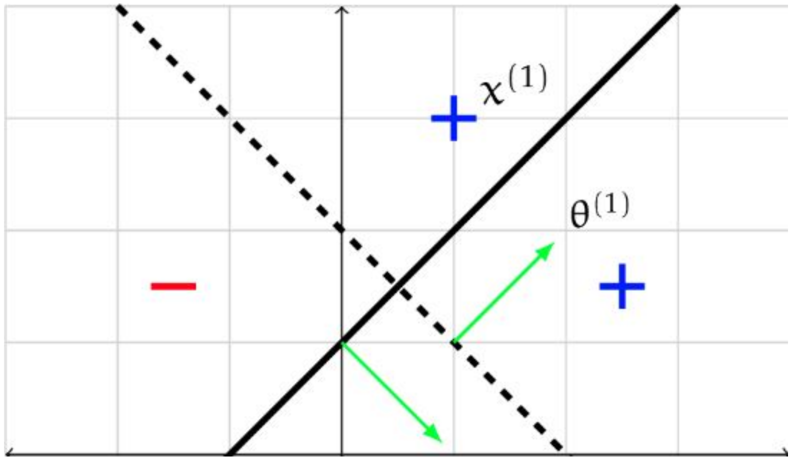
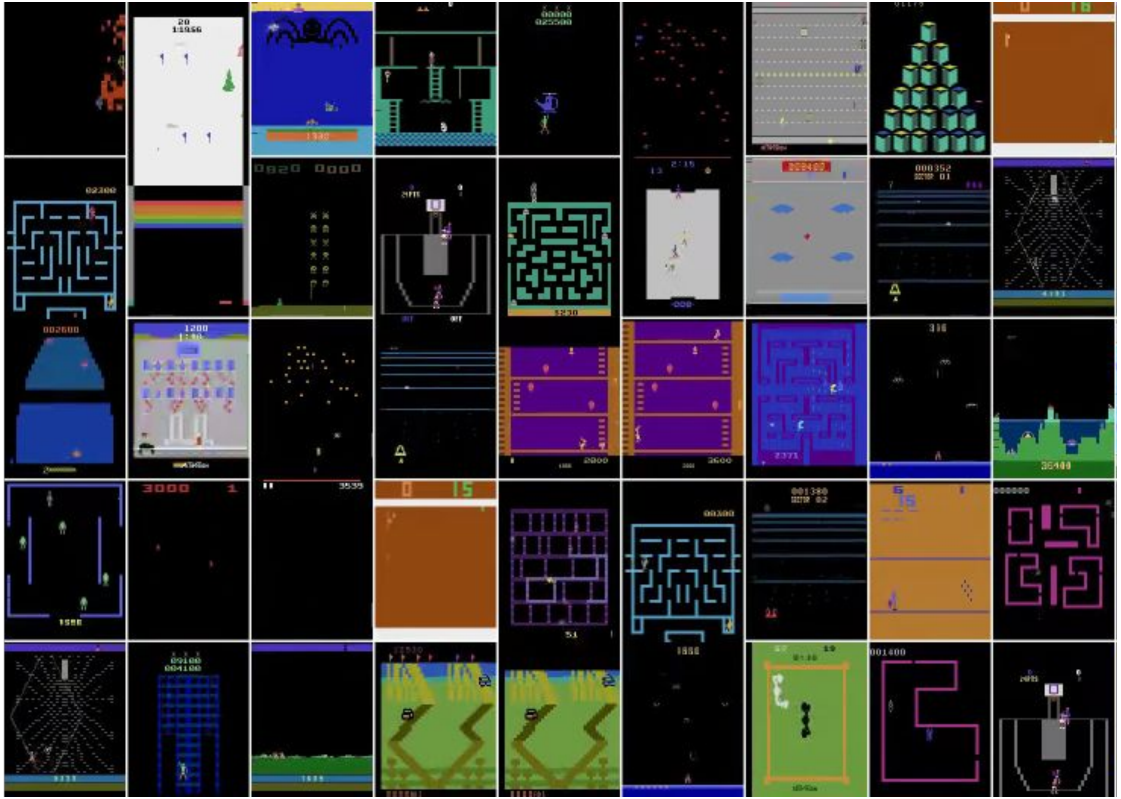
Introduction to Machine Learning



