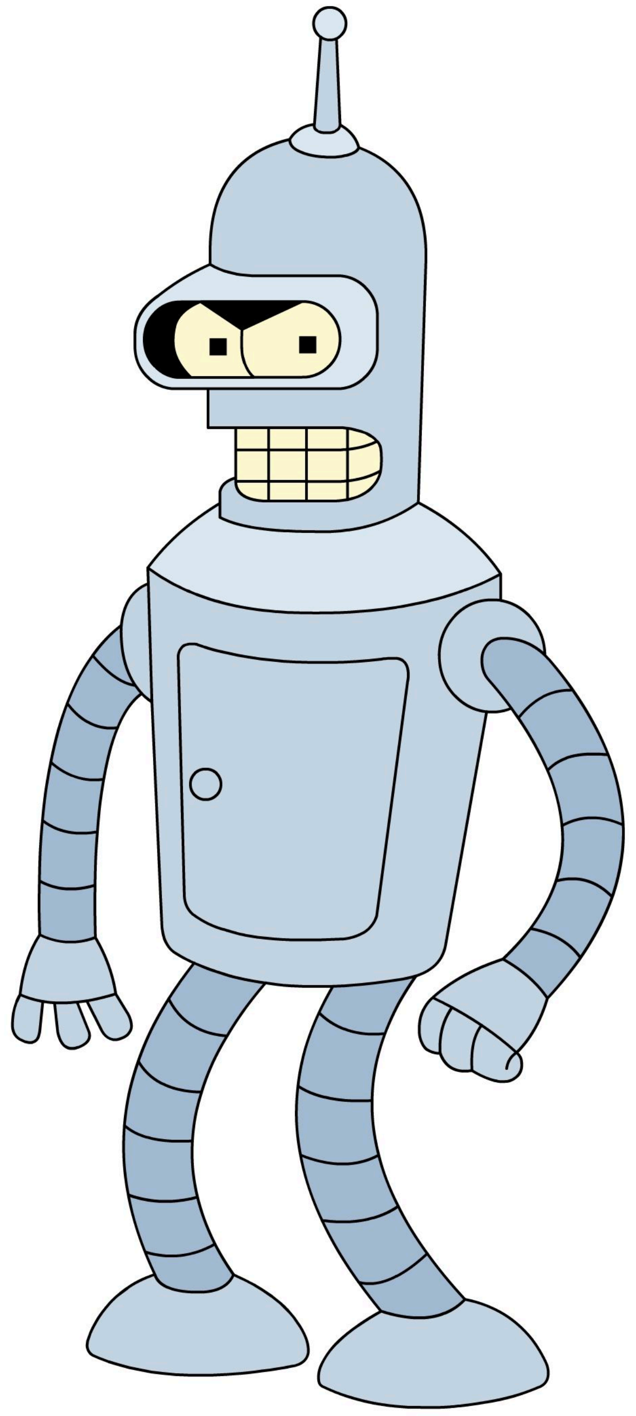
Reinforcement Learning

HSE, autumn - winter 2022

Lecture 2: Model-free RL



Sergei Laktionov
slaktionov@hse.ru
LinkedIn

MDP

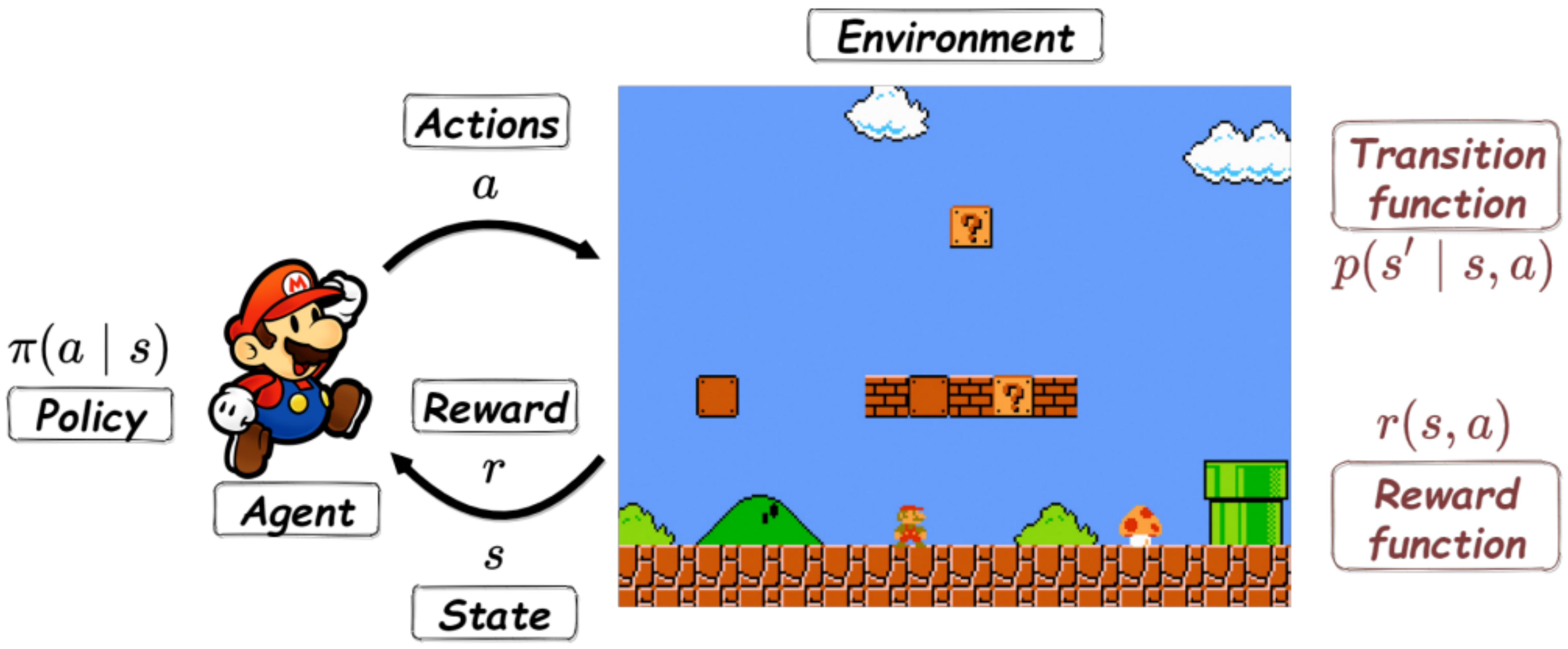
MDP is a 4-tuple $(\mathcal{S}, \mathcal{A}, p, r)$:

1. $\mathcal{A}$ is an action space

2. $\mathcal{S}$ is a state space

3. $p(s' \,|\, s, a) = \mathbb{P}(S_{t+1} = s' \,|\, S_t = s, A_t = a)$
   is a state-transition function

4. $r(s, a) \in \mathbb{R}$ is a reward function



Source

Recap: Objective

Let T is a final time step. If $T < \infty$ then environment is called *episodic*.

Cumulative reward is called a return or reward-to-go. Note that in general it is a random variable.

$$\boxed{G_t} = \boxed{R_t} + \boxed{\gamma} R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=t}^{\boxed{T}} \gamma^{k-t} R_s$$

Episode length

Immediate reward

Discount factor

$$\mathbb{E}_\pi[G_0] \to \max_\pi$$

Recap: Value Functions

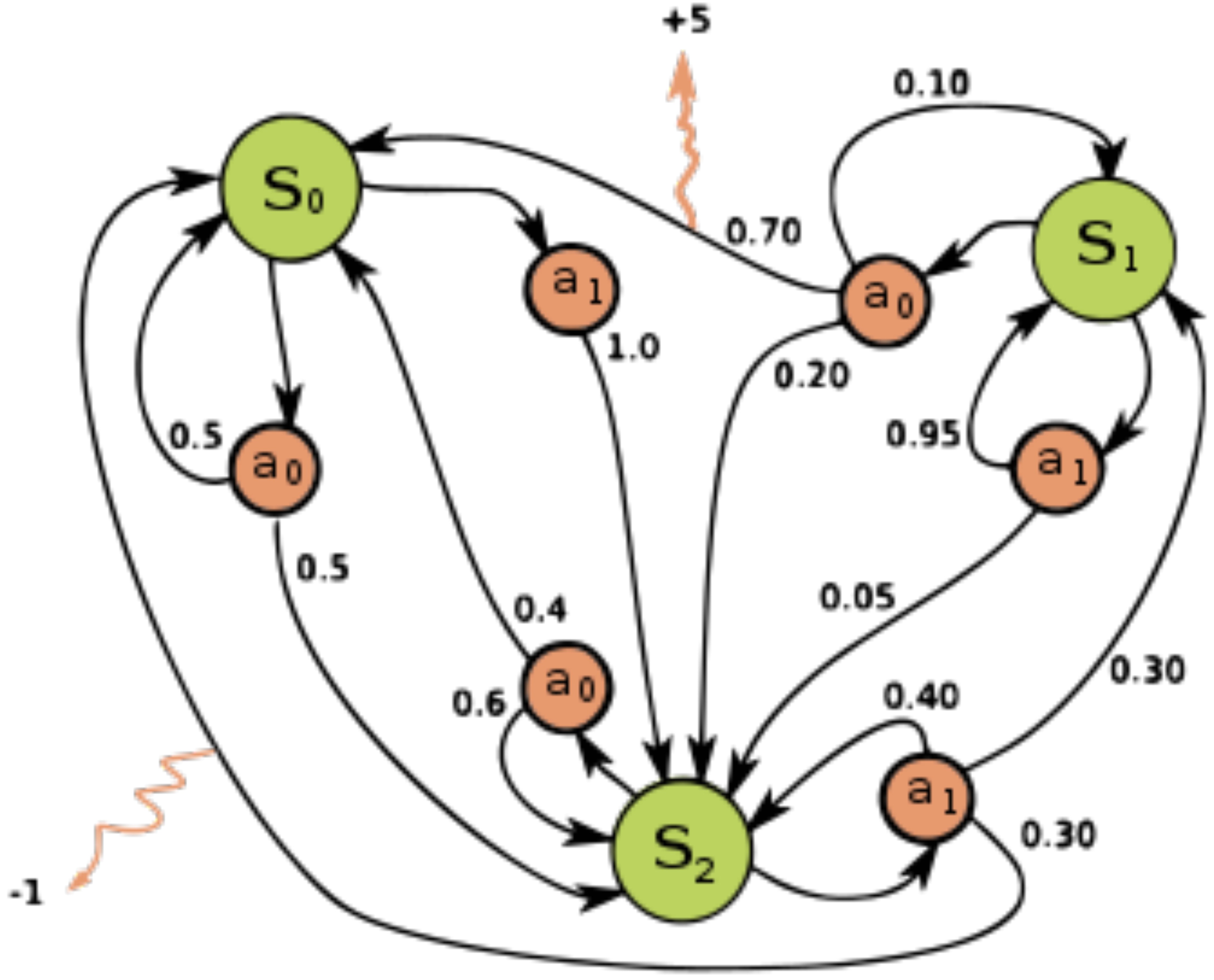
$$G_t = R_t + \gamma G_{t+1}$$

$$V^\pi(s) = \mathbb{E}_\pi[G_t \,|\, S_t = s]$$

$$Q^\pi(s, a) = \mathbb{E}_\pi[G_t \,|\, S_t = s, A_t = a]$$

Recap: Assumptions

1. $p(s' \mid s, a)$ is known
2. State space is finite
3. Action space is finite



Recap: Bellman Equations

Bellman expectation equations:

$$V^{\pi}(s) = \sum_{a} \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \big[r + \gamma V_{\pi}(s') \big]$$

