Reinforcement Learning

Yishay Mansour
Google Inc. & Tel-Aviv University

Outline

- Goal of Reinforcement Learning
- Mathematical Model (MDP)
- Planning
- Learning
- Current Research issues

Goal of Reinforcement Learning

Goal oriented learning through interaction

Control of large scale stochastic environments with
partial knowledge.

Supervised / Unsupervised Learning
Learn from labeled / unlabeled examples

Reinforcement Learning - origins

Artificial Intelligence

Control Theory

Operation Research

Cognitive Science & Psychology

Solid foundations; well established research.

Typical Applications

- Robotics
  - Elevator control [CB].
  - Robo-soccer [SV].
- Board games
  - backgammon [T],
  - checkers [S].
  - Chess [B]
- Scheduling
  - Dynamic channel allocation [SB].
  - Inventory problems.

Contrast with Supervised Learning

The system has a “state”.

The algorithm influences the state distribution.

Inherent Tradeoff: Exploration versus Exploitation.

Mathematical Model - Motivation

Model of uncertainty:

Environment, actions, our knowledge.

Focus on decision making.

Maximize long term reward.

Markov Decision Process (MDP)

Mathematical Model - MDP

Markov decision processes

S- set of states

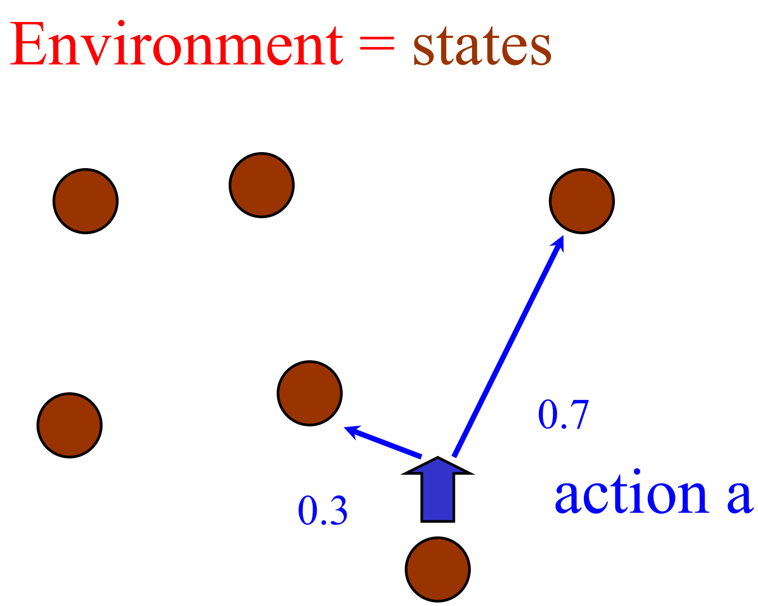
A- set of actions

δ - Transition probability

R - Reward function

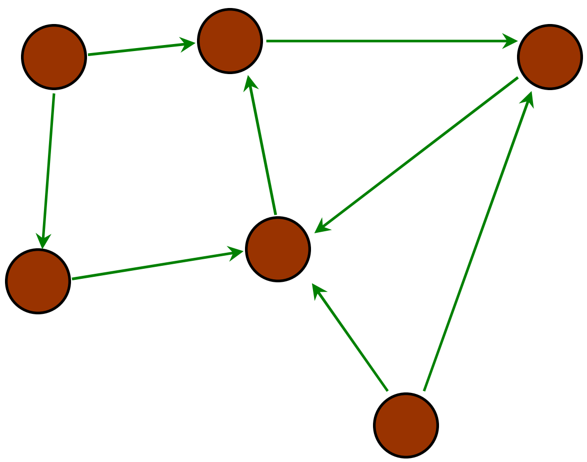
Similar to DFA!

