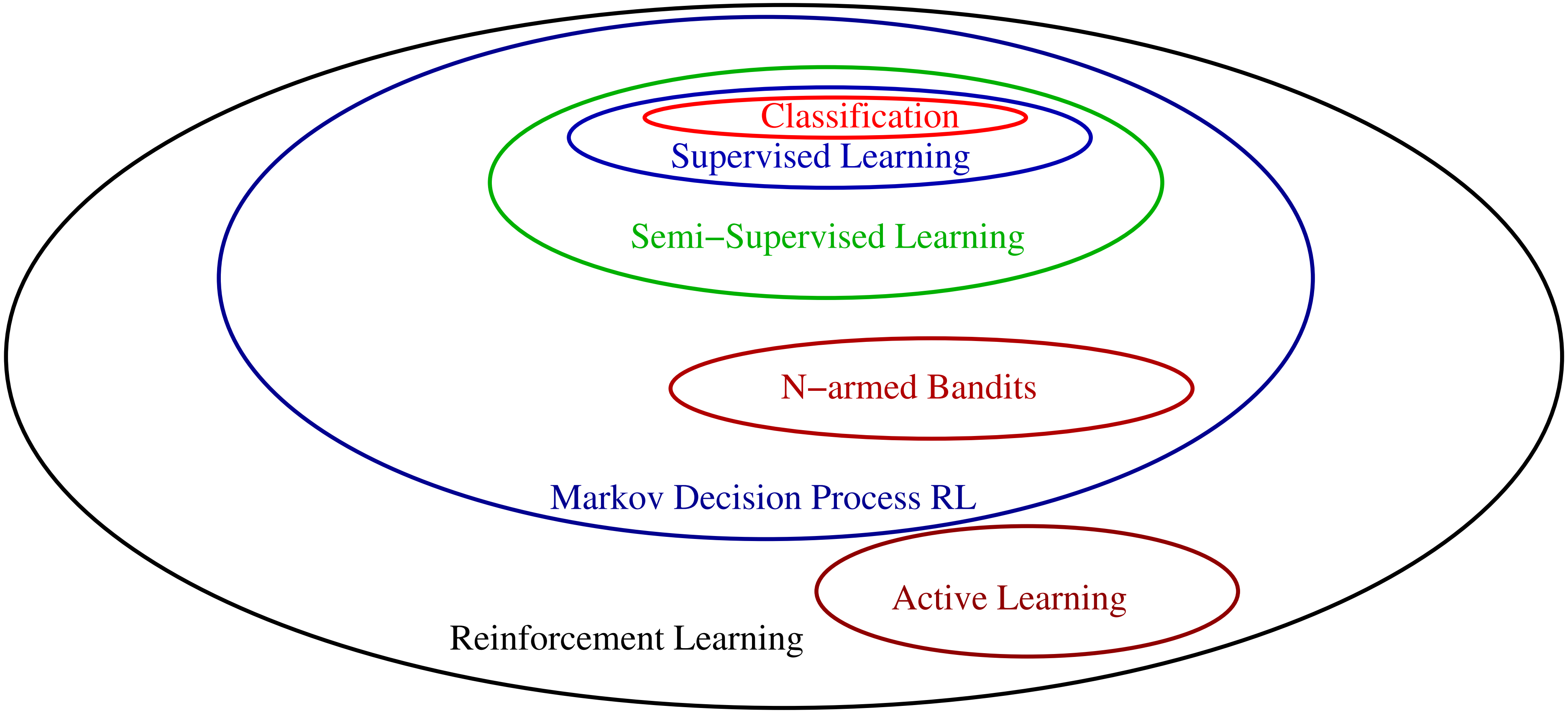
Modern Reinforcement Learning Theory

John Langford

Yahoo Research

Backing Material at: http://hunch.net/~jl/tutorial/RL.html

Reinforcement Learning is Always Relevant



The answer to: "Is this an RL problem?" is always "yes".

The implication: RL theory is broadly applicable.

The other implication: RL theory is often only weakly relevant. (breadth+relevance=hard.)

Understanding a problem as an RL problem is the *beginning* to solving it. Whenever possible, you want to understand how the problem is special.

Outline

1. Sample Complexity Results

2. Limitations of Sample Complexity

3. Reductions Results

What is a sample complexity guarantee?

"With high probability, given $m(\cdot)$ samples, guarantee ϕ holds."

1. S = the number of states in an MDP
2. A = the number of actions/state in an MDP
3. T = the horizon time you care about (or γ = discount factor)
4. O = number of observations
5. ϵ = precision parameter

The E^3 Guarantee (Kearns & Singh 1998)

Trace Model = ability to read current state s, take action a, observe next state s' and reward r. Notation: $TM : A \rightarrow S \times [0, 1]$.

Assume MDP$(S, A, p(s'|s, a))$ with horizon T

1. Original: +assume mixing time $\tau \Rightarrow$ Poly$(S, A, \tau, \frac{1}{\epsilon})$ samples implies ability to act ϵ optimal for $T > \tau$.

2. Modified: Poly$(S, A, T, \frac{1}{\epsilon})$ samples implies ability to act ϵ optimal for T timesteps.

(2) + mixing assumption implies (1). (2) holds even for deterministic worlds. We'll go through (2).

E^3 Theorem Statement

Theorem: There exist an algorithm E^3 such that for all MDP $(S, A, T, p(s'|a, s))$ with rewards $r \in [0, 1]$, with probability $1-\delta$, for all except $\mathrm{Poly}\left(S, A, T, \frac{1}{\epsilon}, \ln\frac{1}{\delta}\right)$ steps $Q_{E^3}(s, E^3(h)) \geq V^*_{T-t \bmod T}(s) - \epsilon$ where h is the history of observations.

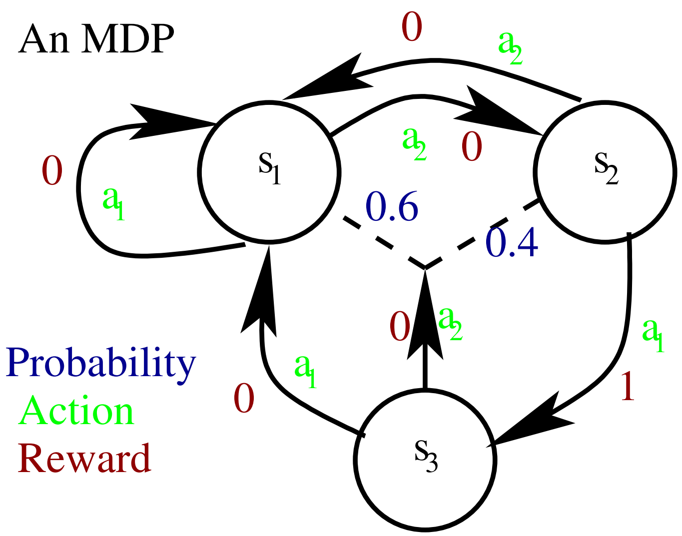
Suboutline:

