Policy Evaluation

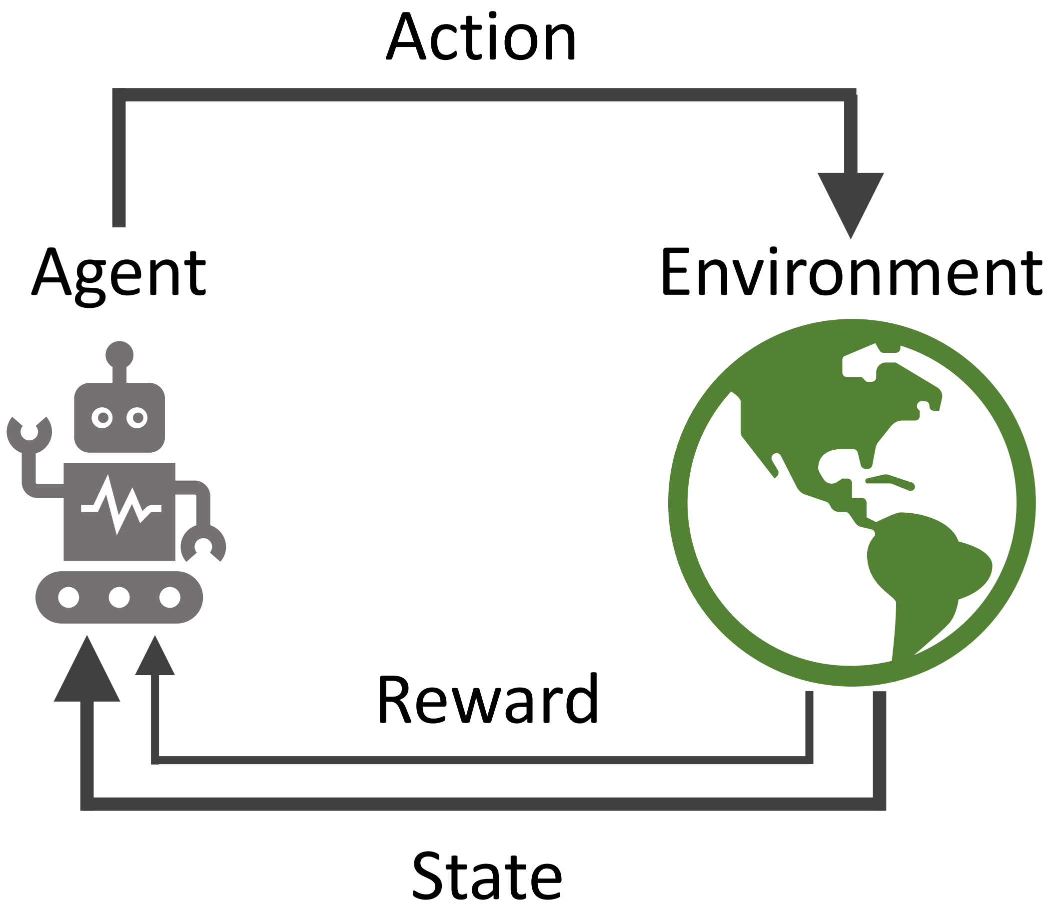
CMPT 729 G100

Jason Peng

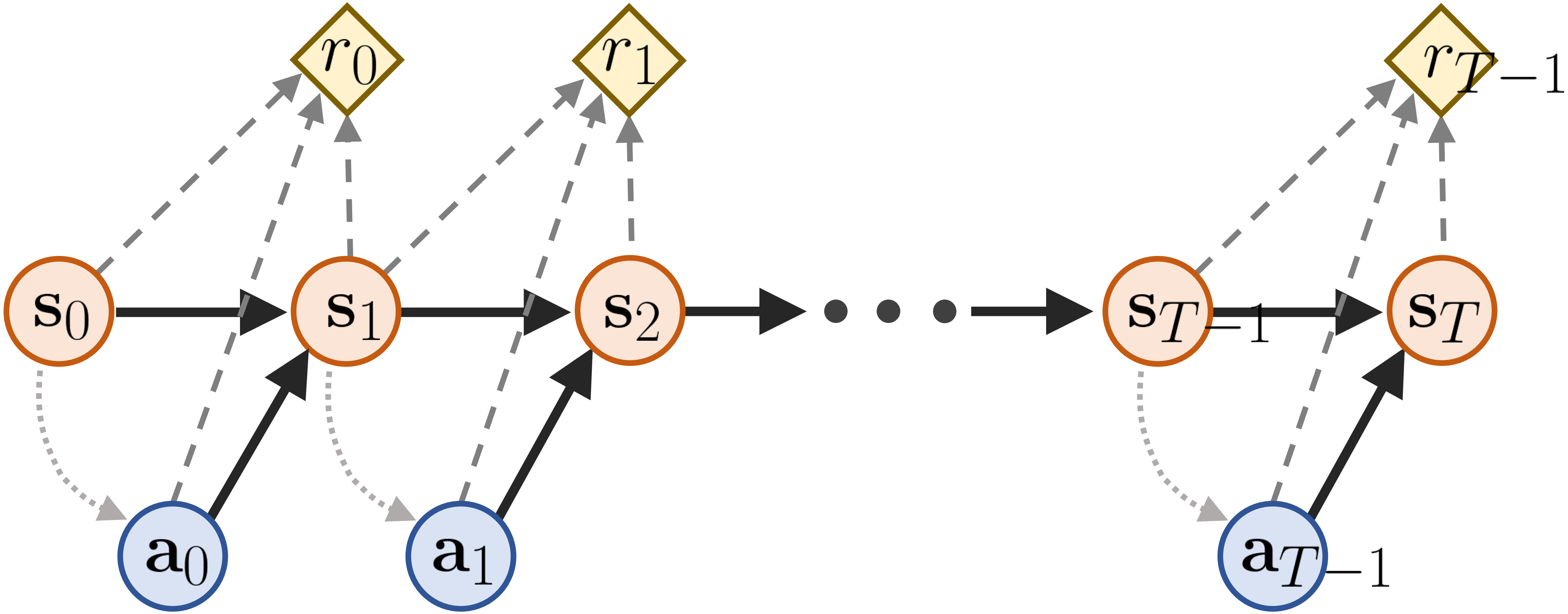
Overview

- Policy Evaluation
- Value Functions
- Monte-Carlo Methods
- Dynamic Programming Methods
- Optimal Policies

Agent-Environment Interface



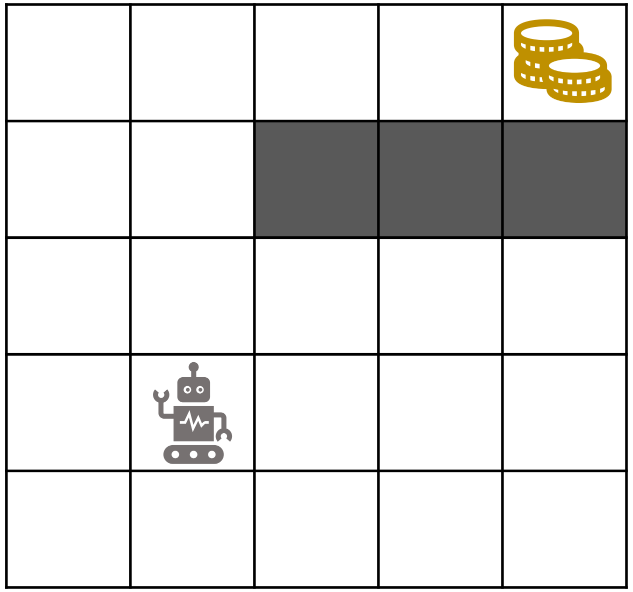
Markov Decision Process

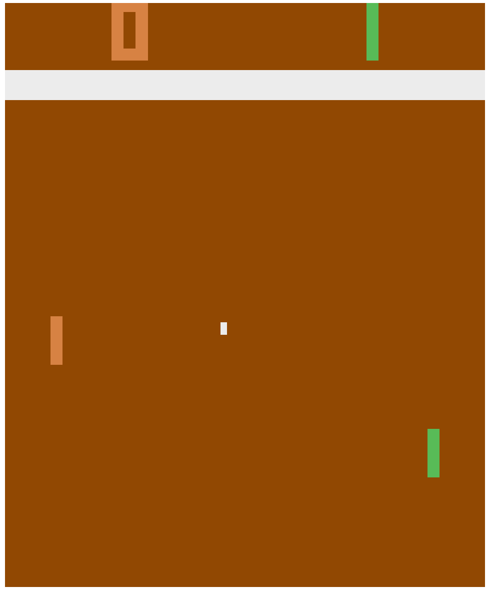


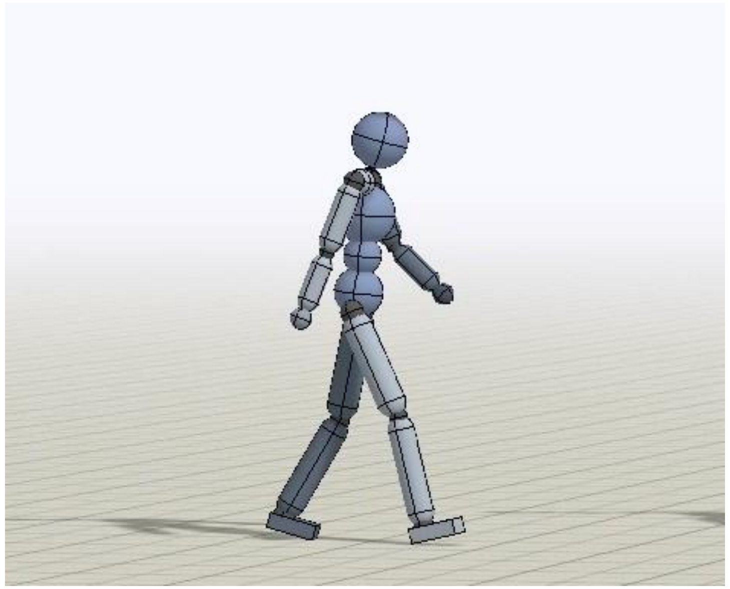
Policy Evaluation

How good is it?

