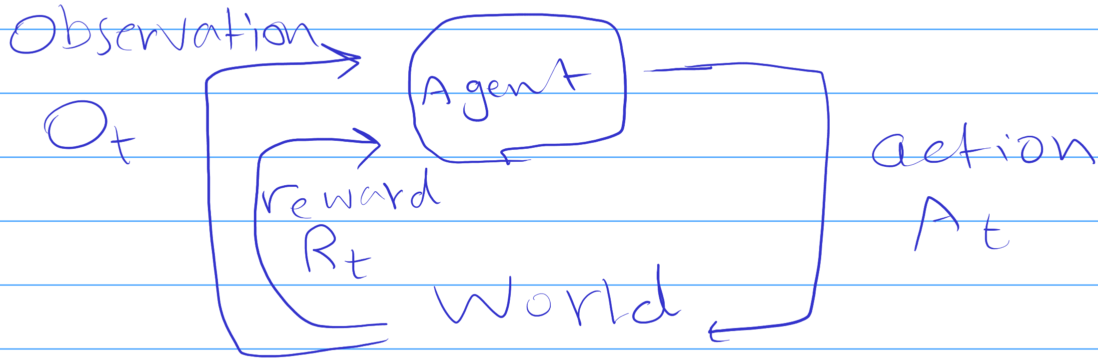


history $\Rightarrow H_t = A_1, O_1, R_1, \ldots A_t O_t R_t$

State → environment state S_t^e
State → agent state S_t^a

☆ S_t is Markov $\iff P(S_{t+1} | S_t) = P(S_{t+1} | H_t)$

the state is a sufficiant statistic of the future.

☆ environment types

① Fully Observable $O_t = S_t^a = S_t^e$
(Markov Decision Process)

② Partially Observable (POMDP)

Method 1 → $S_t^a = H_t$ naive approach

Method 2 → $S_t^a = (P(S_t^e = s^1), \ldots, P(S_t^e = s^n))$ baysian probabilistic approach (Belief)

Method 3 → $S_t^a = \sigma(S_{t-1}^a W_s + O_t W_o)$ recurrent neural networks

an RL agent can have →
- Policy
- Value function
- Model

policy

$S \longrightarrow \boxed{\pi} \longrightarrow A$

map from state to action

↳ deterministic $a = \pi(s)$

↳ probabilistic $\pi(a|s) = P[A=a \mid S=s]$

Value function

prediction of expected future reward

$$v_\pi(s) = E_\pi[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots \mid S_t = s]$$

Model

predicts what environment will do next

↳ transition Model $P^a_{ss'} = P[S' = s' \mid S = s, A = a]$

↳ Rewards Model $R^a_s = E[R \mid S = s, A = a]$

Categorization of RL agents

① value-based (policy ✗ value ✓)

② policy-based (policy ✓ value ✗)

③ Actor-Critic (policy ✓ value ✓)

① Model-free (policy /Value ✓ model ✗)

② Model-based (policy / Value ✓ model ✓)

Sequential Decision Making

① reinforcement learning (through interaltion)

↳ exploration vs expoilation

↳ prediction (given a policy) Vs Control (optimizing a Policy)

② planning (no interaction needed)

Markov Decision Processes (MDP)

★ the environment is fully observable
partially observable can be converted into MDP

Important property of MDP: future is independent of the past, given the present

State transitions:

$$P_{ss'} = P[S_{t+1} = s' \mid S_t = s]$$

$$P = \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn} \end{bmatrix}$$

State Transition Matrix

(each row sums to 1)

a Markov Process is a memoryless random Process
(a sequence of random states with the markov property)

$\langle S, P \rangle$

→ State transition propability matrix

→ (finite) State space

Episode: a Sample of the Markov Chain

Markov Reward Process

$\langle S, P, R, \gamma \rangle$

γ → discount factor $\in [0, 1]$

R → reward function $R_s = E[R_{t+1} \mid S_t = s]$

Return G_t = total discounted reward at t

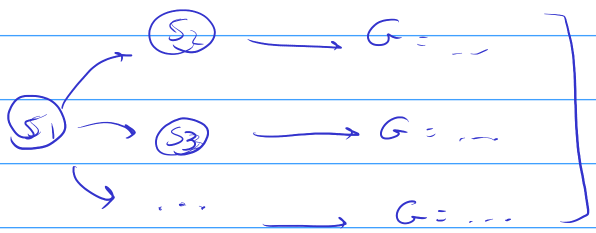
$$G_t = R_{t+1} + \gamma R_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Myopic $0 \leftarrow \gamma \rightarrow 1$ far-sighted