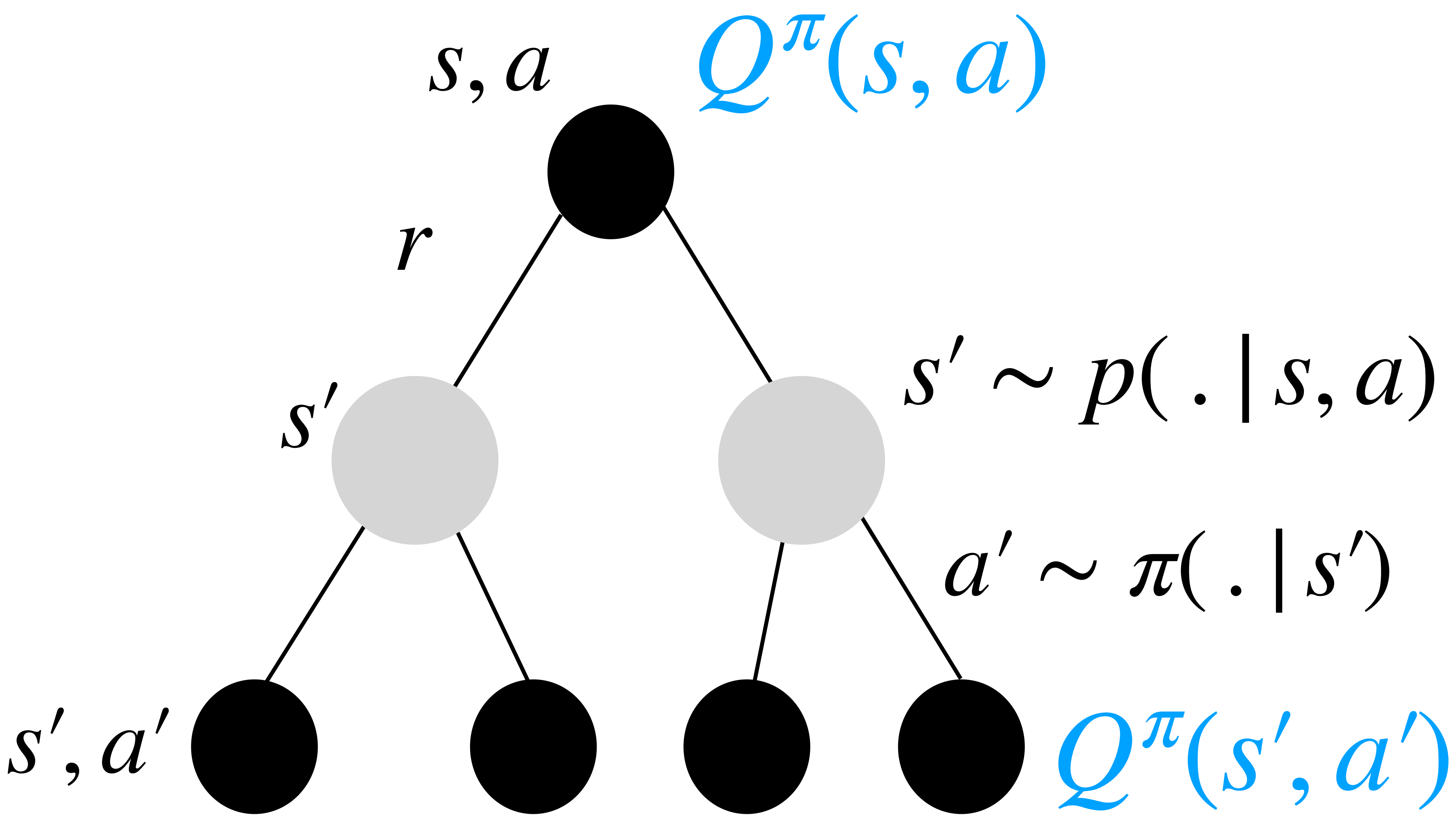
$$Q^{\pi}(s, a) = \sum_{s'} p(s' \mid s, a) \Big[r + \gamma \sum_{a'} \pi(a' \mid s') Q_{\pi}(s', a') \Big]$$

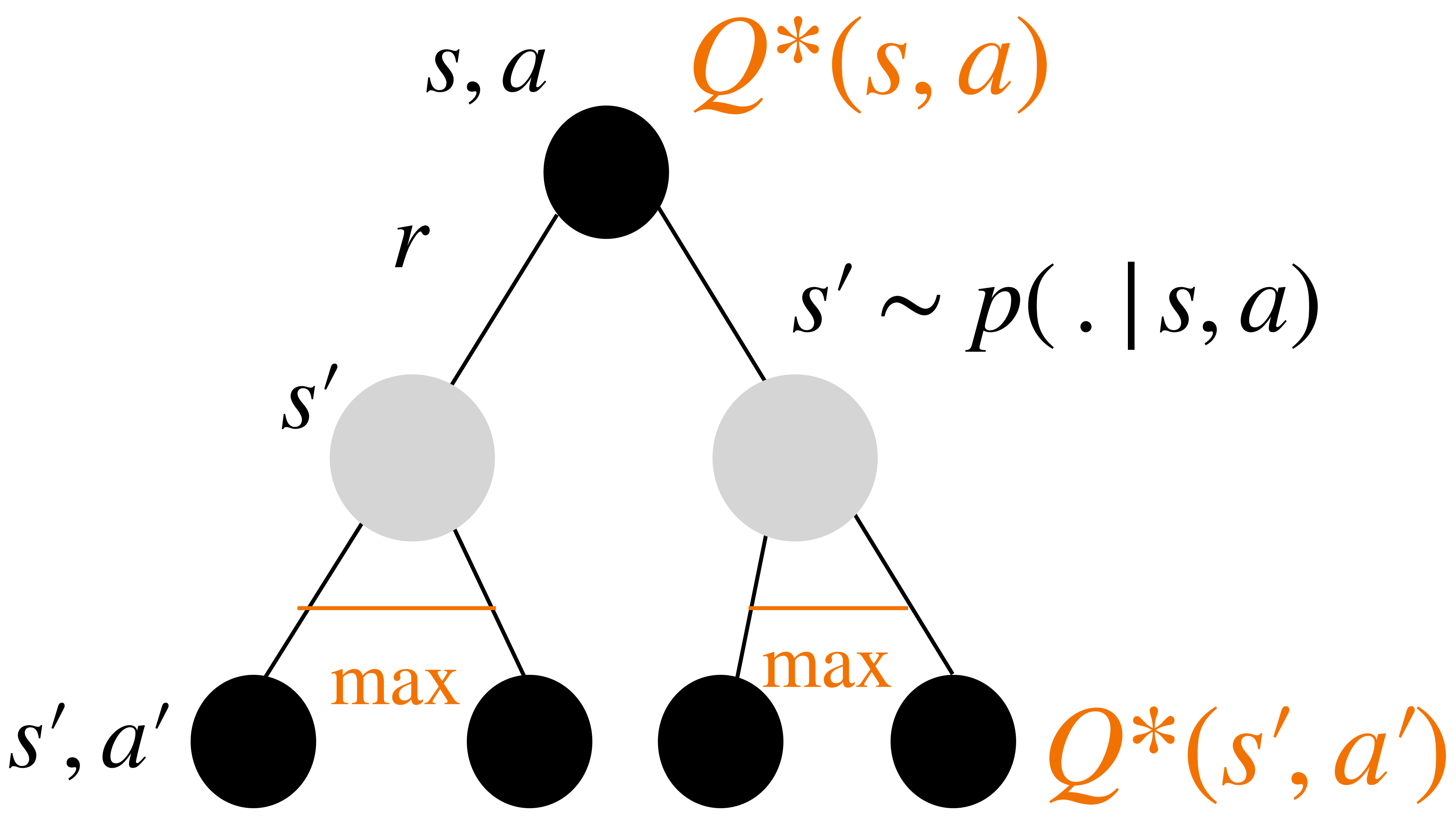
Bellman optimality equations:

$$V^{*}(s) = V^{\pi^{*}}(s) = \max_{a} \sum_{s'} p(s' \mid s, a) [r + \gamma V^{*}(s')]$$

$$Q^{*}(s, a) = Q^{\pi^{*}}(s, a) = \sum_{s'} p(s' \mid s, a) \big[r + \gamma \max_{a'} Q^{*}(s', a') \big]$$

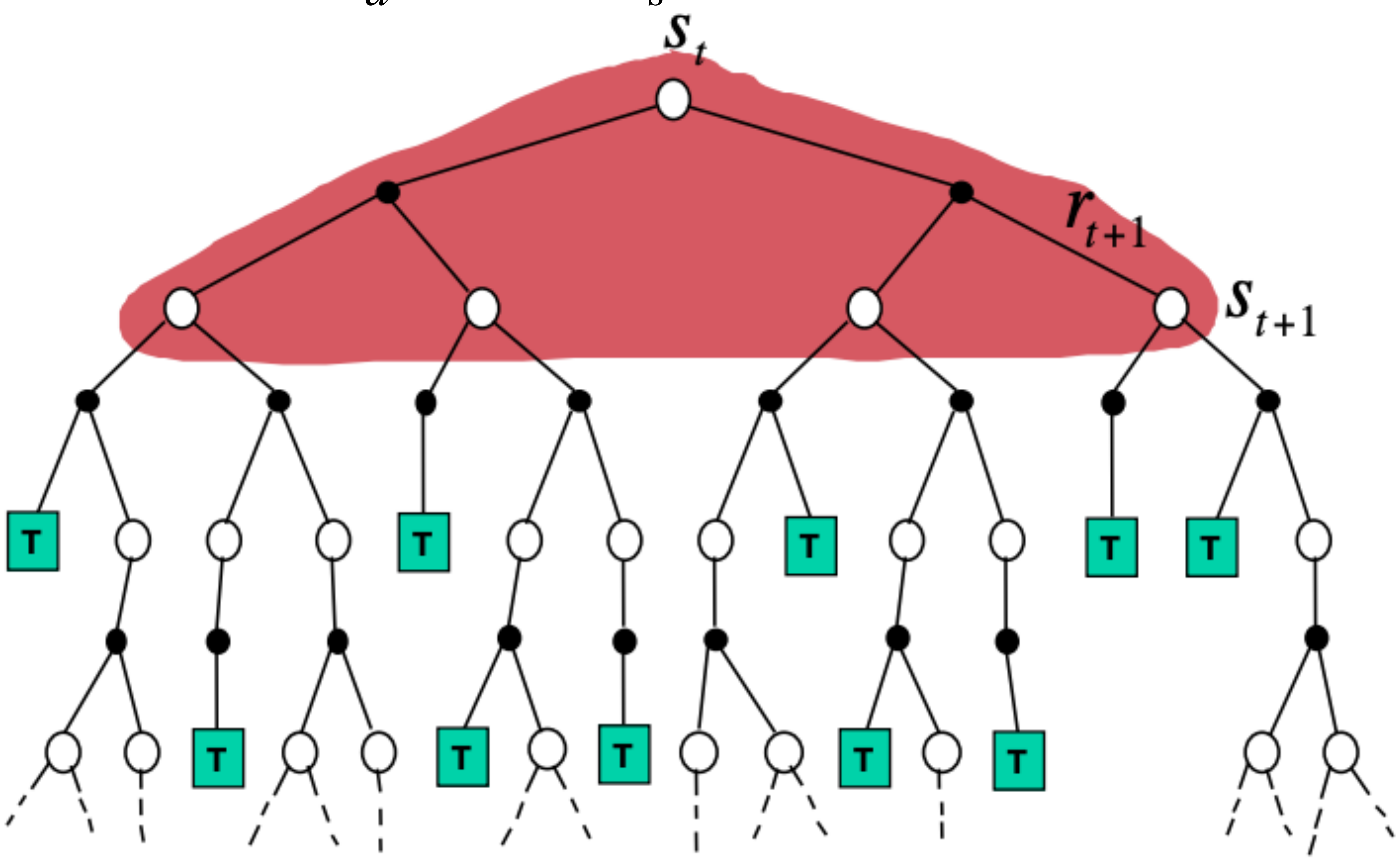
Recap: Bellman Equations





Recap: Policy Evaluation

$$V_{k+1}(s) = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a)\left[r + \gamma V_k(s')\right]$$



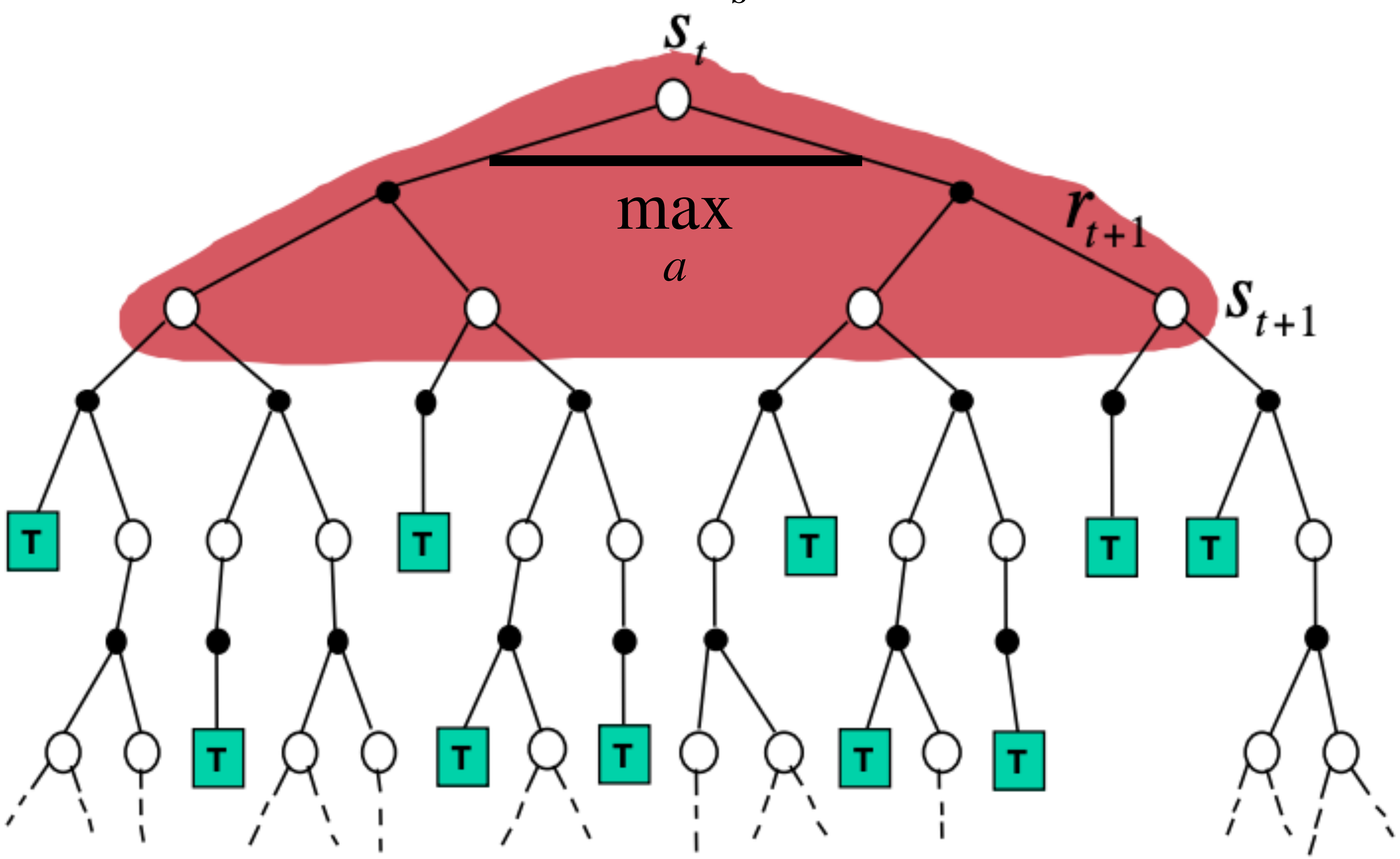
Source

Recap: Policy Improvement

1. If Q_k is known: $\pi(s) = argmax_a Q_k(s, a)$
2. If V_k is known: $\pi(s) = argmax_a \sum_{s'} p(s' | s, a)[r + \gamma V_k(s')]$