$$\mathbf{s} \Rightarrow \boxed{\pi} \Rightarrow \mathbf{a}$$







Performance

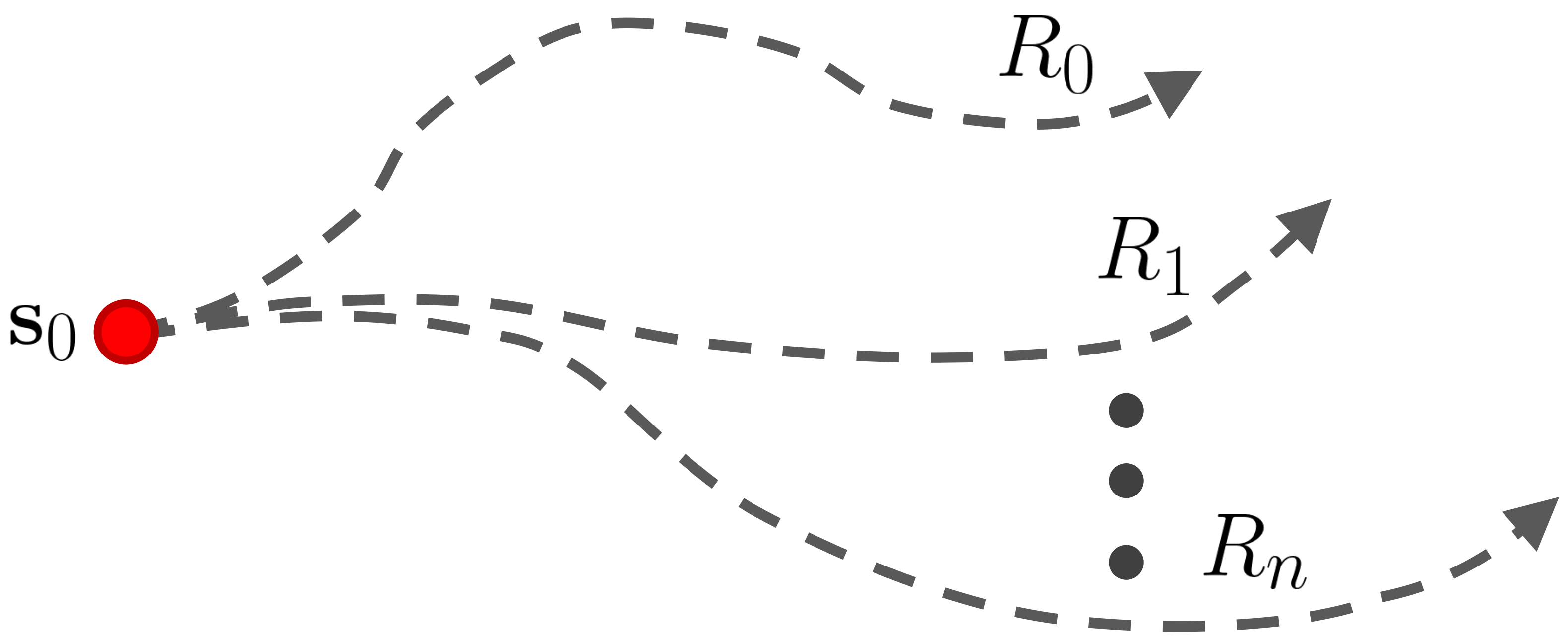
$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

