What is Reinforcement Learning

TBD

Markov Decision Processes

Markov Decision Processes try to give us a framework, the framework need we describe our problem with these important concepts:

Concept	Math present	Describetion
States	S	describes the world's state
Model	$T(s, a, s') = P(s' \| s, a)$	T is Transition Model function it give that the probability that if you were in state s, and you toke action a and you end up transitioning with state s^{\prime}
Actions	$A(s)$, A	the actions you can do in the world
Reward	R(s), R(s, a), R(s, a, s^{\prime})	Reward function give you some rewards when you were in state s, or you were in state s and toke action a, or you were in state a and toke action a and you end up transitioning at s'
Policy	$\pi(s) = a, \pi^*$	$\pi(s)$ is policy we need to find and it give that when you were in state s and what action you should take in the next step and this policy can help you find the answer or get destination or whatever, and π^* is the best policy you found in these all possible policies

States, Model(Transition Model), Actions, Reward they are problem, Policy is the solution.

Reward function

Try to think about whats the different if your reward function is $R(s) = 2$ and $R(s) = -2$, that mean whatever state you are, you will get 2 or -2 reward, that the simplest reward function.
$R(2) = 2$ will encourage you to stay in the world insteal of getting terminal state.
$R(2) = 2$ will keep you want to leave away from the world.

Stationary Preferences

if you use s util function to compare two sequence of state like:
$U_1(s_0, s_1, s_2, s_3, s_4, ...)$
$U_2(s_0, s_1', s_2', s_3', s_4', ...)$
and you get the conclusion that $U_1 > U_2$
then for these two sequence of state:
$U_1(s_1, s_2, s_3, s_4, ...)$
$U_2(s_1', s_2', s_3', s_4', ...)$
you will also think:
$U_2 > U_3$
that is the stationary preferences

Sequences of Reward

if one kind of U function like this:
$U(s_0, s_1, s_2, s_3, ...) = \sum_{t=0}^{\infty} R(s_t) = \infty$
that's true because the reward is always positive.
this is a typical infinite world situation, if your U function like this, each step of your decision making will be nothing.
but if your U function like this:
$U(s_0, s_1, s_2, s_3, ...) = \sum_{t=0}^{\infty} \gamma^t R(t)$
the γ^t will change the thing to a situation that you still in a infinite world but you will reach a point that whatever you choose to go, you never get the bound of the world.
also you will get a equition:
$U <= \sum_{t=0}^{\infty} \gamma^t R_{max} = \frac{R_{max}}{1-\gamma}$
because:
$x = (\gamma^0 + \gamma^1 + \gamma^2 + \gamma^3 + ...)$
$x = \gamma^0 + \gamma \cdot (\gamma^0 + \gamma^1 + \gamma^2 + ...)$
$x = \gamma^0 + \gamma \cdot x$
$x = \frac{\gamma^0}{1-\gamma}$
so:
$\sum_{t=0}^{\infty} \gamma^t R_{max} = \frac{R_{max}}{1-\gamma} = \frac{\gamma^0}{1-\gamma} \cdot R_{max}$

Policies

How we use mathematic way to express policy function:
$\pi^* = \underset{\pi}{argmax} E[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi]$
that means the optimal policy is that if we follow this policy, we can get a sequences of states and it's corresponding rewards sum is max. Also the rewards is discounted by γ factor.

Next how to express the utility of s:
$U^\pi(s) = E[\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi, s_0 = s]$
so the utility of s is the long term reward of current state reward plus all the other rewards follow the policy π which is the rewards from s on to the infinite state.

Note: R(s) is immediately feedback/reward U(s) is long term feedback/reward

if we have utility we have new policy function:
$\pi^*(s) = \underset{a}{argmax} \sum_{s'} T(s,a,s') U(s')$
Now the utility is always follow the optimal policy:
$U(s) = U^{\pi^*}(s)$

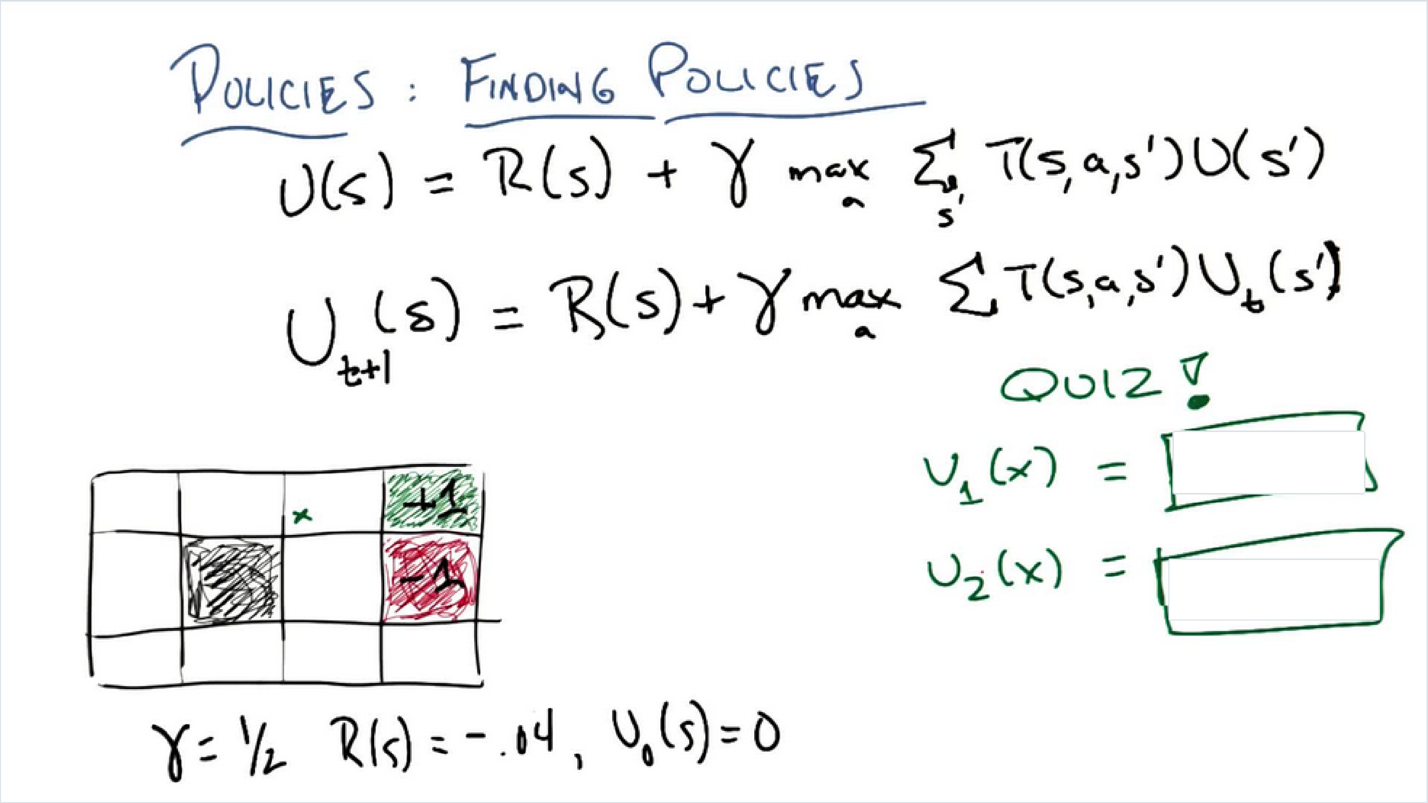
It's means the optimal policy for every state, return the action a that maximizes my expected utility. This is recursive function because we use optimal policy π^* to calculate itself, later we will make it possible.