1. The Algorithm

2. The Proof

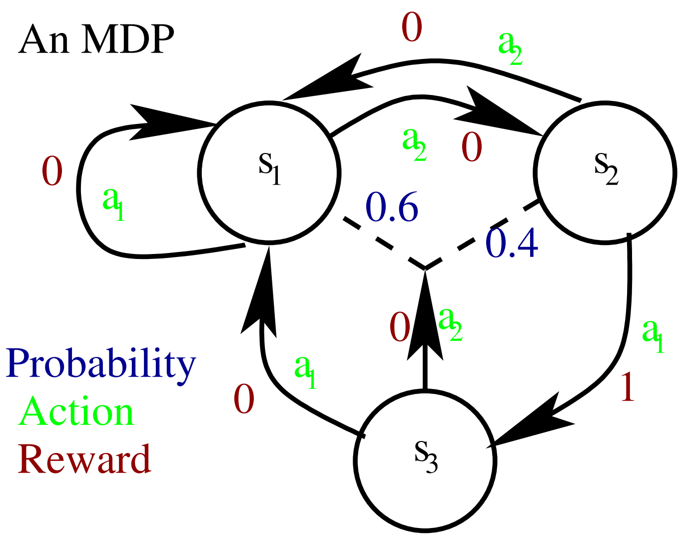
The Known(h) MDP

A state s, all actions a leaving s and the probability of their outcomes is known if all actions a leaving s have been executed at least n times.

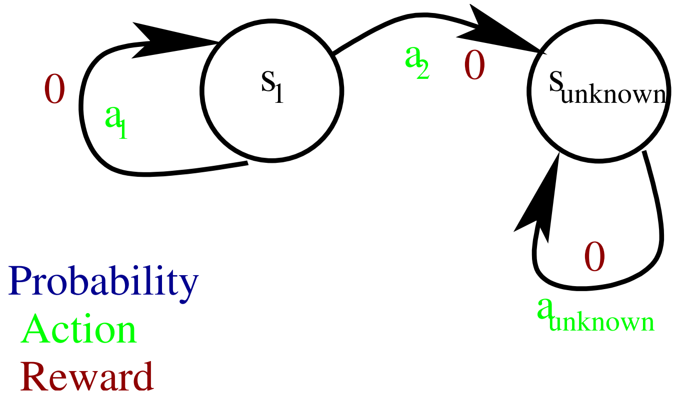


Initially: known MDP = nothing

The Known(h) MDP

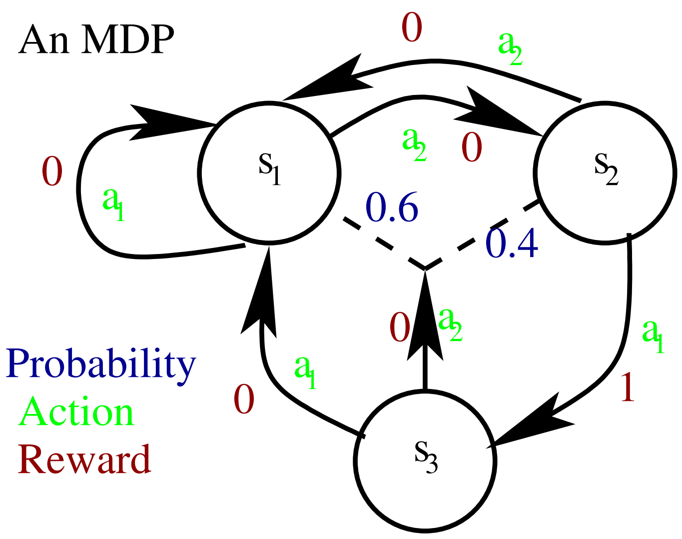


Then:

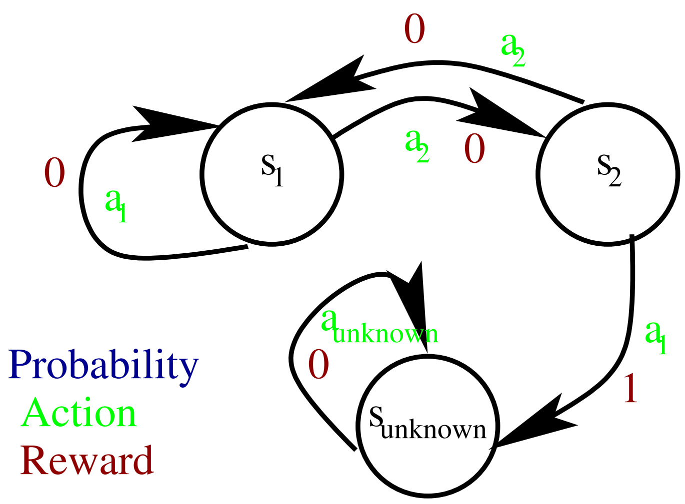


Complete dangling action(s) with one state that always has reward 0.

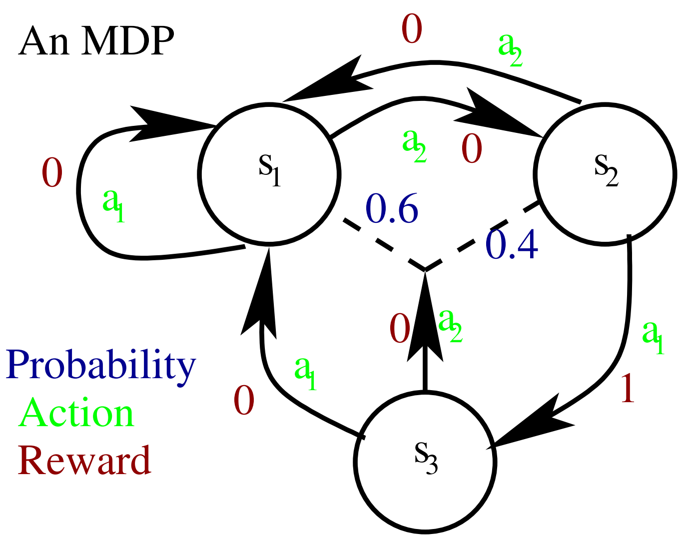
The $\text{Known}(h)$ MDP



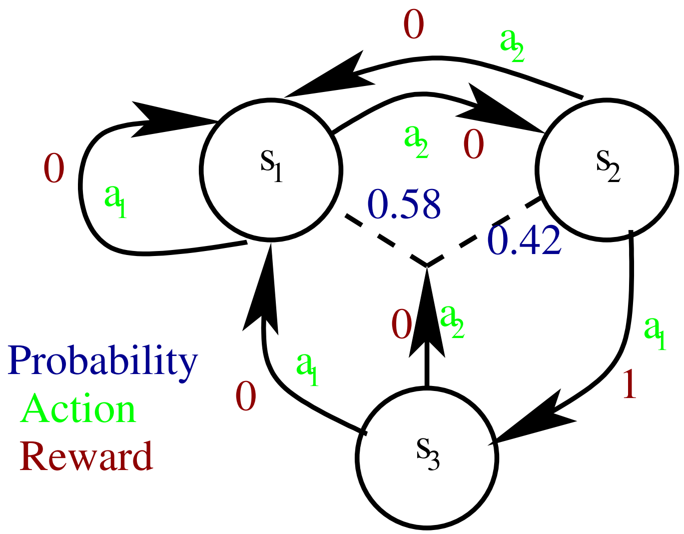
Then:



The Known(h) MDP



Finally:



(note: the probabilities are empirical counts)

The Unknown(h) MDP

Unknown(h) = Known(h) except the reward is 1 for actions which leave the known states and 0 otherwise.