Week 10: MDPs




ALPHAGO
00:00:54
AlphaGo
Lee Sedol


14:32
Catalyst LE
Workers: 3/3
Workers: 25/16
Workers: 3/3
0 AlphaStar 177 /200 945 +2015 758 +873 64 113 940
SUPPLY MINERALS GAS WORKERS ARMY APM PRODUCTION
0 LiquidTLO 147 /172 335 +1595 442 +1030 61 86 1377

Markov Decision Process

Markov Decision Process

$$(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, s_0)$$

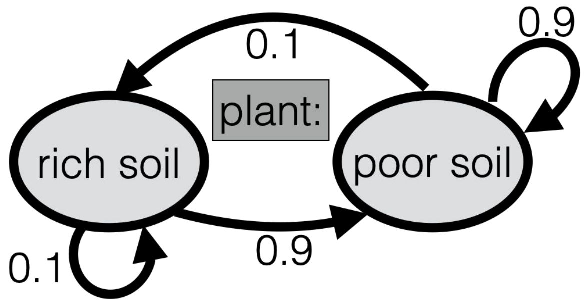
- $\mathcal{S}$ = set of possible states
- $\mathcal{A}$ = set of possible actions
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$: transition model
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: reward function
- γ = discount factor

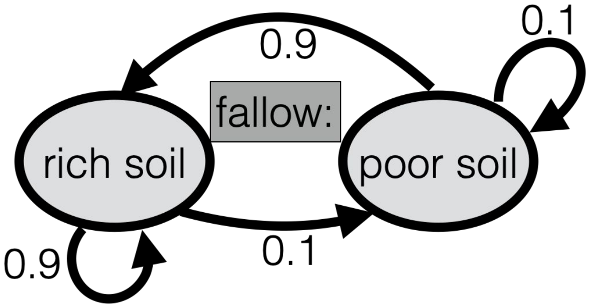
Markov Decision Process

Markov Decision Process

$$(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, s_0)$$

- $\mathcal{S}$ = set of possible states
- $\mathcal{A}$ = set of possible actions
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$: transition model
- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$: reward function
- γ = discount factor



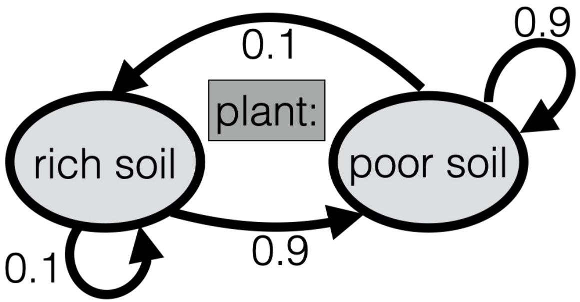


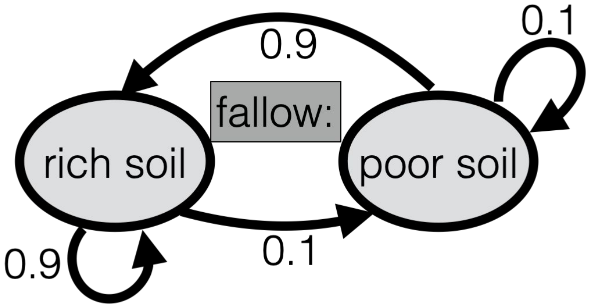
Markov Decision Process

Markov Decision Process

$$(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, s_0)$$

- $\mathcal{S}$ = set of possible states
- $\mathcal{A}$ = set of possible actions
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$: transition model
- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$: reward function
- γ = discount factor





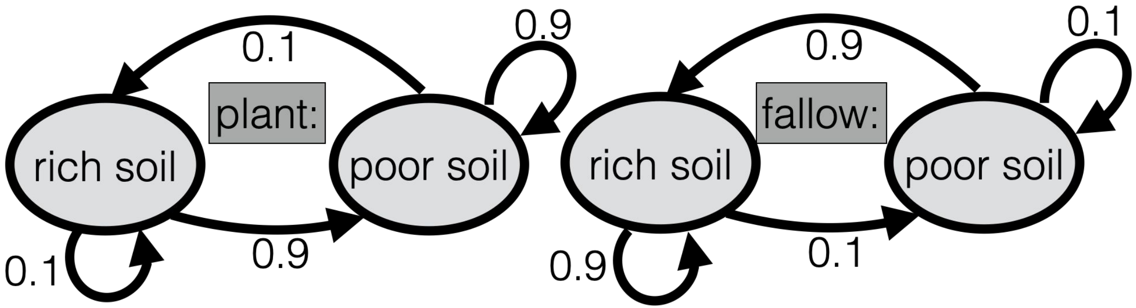
Goal: find a "policy" $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximizes reward

