

Markov Decision Process

Generalization of Search problem, where you are not entirely sure what your actions are going to do, until you try them.

(Non-Deterministic Search)

* Noisy Movement: Actions do not always go as planned.

* The agent receives rewards each time step.

* Goal: Maximize sum of rewards

* Markov Decision Process

⇒ An MDP is defined by:

- A set of states $s \in S$
- A set of actions $a \in A$
- A transition function $T(s, a, s')$

↳ Probability that a from s leads to s'
i.e. $P(s' | s, a)$

↳ Also called the model or the dynamics

- A reward function $R(s, a, s')$
- A start state
- Maybe a terminal state.

* What is Markov about MDPs?

⇒ For Markov decision process, "Markov" means action outcomes depends only on the current state:

$$P(S_{t+1} | S_t, A_t, S_{t-1}, A_{t-1}, \dots, S_0) = P(S_{t+1} | S_t, A_t)$$

* Policies

⇒ In deterministic single-agent search problems we wanted an optimal **plan**.

↳ Sequence of action from start to goal

⇒ For MDPs, we want an optimal **policy** $\pi^*: S \rightarrow A$

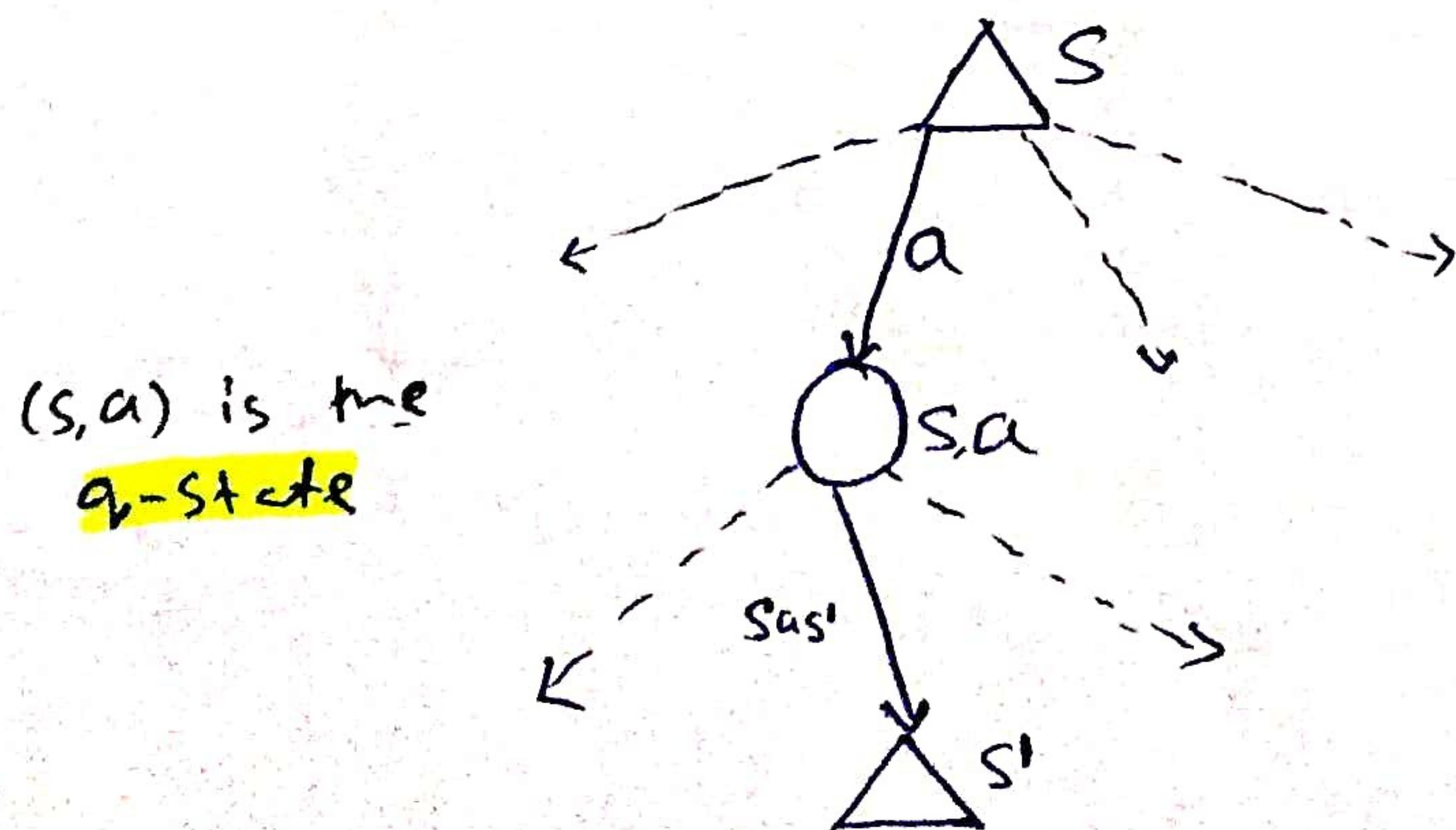
An optimal policy is the one that maximizes expected utility if followed

A policy π gives action for each state

⇒ A explicit policy defines a reflex agent.