Dynamic Program

Fundamental operation: Given MDP M and state s,

$$\mathrm{DP}(M, s, t) = a, v$$

where v = the maximum expected $T - (t \bmod T)$ reward sum and a= action achieving it.

Computation:

$$\mathrm{DP}(M, s, t) = \max_a E_{s', r \sim M(s,a)} r + \mathrm{DP}(M, s', t+1)$$

$$\mathrm{DP}(M, s, nT) = 0$$

$E^3(h)$ Explicit Explore or Exploit Algorithm

1. If last s not in Known(h): choose the least previously used action

2. Else:

   (a) If DP(Unknown(h)) $> \epsilon'$ then act according DP(Unknown(h)) until state is unknown or $t \bmod T = 0$ then go to (1).

   (b) else act according to DP(Known(h)).

The proof uses 5 MDPs

1. MDP — the true MDP
2. Known(h) = known MDP
3. Unknown(h) = unknown MDP
4. $\mathsf{MDP}_{\mathsf{K}(h)}$ = MDP restricted to the known states
5. $\mathsf{MDP}_{\mathsf{U}(h)}$ = MDP restricted to the known states with rewards set to 0 except for escaping rewards.

Proof:

Simulation Lemma: $|\mathsf{DP}(\mathsf{MDP}_{\mathsf{U/K}(h)}) - \mathsf{DP}((\mathsf{Un})\mathsf{Known}(h))| \leq \frac{1}{\mathsf{Poly}(S,A,T,\ln\frac{1}{\delta})}$

Explore/Exploit Lemma: $\forall\alpha$ Either $\mathsf{DP}(\mathsf{MDP}_{\mathsf{K}(h)}) \geq \mathsf{DP}(\mathsf{MDP}) - \alpha$ or executing $\mathsf{DP}(\mathsf{MDP}_{\mathsf{U}(h)})$ has probability at least α/T of reaching an unknown state in T steps.

So $n = \mathsf{Poly}(S, A, T, \ln\frac{1}{\delta})$ implies ability to simulate on known states to precision $\frac{1}{\mathsf{Poly}(S,A,T,\ln\frac{1}{\delta})} << \epsilon$. $\Rightarrow$ Explore/Exploit Lemma $\alpha = \frac{3}{4}\epsilon$ implies probability at least $\epsilon/2$ of encountering unknown state. This can happen only $O(\frac{nSA}{\epsilon})$ times (Using the Chernoff bound). Each exploration uses at most T steps $\Rightarrow$ proof.

Simulation Lemma Proof sketch

n IID coin flips converge to mean at rate = Chernoff bound.

$$\Rightarrow \Pr\left(|\hat{p}(s'|a,s) - p(s'|a,s)| \leq \sqrt{\frac{\ln\frac{2}{\delta}}{2n}}\right) \geq 1-\delta$$

Union $\Rightarrow \Pr\left(\forall s', s, a: \; |\hat{p}(s'|a,s) - p(s'|a,s)| \leq \sqrt{\frac{\ln\frac{2S^2A}{\delta}}{2n}}\right) \geq 1-\delta$

Only S states s' so $1-\gamma$ probability on $\{s' : p(s'|a,s) > \frac{\gamma}{S}\}$ so $n = \tilde{O}\left(\frac{S^2}{\gamma^2}\right)$ implies a $1-\gamma$ fraction of $\hat{p}(s'|a,s)$ indistinguishable from $p(s'|a,s)$.

Trace is T steps long $\Rightarrow$ expectation over states is correct a $1 - T\gamma$ fraction of the time. $\Rightarrow$DP(Known(h)) produces correct value up to $T^2\gamma$ error $\Rightarrow$lemma.

Explore or Exploit Lemma Proof Sketch

$$\mathsf{DP}(\mathsf{MDP}) = E_{(s,a,r)^T \sim \pi^*, \mathsf{MDP}}\left[\sum_t r_t\right]$$

Let $p = \mathsf{Pr}_{\pi^*, \mathsf{MDP}}((s,a,r)^T \in \mathsf{Known}(h))$

$$= pE_{(s,a,r)^T \sim \pi^*, \mathsf{MDP}|p}\left[\sum_t r_t\right] + (1-p)E_{(s,a,r)^T \sim \pi^*, \mathsf{MDP}|\text{not } p}\left[\sum_t r_t\right]$$

$$\leq pE_{(s,a,r)^T \sim \pi^*_{\mathsf{K}(h)}, \mathsf{MDP}_{\mathsf{K}(h)}}\left[\sum_t r_t\right] + \mathsf{DP}(\mathsf{MDP}_{\mathsf{U}(h)})E_{(s,a,r)^T \sim \pi^*, \mathsf{MDP}|\text{not } p}\left[\sum_t\right.$$

$$\leq \mathsf{DP}(\mathsf{MDP}_{\mathsf{K}(h)}) + \mathsf{DP}(\mathsf{MDP}_{\mathsf{U}(h)})T$$

$\Rightarrow \mathsf{DP}(\mathsf{MDP}) - \mathsf{DP}(\mathsf{MDP}_{\mathsf{K}(h)}) \leq \mathsf{DP}(\mathsf{MDP}_{\mathsf{U}(h)})T$ $\Rightarrow$lemma

R-Max(h) Modification (Brafman & Tenenholtz, 2001)

(Roughly) The same result holds for the following algorithm:

1. If last s not in Known(h): choose the least previously used action.

2. Else act according to DP(Known(h)+Unknown(h)).

(Where "+" adds the rewards of each MDP on each edge.)

Delayed Q-learning (Li, Strehl, Wiewiora, Langford & Littman, 2006)

The theorem can be tightened from Poly(S, A) to $\tilde{O}(SA)$ using the Delayed Q-learning algorithm. [see presentation Tuesday@10:30]