MDP model - states and actions



Actions = transitions $\delta(s, a, s')$

MDP model - rewards

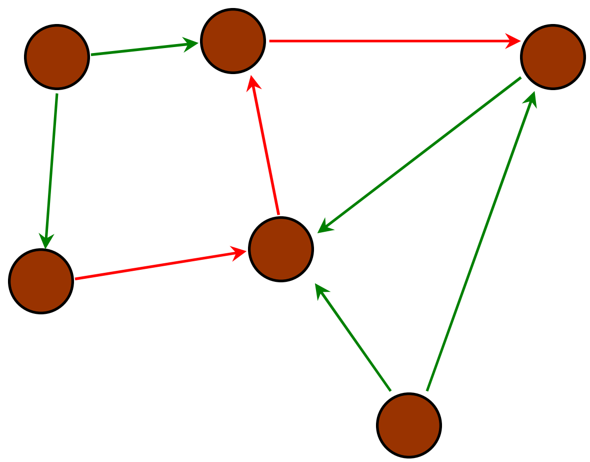


R(s,a) = reward at state ***s*** for doing action ***a*** (a random variable).

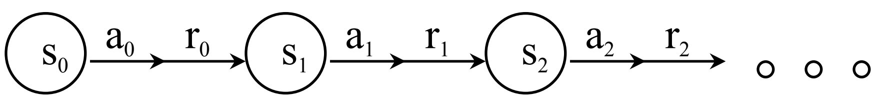
Example:

R(s,a) = -1 with probability 0.5
+10 with probability 0.35
+20 with probability 0.15

MDP model - trajectories



trajectory:



MDP - Return function.

Combining all the immediate rewards to a single value.

Modeling Issues:

Are early rewards more valuable than later rewards?

Is the system "terminating" or continuous?

Usually the return is linear in the immediate rewards.

MDP model - return functions

Finite Horizon - parameter H

$$return = \sum_{1 \le i \le H} R(s_i, a_i)$$

Infinite Horizon

discounted - parameter $\gamma < 1$.

$$return = \sum_{i=0}^{\infty} \gamma^i R(s_i, a_i)$$

undiscounted

$$\frac{1}{N} \sum_{i=0}^{N-1} R(s_i, a_i) \xrightarrow{N \to \infty} return$$

Terminating MDP

MDP model - action selection

AIM: Maximize the expected return.

Fully Observable - can "see" the "entire" state.

Policy - mapping from states to actions

Optimal policy: optimal from any start state.

THEOREM: There exists a deterministic optimal policy

Contrast with Supervised Learning

Supervised Learning:
Fixed distribution on examples.

Reinforcement Learning:
The state distribution is policy dependent!!!

A small local change in the policy can make a huge global change in the return.

MDP model - summary

$s \in S$ - set of states, $|S|=n$.

$a \in A$ - set of k actions, $|A|=k$.

$\delta(s_1, a, s_2)$ - transition function.

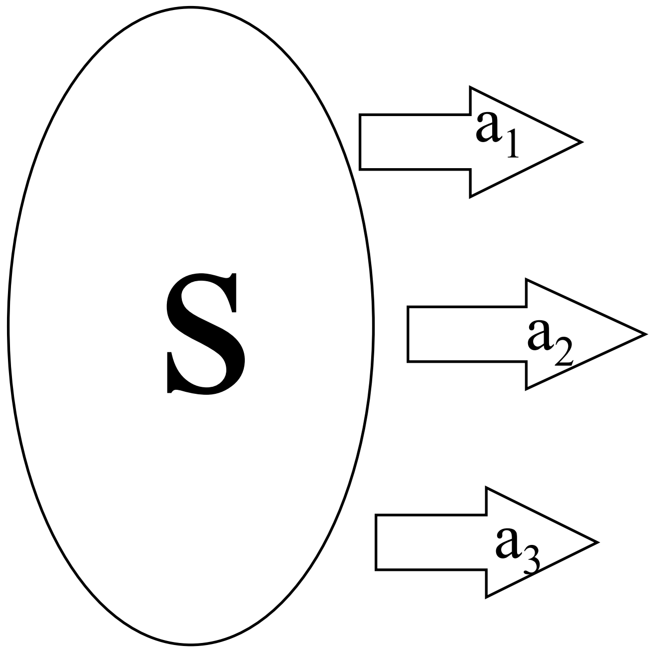
R(s,a) - immediate reward function.