Expected Return

Episodic Return

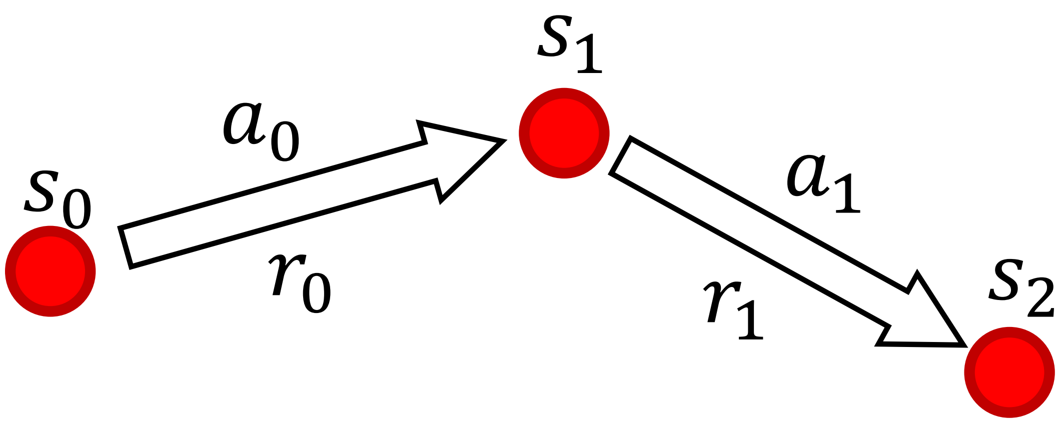
Monte-Carlo Estimate

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

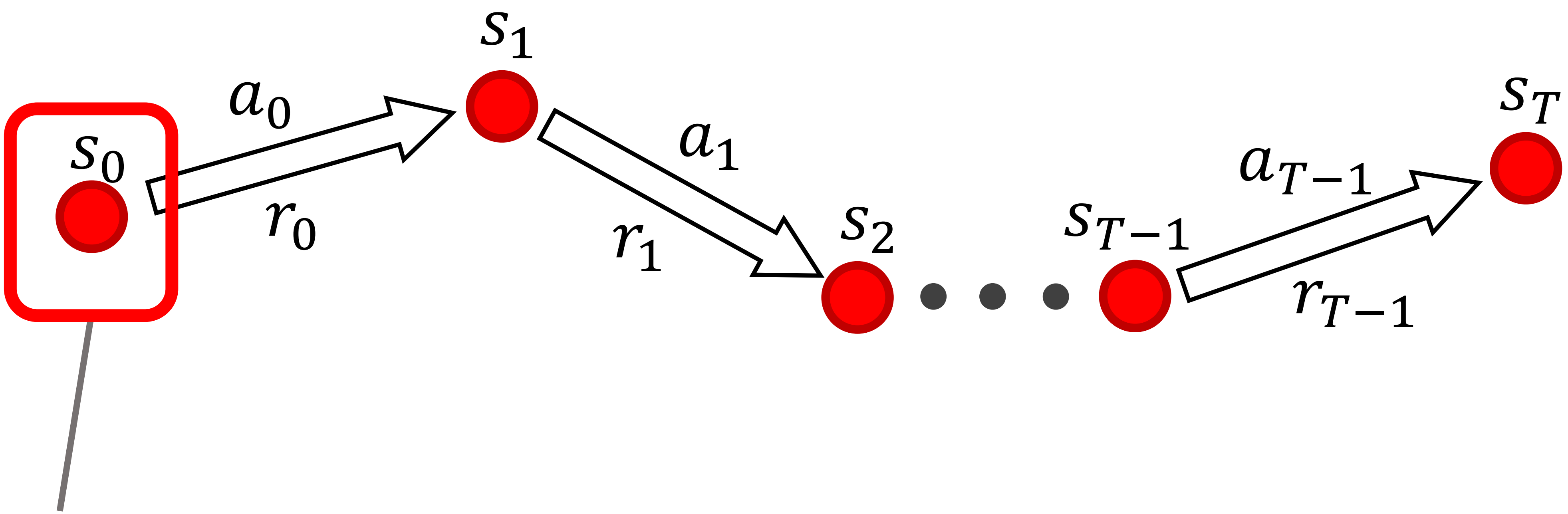


$$J(\pi) \approx \frac{1}{n} \sum_{i=0}^{n} R_i$$

Reward-To-Go

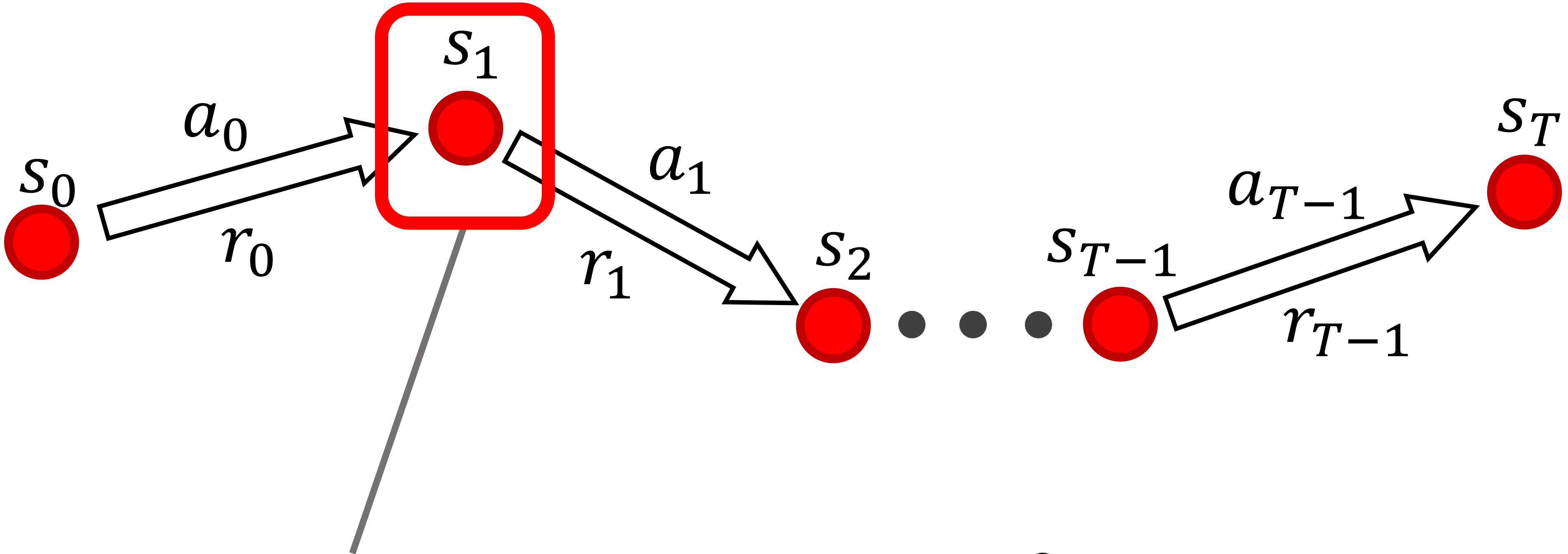


Reward-To-Go



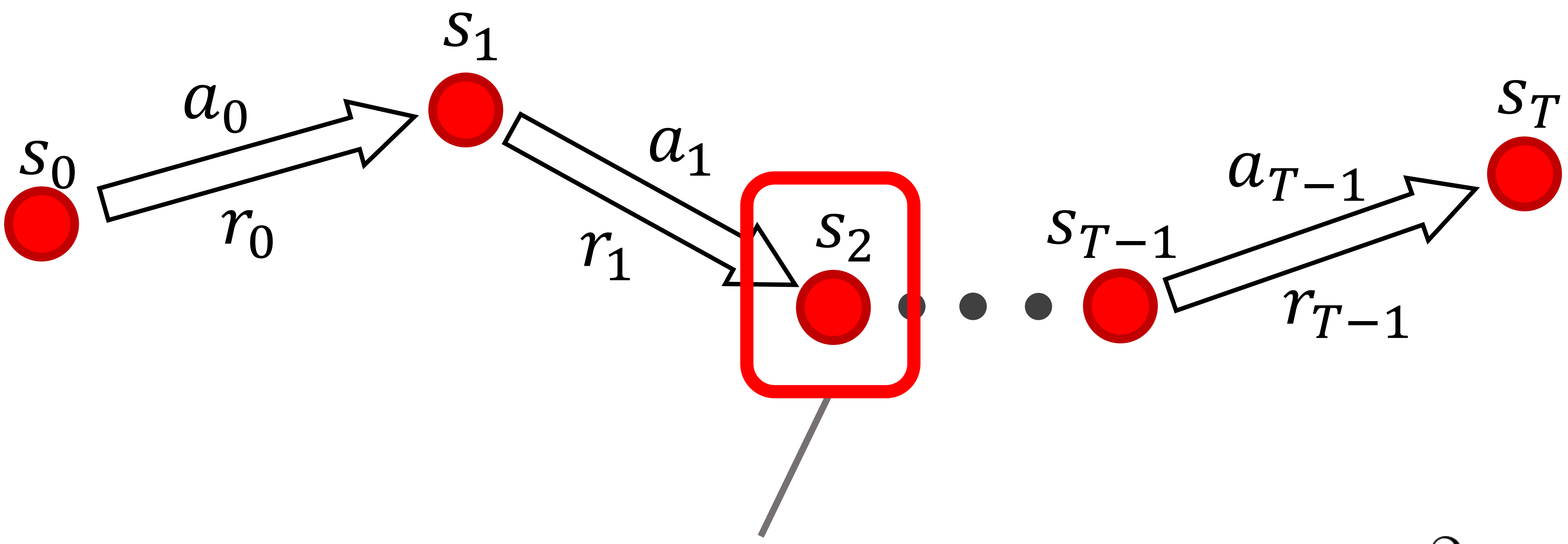
$$R_0 = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

Reward-To-Go



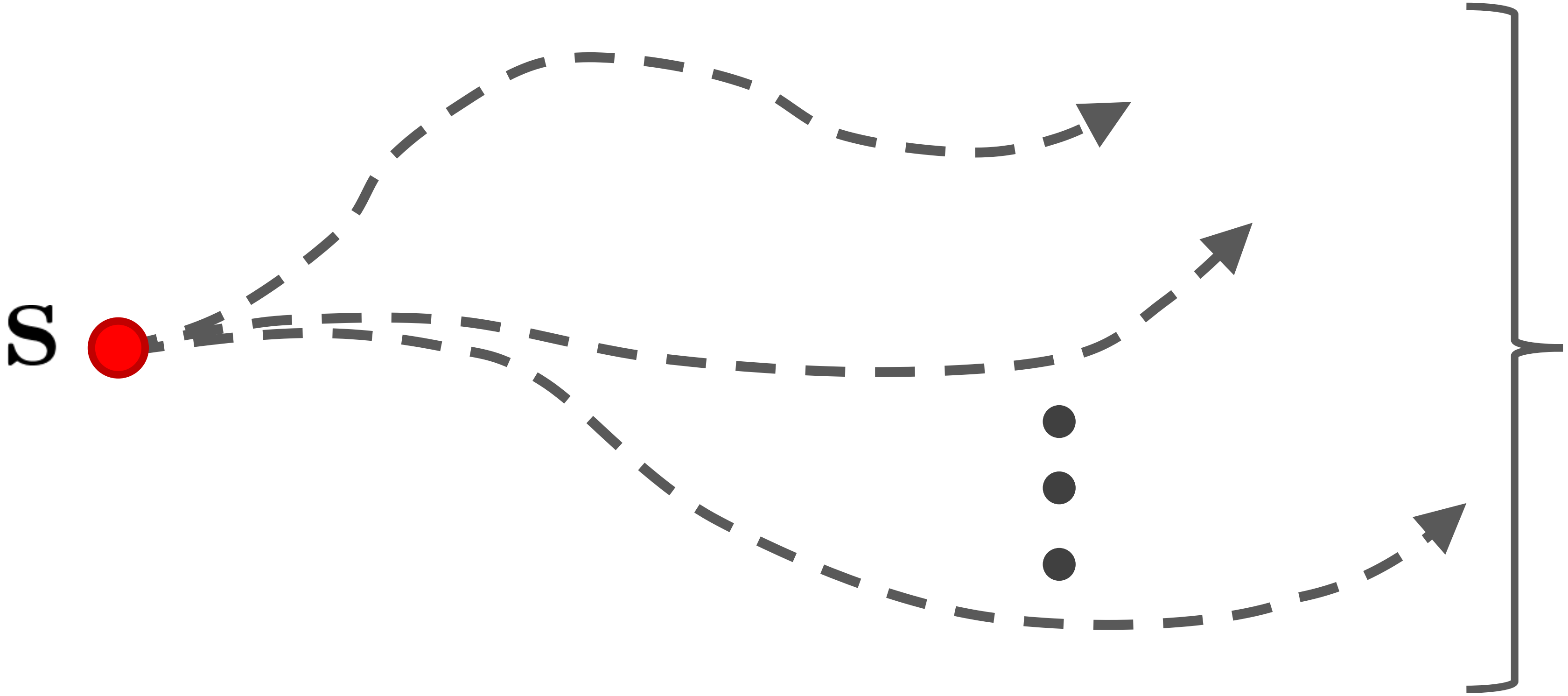
$$R_1 = r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots$$

Reward-To-Go



$$R_2 = r_2 + \gamma r_3 + \gamma^2 r_4 + \ldots$$

Reward-To-Go



$$\mathbb{E}_{\pi}\left[R|\mathbf{s}_0 = \mathbf{s}\right]$$

Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\pi}\left[R | \mathbf{s}_0 = \mathbf{s}\right]$$

Value Function

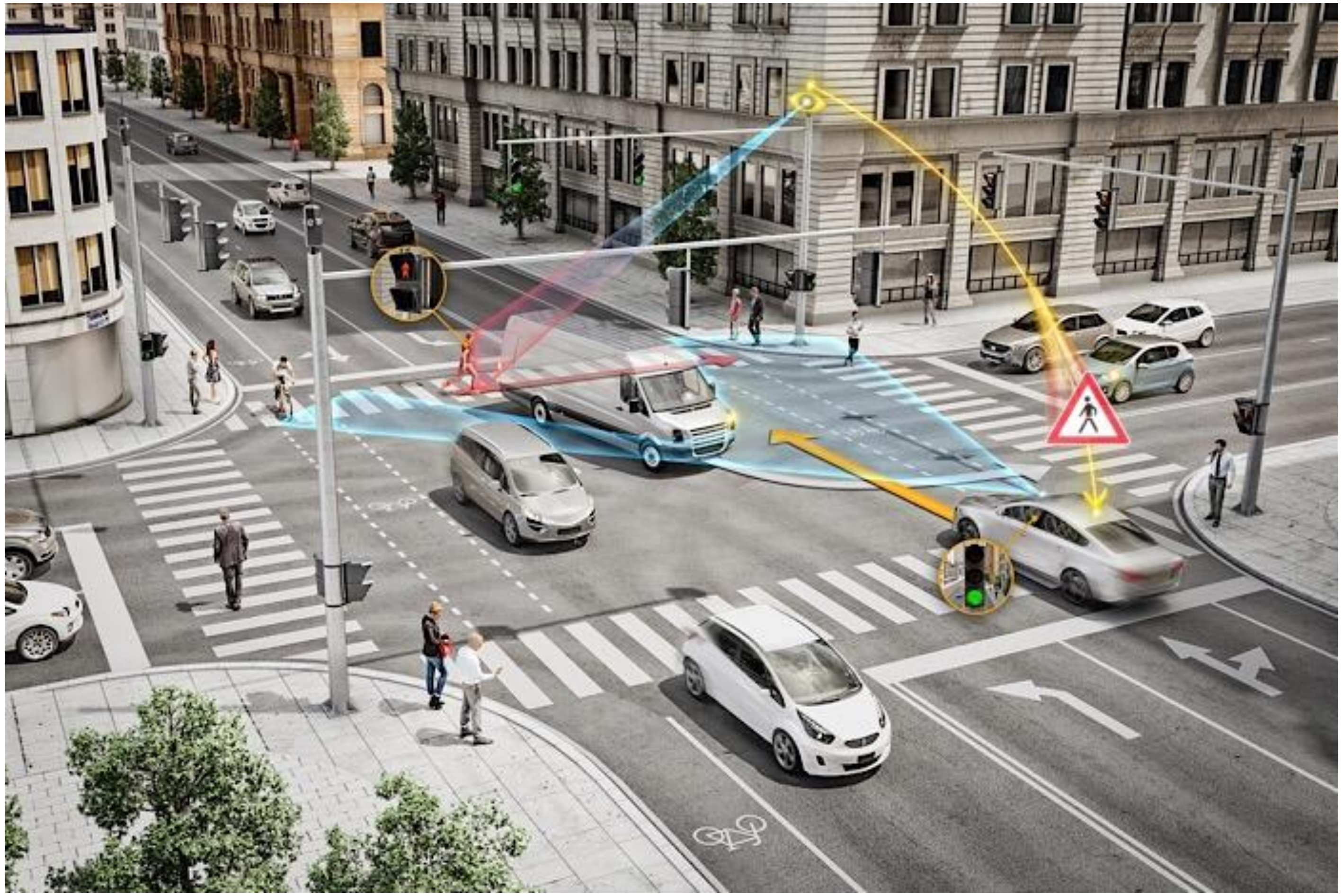
- Input: state $\boldsymbol{s}$
- Output: expected return of following a policy π start at a state $\boldsymbol{s}$