Outline

1. Sample Complexity Results

2. Limitations of Sample Complexity

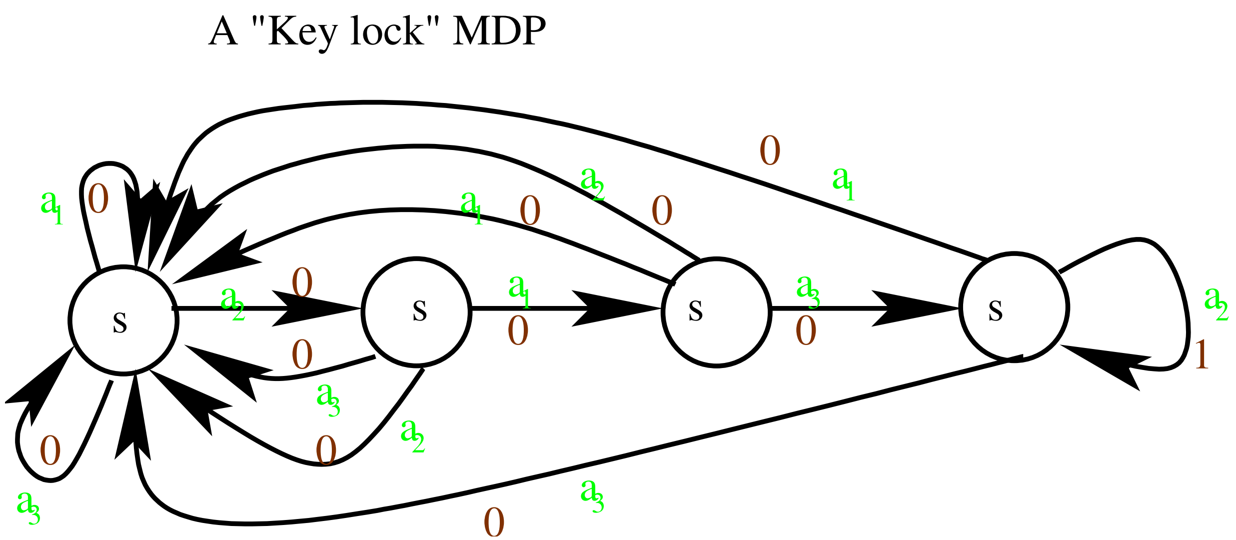
3. Reductions Results

The Limits of Sample Complexity: A lower bound

Theorem: Any algorithm A satisfying the E^3 statement must use at least $\Omega(TSA)$ actions to explore.

(There are stronger lower bounds, but this is sufficient. See Kakade, 2003 thesis.)

Proof



States in a chain. One action leads to next state, all the rest lead to the beginning. The final state has an action with reward 1.

Implications

Lower bound $\Rightarrow$ the really big problems can't be solved.

But the problems *are* solvable: we solve them every day.

$\Rightarrow$More or different assumptions are required.

Attempt 1: Factored-E^3 (Kearns & Koller, 1999)

Assume state is represented by a set of bits $s = (b_1, b_2, ..., b_n)$.

Assume $P(s'|s,a) = \Pi_i P(b_i|B_i, a)$ where B_i = subset of $\{b_1, b_2, ..., b_n\}$.

Theorem: There exist an algorithm Factored-E^3 such that for all Factored MDP, with probability $1 - \delta$, for all except

$$\text{Poly}\left(|MDP|, T, \frac{1}{\epsilon}, \ln \frac{1}{\delta}\right)$$

steps $Q_{\text{Factored}-E^3}(s, \text{Factored} - E^3(h)) \geq V^*_{T-t \bmod T}(s) - \epsilon$ where h is the history of observations and $|MDP|$ = description length.

Problem 1: Factoring isn't enough. Consider this room...

Problem 2: The *computational* complexity of planning is high.

Attempt 2: Metric-E^3 (Kearns, Kakade, and Langford 2003)

Assume there exists Model$(\{(s,a,r,s')^*\}, s'')$ such that with m experiences satisfying $d((s,a),(s'',a'')) \leq \alpha$, Model outputs s', r from almost the same distribution as $P(s', r | s'', a'')$.

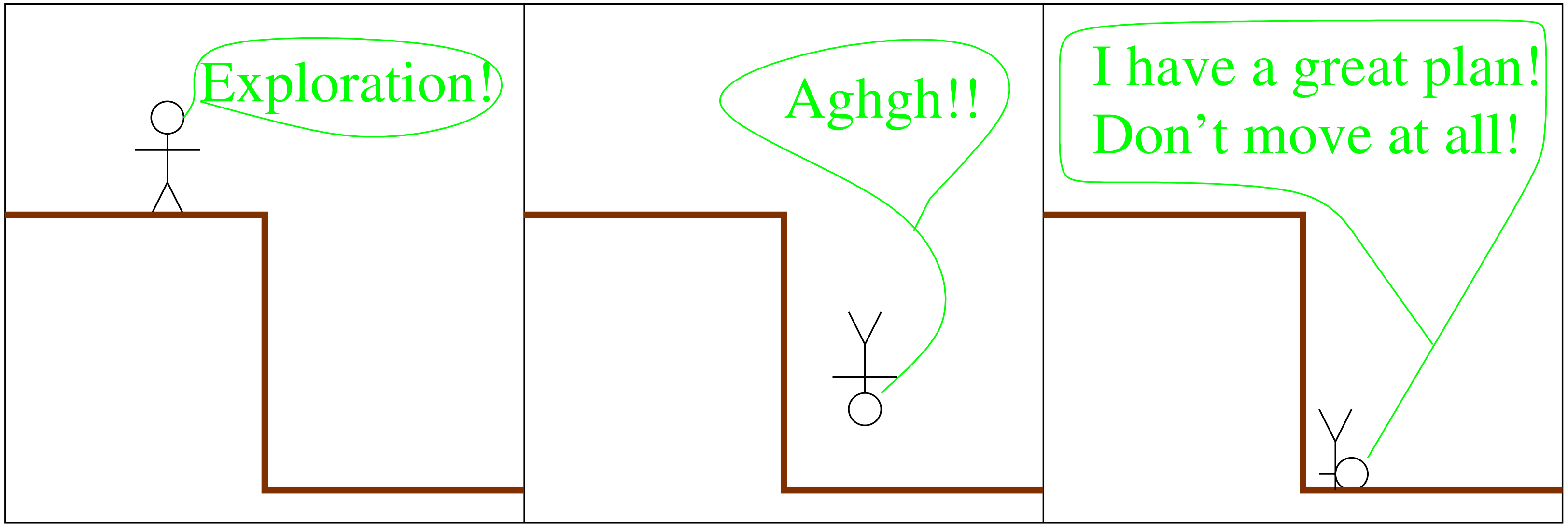
Theorem: There exist an algorithm Metric-$E^3(h)$ such that for all Metric MDPs, with probability $1-\delta$, for all except

$$\text{Poly}\left(Cover(MDP), T, \frac{1}{\epsilon}, \ln\frac{1}{\delta}\right)$$

steps $Q_{\text{Metric-}E^3}(s, E^3(h)) \geq V^*_{T-t \bmod T}(s) - \epsilon$ where h is the history of observations.

Problems: Can we reasonably expect Model to be that accurate?

Do we really want the guarantee these algorithms provide?



See Abbeel & Ng ICML 2005 for a bit of discussion.

Outline

1. Sample Complexity Results

2. Limitations of Sample Complexity

3. Reductions Results

Reduction-style

Philosophy: Start with what we know we can do (even if we can't prove it), then use it repeatedly to see what we can do.

In Machine Learning, we know how to classify. Can we (re)use this ability to solve Reinforcement Learning?

General form: "cost sensitive" classification D on $X \times [0, \infty)^k$:

$$\arg\min_h E_{x,\vec{c} \sim D} \left[c_{h(x)} \right]$$

(Reducible to binary classification. Reduction suggests minimizing $\sum_i c_i$ is good.)

How do we think about a general RL problem?

