

From wiki:

Markov decision processes (MDPs) provide a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker. MDPs are useful for studying a wide range of optimization problems solved via dynamic programming and reinforcement learning. A Markov decision process is a 5-tuple  $(S, A, P(\cdot, \cdot), R(\cdot, \cdot), \gamma)$ .

## DECISION-MAKING & REINFORCEMENT LEARNING

1. Hi Michael. >> Hey Charles. How's it going? >> It is going quite well. Thank you very much for asking. How are things going with you? >> I can't complain because I've been barred by a, a judge. >> [LAUGH] I'm pretty sure you can still complain. Complaining is a fine art that requires lots of practice. Guess what we're going to do today? >> I'm reading decision making and reinforcement learning which is very exciting. >> It is. >> Except the hyphen. >> Except for the hyphen? >> Yeah the hyphens wrong. >> Why? >> because it's not decision dash making. >> You're right. I normally don't do that but people always want to put a dash so what I'm going to do is for your Michael. Just for you. I'm going to remove the dash. >> Thank you. >> And I'm going to put it over here where it belongs. >> [LAUGH] No. There. Well, Michael, this is the first of our discussions on the last mini course of machine learning, Reinforcement Learning, which I believe you know a little bit about. >> I am very interested in Reinforcement Learning. >> Excellent. So I'm just going to do a little bit of background. It's going to be relatively short and straight forward. We're going to do a couple of quizzes, because I know how much you like quizzes. And then we're just going to go back and forth and see what we get to learn about Reinforcement Learning. Sound good? >> Excellent. >> Excellent, okay so, let's get started.

## DECISION MAKING & REINFORCEMENT LEARNING

SUPERVISED LEARNING :  $y = f(x)$  function approx

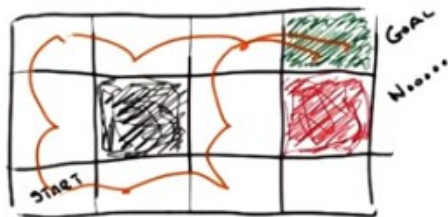
UNSUPERVISED LEARNING :  $f(x)$  clustering description

REINFORCEMENT LEARNING :  $y = f(x)$   
 $z$

2. Okay Michael so before we get started diving into a particular formalism or framework that I want to

talk about, that we are going to use for this mini course or at least the first half or so of it. I want to remind everybody what the differences are between the three types of learning that we said we are going to look at over this entire course. Th, you recall what they are? >> Well reading off the slides supervised, unsupervised and reinforcement. >> That's right. And you know, again these things are all strongly related to one other but it's very useful to think about them as being very separate. So just as a reminder, supervised learning sort of takes the form of function approximation where you're given a bunch of  $x, y$  pairs And your goal is to find a function  $f$  that will map some new  $x$  to a proper  $y$ , you recall that? >> Yep. >> Unsupervised learning is very similar to supervised learning except that it turns out that you're given a bunch of  $x$ 's and your goal is to find some  $f$ . That gives you a compact description of the set of  $x$ 's that you've seen. So we call this clustering, or description as opposed to function approximation, and finally we're getting to reinforcement learning. Now reinforcement learning is actually a, a name that means many things in different fields, and here we tend to talk about it in a relatively specific way, and superficially it looks a lot like Supervised learning, in that we're going to be given a string of pairs of data, and we're going to try to learn some functions. But in the function approximation case, a supervised learning case, we were given a bunch of  $X$  and  $Y$  pairs. We were asked to learn  $f$ , but in reinforcement learning, we were given something totally different. We're instead going to be given  $x$ 's and  $z$ 's, and I'll tell you what the  $x$ 's and the  $z$ 's stand for in a minute, and then we're going to have to learn some  $f$  that is going to generate  $y$ 's, and so even though it's going to look a lot like function approximation, it's going to turn out that it's going to have a very different character. And that's really what the, the next few slides in a little bit of discussion is really about, is understanding what that character is. You'll also notice from the title here that I have decision making reinforcement learning, and that's because reinforcement learning is one mechanism for doing decision making. And again, I'll define that in just a second. Okay, so you with me, Michael? >> I think so. So should  $y$  be circled? because in some sense, you, you underlined the things we were given and you circled the things we needed to find, so  $y$  is something that we're going to find, right? >> Yeah, I suppose that I like that. >> All right. Interesting. >> Now it's a  $\theta$ . >> [LAUGH] It's a  $Y$  trapped inside of a  $\theta$  and it's yelling,  $Y$ ? >> [LAUGH] I like that. I'm a  $Y$  trapped in a  $\theta$ . Hm, I need to write a book about that. Okay, good. That's a good point Michael. So before we were learning, effectively trying to learn one thing and here we're still learning one thing. Because it's going to produce another thing for the deterministically, usually. But it's worth pointing out that we are going to be figuring how to produce both of these things as opposed to being given those things. Good job. OK. Are you ready to move forward, Michael, with an example and a quiz? >> Awesome. >> Excellent.

# THE WORLD



UP, DOWN, LEFT, RIGHT

Quiz!

WHAT IS THE SHORTEST  
SEQUENCE GETTING  
FROM START TO GOAL?

UP UP RIGHT RIGHT RIGHT  
R R U U R

3. Michael, here is a world. In fact, for the purposes of this discussion it is the world. Okay. So imagine the entire universe is well described by this picture over here. Okay? Now **this is called a grid world, which is something that people in reinforcement learning love to think about**, because it's a nice approximation for all the complexity of the entire universe. Now this particular world is a three by four grid. So you have one comma one, two comma one, three comma one, four comma one. You have one comma two, one comma three and so on and so forth. For the purpose of this discussion we can think of the world as being a kind of game where **you start out in state. Which we're going to call the start state. And your able to execute actions, one of these four, up, down, left to right and the purpose is to wonder around this world in such a way that eventually you make it to the goal over here, this little green spot.** You see that Michael? >> Yep. >> **And under all circumstances, you must avoid the red spot.** >> No. >> Exactly. Now for this particular example, up does what you think it does, down does what you think it does, as do left and right. **But if you find yourself at a boundary, such as right up here in the upper-left-hand corner, and you try to go up, you just stay where you are. If you try to go left, you just stay where you are. But if you go right, you do actually end up in the next square.** Got it? >> Think so. >> Okay. Three last things. One, **this little black space here, is a place you can't enter into, so it acts just like a wall. This green space is the goal, and once you're there it's over. The world is over and you get to start over again.** >> Hm. >> **And once you enter into the red spot, the world is also over and you have to start over again. So you can't go through the red square to get to the green square.** Okay, you got it? >> Yeah. >> Excellent. So, here's the quiz. Given this particular world, with the physics I just described to you, and given these actions that you can take, up, down, left and right, what is the shortest sequence of actions that would get us from the start state to the goal state? And you can just type in the words up, down, left and right, separated by spaces, or commas or anything like that. Okay. [LAUGH] Say semicolons are allowed, colons are not. >> Yeah, good. I am glad you made that distinction. >> Well, it's an important one to make. Okay, you got it? >> Yeah, I think so. >> Alright, cool. So lets see what you get then. Go.

4. >> Okay Michael, you go an answer for me? This isn't meant to be a hard quiz. >> Oh good. Cause **I actually have two answers** for you. >> Oh, I like that. >> So I would say right, right, up, up, right. >> Okay, so, right, right, up, up, right. >> But I feel like I could have also said up, up, right, right, right. >> And you could have. Both of those are acceptable. right, right does what you think it does. You move right and then right and then up and then up and then right. And that takes five steps. Or you

could have gone up, up, right, right, right, which would also have taken you five steps. So that's pretty easy, right? >> Yeah, I thought so. >> Rag. So yes, either of these would be correct. **We're going to stick with this one in particular, since they're equal, because we gotta pick one of them.** So this does point out something that often in a decision problem like this, there are in fact multiple answers or multiple optima, if I can tie it back to our randomized optimization discussions. Okay, so you got the quiz, you got the world, you understand what you can do here. Yes? >> Yep. >> Okay, cool. Now I'm going to throw in a little wrinkle.

THE WORLD

Goal

Start

UP, DOWN, LEFT, RIGHT

- ACTIONS EXECUTES .8
- MOVE AT RIGHT ANGLE .1 + .1

Quiz!

WHAT IS THE RELIABILITY OF OUR SEQUENCE

UP UP RIGHT RIGHT RIGHT ?

.32776

$.8^5 = .32768$

$.1^4 \cdot .8 = .00008$

.32776

注意題目中已經給了 sequence: up up right right

5. I'm going to change the world just a little bit Michael. Okay, that last one was really easy, and it was easy for a bunch of reasons, but one of the reasons it was easy is that every time it took an action it did exactly what you expected it to do. **Now I want to introduce a little bit of uncertainty, or stochasticity into the world.** So, let me tell you exactly what that means. **When you execute an action, it executes correctly, with probability of 80%. So 80% of the time, if you go up, it goes up.** Assuming you can go up. If it goes, if you say down, it goes down, assuming you can. Left, it goes left; right, it goes right. Got it? Yeah! >> **Now, 20% of the time, the action you take actually causes you to move at a right angle.** Now, of course, there are two different right angles you could go to, **if you go up, you could go either left or right** at a right angle. And so that 20% gets distributed uniformly. Okay? Does that make sense? >> Yeah, I think so. And so if you, if you're in the start state and you try to go up, then there a 10% chance that you tend to bump into the wall. >> Yes. >> And then you just stay where you are I guess. >> Right. So if you, **if you decide that you'd have x here, you have a 80% chance of moving up, you have a 10% chance of moving to the right. And you have a 10% chance of moving to the left** but, of course, you'd bump a wall and you would end up right back where you started. Got it? >> Yeah. >> Okay, good. >> So, here is my quiz question for you Michael. You recall, we came up with two sequences and the one that I decided to keep was up, up, right, right, right. My question to you, is, what is the reliability of the sequence, up up, right right, actually getting you from the start to the goal, given

these probabilities, this uncertainty? >> And so, we're just going to try that sequence, and ask whether or not it actually got us to the goal. >> Exactly. And [when I say, reliability, I really mean, what's the probability of it actually succeeding?](#) Interesting. Okay. >> Alright. So let's see if we can figure out the answer. You're ready? >> I'm ready. >> Alright. Go!