Value Function

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

$$J(\pi) = V^{\pi}(\mathbf{s}_0)$$

Why a Value Function?

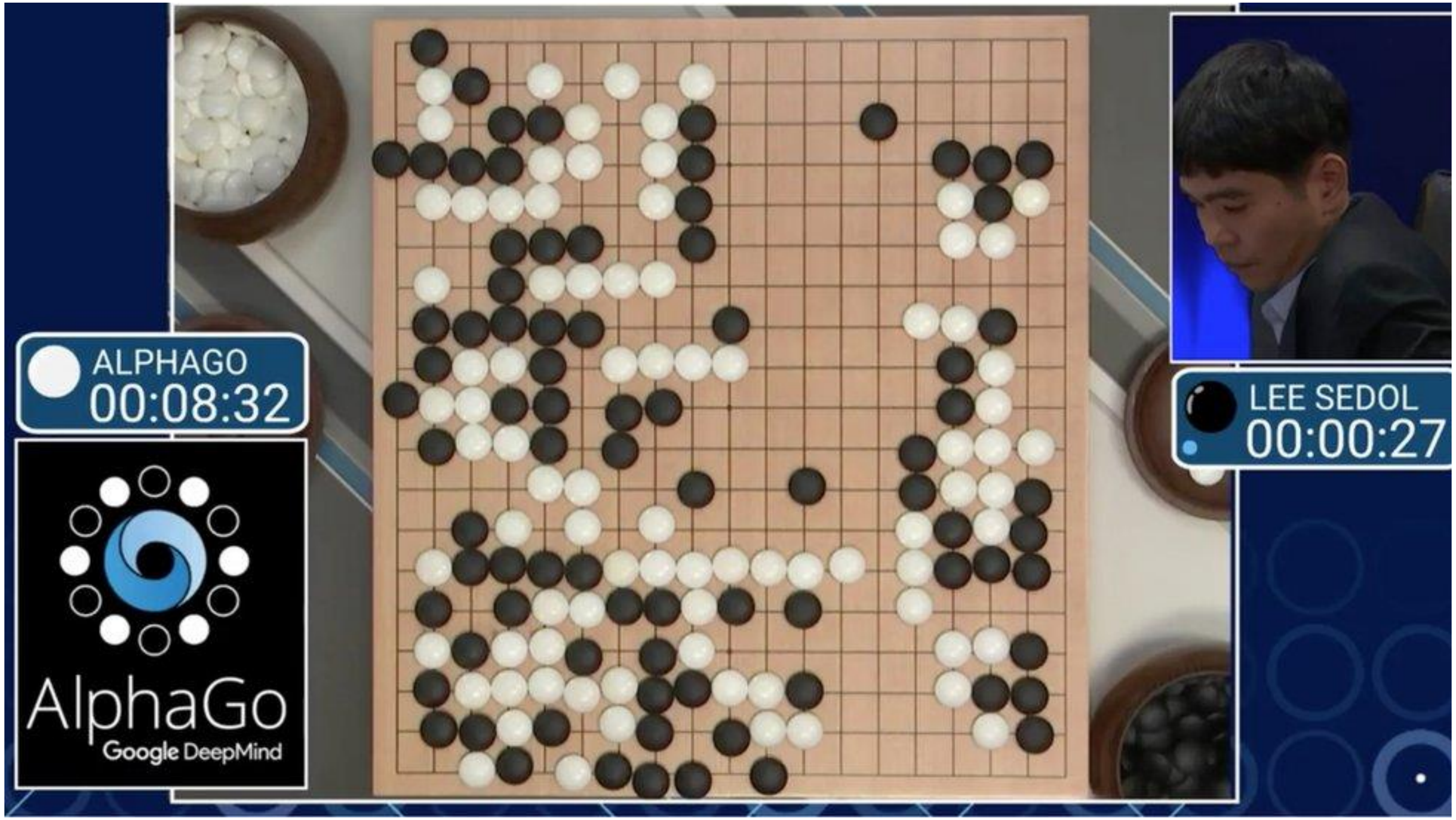


[Impact Lab]

$$V^{\pi}(\mathbf{s})?$$

Can the policy do well here?

Why a Value Function?

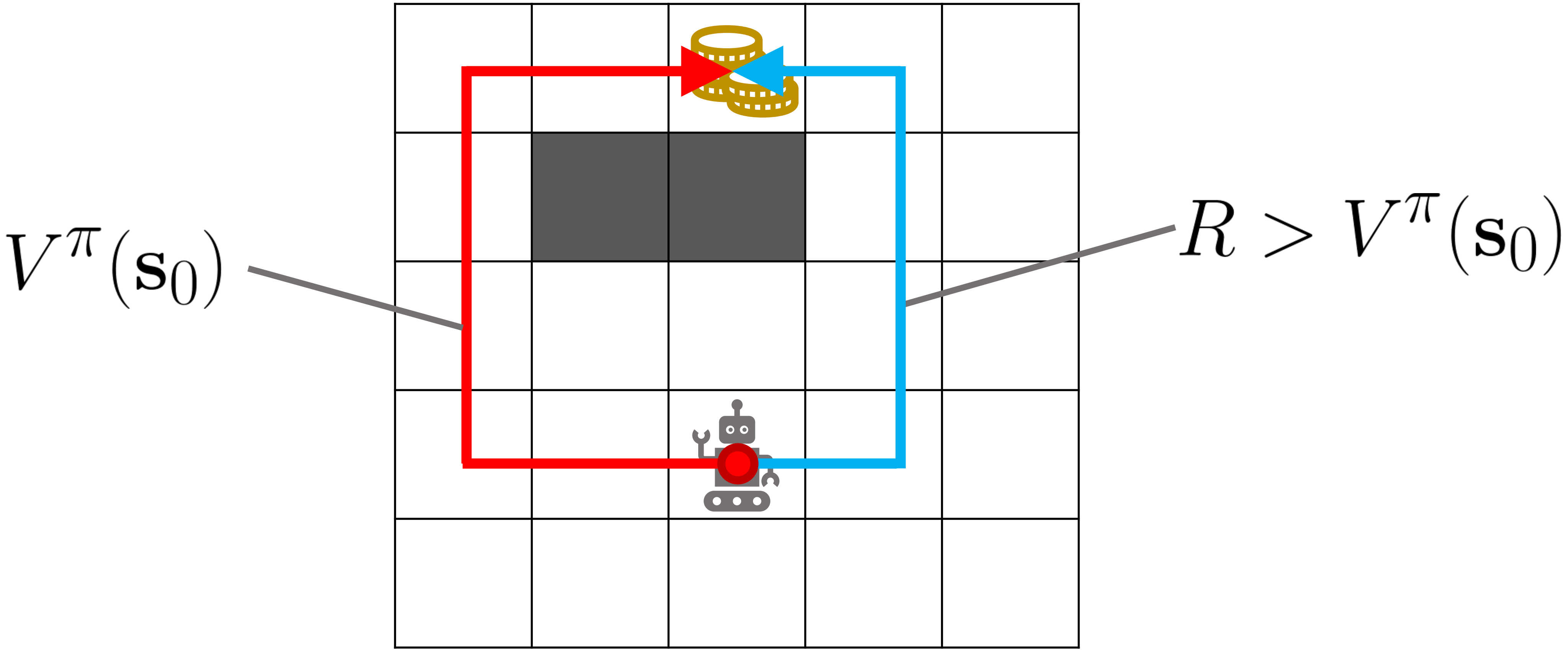


AlphaGo [DeepMind]