$\pi : S \rightarrow A$ - policy.

$\sum_{i=0}^{\infty} \gamma^i r_i$ - discounted cumulative return.

Simple example: N- armed bandit

Single state.



Goal: Maximize sum of immediate rewards.

Given the model: Greedy action.

Difficulty: unknown model.

N-Armed Bandit: Highlights

- Algorithms (near greedy):
  - Exponential weights
    - G_i sum of rewards of action a_i
    - $w_i = \beta^{G_i}$
  - Follow the (perturbed) leader
- Results:
  - For any sequence of T rewards:
  - $E[online] > max_i \{G_i\}$ - $sqrt\{T \log N\}$

Planning - Basic Problems.

Given a complete MDP model.

Policy evaluation - Given a policy π, estimate its return.

Optimal control - Find an **optimal policy** π^* (maximizes the return from any start state).

Planning - Value Functions

$V^{\pi}(s)$ The expected return starting at state s following π.

$Q^{\pi}(s,a)$ The expected return starting at state s with action a and then following π.

$V^*(s)$ and $Q^*(s,a)$ are define using an optimal policy π^*.

$$V^*(s) = max_{\pi} V^{\pi}(s)$$

Planning - Policy Evaluation

Discounted infinite horizon (Bellman Eq.)

$$\mathbf{V^{\pi}(s) = E_{s' \sim \pi(s)} [\, R(s, \pi(s)) + \gamma V^{\pi}(s')]}$$

Rewrite the expectation

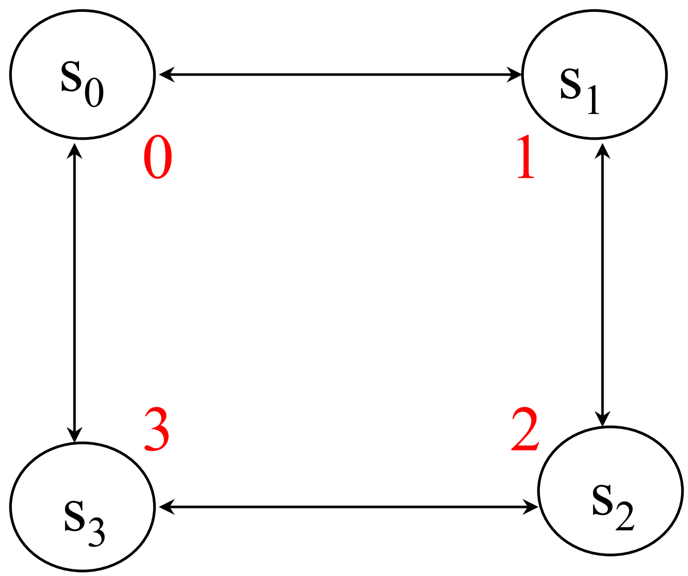
$$V^{\pi}(s) = E[R(s, \pi(s))] + \gamma \sum_{s'} \delta(s, \pi(s), s') V^{\pi}(s')$$

Linear system of equations.