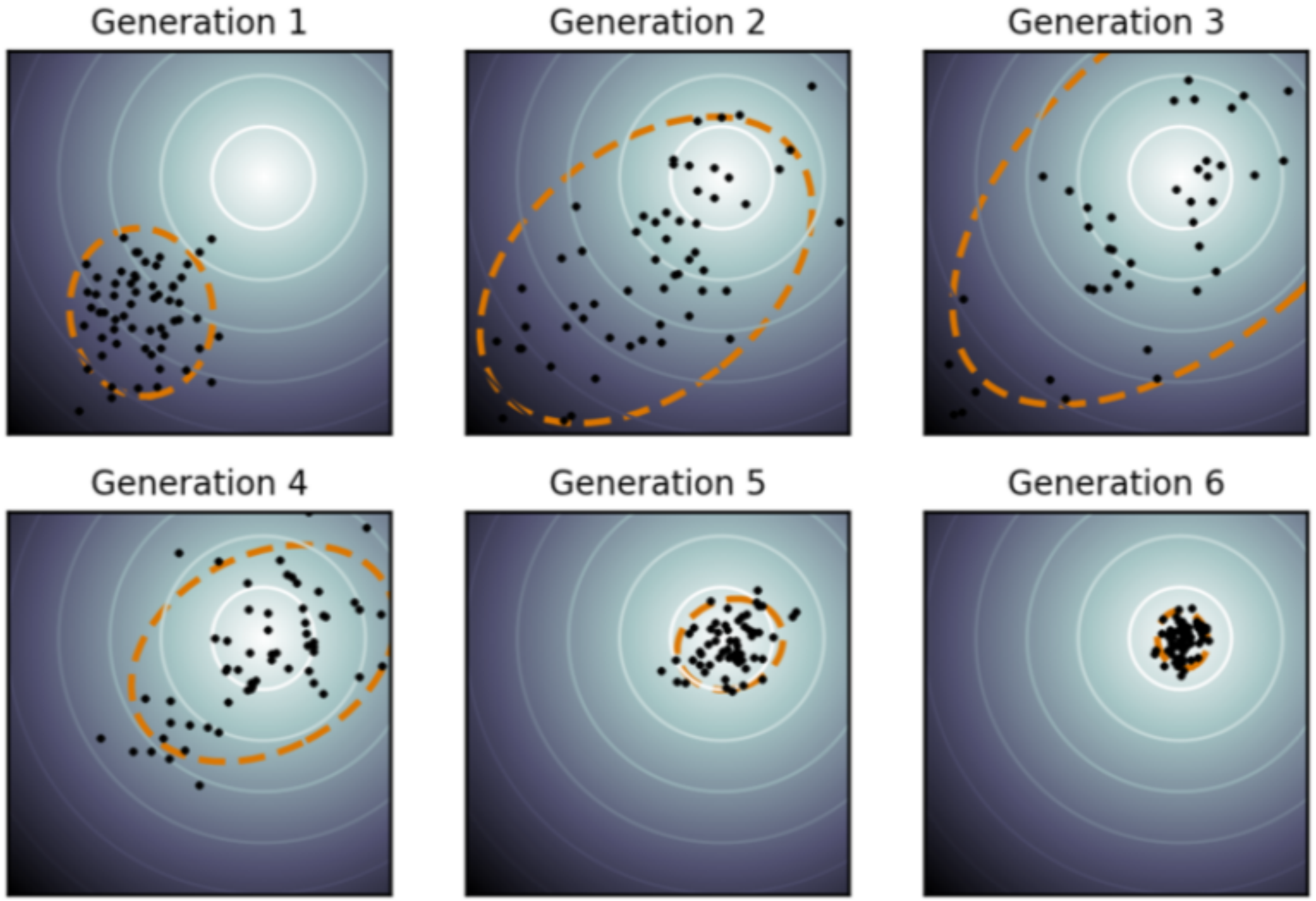
Recap: Value Iteration

$$V_{k+1}(s) = \max_a \left[r + \gamma \sum_{s'} p(s' | s, a) V_k(s') \right]$$



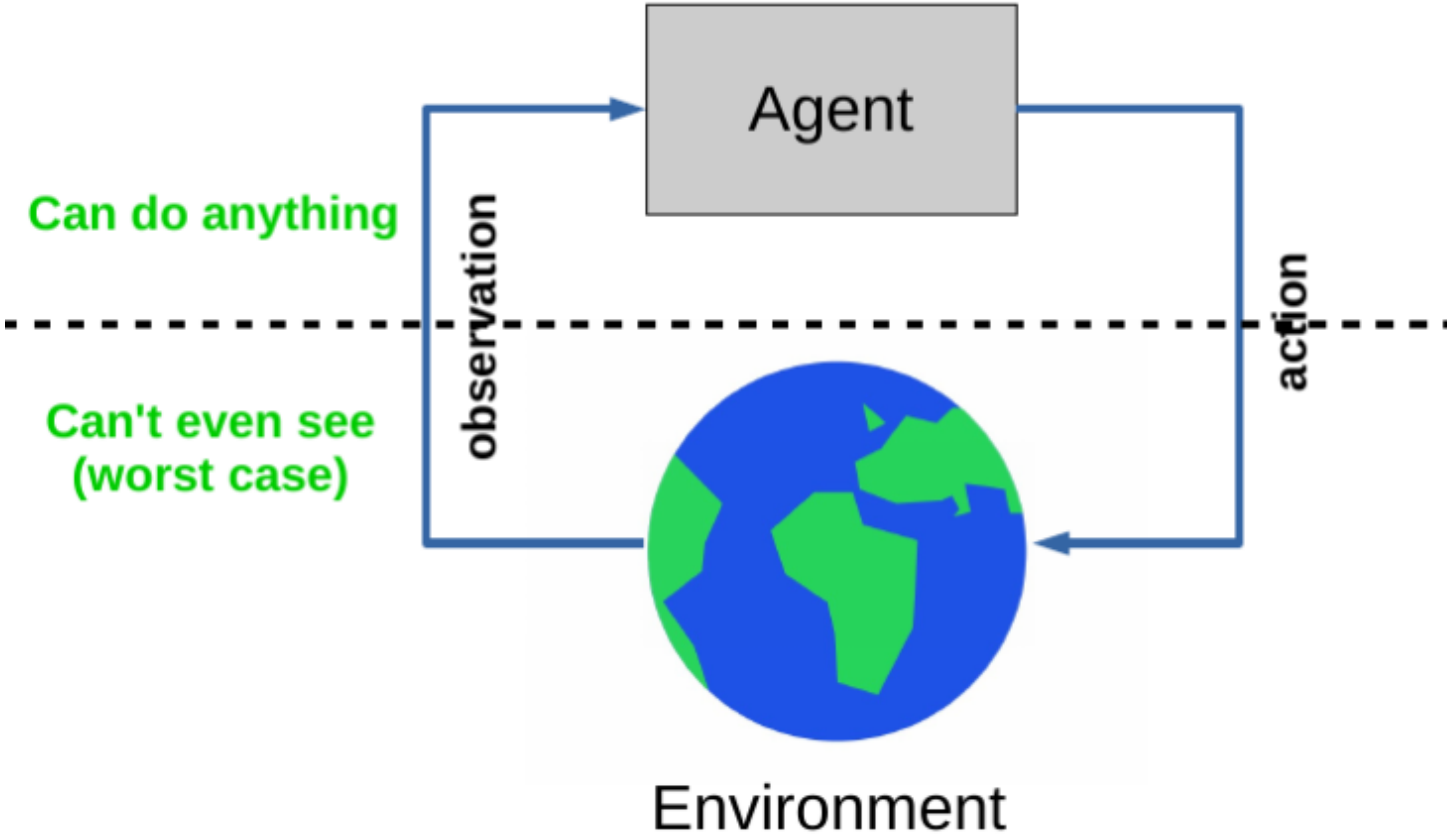
Source

Recap: Metaheuristic Optimisation



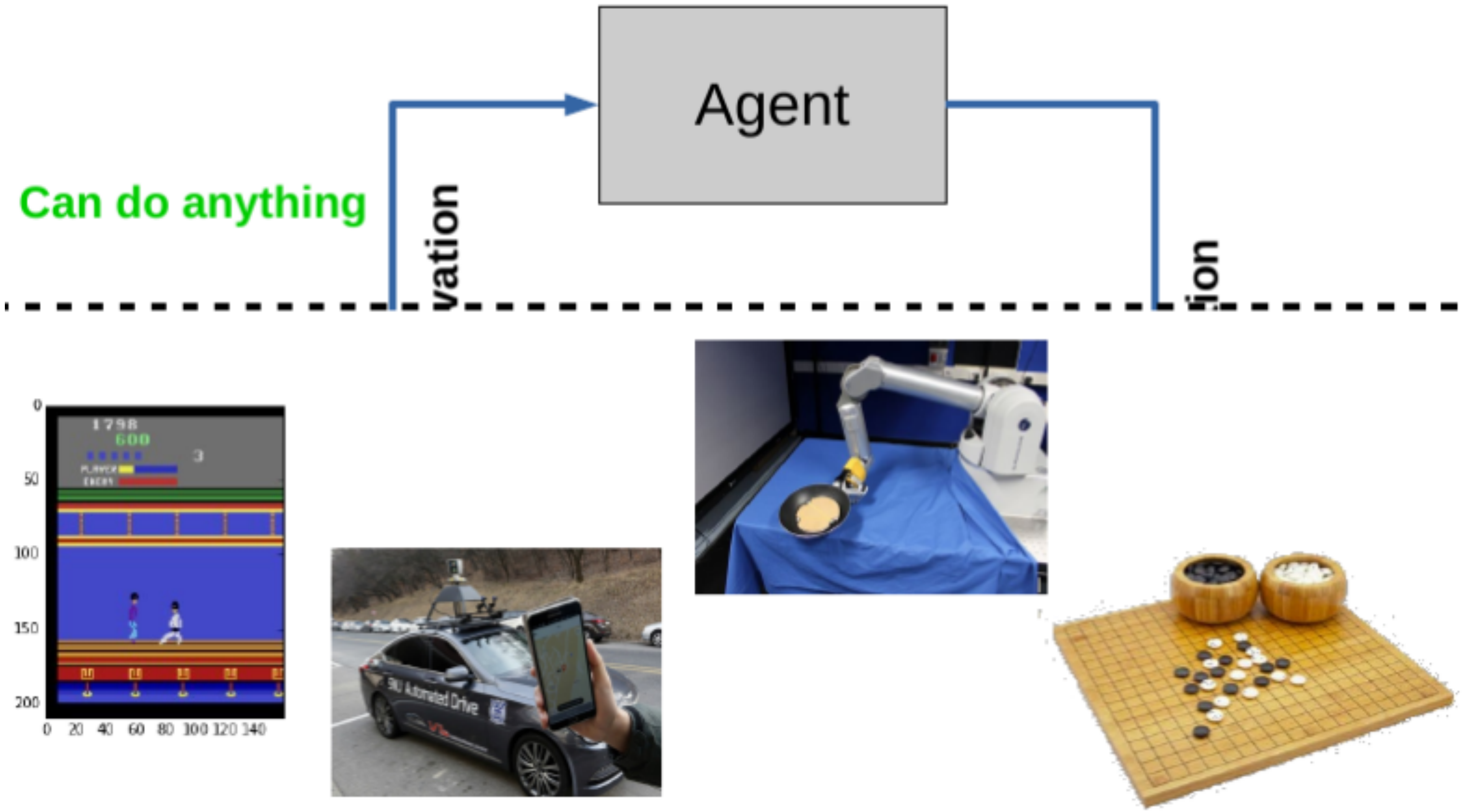
https://lilianweng.github.io/posts/2019-09-05-evolution-strategies/