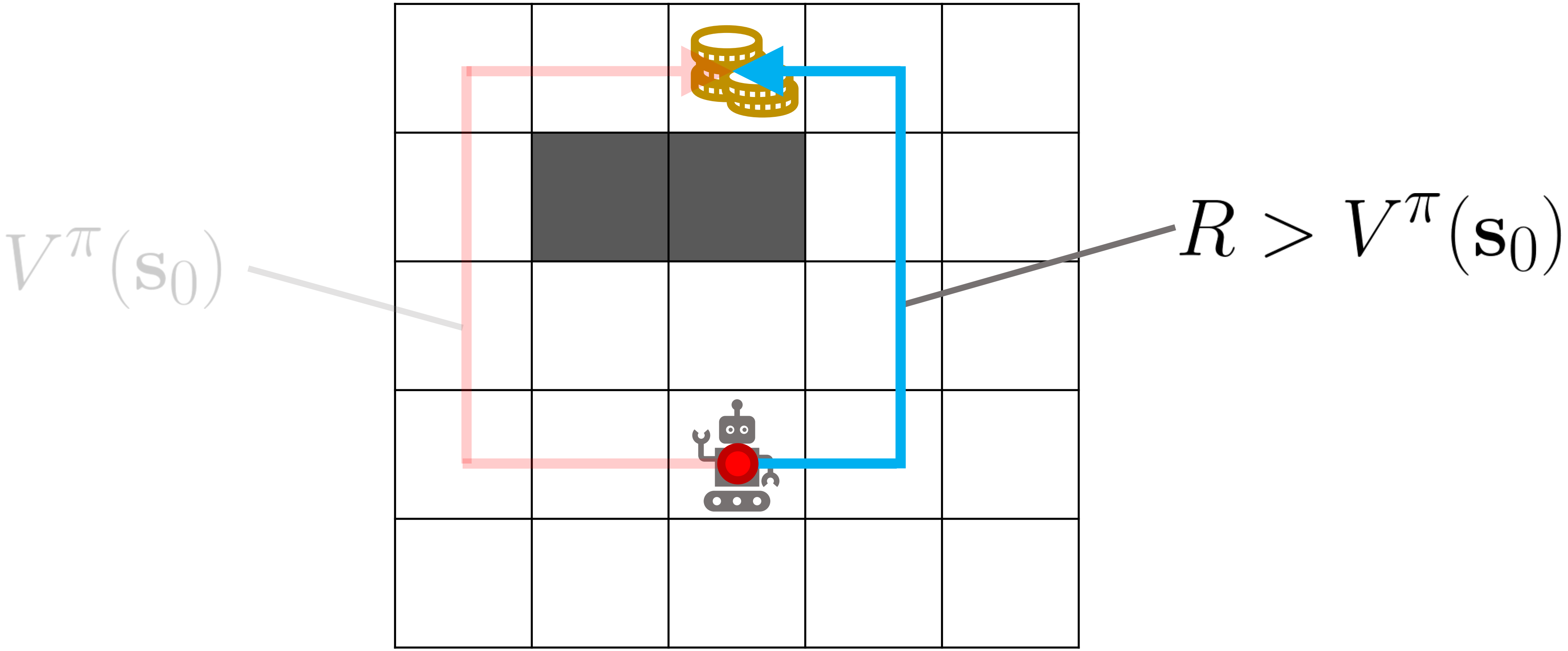
$$V^{\pi}(\mathbf{s})?$$

Can the policy win?

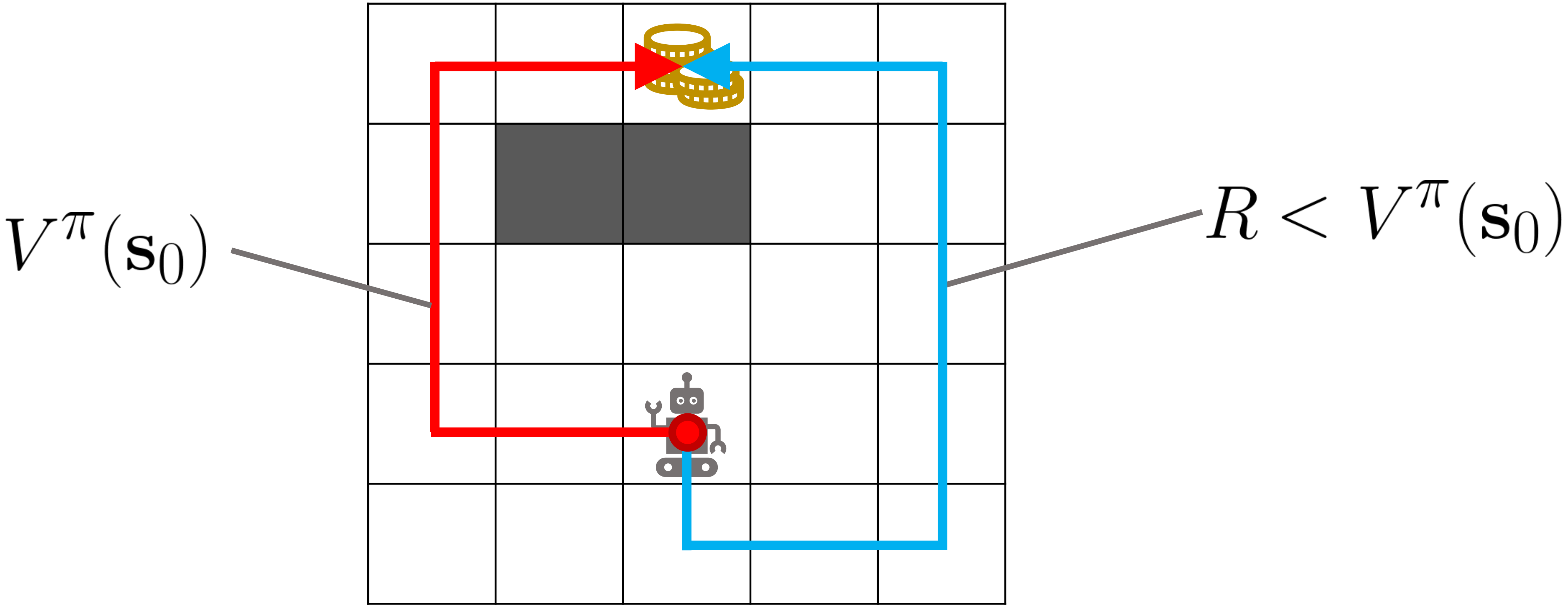
Policy Improvement



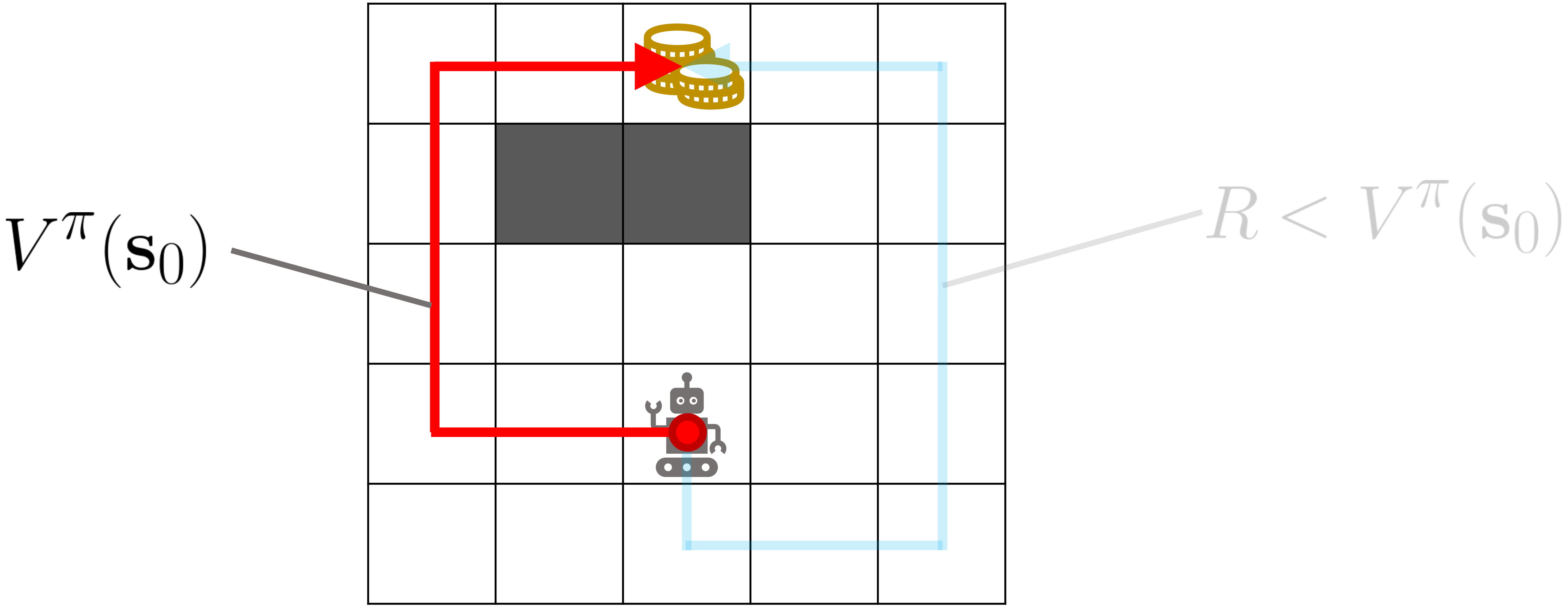
Policy Improvement



Policy Improvement



Policy Improvement



Value Function Approximation

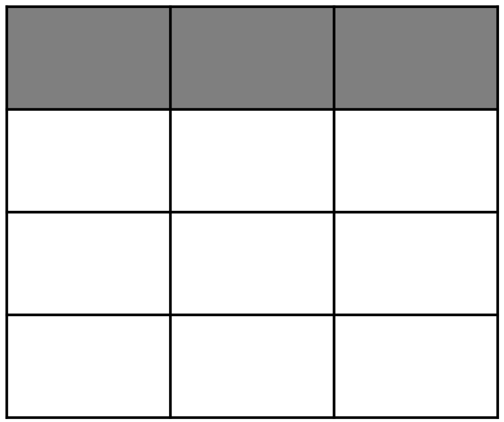
$\boxed{V^{\pi}}$?

