loop

$\theta \leftarrow \theta + \alpha \widehat{\nabla_\theta U(\theta)}$

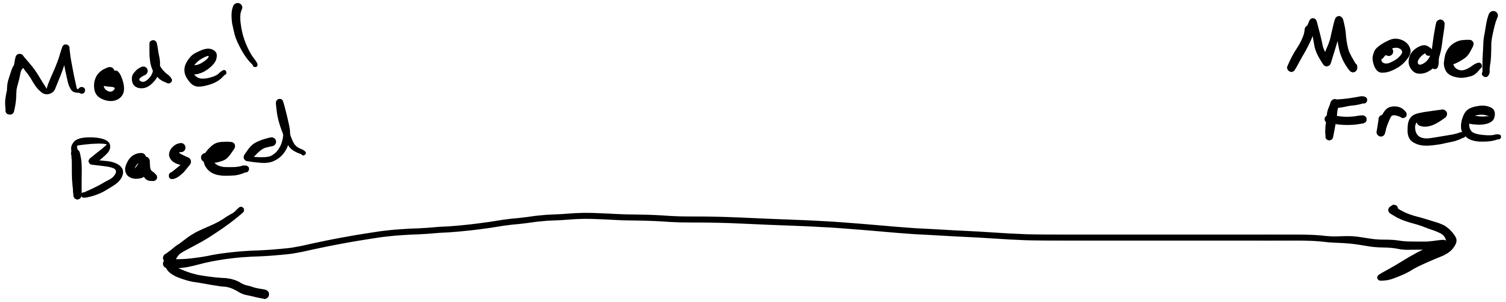
Last Time

$\pi_\theta(a|s)$

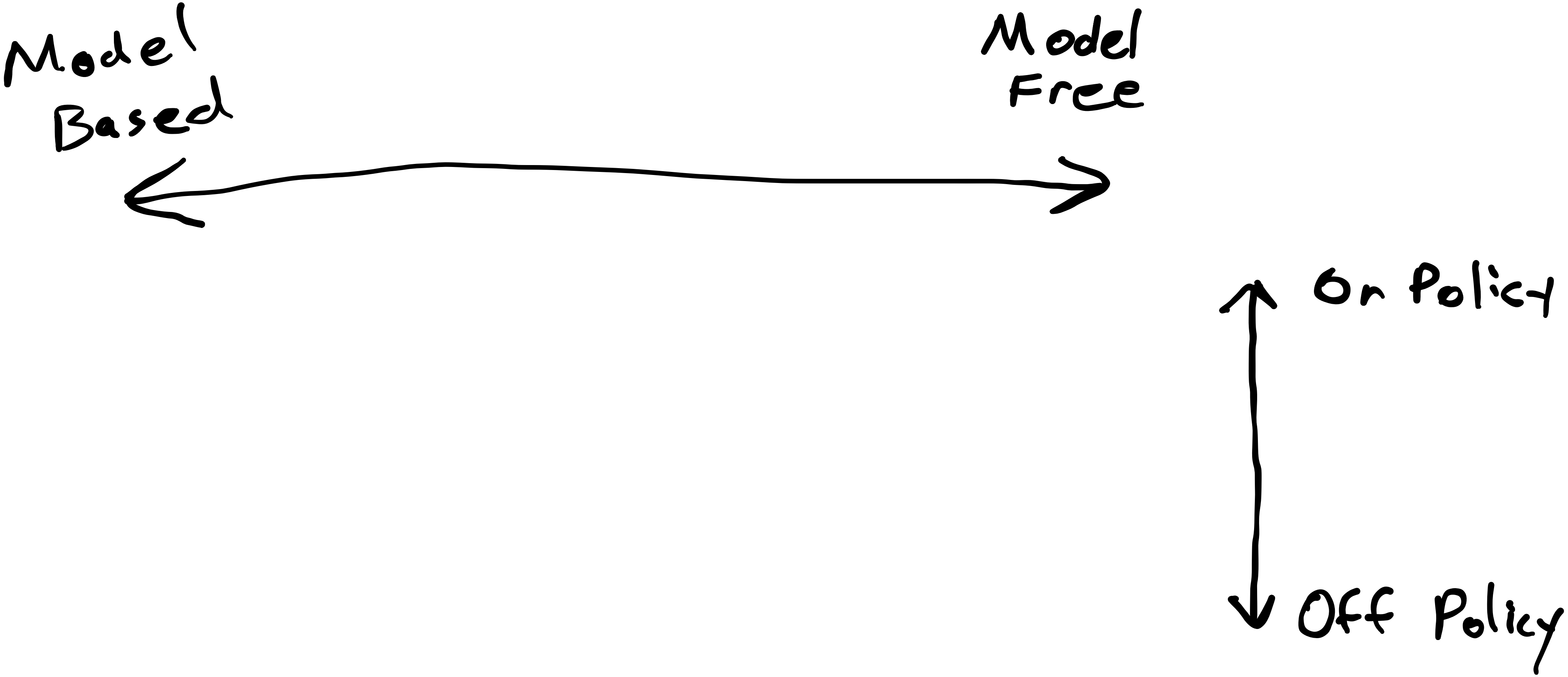
- What is Policy Gradient?
- What tricks are needed to make Policy Gradient work?