Value of a policy

- Given an MDP and a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ we can find the value of a policy by solving a system of linear equations.

Value of a policy

- Given an MDP and a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ we can find the value of a policy by solving a system of linear equations.



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

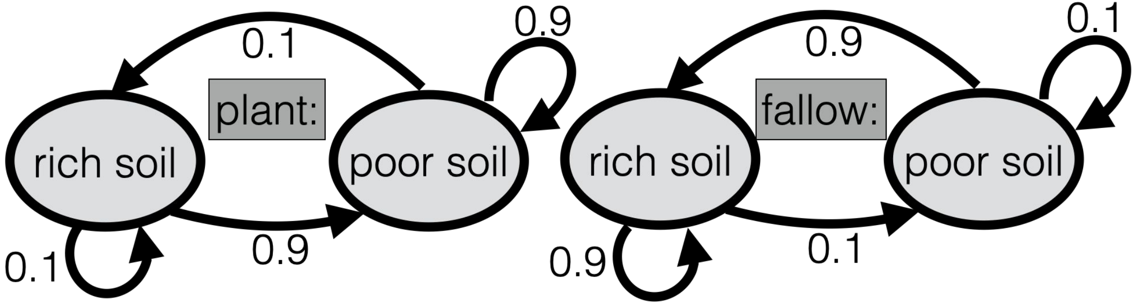
$$V_\pi^0(s) = 0; V_\pi^h(s) = R(s, \pi(s)) + \sum_{s'} T(s, \pi(s), s') \cdot V_\pi^{h-1}(s')$$

- $V_\pi^h(s)$: value of the policy with *h* steps left
- $R(s, \pi(s))$: value of the policy on this time step
- $\sum_{s'} T(s, \pi(s), s') \cdot V_\pi^{h-1}(s')$: (expected) value of the policy across all future time steps

- *h*: horizon (e.g. how many growing seasons left)
- $V_\pi^h(s)$: value (expected reward) with policy π starting at *s*

Value of a policy

- Given an MDP and a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ we can find the value of a policy by solving a system of linear equations.



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

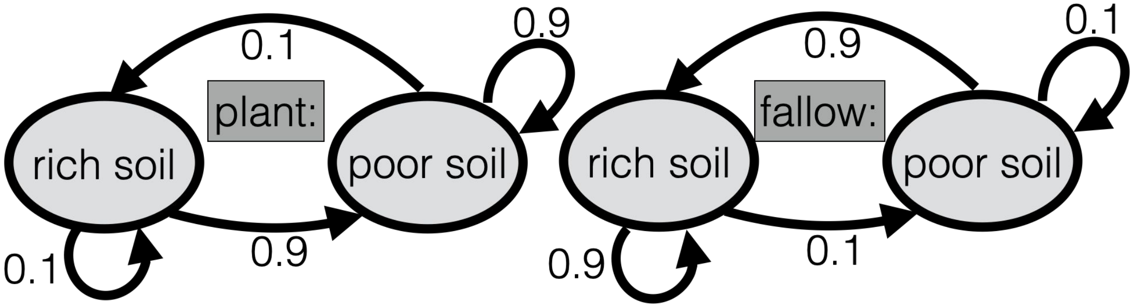
$$V_\pi^0(s) = 0; V_\pi^h(s) = R(s, \pi(s)) + \sum_{s'} T(s, \pi(s), s') \cdot V_\pi^{h-1}(s')$$

$$V_\pi(s) = R(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s') V_\pi(s')$$

- *h*: horizon (e.g. how many growing seasons left)
- $V_\pi^h(s)$: value (expected reward) with policy π starting at *s*

Can use to evaluate which policy is better.
How to compute best policy?