6. Okay Michael do you have an answer? >> I have an answer indeed. >> Okay, what's the answer? >> Okay, maybe I don't have an answer indeed. But I have I have the ability to compute one. >> Okay. >> I'm, I'm doing some math. >> So why don't you walk me through the math you're doing. >> I got .32776. >> That is correct Michael. >> Woo-hoo. That was trickier than I thought. >> Okay well, show me what you did. >> Alright, so [the first thing I did is I said, okay, well this is not so hard, because, from the start-state, if I execute up, up, right, right, right. Each of those things works the way it's supposed to do independently with probability .8.](#) >> Yep. And [so .8 raised to the 5th power, gives me the probability that that entire sequence will work as intended.](#) >> Exactly. And what is .8 to the 5th? Do you know? >> 32,768. with a decimal point in front of it, because you know, powers of 2. >> Wow. That is correct. But I notice that 32768 is not 32776. No they differ by a little smidge. >> Mm-hm. >> So this is [what occurred to me next is, is there anyway that you could have ended up falling into the goal from that sequence of, of commands not following the intended path.](#) So since actions can have unintended consequences, as they often do. >> Mm-hm. >> I was going to ask okay, so [if I go up in the first step, there's a probably .1 that I'll actually go to the right.](#) >> Yep. From there, [if I go up, there's a .1 probability that I'll actually go to the right.](#) >> Mm-hm. >> From there, [the next thing I do is take the right action, but that can actually go up with some probability, .1.](#) >> Mm-hm, yep. >> [And then another .1 to get to for the next right action to actually cause an up to happen.](#) >> Mm-hm. [And then finally, that last right might actually execute correctly and bring me into the goal. So I could go underneath the barrier, instead of around the barrier with that same sequence.](#) It just isn't very likely. >> Okay. Well how unlikely is it? >> Alright. So I did .1 times .1 times .1 times .1 times .8. huh, so the .1 to the 4 times .8 and that's right so this is the probability that the, 4 of the sequences, 4 of these go wrong. In fact exactly the first 4 go wrong. And the last one goes right. Right. And that's equal to, some very, very small number. And when you add it up. You end up with .32776. In fact that's equal to 0.00008. And that's how you get that number and that's correct. And you'll notice that this [this sequence happens to work out to be the second sequence that, we had options for.](#) Yeah, the other thing that took us to the goal, right. >> Yeah. >> Was exactly the probability of executing that action executing that sequence of transitions given the first command. >> Yeah, and [I think it would work the other way,](#) too. No, it wouldn't quite work the other way. If you had done the sequence right, right, what is it? Right, right. Up, up right? >> Yeah. >> In order for that one to work out, you'd add .8 to the 5 and working. And, for it to work out wrong but work out right, the right would have to send you up and then the ups would have to send you right, and then, yeah, so it would actually work out to be the same. >> Yeah. >> [So no matter which of the two sequences you came up with, they would have the same probability of succeeding.](#) Neat. >> Nice. That's actually kind of cool. So, good job, Michael. Very good job on the quiz. A lot of people forget this part. And, in fact, if you forgot that part, and got, just this part right, we actually let you pass. But it was wrong. >> I was kind of expecting you to get it wrong, but I'm glad you got it right too. >> Thank you Michael, I appreciate your faith in me. >> [LAUGH] >> I wrote the quiz. Actually, I stole this from a book, >> Oh. which, whose little words are now showing up in front of you, exactly where you can go through the details of this quiz. Okay, alright, Michael. So you might ask yourself why I brought this up, and the reason I brought this up is because, what we did in the first case where we just came up with the sequence up up right right right, is we sort of planned out what we would do in a world where nothing could go wrong. But once we introduced this notion of uncertainty, this, this randomness, this stochasticity, we have to do something other than work out in advance what the right answer is, and then just go. We have to do something a little bit more complicated. Either we have to try to execute these, and then every once in a while, see if



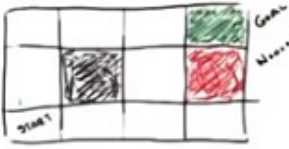
we've drifted away. And then re-x, re-plan, come up with a new sequence wherever it is we happen to end up, or we have to come up with some way to incorporate all of these uncertainties and probabilities. So that, we never really have to think, rethink what to do in case something goes wrong. So, what I'm going to do next is I'm going to introduce the framework, that's very common, that people use as a way of capturing this stuff, capturing these uncertainties directly. Okay? >> Hm. >> You ready? >> Yeah. >> Excellent.

# Markov Decision Processes

(1,1) (4,4)

1 2 3 ... 12

FRID MARCUS



STATES : S

MODEL :  $T(s, a, s') \sim \Pr(s' | s, a)$  UP, DOWN, LEFT, RIGHT

ACTIONS :  $A(s), A$

REWARD :  $R(s), R(s, a), R(s, a, s')$

---

Policy :  $\pi(s) \rightarrow a$

$\pi^*$

2:15 / 2:16

最後一行是 Policy, 而不是 Poucy

7. So this is the framework that we're going to be using through most of the discussions that we'll be having at least on reinforcement learning. The single agent reinforcement learning, and it's called the Markov Decision Process. This should sound familiar to you, Michael. >> Well, you did say we're going to talk about decisions. >> That's true, and we need a process for making decisions. And we're going to introduce something called the Markovian property as a part of this discussion and I'll tell you exactly what that means in a moment. So, I'm just going to write out this frame work and just, and tell you what it is and what the problem it produces for us. And then we're going to start talking about solutions through the rest of the discussion. So a Markov Decision Process tries to capture worlds like this one by dividing up in the following way. We say that there are states. And states are a set of tokens that somehow represent every state, for lack of a better word, that one could be in. So, does that make sense to you, Michael? >> Yeah, I think so. >> So what would the states be in the world that we've been playing around in so far? >> So, the only thing that differs from moment to moment is where, I guess, I am. >> Mm-hm. >> Like, which grid, grid position I'm in. >> Right. >> So, I feel like each different grid position I could be in is a state, maybe there's a state for being successfully done or

unsuccessfully done? >> It's possible. But let's stick with the simple one. I like that one because that's really, I think, easy to grasp. So, there are at least, of all the states one could reach, there's, well let's see there's four times three minus one, since you can never reach this state. Although we could say it is a state we just happen to never reach it. So, at most if we just think of this grid literally as a grid there are something like twelve different states. And we can represent these states as their X,Y coordinates, say. We could call this, the start state as say 1,1, which is sort of how I described it earlier. We could describe the goal state as 4,4. And say this is how we describe our states. Or frankly, it doesn't matter. We could call these states 1,2,3 up to 12. Or we could name them Fred and Marcus. It doesn't really matter. The point is that they're states, they represent something, and we have some way of knowing which state we happen to be in. Okay? >> Sure. >> Okay.

## Markov Decision Processes



STATES : S  
 MODEL :  $T(s, a, s') \sim \Pr(s' | s, a)$  UP, DOWN, LEFT, RIGHT  
 ACTIONS :  $A(s), A$   
 REWARD :  $R(s), R(s, a), R(s, a, s')$

POLICY :  $\pi(s) \rightarrow a$

$\pi^*$

(本圖跟上圖是一樣的)

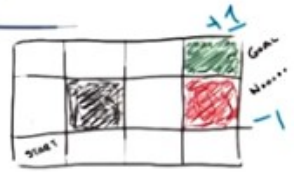
8. Alright so we've got states. So next is what's called the model or the transition model. Sometimes people refer to it as the transition function, and basically it is a function of three variables. It's a state, an action, which I haven't defined for you yet and another state. In fact since I haven't defined what an action is for you yet, let's skip that and actually do define actions for you. So the third part of our model are actions. Actions are the things that you can do in a particular state. So what would the actions be in this world? >> Well the four different decisions I could make were the up, down, left and right. Though it's maybe a little confusing because I think those were also the four possible outcomes. That's true. Well no, there are other outcomes. You could stay where you were. >> Right. >> Right. >> So these are actually the actions. These are the things that when I'm in a given state I'm allowed to execute. I can either go up, go down, go left or go right. You'll notice that as I described before there was no option not to move. Mm. >> But there could have been, and there could have been other actions, like teleport, or there's anything that you can imagine. But the point is that your action set represents all of

the things that the agent, the robot, the person, or whatever it is you're trying to model, is allowed to do. Now, in its full, generalized form, we can think of the set of actions that one can take as being a function of state, because there's some stage(就是說的 stage, 不是 state) you can do some things and some stage you can't do those things. But most of the time people just treat it as a set of actions. And the actions that aren't allowable in them particular states have no effect. Alright, so we understand states. They're the things that describe the world. We understand actions. Those are the things that you can do, the commands you can execute when your in particular states. And what a model describes, what the transition model describes is in some sense the rules of the game that you're playing. It's the physics of the world. So it's a function of three variables: a state, an action and another state, which, by the way, could actually be the same state. And what this function produces is the probability that you will end up transitioning the state  $s'$ , given that you were in state  $s$ , and you took action  $a$  (例如前面的 想往前走, 但實際往走到前面那一格的概率為 80%). Got it? >> I think so. so the  $s$  prime is where you end up, and the  $s$ ,  $a$  is what you're given. So it's, so you plug these three in. Oh I see, and you get a probability Mm-hm. >> But the probabilities have to add up to one if you if you sum it up over all the  $s'$ . >> Right. That's exactly right. So for example if you think about the deterministic case where there was no noise then this is a very simple model. If I'm in the state the start state. And I take the action up, then what's the probability, I end up in the state immediately above it? >> Was that a 0.08? >> No, in the, in, when we first started in the deterministic world. >> Oh, that was probability one. >> Right, and what would be the probability, you ended up in the state to the right? Probability zero. >> Right. In fact, the probability that you end up at any of the other states is zero in the deterministic world. Now what about the case when we were in the non-deterministic world where an action would actually execute faithfully only 80% of the time. If I'm in the start state and I go up, what's the probability that I end up. In the state above. >> That was 0.8 >> Was the probability, I end up in the state to the right. >> That was 0.1. >> And, what was the probability I end up where I started. >> That was also 0.1. >> Right, and 0 everywhere else. And, that's just sort of the way it works. So, the model really is an important thing. And the reason, it's important. Is it really does describe the rules of the game. It tells you what you, what will happen if you do something in a particular place. It captures Everything that you know about the transition  $u$ , of the world is what you know about the rules, got it? >> You called it physics before? >> I called it the physics of the world (think of Standard Model). >> Huh. >> These are the rules that don't change. >> But they're very different from real world physics. >> Well, yeah, although they don't have to be. I mean in some sense, you could argue that a mark off decision process, what we described so far, these three things In fact, do describe the universe, the states are, you know, in the positions of all the atoms. The positions and velocities of all the atoms in the universe. The transition miles as you do, take certain positions in the world whatever they are how the state of the universe changes in response to that. And the actions or whatever those set of actions could be. And it can be probabilistic or it's not probabilistic. >> It's definitely probabilistic. The transition models are by their very definition probabilistic. >> Gotcha.