$D(o', r|(o, a, r)^*, o)$ = conditional probability table

1. $o', o \in O$ = observations

2. $r \in [0, \infty)$ = reward

3. $a \in A$ = action:

Find $\pi((o, a, r)^*, o) \rightarrow a$ maximizing:

$$\eta(\pi) = E_{(o,a,r)^T \sim \pi, D}\left[\sum_{t=1}^{T} r_t\right]$$

An “Algorithm” (Langford & Zadrozny 05)

Let π^*= optimal policy

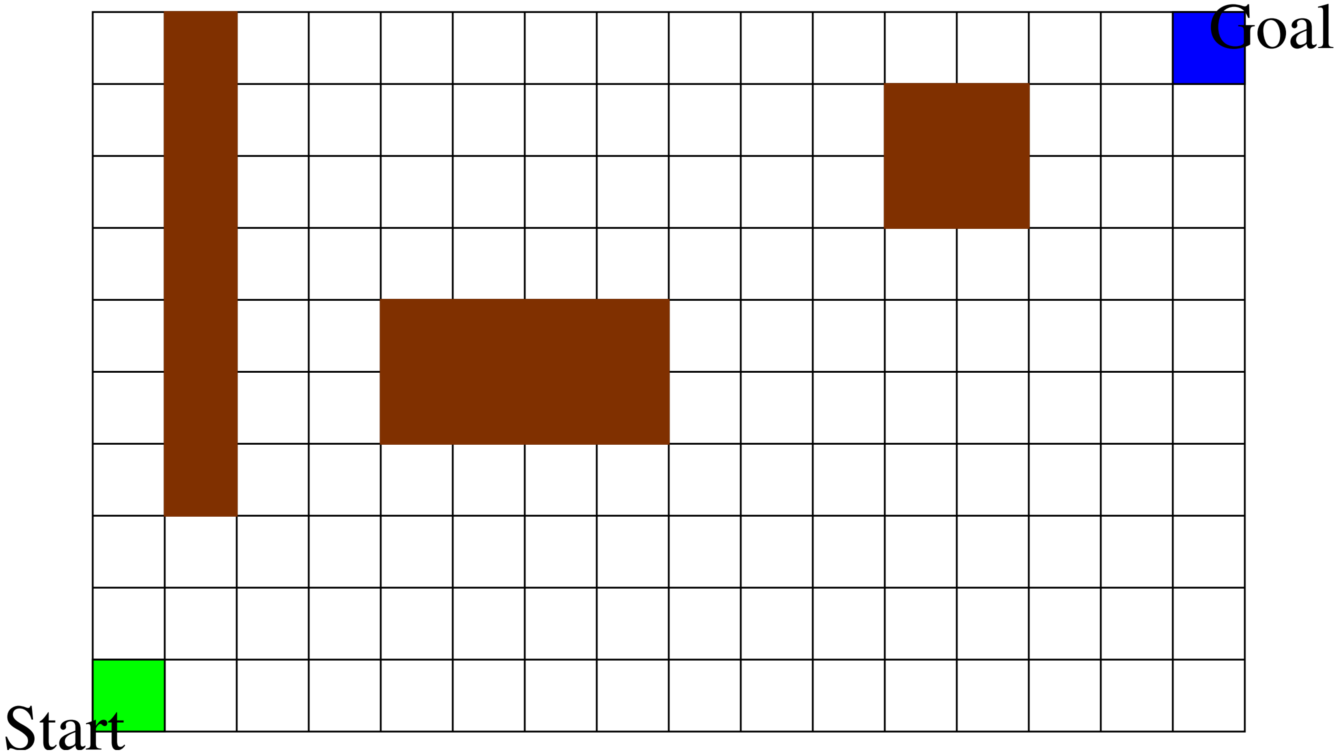
1. Given empty history, predict optimal a assuming π^* followed for $T-1$ timesteps afterwards.

2. Given first prediction, 1 step history, predict optimal a assuming π^* followed for $T-2$ timesteps afterwards.

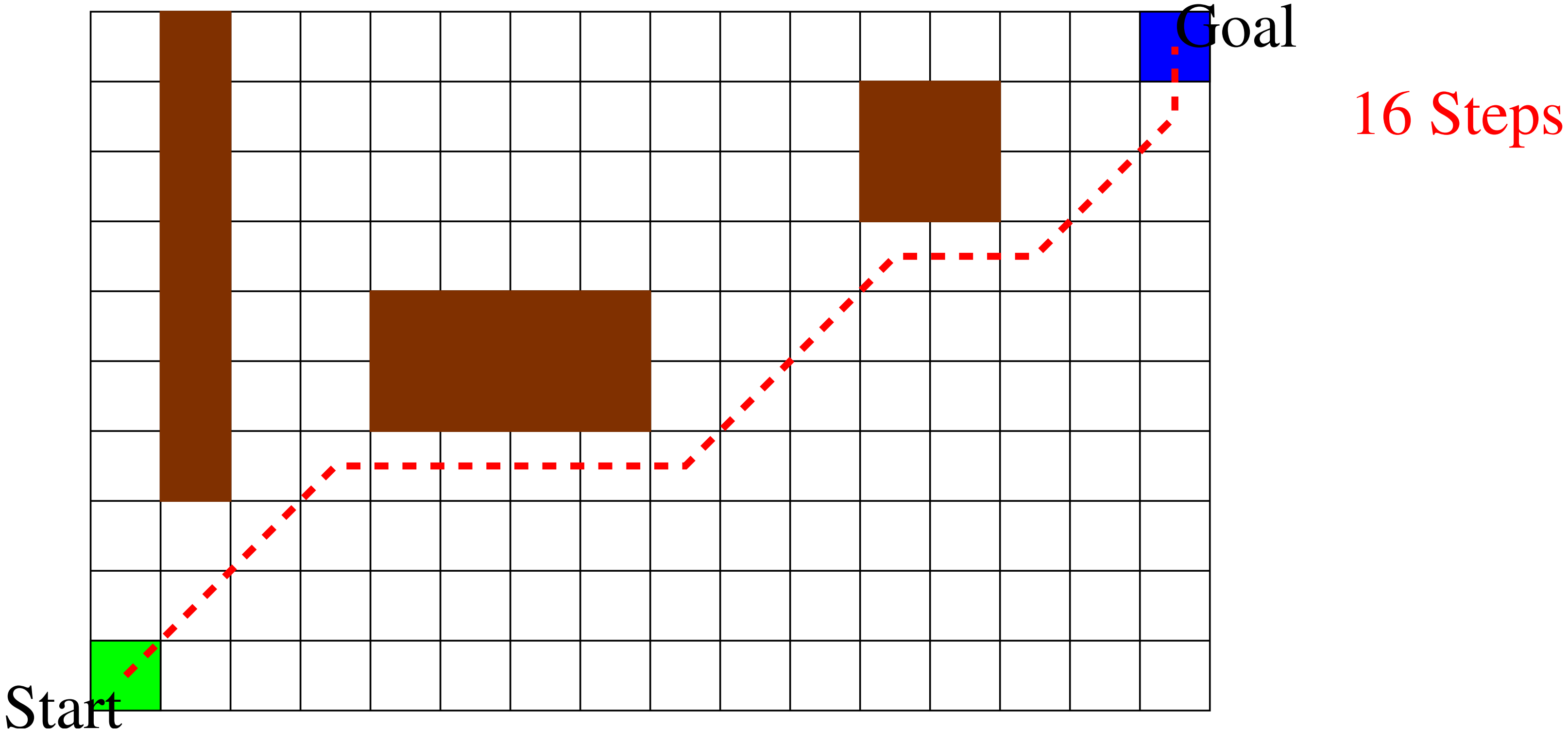
3. Given 2 step history, predict optimal a assuming π^* followed for $T-3$ timesteps afterwards.