Map

Map



Map



Map

Model Based

Model Free

On Policy

Off Policy

MLMBTRL
(learn T,R)

Map

Model Based

Model Free

learn Q

learn π

On Policy

Off Policy

MLMBTRL
(learn T, R)

Map

Model Based

Model Free

learn Q

learn π

Policy Gradient

On Policy

Off Policy

MLMBTRL
(learn T,R)

Map

Model Based

Model Free

learn Q

SARSA

learn π

Policy Gradient

On Policy

Off Policy

MLMBTRL

(learn T, R)

Map

Model Based

Model Free

learn Q

SARSA

learn π

Policy Gradient

On Policy

MLMBTRL

(learn T,R)

Q-learning

Off Policy

Today

Today

- Basic On- and Off-Policy **value based** model free RL algorithms

Today

- Basic On- and Off-Policy **value based** model free RL algorithms
- Tricks for tabular value based RL algorithms

Today

- Basic On- and Off-Policy **value based** model free RL algorithms
- Tricks for tabular value based RL algorithms
- Understanding of On- vs Off-Policy

Why learn Q?

$\hat{U}(s)$ $\hat{R}(s,a)$ $\hat{Q}(s,a)$

$$Q(s,a) = R(s,a) + \gamma E[\hat{U}(s')]$$

$$\underset{a}{\operatorname{argmax}}\, Q(s,a)$$

Incremental Mean Estimation

Incremental Mean Estimation

$$\hat{x}_m = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

Incremental Mean Estimation

$$
\begin{aligned}
\hat{x}_m &= \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \\
&= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right)
\end{aligned}
$$

Incremental Mean Estimation

$$\hat{x}_m = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

$$= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right)$$

$$= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right)$$

Incremental Mean Estimation

$$
\begin{aligned}
\hat{x}_m &= \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \\
&= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right) \\
&= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right) \\
&= \hat{x}_{m-1} + \frac{1}{m}\left(x^{(m)} - \hat{x}_{m-1}\right)
\end{aligned}
$$

Incremental Mean Estimation

$$
\begin{aligned}
\hat{x}_m &= \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \\
&= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right) \\
&= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right) \\
&= \hat{x}_{m-1} + \frac{1}{m}\left(x^{(m)} - \hat{x}_{m-1}\right)
\end{aligned}
$$

```julia
function simulate!(π::MonteCarloTreeSearch, s, d=π.d)
    if d ≤ 0
        return π.U(s)
    end
    𝒫, N, Q, c = π.𝒫, π.N, π.Q, π.c
    𝒜, TR, γ = 𝒫.𝒜, 𝒫.TR, 𝒫.γ
    if !haskey(N, (s, first(𝒜)))
        for a in 𝒜
            N[(s,a)] = 0
            Q[(s,a)] = 0.0
        end
        return π.U(s)
    end
    a = explore(π, s)
    s′, r = TR(s,a)
    q = r + γ*simulate!(π, s′, d-1)
    N[(s,a)] += 1
    Q[(s,a)] += (q-Q[(s,a)])/N[(s,a)]
    return q
end
```

Incremental Mean Estimation

$$
\begin{aligned}
\hat{x}_m &= \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \\
&= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right) \\
&= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right) \\
&= \hat{x}_{m-1} + \frac{1}{m}\left(x^{(m)} - \hat{x}_{m-1}\right)
\end{aligned}
$$

```
function simulate!(π::MonteCarloTreeSearch, s, d=π.d)
    if d ≤ 0
        return π.U(s)
    end
    𝒫, N, Q, c = π.𝒫, π.N, π.Q, π.c
    𝒜, TR, γ = 𝒫.𝒜, 𝒫.TR, 𝒫.γ
    if !haskey(N, (s, first(𝒜)))
        for a in 𝒜
            N[(s,a)] = 0
            Q[(s,a)] = 0.0
        end
        return π.U(s)
    end
    a = explore(π, s)
    s′, r = TR(s,a)
    q = r + γ*simulate!(π, s′, d-1)
    N[(s,a)] += 1
    Q[(s,a)] += (q-Q[(s,a)])/N[(s,a)]
    return q
end
```