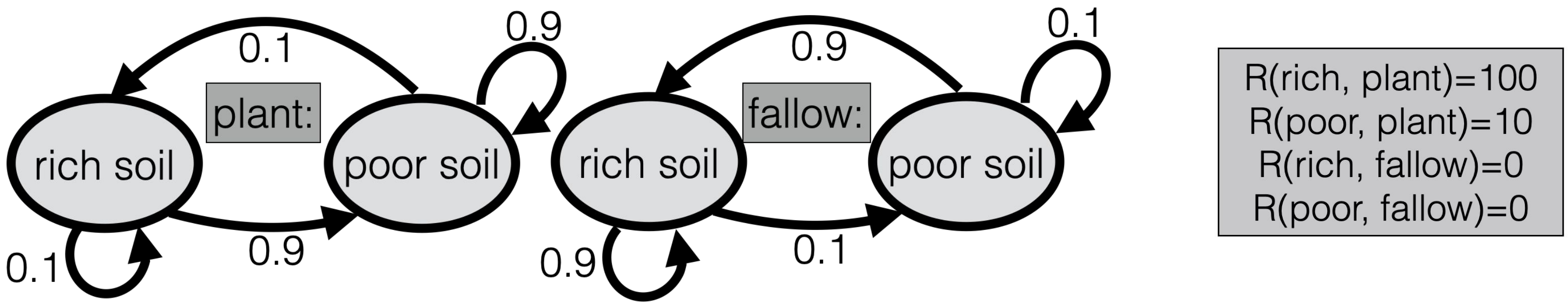
Optimal policy in a known MDP



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s, a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s, a)$

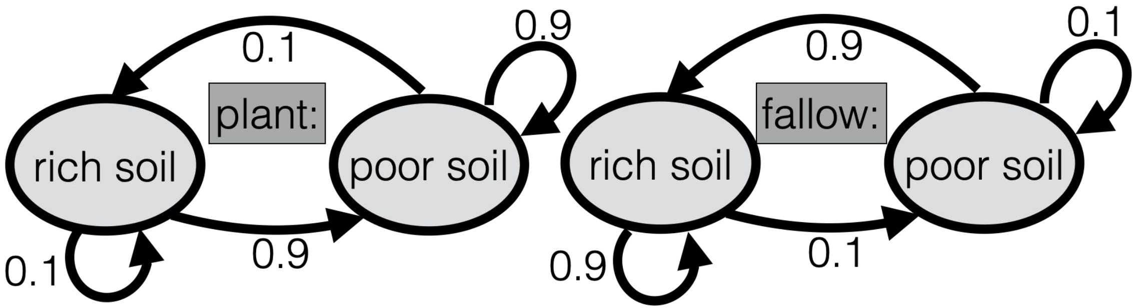
Optimal policy in a known MDP



- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s, a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s, a)$

$$Q^0(s, a) = 0; Q^h(s, a) = R(s, a) + \sum_{s'} T(s, a, s') \max_{a'} Q^{h-1}(s', a')$$

Optimal policy in a known MDP



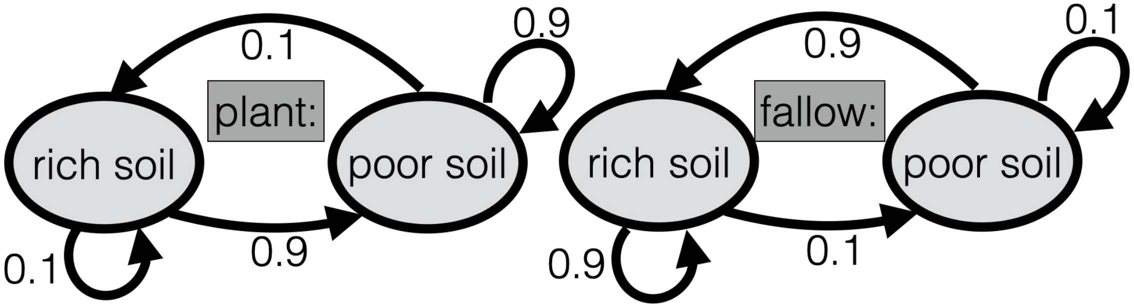
R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s, a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s, a)$

$$Q^0(s, a) = 0; Q^h(s, a) = R(s, a) + \sum_{s'} T(s, a, s') \max_{a'} Q^{h-1}(s', a')$$

$$Q^1(\text{rich}, \text{plant}) = 100; Q^1(\text{rich}, \text{fallow}) = 0;\ Q^1(\text{poor}, \text{plant}) = 10; Q^1(\text{poor}, \text{fallow}) = 0$$