Algorithms - Policy Evaluation
Example

$A=\{+1,-1\}$
$\gamma = 1/2$
$\delta(s_i,a)= s_{i+a}$
π random

$\forall a: R(s_i,a) = i$

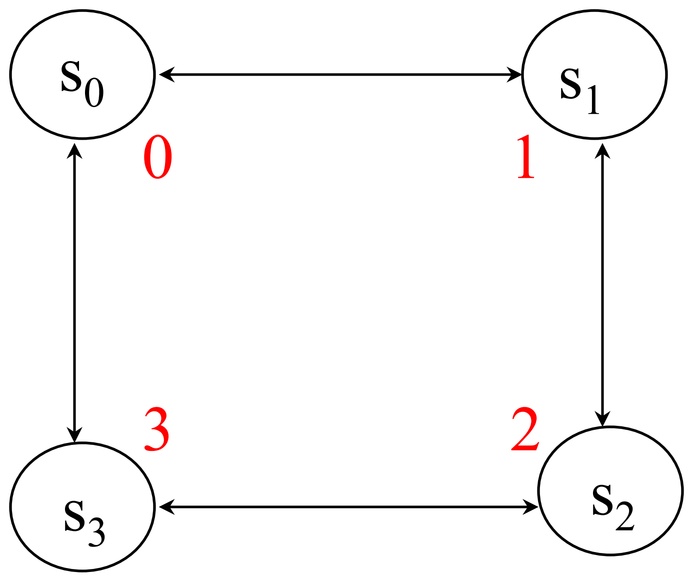


$$V^{\pi}(s_0) = 0 + \gamma\ [\pi(s_0,+1)V^{\pi}(s_1) + \pi(s_0,-1)\ V^{\pi}(s_3)\ ]$$

Algorithms -Policy Evaluation
Example

$A=\{+1,-1\}$
$\gamma = 1/2$
$\delta(s_i,a)= s_{i+a}$
π random

$\forall a\colon R(s_i,a) = i$



$V^\pi(s_0) = 5/3$
$V^\pi(s_1) = 7/3$
$V^\pi(s_2) = 11/3$
$V^\pi(s_3) = 13/3$

$$V^\pi(s_0) = 0 + (V^\pi(s_1) + V^\pi(s_3))/4$$

Algorithms - optimal control

State-Action Value function:

$$Q^{\pi}(s,a) = E\,[\,R(s,a)] + \gamma\, E_{s' \sim (s,a)}\,[\,V^{\pi}(s')]$$

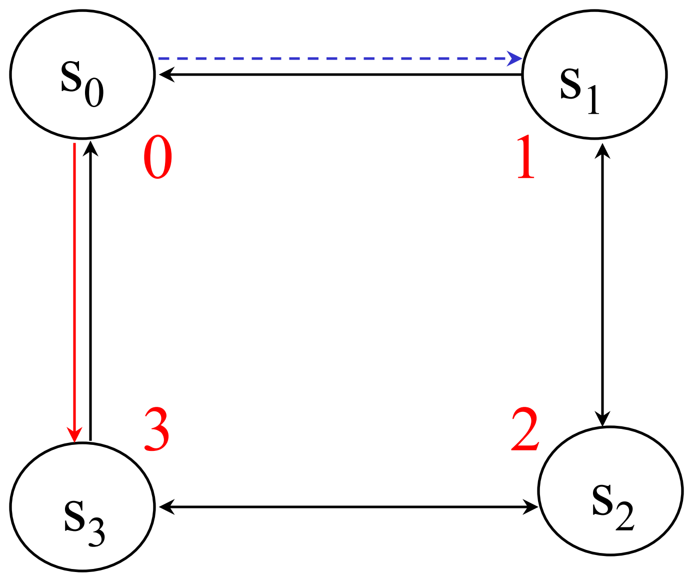
Note $V^{\pi}(s) = Q^{\pi}(s, \pi(s))$

For a deterministic policy π.

Algorithms -Optimal control
Example

$A=\{+1,-1\}$
$\gamma = 1/2$
$\delta(s_i,a)= s_{i+a}$
π random

$R(s_i,a) = i$



$Q^{\pi}(s_0,+1) = 5/6$
$Q^{\pi}(s_0,-1) = 13/6$

$Q^{\pi}(s_0,+1) = 0 + \gamma V^{\pi}(s_1)$

Algorithms - optimal control

CLAIM: A policy π is optimal if and only if at each state *s*:

$$V^{\pi}(s) = MAX_a \{Q^{\pi}(s,a)\} \qquad \textbf{(Bellman Eq.)}$$

PROOF: Assume there is a state *s* and action *a* s.t.,

$$V^{\pi}(s) < Q^{\pi}(s,a).$$

Then the strategy of performing *a* at state *s* (the first time) is better than π.