* MDP Search Tree

⇒ Each MDP state projects an expectimax-like search tree.



(S, a, S') called a transition
 $T(S, a, S') = P(S' | S, a)$
 $R(S, a, S')$

* Discounting

- ⇒ It's reasonable to maximize the sum of rewards
- ⇒ It is also reasonable to prefer rewards now to rewards later.
- ⇒ One Solution: Values of rewards decay exponentially

$$1 \rightarrow \gamma \rightarrow \gamma^2$$

* Stationary Preferences

- Theorem: If we assume stationary preferences:

$$[a_1, a_2, \dots] \succ [b_1, b_2, \dots]$$

$$[\gamma a_1, a_1, a_2, \dots] \succ [\gamma b_1, b_1, b_2, \dots]$$

- ⇒ There are only two ways of defining utilities:

- Additive utility: $U([r_0, r_1, r_2, \dots]) = r_0 + r_1 + r_2 + \dots$

- Discounted utility: $U([r_0, r_1, r_2, \dots]) = r_0 + \gamma r_1 + \gamma^2 r_2 + \dots$

* Infinite Utilities

Problem: What if the game lasts forever? Do we get infinite rewards?

Solution

① Finite horizon (Similar to depth-limited search)

- Terminate episode after a fixed T step.
- Gives non-stationary policies (π depends of time left)

② Discounting: Use $0 < \gamma < 1$

$$\rightarrow U([s_0, s_1, \dots, s_\infty]) = \sum_{t=0}^{\infty} \gamma^t r_t \ll \frac{R_{\max}}{1-\gamma}$$

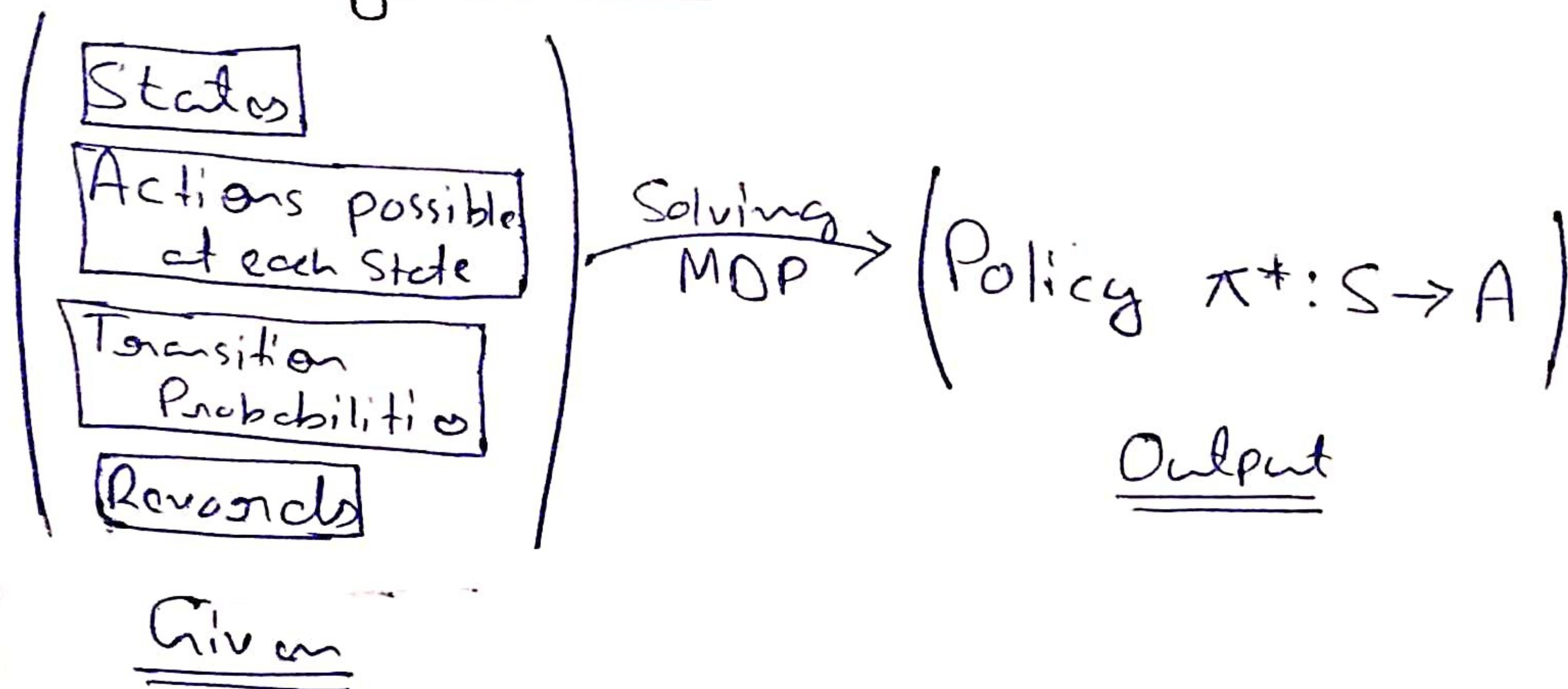
\rightarrow Smaller γ means Smaller "horizon"
(Shorter term focus)

③ Absorbing State

\rightarrow Guarantee that for every policy, a terminal state will eventually be reached.

