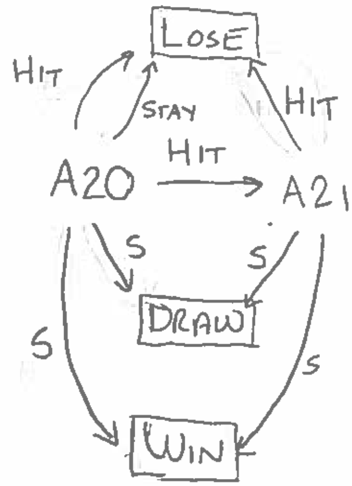
Passive Reinforcement Learning

1) Consider the following MDP for blackjack:



$R_t(\text{WIN}) = 1$

$R_t(\text{LOSE}) = -1$

$R_t(\text{DRAW}) = 0$

$R_t(\text{A20}) = R_t(\text{A21}) = 0$

$P(\text{LOSE} \mid \text{A20}, \text{STAY}) = 0.12$

$P(\text{DRAW} \mid \text{A20}, \text{STAY}) = 0.3$

$P(\text{WIN} \mid \text{A20}, \text{STAY}) = 0.58$

etc.

2) What if we only knew the states, actions, and rewards?

$Q = \{\text{A20}, \text{A21}, \text{WIN}, \text{LOSE}, \text{DRAW}\}$

$\Sigma = \{\text{HIT}, \text{STAY}\}$

$\Delta = ???$

$q_0 = \text{A20}$

$F = \{\text{WIN}, \text{LOSE}, \text{DRAW}\}$

$R_t(\text{WIN}) = 1$

$R_t(\text{LOSE}) = -1$

$R_t(\text{DRAW}) = 0$

$R_t(\text{A20}) = R_t(\text{A21}) = 0$

$P(q' \mid q, \sigma) = ???$

Can we still compute the expected utility of a particular policy, e.g. $\pi(\text{A20}) = \text{HIT}$, $\pi(\text{A21}) = \text{STAY}$?

Passive Reinforcement Learning

③ Well, not without more information. However, suppose we play several games using this policy:

(i) $A20 \xrightarrow{\text{HIT}} \text{LOSE}$ (utility $= -1$)

(ii) $A20 \xrightarrow{\text{HIT}} \text{LOSE}$ (utility $= -1$)

(iii) $A20 \xrightarrow{\text{HIT}} A21 \xrightarrow{\text{STAY}} \text{WIN}$ (utility $= 1$)

(iv) $A20 \xrightarrow{\text{HIT}} \text{LOSE}$ (utility $= -1$)

(v) $A20 \xrightarrow{\text{HIT}} A21 \xrightarrow{\text{STAY}} \text{DRAW}$ (utility $= 0$)

---

④ The simplest strategy is to estimate the expected utility directly:

$$\begin{aligned} U^{\pi}(A20) &= \frac{(-1)+(-1)+1+(-1)+0}{5} \\ &= \frac{-2}{5} \\ &= -0.4 \end{aligned}$$

This is just the average utility I've experienced over the 5 times I've been in state A20.

---

⑤ The main downside to this strategy (called <u>direct utility estimation</u>) is that it treats each state independently, e.g. $U^{\pi}(A20)$ and $U^{\pi}(A21)$ are estimated as if they have nothing to do with each other.

(6) Another strategy is to estimate the transition probabilities of the MDP by keeping track of how many times we experience each transition $\langle q, \sigma, q' \rangle$.

(i) $A20 \xrightarrow{HIT} LOSE$

(ii) $A20 \xrightarrow{HIT} LOSE$

(iii) $A20 \xrightarrow{HIT} A21 \xrightarrow{STAY} WIN$

(iv) $A20 \xrightarrow{HIT} LOSE$

(v) $A20 \xrightarrow{HIT} A21 \xrightarrow{STAY} DRAW$

	Count
$A20 \xrightarrow{HIT} LOSE$	3
$A20 \xrightarrow{HIT} A21$	2
$A21 \xrightarrow{STAY} WIN$	1
$A21 \xrightarrow{STAY} DRAW$	1

These counts allow us to estimate the transition probabilities in a straightforward way:

$$P(LOSE \mid A20, HIT) = \frac{count(A20 \xrightarrow{HIT} LOSE)}{count(A20 \xrightarrow{HIT} LOSE) + count(A20 \xrightarrow{HIT} A21)} = \frac{3}{5}$$

$$P(A21 \mid A20, HIT) = \frac{count(A20 \xrightarrow{HIT} A21)}{count(A20 \xrightarrow{HIT} LOSE) + count(A20 \xrightarrow{HIT} A21)} = \frac{2}{5}$$

etc.