Decision Processes



Source

Decision Processes



Source

Policy Improvement

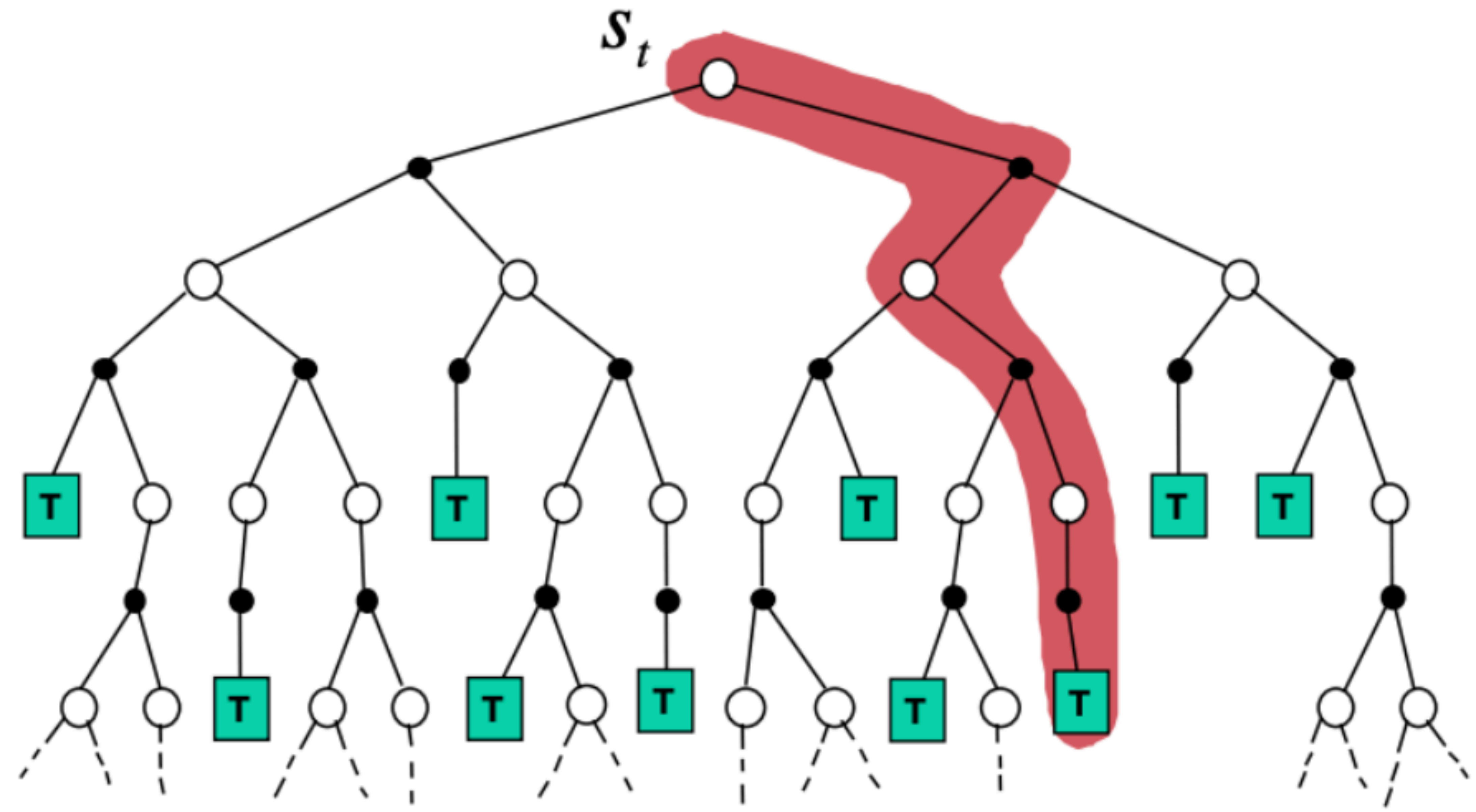
1. If Q_k is known: $\pi(s) = argmax_a Q_k(s, a)$
2. If V_k is known: $\pi(s) = argmax_a \sum_{s'} \cancel{p(s' | s, a)}[r + \gamma V_k(s')]$

No more available

Monte-Carlo Policy Evaluation

$$\tau_k = \{s_{k0} = s, a_{k1} = a, s_{k1}, a_{k1}, \ldots\}, G(\tau_k) = \sum_{t=0}^{T} \gamma^t r_{ko}$$

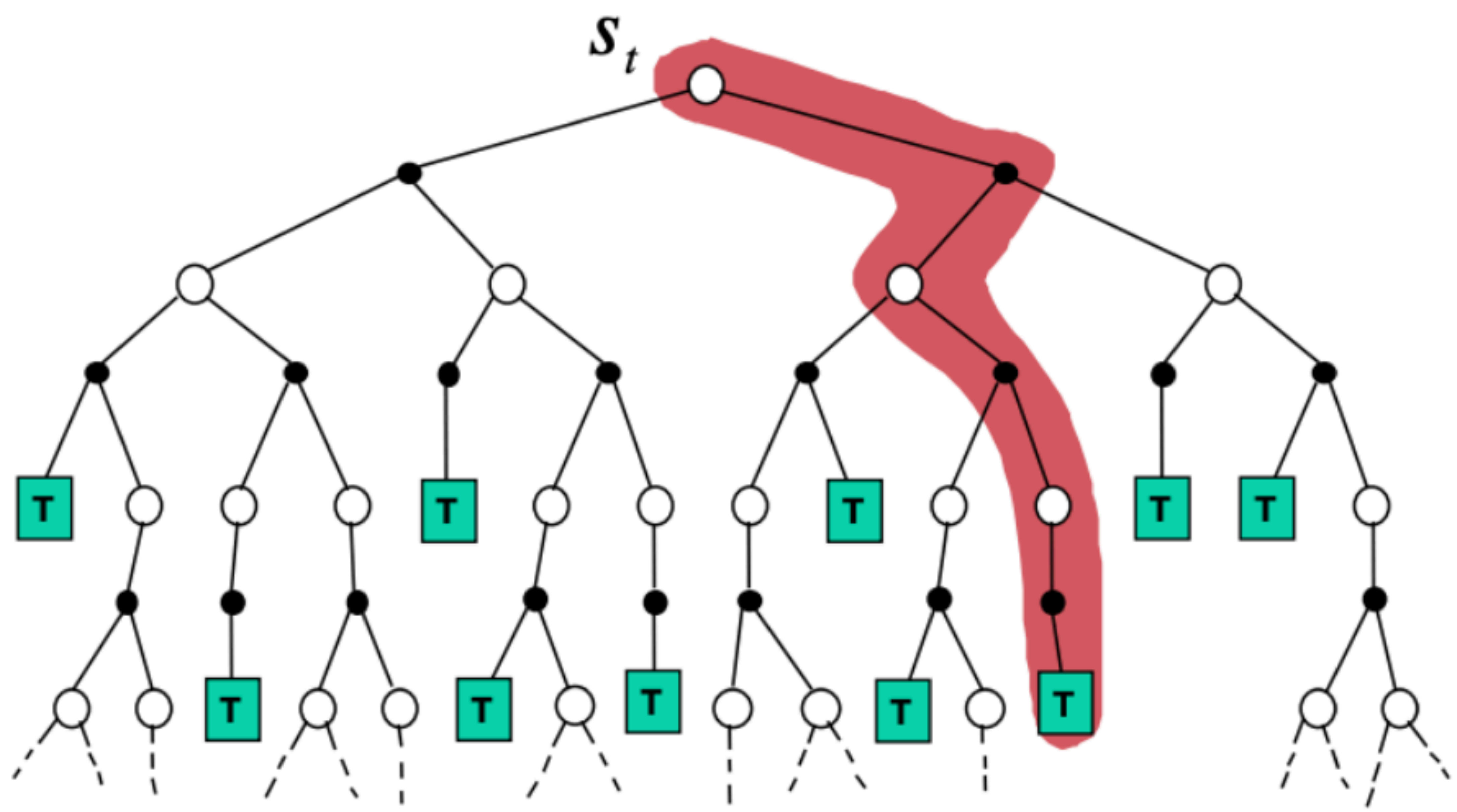
$$Q(s, a) \approx \frac{1}{N} \sum_{i=k}^{N} G(\tau_k) = \frac{N-1}{N} \sum_{k=1}^{N-1} G(\tau_k) + \frac{1}{N} G(\tau_N)$$



Monte-Carlo Policy Evaluation

$$\tau_k = \{s_{k0} = s, a_{k1} = a, s_{k1}, a_{k1}, \dots\}, G(\tau_k) = \sum_{t=0}^{T} \gamma^t r_{ko}$$

$$Q(s, a) \approx \frac{1}{N} \sum_{i=k}^{N} G(\tau_k) = \frac{N-1}{N} \sum_{k=1}^{N-1} G(\tau_k) + \frac{1}{N} G(\tau_N)$$



Pros:

- Unbiased
- Convergence's guarantees

Cons:

- High variance
- Must complete the episodes
- Accrue no information about not visited (s, a)

Stochastic Approximation

https://en.wikipedia.org/wiki/Stochastic_approximation

- Would like to solve: $x = \mathbb{E}_{\epsilon \sim p(\epsilon)}[F(x, \epsilon)]$
- Replace it with the following iterative procedure:

$$x_k = (1 - \alpha_k)x_{k-1} + \alpha_k F(x_{k-1}, \epsilon_{k-1}), \epsilon_{k-1} \sim p(\epsilon)$$

Stochastic Approximation

https://en.wikipedia.org/wiki/Stochastic_approximation

- Would like to solve: $x = \mathbb{E}_{\epsilon \sim p(\epsilon)}[F(x, \epsilon)]$

- Replace it with the following iterative procedure:

$$x_k = (1 - \alpha_k)x_{k-1} + \alpha_k F(x_{k-1}, \epsilon_{k-1}),\ \epsilon_{k-1} \sim p(\epsilon)$$

$$x_k = x_{k-1} + \alpha_k (F(x_{k-1}, \epsilon_{k-1}) - x_{k-1}),\ \epsilon_{k-1} \sim p(\epsilon)$$

Robbins–Monro theorem:

- $\sum_{k=1}^{+\infty} \alpha_k = +\infty,\ \sum_{k=1}^{+\infty} \alpha_k^2 < +\infty$
- Some technical conditions