Bellman Equation

Now we introduce bellman equation:
$U(s) = R(s) + \gamma \underset{a}{max} \sum_{s'} T(s,a,s') U(s')$
We ganna use $U(s')$ to calculate $U(s)$, the utility equals immediately reward at state s plus discounted utility that use the action a which maximizes the long term rewards from s on.

Finding Policies 1

Right now we have Bellman equation, and we don't know how to solve U(s), the Value Iteration method could be a way to solve it but I don't know why, so let do a quiz:



POLICIES : FINDING POLICIES

$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U(s')$

$U_{t+1}(s) = R(s) + \gamma \max_a \sum T(s,a,s')U_t(s')$

QUIZ!

$U_1(x) =$

$U_2(x) =$

$\gamma = 1/2 \quad R(s) = -.04, \quad U_0(s) = 0$

What we need to know is, all the state initial utility is ZERO except green grid and red grid its One and negative One:

$$U_1(x) = R(x) + \gamma \underset{a}{max} \begin{cases} \sum_{a_{up}} T(x, a_{up}, x^{up})U_0(x^{up}) \\ \sum_{a_{down}} T(x, a_{down}, x^{down})U_0(x^{down}) \\ \sum_{a_{left}} T(x, a_{left}, x^{left})U_0(x^{left}) \\ \sum_{a_{right}} T(x, a_{right}, x^{right})U_0(x^{right}) \end{cases}$$

then we choose the max one:

$$U_1(x) = -0.04 + 0.5 \times \underset{a}{max} \begin{cases} 0.8 \times 0 + 0.1 \times 1 + 0.1 \times 0 = 0.1 \\ 0.8 \times 0 + 0.1 \times 1 + 0.1 \times 0 = 0.1 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \\ 0.8 \times 1 + 0.1 \times 0 + 0.1 \times 0 = 0.8 \end{cases} = -0.04 + 0.5 \times (0.8 \times 1 + 0.1 \times 0 + 0.1 \times 0) = 0.36$$

This was because we always want max value so we first choose to go right to red grid at same time we have 0.2 probability to go wrong direction to go down and go up.
When we go up and down, we'll get ZERO utility because initial utility is ZERO. Next we use $U_1(x)$ to solve $U_2(x)$:

$$U_2(x) = R(x) + \gamma \underset{a}{max} \begin{cases} \sum_{a_{up}} T(x, a_{up}, x^{up})U_1(\text{ x }) \\ \sum_{a_{down}} T(x, a_{down}, x^{down})U_1(x^{down}) \\ \sum_{a_{left}} T(x, a_{left}, x^{left})U_1(x^{left}) \\ \sum_{a_{right}} T(x, a_{right}, x^{right})U_1(x^{right}) \end{cases}$$

as you see we need to get $U_1(x^{up|down|left|right})$, we can fellow the function $U_1(x)$ way to get it, so:
$U_1(x^{up})$ is out of grid so we assume $U_1 x^{up} = 0$.

$$U_1(x^{down}) = -0.04 + 0.5 \times \underset{a}{max} \begin{cases} 0.8 \times 0 + 0.1 \times 0 + 0.1 \times -1 = -0.1 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times -1 = -0.1 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \\ 0.8 \times -1 + 0.1 \times 0 + 0.1 \times 0 = -0.8 \end{cases}$$
$$= -0.04 + 0.5 \times (0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0) = -0.04$$

$$U_1(x^{left}) = -0.04 + 0.5 \times \underset{a}{max} \begin{cases} 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \\ 0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0 = 0 \end{cases}$$
$$= -0.04 + 0.5 \times (0.8 \times 0 + 0.1 \times 0 + 0.1 \times 0) = -0.04$$