Incremental Mean Estimation

$$
\begin{aligned}
\hat{x}_m &= \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \\
&= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right) \\
&= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right) \\
&= \hat{x}_{m-1} + \frac{1}{m}\left(x^{(m)} - \hat{x}_{m-1}\right)
\end{aligned}
$$

m

```
function simulate!(π::MonteCarloTreeSearch, s, d=π.d)
    if d ≤ 0
        return π.U(s)
    end
    𝒫, N, Q, c = π.𝒫, π.N, π.Q, π.c
    𝒜, TR, γ = 𝒫.𝒜, 𝒫.TR, 𝒫.γ
    if !haskey(N, (s, first(𝒜)))
        for a in 𝒜
            N[(s,a)] = 0
            Q[(s,a)] = 0.0
        end
        return π.U(s)
    end
    a = explore(π, s)
    s′, r = TR(s,a)
    q = r + γ*simulate!(π, s′, d-1)
    N[(s,a)] += 1
    Q[(s,a)] += (q-Q[(s,a)])/N[(s,a)]
    return q
end
```

loop

$$\hat{x} \leftarrow \hat{x} + \alpha\,(x - \hat{x})$$

Incremental Mean Estimation

$$\hat{x}_m = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

$$= \frac{1}{m}\left(x^{(m)} + \sum_{i=1}^{m-1} x^{(i)}\right)$$

$$= \frac{1}{m}\left(x^{(m)} + (m-1)\,\hat{x}_{m-1}\right)$$

$$= \hat{x}_{m-1} + \frac{1}{m}\left(x^{(m)} - \hat{x}_{m-1}\right)$$

```
function simulate!(π::MonteCarloTreeSearch, s, d=π.d)
    if d ≤ 0
        return π.U(s)
    end
    𝒫, N, Q, c = π.𝒫, π.N, π.Q, π.c
    𝒜, TR, γ = 𝒫.𝒜, 𝒫.TR, 𝒫.γ
    if !haskey(N, (s, first(𝒜)))
        for a in 𝒜
            N[(s,a)] = 0
            Q[(s,a)] = 0.0
        end
        return π.U(s)
    end
    a = explore(π, s)
    s′, r = TR(s,a)
    q = r + γ*simulate!(π, s′, d-1)
    Q[(s,a)] += (q-Q[(s,a)])/N[(s,a)]
end
```

loop

$$\boxed{\hat{x} \leftarrow \hat{x} + \alpha\,(x - \hat{x})}$$

"Temporal Difference (TD) Error"

Q Learning

Want: $Q(s,a) \leftarrow Q(s,a) + \alpha\left(\hat{q}(s,a,r,s') - Q(s,a)\right)$

$$Q(s,a) = R(s,a) + \gamma\, E[V(s')]$$

$$= R(s,a) + \gamma\, E[\max_{a'} Q(s',a')]$$

$$= E[\underbrace{r + \gamma \max_{a'} Q(s',a')}_{\hat{q}}]$$

Q learning and SARSA

Q learning and SARSA

Q-Learning

$Q(s,a) \leftarrow 0$

$s \leftarrow s_0$

loop

$\quad a \leftarrow \text{argmax}\, Q(s,a)\ \text{w.p.}\ 1-\epsilon, \quad \text{rand}(A)\ \text{o.w.}$

$\quad r \leftarrow \text{act!}(\text{env}, a)$

$\quad s' \leftarrow \text{observe}(\text{env})$

$\quad Q(s,a) \leftarrow Q(s,a) + \alpha\ (r + \gamma \max_{a'} Q(s',a') - Q(s,a))$

$\quad s \leftarrow s'$

Q learning and SARSA

(s, a, r, s', a)

Q-Learning

$Q(s,a) \leftarrow 0$

$s \leftarrow s_0$

loop

$a \leftarrow \text{argmax}\, Q(s,a)$ w.p. $1-\epsilon, \quad \text{rand}(A)$ o.w.