# Markov Decision Processes

↳ only present matters!  
stationary



✓ STATES :  $S$

✓ MODEL :  $T(s, a, s') \sim \Pr(s' | s, a)$  UP, DOWN, LEFT, RIGHT

✓ ACTIONS :  $A(s), A$

✓ REWARD :  $R(s), R(s, a), R(s, a, s')$

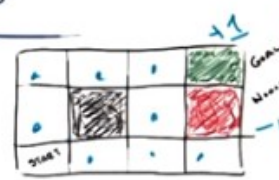
Policy :  $\pi(s) \rightarrow a$

9. Now, I actually snuck something important in here, I actually snuck two things that are important in here. The first is, the, what's called the Markovian property. Do you remember what the Markovian property is Michael? >> From what? >> From, I dunno. Actually, where does the Markovian property come from? >> I'm going to say Russia. >> Okay, yeah, from the, from the Russian. >> Yeah, so I have Russian ancestors, and they passed onto me this idea that. Markov means that you don't have to condition on anything passed the most recent state. >> That's exactly right. >> **The Markovian property is that only the present matters.** >> And they had to pass that down to me you know, one generation at a time, because you know, Markov property. >> Exactly right, that was very good Michael. So what does this mean? **What this means is our transition function (即 model), which shows you 'the probability you end up in some state S', given that you're in state S and took action A' only depends upon the current state S. If it also depended upon where you were 20 minutes before then, you would have to have more S's here. And then you would be violating the Markovian property.** >> So is this like, do historians hate this? >> Well, you know, one never learns anything from history. >> No, you're supposed to learn from history, or you're doomed to, I don't know, let me make something up, repeat it. >> [LAUGH] Fair enough. **Historians probably don't like this, but there is a way for mathematicians to convince them that they're okay with it. And the way that, you, mathematicians convince you that you're okay with this is to point out that you can turn almost anything, into a Markovian Process by simply making certain that your current state remembers everything you need to remember from the past.** >> I see. >> So, in general, even if something isn't really Markovian, you need to know what you were, not only what you're doing now, but what you were doing five minutes ago. You could just turn your current state into what you're doing now and what you were doing five minutes ago. The obvious problem with that, of course, is that if you have to remember everything from the beginning of time. You're only going to see every state once and it's going to be very difficult to learn anything. But, that Markovian property turns out to be, turns out to be important and it actually allows us to solve these problems in a tractable way. But I snuck in something else, Michael. What I snuck in is this idea about the transition model, is that nothing ever changes. So this **second property that matters for Markov decision processes**, at least for the sets of things that we're going to be talking about in the beginning, **is that things are stationary.** That is these for the purposes of this discussion, it

means that these rules don't change over time. That's one notion of stationary, okay? >> Does that mean that the agent can never leave the start state? >> No, the agent can leave the start state any time it takes the action, then it gets another start state. >> Then how is it, how is it stationary, then? >> It's not stationary, the world is stationary. The, the transition model is stationary, the physics are stationary, the rules don't change any. >> The rules don't change. >> Right. >> I, I, okay. I see. >> Then there's another notion of stationary that we'll see a little bit later. Okay, last thing to point out about the definition of a mark on decision process, is the notion of reward. So, reward is simply a scalar value, that you get for being in a state. So for example we might say, you know, this green goal is a really good goal. And so, if you get there, we're going to give you a dollar. This red goal, on the other hand, is a very, very bad state. We don't want you to end there. And so, if you end up there we're going to take away a dollar from you. >> What if I don't have a dollar? >> Someone will give you the dollar. The universe will give you the dollar. >> [LAUGH] And then take it away? >> Or, the universe will take it away. Even if you don't have it. You'll have negative dollars. >> Oh, man. >> Okay, so Reward is very important here in a of couple ways. Reward is one of the things that, as I'm always harping on encompasses our domain knowledge. So, the Rewards you get from the state tells you the usefulness of entering into that state. Now. I wrote out three different definitions of R here, because sometimes it's very useful to think about them differently. I've been talking about the reward you get for entering into the state (即  $R(s)$ ), but there's also a notion of reward that you get for entering into a state and taking an action (即  $R(s, a)$ ). There's a reward, or, being in a state and taking an action, there's a reward that you could get for being in a state, taking an action, and then ending up in another state  $S'$  (即  $R(S, a, S')$ ). It turns out these are all mathematically equivalent. But often it's easier to think about one form or the other. But for the purposes of the, you know, for the rest of this discussion, really you can just focus on that one (即  $R(s)$ ), the reward of the value entering into a state. And those four things (states, model, actions, reward), by themselves, along with this Markov property and non-stationarity, defines what's called the Markov Decision Process. Or an MDP. Got it? >> I'm a little stuck on the, how those could be mathematically equivalent. >> Well we'll get to that later. Would you like a little bit of intuition? >> Sure. >> Well, you can imagine that just as before we were dealing with the the notion of making a non-Markovian thing Markovian by putting a little bit of history into your state. You can always fold in the action that you took to be in a state or the action that you took to get to a state, as a part of your state. >> But that would be a different Markov Decision Process. >> It would, but they would work out to have the same solution. >> Oh, I see.

# Markov Decision Processes

↳ only present matters!  
↳ stationary



STATES :

MODEL:  $T(s, a, s') \sim \Pr(s' | s, a)$

ACTIONS : A(s), A

✓ ACTIONS:  $R(s)$ ,  $R(s, a)$ ,  $R(s, a, s')$

$$\begin{aligned}\pi(\text{START}) &\rightarrow \text{UP} \\ \pi(s) &\rightarrow ?\end{aligned}$$

→ Policy:  $\pi(s) \rightarrow a$   $\langle s, a \rangle \langle s, a \rangle \dots$   
 $\pi^*$   $\langle s, a, r \rangle \langle s, a, r \rangle$

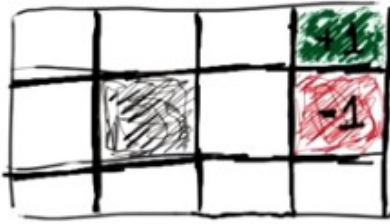
10. Speaking of solutions, this is the last little bit of thing that you need to know. And that is. This (即框中的 states, model, actions, reward 這四個東西) defines a problem. But what we also want to have, whenever we have a problem, is a solution. So, the solution to the Markov Decision Process, is something called a policy. And, what a policy does. Is, it's a function, that takes in a state, and returns an action, in other words, for any given state that you're in, it tells you the action that you should take

>> Like as a hint? >> No, it just tells you, this is the at. Well, I mean, I suppose you don't have to do it, but the way we think about Markov Decision Processes, is that this is the action that will be taken. >> I see, so it's more of an order. >> Yes, it's a command. Okay. >> So that's all a policy is. A policy is solution to a Markov Decision Process. And there is a special policy, which I'm writing here as policy star, or the optimal policy, and that is the policy that maximizes your long-term expected reward. So if all the policies you could take, of all the decisions you might take, this is the policy that optimizes the amount of reward that you're going to receive or expect to receive over your lifetime. >> So, like, at the end? >> Well, at yeah, at the end, or at any given point in time, how much reward you're receiving. >From the Markov Decision Process point of view, there doesn't have to be an end. Okay. Though in this example, you don't get anything, and then at the end, you get paid off. >> Right, or unpaid off. >> Right. >> If you fall into the red square. So actually, your question points out something very important here. I mentioned earlier when I talked about the three kinds of learning that there, supervised learning and reinforced learning were sort of similar, except that instead of getting Ys and Xs we were given Ys and, Xs and Zs. And this is exactly what's happening here. Here what we would like to have if we wanted to learn a policy is a bunch of <S, a> pairs as training examples. Well here's the state and the action you should've took, taken, here's another state and the action you should've taken, so on and so forth (即實例). And then we would learn a function, the policy, that maps states to actions (即總結出來的, 可以做 prediction). But what we actually see in the reinforcement learning world, in the Markov Decision Process world, is we see states, actions, and then the rewards that we received. And so in fact, this problem of seeing a sequence of states, actions, and rewards. It's very different from the problem of being told: this is the correct action to take to maximize a function, or find a function that maps from state to action. Instead, we say well, if you're in this state, and you take