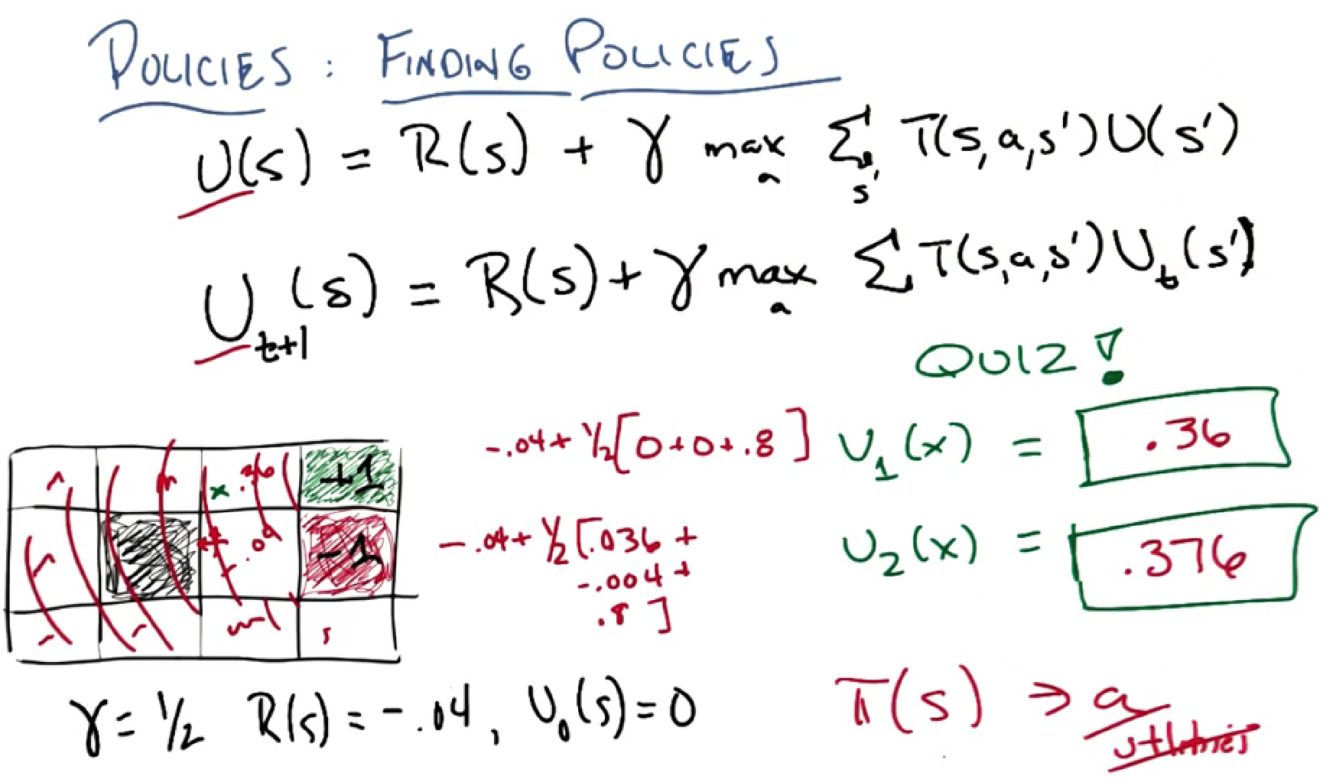
$U_1(x^{right})$ is already given with 1 so $U_1(x^{right}) = 1$

now let's use above result to get $U_2(x)$:

$$U_2(x) = -0.04 + 0.5 \times \underset{a}{max} \begin{cases} 0.8 \times 0.36 + 0.1 \times -0.04 + 0.1 \times 1 = 0.384 \\ 0.8 \times -0.04 + 0.1 \times -0.04 + 0.1 \times 1 = 0.064 \\ 0.8 \times -0.04 + 0.1 \times 0.36 + 0.1 \times -0.04 = 0 \\ 0.8 \times 1 + 0.1 \times 0.36 + 0.1 \times -0.04 = 0.832 \end{cases}$$
$$= -0.04 + 0.5 \times (0.8 \times 1 + 0.1 \times 0.36 + 0.1 \times -0.04) = 0.376$$

$$U_2(x) = -0.04 + 0.5(0.8 \times 1 + 0.1 \times 0.36 + 0.1 \times -0.04) = 0.376$$

Now we get $U_2(x)$, the most interesting thing is I found current Utility of state is similar to anergy spreading from center on, in our situation, the anergy center is x^{right} who's Utility is 1, other utility of state is like under anergy spreading and their value is smaller than center, the smallest one is most faraway one:



POLICIES : FINDING POLICIES

$$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U(s')$$

$$U_{t+1}(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U_t(s')$$

QUIZ!

$-.04 + \frac{1}{2}[0+0+.8]$ $U_1(x) = \boxed{.36}$

$-.04 + \frac{1}{2}[.036 + -.004 + .8]$ $U_2(x) = \boxed{.376}$

$\gamma = \frac{1}{2}$ $R(s) = -.04$, $U_0(s) = 0$

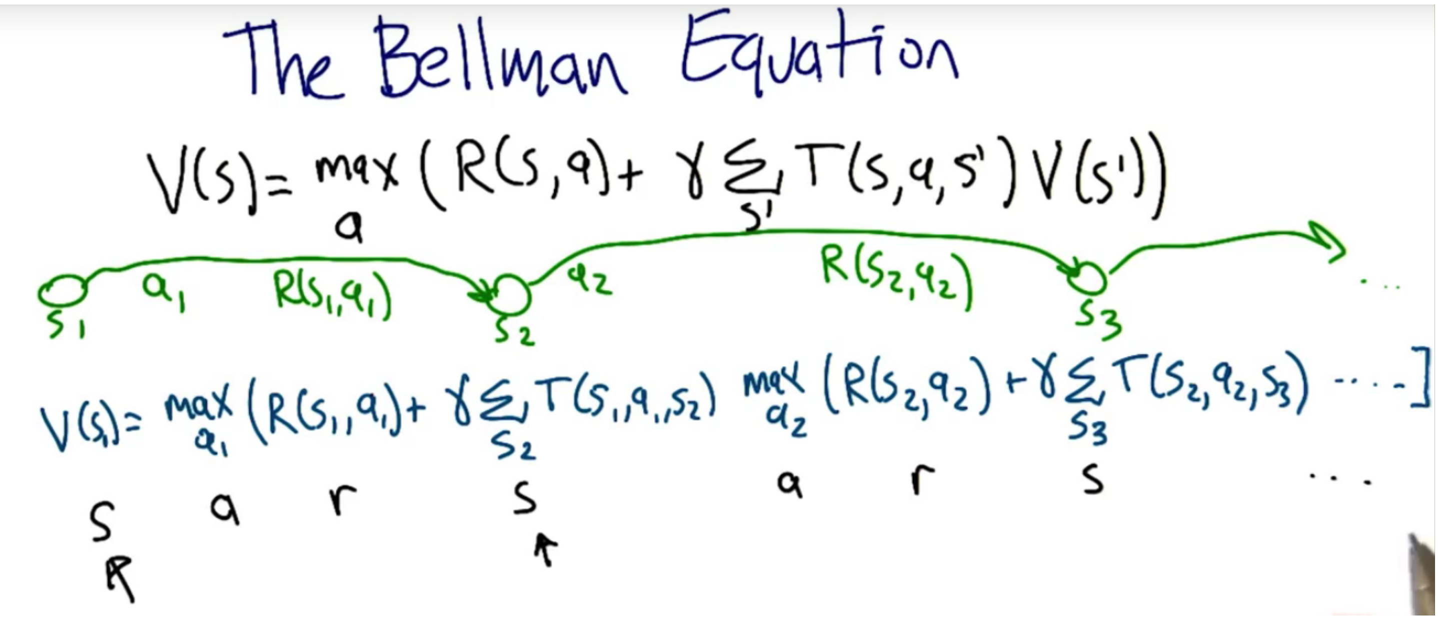
$\pi(s) \rightarrow a$ ~~utilities~~

Finding Policies 2

So far we learned about getting Utility, and before we have been teached that the policy equation $\pi^*(s) = a$ need the core components U(s) to solve it, so next we will use the important result to find policy.