$r \leftarrow \text{act!}(\text{env}, a)$

$s' \leftarrow \text{observe}(\text{env})$

$Q(s,a) \leftarrow Q(s,a) + \alpha\, \underbrace{(r + \gamma \max_{a'} Q(s',a') - Q(s,a))}_{\text{TD}}$

$s \leftarrow s'$

SARSA

$a \leftarrow \epsilon$-greedy

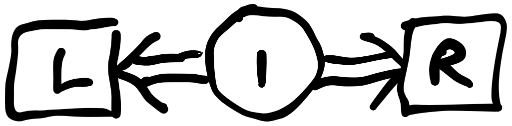
$a' \leftarrow \epsilon$ greedy

$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma Q(s',a') - Q(s,a))$

$a \leftarrow a'$

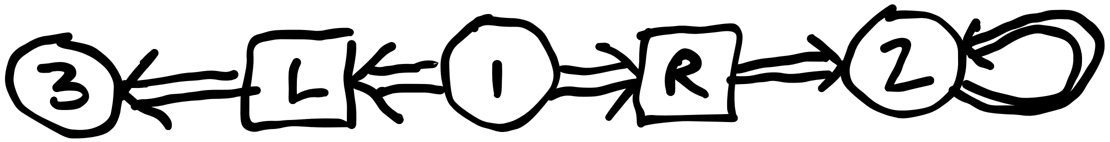
Illustrative Problem

Illustrative Problem

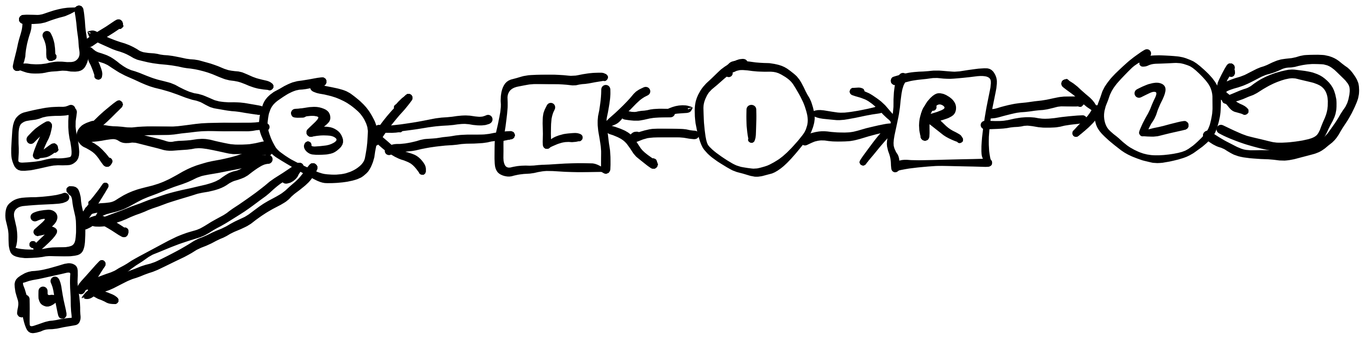


Illustrative Problem

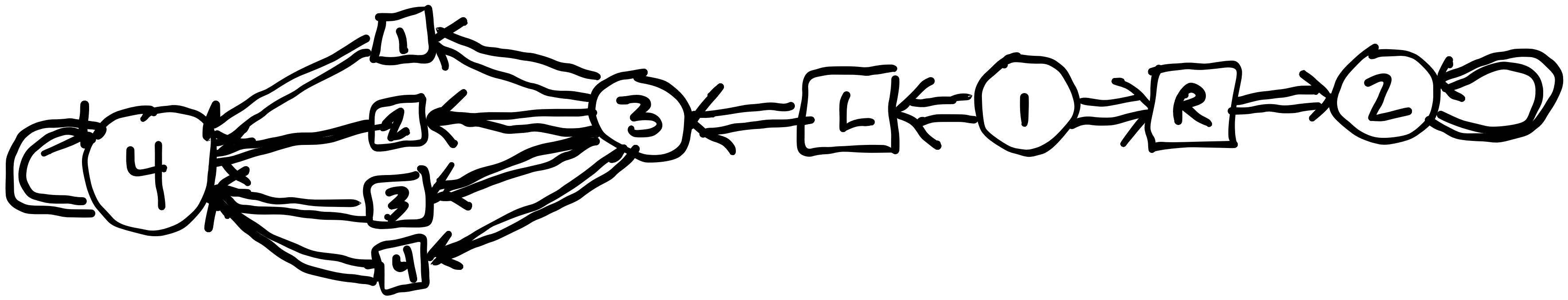
Illustrative Problem



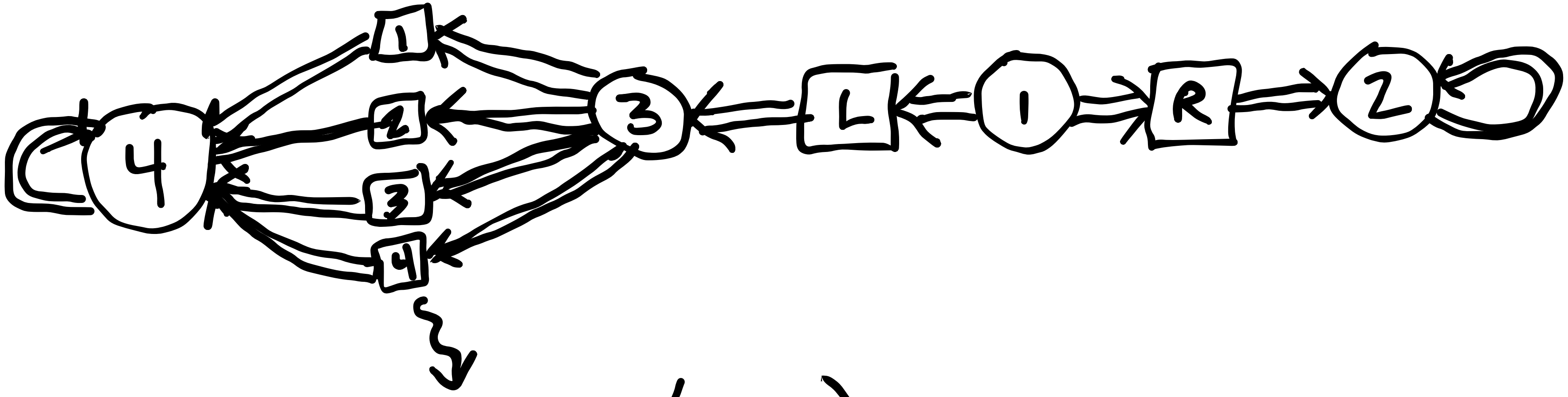