Value Function Approximation

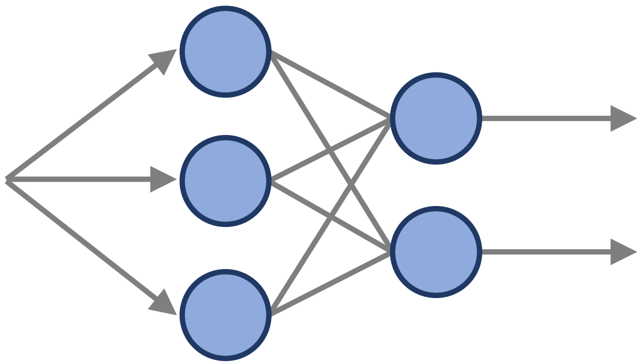
$$\underline{\hat{V}^{\pi}} \approx V^{\pi}$$

Function Approximator

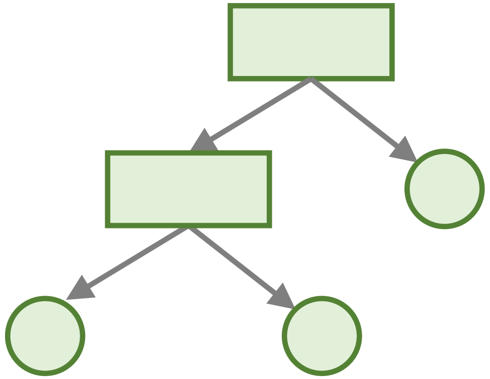
Look-Up Table



Neural Network



Decision Tree



Etc...

Learning

$$\mathbf{s}_0 \xrightarrow[r_0]{\mathbf{a}_0} \mathbf{s}_1 \xrightarrow[r_1]{\mathbf{a}_1} \mathbf{s}_2 \cdots \mathbf{s}_{T-1} \xrightarrow[r_{T-1}]{\mathbf{a}_{T-1}} \mathbf{s}_T$$

$$\left.\begin{array}{l} R_0 = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \Rightarrow y_0 \\ R_1 = r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots \Rightarrow y_1 \\ R_2 = r_2 + \gamma r_3 + \gamma^2 r_4 + \ldots \Rightarrow y_2 \\ \vdots \end{array}\right\} \begin{array}{c} \text{Dataset} \\ \{\mathbf{s}_i, y_i\} \end{array}$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \ \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[||y_i - V(\mathbf{s}_i)||^2 \right]$$

Supervised Learning

$$\underline{\hat{V}^{\pi}} = \arg\min_{V} \ \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[||y_i - V(\mathbf{s}_i)||^2 \right]$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \; \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[||y_i - V(\mathbf{s}_i)||^2 \right]$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \underbrace{\mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[||y_i - V(\mathbf{s}_i)||^2 \right]}_{\text{Mean Prediction Error}}$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \ \underbrace{\mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)}}_{\text{Collect data from policy}} \left[||y_i - V(\mathbf{s}_i)||^2 \right]$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \; \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[\underbrace{||y_i - V(\mathbf{s}_i)||^2}_{\text{Prediction Error}} \right]$$

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \; \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y | \pi)} \left[|| \underline{y_i} - V(\mathbf{s}_i) ||^2 \right]$$

"Target Value"

Monte-Carlo Estimate

$$y = \sum_{t=0}^{T-1} \gamma^t r_t$$

Monte-Carlo Method

$$y = \sum_{t=0}^{T-1} \gamma^t \underline{r_t}$$

Random Variable

✓ Unbiased

✗ High variance → Slow convergence

Dynamic Programming

Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

Likelihood of a trajectory under π starting at $\mathbf{s}$

Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

Recursive Property of Value Function

$$
\begin{aligned}
V^{\pi}(\mathbf{s}) &= \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right] \\
&= \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})}\left[\underline{r_0} + \sum_{\underline{t=1}}^{T-1} \gamma^t r_t\right]
\end{aligned}
$$

Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s})}\left[\sum_{t=0}^{T-1}\gamma^t r_t\right]$$

$$= \mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s})}\left[r_0 + \sum_{t=1}^{T-1}\gamma^t r_t\right]$$

$$= \mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s})}\left[r_0 + \underline{\gamma}\sum_{t=1}^{T-1}\gamma^{t-1} r_t\right]$$

Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

Return starting at $\mathbf{s}_1$

Immediate reward at $\mathbf{s}_0$

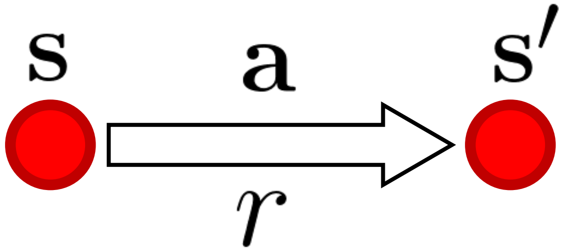
Recursive Property of Value Function

$$V^\pi(\mathbf{s}) = \underline{\mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})}} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

$$= \underline{\mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')}} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

Recursive Property of Value Function

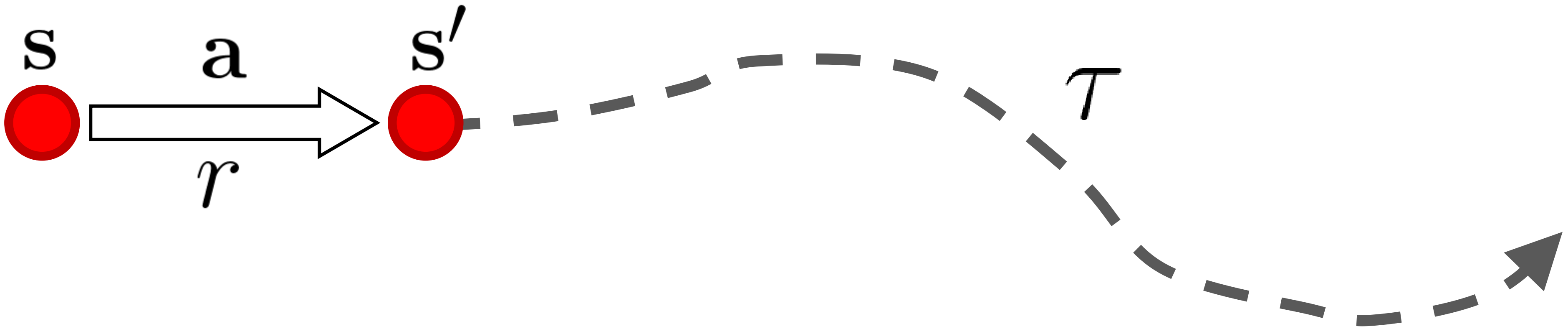
$$
\begin{aligned}
V^{\pi}(\mathbf{s}) &= \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})}\left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right] \\
&= \underline{\mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})}} \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')}\left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]
\end{aligned}
$$



Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

$$= \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \underline{\mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')}} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$



Recursive Property of Value Function

$$
\begin{aligned}
V^{\pi}(\mathbf{s}) &= \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right] \\
&= \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right] \\
&= \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \left[r_0 + \gamma \underline{\mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')} \left[\sum_{t=0}^{T-1} \gamma^{t} r_t \right]} \right]
\end{aligned}
$$

Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s})} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

$$= \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')} \left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t \right]$$

$$= \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \mathbb{E}_{\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \left[r + \gamma \underline{\mathbb{E}_{\tau \sim p(\tau|\pi, \mathbf{s}_0 = \mathbf{s}')} \left[\sum_{t=0}^{T-1} \gamma^{t} r_t \right]} \right]$$

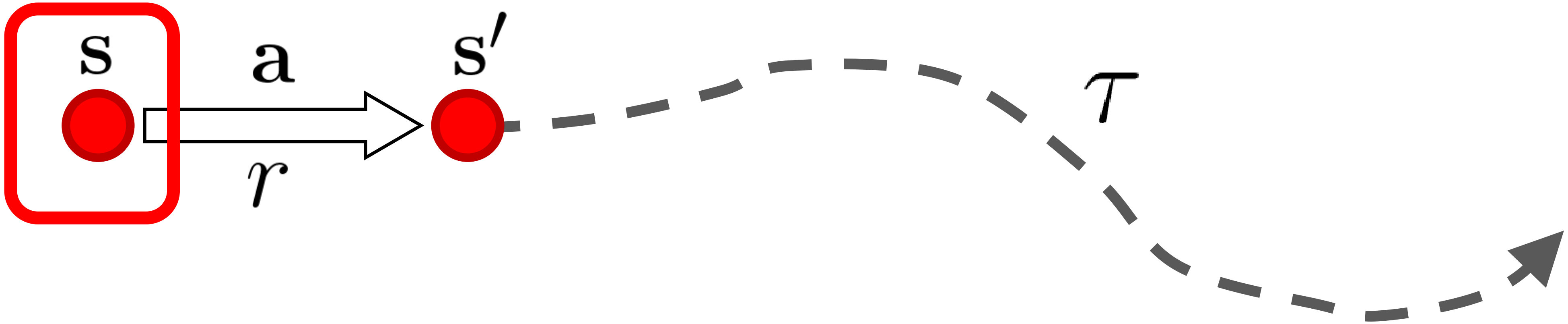
$$= V^{\pi}(\mathbf{s}')$$

Recursive Property of Value Function

$$
\begin{aligned}
V^\pi(\mathbf{s}) &= \mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s})}\left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t\right] \\
&= \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s}')}\left[r_0 + \gamma \sum_{t=1}^{T-1} \gamma^{t-1} r_t\right] \\
&= \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma \mathbb{E}_{\tau\sim p(\tau|\pi,\mathbf{s}_0=\mathbf{s}')}\left[\sum_{t=0}^{T-1} \gamma^{t} r_t\right]\right] \\
&= \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma V^\pi(\mathbf{s}')\right]
\end{aligned}
$$

