MDP and RL

CS221: Section 4

Policy Iteration vs Value Iteration

SARSA vs Q-Learning

Function Approximation and Q-Learning

Policy Iteration vs Value Iteration

SARSA vs Q-Learning

Function Approximation and Q-Learning

Markov Decision Process (MDP)

$\boldsymbol{s} \in States$

$\boldsymbol{Actions}(\boldsymbol{s})$: $actions\ from\ state\ s$

$\boldsymbol{T}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}')$: $probability\ next\ state\ is\ s', given\ action\ a\ in\ state\ s$

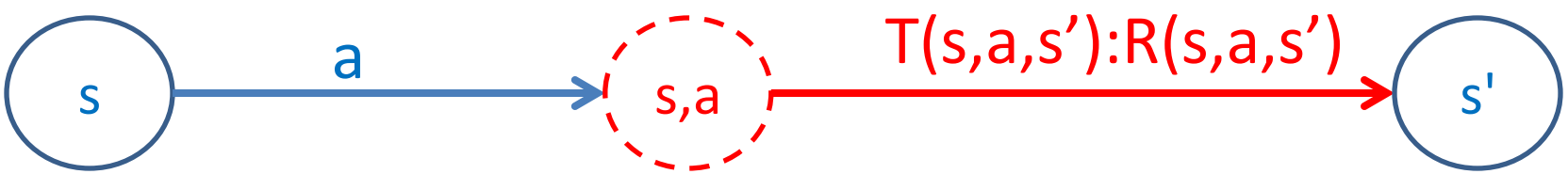
$$Note\ that \sum_{s'} T(s, a, s') = 1\ for\ all\ s \in States, a \in Actions(s)$$

$\boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}')$: $reward\ for\ transition\ (s, a, s')$

$\boldsymbol{IsEnd}(\boldsymbol{s})$: $whether\ state\ marks\ end\ of\ game$

$Alternatively, T(s, a, s) = 1\ and\ R(s, a, s) = 0$

$\boldsymbol{0 \leq \gamma \leq 1}$: $discount\ factor$



Markov Decision Process (MDP)

Given MDP

$\boldsymbol{\pi}: policy$

$take\ action\ \pi(s) in\ state\ s$

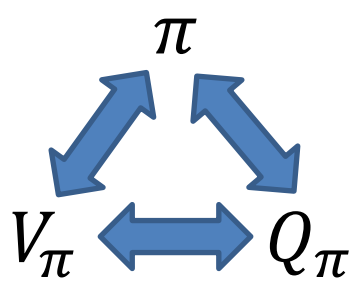
$\boldsymbol{V_\pi}: value\ function\ for\ policy\ \pi$

$V_\pi(s)\ is\ expected\ utility\ of\ following\ \pi\ starting\ in\ s$

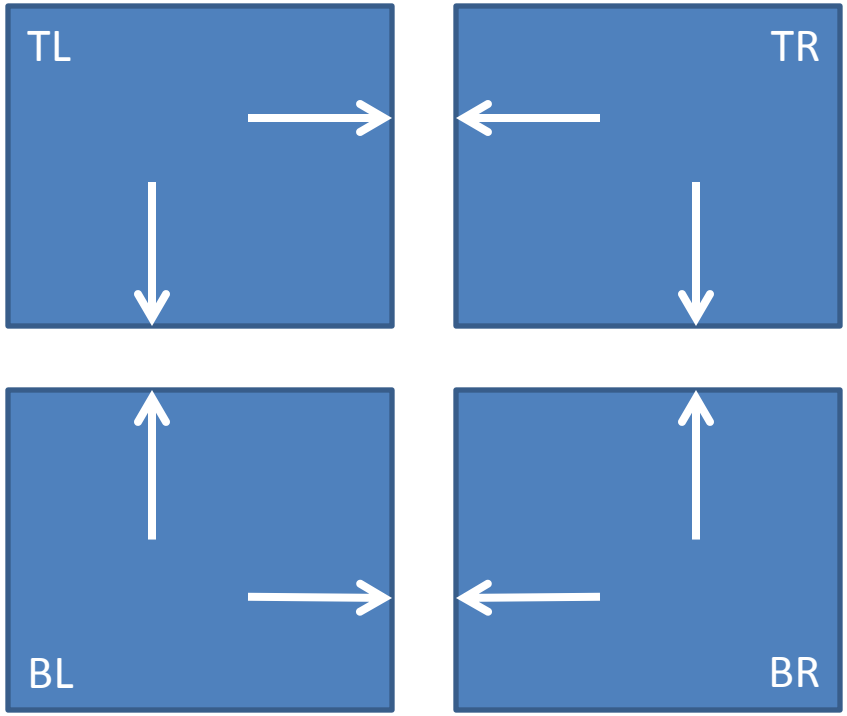
$\boldsymbol{Q_\pi}: Q - value\ of\ policy\ \pi$

$Q_\pi(s, a)\ is\ expected\ utility\ of\ taking\ action\ a\ in\ s, then\ following\ \pi$

Want to find optimal policy (that maximizes expected sum of discounted rewards)



Running Example



$States = \{TL, TR, NL, BR\}$
$EndState = BR$

$Actions(s) = \{H, V\}, \forall s$

$$\mathrm{T}(s,a,s') = \begin{cases} s'(s,a), & w.p.\ 3/4 \\ s'(s,a^c), & w.p.\ 1/4 \end{cases}$$
$except\ T(BR, a, BR) = 1$

$Reward(s,a,s') = 4\ if\ s' = BR\ else\ -1$

Simple $policy\ \pi(s) = H, \forall s$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π $\quad \pi \Rightarrow V_\pi$

2. **Policy improvement**: given V_π , compute improved policy π' $\quad V_\pi \Rightarrow \pi'$

3. **Repeat** until policy converges/does not change

$$\pi \Rightarrow V_\pi \Rightarrow \pi' \Rightarrow V_{\pi'} \Rightarrow \pi''$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π
   a) Choose any V

$$\pi \Rightarrow V_\pi$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π
   a) Choose any V
   b) Update

$$V(s) \coloneqq \sum_{s'} T(s, \pi(s), s')\,[\,R(s, \pi(s), s') + \gamma V(\,s')], \qquad \forall s$$

$$\pi \Rightarrow V_\pi$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π
   a) Choose any V
   b) Update

$$\pi \Rightarrow V_\pi$$

$$V(s) \coloneqq \sum_{s'} T(s, \pi(s), s')\,[\,R(s, \pi(s), s') + \gamma V(\,s')], \qquad \forall s$$

or, $$V \coloneqq F_\pi V \quad (compact)$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π
   a) Choose any V
   b) Update $V \coloneqq F_\pi V$

$$\pi \Rightarrow V_\pi$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute V_π
   a) Choose any V
   b) Update $V \coloneqq F_\pi V$
   c) Repeat b) until convergence to obtain V_π
      We know $V_\pi = F_\pi F_\pi \dots F_\pi V = F_\pi^\infty V$
      Practically, update t_{PE} times to get reasonable approximation

$$\pi \Rightarrow V_\pi$$

Policy Evaluation Example

V_0

TL 0	TR 0
0 BL	0 BR

$\text{Simple policy } \pi(s) = H, \forall s$
$Evaluate\ V_\pi$
$V_0(s) = 0, \forall s$

Policy Evaluation Example

V_0

TL 0	TR 0
0 BL	0 BR

V_1

TL -	TR -
- BL	- BR

$$\text{Simple policy } \pi(s) = H, \forall s$$
$$Evaluate\ V_\pi$$
$$V_0(s) = 0, \forall s$$

$$\begin{aligned} V_1(TL) &= \frac{3}{4} \times [-1 + V_0(TR)] + \frac{1}{4} \times [-1 + V_1(BL)] \\ &= \frac{3}{4} \times [-1 + \quad 0 \quad] + \frac{1}{4} \times [-1 + \quad 0 \quad] \\ &= -1 \end{aligned}$$

Policy Evaluation Example

V_0

TL 0	0 TR
0 BL	0 BR

V_1

TL -1	- TR
- BL	- BR

$$\text{Simple policy } \pi(s) = H, \forall s$$
$$Evaluate\ V_\pi$$
$$V_0(s) = 0, \forall s$$

$$V_1(TR) = \frac{3}{4} \times [-1 + V_0(TL)] + \frac{1}{4} \times [4 + V_0(BR)]$$
$$= \frac{3}{4} \times [-1 + \quad 0 \quad] + \frac{1}{4} \times [4 + \quad 0 \quad]$$
$$= \frac{1}{4}$$