Illustrative Problem



Illustrative Problem

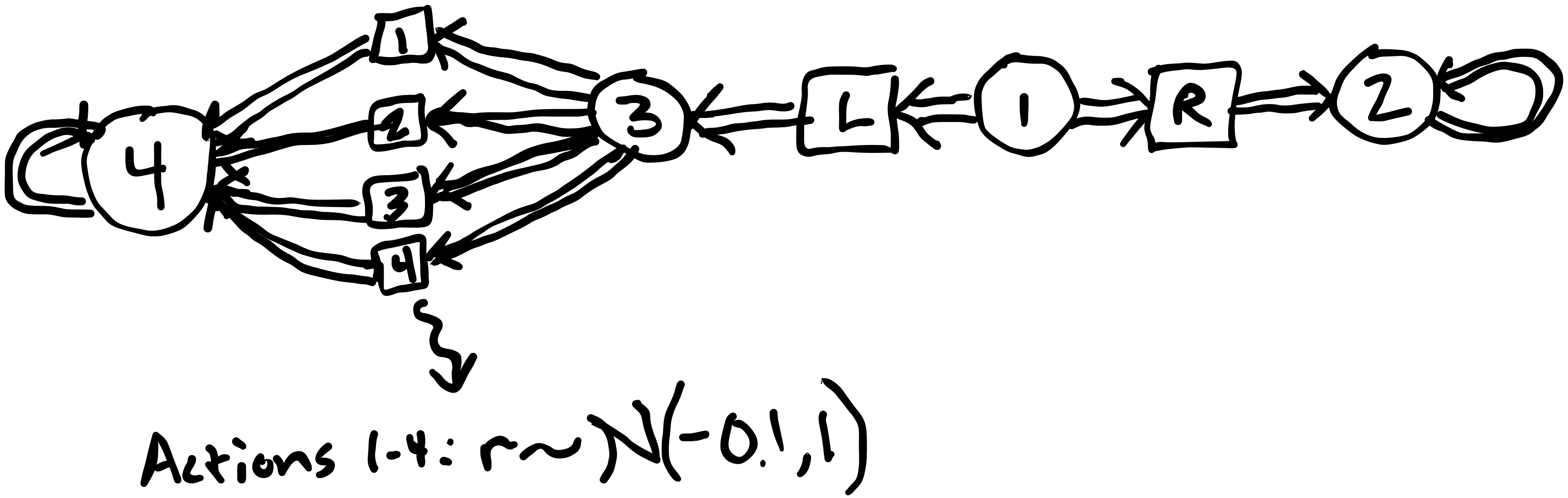


Illustrative Problem



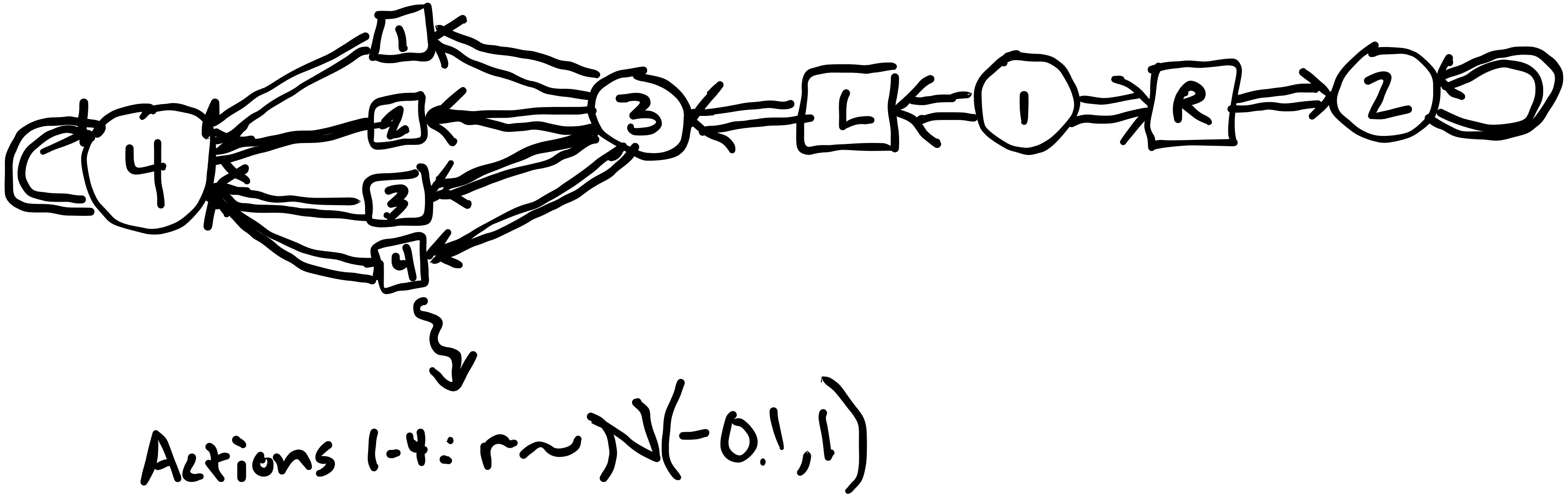
Actions 1-4: $r \sim \mathcal{N}(-0.1, 1)$

Illustrative Problem



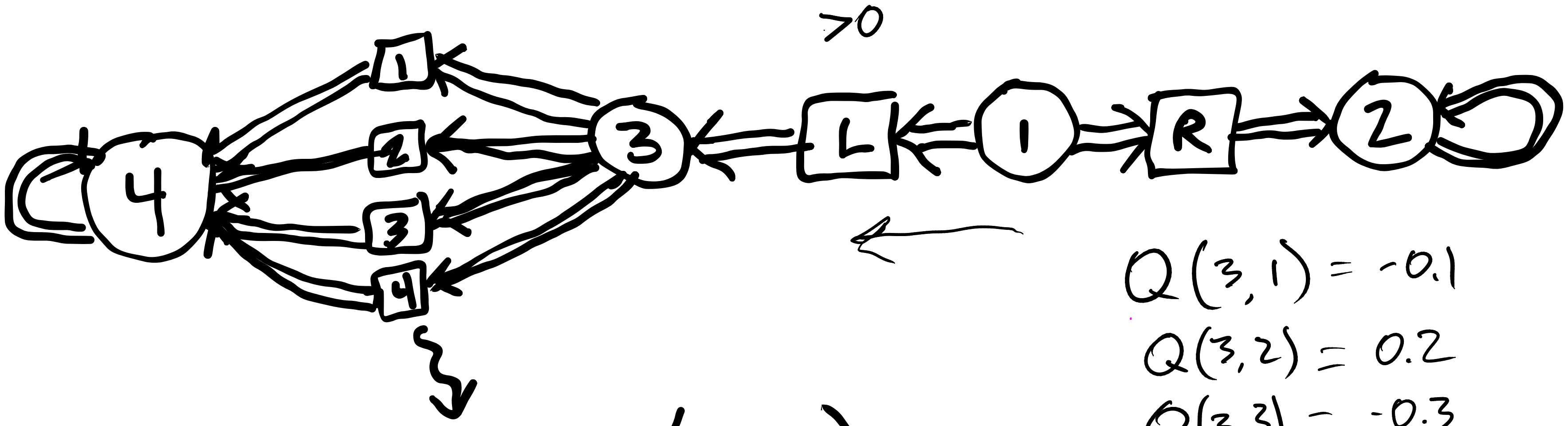
1. After a few episodes, what is Q(3, a) for a in 1-4?