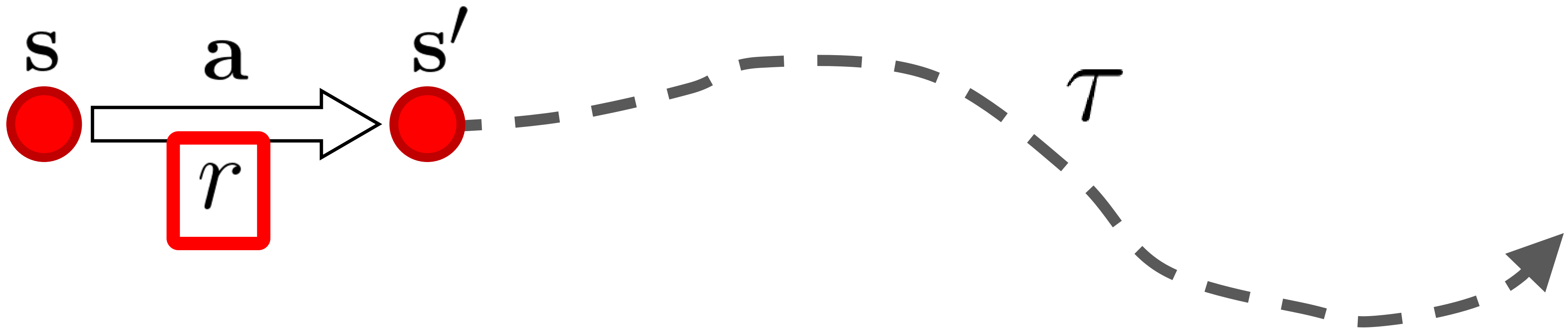
Recursive Property of Value Function

$$\underline{V^\pi(\mathbf{s})} = \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma V^\pi(\mathbf{s}')\right]$$



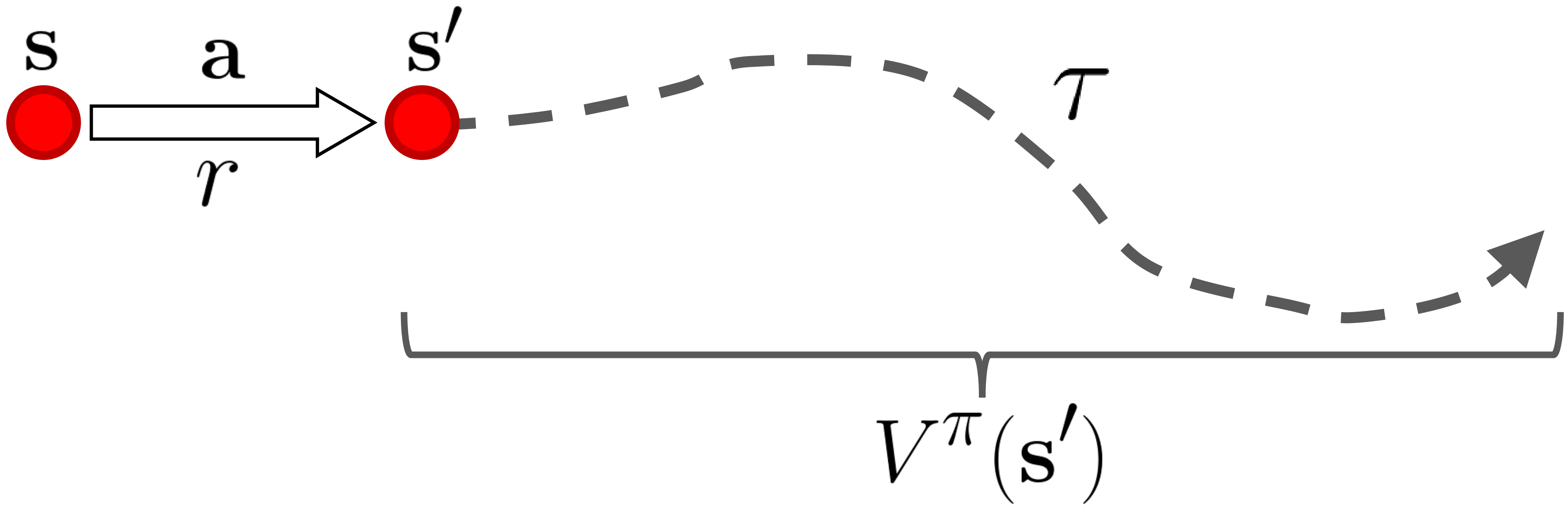
Recursive Property of Value Function

$$V^{\pi}(\mathbf{s}) = \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[\underline{r} + \gamma V^{\pi}(\mathbf{s}')\right]$$



Recursive Property of Value Function

$$V^\pi(\mathbf{s}) = \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma \underline{V^\pi(\mathbf{s}')}\right]$$



Recursive Property of Value Function

$$\underbrace{V^\pi(\mathbf{s}) = \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma V^\pi(\mathbf{s}')\right]}_{\text{Bellman equation for } V^\pi}$$

Recursive Property of Value Function

$$y = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

$$y = r_0 + \gamma V^{\pi}(\mathbf{s}_1)$$

Recursive Property of Value Function

$$y = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

$$y = \underbrace{r_0}_{\substack{\text{random variable} \\ \text{high-variance}}} + \gamma \underbrace{V^\pi(\mathbf{s}_1)}_{\substack{\text{deterministic} \\ \text{low-variance}}}$$

✓ Unbiased

✓ Low-variance

Recursive Property of Value Function

$$y = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

$$y = r_0 + \gamma \boxed{V^\pi}(\mathbf{s}_1)$$

✓ Unbiased

✓ Low-variance

How do we get this?

Supervised Learning

$$\hat{V}^{\pi} = \arg\min_{V} \; \mathbb{E}_{(\mathbf{s}_i, y_i) \sim p(\mathbf{s}, y|\pi)} \left[||y_i - V(\mathbf{s}_i)||^2 \right]$$

$$\hat{V}^{\pi} \approx V^{\pi}$$

Bootstrapping

$$y = r + \gamma V^{\pi}(\mathbf{s}')$$

$$\Downarrow$$

$$y = r + \gamma \hat{V}^{\pi}(\mathbf{s}')$$

✗ Biased

✓ Low-variance

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, ..., k-1$ **do**
  $\mathcal{D} \leftarrow \emptyset$ initialize dataset
  for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
    $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
    Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
  end for

  $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, \ldots, k-1$ **do**
    $\mathcal{D} \leftarrow \emptyset$ initialize dataset
    for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
        $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
        Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
    end for

    $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, \ldots, k-1$ **do**
  $\mathcal{D} \leftarrow \emptyset$ initialize dataset
  for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
    $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
    Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
  end for

  $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, ..., k-1$ **do**
    $\mathcal{D} \leftarrow \emptyset$ initialize dataset
    for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
        $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
        Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
    end for

    $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j)\sim\mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2\right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, \ldots, k-1$ **do**
  $\mathcal{D} \leftarrow \emptyset$ initialize dataset
  for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
    $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
    Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
  end for

  $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

```
input π: policy
B ← collect trajectories τ from π
V^0 ← initialize value function

for iteration i = 0, ..., k − 1 do
    D ← ∅  initialize dataset
    for (s_j, r_j, s'_j) in B do
        y_j = r_j + γ V^i(s'_j)
        Store (s_j, y_j) in dataset D
    end for

    V^{i+1} = arg min_V E_{(s_j, y_j)~D} [||y_j − V(s_j)||^2]
end for

return V^k
```

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, ..., k-1$ **do**
  $\mathcal{D} \leftarrow \emptyset$ initialize dataset
  for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
    $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
    Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
  end for

  $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, \ldots, k-1$ **do**
    $\mathcal{D} \leftarrow \emptyset$ initialize dataset
    for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
        $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
        Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
    end for

    $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, ..., k-1$ **do**
    $\mathcal{D} \leftarrow \emptyset$ initialize dataset
    for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
        $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
        Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
    end for

    $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

input π: policy
$\mathcal{B} \leftarrow$ collect trajectories τ from π
$V^0 \leftarrow$ initialize value function

for iteration $i = 0, \ldots, k-1$ **do**
    $\mathcal{D} \leftarrow \emptyset$ initialize dataset
    for $(\mathbf{s}_j, r_j, \mathbf{s}'_j)$ in $\mathcal{B}$ **do**
        $y_j = r_j + \gamma V^i(\mathbf{s}'_j)$
        Store $(\mathbf{s}_j, y_j)$ in dataset $\mathcal{D}$
    end for

    $V^{i+1} = \arg\min_V \mathbb{E}_{(\mathbf{s}_j, y_j) \sim \mathcal{D}} \left[||y_j - V(\mathbf{s}_j)||^2 \right]$
end for

return V^k

Dynamic Programming

ALGORITHM: DP Policy Evaluation

```
input π: policy
B ← collect trajectories τ from π
V^0 ← initialize value function

for iteration i = 0, ..., k − 1 do
    D ← ∅  initialize dataset
    for (s_j, r_j, s'_j) in B do
        y_j = r_j + γ V^i(s'_j)
        Store (s_j, y_j) in dataset D
    end for

    V^{i+1} = arg min_V E_{(s_j, y_j)~D} [ ||y_j − V(s_j)||^2 ]
end for

return V^k
```

Reinforcement Learning: An Introduction 2nd Ed (Ch 6)
[Sutton and Barto 2018]