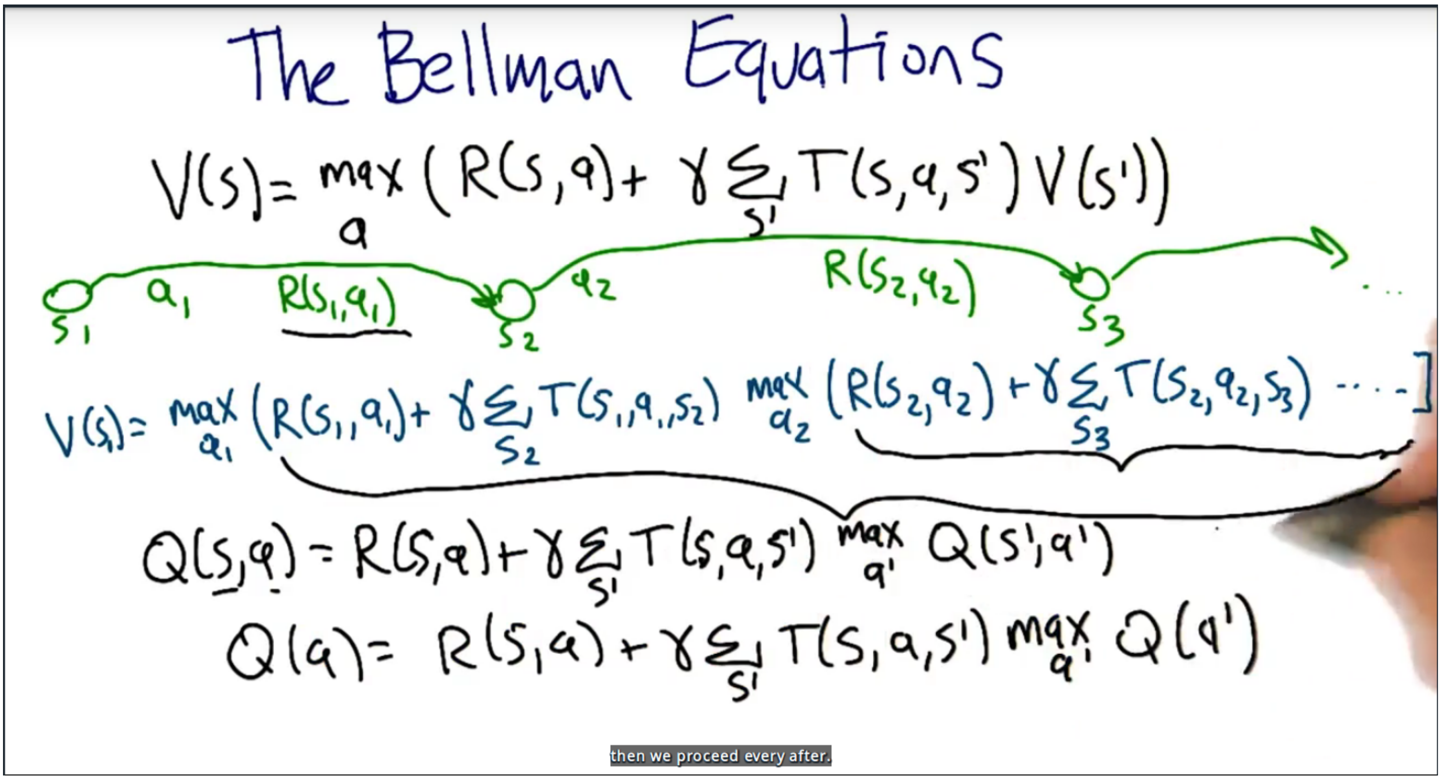
First we need to introduce the difference presentation of Bellman Equation
Let's see this one:
$$V(s) = \underset{a}{max}(R(s,a) + \gamma \sum_{s'} T(s,a,s')V(s'))$$
You may found the difference is we use V to express Value instead of U to express Utility, we move ahead the $\underset{a}{max}$ to express take maximize action, we use $R(s,a)$ to express taking a action in a state and get reward instead of get reward when you get a state.

If we expanded the equation, we can understand the equation by this way:


The Bellman Equation
V(s) = max_a (R(s,a) + γ Σ_s' T(s,a,s') V(s'))
s1 a1 R(s1,a1) s2 a2 R(s2,a2) s3 ...
V(s1) = max_a1 (R(s1,a1) + γ Σ_s2 T(s1,a1,s2) max_a2 (R(s2,a2) + γ Σ_s3 T(s2,a2,s3) ...]
s a r s a r s ...
R

Semantic description is :
We start at a specific state s, and we take a action a, and we get the reward of taking action a in state s, and this action lands the new state, and we recursively execute the process.

Let's see another one:

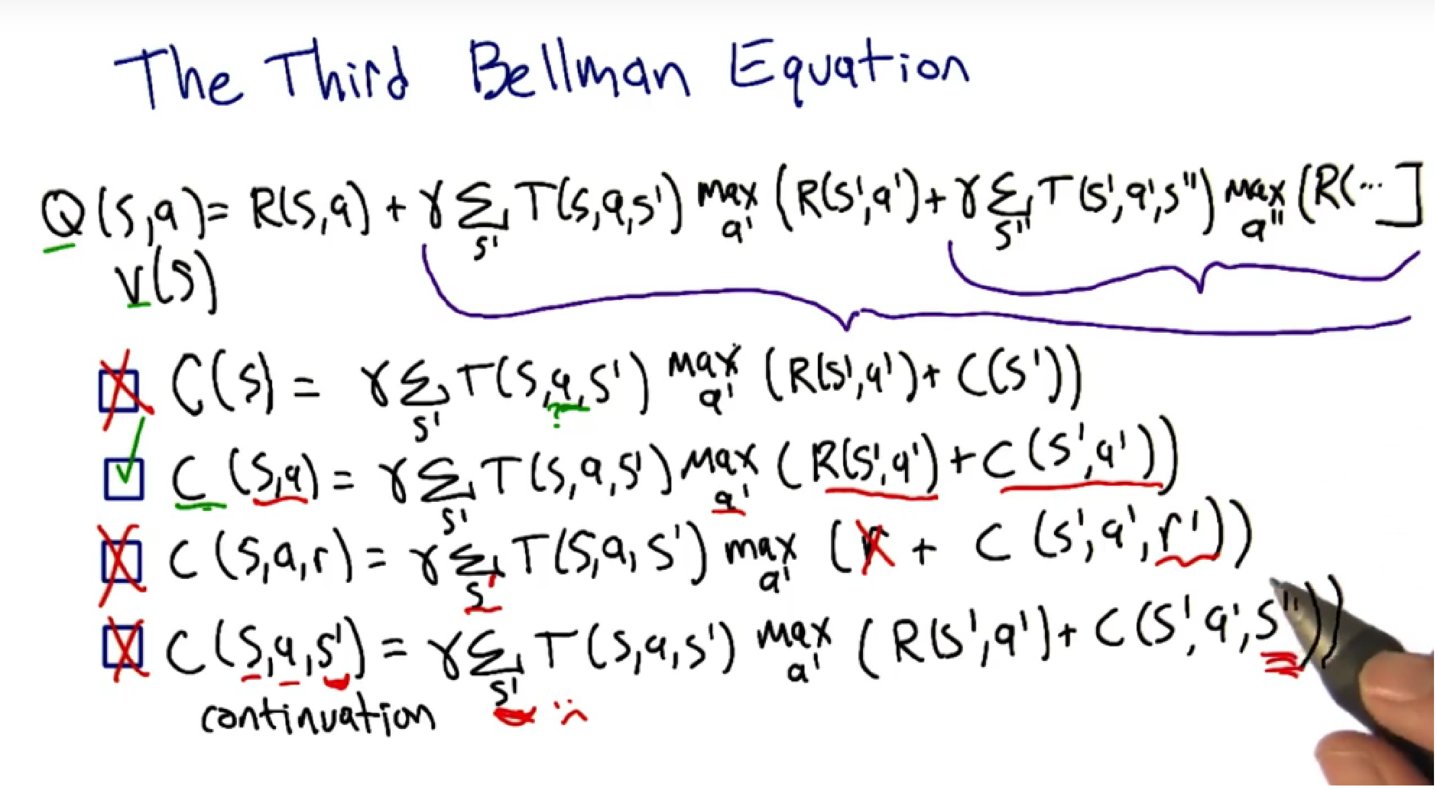

The Bellman Equations
V(s) = max_a (R(s,a) + γ Σ_s' T(s,a,s') V(s'))
s1 a1 R(s1,a1) s2 a2 R(s2,a2) s3 ...
V(s1) = max_a1 (R(s1,a1) + γ Σ_s2 T(s1,a1,s2) max_a2 (R(s2,a2) + γ Σ_s3 T(s2,a2,s3) ...]
Q(s,a) = R(s,a) + γ Σ_s' T(s,a,s') max_a' Q(s',a')
Q(a) = R(s,a) + γ Σ_s' T(s,a,s') max_a' Q(a')
then we proceed every after.

If we use a difference view, we may found that the V equation have another recursive sub-sequence from first R(s,a) on to next R(s,a), so we can have another equation:

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \underset{a'}{max} Q(s',a')$$

Semantic description is :
We start at some state s and we take action a, and start to process every after, this is we get a reward of taking action a at state s, and this action lands we transform to state s' and we recursively execute the process.

Let's see another one:



The Third Bellman Equation

$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} (R(s',a') + \gamma \sum_{s''} T(s',a',s'') \max_{a''} (R(\cdots]$

$V(s)$

☒ $C(s) = \gamma \sum_{s'} T(s,a,s') \max_{a'} (R(s',a') + C(s'))$

☑ $C(s,a) = \gamma \sum_{s'} T(s,a,s') \max_{a'} (R(s',a') + C(s',a'))$

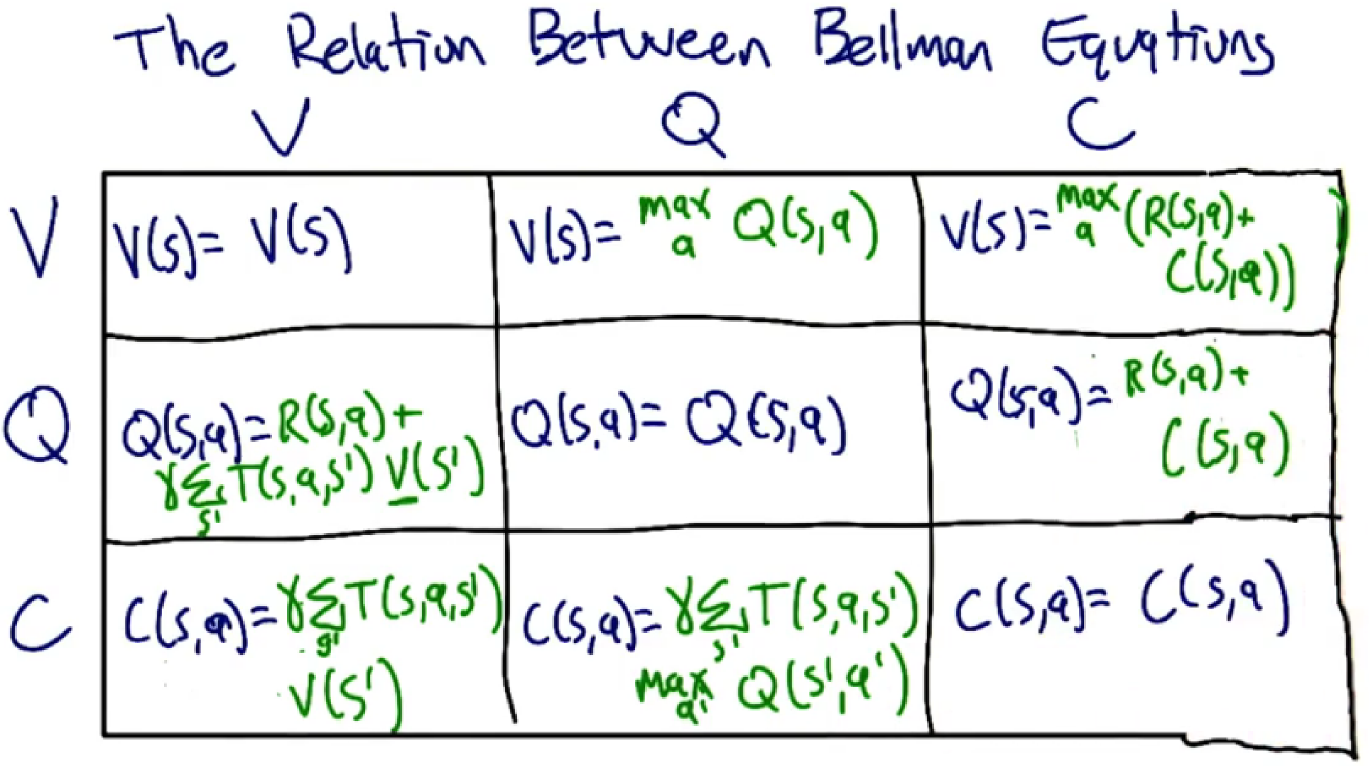
☒ $C(s,a,r) = \gamma \sum_{s'} T(s,a,s') \max_{a'} (\cancel{R} + C(s',a',r'))$

