(Discrete time Finite State)

Markov Decision Process (MDP)

(Choice)

(Sequence in time)

Markov Assumption

{For a Stochastic discrete time dynamical System}

$$P(X_{t+1} \mid X_t, U_t, X_{t-1}, U_{t-1} \cdots X_0, U_0) = P(X_{t+1} \mid X_t, U_t)$$

"Given the present state, future [illegible] are independent of past [illegible]"

"Conditioned on the present state future and past [illegible] are independent"

"State is the Complete Summary of the past"

⇒ It assumes that for dynamical System there exist ~~a state~~ quantity (we are calling it State), such that Conditioned on the present state, future and past of the System is independ.

Markov Chain

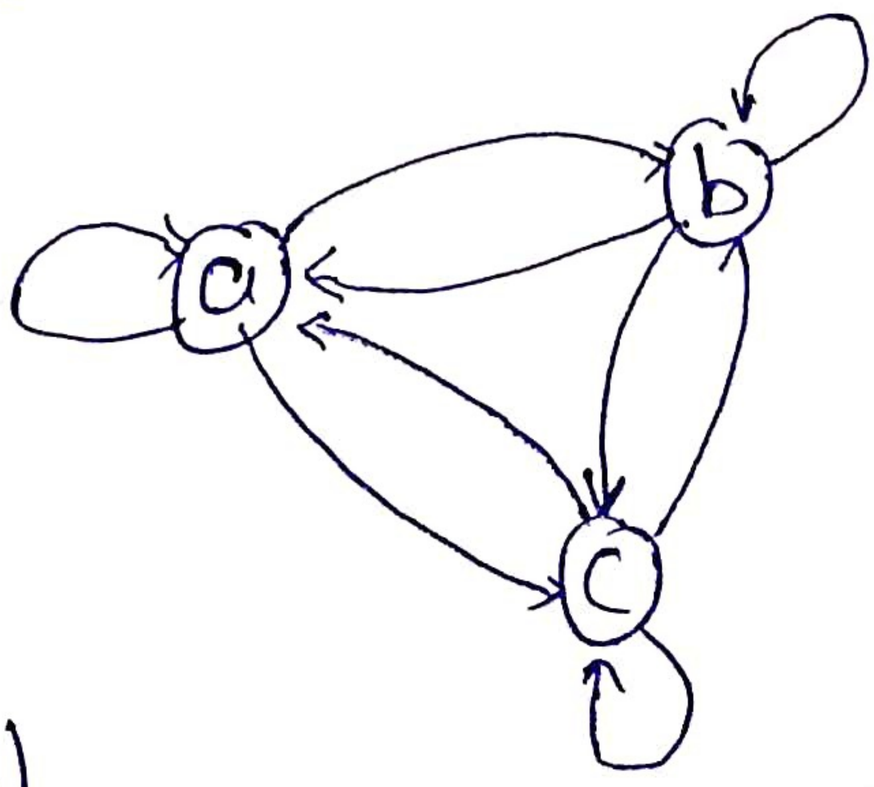
→ $S \in S$ state

→ [illegible]

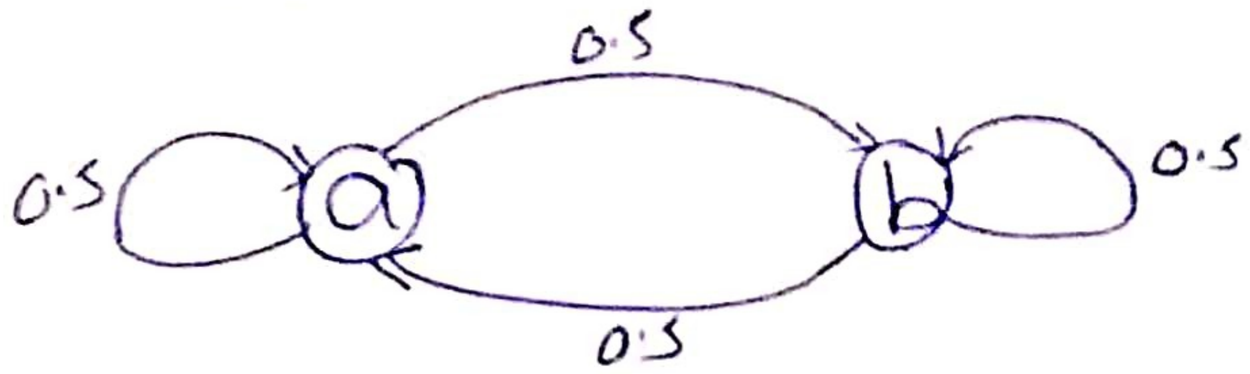
→ $T(S, S') = P(S' \mid S)$

{Probability that the system will transition to state S', given its present state is S in next time step}