Illustrative Problem



1. After a few episodes, what is Q(3, a) for a in 1-4?
2. After a few episodes, what is Q(1, L)?

Illustrative Problem



Actions 1-4: $r \sim \mathcal{N}(-0.1, 1)$

$Q(3,1) = -0.1$
$Q(3,2) = 0.2$
$Q(3,3) = -0.3$
$Q(3,4) = -0.1$

$Q(1,L) \approx \max_{a'} Q(3,a')$

≈ 0.2

$r + \gamma \max_{a'} Q(s',a')$

1. After a few episodes, what is Q(3, a) for a in 1-4?
2. After a few episodes, what is Q(1, L)?
3. Why is this a problem and what are some possible solutions?

Big Problem: Maximization Bias

Big Problem: Maximization Bias

Even if all $Q(s', a')$ unbiased, $\max_{a'} Q(s', a')$ is biased!

Big Problem: Maximization Bias

Even if all $Q(s', a')$ unbiased, $\max_{a'} Q(s', a')$ is biased!

Solution: Double Q Learning

Big Problem: Maximization Bias

Even if all $Q(s', a')$ unbiased, $\max_{a'} Q(s', a')$ is biased!

Solution: Double Q Learning Q_1, Q_2

Big Problem: Maximization Bias

Even if all $Q(s', a')$ unbiased, $\max_{a'} Q(s', a')$ is biased!

Solution: Double Q Learning Q_1, Q_2

$$\underline{Q_1}(s,a) \leftarrow \underline{Q_1}(s,a) + \alpha \left(r + \gamma \, \underline{Q_2} \left(s', \operatorname*{argmax}_{a'} \underline{Q_1}(s', a') \right) - \underline{Q_1}(s,a) \right)$$