Policy Evaluation Example

V_0

TL 0	TR 0
BL 0	BR 0

V_1

TL -1	TR 1/4
BL 1/4	BR 0

Policy Evaluation Example

V_0

TL 0	0 TR
0 BL	0 BR

V_1

TL -1	1/4 TR
1/4 BL	0 BR

V_2

TL -1/8	-1/2 TR
5/2 BL	0 BR

…

V_π obtained

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'

$$V_\pi \Rightarrow \pi'$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'
   a) Compute

$$V_\pi \Rightarrow \pi'$$

$$Q_\pi(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s, a$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'
   a) Compute

$$V_\pi \Rightarrow \pi'$$

$$Q_\pi(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s, a$$

   b) Compute $\pi'(s) = \arg\max_{a \in Actions(s)} Q_\pi(s,a), \forall s$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

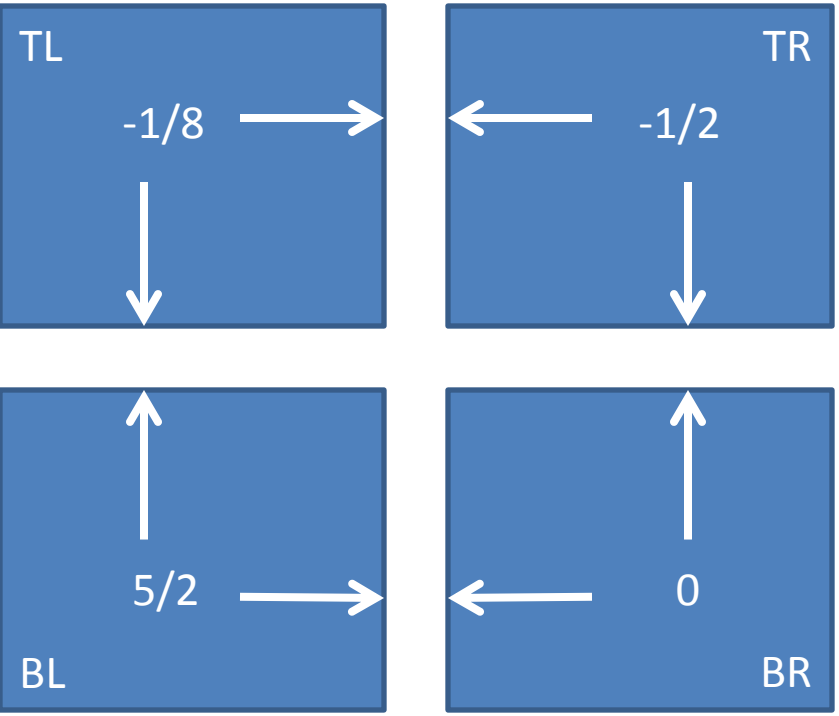
$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'

$$V_\pi \Rightarrow \pi'$$

$$\pi'(s) = \arg\max_{a \in Actions(s)} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s$$

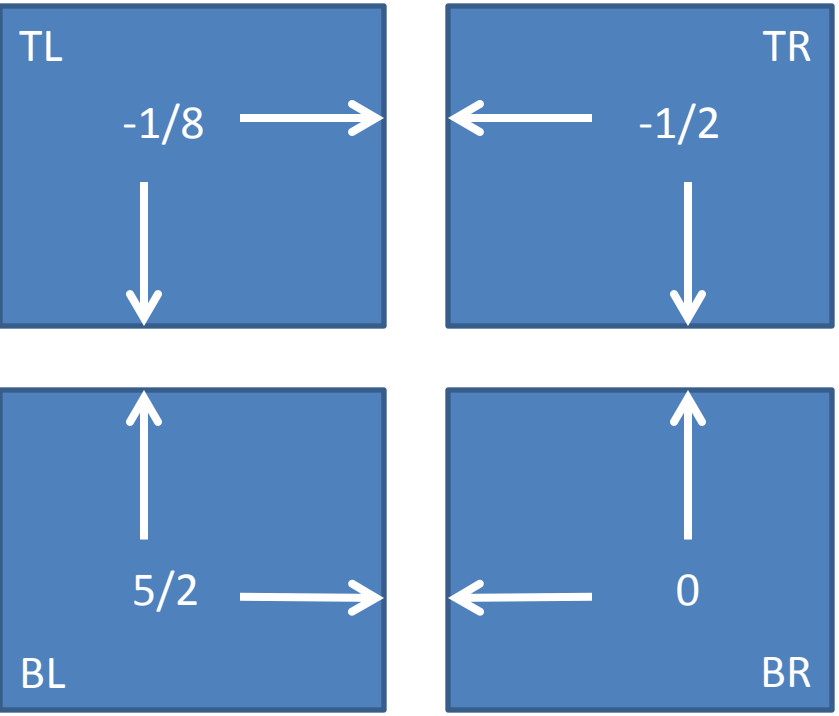
Policy Improvement Example



$Given\ V_\pi, evaluate\ \pi'\ (improved\ policy)$

$Find\ Q_\pi(s,a) for\ all\ states, choose\ best\ action$

Policy Improvement Example

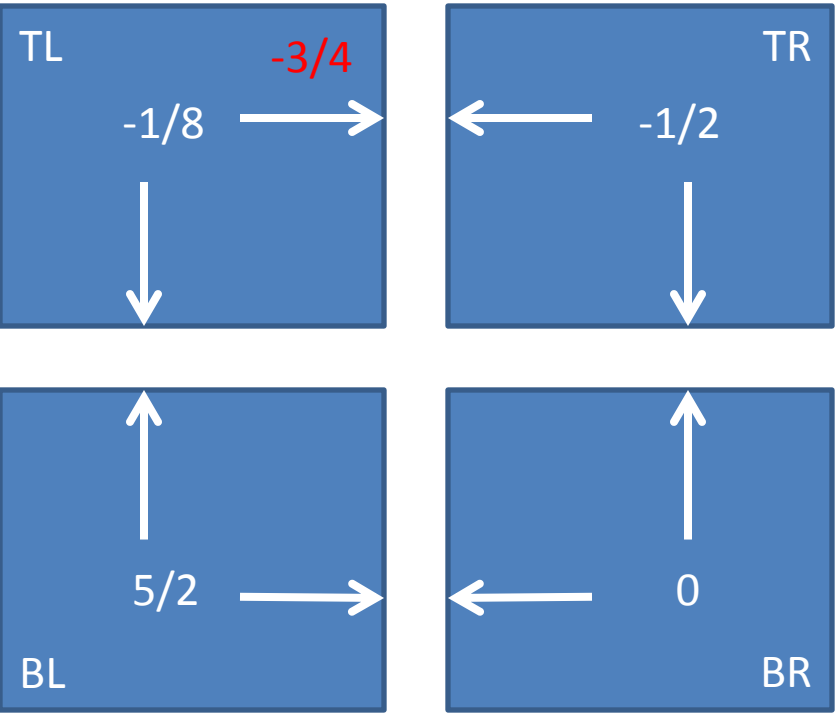


$Given\ V_\pi, evaluate\ \pi'\ (improved\ policy)$

$Find\ Q_\pi(s,a) for\ all\ states, choose\ best\ action$

$$Q_\pi(TL,H) = \frac{3}{4} \times \left[-1 + \frac{-1}{2}\right] + \frac{1}{4} \times \left[-1 + \frac{5}{2}\right] = \frac{-3}{4}$$

Policy Improvement Example

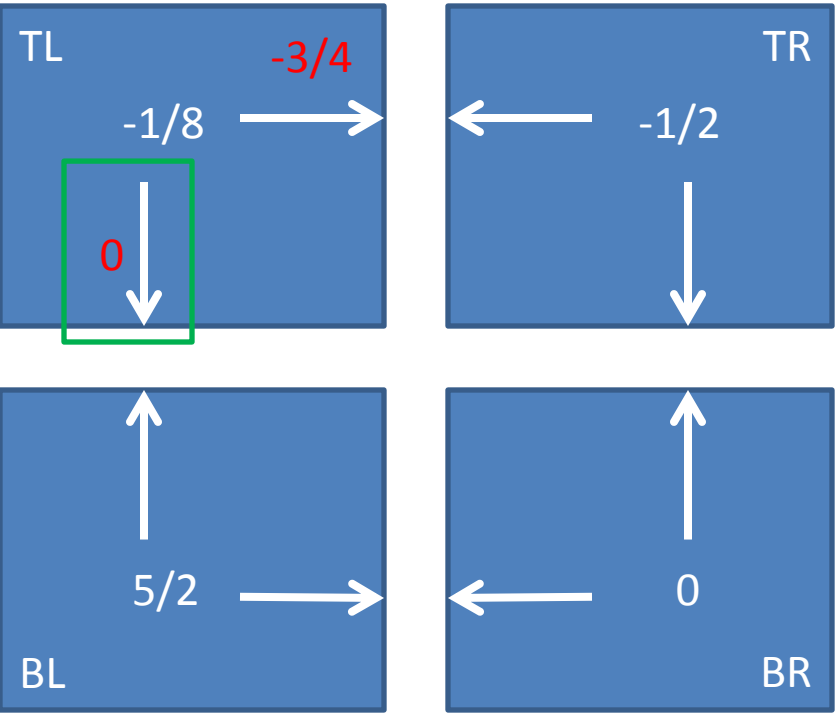


$Given\ V_\pi, evaluate\ \pi'\ (improved\ policy)$

$Find\ Q_\pi(s,a) for\ all\ states, choose\ best\ action$

$$Q_\pi(TL,V) = \frac{1}{4} \times \left[-1 + \frac{-1}{2}\right] + \frac{3}{4} \times \left[-1 + \frac{5}{2}\right] = 0$$