☒ $C(s,a,s') = \gamma \sum_{s'} T(s,a,s') \max_{a'} (R(s',a') + C(s',a',s''))$

continuation

When we get Q equation, we again may found another recursive sub-sequence form first γ on to next γ, so we can have another equation:
$C(s,a) = \gamma \sum_{s'} \underset{a'}{max}(R(s',a') + C(s',a'))$

With this V function and Q function and C function, we can use one of them to express another one of them:
For example:
$V(s) = \underset{a}{max}(Q(s,a))$
$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s')V(s')$
$C(s,a) = \gamma \sum_{s^{prime}} T(s,a,s')V(s')$
I guess you can find more like above those.



The Relation Between Bellman Equations

	V	Q	C
V	$V(s)=V(s)$	$V(s)=\max_a Q(s,a)$	$V(s)=\max_a (R(s,a)+C(s,a))$
Q	$Q(s,a)=R(s,a)+\gamma\sum_{s'}T(s,a,s')\underline{V}(s')$	$Q(s,a)=Q(s,a)$	$Q(s,a)=R(s,a)+C(s,a)$
C	$C(s,a)=\gamma\sum_{s'}T(s,a,s')V(s')$	$C(s,a)=\gamma\sum_{s'}T(s,a,s')\max_{a'}Q(s',a')$	$C(s,a)=C(s,a)$

These three function indeed have difference meaning in after introduction, let me refer it out slowly.

Reinforcement Learning Basic

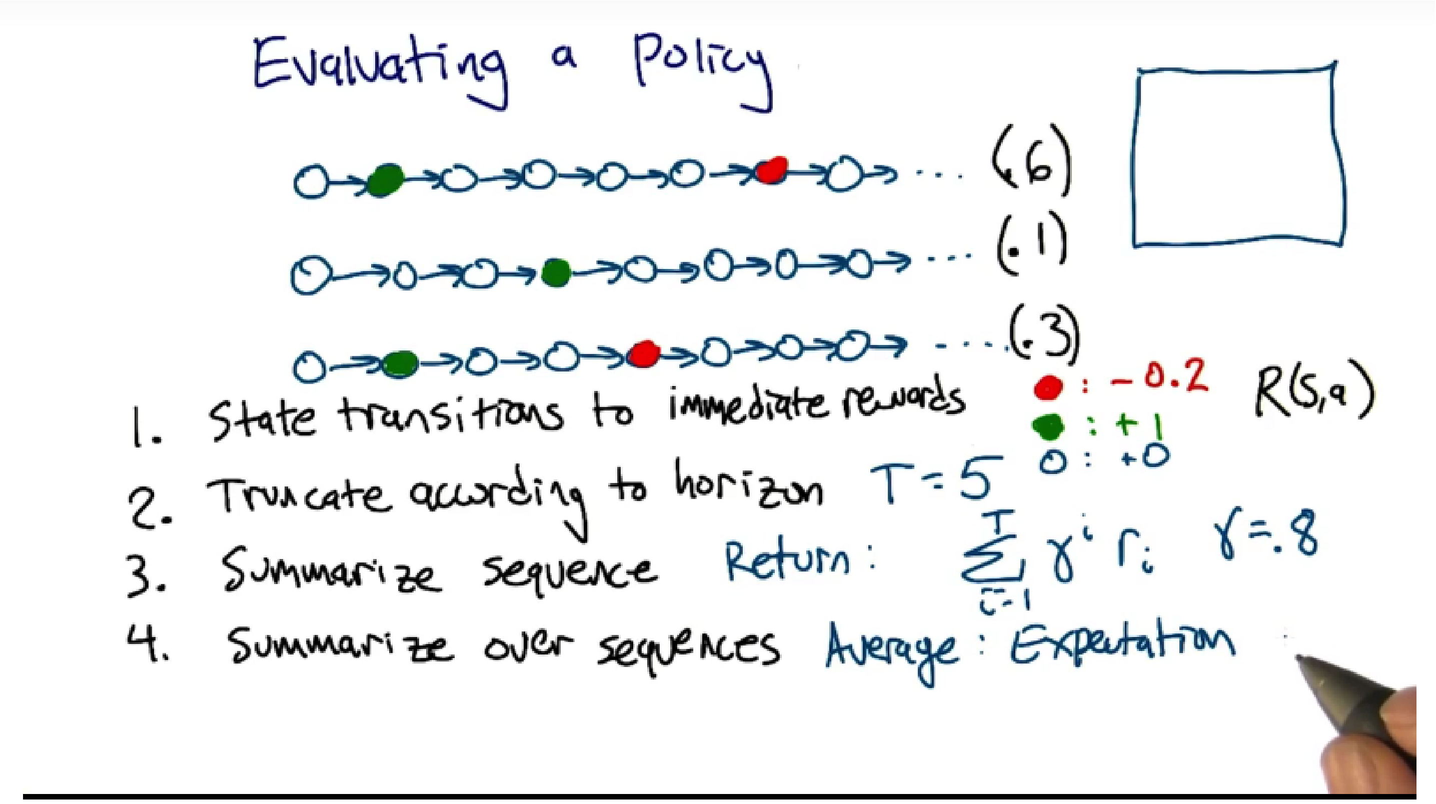
The word Learning here is to learn what kind of action should require for a given environment and agents, and
what is the optimal policy.

