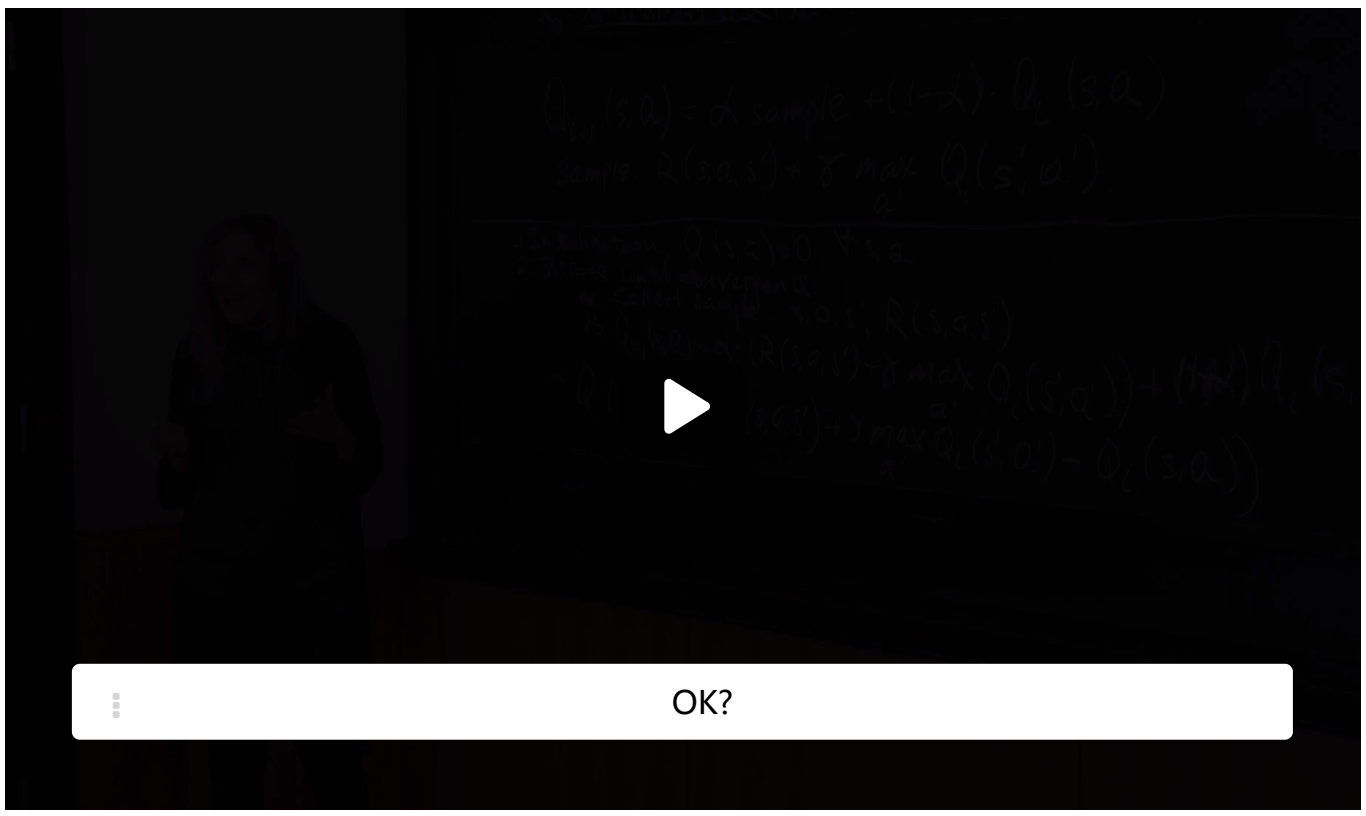
3. Q value iteration by sampling
Q value iteration by sampling



18:33 / 18:33 1.0x

So under this condition, this algorithm
is guaranteed to converge.
And this is pretty much it.
What this algorithm will let you do--
each time when you act in the world,
you incorporate evidence, and you incrementally improve
the Q's.
And when you are done, you can say, now, I have my policy.
OK?

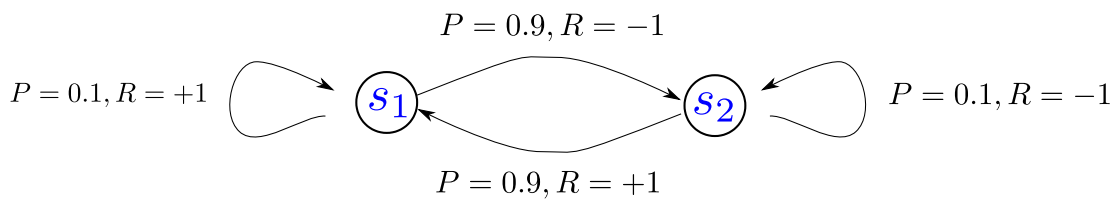
End of transcript. Skip to the start.