Policy Improvement Example

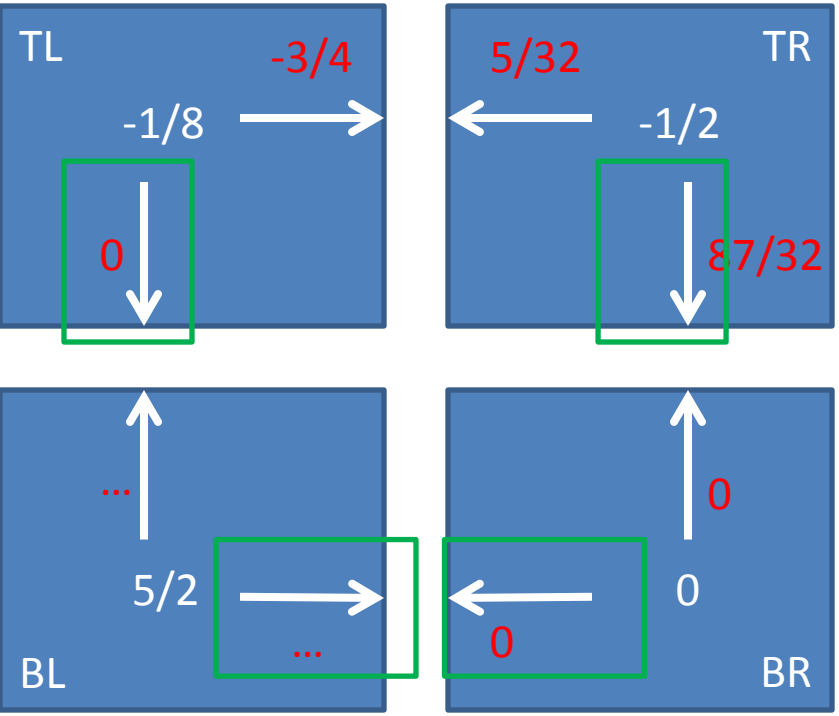


$Given\ V_\pi, evaluate\ \pi'\ (improved\ policy)$

$Find\ Q_\pi(s,a) for\ all\ states, choose\ best\ action$

$Optimal\ action$
$\pi'(TL) = V$

Policy Improvement Example



$Given\ V_\pi, evaluate\ \pi'\ (improved\ policy)$

$Find\ Q_\pi(s,a) for\ all\ states, choose\ best\ action$

$Optimal\ action$
$\pi'(TL) = V$
$\pi'(TR) = V$
$\pi'(BL) = \mathrm{H}$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'

$$V_\pi \Rightarrow \pi'$$

$$\pi'(s) = \arg\max_{a \in Actions(s)} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s$$

3. **Repeat** until policy converges/does not change

$$\pi \Rightarrow V_\pi \Rightarrow \pi' \Rightarrow V_{\pi'} \Rightarrow \pi''$$

Policy Iteration Example

π_0

→	→
→	→

V_0

TL 0	0 TR
0 BL	0 BR

V_1

TL -1	1/4 TR
1/4 BL	0 BR

V_2

TL -1/8	-1/2 TR
5/2 BL	0 BR

...

π_1

↓	↓
→	→

V_0

TL 0	0 TR
0 BL	0 BR

...

Policy Iteration Example

π_0

→	→
→	→

V_0

TL 0	TR 0
0 BL	0 BR

V_1

TL -1	TR 1/4
1/4 BL	0 BR

V_2

TL -1/8	TR -1/2
5/2 BL	0 BR

…

π_1

↓	↓
→	→

V_0

TL -1/8	TR -1/2
5/2 BL	0 BR

…

Warm start!

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'

$$V_\pi \Rightarrow \pi'$$

$$\pi'(s) = \arg\max_{a \in Actions(s)} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s$$

3. **Repeat** until policy converges/does not change

$$\pi \Rightarrow V_\pi = F_\pi^\infty V \Rightarrow \pi' \Rightarrow V_{\pi'} = F_\pi^\infty V \Rightarrow \pi''$$

Policy Iteration

Start with any π

1. **Policy evaluation**: given π, compute $V_\pi = F_\pi^\infty V$

$$\pi \Rightarrow V_\pi$$

2. **Policy improvement**: given V_π , compute improved policy π'

$$V_\pi \Rightarrow \pi'$$

$$\pi'(s) = \arg\max_{a \in Actions(s)} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_\pi(s')], \qquad \forall s$$

3. **Repeat** until policy converges/does not change

$$\pi \Rightarrow V_\pi = F_\pi^\infty V \Rightarrow \pi' \Rightarrow V_{\pi'} = F_\pi^\infty V_\pi \Rightarrow \pi''$$

Warm start!

Value Iteration

Start with any V

1. **Compute Q-values**: given V, compute Q

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

Value Iteration

Start with any V

1. **Compute Q-values**: given V, compute Q

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

2. **Compute V':** given Q, compute V'

$$V'(s) = \max_{a \in Actions(s)} Q(s,a)\,, \qquad \forall s$$

Value Iteration

Start with any V

1. **Compute Q-values**: given V, compute Q

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

2. **Compute V':** given Q, compute V'

$$V'(s) = \max_{a \in Actions(s)} Q(s,a)\,, \qquad \forall s$$

3. **Repeat** until value function converges

Value Iteration

Start with any V

1. **Compute V'**: given V, compute V'

$$V'(s) = \max_{a \in Actions(s)} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')], \qquad \forall s$$

2. **Repeat** until value function converges

Value Iteration

Start with any V

1. **Compute V'**: given V, compute π', then compute V'
   a) Compute $\pi'(s) = \underset{a \in Actions(s)}{\arg\max} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')]\,,\ \forall s$
   b) Compute $V' = F_{\pi'} V$
2. **Repeat** until value function converges

Value Iteration

Start with any V

1. **Compute V'**: given V, compute π', then compute V'
   a) Compute $\pi'(s) = \underset{a \in Actions(s)}{\arg\max} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V(s')], \ \forall s$
   b) Compute $V' = F_{\pi'} V$
2. **Repeat** until value function converges

$$V \Rightarrow \pi' \Rightarrow V' = F_{\pi'} V \Rightarrow \pi'' \Rightarrow V'' = F_{\pi''} V'$$

Policy Iteration

π_0

→	→
→	→

V_0

TL 0	TR 0
BL 0	BR 0

V_1

TL -1	TR 1/4
BL 1/4	BR 0

V_2

TL -1/8	TR -1/2
BL 5/2	BR 0

…

Precise estimate of the value of policies!

π_1

↓	↓
→	→

V_0

TL -1/8	TR -1/2
BL 5/2	BR 0

V_1

TL ..	TR ..
BL ..	BR ..

V_2

TL ..	TR ..
BL ..	BR ..

…

Value Iteration

V_0

TL 0	TR 0
BL 0	BR 0

π_0

↓	↓
→	→

V_0

TL 0	TR 0
BL 0	BR 0

V_1

TL -1	TR 1/4
BL 1/4	BR 0

π_1

↓	↓
→	→

V_1

TL -1	TR 1/4
BL 1/4	BR 0

V_2

TL ..	TR ..
BL ..	BR ..

Crude estimate of the value of policies!

Value Iteration

V_0

TL 0	TR 0
BL 0	BR 0

π_0

↓	↓
→	→

V_0

TL 0	TR 0
BL 0	BR 0

V_1

TL -1	TR 1/4
BL 1/4	BR 0

π_1

↓	↓
→	→

V_1

TL -1	TR 1/4
BL 1/4	BR 0

V_2

TL ..	TR ..
BL ..	BR ..

Crude estimate of the value of policies!