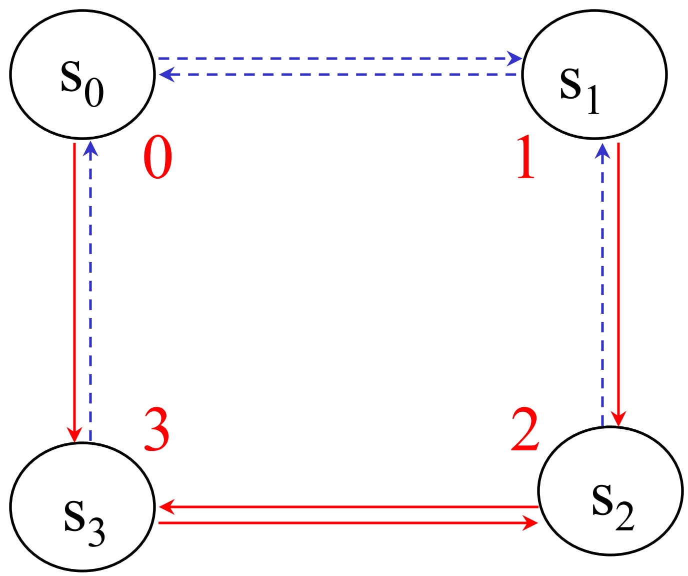
This is true each time we visit *s*, so the policy that performs action *a* at state *s* is better than π. □

Algorithms -optimal control

Example

$A=\{+1,-1\}$
$\gamma = 1/2$
$\delta(s_i,a)= s_{i+a}$
π random

$R(s_i,a) = i$



Changing the policy using the state-action value function.

MDP - computing optimal policy

1. Linear Programming

2. Value Iteration method.

$$V^{i+1}(s) \leftarrow \max_a \{R(s,a) + \gamma \sum_{s'} \delta(s,a,s')\, V^i(s')\}$$

3. Policy Iteration method.

$$\pi_i(s) = \arg\max_a \{Q^{\pi_{i-1}}(s,a)\}$$

Convergence: Value Iteration

- Distance of V^i from the optimal V^* (in L_∞)

$$Q^i(s,a) = R(s,a) + \gamma \sum_{s'} \delta(s,a,s')\, V^i(s')$$

$$V^*(s) - Q^i(s,a^*) = \gamma \sum_{s'} \delta(s,a^*,s')\, [V^*(s') - V^i(s')]$$

$$\leq \gamma \| V^* - V^i \|_\infty$$

$$V^*(s) - V^{i+1}(s) \leq V^*(s) - Q^i(s,a^*)$$

$$\| V^* - V^{i+1} \|_\infty \leq \gamma \| V^* - V^i \|_\infty$$

Convergence Rate: $1/(1-\gamma)$ ONLY Pseudo Polynomial

Convergence: Policy Iteration

- Policy Iteration Algorithm:
  - Compute $Q^{\pi}(s,a)$
  - Set $\pi(s) = \arg\max_a Q^{\pi}(s,a)$
  - Reiterate
- Convergence:
  - Policy can only improve
    $\forall s \;\; V^{t+1}(s) \geq V^{t}(s)$
    - Less iterations then Value Iteration, but
    - more expensive iterations.
- OPEN: How many iteration does it require ?!
  - LB: linear UB: $2^n/n$ (2-action MDP) [MS]