4. ...

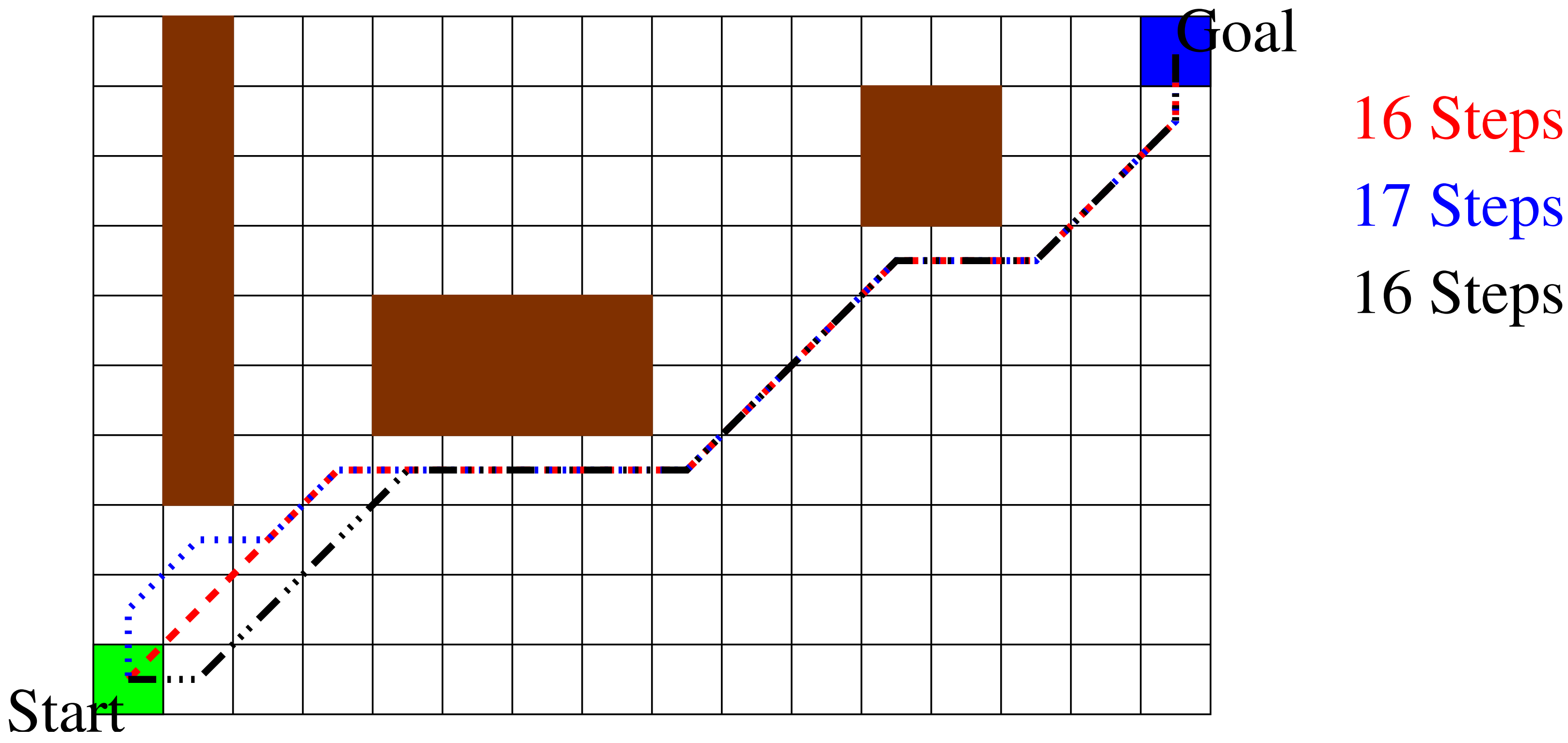
The Algorithm: Pretty Pictures



The Algorithm: Pretty Pictures

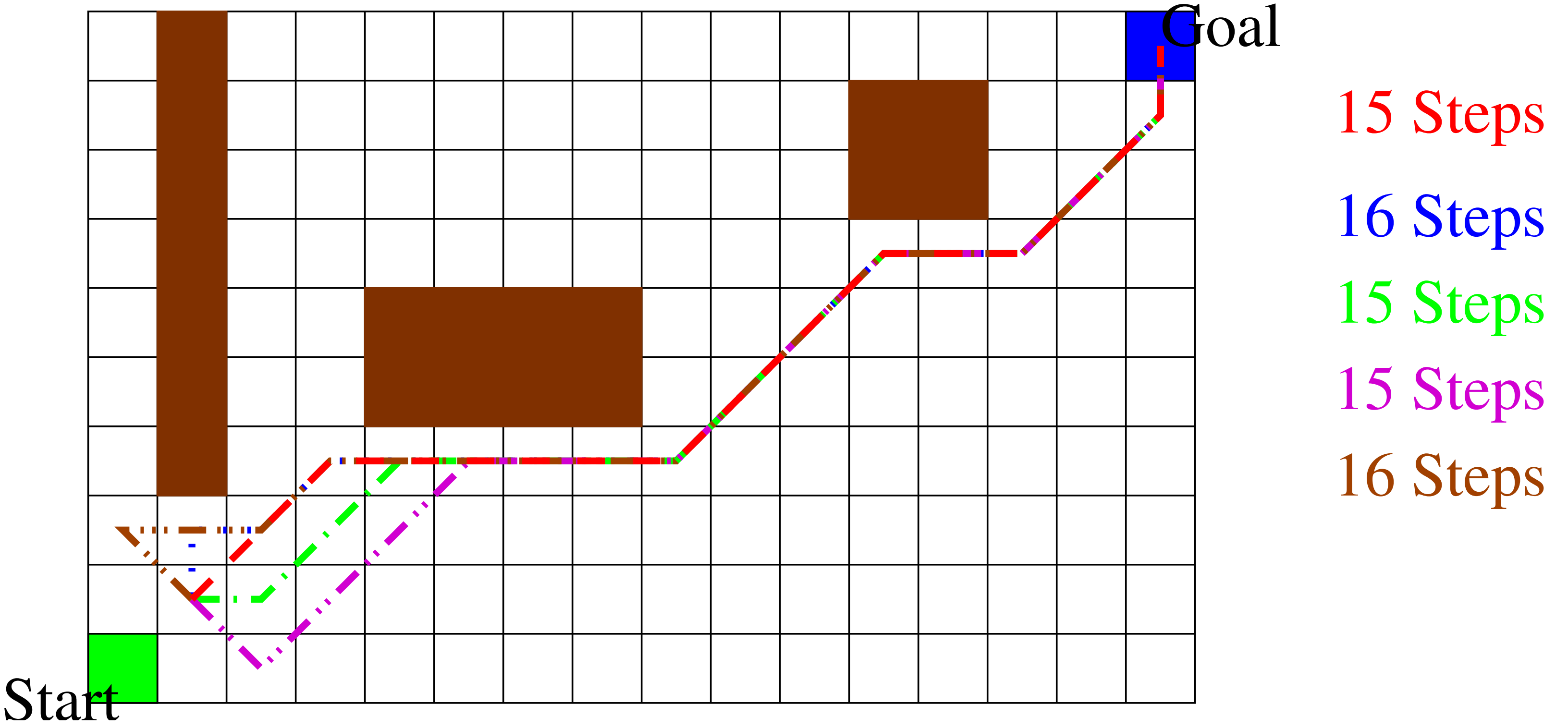