Bias-Variance Tradeoff

Monte-Carlo

$$y = \sum_{t=0}^{T-1} \gamma^t r_t$$

✓ Unbiased

✗ High-variance

Bootstrap

$$y = r_0 + \gamma \hat{V}^{\pi}(\mathbf{s}_1)$$

✗ Biased

✓ Low-variance

N-Step Bootstrapping

1-step bootstrap: $y = r_0 + \gamma \hat{V}^\pi(\mathbf{s}_1)$

2-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 \hat{V}^\pi(\mathbf{s}_2)$

3-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 \hat{V}^\pi(\mathbf{s}_3)$

$\vdots$

n-step bootstrap: $$y = \sum_{t=0}^{n-1} \gamma^t r_t + \gamma^n \underbrace{\hat{V}^\pi(\mathbf{s}_n)}_{\text{Bias}}$$

N-Step Bootstrapping

1-step bootstrap: $y = r_0 + \gamma \hat{V}^\pi(\mathbf{s}_1)$

2-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 \hat{V}^\pi(\mathbf{s}_2)$

3-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 \hat{V}^\pi(\mathbf{s}_3)$

⋮

n-step bootstrap: $y = \sum_{t=0}^{n-1} \gamma^t r_t + \underline{\gamma^n} \hat{V}^\pi(\mathbf{s}_n)$

decays exponentially

N-Step Bootstrapping

1-step bootstrap: $y = r_0 + \gamma \hat{V}^\pi(\mathbf{s}_1)$

2-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 \hat{V}^\pi(\mathbf{s}_2)$

3-step bootstrap: $y = r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 \hat{V}^\pi(\mathbf{s}_3)$

$\vdots$

n-step bootstrap: $y = \sum_{t=0}^{n-1} \gamma^t r_t + \underline{\gamma^n} \hat{V}^\pi(\mathbf{s}_n)$

$n \to \infty, \quad \gamma^n \to 0$

N-Step Bootstrapping

$$y = \sum_{t=0}^{n-1} \gamma^t r_t + \gamma^n \hat{V}^\pi(\mathbf{s}_n)$$

$n = 1$

$n \to \infty$

Bootstrap

$$y = r_0 + \gamma \hat{V}^\pi(\mathbf{s}_1)$$

Monte-Carlo

$$y = \sum_{t=0}^{T-1} \gamma^t r_t$$

N-Step Bootstrapping

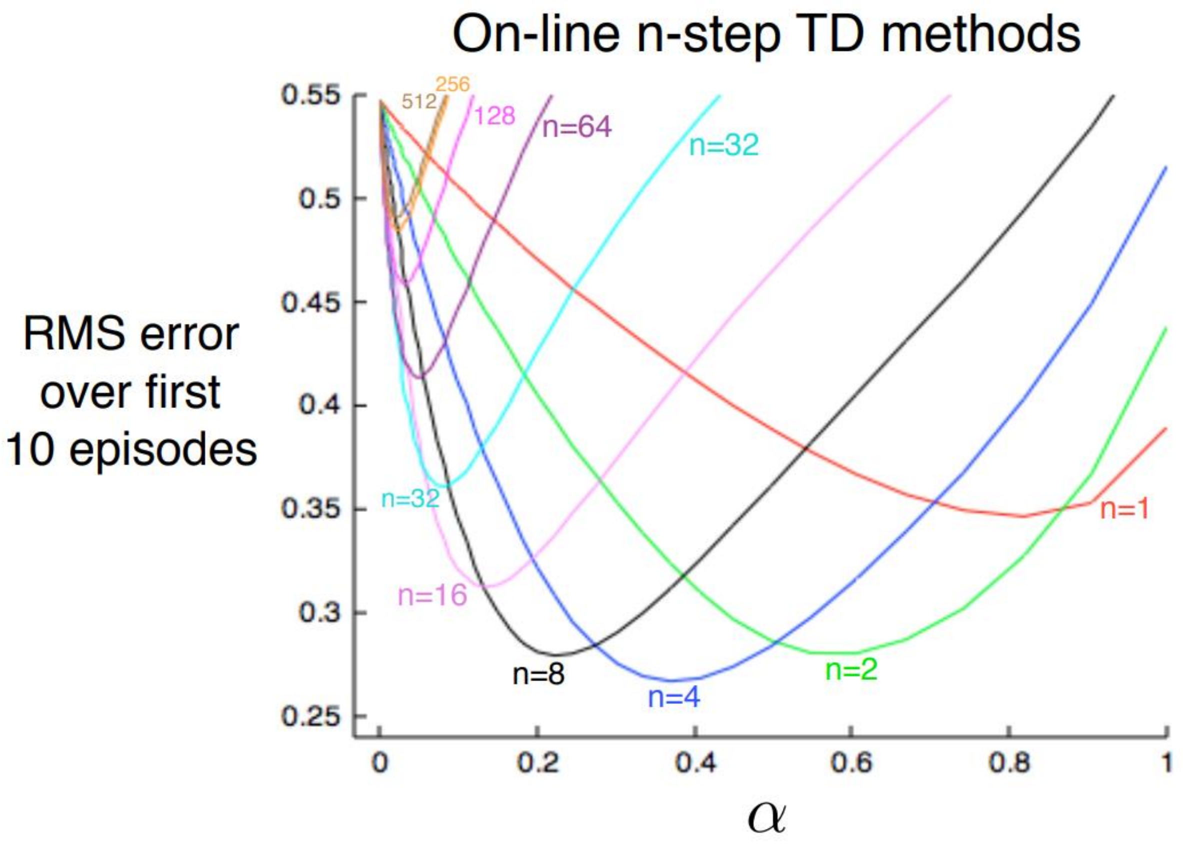
Small n :

- High bias
- Low variance

Large n :

- Low bias
- High variance

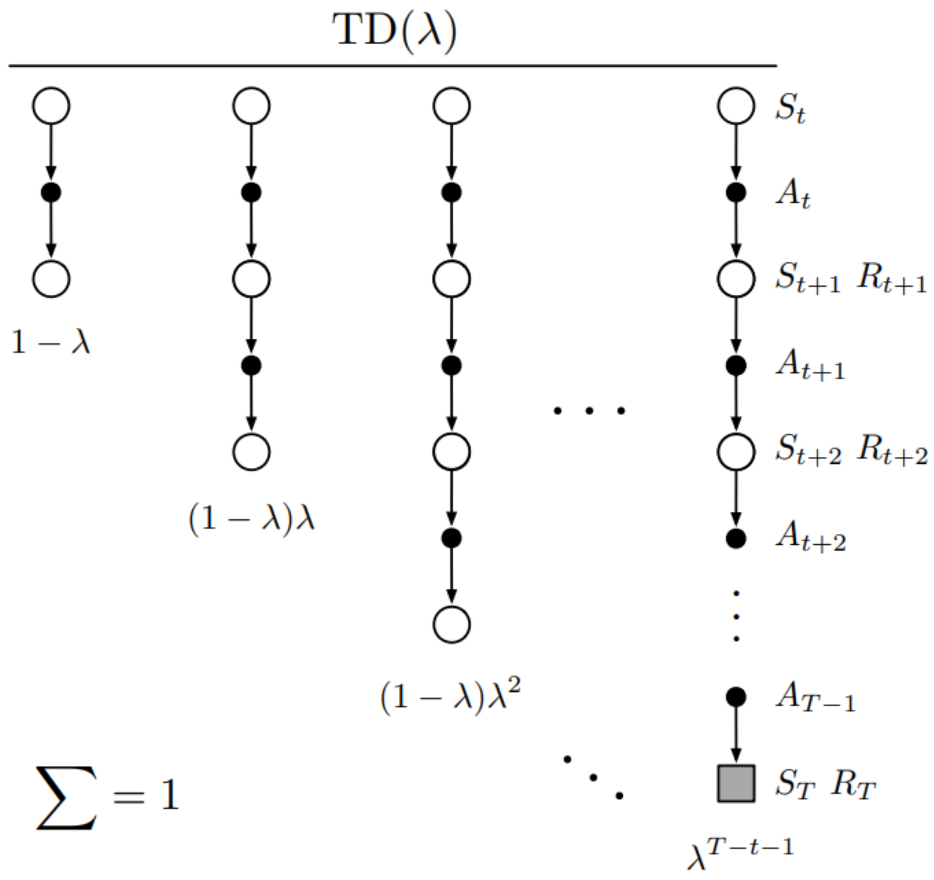
N-Step Bootstrapping



Reinforcement Learning: An Introduction 2nd Ed
[Sutton and Barto 2018]

TD(λ)

- How to we pick n?
- TD(λ):
  - Average multi-step returns across **all** lengths n!



Reinforcement Learning: An Introduction
[Sutton and Barto 2018]

Optimal Policies

Optimal Policy

$$\pi^* = \arg\max_{\pi} \; J(\pi)$$

$$\Downarrow$$

$$J(\pi^*) \geq J(\pi) \quad \text{for all } \pi$$

Optimal Value Function

$$V^*(\mathbf{s})$$

Optimal Value Function

$$V^*(\mathbf{s}) \geq V^\pi(\mathbf{s}) \quad \text{for all } \pi \text{ and } \mathbf{s}$$

$$\underbrace{V^*(\mathbf{s}) = \mathbb{E}_{\mathbf{a}\sim\pi^*(\mathbf{a}|\mathbf{s})}\mathbb{E}_{\mathbf{s}'\sim p(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left[r + \gamma V^*(\mathbf{s}')\right]}_{\text{Bellman equation of the optimal policy}}$$

Optimal Value Function

- For a given MDP
  - The optimal value function is unique
  - Can be many optimal policies
  - Given an optimal policy, can recover the optimal value function
  - Given the optimal value function, can recover an optimal policy

Testing

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \boxed{\gamma^t} r_t \right]$$

Biased towards earlier steps

Testing

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

No discount during testing

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} r_t \right]$$

Overview

- Policy Evaluation
- Value Functions
- Monte-Carlo Methods
- Dynamic Programming Methods
- Optimal Policies