Outline

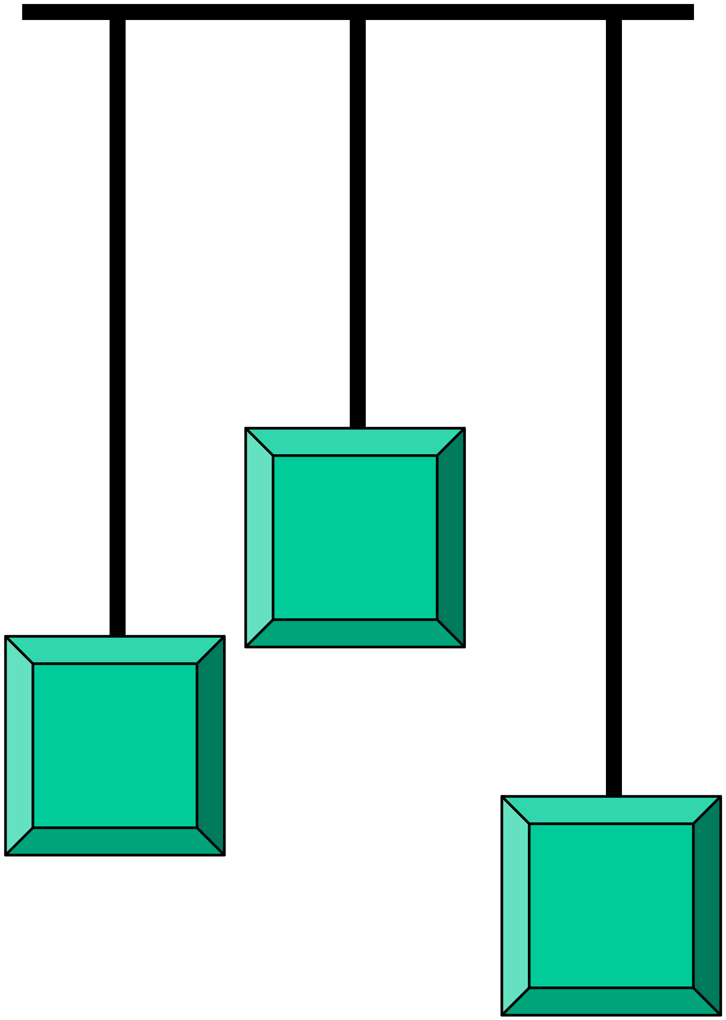
- Done
  - Goal of Reinforcement Learning
  - Mathematical Model (MDP)
  - Planning
    - Value iteration
    - Policy iteration
- Now: Learning Algorithms
  - Model based
  - Model Free

Planning versus Learning

Tightly coupled in Reinforcement Learning

Goal: maximize return while learning.

Example - Elevator Control



Learning (alone):
    Model the arrival model well.

Planning (alone) :
    Given arrival model build schedule

Real objective: Construct a schedule while updating model

Learning Algorithms

Given access only to actions perform:

1. policy evaluation.
2. control - find optimal policy.

Two approaches:

1. Model based (Dynamic Programming).
2. Model free (Q-Learning).

Learning - Model Based

Estimate the model from the observation.
(Both transition probability and rewards.)

Use the estimated model as the true model,
and find optimal policy.

If we have a “good” estimated model, we should
have a “good” estimation.

Learning - Model Based: off policy

- Let the policy run for a “long” time.
  - what is “long” ?!
  - Assuming some “exploration”
- Build an “observed model”:
  - Transition probabilities
  - Rewards
- Use the “observed model” to estimate value of the policy.