Markov Decision Process

* Solving MDPs



* Optimal Quantities

\Rightarrow The value (Utility) of a state S :

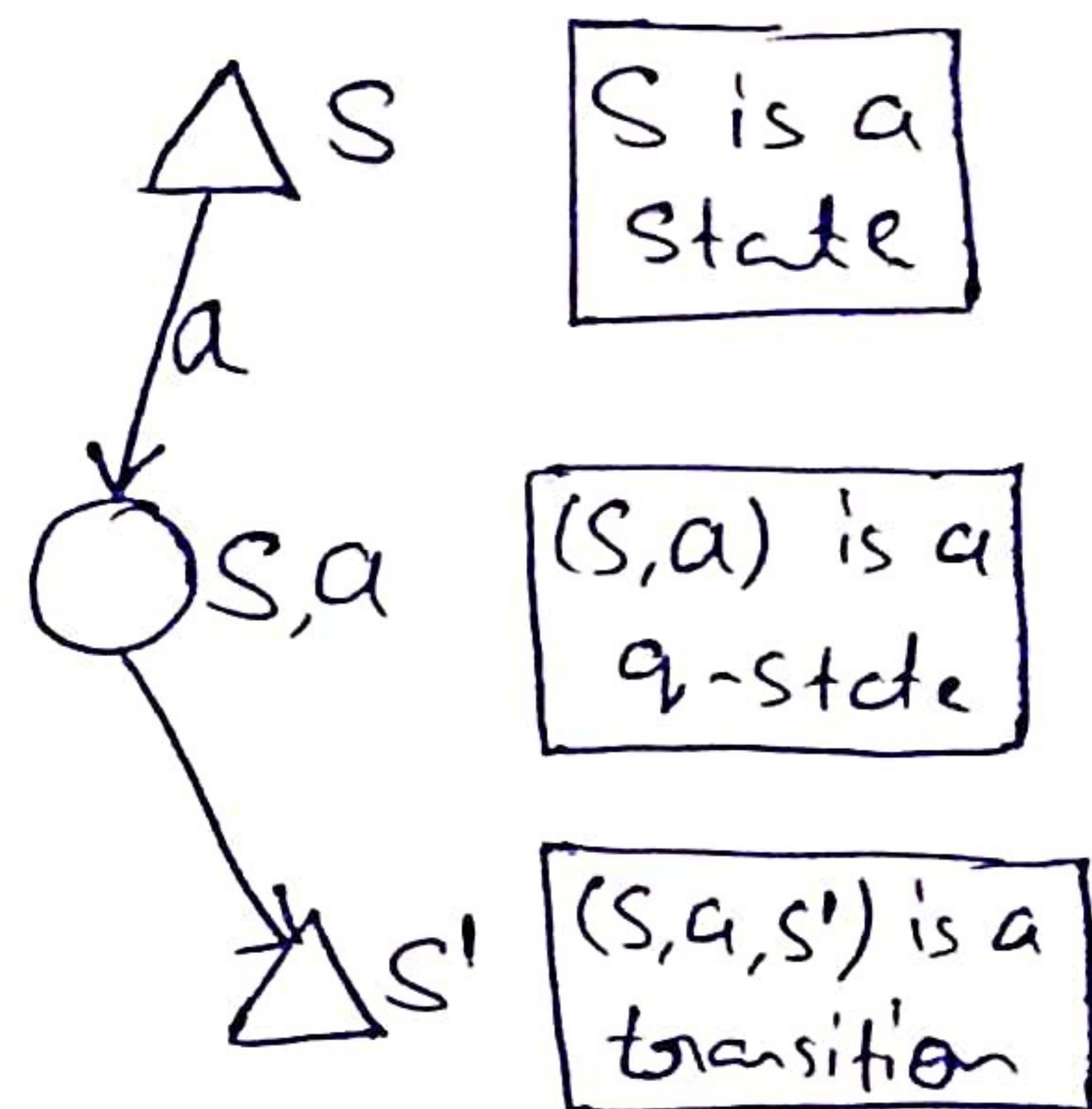
$V^*(s) =$ Expected utility starting in s and acting optimally.

\Rightarrow The value (Utility) of a q-state (s, a) :

$Q^*(s, a) =$ Expected utility starting out having taken action a from state s & thereafter acting optimally.

\Rightarrow The Optimal policy:

$\pi^*(s) =$ optimal action from state s .



* Values of State

Fundamental operation: Compute the (expectimax) value of a state.

↳ Average sum of (discounted) rewards.

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

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

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

* Time-Limited Values

⇒ Define $V_k(s)$ to be the optimal value of s if the game ends in k more time steps.

* Value Iteration

■ Start with $V_0(s) = 0$: no time steps left means an expected reward sum of zero.

■ Given vector of $V_k(s)$ values, do one ply of expectation for each state

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

■ Repeat until convergence.

■ Complexity of each iteration: $O(s^2 A)$