→ State state



⇒ Markov chain is evolution of this kind of system over time.



a a b a b a b b a a b a b b

{Sample of Markov Chain}

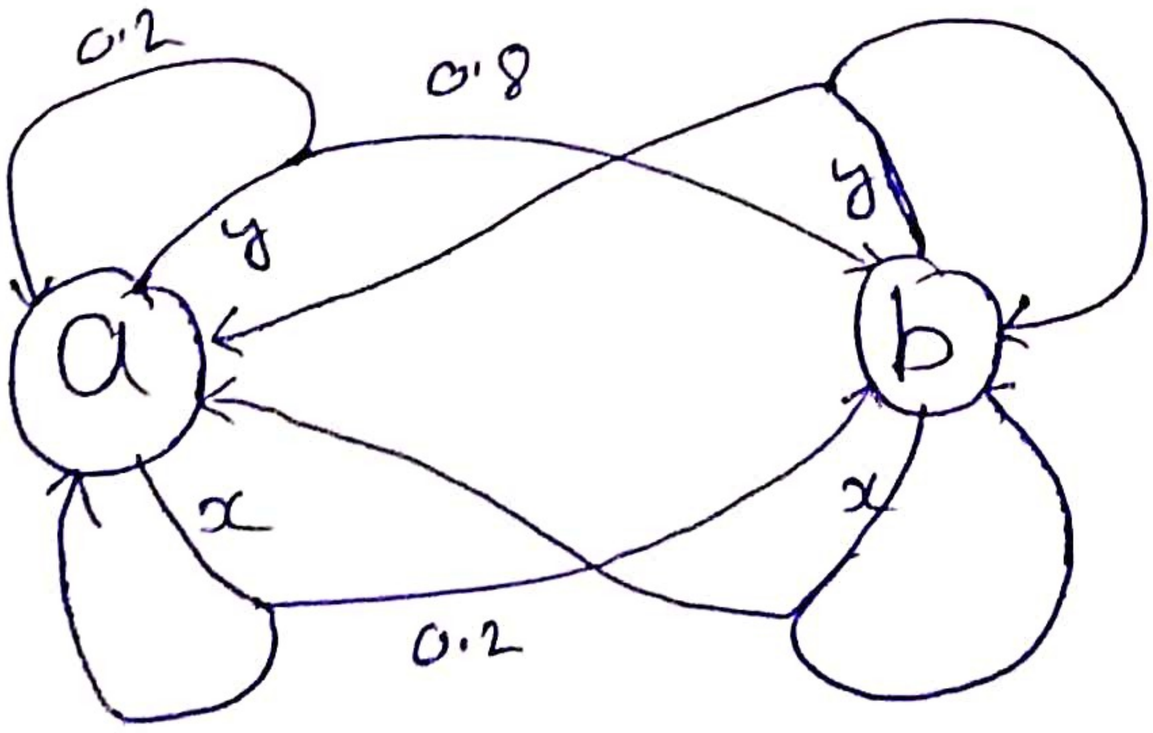
Markov Chain + Decision = Markov Decision Process

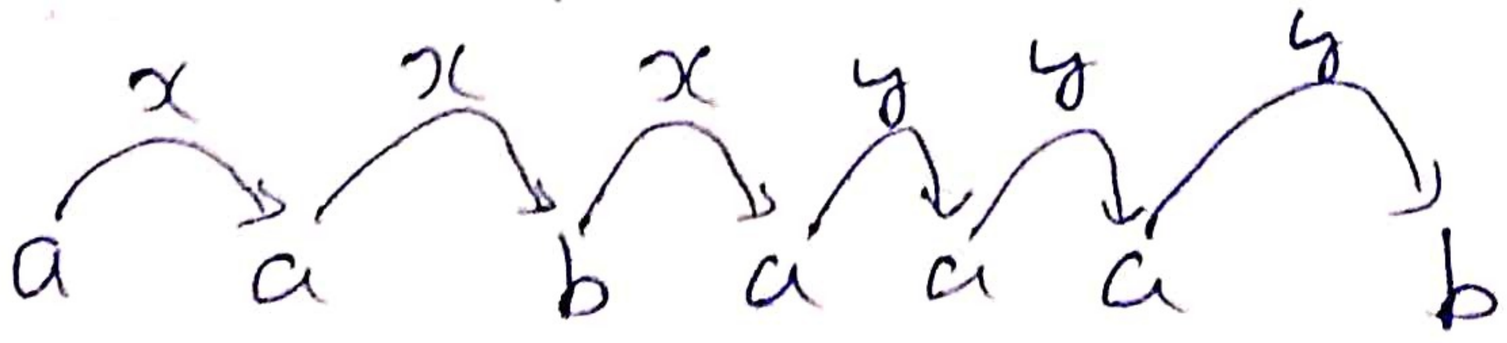
→ System with S States

→ a actions

→ $T(S, a, S') = P(S' | S, a)$

→ Start state





⇒ Evolution of this kind of system is referred to as markov decision process.