Optimal policy in a known MDP



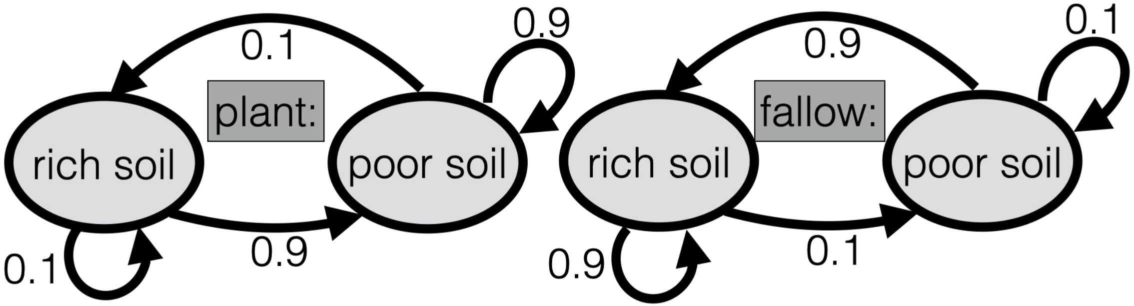
- h: horizon (e.g. how many planting seasons)
- $Q^h(s, a)$: expected reward of starting at s, making action a, and then making the "best" action for the h-1 steps left
- With Q, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s, a)$

$$Q^0(s, a) = 0; Q^h(s, a) = R(s, a) + \sum_{s'} T(s, a, s') \max_{a'} Q^{h-1}(s', a')$$

$$Q^1(\text{rich}, \text{plant}) = 100; Q^1(\text{rich}, \text{fallow}) = 0;\ Q^1(\text{poor}, \text{plant}) = 10; Q^1(\text{poor}, \text{fallow}) = 0$$

$$Q^2(\text{rich}, \text{plant}) = R(\text{rich}, \text{plant}) + T(\text{rich}, \text{plant}, \text{rich}) \max_{a'} Q^1(\text{rich}, a') + T(\text{rich}, \text{plant}, \text{poor}) \max_{a'} Q^1(\text{poor}, a')$$

Optimal policy in a known MDP



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

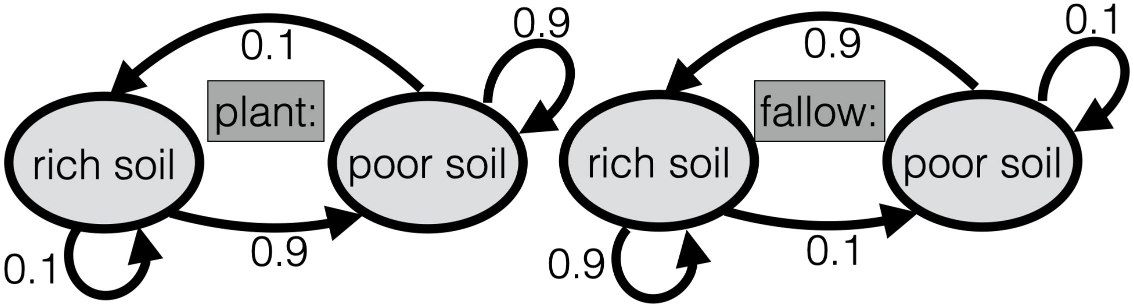
- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s,a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s,a)$

$$Q^0(s,a) = 0; Q^h(s,a) = R(s,a) + \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

$$Q^1(\text{rich}, \text{plant}) = 100; Q^1(\text{rich}, \text{fallow}) = 0;\ Q^1(\text{poor}, \text{plant}) = 10; Q^1(\text{poor}, \text{fallow}) = 0$$