Video
Download video file

Transcripts
Download SubRip (.srt) file
Download Text (.txt) file

Let us consider a toy example which might not be very realistic but which neverthless can help delineate the Q-value iteration for RL using sampling approach.

For this example, assume that there are only two states, s_1, s_2 and only one action possible from each of these states. Let a_{s_1}, a_{s_2} be the actions that could be taken from s_1 and s_2 respectively.



The state transition probabilities are listed below and are also shown in the figure above.

$$T(s_1, a_{s_1}, s_1) = 0.1$$

$$T(s_1, a_{s_1}, s_2) = 0.9$$

$$T(s_2, a_{s_2}, s_2) = 0.1$$

$$T(s_2, a_{s_2}, s_1) = 0.9$$

The rewards for these actions are given by

$$R(s_1, a_{s_1}, s_1) = 1$$

$$R(s_1, a_{s_1}, s_2) = -1$$

$$R(s_2, a_{s_2}, s_2) = -1$$

$$R(s_2, a_{s_2}, s_1) = 1$$

Note that we resort to finding optimal Q^* function by sampling for tasks where we don't have access to the exact T, R functions. However, for this toy example we will assume that the Q-value iteration algorithm isn't directly provided with the above specified values of T, R and has to resort to sampling to estimate the Q function.

Let's say that the agent starts out from state s_1 and collects few samples. Each sample can be described by the following tuple $(s, a, s', R(s, a, s'))$which indicates that the agent received a reward of $R(s, a, s')$ when it reached state s' by taking action a from the state s.

The collected samples are described as follows in the order in which they are presented to the Q-value iteration algorithm.

$$(s_1, a_{s_1}, s_1, +1)$$

$$(s_1, a_{s_1}, s_2, -1)$$

$$(s_2, a_{s_2}, s_1, +1)$$

Let $S_k^{Q(s,a)}$ be used to denote the k^{th} sample of $Q(s, a)$ $(k = i + 1)$. Then recall that

$$\hat{Q}_{i+1}(s, a) = \alpha * S_k^{Q(s,a)} + (1 - \alpha) * \hat{Q}_i(s, a)$$

For all of the following problems, assume that the discount factor $\gamma = 0.5$, $\alpha = 0.75$ and that all the Q values are initialized to 0 to start with. That is,

$$\hat{Q}_0(s, a) = 0 \forall s, a$$

Numerical Example

1/1 point (graded)
Enter below the value of $Q(s_1, a_{s_1})$ after the first sample is processed by the Q-value iteration algorithm

0.75 ✔ **Answer:** 0.75

Solution:

Let $S_k^{Q(s,a)}$ be used to denote the k^{th} sample of $Q(s, a)$.

$$\begin{aligned} S_1^{Q(s_1,a_{s_1})} &= R(s_1, a_{s_1}, s_1) + \gamma * \max_{a'} Q(s_1, a') \\ S_1^{Q(s_1,a_{s_1})} &= +1 + 0.5 * 0 = 1 \\ Q_1(s_1, a_{s_1}) &= \alpha * S_1^{Q(s_1,a_{s_1})} + (1 - \alpha) * Q_0(s_1, a_{s_1}) \\ Q_1(s_1, a_{s_1}) &= .75 * 1 + (1 - .75) * 0 = .75 \end{aligned}$$

这里我们没有从s1做了a'后，任何Q的data，所以用我们的初始化的值，0.

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

Numerical Example - 2

1/1 point (graded)
Enter below the value of $Q(s_1, a_{s_1})$ after the second sample is seen by the Q-value iteration algorithm

-0.5675 ✔ **Answer:** -0.5625

Solution:

Let $S_k^{Q(s,a)}$ be used to denote the k^{th} sample of $Q\left(s,a\right)$. Note that from the previous example,

$$Q_1\left(s_1,a_{s_1}\right)=0.75$$

Now we find $S_2^{Q(s_1,a_{s_1})}$:

同样，这里我们没有从s2做了a'后，任何Q的data，所以用我们的初始化的值，0.

$$\begin{aligned}
S_2^{Q(s_1,a_{s_1})} &= R\left(s_1,a_{s_1},s_2\right)+\gamma*\max_{a'}Q\left(s_2,a'\right)\\
S_2^{Q(s_1,a_{s_1})} &= -1+0.5*0=-1\\
Q_2\left(s_1,a_{s_1}\right) &= \alpha*S_2^{Q(s_1,a_{s_1})}+(1-\alpha)*Q_1\left(s_1,a_{s_1}\right)\\
Q_2\left(s_1,a_{s_1}\right) &= 0.75*-1+0.25*0.75=-0.5625
\end{aligned}$$

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

Discussion

Show Discussion

Topic: Unit 5 Reinforcement Learning (2 weeks) :Lecture 18. Reinforcement Learning 2 / 3. Q value iteration by sampling