Exact numbers on slide may not be correct

Policy Iteration vs Value Iteration

Policy Iteration

$$\pi \Rightarrow V_\pi = F_\pi^\infty V \Rightarrow \pi' \Rightarrow V_{\pi'} = F_\pi^\infty V$$

$$V \Rightarrow \pi \Rightarrow V' = F_\pi V \Rightarrow \pi' \Rightarrow V'' = F_{\pi'} V'$$

Value Iteration

Policy Iteration vs Value Iteration

Policy Iteration

Precise Policy Evaluation + Policy Improvement

$$\pi \Rightarrow V_\pi = F_\pi^\infty V \Rightarrow \pi' \Rightarrow V_{\pi'} = F_\pi^\infty V$$

$$V \Rightarrow \pi \Rightarrow V' = F_\pi V \Rightarrow \pi' \Rightarrow V'' = F_{\pi'} V'$$

Crude Policy Evaluation + Policy Improvement

Value Iteration

Policy Iteration vs Value Iteration

SARSA vs Q-Learning

Function Approximation and Q-Learning

Reinforcement Learning (RL)

$\boldsymbol{s} \in States$

$\boldsymbol{Actions}(\boldsymbol{s})$: $actions\ from\ state\ s$

$\boldsymbol{T}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}')$: $probability\ next\ state\ is\ s', given\ action\ a\ in\ state\ s$

$Note\ that \sum_{s'} T(s, a, s') = 1\ for\ all\ s \in States, a \in Actions(s)$

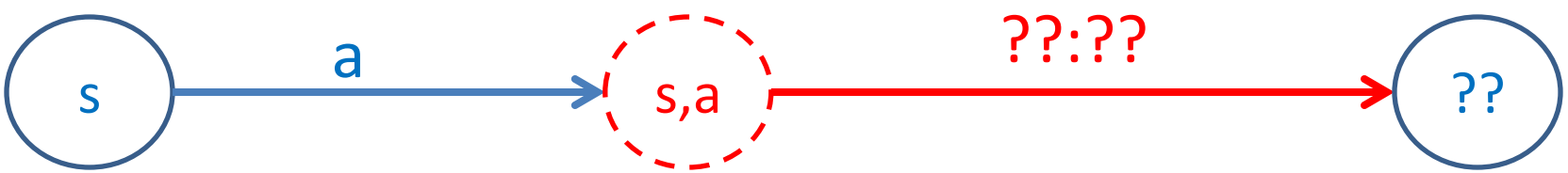
UNKNOWN

$\boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}')$: $reward\ for\ transition\ (s, a, s')$

$\boldsymbol{IsEnd}(\boldsymbol{s})$: $whether\ state\ marks\ end\ of\ game$

$Alternatively, T(s, a, s) = 1\ and\ R(s, a, s) = 0$

$\boldsymbol{0 \leq \gamma \leq 1}$: $discount\ factor$



Reinforcement Learning (RL)

Algorithm	Estimate	Comments
Model-based Monte Carlo	$T(s,a,s'), R(s,a,s')$	Find Q_{opt}, π_{opt} afterward
Model-free Monte Carlo	Q_π using utility u	No optimal policy (on-policy)
SARSA	Q_π using utility r, Q_π	No optimal policy (on-policy)
Q-Learning	Q_{opt} using utility r, Q_{opt}	Optimal policy (off-policy)

SARSA vs Q-Learning

SARSA: Update $\hat{Q}_\pi(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta\left[\, r + \gamma \hat{Q}_\pi(s',a') \,\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

SARSA evaluates policy π. Policy Improvement comes next. Iterate.

SARSA vs Q-Learning

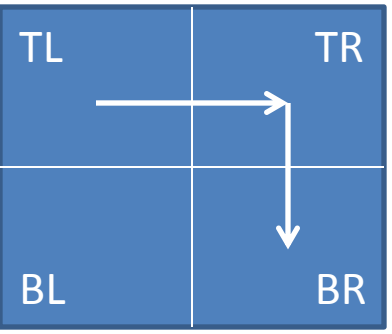
SARSA: Update $\hat{Q}_\pi(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_\pi(s,a) \leftarrow \hat{Q}_\pi(s,a) - \eta\left[\hat{Q}_\pi(s,a) - (r + \gamma \hat{Q}_\pi(s',a'))\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

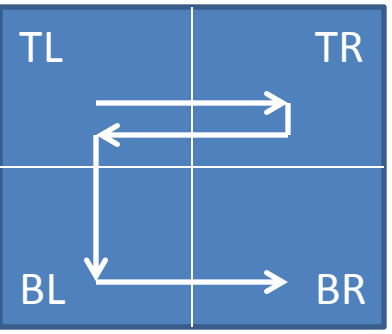
SARSA evaluates policy π. Policy Improvement comes next. Iterate.

SARSA Example



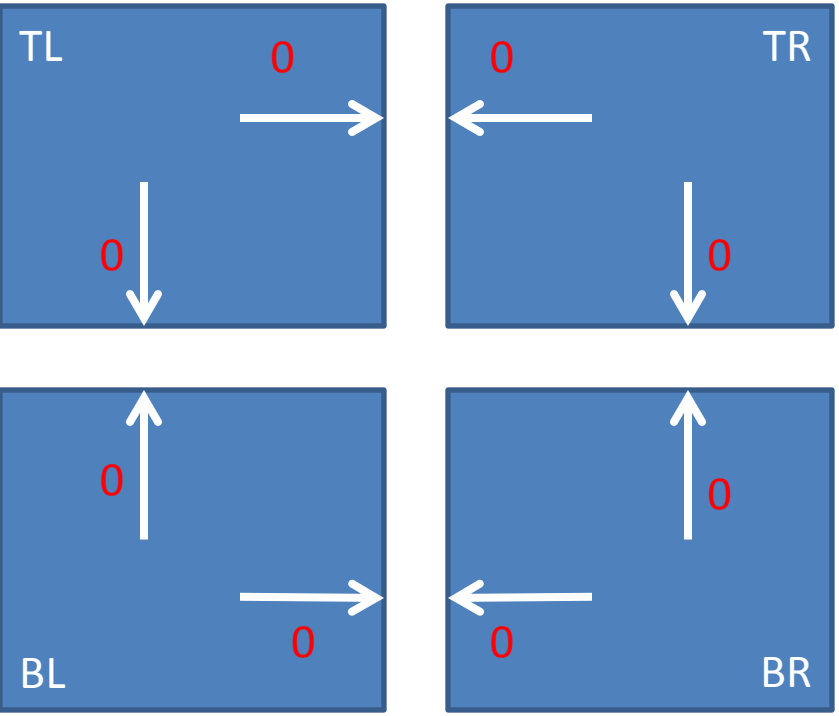
$Fixed\ policy\ \pi(s) = H, \forall s$
$Generate\ sample\ paths$
$1. TL; H, -1, TR; H, 4, BR$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$
$Generate\ sample\ paths$
$1. TL; H, -1, TR; H, 4, BR$
$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

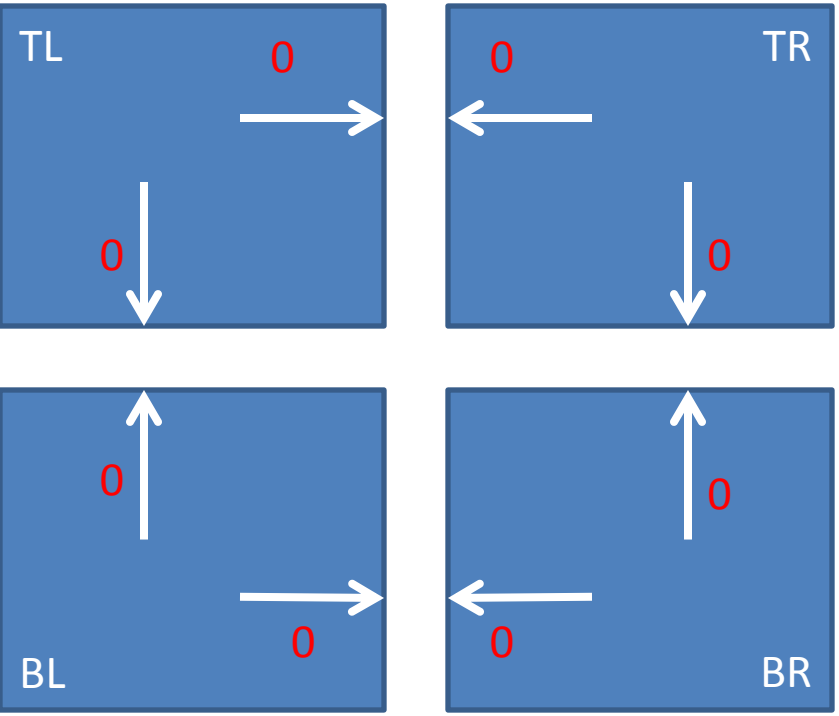
$1. TL; H, -1, TR; H, 4, BR$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{SARSA}$