$$Q^2(\text{rich}, \text{plant}) = 100 + (0.1)(100) + (0.9)(10) = 119$$

Optimal policy in a known MDP



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

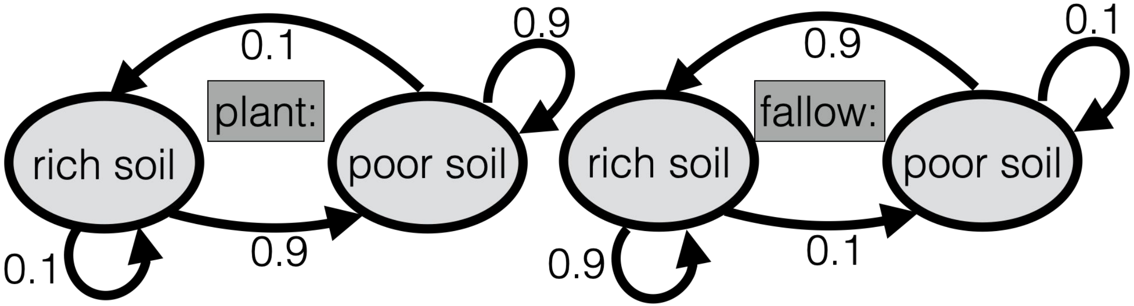
- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s,a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi^*_h(s) = \arg\max_a Q^h(s,a)$

$$Q^0(s,a) = 0; Q^h(s,a) = R(s,a) + \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

$$Q^1(\text{rich},\text{plant}) = 100; Q^1(\text{rich},\text{fallow}) = 0;\ Q^1(\text{poor},\text{plant}) = 10; Q^1(\text{poor},\text{fallow}) = 0$$

$$Q^2(\text{rich},\text{plant}) = 119; Q^2(\text{rich},\text{fallow}) = 91; Q^2(\text{poor},\text{plant}) = 29; Q^2(\text{poor},\text{fallow}) = 91$$

(Finite-Horizon) Value Iteration



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

- *h*: horizon (e.g. how many planting seasons)
- $Q^h(s, a)$: expected reward of starting at *s*, making action *a*, and then making the "best" action for the *h*-1 steps left
- With *Q*, can find **an optimal policy**: $\pi_h^*(s) = \arg\max_a Q^h(s, a)$

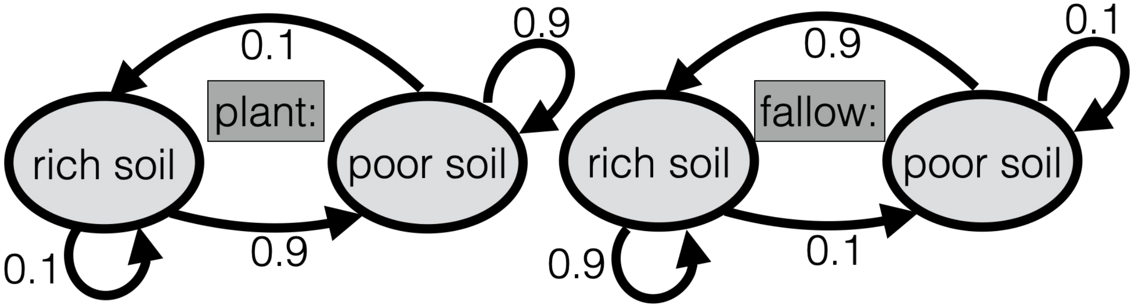
$$Q^0(s, a) = 0; Q^h(s, a) = R(s, a) + \sum_{s'} T(s, a, s') \max_{a'} Q^{h-1}(s', a')$$

$$Q^1(\text{rich, plant}) = 100; Q^1(\text{rich, fallow}) = 0;\ Q^1(\text{poor, plant}) = 10; Q^1(\text{poor, fallow}) = 0$$

$$Q^2(\text{rich, plant}) = 119; Q^2(\text{rich, fallow}) = 91; Q^2(\text{poor, plant}) = 29; Q^2(\text{poor, fallow}) = 91$$

What's best? Any *s*, $\pi_1^*(s) = \text{plant}; \pi_2^*(\text{rich}) = \text{plant}, \pi_2^*(\text{poor}) = \text{fallow}$