⇒ You can also imagin the System can ↑ AI agent ~~[illegible]~~ making the Choices/Decision. in every time step.

⇒ ~~To give a motivation to the agent~~ To do that we add rewards for transition.

$R(s, a, s')$ { Sometime, it only depends of state / on sometime the triple }

⇒ <u>Rational Agent</u> : Select actions that maximize its (expected) utility

{ Sum of rewards }

↳ It is also common to define utility as sum of discounted rewards.

↳ <u>Idea</u> : reward now is better than same reward in future.

$$r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots$$

$\boxed{\gamma \leq 1}$

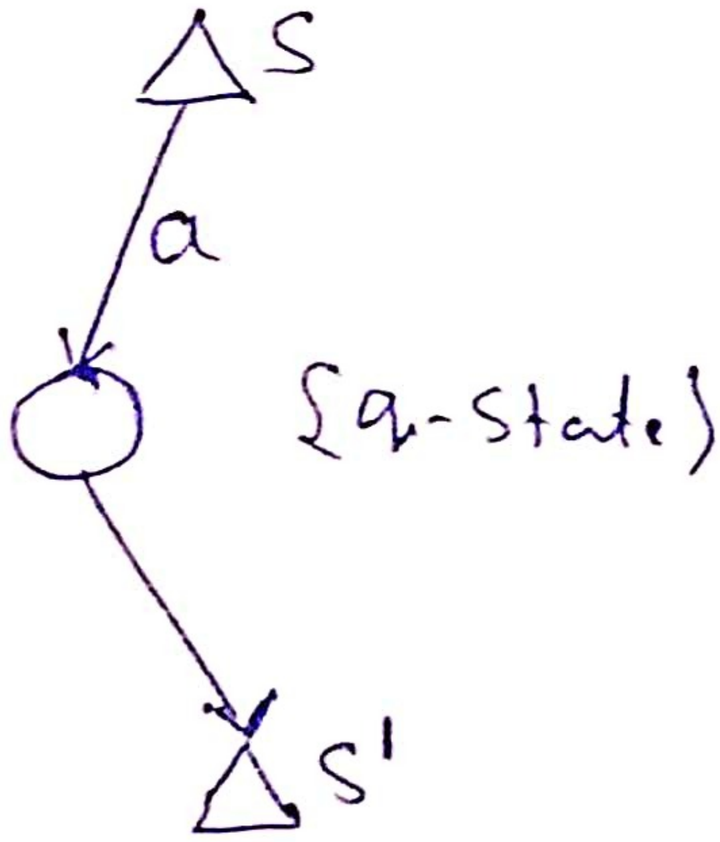
Policy

$\pi : S \rightarrow A$

(optimal policy) π^* ⟶ Policy that maximizes the expected utility.

Plan V Policy

Sequence of Action



Value of State

$V^*(s)$ ⟶ Expected utility starting at state s and acting optimally.

{Similar we can define $Q^*(s,a)$ as the value of q-state.}

$$V^*(s) = \max_a Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$

$$\boxed{V^*(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]}$$

{Bellman Equation}

Time Limited Value

$V_k(s)$ → Optimal value of state S if the game ends in k Steps.

$$\boxed{V_{k+1}(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]}$$

★ Value Iteration {An Algorithm for Solving MDP}

1. $V_0(s) = 0 \quad \forall s \in S$

2. Use Iterative bellman Equation to compute $V_1, V_2 \cdots V_m$ for all states.

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

3. $\pi_{k+1}(s) \leftarrow \operatorname{argmax}_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$

→ Policy at $k+1^{st}$ iteration.

⇒ Problem with value iteration is that, it is very slow!

- → Complexity (as^2)
- → actions (Policy) Converge much before value.

★ Policy Iteration

Let $V^{\pi}(s)$ ⇒ Expected utility starting at state s and following policy π.

1. Start with some initial policy π_0

2. Find value of all the states given the policy by iteratively using the given below equation:

$$V_{K+1}^{\pi_i}(s) \leftarrow \sum_{s'} T(s, \pi_i(s), s') \left[R(s, \pi_i(s), s') + \gamma V_{K+1}^{\pi_i}(s') \right]$$

3. Update the policy using the following.

$$\pi_{i+1} = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V^{\pi_i}(s') \right]$$

4. Repeat until policy Converge