$Set\ Q_\pi(s, a) = 0$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1.\ \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; \boldsymbol{H}, 4, BR$

$2.\ TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{SARSA}$

$Set\ Q_\pi(s, a) = 0$

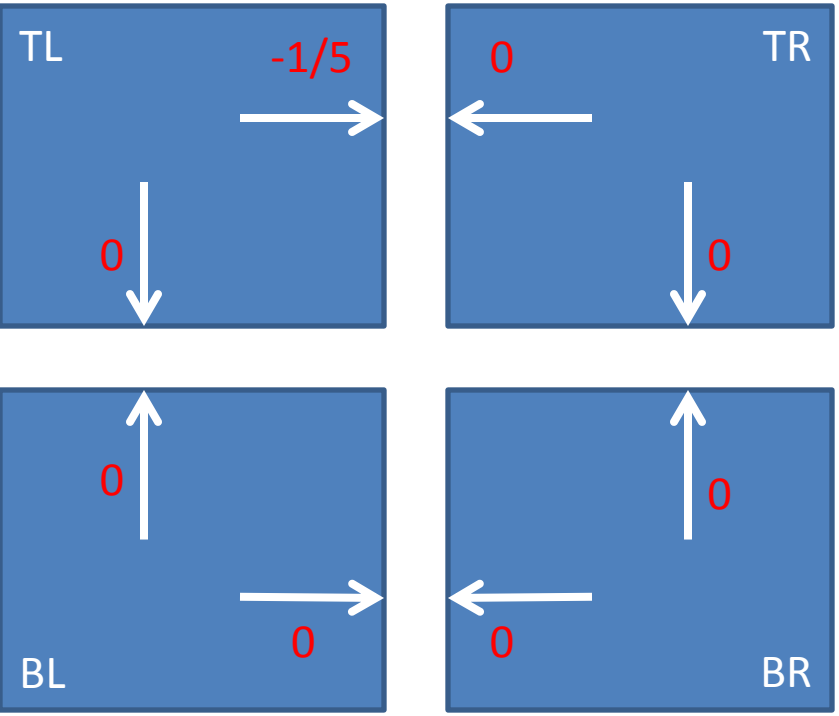
$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}, \boldsymbol{H}$

$s \quad a \quad r \quad s' \quad a'$

$$Q_\pi(TL, H) := \frac{4}{5} \times Q_\pi(TL, H) + \frac{1}{5} \times [-1 + Q_\pi(TR, H)]$$

$$= \frac{4}{5} \times \quad 0 \quad + \frac{1}{5} \times [-1 + \quad 0 \quad] = \frac{-1}{5}$$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, \boldsymbol{TR}; \boldsymbol{H}, \boldsymbol{4}, \boldsymbol{BR}$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{SARSA}$

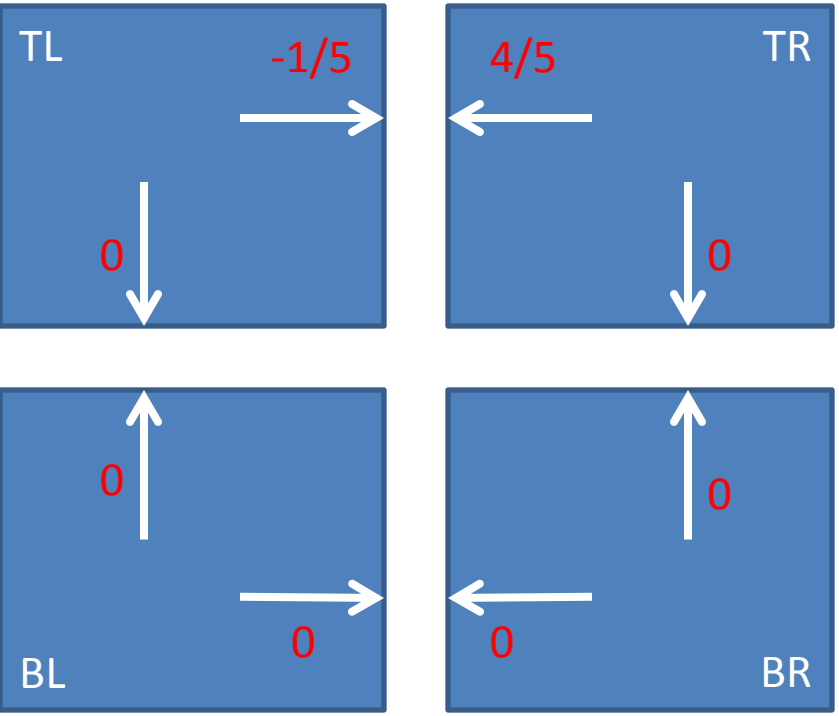
$Set\ Q_\pi(s, a) = 0$

$Consider\ \ \underset{s}{\boldsymbol{TR}}, \underset{a}{\boldsymbol{H}}, \underset{r}{\boldsymbol{4}}, \underset{s'}{\boldsymbol{BR}}, \underset{a'}{-}$

$$Q_\pi(TR, H) := \frac{4}{5} \times Q_\pi(TR, H) + \frac{1}{5} \times [4 + Q_\pi(BR, -)]$$

$$= \frac{4}{5} \times \quad 0 \quad + \frac{1}{5} \times [4 + \quad 0 \quad] = \frac{4}{5}$$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, TR; H, 4, BR$

$2. \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; \boldsymbol{H}, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{SARSA}$

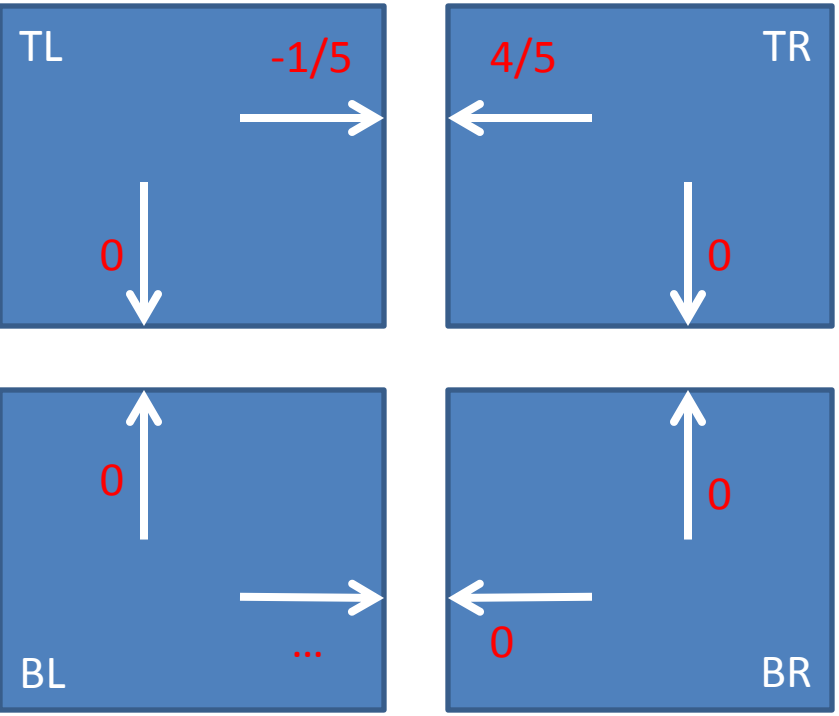
$Set\ Q_\pi(s, a) = 0$

$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}, \boldsymbol{H}$

$s \quad a \quad r \quad s' \quad a'$

$Q_\pi(TL, H) := \cdots$

SARSA Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, TR; H, 4, BR$

$2. \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; \boldsymbol{H}, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{SARSA}$

$Set\ Q_\pi(s, a) = 0$

$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}, \boldsymbol{H}$

$s \quad a \quad r \quad s' \quad a'$

$Q_\pi(TL, H) := \cdots$

Like Policy Evaluation – given enough iterations, get precise estimates

SARSA vs Q-Learning

SARSA: Update $\hat{Q}_\pi(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta\left[\, r + \gamma \hat{Q}_\pi(s',a') \,\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

SARSA evaluates policy π. Policy Improvement comes next. Iterate.

Q-Learning: Update $\hat{Q}_{opt}(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s')$

$$\hat{Q}_{opt}(s,a) \leftarrow (1-\eta)\hat{Q}_{opt}(s,a) + \eta\left[\, r + \gamma \hat{V}_{opt}(s') \right]$$

Helps evaluate optimal policy by sampling transitions and rewards
$\Rightarrow$ **Off-policy** algorithm

Use (randomized) policy to explore all (s,a), find greedy policy!