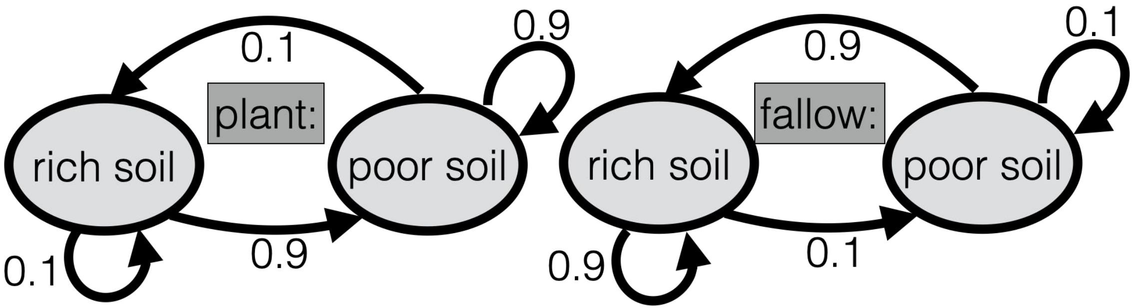
(Infinite-Horizon) Value Iteration



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

- What if I don't stop farming? Is there any optimal policy?
- **Theorem**. There exists a (stationary) optimal policy π^*. I.e., for every policy π and for every state $s \in \mathcal{S}$, $V_{\pi^*}(s) \geq V_\pi(s)$

(Infinite-Horizon) Value Iteration



R(rich, plant)=100
R(poor, plant)=10
R(rich, fallow)=0
R(poor, fallow)=0

- What if I don't stop farming? Is there any optimal policy?
- **Theorem**. There exists a (stationary) optimal policy π^*. I.e., for every policy π and for every state $s \in \mathcal{S}$, $V_{\pi^*}(s) \geq V_\pi(s)$
- $Q^*(s,a)$: expected reward if we make best actions in future
  - If we knew $Q^*(s,a)$, then: $\pi^*(s) = \arg\max_a Q^*(s,a)$
- Note: $Q^*(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^*(s',a')$
  - Not linear in $Q^*(s,a)$, so not as easy to solve as $V_\pi(s)$

(Infinite-Horizon) Value Iteration

Finite-horizon value iteration:

$$Q^0(s,a) = 0 \quad Q^1(s,a) = R(s,a)$$

$$Q^h(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

(Infinite-Horizon) Value Iteration

Finite-horizon value iteration:

$$Q^0(s,a) = 0 \quad Q^1(s,a) = R(s,a)$$

$$Q^h(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

```
Infinite-Horizon-Value-Iteration(𝒮, 𝒜, T, R, γ, ε)
  for each state s ∈ 𝒮 and each action a ∈ 𝒜
    Initialize Q_old(s, a) = 0
```

(Infinite-Horizon) Value Iteration

Finite-horizon value iteration:

$$Q^0(s,a) = 0 \quad Q^1(s,a) = R(s,a)$$

$$Q^h(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

Infinite-Horizon-Value-Iteration $(\mathcal{S}, \mathcal{A}, T, R, \gamma, \epsilon)$

  for each state $s \in \mathcal{S}$ and each action $a \in \mathcal{A}$

    Initialize $Q_{\text{old}}(s,a) = 0$

  while True

    for each state $s \in \mathcal{S}$ and each action $a \in \mathcal{A}$

      $Q_{\text{new}}(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q_{\text{old}}(s',a')$

(Infinite-Horizon) Value Iteration

Finite-horizon value iteration:

$$Q^0(s,a) = 0 \quad Q^1(s,a) = R(s,a)$$

$$Q^h(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^{h-1}(s',a')$$

```
Infinite-Horizon-Value-Iteration(𝒮, 𝒜, T, R, γ, ε)
  for each state s ∈ 𝒮 and each action a ∈ 𝒜
    Initialize Q_old(s, a) = 0
  while True
    for each state s ∈ 𝒮 and each action a ∈ 𝒜
      Q_new(s, a) = R(s, a) + γ Σ_{s'} T(s, a, s') max_{a'} Q_old(s', a')
    if max_{s,a} |Q_old(s, a) − Q_new(s, a)| < ε
      return Q_new
    Q_old = Q_new
```