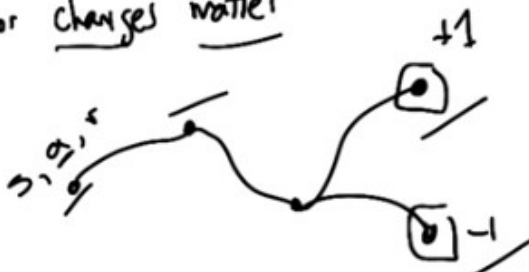
this action, this is the reward that you would see. And then from that, we need to find the optimal action. >> So  $\pi^*$  is being the  $f$  from that previous slide? >> Right. >> And  $R$  is being  $Z$ ? >> Yes. And  $Y$  is being  $a$ . >> And  $S$  is being  $X$  or  $x$  is being  $s$  >> Got you. >> Right. >> So but, I'm, okay I'm a little confused about this notion of a policy. So we have the, the, the thing we tried to do to get the goals was up, up, right, right, right. Yes. >> I don't see how to capture that as a policy. >> It's actually fairly straightforward. What a policy would say is: What state are you in? Tell me what action you should take. So, the policy, basically is this: When you're in the state, start, the start state, the action you should take is up. And it would have a mapping. For every state that you might see, whether it's this state, this state, this state, this state, this state, this state, this state, or even these two states, and it will tell you what action you should take. And that's what a policy is. A policy, very simply, is nothing more than a function that tells you what action to take at every, in any state you happen to come across. >> Okay, but the, but the. The question that you asked before was about up, up, right, right, right. >> Mm hm. >> And, it seems like, because of the stochastic transitions. You might not be in the same state. Like, you don't know what state you're in, when you take those actions. >> No, so, one of the things for what we're talking about here, for the Markov Decision Process. Is, there're states, there're actions, there're rewards. You always know what state you're in, and you know what reward you receive. >> So does that mean you can't do up, up right right right? >> Well, the way it would work in a Markov Decision Process, so what you're describing is is what's often called a plan. You know, it's, tell me what sequence of actions I should take from here. [What Markov Decision Process does and what a pr, a policy does is it doesn't tell you what sequence of actions to take from a particular state. It tells you what action to take in a particular state.](#) You will then end up in another state because of the transition model, the transition function. And then when you're in that state you ask the policy what actions should I take now? Okay. >> Right, so this is actually a key point. Although we talked about it in the language of planning, which is very common for the people who do, for example take any ag course, the thing about this in terms of planning, what are the things that I can do to accomplish my goals? [The Markov Decision Process way of thinking about it, the reinforcement way of thinking about it, or the typical reinforcement learning way of thinking about it, really doesn't talk about plans directly. But instead, talks about policies. Which, from which you can infer a plan, but this has the advantage that it tells you what to do everywhere.](#) And it's robust to the underlying stochastic of the world. >> So, is it clear that's all you need to be able to behave well. >> Well, it's certainly the case, that if you have a policy and that policy is optimal, it does tell you what to do, no matter what situation you're in. >> 'Kay. >> And so, if you have that, then that's definitely all you need to behave well. But I mean could it be that you wanted to do something like up, up, right, right, right which you can't write down as a policy? >> And why can't you write that down as a policy? >> Because the policies are only telling you what action to do as a function of the state not sort of like how far along you are in the sequence. >> Right unless, of course, you fold that into your state some how. But that's exactly right, the way to think about this is. The idea of coming up with a concrete plan of what to do for the next 20 time steps is different from the problem of whatever step I happen to be in, whatever state I happen to be in, what's the next best thing I can do? And just always asking that question. >> Hm. >> If you always ask that question, that will induce a sequence, but that sequence is actually dependent upon the set of states that you see. Whereas in the other case where we wrote down a particular policy, you'll notice that was only dependent upon the state you started in and it had to ignore the states that you saw along the way. >> And the only way to fix that would be to say, well, after I've taken an action, let me look at the state I'm in and see if I should do something different with it. But if you're going to do that, then why are you trying to compute the complete set of states? Or I'm sorry, the complete set of actions that you might take. >> Okay. >> Okay, so there you go. Now, a lot of what we're going to be talking about next Michael, is, given that we have MDP, we have this Markov Decision Process defined like this. How do we go from this problem definition to finding a good policy, and in particular, finding the optimal policy? That makes sense. >> Good. And there you go.



## MDPs: More ABOUT REWARDS



- delayed reward
- minor changes matter



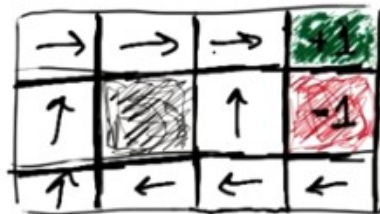
TEMPORAL  
CREDIT  
ASSIGNMENT

11. Okay, so, I want to talk about two things on this little slide, here. The first one is kind of a general observation about rewards and what makes the reinforcement learning MDP problem different from the supervised learning problems that we did before. And that's the notion of not just rewards, but this notion of delayed rewards. So, what do I mean by that. Well, so, if you think about the way we've set up kind of problem like, like we have here where we're trying to. Start in this little bottom left-hand square, and wind our way up into this plus one. There's really this notion of sequences of action. So in the reinforcement learning context, and all the things that we're talking about, at least in the foreseeable future, there's this sort of problem where you take some action and that puts you in some place and then you take another action and that puts you in some place. And then you take another action and that puts you in some place and maybe it puts you at a place where you get plus one. Or maybe it puts you at a place where you get minus one. And what really is going on here is this idea that you take actions that will set you up for other actions that will set you up for other actions. And only then do you know how good those particular actions you took were. So this reward is not just an idea of getting a reward at every state, it's an idea of getting delayed reward. So you don't know how your immediate action is going to lead to things down the road. So, let me give you a concrete example about that. So, have you ever played chess? >> Sure. >> So let's say we played a long game of chess and maybe took 60, 61, 62 moves. And then at the end, I win the game. So, what do you know about the way you played that game? >> That I probably made a bad decision around the time when I decided to play chess against you. >> That's possible, but maybe you made a good decision and you kept making good decisions, and you only messed up at the very last move, when you could have mated me but you didn't. >> I see. Well you, what, my experience in playing chess is that usually, that's not what happens. Usually it's not that I make a bad move and then there's a problem. It's usually I make a move that I think is reasonable that only later I discover put me in a position where I can't possibly do anything good. >> Right. Well that's actually the game that I played in New York many, many years ago that I lost was exactly like that. The game went on for like literally a hundred moves. I really lost the game on the third move. >> Oh. >> I made a mistake, I transposed two moves because it was a new opening that I was just learning



and I knew at the time that i screwed up, i played beautiful chess from that point on, but the truth is the other player had a positional advantage from that point on that I could never overcome, so I lost the game, but not because I played poorly for, you know, 80 moves. It's because I paid, played poorly for one move, and that move happened to be fairly early. So this is this notion of delayed reward, that I played this long game of chess, and maybe it's I played well, and I screwed up in the end. Maybe I played medio, you know, mediocre game, but I had a couple of brilliant moves, and that's why I won, or maybe I played very well in the beginning, poorly at the end or the other way around. And the truth is you don't really know. All you know is that you're taking a bunch of actions. You get rewards signals back from the environment. Like I won the game or I lost the game. And then [you have this problem of figuring out of all the actions that I took, what was the action that led to me ultimately winning or losing, or getting whatever reward that I got at the end of a sequence.](#) Does that make sense? >> Yeah it seems like [that could be really challenging.](#) You probably need your own like sportscaster listening and, and commenting. >> Yeah and in fact the sportscaster who is listening and commenting at least in the NDP world is in fact the sequence of rewards that you get in the sequence of states that you see right? It really is sort of a play by play. In this state this action got this reward and you want to take all of that and figure out whether this was a good action you're first action or a poor action or your second action was good or poor and so on and so forth. [Now contrast that with supervised learning, so in supervised learning what you would be getting is while I was in this state This first date and then this was the proper action I was supposed to take lets say action seventeen. And your goal then is just to simply learn a function from states to specific actions, that's how the supervised learning problem is setup, right?](#) >> Yeah exactly. >> [In this particular case you're in some state you take some action and you get some reward for the action you took. Or may be the state that you ended up in or something. And you get a sequence of these <state, action, award> triples, and ultimately you have to figure out for the given states you're in, what was the action you took that helped to deter, or actions you took that helped to determine the ultimate sequence of rewards that you saw. Perhaps this one plus one or this minus one that you got at the end. This is really difficult problem, its got its own name, its called the credit assignment problem. In fact, we're talking about a sequence of events over time, we typically refer to this type of problem as the temporal credit assignment problem.](#) Does that make sense Michael. >> Yeah that's really cool. Although I guess its credit but also blame right? >> Well credit and blame you know, blame is just the minus of credit. So without loss of generality Assume that credit can assume a negative or positive vibe. >> And yet when I go to the supermarket they never say blame or credit. >> Well that's just because they don't understand the technical terms. >> Got it.

## MDPs: More ABOUT REWARDS



- delayed reward
- minor changes matter

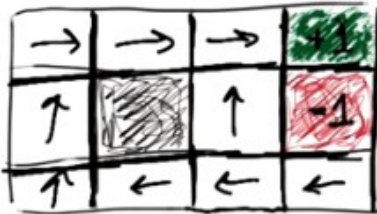
$$R(s) = -.04$$

12. We're really interested in solving this temporal credit assignment problem as we will see as we talk later on in the course. This sort of issue comes up again and again and again. But I didn't really want to talk about that in any great detail, I just wanted to make certain that, that people saw the difference between supervised learning and then sort of problems we're trying to solve now. That really has to do with sequences and time. What I want to just is focus a little bit more on rewards. And in particular, I want to look at the little grid world that we've been playing with so far, and think about how we would learn how to get from our start state. We always start it over here, over to one of the plus one or the minus 1, depending upon the kind of rewards that we see. So, let's push on this rewards notion for a minute, if we think about how MDP's are made up. And I'm just going to write down for you what the reward is, for the states in this particular grid world, okay? >> Yeah. >> So, let's say I'm going to set the rewards to be equal to -0.04. So, the first thing I want to do, is I want to make certain you understand what I mean by this. So, what I mean when I write this down, for this particular example. Is that the reward that I'm going to receive in every single state is -0.04, except of course for these two states which I've already labelled as +1 and -1. Okay? >> Alright, so, it seems like you should just never ever move then, right? >> no, I don't think that's true. Let's think about that for a little while. So, if I set this reward to be minus 0.04 and you're starting out here in this state, and you have to keep living, you always have to take an action, right? That's the way I've set this up. And you don't get to stop until you reach either plus 1 or negative 1, we'll call these terminating or absorbing states. Once you reach into one of these things that ends the game. What would having a small negative reward like, this encourage you to do? >> It's, it's making me think about kind of walking across hot beach, because like each time you take a step you're going to get a little bit of punishment and the only thing that ends it is if you can get into that nice cool ocean. And then you get a +1 for that and the -0.04s, stop. >> Right, so, that means, so wait how would you put that another way. Sounds to me like what you're saying is, by having a small negative reward everywhere, it encourages you to end the game. >> Yeah, that's [UNKNOWN]. Yeah, I agree with that. >> Okay, yeah and, and I think that's exactly what I like that analogy a lot. Your in a, well, it's not a very hot beach it's just a you know, a slightly unpleasantly warm beach and you really want to hurry up and get into the ocean. In fact, you want to hurry up and get into the ocean even though on your way to getting to the ocean, you might step on some glass, that the -1 is. So, let's go back to this notion of policy of mapping states to action and let's think about as being in this world where you have a reward of minus 0.04 every where except in the