SARSA vs Q-Learning

SARSA: Update $\hat{Q}_\pi(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta\left[\, r + \gamma \hat{Q}_\pi(s',a') \,\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

SARSA evaluates policy π. Policy Improvement comes next. Iterate.

Q-Learning: Update $\hat{Q}_{opt}(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s')$

$$\hat{Q}_{opt}(s,a) \leftarrow \hat{Q}_{opt}(s,a) - \eta\left[\hat{Q}_{opt}(s,a) - (r + \gamma \hat{V}_{opt}(s'))\right]$$

Helps evaluate optimal policy by sampling transitions and rewards
$\Rightarrow$ **Off-policy** algorithm

Use (randomized) policy to explore all (s,a), find greedy policy!

SARSA vs Q-Learning

SARSA: Update $\hat{Q}_\pi(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta\left[\, r + \gamma \hat{Q}_\pi(s',a') \,\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

SARSA evaluates policy π. Policy Improvement comes next. Iterate.

Q-Learning: Update $\hat{Q}_{opt}(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s')$

$$\hat{Q}_{opt}(s,a) \leftarrow (1-\eta)\hat{Q}_{opt}(s,a) + \eta\left[\, r + \gamma \max_{a' \in A(s')} \hat{Q}_{opt}(s',a') \,\right]$$

Helps evaluate optimal policy by sampling transitions and rewards
$\Rightarrow$ **Off-policy** algorithm

Use (randomized) policy to explore all (s,a), find greedy policy!

SARSA vs Q-Learning

SARSA: Update $\hat{Q}_{\pi}(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s', a')$

$$\hat{Q}_{\pi}(s,a) \leftarrow (1-\eta)\hat{Q}_{\pi}(s,a) + \eta\left[\, r + \gamma \hat{Q}_{\pi}(s',a') \,\right]$$

Helps evaluate policy π by sampling transitions and rewards from policy π
$\Rightarrow$ **On-policy** algorithm

SARSA evaluates policy π. Policy Improvement comes next. Iterate.

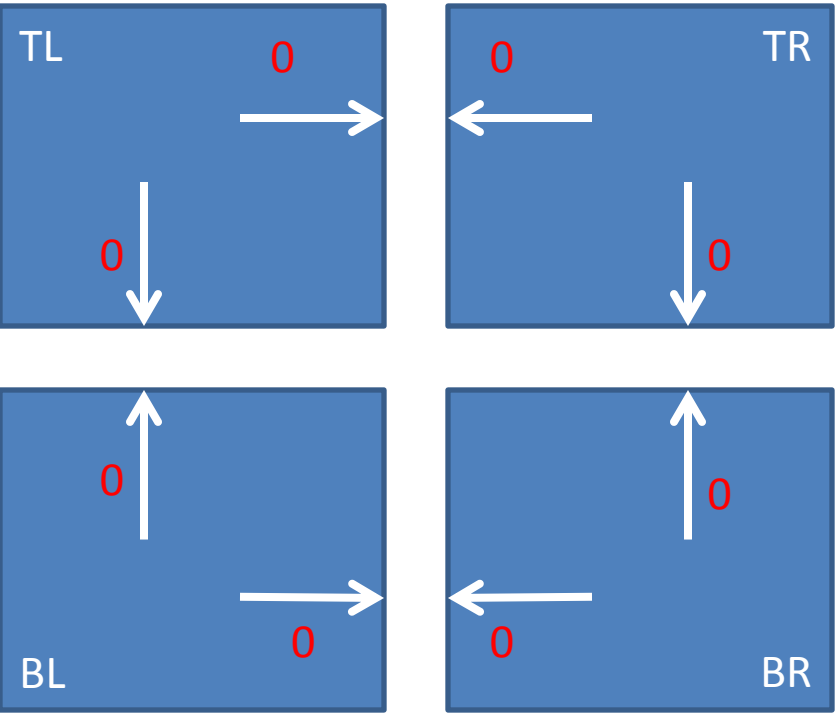
Q-Learning: Update $\hat{Q}_{opt}(s,a)$ based on $(\mathrm{s}, \mathrm{a}, \mathrm{r}, s')$

$$\hat{Q}_{opt}(s,a) \leftarrow (1-\eta)\hat{Q}_{opt}(s,a) + \eta\left[\, r + \gamma \max_{a' \in A(s')} \hat{Q}_{opt}(s',a') \,\right]$$

Helps evaluate optimal policy by sampling transitions and rewards
$\Rightarrow$ **Off-policy** algorithm

Use (randomized) policy to explore π to explore all (s,a), find greedy policy!

Q-Learning Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1.\ \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; H, 4, BR$

$2.\ TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning}$

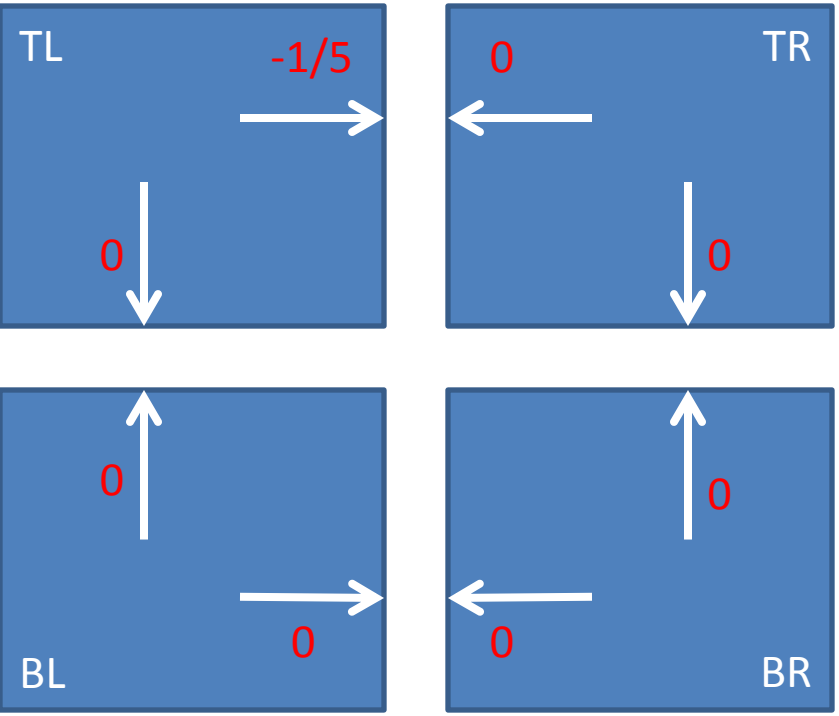
$Set\ Q_{opt}(s, a) = 0$

$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}$

$s \quad a \quad r \quad s'$

$$Q_{opt}(TL, H) := \frac{4}{5} \times Q_{opt}(TL, H) + \frac{1}{5} \times \left[-1 + \max_{a \in A(s)} Q_{opt}(TR, a)\right]$$

Q-Learning Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1.\ \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; H, 4, BR$

$2.\ TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning}$

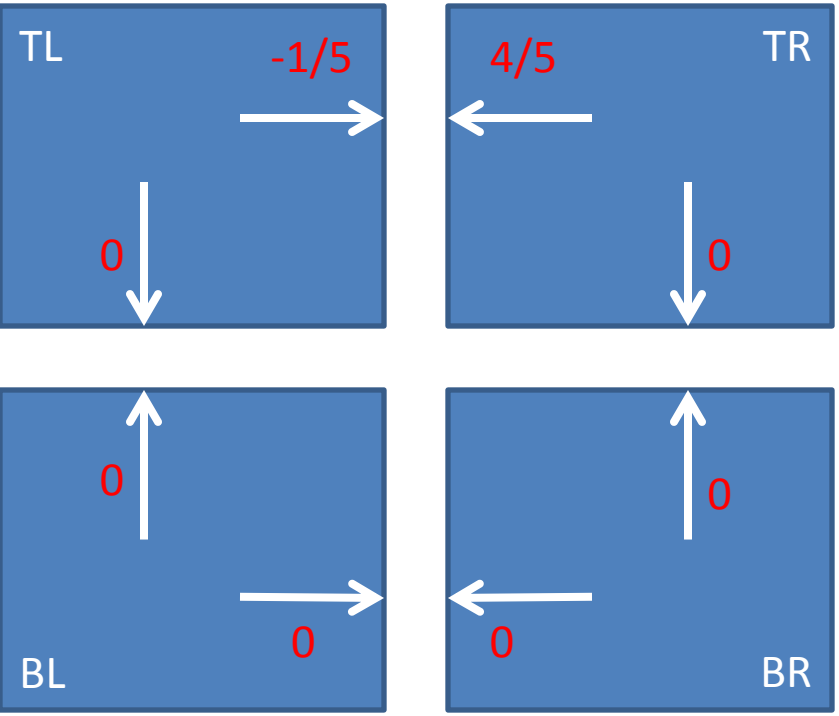
$Set\ Q_{opt}(s, a) = 0$

$Consider\ \underset{s}{\boldsymbol{TL}}, \underset{a}{\boldsymbol{H}}, \underset{r}{\boldsymbol{-1}}, \underset{s'}{\boldsymbol{TR}}$

$$Q_{opt}(TL, H) := \frac{4}{5} \times Q_{opt}(TL, H) + \frac{1}{5} \times \left[-1 + \max_{a \in A(s)} Q_{opt}(TR, a)\right]$$

$$= \frac{4}{5} \times \quad 0 \quad + \frac{1}{5} \times [-1 + \quad 0 \quad] = \frac{-1}{5}$$