Behavior Structures

1. Plan
    1. Plan is a fixed sequence of actions
2. Conditional Plan
    1. Conditional Plan is a action tree each branch means a "if" sentence there
3. Stationary Policy/ Universal Plan
    1. for every states there are same "if" or there a universal "if" can handle every states
    2. very large

Evaluating Policy

Just a way that use one number to describe a policy



Evaluating a Policy

🔴 : −0.2
🟢 : +1
○ : +0
R(s,a)

1. State transitions to immediate rewards
2. Truncate according to horizon $T=5$
3. Summarize sequence Return: $\sum_{i=1}^{T} \gamma^i r_i$ $\gamma=.8$
4. Summarize over sequences Average: Expectation

Evaluating Learner

1. Value of returned Policy
   - How good returned Policy is
2. Computational complexity (time)
3. Sample complexity (time)
   - how much data it needs

Normally space complexity is not interesting because now we won't be limited my space issue.

TD and Friends

RL Context

There difference forms of RL

1. Model-based

$$< s, a, r >^* \rightarrow [ModelLearner] \rightarrow [T/Rfunction] \rightarrow [MDPSolver] \rightarrow Q^* \rightarrow [argmax] \rightarrow \pi$$

*[Model Learner] takes T/R function as a feedback
2. Value-function-based / Model-free

$$< s, a, r >^* \rightarrow [ValueUpdate] \rightarrow Q \rightarrow [argmax] \rightarrow \pi$$

*[Value Update] takes Q as a feedback

3. Policy Search
$< s, a, r >^* \to [PolicyUpdate] \to \pi$
*[Policy Update] takes π as a feedback

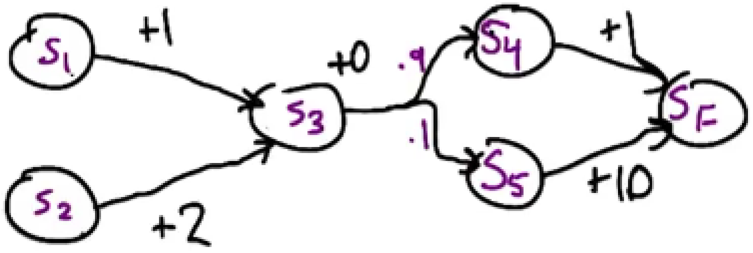
TD(λ)

Predict with given markov chain and terminal function:

Temporal Difference Lambda is try to predict Value(s) at any time, for example:

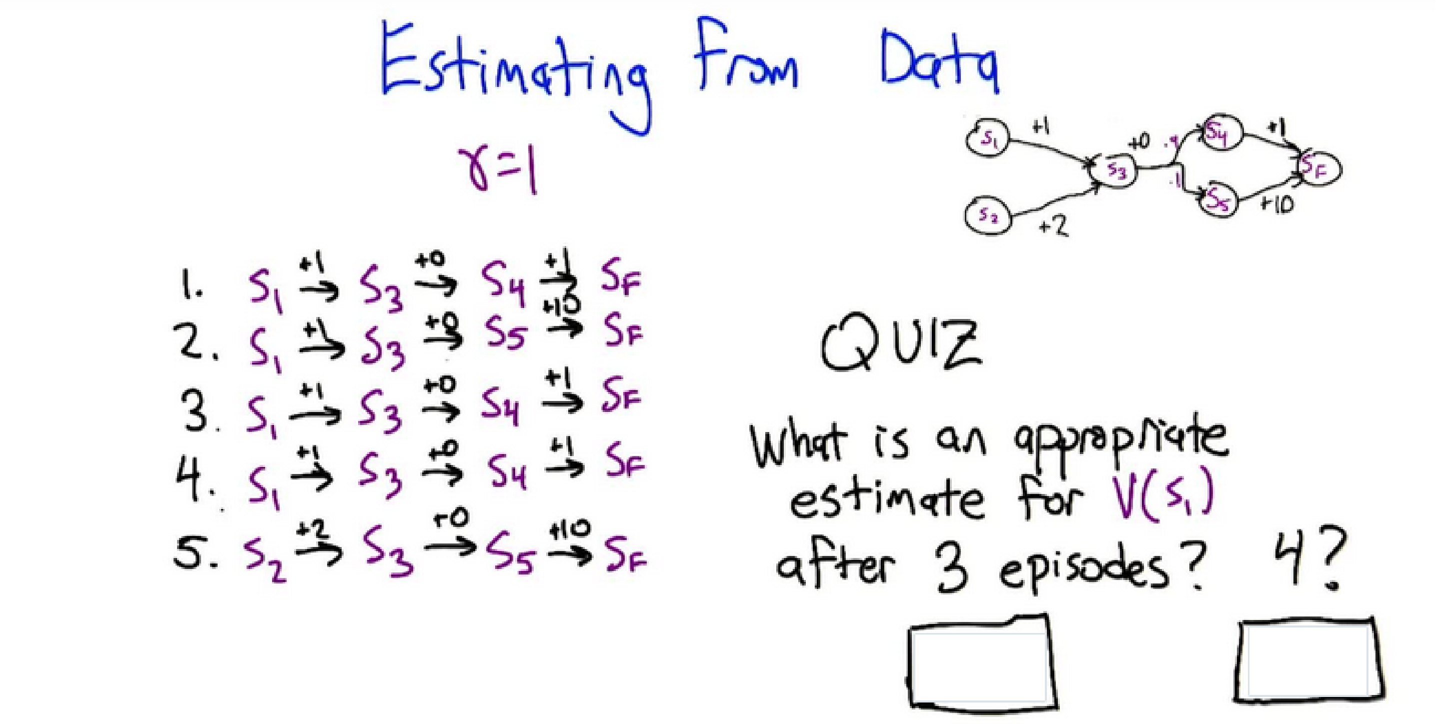
$$V(S) = \begin{cases} 0, S = S_F \\ E[R + \gamma V(S')], S! = S_F \end{cases}$$

and we have this markov chain:



here the prediction work is to work out $V(S_3) = ?$
accroding to above markov chain and terminal state description function, we can recursively get $V(S_3) = 1.9$

Predict with data: sequences of action and reward



Estimating From Data

$\gamma = 1$

1. $S_1 \xrightarrow{+1} S_3 \xrightarrow{+0} S_4 \xrightarrow{+1} S_F$
2. $S_1 \xrightarrow{+1} S_3 \xrightarrow{+0} S_5 \xrightarrow{+10} S_F$
3. $S_1 \xrightarrow{+1} S_3 \xrightarrow{+0} S_4 \xrightarrow{+1} S_F$
4. $S_1 \xrightarrow{+1} S_3 \xrightarrow{+0} S_4 \xrightarrow{+1} S_F$
5. $S_2 \xrightarrow{+2} S_3 \xrightarrow{+0} S_5 \xrightarrow{+10} S_F$

QUIZ

What is an appropriate estimate for $V(S_1)$ after 3 episodes? 4?