absorbing states. What do you think the best set of actions are to take in these different states? What do you think the best policy is? Actually before you tell me, [let me write down what the actual policy is](#). I'm not going to tell you, how we get here but I'm about to write down the policy that's sort of the best one to take in a world where you have these sets of rewards. Okay. >> Sure. >> Okay, you see this Michael. >> Yeah, and it makes a lot of sense. Like if you're basically heading towards the, the rewarding state, the green state. >> Right, so [if I start out here in the leftmost bottom state, basically, it says, go up and then go to the right which, coincidentally is the policy that we had found before](#). >> Cool. [There's a couple spots that are a little bit strange, though](#). >> Like what? >> Well, I'm thinking about the one, not directly under the minus 1, that makes sense to me, but [the one \(最後一行右起第二個\)](#), it takes you now to a position. [Where it seems like you want to go straight up to the goal. But yet, it's going the long way around. It just didn't see the goal](#). >> No that's not it at all because of the way we are going to learn this, you're going to. So this is a global policy and I'm just telling you it's the optimal policy given the rewards. So let's kind of work out why going to the left makes sense here. Well, so [here's my argument](#). What this basically says is take the long way around. Yes? >> Yeah? >> But, by taking [the long way around](#), what's happening? Well, on the downside I'm going to pick up a little bit of negative reward, for a while, you know, one, two, three, four, five, six or so which is something like negative 0.2, negative 0.3, something like that, right? But, [by doing this, I avoid every getting into a state where it might fall into the -1](#). >> Because it's, it's stochastic. >> Right, it's exactly, so [stochastic means if I'm in this state here \(-1 左邊那格\) no matter what I do I have some, if I go up I have some chance of moving to the right and following into the -1. It's a relatively low chance, it's only 10%. But, it works out that moving from here to here, the probability of me falling into here \(-1\). Is too high compared to the sort of near impossibility of me ending up falling into the minus 1, if I can follow this path \(the long way around\)](#). >> Interesting, okay, I guess that makes sense, it's cool. >> Which actually suggests sort of the point that I want to make here which is, that [minor changes to your reward function actually matter](#). So, if we had a slightly different reward here, say. Not minus 0.04 but something else, you might find that some of these decisions would be different. I think you can see that? >> Hm. I mean at the level that I'm understanding it, it seems like if it's zero, that then it's in no particular hurry. If it's minus something big, then maybe it's in a bigger hurry. But, I don't, yeah [I don't see exactly what the difference is going to be](#). >> Okay, so [let's see if we can help you see the difference by making you take a quiz](#). >> Aha!

# MDPs: More ABOUT REWARDS

Quiz



- delayed reward
- minor changes matter

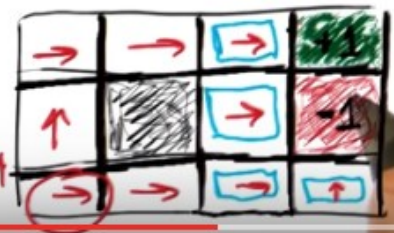
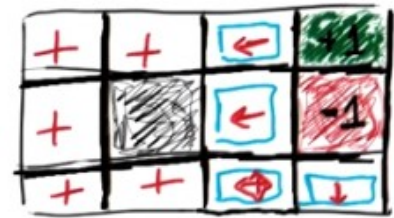
$$R(s) = -.04$$

$$R(s) = +2$$

U, D, R, L

$$R(s) = -2$$

$$R(s) < -1.6291$$



+2 右邊的那個 board 中, 最後一行左邊的藍框表示的是 上下左右 四個方向都可以走. 那些白色框中的+也表示 上下左右 四個方向都可以走. 弄成那樣的結果就是, 不管從哪裡出發, 永遠都不會掉進+1 和-1 框中, 可以永遠地賺 rewards.

13. >> Okay, Michael, so here's the quiz. Are you ready? >> Sure. >> Okay, so here's what's going on. I have copied over our little grid world over here twice. So it's the same grid world except I have changed the rewards that you will receive in all of the states other than the two absorbing states. So for the top one you get a reward of +2. In every single state, except for the absorbing states (那個+1 和-1 的兩格), and in the bottom one you receive a reward of -2 in every single state but the absorbing states, got it? >> Interesting. >> So, what I'm going to want you to do is, I'm going to want you to fill in for these boxes that I've indicated, these same four boxes in the top and the bottom, whether you should go up, down, left or right. So enter into each one of them either Up U, D down, R right, L left, okay? >> Yeah, I appreciate you making the numbers nice and big, because I was worried you're going to say something like minus 0.13. But this, this I think. >> I can do that. >> No, no, no, that wasn't a suggestion. Yeah, let's go with the plus two and minus two. I feel like there's a better chance I might be able to get it. >> [LAUGH] Okay, so you think you got the what we're trying to do here? >> I do. Now just to, just to keep, just to make sure that I'm there, the plus one and the minus one end the game, and there's no more reward after that, it's like zero forever. >> Yep. >> And the other things, the blueish rewards, I would get them. >> Mm-hm. >> Continually until a state is reached. >> Right, exactly. >> Where the game ends, okay Yeah. No, that's, that's, that's cool. >> Okay, cool. Alright so go.

14. Okay, Mikey, you got answers for me? >> sure. >> Alright. >> Wait, should I, do need to make any assumption, or can I make any assumptions I want about the non-blue states? >> No. It doesn't matter. [LAUGH] Actually. [LAUGH] >> So the answer is no. >> Maybe it does. >> The answer is no. But it doesn't actually matter because Remember, it's all very stationary and Markovian. So the only thing that matters is where you are. >> You're stationary Markovian. Alright, so I think, okay. So let's, let's do the first one, then. >> Okay. >> So, so this is, this is pretty cool. So by changing the reward the way that you did, it is now, it's not a hot beach anymore. It's like. You know like, super awesome

money beach. >> Yes, it's like a beach made of bacon. >> Sure. Okay. Let's think of it that way. Although neither of us is eating bacon at the moment. >> At the moment. >> But. [CROSSTALK] >> Great, because, because then our mouths would be full and it would be very difficult to give this lecture. Alright so, no, no, no, no, no I mean like, even. Alright, never mind. The point is that it's awesome, and [there's like diamonds](#). I don't know, I just saw The Hobbit movie, and [there is a room filled entirely with treasures](#). And so, [this is kind of like that. And so now the question is, what should I be doing, like I should never go to the +1 or the -1, because that ends my treasure gathering](#). I should just continue to stay in states that aren't those states. >> Okay. >> So I would say left for the, for the top state. >> Mm-hm. Let's say left for the bottom state. >> This one? >> The, yes, uh-huh. >> Okay. >> And the other two, I need to think about for a moment. So I feel like I'd like to get away from the minus 1. >> Mm-hm. >> So, like, you know, to go up. But then that would give me some chance of slipping, wouldn't it? >> Mm-hm. Lets see, I don't want to go to the right. I don't want to go down. I don't want to go. I'm going to go to the left. >> Mm-hm. >> I'm going to bash my head against the wall, because then I get money. >> Yeah. >> That's weird. There's really no pain for running into the wall? >> No. You just stay where you are. >> Alright and for the bottom one, I feel like it's the same kind of thing. I don't want to go to the right or to the left, cause then there's a probability that I'll slip and end up ending my game. So I'm going to go down. >> yeah. I like that. So most of these are right, and one of them is wrong. Or, it's not that it's wrong, it's just that's not as right as it could be. Which one do you think that is? >> I think they're right. But if, if there's one that I could imagine you not being so happy with, which is the bottom one, There are two bottom ones. You mean this one? >> Yeah. >> Yeah. What would I like? >> I think you might prefer if I bash my head against the wall. >> No, actually, it turns out that this is a correct answer so all four of these are correct. But in this particular state, it actually doesn't matter which way you go. Oh, I see. A more complete answer. >> Yes. >> Would be, doesn't matter. >> Yeah. >> And is that true of any of the other ones? No. I guess, I guess you're right. The other ones, you have to go in the direction indicated. >> Right. So, by the way, just like here, it doesn't matter which direction you go in. It's actually true for all of the other states as well. It doesn't matter where you go because In these three states, these are the only states where you can accidentally, you could end the game. And in each of them, you can take an action that guarantees that you don't end the game. So it really doesn't matter about where you do it in the other states. >> Cool. >> Right? So that's a good argument. This makes a lot of sense. So at the end of the day, basically, [because the reward is positive. And you're just accumulating reward or diamonds and treasure](#), as you put it. You just never want to leave, [so you always want to avoid either of these states. It doesn't matter that there's a +1 here and a -1 here. If I can stay here forever, then I just accumulate positive reward forever.](#)



# MDPs: More ABOUT REWARDS

Quiz



- delayed reward
- minor changes matter

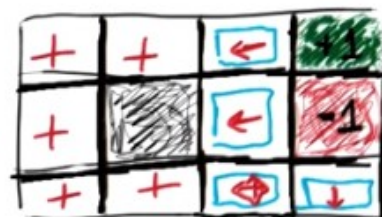
$$R(s) = -.04$$

$$R(s) = +2$$

U, D, R, L

$$R(s) = -2$$

$$R(s) < -1.6291$$



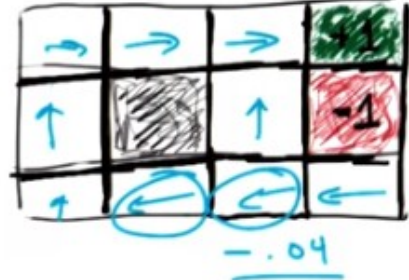
(此圖是上圖拷過來的, 為了方便看)

Okay, so what's the next one? >> So, interesting. So now, now the beach is really, really hot. It's -2, so.  
 >> Mm-hm. I want to get off here as quickly as I can. So definitely the top one, I would go to the right.  
 Just try to get, get that +1 and get outta here. >> Yeah. >> Alright, so then, let's think about the bottom  
 right (即最後一行 左邊那個藍框). >> Okay. >> So, in the best of all possible worlds I go around and  
 dump into the +1. And that would take me again if, if I don't slip at all one, two, three, four steps to do  
 that >> Mm-hm. >> Actually wait one, two, three, three steps one the hot sand. >> Yup >> And I'm  
 getting and -2 for each of those. So that's like -6. >> Mm-hm. >> So I could get that plus 1 at the end,  
 and that makes it only minus 5. But I think I'd rather just get off the beach. So I'm going to say in the  
 bottom right to actually dive into the pit of pain, because it can't be as bad as where I am. >> That's  
 exactly right. >> So, okay, so then the, the one next to it on the left (-1 左邊那個) is a little trickier. If I  
 go up to the plus 1, I get a minus 2 Followed by a +1 which is also -1. >> Mm-hm. >> So okay. I'm  
 not so sure about that one. >> Well remember you'll always have some probablity of ending up back  
 where you started >> Is that true? Oh because, if I. Well no, no, that's not true if I dive straight into the  
 -1. Then I have 0.8 chance of going to the -1. 0.1 of going up. To the arrow that's labeled, 0.1 going  
 down to the blue box that's not labeled >> Mm-hm. >> Wait, 0.2? No, 0.1 >> Mm-hm. >> Whereas if  
 I go up, right, I might stay where I am. And accrue -2, like, yeah I feel like I need a calculator for this.  
 >> Yeah, but you're, I think your intuition is right, though. Just think about it as How much, I mean  
 what do you think has the best chance of most quickly ending things? >> Well certainly jumping into  
 the -1. >> Right. >> I think that's going to be marginally better because there's only one chance of  
 slippage and delay. Where as if I take additional step, there's two chances of slippage and delay. >>  
 Exactly, that's exactly the right kind of argument to make and, and because the -2 is so big. You're  
 you're going to, you're just better off ending it even if you ended in pain, in the same way that you were  
 in the bottom right hand corner. >> And so bottom left I would say the cleanest thing would be to jump  
 into the -1. So to go to the right so that I can go up and get into the -1. >> And what's nice about that  
 move is that is that you either end up in the bottom right hand corner. Or you end up staying where you  
 are, or you end up moving up. But in either case, you are always sort of, you know, you never get  
 farther away, right? Whereas if you try to go up, there's a chance that you will get farther away. >>