Value function : long-term value of state s

State-Value function : expected return starting from s

$$v(s) = E[G_t \mid S_t = s]$$



$v(S_1) \approx Avg(G_n)$

$$\begin{aligned} v(s) &= E[G_t \mid S_t = s] \\ &= E[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= E[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s] \end{aligned}$$

Bellman Equation for MRPs

$$v(s) = E[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

$$v(s) = R_s + \gamma \sum_{s' \in S} P_{ss'} v(s')$$

$$v = R + \gamma P v$$

$$\begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} = \begin{bmatrix} R_1 \\ \vdots \\ R_n \end{bmatrix} + \gamma \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & & \\ P_{n1} & \cdots & P_{nn} \end{bmatrix} \begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix}$$

Matrix form of Bellman Equation

now, Bellman equation is linear and solvable:

$$v = (I - \gamma P)^{-1} R \qquad O(n^3) \text{ for } n \text{ states}$$

iterative methods
- ↳ Dynamic Programming
- ↳ Monte Carlo evaluation
- ↳ Temporal-Difference learning

Markov Decision Process : MRP with decisions

(still all states are markov)

$\langle S, A, P, R, \gamma \rangle$

- γ → discount factor $\gamma \in [0, 1]$
- R → reward function
- P → State transition Probability matrix
  $P_{ss'}^{a} = P[S_{t+1} = s' \mid S_t = s, A_t = a]$
- A → Reward function
  $R_s^a = E[R_{t+1} \mid S_t = s, A_t = a]$
- S → finite set of states

Policy → distributions over actions, given state

$$\pi(a|s) = P[A_t = a \mid S_t = s]$$

MDP → state fully characterizes the path onwards

→ policies are stationary (time-independent)

$$A_t \sim \pi(\cdot \mid S_t), \forall t > 0$$

$S_1, R_2, S_2, \dots \Rightarrow$ MRP $(\langle S, P^\pi, R, \gamma \rangle)$

<u>recovering MRP</u>
<u>from policy</u>

$$P^\pi_{ss'} = \sum_{a \in A} \pi(a|s) P^a_{ss'}$$

$$R^\pi_s = \sum_{a \in A} \pi(a|s) R^a_s$$

state-value function → $v_\pi(s) = E_\pi[G_t \mid S_t = s]$

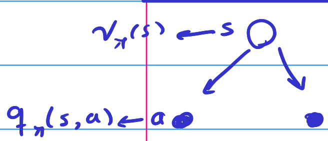
action-value function → $q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a]$

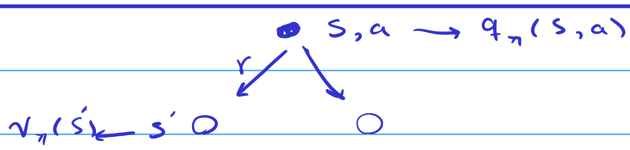
if we're following a specific policy!

⇒ Bellman Expectation Equation

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$$

$$q_\pi(s,a) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$



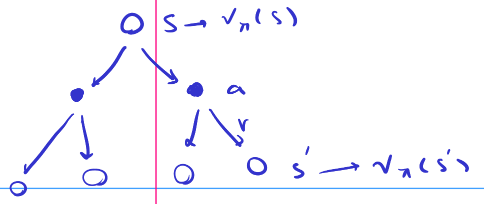


$$v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s,a)$$

calculate state value from action values by averaging over actions

$$q_\pi(s,a) = R^a_s + \gamma \sum_{s' \in S} P^a_{ss'} v_\pi(s')$$

average over dynamics of env



] my actions

] environment dynamics

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s') \right)$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s') q_\pi(s',a')$$

Optimal state-value func

$$v_*(s) = \max_\pi v_\pi(s)$$

Optimal action-value func

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

$\pi \geq \pi'$ if $v_\pi(s) \geq v_{\pi'}(s)$; $\forall s$

* for any MDP there exists an deterministic optimal policy $\pi_* \geq \pi : \forall \pi$

* there can exist multiple optimal policies

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a \in A} q_*(s,a) \\ 0 & \text{otherwise} \end{cases}$$

$$v_*(s) = \max_a q_*(s,a)$$

$$q_*(s,a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \max_{a'} q_*(s',a')$$

$$v_*(s) = \max_a R_s^a + \gamma \sum_{s'} P_{ss'}^a v_*(s')$$

$$q_*(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_*(s')$$

Bellman Optimality Equations are non-linear

↳ value iteration ↳ policy iteration ↳ Q-learning ↳ sarsa