Learning - Model Based
sample size

Sample size (optimal policy):

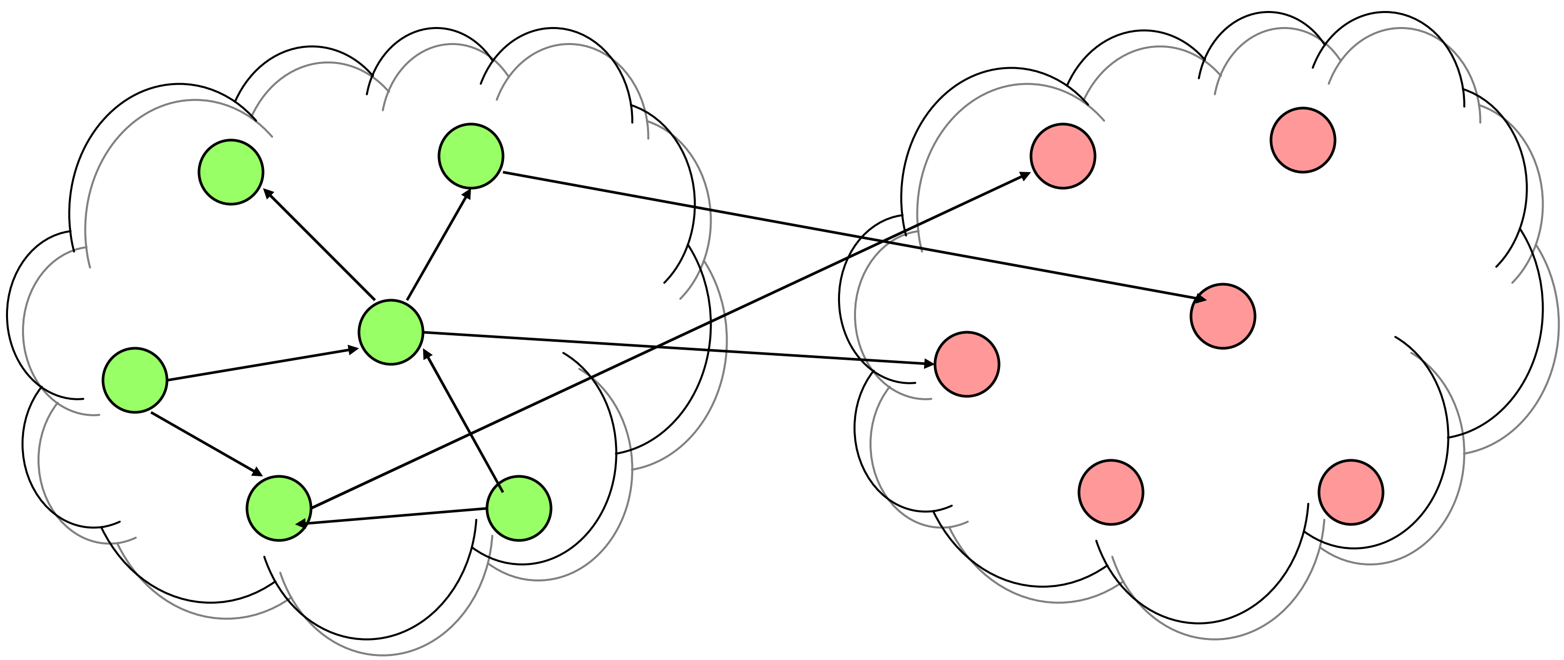
Naive: $O(|S|^2 |A| \log (|S| |A|))$ samples.
(approximates each transition $\delta(s,a,s')$ well.)

Better: $O(|S| |A| \log (|S| |A|))$ samples.
(Sufficient to approximate optimal policy.)
[KS, NIPS'98]

Learning - Model Based: on policy

- The learner has control over the action.
  - The immediate goal is to lean a model
- As before:
  - Build an "observed model":
    - Transition probabilities and Rewards
  - Use the "observed model" to estimate value of the policy.
- Accelerating the learning:
  - How to reach "new" places ?!