Interesting. Is there a way to be sure about these? >> Yeah, you just have to do the math. >> [LAUGH] Do we know how to do the math? >> Yeah, you just start, you just do expectations. You can basically just say, well what's the expected time. Since you know what you're trying to do here is to get to the end. You can just, you can calculate the expected length of time it'll take you to get there. And then just, you know, multiply by all the rewards you're going to get in the meantime. If it helps, though, I will tell you that for any value where the reward is less than  $-1.6284$ , this is the right thing to do. >> [LAUGH] Okay. >> Just so you know. >> Alright, so you're saying you actually do know that this is the optimum. The thing that we have here. >> Yes, I know that this is the optimum. >> Alright. Let me just draw out the rest of it for you. So it's, it's I think interesting to compare this one. The bottom right one (即-2 那個 board), where you have given me negative rewards to encourage me to end the game. With our main one up here in the upper left (即-.04 那個 board). Where you have also given me some negative reward to encourage me to end the game. If you look carefully you'll notice that in this one, you're encouraged to end the game. But you're really encouraged to end the game at a plus 1. Here the reward is so strongly negative, you just need to end the game. And so you end up with a different response, here (-2 那個 board 中左下角劃圈那格). Because this is, is just as quick to get to the -1 and into pain as to, take this extra step and try to get to the plus 1. And you end up with a different response here. Here, here, and here. >> That's really interesting. >> Yeah, it is. And so, these changes matter. They matter a great deal. >> So, how am I suppose to choose my rewards? >> Carefully and with some forethought. >> Nice. >> But, you know, again There's lots of ways to think about this. So the way we've been sort of talking about MDPs is kind of the way we've been talking about we talked in the first part of the course. When we talked about having teachers and learners. You can think of your reward as sort of the teaching signal. Sort of the teacher telling you what you ought to do, and what you ought not to do. But another way of thinking about this is that because the rewards define the MDP The rewards are our domain knowledge. >> I see. >> So if you're going to design an MDP to capture some world. Then you want to think carefully about how you set the rewards in order to get the behavior that you wish. >> That's fair. I mean, it seems a little bit like a cop out, but I think it seems like a necessary one. >> Yeah. I mean, there's really, I mean, again. No matter what you do, you've gotta be able to inject domain knowledge somehow. Otherwise, there's no learning to do. And in this case, the reward basically is telling you how important it is to get to the end. >> Cool. >> Okay. Alright. Let's move on.

# SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS  $\Rightarrow$  stationary
- UTILITY OF SEQUENCES



$$ib \quad U(s_0 s_1 s_2 \dots) > \\ U(s_0 s'_1 s'_2 \dots)$$

$$then \quad U(s_1 s_2 \dots) > \\ U(s'_1 s'_2 \dots)$$

$$\pi(s) \rightarrow a \\ \pi(s, t) \rightarrow a$$

stationary  
preferences

最後一段原文: how do utilities differ from rewards? The utilities factor in the long term aspects, and the rewards are just telling you the moment to moment.

15. Alright. So having gone through that exercise, Michael, I think it's, it's worthwhile to step back a little bit and think about the assumptions that we've been making that have been mostly unspoken. And I'm going to say that the main assumption that we've been making in some sense boils down to a single word. And that word is stationary. So let me tell you what I mean by that and why by kind of illustrating what it is we've been sort of doing for a little while. Okay? >> Sure. >> Okay. So the first thing I'm going to say is that we've actually been kind of assuming infinite horizons. So what do I mean by that. When, when we think about the last grid world that we were playing with, we basically said well, you know, I want to avoid going to the end as quickly as possible if I have rewards of a certain value or whatever. Because, you know, the game doesn't end until I get to an absorbing state. Well, that sort of implies. That you basically can live forever. That you have an infinite time horizon to work with. Now can you, can you imagine why if you didn't have an infinite time horizon to work with you might end up doing something very different? >> Different then what, what we're doing in the grid world? >> Right, so here let me let me show you the game that we were, the grid world that we were doing before might help you think about it. So here's the grid world we had before. And as you recall. We had a particular policy that sort of made sense. Here, I'll, I'll write it out for you again. And this was with a case where we had a reward of minus 0.04. Remember? We just did this. Remember? >> Yep. >> Okay, and this was the policy that turned out to be optimal, and in the future I want you to pay attention to here is that when you're over right here (-.04 上面那格, 即最後一行右起第二格) near possible end state, rather than going up, it made sense to take the long way around. Because you're going to get some negative reward but it's a small enough negative reward compared to where you might end up, Okay with a positive one. >> Yeah, I see. >> And that makes some sense. Well, that only makes sense if you're going to be living long enough that you can take the long route around. What if I told you you only had say three times steps left, and then the game is going to end no matter where you end up? >>

Well, it might be, it might make more sense to take some risk and just try to take the short way because otherwise there's really no chance you're going to get to the +1. I'm entirely convinced of that though because there's still a chance you'll fall into the -1 along the way. >> Right, so the exact val, whether it makes sense to take the risk or not is going to depend upon two things, we've already talked about one of them which is the actual reward that you get. If this reward were, you know, negative enough, then clearly it makes sense to just try to end things quickly, right? We just showed that in the last quiz. But another thing that it's going to depend upon is how much time you have in order to get to where you're going. If you've only got one or two time steps before everything's going to end. You can imagine that there are cases where, without changing the reward too much it makes a lot of sense to try to go ahead and quickly get to this plus 1, even though you have some chance of falling into the minus 1. As opposed to, trying to move away, where you're then kind of, all but guaranteed that you're never going to reach the +1. So. Whether it makes sense to take the risk or not will depend upon the reward but it's also going to depend upon whether you have an infinite amount of time to get to where you want to get to or whether you have a finite amount of time. And the real major thing I want you to get out of that is that if you don't have an infinite horizon but you have a finite horizon then two things happen. One is the policy might change because things might end. But secondly, and more importantly, the policy can change, or will change, even though you're in the same state. So, if I told you, if you're in this state right here, and I told you you didn't have an infinite amount of time, but you still had 100 million time steps then, it, I think it's clear that it still makes sense to go the long way around, right? Yeah, I mean the, the probability that this policy is going to last for a million timesteps has got to be tiny. >> Right. So, I might as well. It's 100 million timesteps might as well be infinity. But if I make that number not 100 million but I make it 2, or 3, or 4. Then suddenly your calculus might change. In particular, your calculus will change even though I'm in the same state. Right? So maybe this state right here, if I've got a million, 100 million timesteps I still want to go the long way around, but if I've only got a few time steps, the only way I'm ever going to get a positive reward is to go this way (近的那條路). Does that make sense? >> I guess so. So, you're saying, for example, even within the single run, it could be that I'm in a state and I try an action and maybe it doesn't work and I stay where I am. And I try it again, and maybe it doesn't work and I stay where I am. It might then switch to a different action, not because the other one wasn't working, but because now it's running out of time. Right, exactly. So we talked about this notion of a policy which maps states to actions, we talked about this notion about stationarity. So you believe that this sort of Markovian thing said, it doesn't matter where I've been it only matters where I am. And so if i'm in this state, since it only matters where I am, I'm always going to want to take the same action. Well that's only true in this kind of infinite horizon case. If you're in a finite horizon case. And that finite horizon, of course, is going to keep counting down, every time you take a step. Well then suddenly, depending upon the time step that's left, you might take a different action. So we could write that I think, just for the sake of kind of seeing it as some thing like, your policy is a function both the state and, and the time step you're in. >> Hm. And that might lead you to a different set of actions. So this is important, this is important, I mean were not, we are not, for, for this course going to talk about this case at all, where you're in a finite horizon, but I think it's important for you to understand that the, without this infinite horizon assumption here. You lose this function of stationarity in your policies. Okay? >> Yeah. Interesting. >> Okay. So, that all, I think is, you know, making our something that's obvious, but becomes obvious after someone points it out to you. So, the second thing that I want to talk about, I think, is a little bit more subtle. And, and this notion of utility of sequences. So, as we've been talking, Michael, we have been sort of. Implicitly discussing not just the rewards we get in a single state, but the rewards that we get through a sequences of states that we take. And so I just want to point out a little fact that that comes from that, and where that ends up leading us. And then we'll get to some nice little cute series of math. So. Here's what I want to point out what utilities, what we mean by utilities sequences. It means we have some function I'm going to call U for utility (後面會定義, 後面也將 utility 稱為 value function, 故有 value iteration 算法) over the sequence