The Algorithm: Pretty Pictures



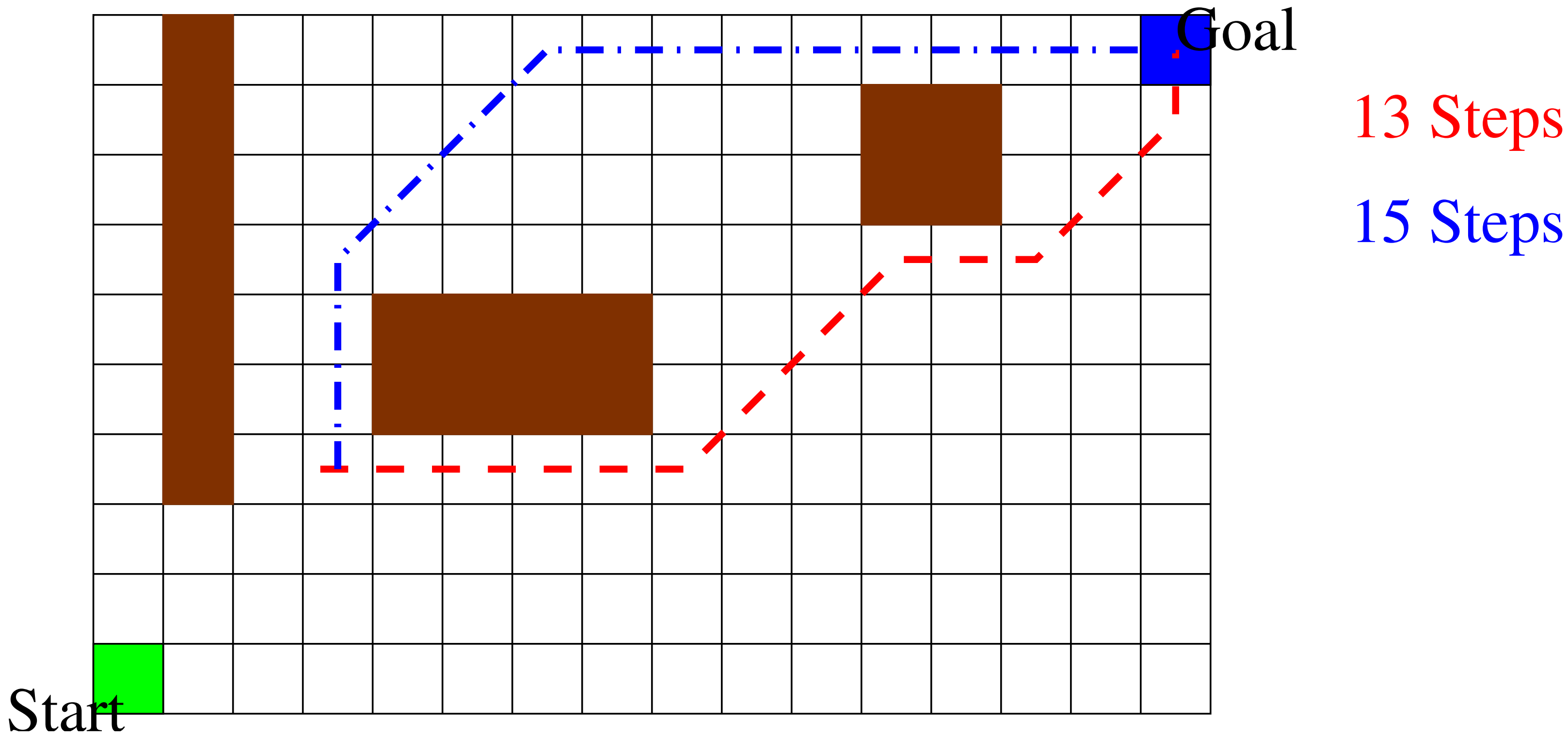
Cost for action $a = \text{steps}(a) - \min_{a'} \text{steps}(a') = 0, 1, 0$