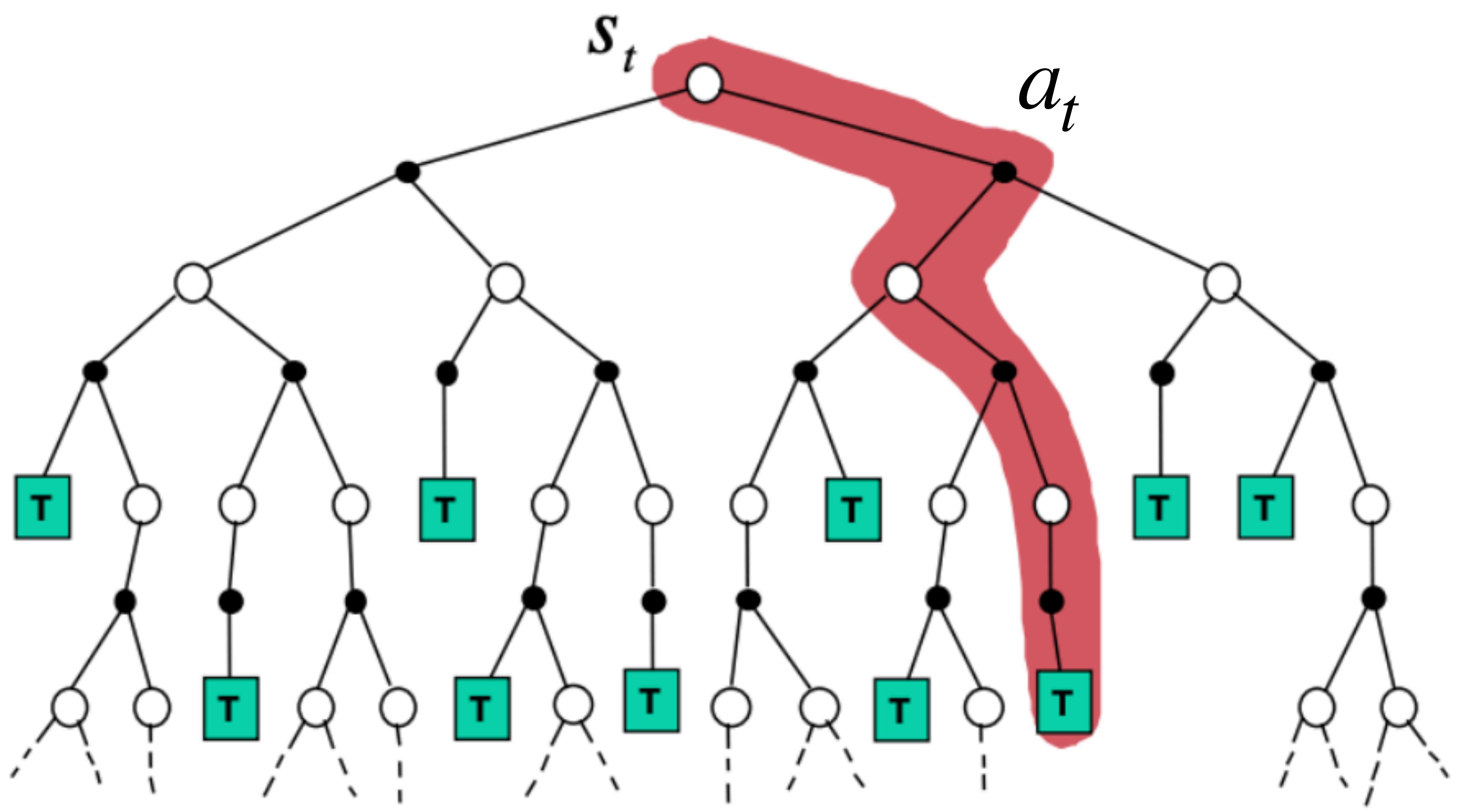
$\longrightarrow$

$$x_k \to^{\mathbb{P}} x^*$$

$$x^* = \mathbb{E}_{\epsilon \sim p(.)}[F(x^*, \epsilon)]$$

Monte-Carlo Policy Evaluation

$$Q_{k+1}(s,a) = Q_k(s,a) + \alpha_k(s,a)(G_k - Q_k(s,a))$$



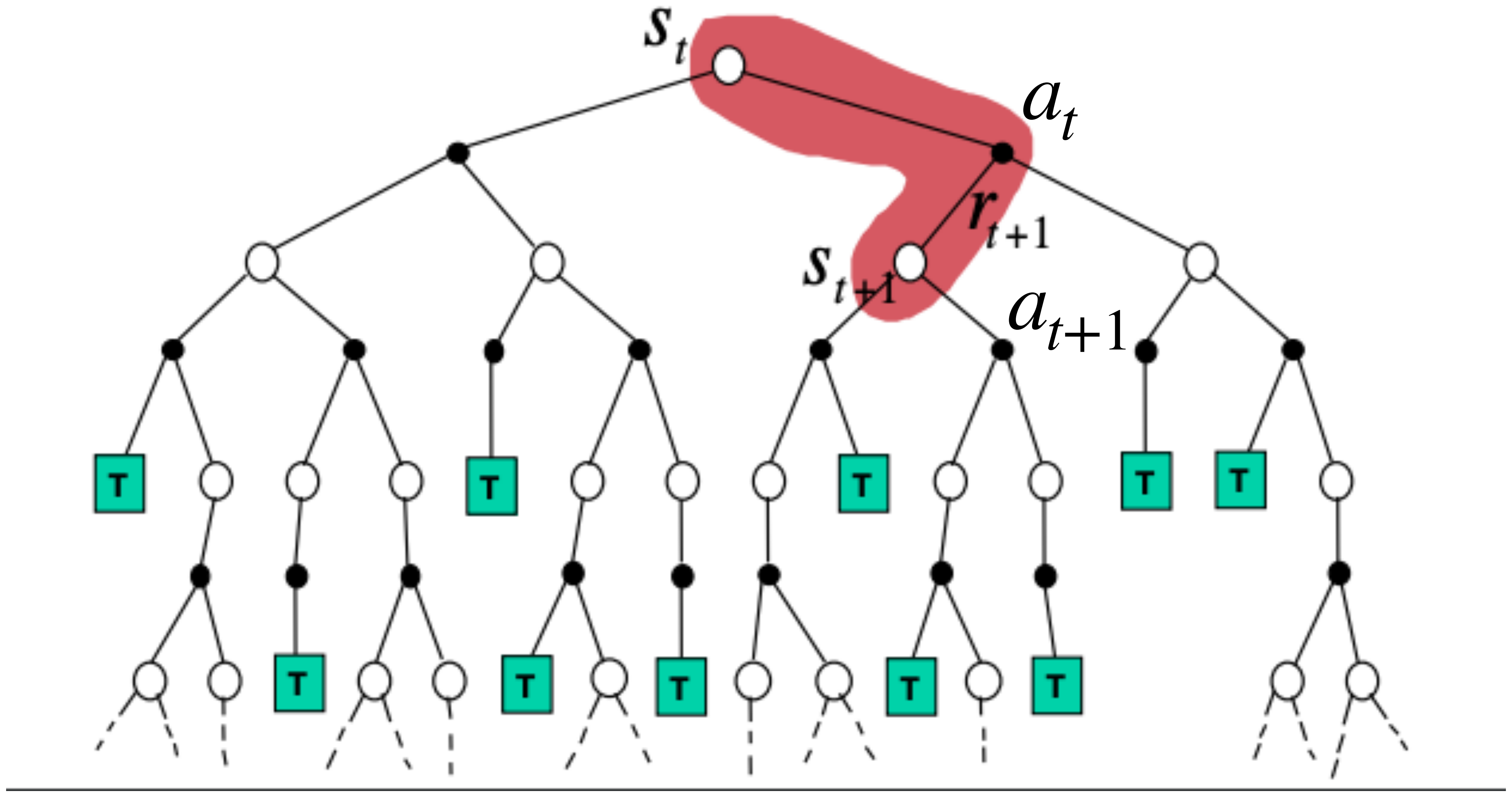
Temporal Difference Learning

Target

$$Q_{k+1}(s,a) = Q_k(s,a) + \alpha_k(s,a)\left(\boxed{r + \gamma Q_k(s',a')} - Q_k(s,a)\right)$$

Temporal difference

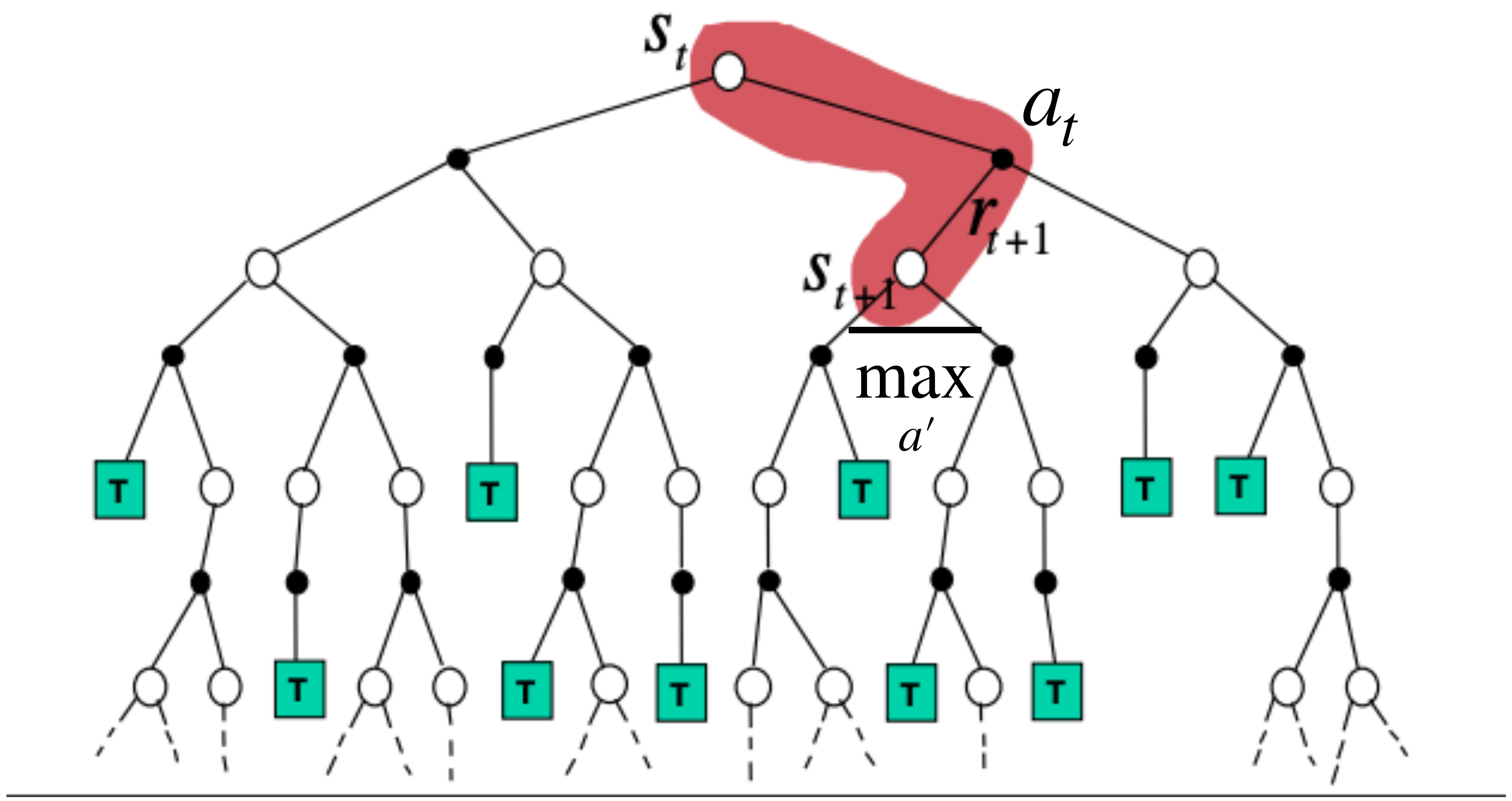
$$s' \sim p(\,.\,|\,s,a),\, a' \sim \pi(\,.\,|\,s)$$



Temporal Difference Learning

$$Q_{k+1}(s,a) \leftarrow Q_k(s,a) + \alpha_k(s,a)(r + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a))$$

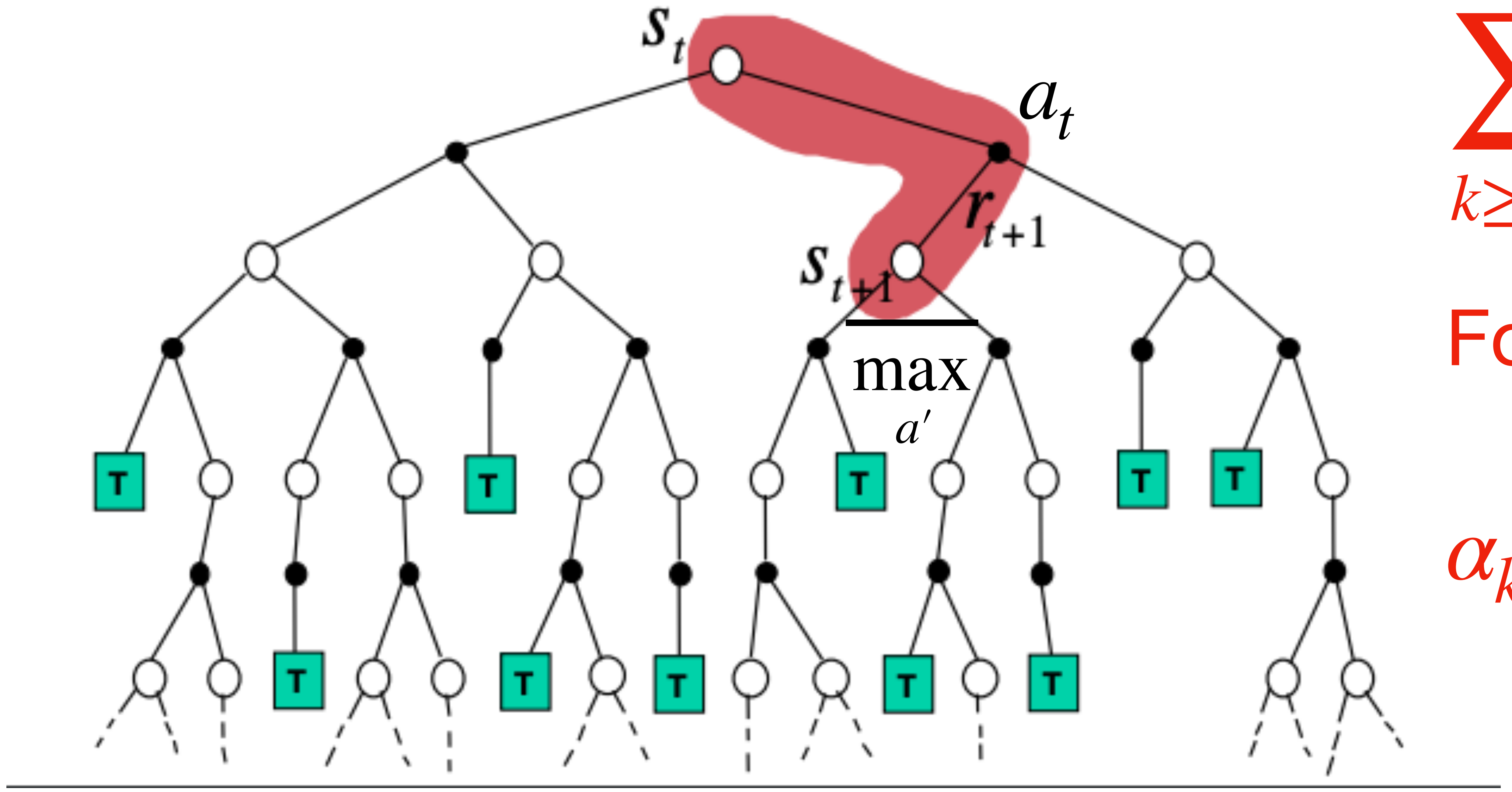
$$s' \sim p(\,.\,|\,s,a)$$



Temporal Difference Learning

$$Q_{k+1}(s,a) \leftarrow Q_k(s,a) + \alpha_k(s,a)(r + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a))$$