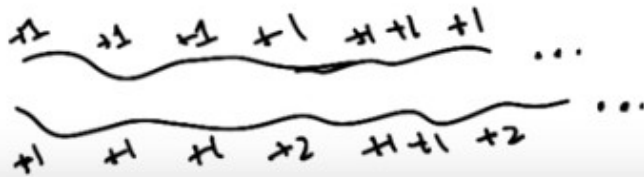
of states that we're going to see. Let's call them,  $S_0, S_1, S_2$  and so on and so forth. Well, I think an assumption that we've been making even if we haven't been very explicit about it is that if we had two sequences of states.  $S_0, S_1, S_2, \dots$ . And a different sequence  $S_0$ , then  $S_1'$  and  $S_2', \dots$  that is two sequences that might differ from  $S_1$  on, but all start in the same start state. Okay? If we have a utility for the first, and that utility happens to be greater than the utility for the second, then it also turns out that we believe. That the utility for  $S_1, S_2, \dots$ , is greater than the utility for  $S_1', S_2', \dots$ . >> Alright so these are two different sequences,  $S$  one, the  $S$ 's and the  $S$  prime's are two different sequences. >> Yes. >> And in the beginning we're comparing them with  $S_0$  stuck in front of both of them. And we're saying if I prefer the  $S_0$  followed by all the  $S$ 's, to  $S_0$  followed by the  $S$  primes, then I have that same preference even with those  $S_0$ s missing. >> Right, and so this is called stationarity of preferences. And, another way of saying it is. That if I prefer one sequence of states today over another sequence of states, then I prefer that sequence of states over the same sequence of states tomorrow. >> So isn't, isn't this just obvious? Because the whatever the rewards for those two cases, we're just adding the reward we get for  $S_0$ . So. It's going to be the same. >> But listen to what you just said. You just said, well, it'll be the same, because all we're doing is adding the reward for  $S_0$ . But what did we ever say about adding up rewards? >> I thought, I thought that's what we were doing. >> That's right, that is what we were doing. But we never actually sat down and wrote that down and said, this is what it means. To talk about the utility of a sequence of states as opposed to the reward that you get in one state. >> Okay, so you're saying that if we, if we are adding rewards, then this follows. >> Right. >> Okay. >> And then I've actually been saying something even stronger, which is, I will show you on the next slide, which is if you believe that this is true, that the utility of one sequence of states is greater than the utility of another sequence of states. Both today and tomorrow. Then it actually forces you to do some variation of what you said which is just adding sequences of states. Or adding the rewards of the sequence of states that we see. >> That's really interesting. So then, so the adding isn't really an arbitrary thing it follows from this, this deeper assumption. >> Right, and the reason I bring this up is because. It would make sense if you were to just to grab someone off the street and start talking about Markov Decision Processes. One of two things will happen. Either they'd run screaming from you like you're a crazy person or they would sit and they would listen and if they listen they would just completely buy into the idea that you just add up sequences of rewards. You know, sequences of rewards that you see as a way of talking about how good the states are because that's a very natural thing to do. But it turns out that mathematically if you have this notion. A sort of stationarity of preferences and this sort of infinite arise in world. You really are in a case where this has to be true. And it has to be the case if you have to do some form of addition. Because nothing else sort of can be guaranteed to maintain this property over stationary preferences. I mean, as you said, if I got one sequence of states and another sequence of states and by just prepending or appending another set of states to it, I'm still going to always guarantee that one's greater than the other. You kind of have to do some form of adding the reward that you see in the states in both cases. because if you don't do that, then eventually this inequality will not hold. So, let me write that down in math terms. And see where that gets us, okay? >> Cool.



## SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS
- UTILITY OF SEQUENCES

$$U(s_0, s_1, s_2, \dots) = \sum_{t=0}^{\infty} R(s_t)$$

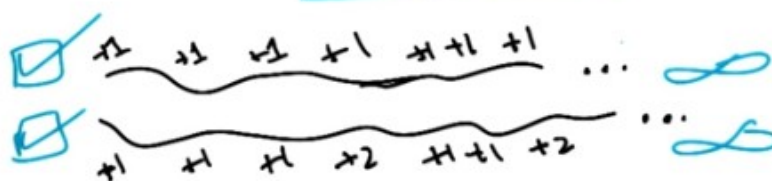


16. >> Alright, Michael. So, let's write that down in math, okay? You like math, right? >> I do. >> Okay. So here's the math version of it. I'm going to just say that the utility that we receive for visiting a sequence of states,  $s_0, s_1, s_2$  ellipsis, is simply the sum of all the rewards that we will receive for visiting those states. >> Sure. >> Does that make sense? >> Yeah. >> Is that consistent with what you were doing when you were thinking about the grid world? >> Yeah, exactly. But I thought we were going to make that not definitional, we were going to derive that it had to be that way. >> No, we're not. They can read about it. It'd take me, like, an hour and a half to do it. >> Alright. >> And I'm already losing my voice. But. What I do want people, I want you to believe is that the utility of the sequence of states, thinking of it as the sum of all the rewards, makes sense, right? And then if nothing else, at least it makes, it's consistent with what you've been doing as we've been talking about this grid work. >> Yeah, absolutely. I mean it, it also kind of makes me think about money. >> Go on. >> Well, so, so, I mean, that's sort of how money works. If we get some kind of payoff each day, those payoffs get added to each other. They don't get, you know, subtracted or multiplied or square rooted or whatever. They just, you know, they go into your bank account, and things add. So this feels like, kind of like that. It's like money in my pocket. >> Sure, if you have a big enough pocket. Okay, I'm with you on that. Alright, so I'm going to say that this all makes sense. It, it's really awesome and it sort of doesn't work. And to illustrate why this doesn't work, I'm going to give you a quiz. >> Yay. >> because you like quizzes. >> So I've been told.

## SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS
- UTILITY OF SEQUENCES

$$U(s_0, s_1, s_2, \dots) = \sum_{t=0}^{\infty} R(s_t)$$



Quiz

WHICH IS BETTER?

IMMORTAL

17. >> So, here's the quiz. You see on the screen [this sort of unexplained two little squiggles with a bunch of numbers](#) in front, on top of them or under them or near them? >> Yeah I, I assume that's a river and on one bank it's the land of the plus ones and the other bank has been infiltrated by plus twos. >> That's not what I intended at all but I actually like that enough that I'm going to pretend, that that's what I intended. So [these work out to be sequences of states](#). >> Oh, I see now. >> Okay, and [rewards that you receive](#). No, no, no, it's a riverbank and on one side of the bank, [as you walk along it, you get a bunch of plus ones. On the other side you get a bunch of plus ones and some plus twos. And this just goes on forever](#), okay? And, I'm not going to tell you what all the rewards are after that except that they, they look similar to the rewards that you see here. >> Okay. >> Plus ones and plus twos, okay? And let's say the top, [the top one, in fact, nothing but plus ones. And the bottom ones are some plus ones with maybe some plus twos sprinkled in and out](#), but those are really the only options. Okay? >> Yeah. >> Here's my question to you. So, [would you rather be on the top side of the river bank, or the bottom side of the river bank?](#) Okay? You think you know the answer already, don't you. >> Yeah. >> Okay, cool. Well then, let's, let's give our listeners a, a chance to come up with the right answer.

18. Okay Michael, I didn't give you as much time as I normally do because you think you already know the answer. So what's the answer? >> So, you know, I'd rather get the plus 2's occasionally, so I'll say the bottom one. >> No. >> Wait, what? You're not going to tell me the top one's better? >> no, it's not. >> Okay then that feels like a trick question. >> It's not a trick question. The answer is, [neither one is better than the other](#). >> Oh, so I had to give neither? >> Or both. >> Which is better, and I can click on neither. >> Or you could click on both. Okay. So you thought that the bottom one was better, what was your reasoning for that? >> Because, sort of moment to moment, sometimes I'm getting plus one's, and that matches what I would've gotten if I had been on the other bank. But then sometimes I get plus two's, which is actually better than what I would have gotten on the other bank. And so, I'm, I'm, I never feel any regret being on the bottom bank. I only feel regret being on the top bank. >>

That's fine. So, what would you say the utility of the sequence along the bottom actually is? >> 1 plus 1 plus 2 plus 1 plus 1 plus 2 plus dot, dot, dot. >> Which is equal to? >> infinity, I guess. >> That's right. What about the utility of the top one? >> 1 plus 1 plus 1 plus, that's also infinity. >> Yep. So the utility of both of those sequences is equal to infinity. Do you think one of them is bigger than the other? >> You drew the bottom one a larg, a little bit larger. >> I did. Or longer, anyway. >> Yeah. >> But, in the end, the sum of the rewards that you get are both going to be infinity. So the truth is, neither one of them is better than the other. >> I still don't think you can say that both are better. >> You can't get around that. >> But yeah, I see, I see, neither, I can see neither is better. >> Or that both are better. Neither is better, both is better, whatever. The point is that, they're both equal to infinity. >> Hm. >> And the reason they're equal to infinity is because all we're doing is accumulating rewards. And, if we're always going to be able to get positive rewards no matter what we do, then it doesn't matter what we do. This is the existential dilemma of being immortal. >> Oh, living forever. >> Right. >> So if you live forever then, like why should you care about anything ever? >> Right I mean, everyone, every all the mortal people are going to die and one day they'll all be, you know, an infinite amount of time in your past. I could do this thing here, which is pleasurable, or I could do this thing right now, that, you know, will, is less pleasurable, but will eventually get me to a better place. But if I'm going to live forever, and I can always get to a better place, than it really doesn't matter what I do. It really doesn't matter. >> Mm. >> Because I'm just accumulating rewards, I'm living forever, and I'm going to, infinity is infinity and there's no really no way to compare them. Having said that, your original notion that, look, it feels like I should never regret having taken the second path compared to the first because I will occasionally do better. Seems like the right intuition to have. >> I see but it's just not built into this particular utility scheme. >> Right, but it turns out there's a very easy way we can build it into this utility scheme by just making one tiny little change. Would you like to see it? >> Yes. >> Beautiful, let's see it then

## SEQUENCES OF REWARDS: ASSUMPTIONS

- INFINITE HORIZONS
- UTILITY OF SEQUENCES

$$\begin{aligned}
 U(s_0, s_1, s_2, \dots) &= \sum_{t=0}^{\infty} R(s_t) \quad \leftarrow = \infty \\
 &= \sum_{t=0}^{\infty} \gamma^t R(s_t) \quad \underline{0 \leq \gamma < 1} \\
 \text{discounted} &\Rightarrow \text{geometric} \\
 \text{infinite} &\Rightarrow \text{finite} \leq \sum_{t=0}^{\infty} \gamma^t R_{\max} = \frac{R_{\max}}{1-\gamma}
 \end{aligned}$$