The Algorithm: Pretty Pictures



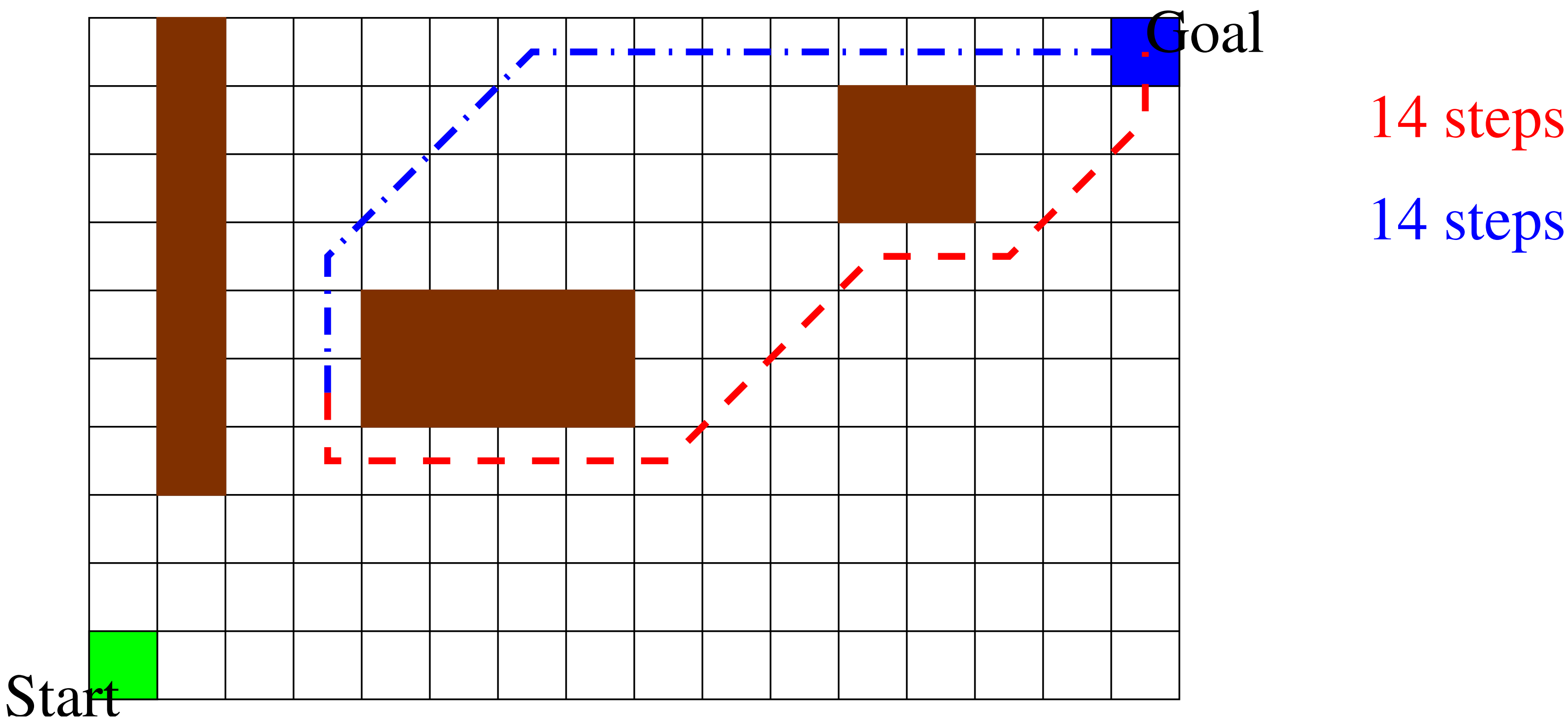
cost = 0, 1, 0, 0, 1

The Algorithm: Pretty Pictures



Cost $=$ 0, 2

The Algorithm: Pretty Pictures



Cost = 0, 0 Note: condition on suboptimal outcome at previous step.

The Critical Lemma

$l_t(D, \pi, h_t) = E_{(a,o,r)^T \ \ D,(\pi,h_t,\pi^*)} \left[\sum_{t'=t}^{T} r_{t'}\right]$ = Expected reward sum when starting with π, acting according to h_t and completing with the optimal policy.

Lemma: For all D, $\pi = (h_1, ..., h_T)$,

$$\eta(\pi^*) - \eta(\pi) = \sum_{t=1}^{T} l_t(D, \pi, \pi_t^*) - l_t(D, \pi, h_t)$$

= sum of cost sensitive losses.

Proof

$$\eta(\pi^*) - \eta(\pi) = E_{(o,a,r)^T \sim D, \pi^*}\left[\sum_{t=1}^T r_t\right] - E_{(o,a,r)^T \sim D, \pi}\left[\sum_{t=1}^T r_t\right]$$

$$= \sum_{t'=1}^T E_{(o,a,r)^T \sim D, (\pi, \pi^*_{t'}, \pi^*)}\left[\sum_{t=1}^T r_t\right] - E_{(o,a,r)^T \sim D, (\pi, \pi_{t'}, \pi^*)}\left[\sum_{t=1}^T r_t\right]$$

$$= \sum_{t'=1}^T E_{(o,a,r)^T \sim D, (\pi, \pi^*_{t'}, \pi^*)}\left[\sum_{t=t'}^T r_t\right] - E_{(o,a,r)^T \sim D, (\pi, h_{t'}, \pi^*)}\left[\sum_{t=t'}^T r_t\right]$$

$$= \sum_{t=1}^T l_t(D, \pi, \pi^*_t) - l_t(D, \pi, h_t)$$

Ways to think about the result

1. Reduction of RL to classification given oracle access to optimal policy. (A form of "apprenticeship learning".) [Surprisingly useful. See presentation Thursday@9am workshop on structured output spaces]

2. A *nonconstructive* reduction of RL to classification. There exists a set of classification problems such that if they are solved well, we solve RL. (But which problem is unclear.)

What can we do constructively?

Given side information, we can subvert the exploration problem.

1. Generative model (GM) (high exponential) (KMN 1999)

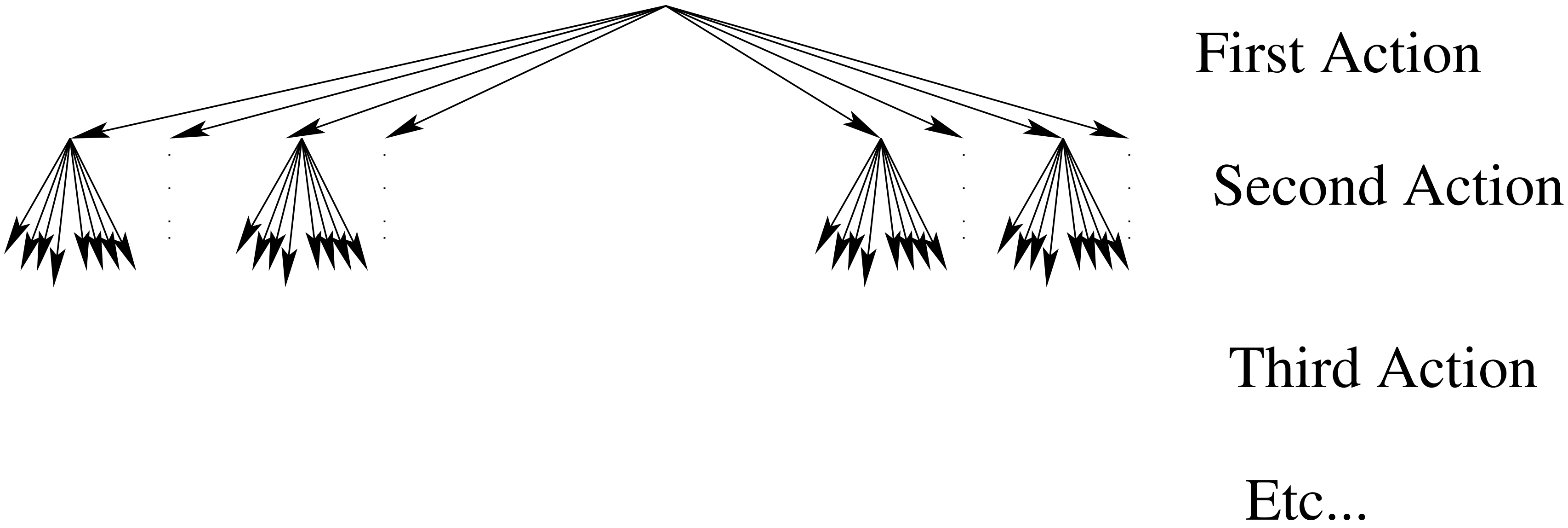
2. (Error transform) State distribution of (near) optimal policy + GM. (KL02 and BKNS 2003)

3. (Error transform) Generative model (lower exponential) or many traces with actions chosen randomly. (LZ 2003)

Sparse Sampling

1. For every first action, sample k times
2. For every result, for every second action, sample k times.
3. For every result, for every third action, sample k times.
4. etc..

Sparse Sampling, the picture



Picture is difficult, very high sample complexity.

Sparse Sampling Decisions

1. For each action, expectation over samples at step T

2. Max over actions.

3. For each action, expectation over T maxes at step $T-1$.

4. Etc...

$\Rightarrow$ Estimate of $l_t(D, \pi, h_t)$ accurate to $(T-t)\sqrt{\frac{\log A + \log \frac{1}{\delta}}{k}}$.

$\Rightarrow$ can apply reduction

The Future

1. Specializations of Reinforcement Learning to particular domains.

   (a) Robotics

   (b) Games

   (c) Stock Market

2. Exploration: the elephant in the closet.

3. New models and new methods of analysis needed and welcome.