$$s' \sim p(\,.\,|\,s,a)$$



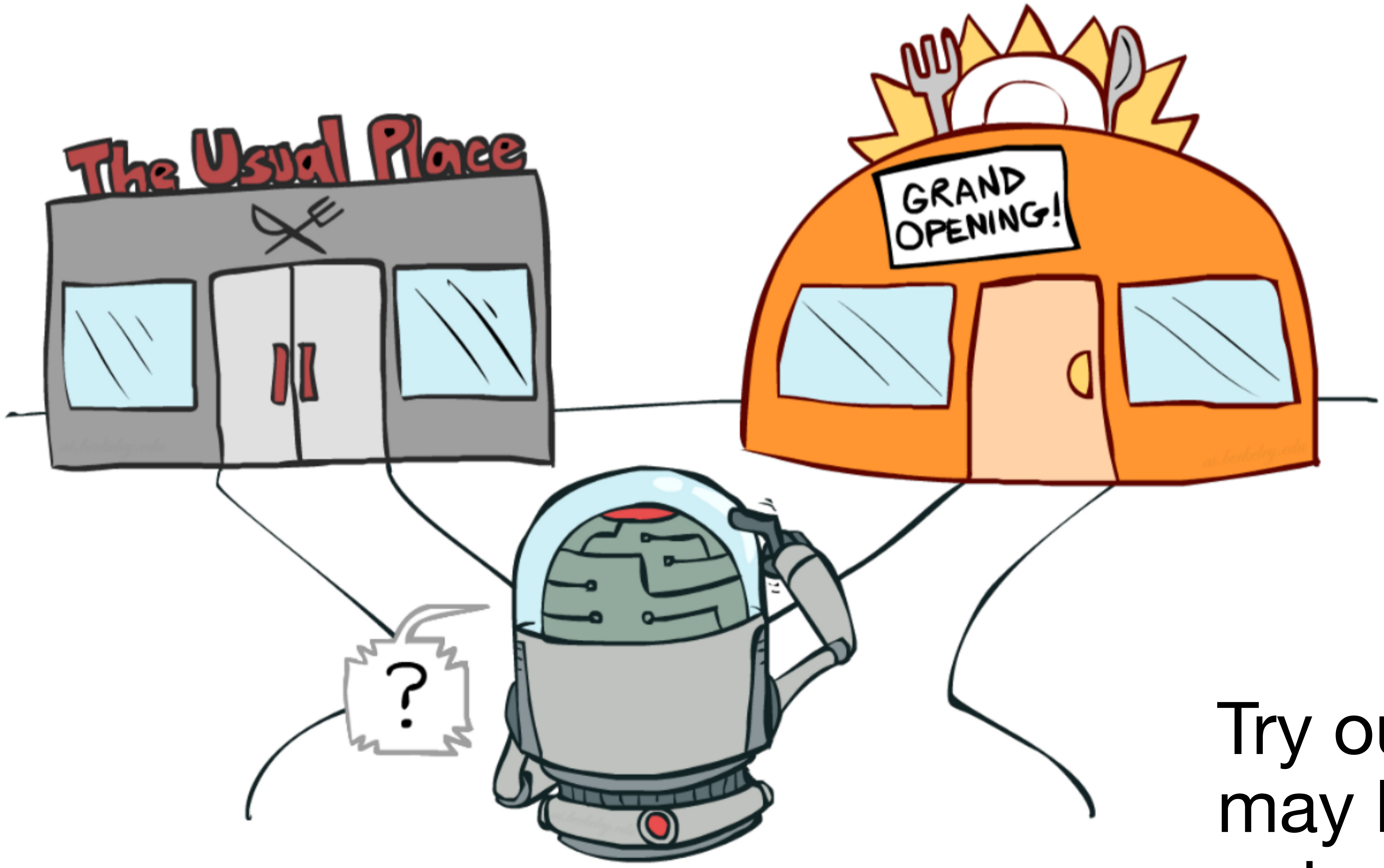
Infinite visitation:

$$\sum_{k\geq 1} \alpha_k(s,a) = +\infty$$

For instance:

$$\alpha_k(s,a) = \frac{1}{n(s,a)}$$

Exploration-Exploitation Trade-off



Choose the best option based on current knowledge (which may be incomplete)

Try out new options that may lead to better outcomes in the future at the expense of an exploitation opportunity

Exploration vs Exploitation

Stochastic policy $\mu(a \mid s)$ s.t.

$\forall s, a : \mu(a \mid s) > 0$:

Deterministic policy $\pi(s)$:

- Greedy policy:
  $\pi(s) = argmax_a Q(s, a)$

Exploration vs Exploitation

Stochastic policy $\mu(a \mid s)$ s.t.

$\forall s, a : \mu(a \mid s) > 0$:

- ε-greedy policy:

$$\mu = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q(s, a) \text{ with probabily } 1 - \varepsilon \end{cases}$$

Deterministic policy $\pi(s)$:

- Greedy policy:
  $\pi(s) = argmax_a Q(s, a)$

Exploration vs Exploitation

Stochastic policy $\mu(a \mid s)$ s.t.

$\forall s, a : \mu(a \mid s) > 0$:

- ε-greedy policy:

$$\mu = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q(s, a) \text{ with probabily } 1 - \varepsilon \end{cases}$$

- Boltzmann policy:

$$\mu(a \mid s) = \text{softmax}(\frac{Q(s, a)}{\alpha})$$

Deterministic policy $\pi(s)$:

- Greedy policy:
  $\pi(s) = argmax_a Q(s, a)$

Q-Learning

- Parameters: ε, α
- $\forall s, a \; Q_0(s,a) = 0$
- For $k = 0,1,\ldots$

1. $a \sim \mu(\,.\,|\,s) = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q_k(s,a) \text{ with probabily } 1-\varepsilon \end{cases}$
2. Observe r and $s' \sim p(\,.\,|\,s,a)$
3. $Q_{k+1}(s,a) \leftarrow Q_k(s,a) + \alpha(r + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a))$

Q-Learning learns Q^* from samples from another policy!

Q-Learning

- Parameters: ε, α
- $\forall s, a \; Q_0(s, a) = 0$
- For $k = 0,1,\ldots$

1. $a \sim \mu(\,.\,|\,s) = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q_k(s, a) \text{ with probabily } 1 - \varepsilon \end{cases}$
2. Observe r and $s' \sim p(\,.\,|\,s, a)$
3. $Q_{k+1}(s, a) \leftarrow Q_k(s, a) + \alpha(r + \gamma \max_{a'} Q_k(s', a') - Q_k(s, a))$

SARSA

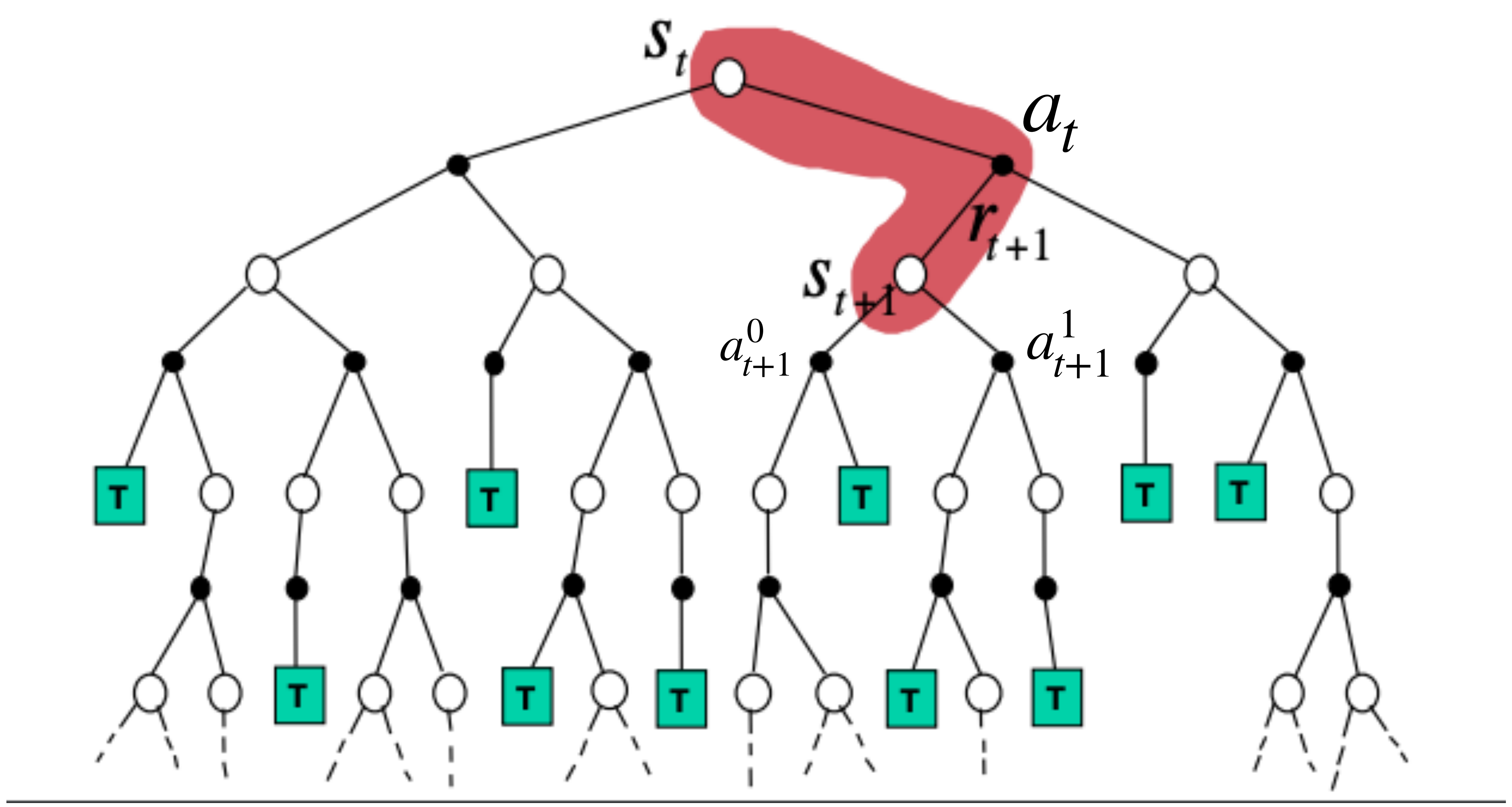
- Parameters: ε, α
- $\forall s, a \; Q_0(s, a) = 0$
- For $k = 0, 1, \ldots$

1. $a \sim \mu(\,.\,|\,s) = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q_k(s, a) \text{ with probabily } 1 - \varepsilon \end{cases}$
2. Observe r and $s' \sim p(\,.\,|\,s, a)$
3. $a \sim \mu(\,.\,|\,s') = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q_k(s', a) \text{ with probabily } 1 - \varepsilon \end{cases}$
4. $Q_{k+1}(s, a) \leftarrow Q_k(s, a) + \alpha(r + \gamma Q_k(s', a') - Q_k(s, a))$

Expected SARSA

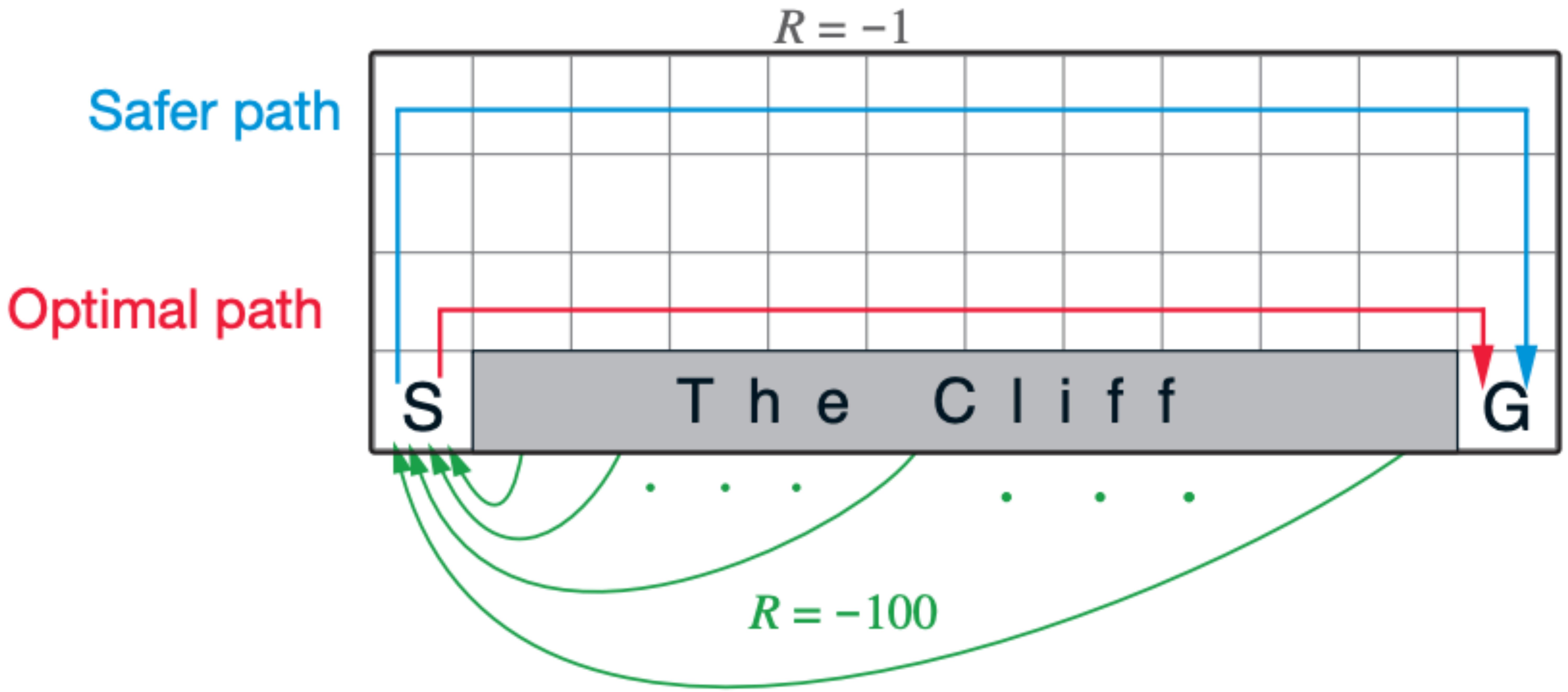
$$Q_{k+1}(s,a) \leftarrow Q_k(s,a) + \alpha(r + \gamma \mathbb{E}_{a' \sim \mu_k(.|s')} Q_k(s',a') - Q_k(s,a))$$

$$\mu_k(\,.\,|\,s') = \begin{cases} \text{select random action with probabily } \varepsilon \\ argmax_a Q_k(s',a) \text{ with probabily } 1-\varepsilon \end{cases}$$