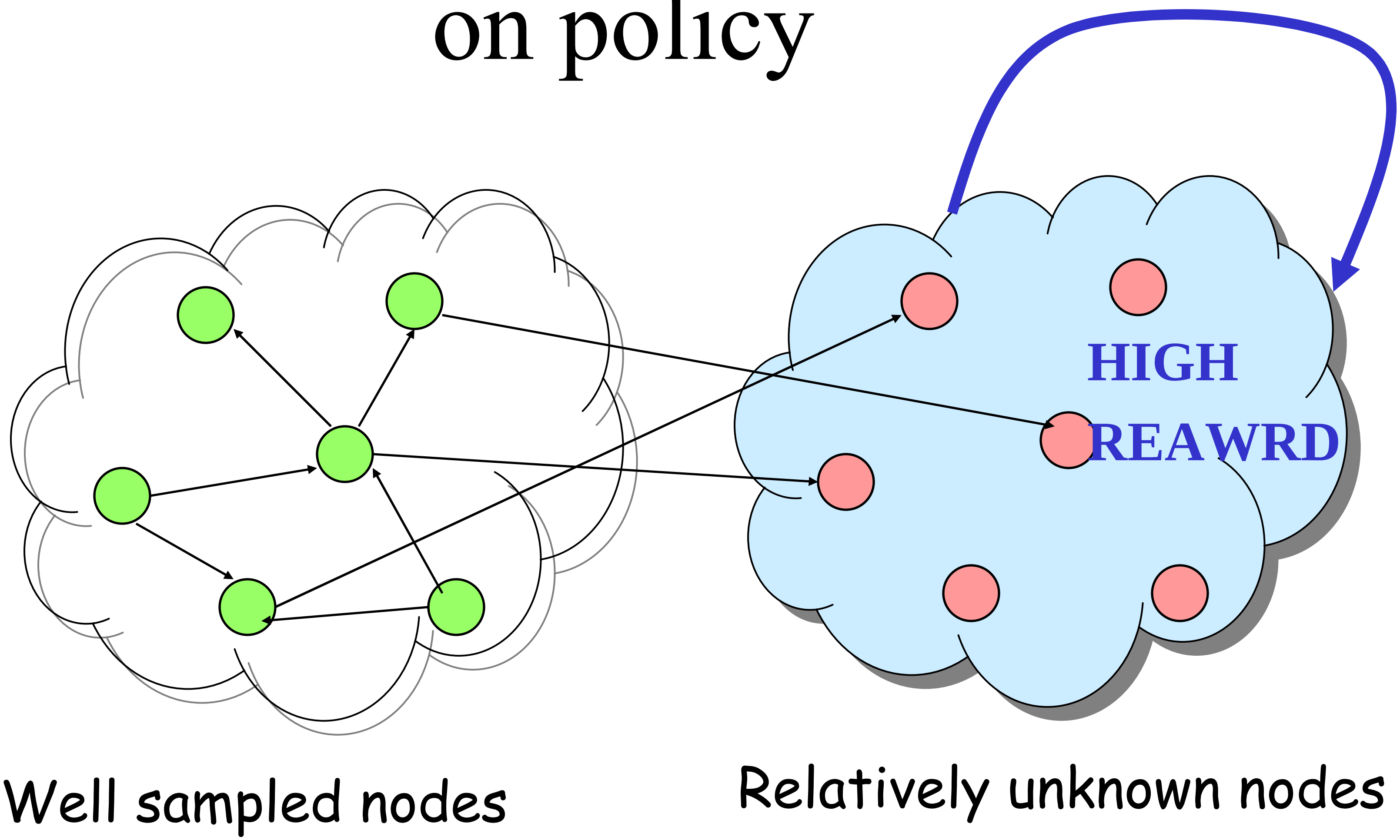
Learning - Model Based: on policy



Well sampled nodes

Relatively unknown nodes

Learning - Model Based: on policy



Exploration → Planning in new MDP

Learning: Policy improvement

- Assume that we can perform:
  - Given a policy π,
  - Estimate V and Q functions of π
- Can run policy improvement:
  - π = Greedy (Q)
- Process converges if estimations are accurate.