Q-Learning Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, \boldsymbol{TR}; \boldsymbol{H}, \boldsymbol{4}, \boldsymbol{BR}$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning}$

$Set\ Q_{opt}(s, a) = 0$

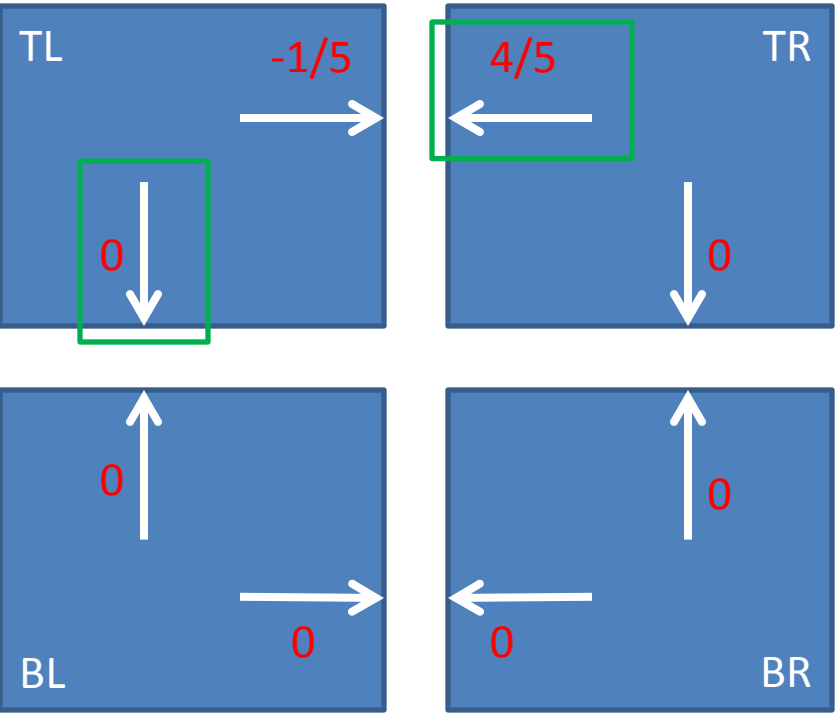
$Consider\ \boldsymbol{TR}, \boldsymbol{H}, \boldsymbol{4}, \boldsymbol{BR}$

$s \quad a \quad r \quad s'$

$$Q_{opt}(TR, H) := \frac{4}{5} \times Q_{opt}(TR, H) + \frac{1}{5} \times \left[4 + \max_{a \in A(s)} Q_{opt}(BR, a)\right]$$

$$= \frac{4}{5} \times \quad 0 \quad + \frac{1}{5} \times [4 + \quad 0 \quad] = \frac{4}{5}$$

Q-Learning Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, \boldsymbol{TR}; \boldsymbol{H}, \boldsymbol{4}, \boldsymbol{BR}$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

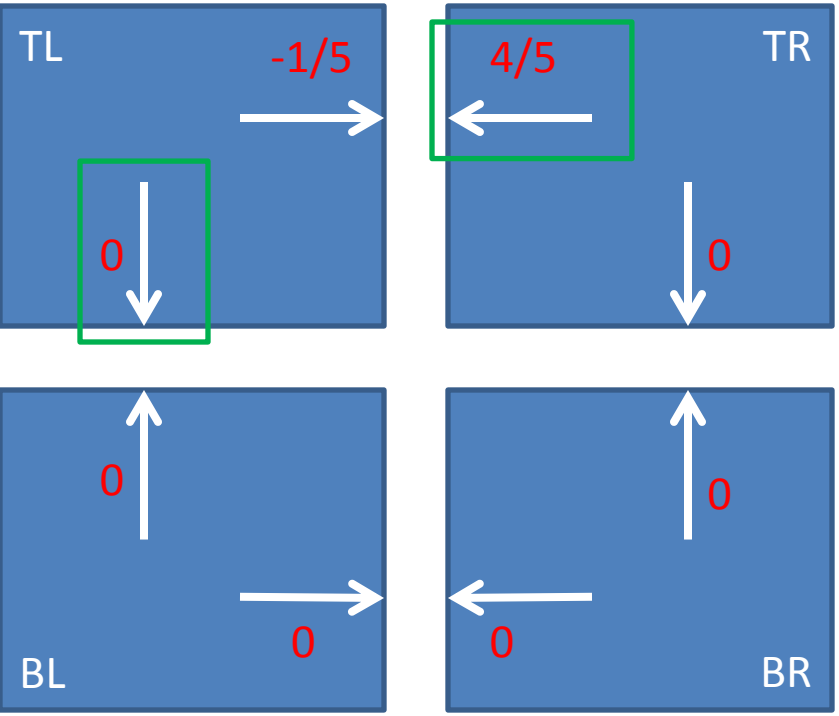
$\boldsymbol{Q - learning}$

$Set\ Q_{opt}(s, a) = 0$

$\boldsymbol{Optimal\ policy\ different\ from\ \pi}$

$\boldsymbol{\Rightarrow off - policy\ algorithm}$

Q-Learning Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, \boldsymbol{TR}; \boldsymbol{H}, \boldsymbol{4}, \boldsymbol{BR}$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning}$

$Set\ Q_{opt}(s, a) = 0$

$\boldsymbol{Optimal\ policy\ different\ from\ \pi}$

$\boldsymbol{\Rightarrow off - policy\ algorithm}$

Problems: *Q(s,V)* never explored

Need to use exploration/randomized policy

Policy Iteration vs Value Iteration

SARSA vs Q-Learning

Function Approximation and Q-Learning

Function Approximation

For optimal policy, compute $Q_{opt}(s, a)$ for all states $s \in S, a \in A(s)$

Our current setup does not generalize to unseen states!

What if

- Number of states $|S|$ is huge?
    - Extreme: what if continuous state? Infinitely many of them…

- Number of actions $|A(s)|$ is huge?

- Majority of neighboring states should have similar Q-values

Function Approximation

IDEA:
Express $Q_{opt}(s, a)$ as a weighted combination of its features

$$Q_{opt}(s, a) \approx w \,.\, \Phi(s, a) = w_1\phi_1(s, a) + w_2\phi_2(s, a) + \ \ldots + w_n\phi_n(s, a)$$

$\Phi(s, a) =$ Feature vector
$w =$ Weights

Function Approximation

IDEA:
Express $Q_{opt}(s,a)$ as a weighted combination of its features

$$Q_{opt}(s,a) \approx w \,.\, \Phi(s,a) = \mathrm{Q_{opt}(s,a;w)}$$

Q-Learning with function approximation: Update $\hat{Q}_{opt}(s,a;w)$ based on $(\mathrm{s,a,r},s')$

$$w \leftarrow w - \eta \left[\hat{Q}_{opt}(s,a;w) - \left(r + \gamma \hat{V}_{opt}(s';w) \right) \right] \Phi(s,a)$$

Amounts to SGD on:

$$\min_{w} \sum_{(s,a,r,s')} \left(\hat{Q}_{opt}(s,a;w) - \left(r + \gamma \hat{V}_{opt}(s';w) \right) \right)^2$$