Example: Cliff Walking

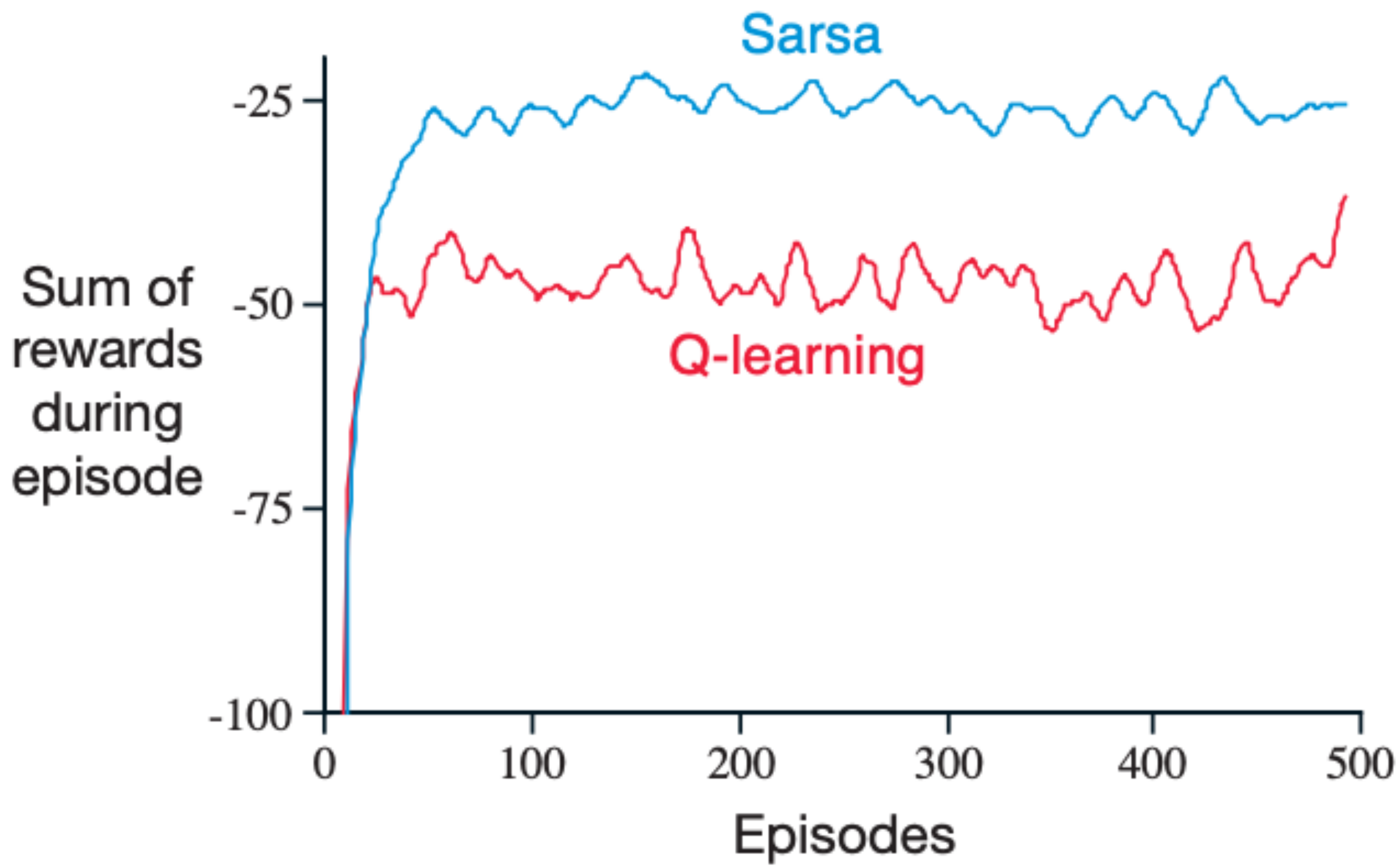
$\gamma = 1, \varepsilon = 0.1$. Agent gets -1 for each step.



Source

Which policy is learned by Q-learning and SARSA?

Example: Cliff Walking



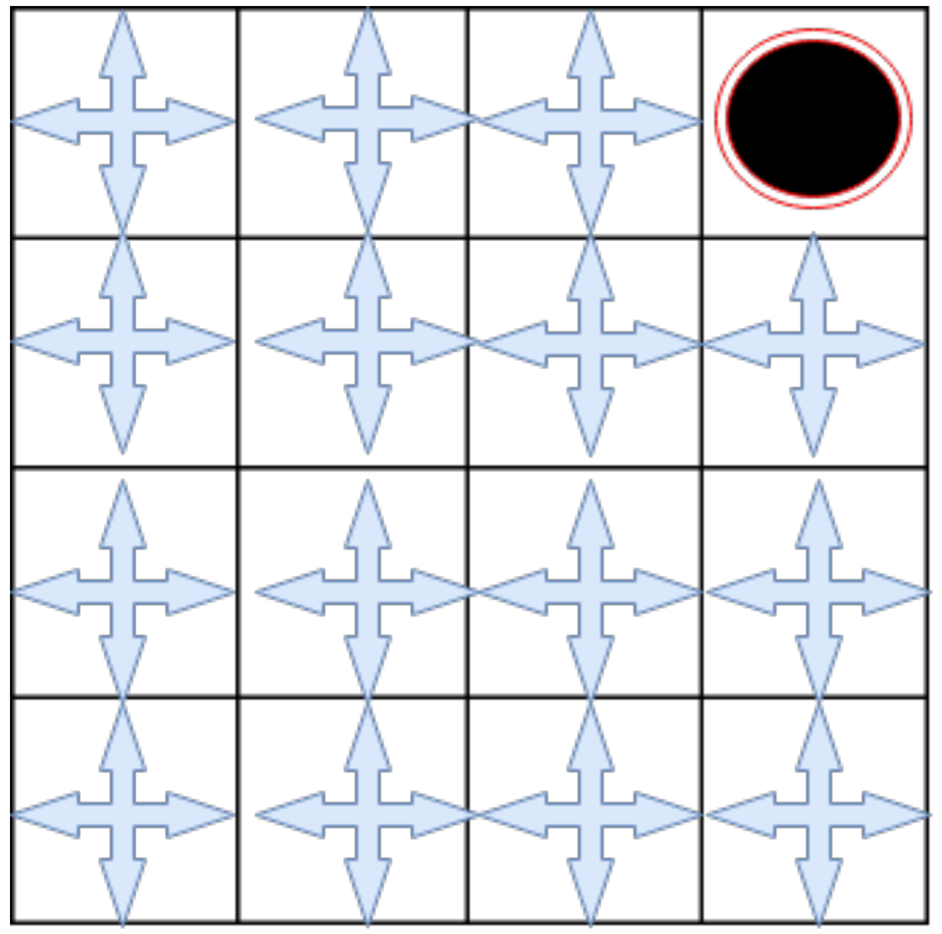
Source

Of course, if ε were gradually reduced, then both methods would asymptotically converge to the optimal policy.

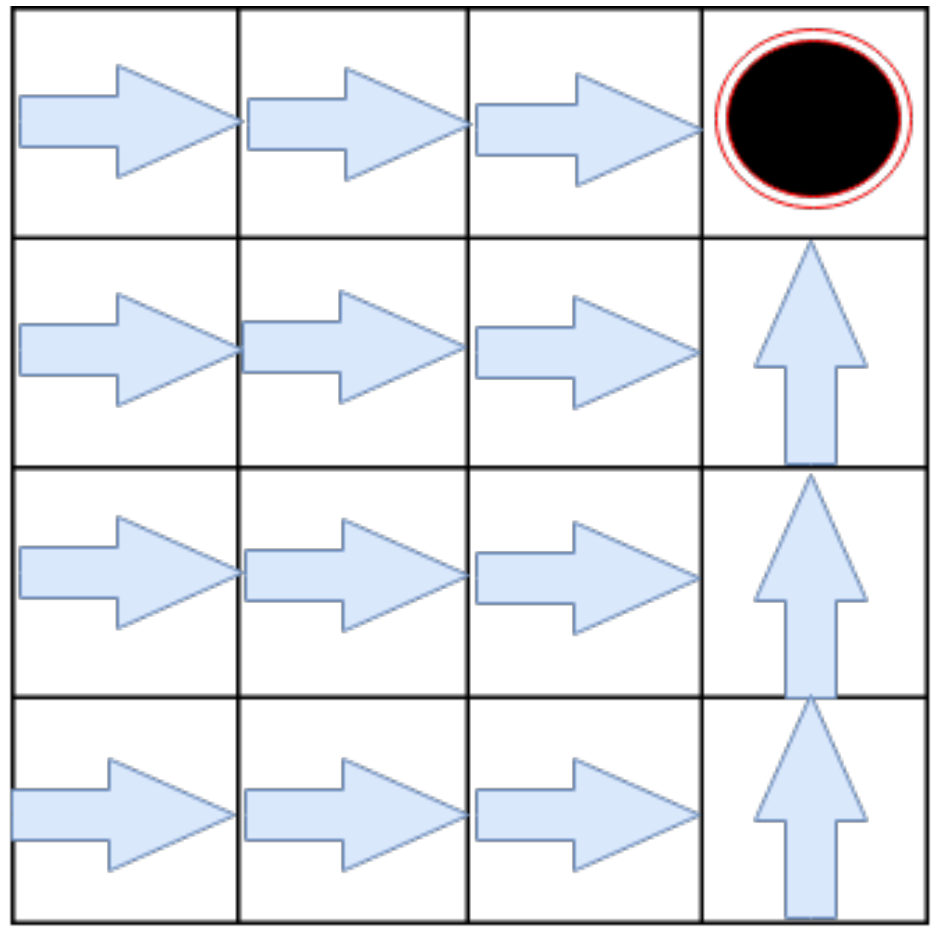
On-policy vs Off-Policy

On-policy learning

- Learn about behaviour policy μ from experience sampled from μ



Behavior Policy



Target Policy

Off-policy learning

- Learn about target policy π from experience sampled from μ
- Learn from observing humans or other agents (e.g., from logged data)
- Learn about multiple policies while following one policy
- Learn about greedy policy while following exploratory policy
- Reuse experience from old policies (e.g., from your own past experience)

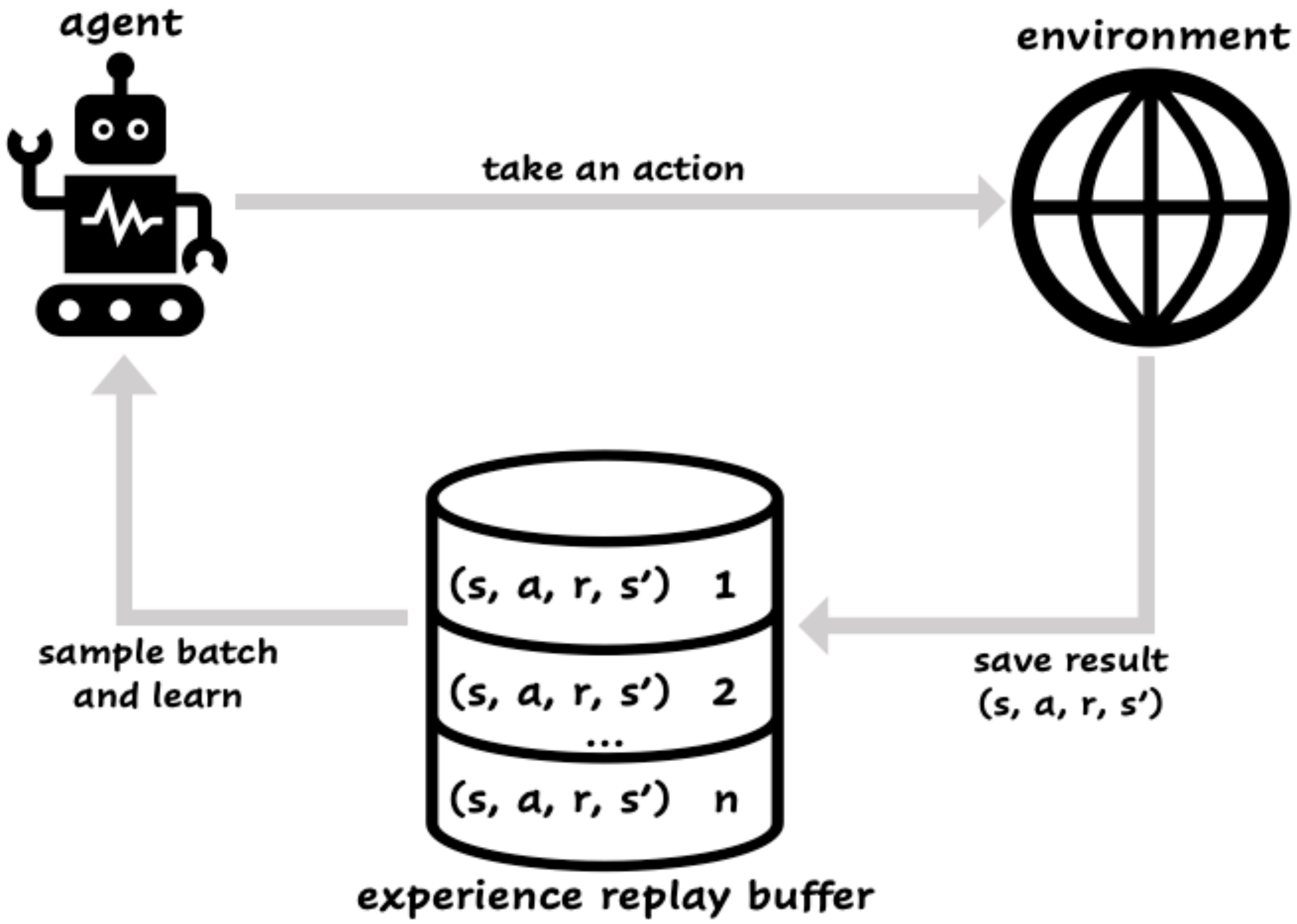
Experience Replay Buffer

Advantages:

- No need to revisit same states many times
- Make the estimators consistent with the current policy, update estimators
- Decorrelate update samples to maintain i.i.d. assumption

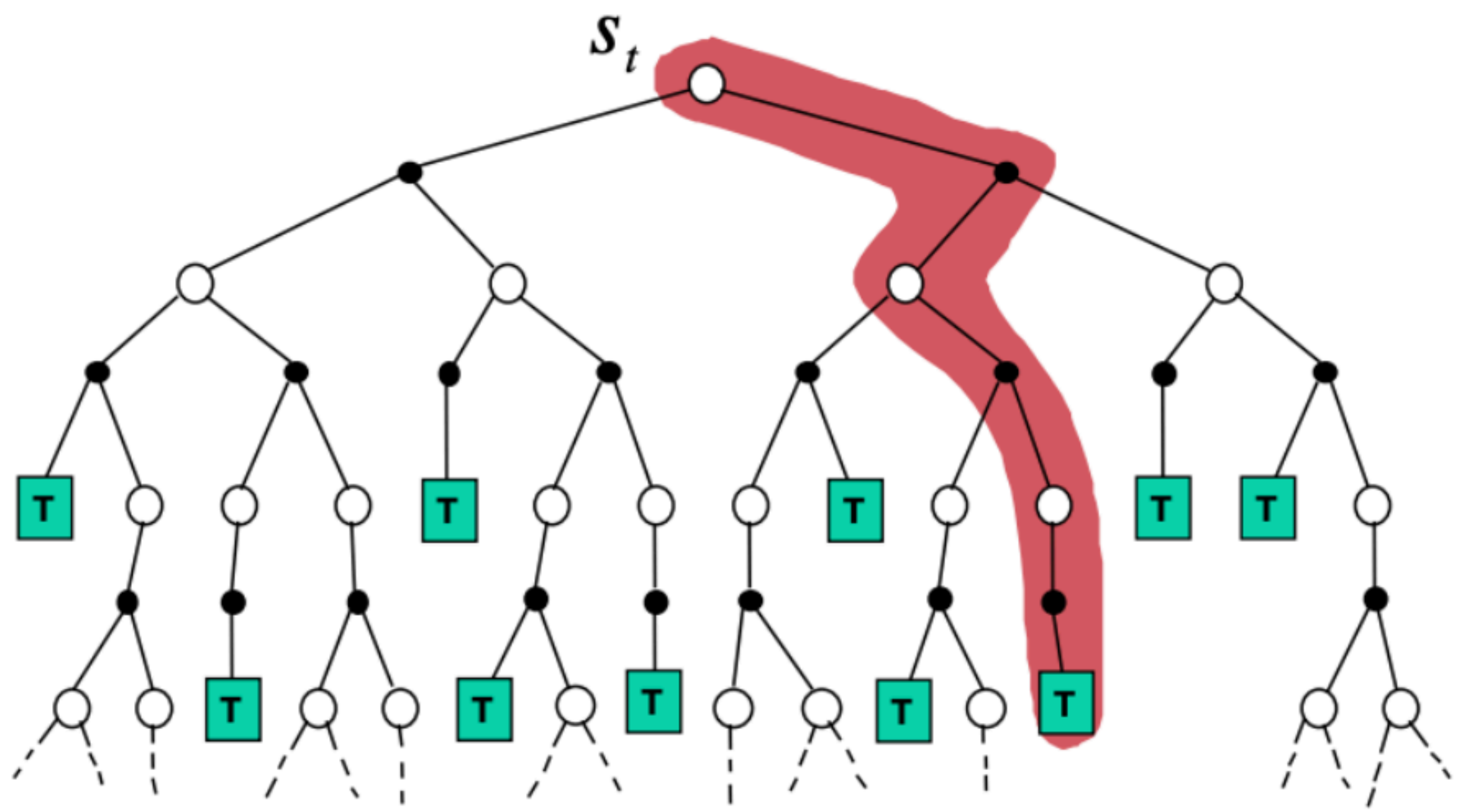
Disadvantages:

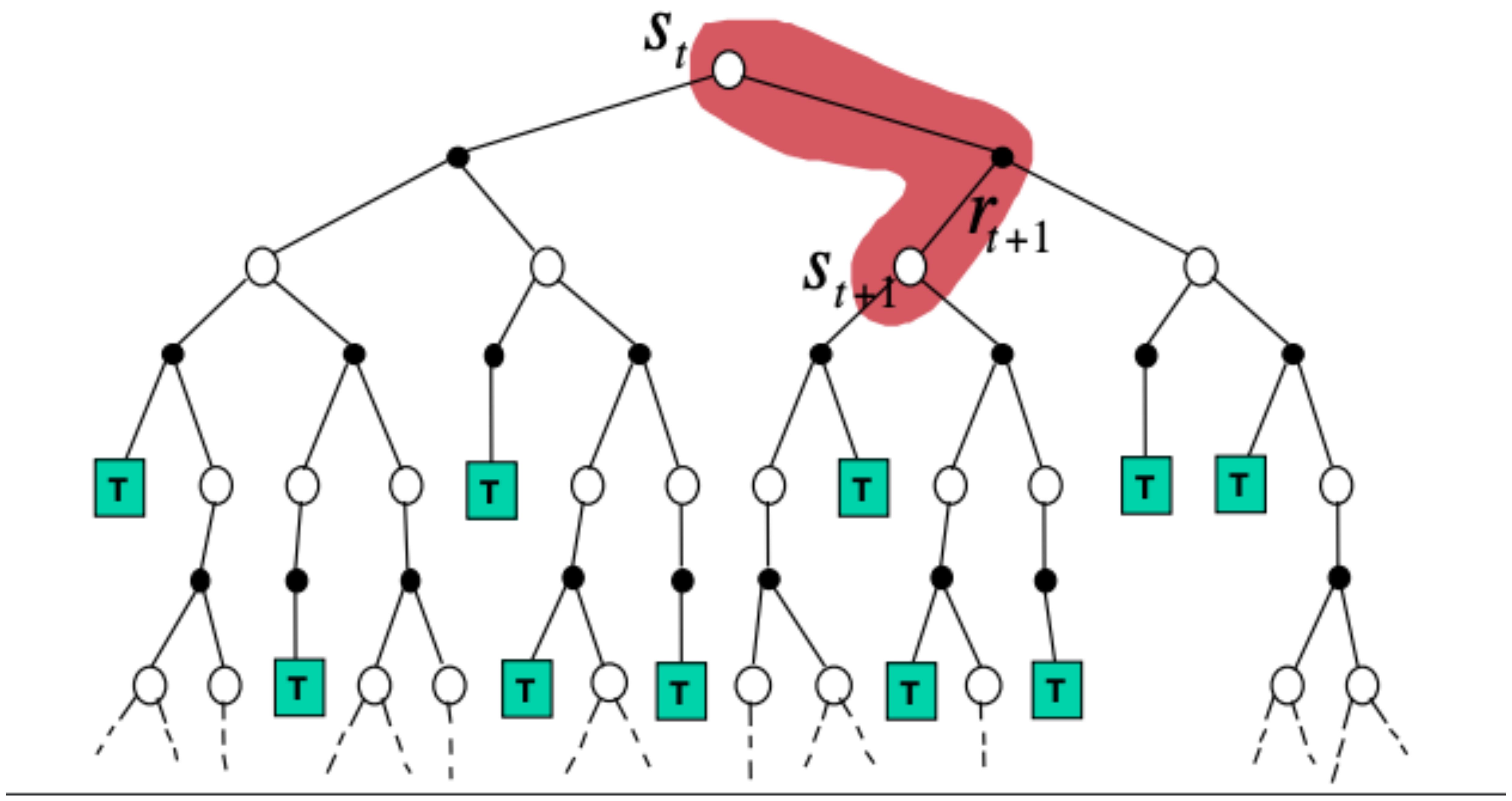
- Not applicable for the on-policy learning



Bias-Variance Trade-off

$$Q(s,a) \leftarrow Q(s,a) + \alpha(y_Q - Q(s,a))$$





TD(N)

$$Q(s, a) \leftarrow Q(s, a) + \alpha(y_Q - Q(s, a))$$

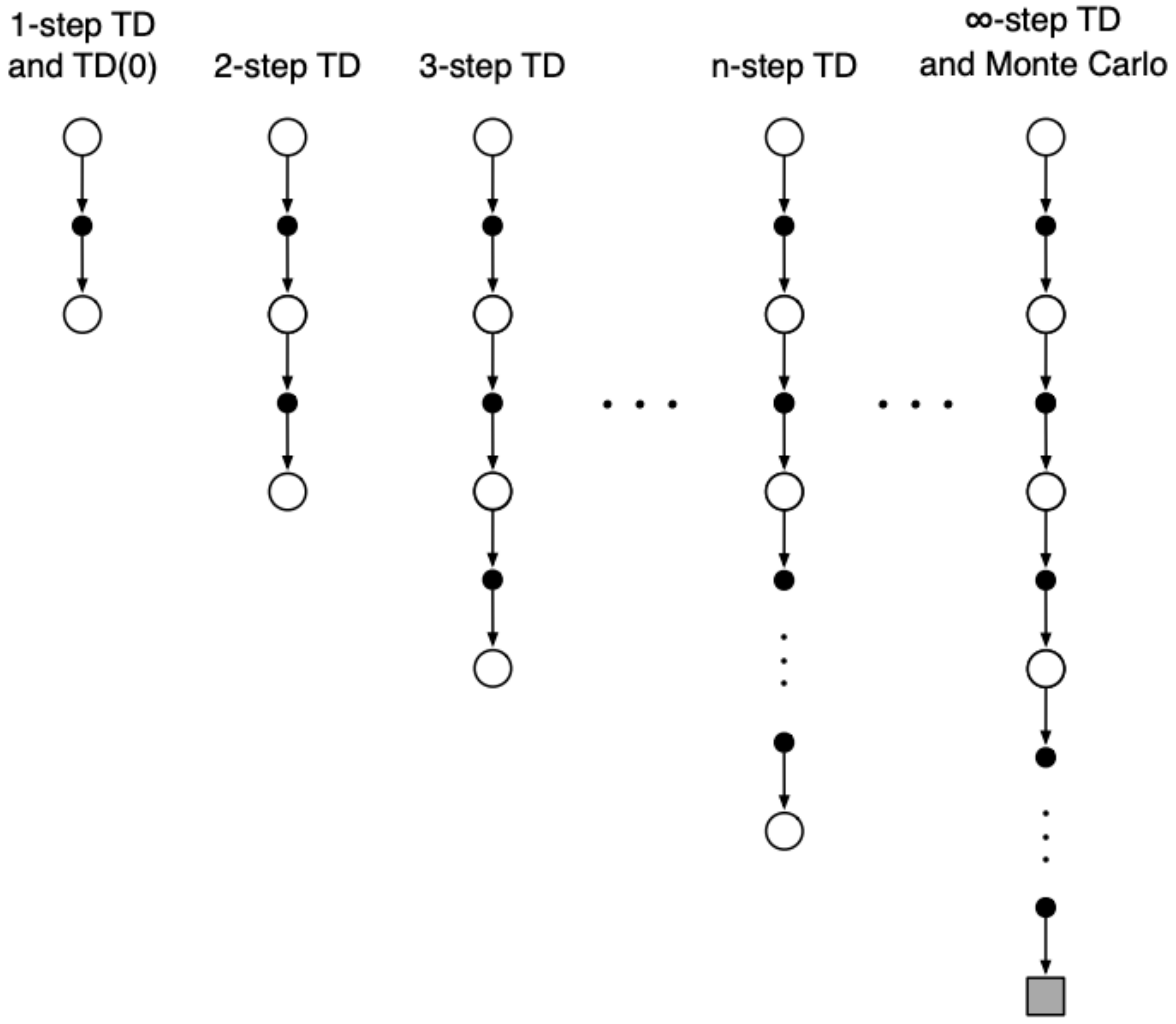
$$y_Q = r + \gamma r' + \gamma^2 r'' + \ldots + \gamma^N Q(s^{(N)}, a^{(N)})$$

$$s^{(n)} \sim p(\,.\,|\,s^{(n-1)}, a^{(n-1)}),\, a^{(n)} \sim \mu(a\,|\,s)$$

Ranging N we can control the trade-off between bias and variance.

There are more advanced methods, applicable even for off-policy algorithms:

See TD(λ) and Retrace(λ)



Background

1. Reinforcement Learning Textbook (in Russian): 3.4 - 3.5
2. Sutton & Barto, Chapter 5 + 6 + 7*
3. Practical RL course by YSDA, week 3
4. DeepMind course, lectures 5 + 6

Thank you for your attention!