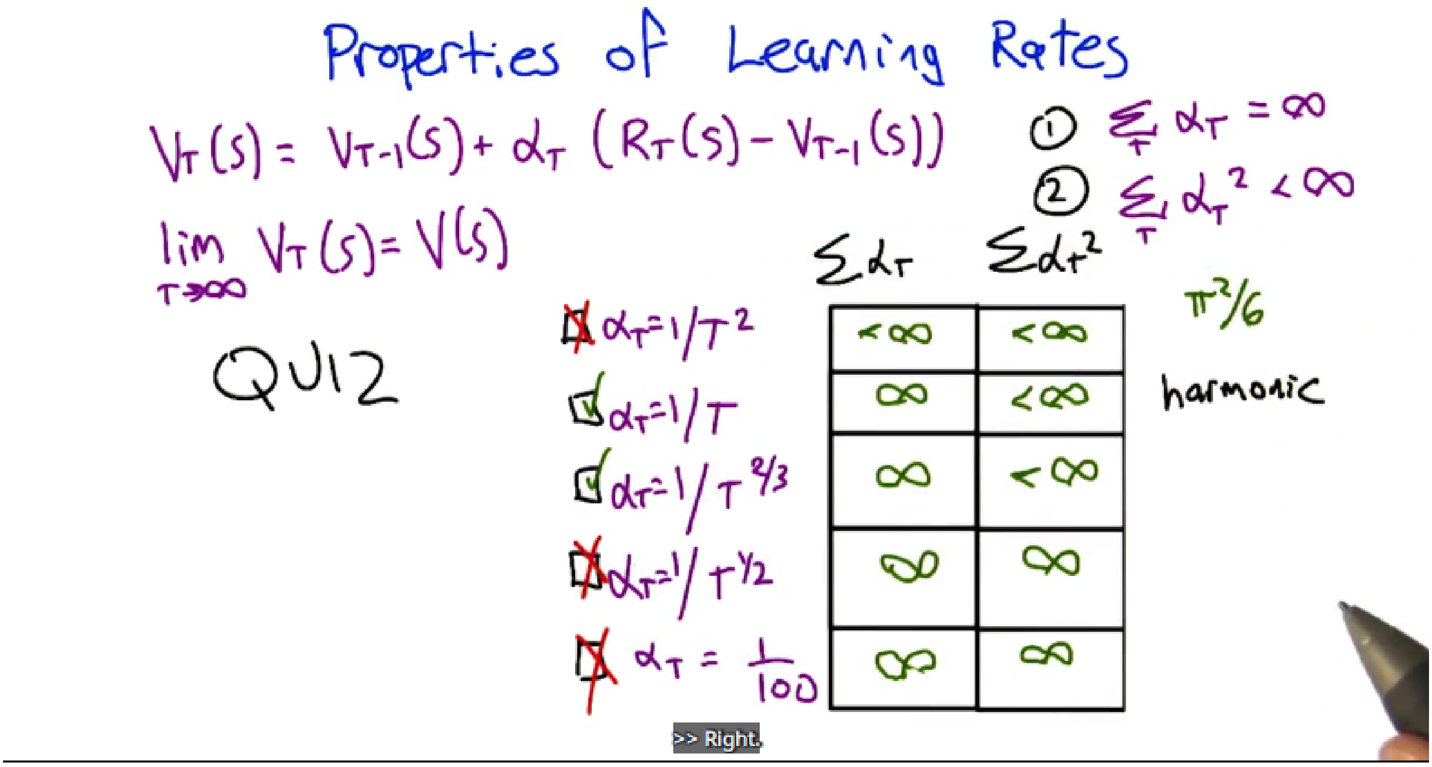
Above picture means that we use data to get same result of value of a state instead of use markov chain and corresponding function to derive.
What we know is immediately reward and a lot of episodes,
each episode is a sequence of state and each state transform takes it's reward.
So think about what is most easy way to predict data, it's average operation.
So after we getting 3 episodes, we can use below method:
$V(s_1)' = \frac{\sum_i^3 \sum R(S)}{3} = (2+11+2)/3 = 5$
After getting 4 episodes:
$V(s_1)' = \frac{\sum_i^4 \sum R(S)}{4} = (2+11+2+2)/4 = 4.25$
As more data get, we can found that value of state will approximate true value.

Computing Estimates Incrementally

Now we give you input as:
$V_{T-1}(S_1) = 5, R_T(S_1) = 2, V_T(S_1) = ?$
How do we estimate $V_T(S_1)$?
One simple way is use average:
We can think $V_{T-1}(S_1)$ is a averaged value by $(T-1)$ times, so the total value is $5 \times (T-1)$, and if we get a reward and transform to state S, predicted value need divided by T times:
$V_T(S) = \frac{V_{T-1}(S_1) \times (T-1) + R_T(S_1)}{T}$
$V_T(S) = \frac{T-1}{T} V_{T-1}(S_1) + \frac{1}{T} R_T(S_1)$
$V_T(S) = V_{T-1}(S_1) + \frac{1}{T}(R_T(S_1) - V_{T-1}(S_1))$
$\frac{1}{T} = \alpha_T$

Here α_T is called learning rate, it will be smaller and smaller as time going on, and $(R_T(S_1) - V_{T-1}(S_1))$ is an error shows how $R_T(S_1)$ effects the estimation.

Properties of learning rate



Properties of Learning Rates

$V_T(s) = V_{T-1}(s) + \alpha_T \left(R_T(s) - V_{T-1}(s)\right)$

① $\sum_T \alpha_T = \infty$

② $\sum_T \alpha_T^2 < \infty$

$\lim_{T \to \infty} V_T(s) = V(s)$

QUIZ

	$\sum \alpha_T$	$\sum \alpha_T^2$	
☒ $\alpha_T = 1/T^2$	$<\infty$	$<\infty$	$\pi^2/6$
☑ $\alpha_T = 1/T$	∞	$<\infty$	harmonic
☑ $\alpha_T = 1/T^{2/3}$	∞	$<\infty$	
☒ $\alpha_T = 1/T^{1/2}$	∞	∞	
☒ $\alpha_T = \frac{1}{100}$	∞	∞	

>> Right.

this photo shows one thing, converge or not converge.
so it will refer question about selecting learning rate.

TD(1) Update Rule

TD(0) Update Rule

TD(λ) Update Rule