Function Approximation

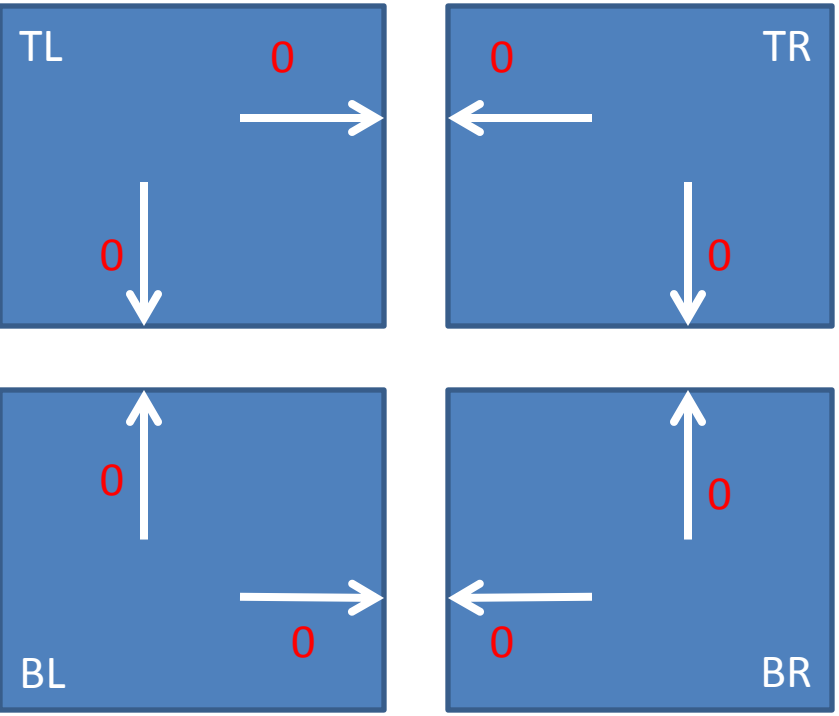
IDEA:
Express $Q_{opt}(s,a)$ as a weighted combination of its features

$$Q_{opt}(s,a) \approx w \,.\, \Phi(s,a) = \mathrm{Q_{opt}(s,a;w)}$$

Q-Learning with function approximation: Update $\hat{Q}_{opt}(s,a;w)$ based on $(\mathrm{s,a,r},s')$

$$w \leftarrow w - \eta \left[\hat{Q}_{opt}(s,a;w) - \left(r + \gamma \max_{a' \in A(s')} \hat{Q}_{opt}(s',a';w) \right) \right] \Phi(s,a)$$

Q-Learning with Function Approximation Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1. TL; H, -1, TR; H, 4, BR$

$2. TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

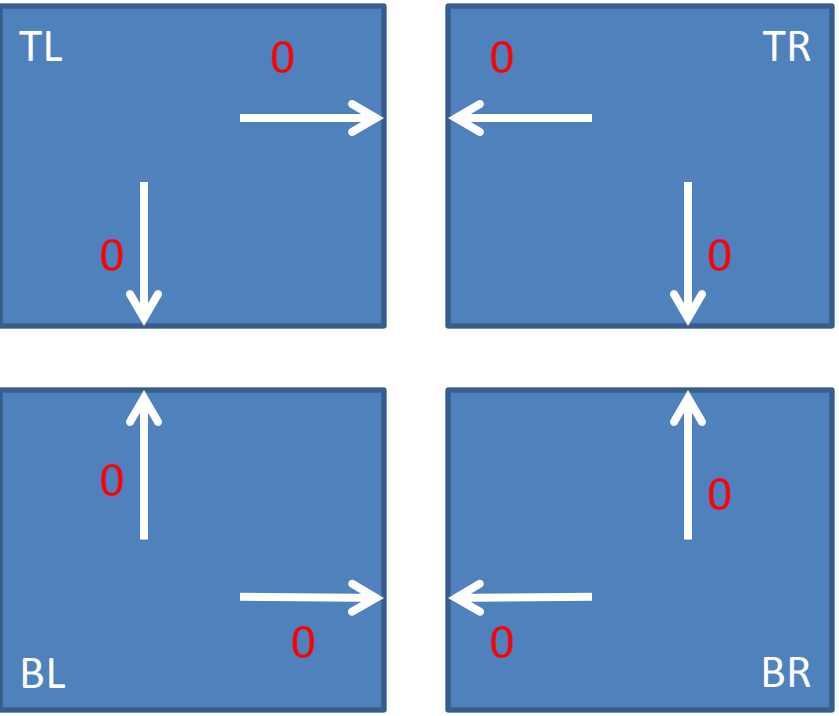
$\boldsymbol{Q - learning\ \ with\ function\ approximation}$

$\Phi(s, a) = [1, \mathbf{1} = \{s =* L\}, \mathbf{1} = \{a = H\},$
$\mathbf{1} = \{s =* L\}\mathbf{1} = \{a = H\}]^T$

$w = [w_1, w_2, w_3, w_4]^T$

$Set\ w = 0 \Rightarrow\ Q_{opt}(s, a; w) = 0$

Q-Learning with Function Approximation Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1.\ \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; H, 4, BR$

$2.\ TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning\ \ with\ function\ approximation}$

$\Phi(s,a) = [1, \mathbf{1} = \{s =* L\}, \mathbf{1} = \{a = H\},$
$\mathbf{1} = \{s =* L\}\mathbf{1} = \{a = H\}]^T$

$w = [w_1, w_2, w_3, w_4]^T$

$Set\ w = 0 \Rightarrow\ Q_{opt}(s, a; w) = 0$

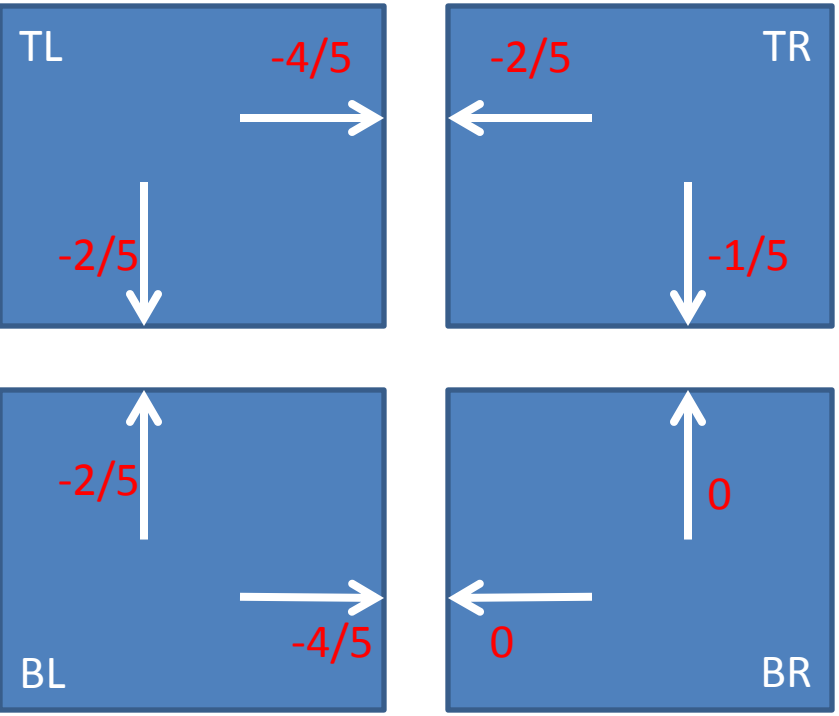
$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}$

$s \quad a \quad r \quad s'$

$$w := [0\ 0\ 0\ 0]^T - \frac{1}{5}\Big[Q_{opt}(TL, H; w) - \Big(-1 + \max_{a' \in A(TR)} Q_{opt}(TR, a'; w)\Big)\Big]\Phi(TL, H)$$

$$= [0\ 0\ 0\ 0]^T - \frac{1}{5}\Big[\quad 0 \quad - \Big(-1 + \max_{a' \in A(TR)} \quad 0 \quad \Big)\Big][1\ 1\ 1\ 1]^T$$

Q-Learning with Function Approximation Example



$Fixed\ policy\ \pi(s) = H, \forall s$

$Generate\ sample\ paths$

$1.\ \boldsymbol{TL}; \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}; H, 4, BR$

$2.\ TL; H, -1, TR; H, -1, TL; H, -1, BL; H, 4, BR$

$\boldsymbol{Q - learning\ \ with\ function\ approximation}$

$\Phi(s,a) = [1, \mathbf{1} = \{s =* L\}, \mathbf{1} = \{a = H\},$
$\mathbf{1} = \{s =* L\}\mathbf{1} = \{a = H\}]^T$

$w = [w_1, w_2, w_3, w_4]^T$

$Set\ w = 0 \Rightarrow\ Q_{opt}(s, a; w) = 0$

$Consider\ \boldsymbol{TL}, \boldsymbol{H}, \boldsymbol{-1}, \boldsymbol{TR}$

$s \quad a \quad r \quad s'$

$$w := [0\ 0\ 0\ 0]^T - \frac{1}{5}\left[Q_{opt}(TL, H; w)\ - \left(-1 + \max_{a' \in A(TR)} Q_{opt}(TR, a'; w)\right)\right]\Phi(TL, H)$$

$$= \left[\frac{-1}{5}\ \frac{-1}{5}\ \frac{-1}{5}\ \frac{-1}{5}\right]^T$$