This technique is called Adaptive Dynamic Programming (ADP).

PASSIVE REINFORCEMENT LEARNING

7) Once we know the transition probabilities, we can estimate the utilities using the standard techniques (e.g. value iteration) for fully-specified MDPs.

8) Both of these methods (direct utility estimation and ADP) require us to store all our games and their results in memory. These are called <u>offline</u> methods.

Can we play a game, update our utilities <u>online</u>, and then forget about the game (i.e. don't store the results, except via our updated utilities)?

9) Consider the task of computing the mean of a sequence of numbers:

15, 3, 10, 4, 3

Everybody probably knows the offline method:

$$\text{mean} = \frac{15 + 3 + 12 + 2 + 3}{5} = \frac{35}{5} = 7$$

PASSIVE REINFORCEMENT LEARNING

(10) But there's also a convenient online method:

n	next num x_n	online mean m_n
1	15	15
2	3	$m_1 + \frac{1}{2} \cdot (x_2 - m_1) = 9$
3	12	$m_2 + \frac{1}{3}(x_3 - m_2) = 10$
4	2	$m_3 + \frac{1}{4}(x_4 - m_3) = 8$
5	3	$m_4 + \frac{1}{5}(x_5 - m_4) = 7$

(11) In pseudocode:

```
m = 0
for n = 1 to N:
    m = m + (1/n) · (x_n − m)
```

Note that we only need to ever have 3 numbers in memory at one time: m, n, and x_n.

This is true regardless of how many numbers we are averaging.

Passive Reinforcement Learning

(12) Now consider applying this to compute expected utility of a state, given a policy π. As we play the game according to our policy, we get a sequence of utilities, e.g.

$A21 \xrightarrow{\text{STAY}} \text{WIN} \qquad (\text{utility} = U^\pi(\text{WIN}) = 1)$

$A21 \xrightarrow{\text{STAY}} \text{WIN} \qquad (\text{utility} = U^\pi(\text{WIN}) = 1)$

$A21 \xrightarrow{\text{STAY}} \text{DRAW} \qquad (\text{utility} = U^\pi(\text{DRAW}) = 0)$

The expected utility of state A21 is the average of these utilities as our number of observations goes to infinity.

---

(13) In other words, we want to average

$$R(q) + \gamma \cdot U^\pi(q')$$

reward + the discounted utility of the next state

for each transition $q \xrightarrow{\pi(q)} q'$ we observe.

So we can compute this using our online technique:

$U^\pi(q) = 0$

for $n = 1$ to ∞:

$$U^\pi(q) = U^\pi(q) + \frac{1}{n} \cdot \left(\overbrace{\left[R(q) + \gamma U^\pi(q')\right]}^{\text{"}x_n\text{"}} - \overbrace{U^\pi(q)}^{\text{"}m\text{"}} \right)$$

(14) Turns out this works even when we compute the expected utility of each state simultaneously:

TDLEARNING(Q, Σ, R, π):

set $U^\pi(q) = 0$, $n_q = 1$ for all $q \in Q$

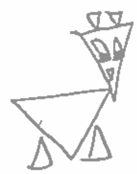
repeat:

observe transition $q \xrightarrow{\pi(q)} q'$.

update $U^\pi(q) = U^\pi(q) + \boxed{\frac{1}{n_q}}\left(\left[R(q) + \gamma U^\pi(q')\right] - U^\pi(q)\right)$

$n_q \mathrel{+}= 1$

if we keep taking observations forever, then this can be replaced with any $\alpha(n_q) = O\left(\frac{1}{n_q}\right)$ and it will converge to the mean as $n_q \to \infty$



This is called <u>temporal difference (TD) learning</u>.