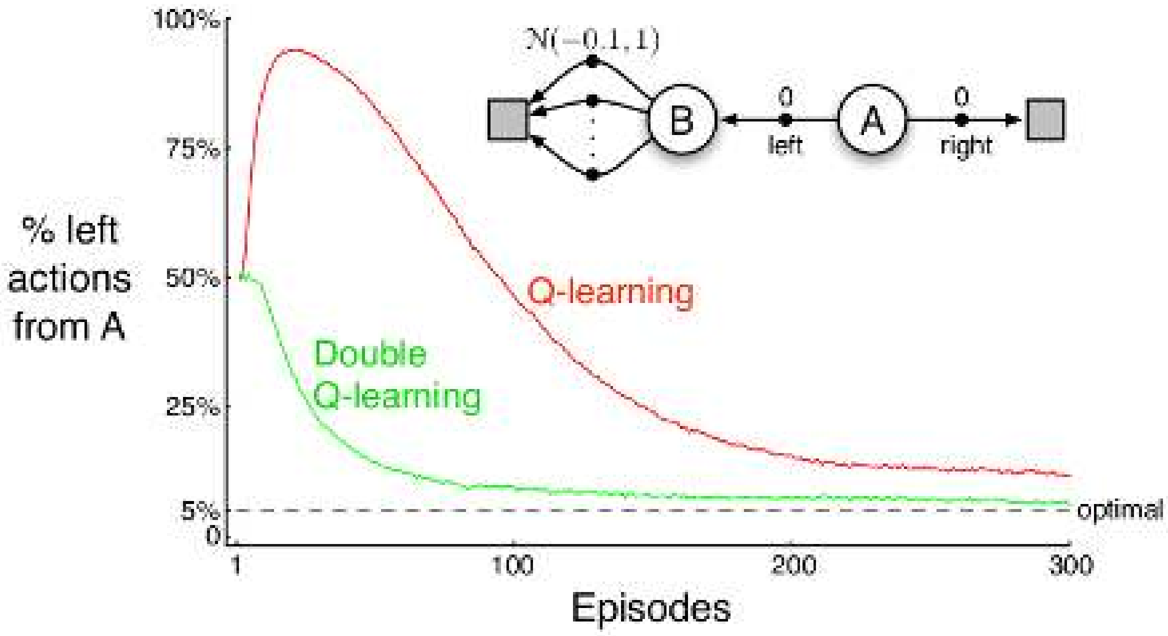
$Q_2 \leftrightarrow Q_1$

Big Problem: Maximization Bias

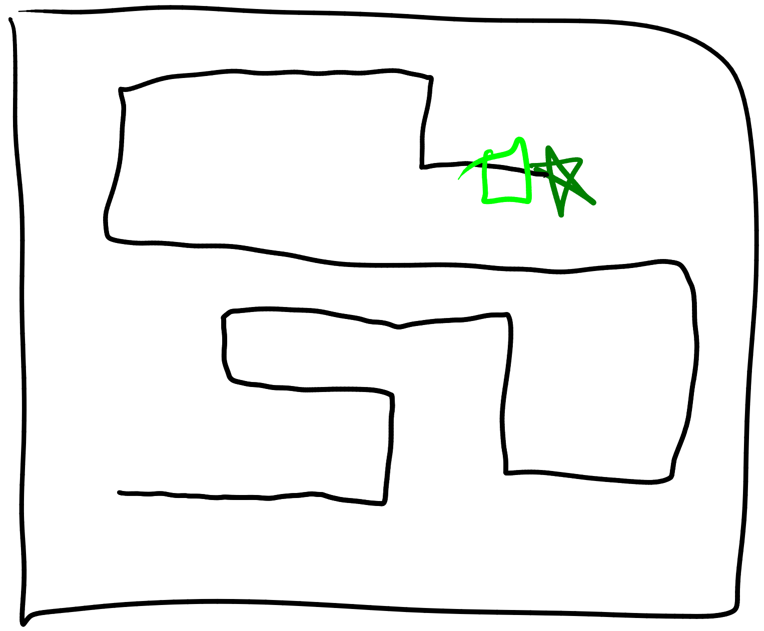
Even if all $Q(s', a')$ unbiased, $\max_{a'} Q(s', a')$ is biased!

Solution: Double Q Learning Q_1, Q_2

$$Q_1(s,a) \leftarrow Q_1(s,a) + \alpha \left(r + \gamma\, Q_2 \left(s', \underset{a'}{\operatorname{argmax}} Q_1(s', a') \right) - Q_1(s,a) \right)$$



Eligibility Traces



SARSA-λ

SARSA-λ

Games

Half-Life at 20: why it is the most important shooter ever made

From its opening scenes, Valve's pioneering sci-fi horror game reinvented storytelling and universe building - what made it such a terrifying success?

'It taught a whole generation of big-budget game developers how to tell stories' … the Half-Life box art. Illustration: Valve

SARSA-λ

Games

Half-Life at 20: why it is the most important shooter ever made

From its opening scenes, Valve's pioneering sci-fi horror game reinvented storytelling and universe building - what made it such a terrifying success?

'It taught a whole generation of big-budget game developers how to tell stories' ... the Half-Life box art. Illustration: Valve

$Q(s,a), N(s,a) \leftarrow 0$

initialize s, a, r, s'

loop

$a' \leftarrow \operatorname{argmax} Q(s', a)$ w.p. $1-\epsilon$, $\quad$ rand(A) o.w.

$N(s,a) \leftarrow N(s,a) + 1$

$\delta \leftarrow \underline{r + \gamma Q(s', a') - Q(s,a)}$ ← TD

$Q(s,a) \leftarrow Q(s,a) + \alpha\delta\, N(s,a) \quad \forall s, a$

$N(s,a) \leftarrow \gamma\lambda N(s,a)$ $\qquad \lambda \in [0,1)$

$s \leftarrow s', \quad a \leftarrow a'$

$r \leftarrow$ act!(env, a)

$s' \leftarrow$ observe(env)

Convergence

Convergence

- Q learning converges to optimal Q-values w.p. 1
  (Sutton and Barto, p. 131)

Convergence

- Q learning converges to optimal Q-values w.p. 1 (Sutton and Barto, p. 131)
- SARSA converges to optimal Q-values w.p. 1 ***provided that***

  $\pi \rightarrow \text{greedy}$

  (Sutton and Barto, p. 129)

On vs Off-Policy

On vs Off-Policy

On Policy

On vs Off-Policy

On Policy

Off Policy

On vs Off-Policy

On Policy

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \; (r + \gamma Q(s',a') - Q(s,a))$$

Off Policy

On vs Off-Policy

On Policy

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\ (r + \gamma Q(s',a') - Q(s,a))$$

Off Policy

Q-learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\ (r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

On vs Off-Policy

On Policy

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma Q(s',a') - Q(s,a))$$

Off Policy

Q-learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

Will eligibility traces work with Q-learning?

On vs Off-Policy

On Policy

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma Q(s',a') - Q(s,a))$$

Off Policy

Q-learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

Will eligibility traces work with Q-learning?

Not easily

On vs Off-Policy

On Policy

< more stable
more data

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma Q(s',a') - Q(s,a))$$

Off Policy

less stable
less data

Q-learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \ (r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

Will eligibility traces work with Q-learning?

Not easily

Policy Gradient:

$$\theta \leftarrow \theta + \alpha \sum_{k=0}^{d} \nabla_\theta \log \pi_\theta(a_k \mid s_k) R(\tau)$$

Today

SARSA Q-Learning

- Basic On- and Off-Policy **value based** model free RL algorithms
- Tricks for tabular value based RL algorithms ← Double Q Learning
  Eligibility Traces
- Understanding of On- vs Off-Policy