後面給  $\gamma$  叫的 discount

19. Okay Michael so here's the little trick. So all I've done is I have replaces the equation at the top with

an alternate version of the equation. >> 'Kay. So it looks there's, you're now exponentially blowing up the reward. >> I am not exponentially blowing up the reward. Instead of what I've done. Is I've added of the see, the rewards that I'm going to see. For the states that I'm going to see. And I multiplied by gamma to the T. The gamma is between 0 and 1. >> I see. So... >> Do you? >> Well, kind of. So, so it doesn't exponentially blow up. It exponentially implodes. >> Right. >> So, so things that are like a thousand steps in the future If we multiply, if we take something that's less than 1 and we raise it to that power, it goes essentially to 0. So it's like it starts off the rewards are kind of substantial and then they get, they trail off quickly. >> Right. And in fact we can write down mathematically what this is If you actually, if you stare at it long enough and you remembered your intermediate algebra or your calculus or wherever it is you learned this stuff. You would probably recognize this as a special kind of sequence or series. Do you remember what it is? >> Television series? >> [LAUGH]. >> No, but that's remarkably close. So let's see if we can bound this particular equation. So we know that this version of the equation eventually ends up being infinity if all the rewards are positive. Right? >> Mm, hm-mm. >> So what does this end up being even in the case where all the rewards are positive? Well, we can bound this from above by the largest reward that we will ever see, in the following way. So all I've then is said well I don't know what the rewards are but [there is some maximum reward I'm going to call it Rmax](#). And if I pretended that I always got the Rmax reward. So long as that, as long as that's an actual number. A finite number. I know that this is bounded from above by this expression. Well what does this look like. Maybe this does look like a series you remember. >> Geometric series? >> Yes. This is the geometric series. And this is exactly equal to this. . >> And I assume that the summation causes the max to become lower case. >> Yes, yes it does. That's a, that's a trick that they don't really go over deeply in calculus, but it's true. >> Oh, that's cool. Alright so, the reward, the maximum reward divided by 1 minus gamma. So when gamma is really close to 0, you're just getting, oh you're getting just that one reward in the beginning and then everything falls off to nothing after that. >> Yep. >> And if gamma is really close to 1. You're dividing by something that's teeny tiny, which actually is like multiplying it by something that's really big. So it kind of magnifies the reward out, but it's not infinity until gamma get's all the way up to 1 and that's the same case we were in before. >> Right, so in fact you'll notice the way I wrote this, gamma has to be between 0 and 1, 0 inclusive but strictly less than 1. If I made it so that it could include 1, well that's actually the first case. >> Got it. >> [Because 1 to a power is always 1, and so, this is actually a generalization of the sort of, infinite sum of rewards. This is called discounted rewards, or discounted series, or discounted sums, and it allows us, it turns out, to go an infinite distance in finite time](#). Wait. >> That makes sense. >> No. No. [LAUGH] An infinite distance? >> Yeah. That's at least the way I like to think about it. So, if we're discounted in this particular way, then that gives us a geometric series. And what it does is [it allows us to add an infinite number of numbers, but it actually gives us a finite number](#). Cool. >> So this means we can still have our, and it turns out, by the way, just to be clear, that this world that we're in. Where we are in between 0 and 1, 0 inclusive is still consistent with our infinite horizons and our stationarity over preferences. So we can add things together and be consistent with those assumptions we were making before Or we can do discounted rewards and we're still going to be consistent with what we're doing before. But the difference between this case and this case is that by doing the discount, we get to add infinite sequences and still get something finite. The intuition here is that Since, if gamma's less than 1, then eventually as you raise it to a power, it will basically become 0. I mean, that's why it's a geometric series. Which is like having a finite horizon. But that horizon is always the same finite distance away, no matter what, where you are in time. >> Which means it's effectively infinite. >> Or unbounded, or however you want to think about it. So it's like you take a step, but you're no closer than where, when you started. >> Right, so, what does that mean in, a world where you meant to say you're no closer than where you were trying to end up? So that means that even though, you've taken a step forward, the horizon sort of remains a fixed distance away from where you are. >> Mmm, mmhmm. >> That's what it means to To, to have this kind of this gamma value here and so you can still treat it as if it's always going to be

infinite, even though it gives you a finite value, and that gives you your stationary. So does that make sense? I mean that's sort of the intuition. >> Yeah. >> You really are have an in, you're always in infinite, It's always in, infinity. But the gamma value means that at any given point in time, you only really have to think about what amounts to a finite distance. You never get closer to the horizon, even though you're always taking steps towards it. But there still is a horizon that you can see, and I can kill that analogy. >> [LAUGH] So can we, can you explain to me why it has this form, this  $R \max$  over  $1 - \gamma$ ? >> I could. There's a You can kind of prove that, in a little cute math way, and I'm happy to take that diversion if you want. >> Yeah, I think so. I mean, unless, unless you were going to tell me something else. >> Well how about I tell you something else and then I take that diversion? >> Okay. >> Okay, so here's the something else. In the beginning I said, well, this is like going in infinite amount of distance in finite time, which you rebelled against. And my explanation is more about. How infinity looks finite which is not the same thing as going an infinite amount of distance in finite time. The reason I said that is because of the singularity. >> What? >> The singularity so, some of our listeners no doubt have heard of the singularity. You never heard of the singularity? >> The singularity is like when computers get so fast that you do infinite computation and, oh. Right. So [the singularity, for those of you who don't know is is this sort of observation that has been made by many folks](#). That the sort of limit to computer power growing faster is the fact that it takes us amount, some amount of time to design the next generation of computers. Right, So [Moore's law says everything doubles you know, everything 6, 18 months or whatever](#). But if we could design things faster, then we could actually make it double more quickly. So, one day we'll get to the point where the computer, which I will draw here, like that. The computer [can actually design the next generation of computer](#). Well, when it designs the next generation of computer, that computer will also have the ability to design the next generation of computer. But it will be able to design it twice as fast. And then the next one will be able to design its next successor twice as fast, and so on and so forth. So this little time between generations, will keep Having every single time, which looks remarkably like a geometric sequence. And so once we reach this point, where the computer can actually design its successor, we will then be able to do an infinite number of successors. In finite time. >> [LAUGH] >> [And that's when the world comes to and end. And that's what's called the singularity. Because we can't understand what happens after that point. So that's what I mean by going an infinite amount of distance in finite time.](#) >> It just doesn't seem like distance. But, yeah, that's a weird example, for sure. >> I think it makes sense. >> [INAUDIBLE] infinite number of doublings in it. Yeah, [INAUDIBLE], no. [LAUGH] Well, we'll, we'll, we'll do we'll do some assignment where we allow people to decide whether this makes sense or one of your analogies make sense. And I'll just remind the listener that I'm the one who'll be assigning final grades. >> Okay. >> So with that let's go and do your little diversion. And and then we'll come back to. Doing a whole lot of math. So I actually think this little nice diversion will be warmup to all the math we're going to be doing. Okay? >> Oooh. Hm. More math. >> Hm.



## SEQUENCES OF REWARDS: ASSUMPTIONS

$$\left(\sum \gamma^t\right) R_{\max}$$

$$x = \left( \gamma^0 + \gamma^1 + \gamma^2 + \dots \right)$$
$$\gamma^0 + \gamma^1 + \gamma^2 + \dots$$

$$x = \gamma^0 + \gamma x$$

$$x - \gamma x = \gamma^0$$

$$x(1 - \gamma) = 1$$

GEOMETRY  $\Rightarrow$   
EASY



2:50 / 2:52

$$x = 1 / (1 - \gamma) \cdot R_{\max}$$



YouTube



(本圖的目的就是證明  $\sum \gamma = 1 / 1 - \gamma$ , 沒甚麼別的. 圖中說 Geometry is easy, 是因為這是 geometric sequence)

20. >> So, if we think about the equation that we were doing before, which was you know, the sum of all these gammas times some  $R_{\max}$ . We can basically take out the  $R_{\max}$  and what we end up with is something that looks like that, right? You're summing together a bunch of gammas and then multiplying the result by  $R_{\max}$ . So what does that actually look like? Well that looks like this. Gamma to the zero, plus gamma to the one, plus gamma squared, plus dot dot dot. Eventually multiplied by  $R_{\max}$ . Right? So let's call this  $x$ , okay? >> Does  $x$  include the  $R_{\max}$  or not? >> No. So we'll just call the little sequence of, it's going to turn out not to matter. Let's just call the sequence of gammas we're adding together, let's just call that  $x$ . >> Good. >> Well, if you look at it, this is actually recursive. Right? Because this is an infinite sequence, if you sort of shifted it one over in time, you would end up with just the repeated sequence again. All right? Which means that we can write  $x$  in terms of itself.  $x$  equals gamma zero plus gamma times  $x$ . >> I see, so it's shifted over, but then you have to multiply it again by gamma, to get up to gamma one, gamma two. >> Right, exactly. So, so, this is just, you know, it's just math, that's just all it is. so, we can try to solve for  $x$  and figure out what  $x$  is, right? >> Cool. >> So, what is  $x$ ? Well, we can subtract from both sides and so we end up with like, something like  $x$  minus gamma  $x$  equals gamma zero. Right? >> Seems like we can stop writing gamma zero. Isn't that just one? >> Yes, but I've already, I've gone too far, Michael. I've already, I've already written gamma zero. >> Alright. [LAUGH] >> so, this becomes  $x$  times 1 minus gamma equals gamma zero, or as Michael so astutely points out, 1. [LAUGH]. Alright. Which means what? It means that. >> So then we divide by 1 minus gamma. >>  $x$  is 1 over 1 minus gamma. >> Neat. >> And that, of course, we are going to multiply by  $R_{\max}$ . >> To get the formula that we had. That's, yeah, that's nifty. >> Yeah, there we go. So, why do we do this? Well, because you like stuff like this and it points out something very simple, which is that geometry is easy. >> [LAUGH] I don't think that's the kind of geometry that people usually think of. >> Well, they should be thinking of this kind of geometry. So

geometry is easy. And by doing a little cute algebra, we can derive ridiculous equations that turn out to help us deal with go an infinite distance in finite time. >> [LAUGH] I don't think that's what they're doing, but okay. >> [LAUGH] So, let's think of this as warmup for what I actually wanted to show you, which is going to turn out to be a whole lot of math. Okay? >> Alright, let's, let's go for it. >> Alright, let's do that.

POLICIES

$$\pi^* = \operatorname{argmax}_{\pi} E \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi \right]$$

$$R(s) \neq U(s) = E \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, s_0 = s \right]$$

immediate
long term
delayed reward

21. Okay, so Michael, in the spirit of what we just went through in deriving the geometric series, I'm now going to write down a bunch of math. And what I'm going to do is I'm just going to say it at you, and you're going to interrupt me if it doesn't make sense. Okay? >> That makes sense. >> It does. Okay, so here's the first thing I'm going to write down. I've decided that by going through this exercise of Of utilities in this kind of reward, we can now write down what the optimal policy is. The optimal policy, which as you recall is simply  $\pi^*$ , is simply the one that maximizes our long-term expected reward. Which looks like what? Well, it looks like this. There, does that make sense? >> Let me think, so. We have an expected value of the sum, of the discounted rewards, at time t. And, given  $\pi$ , meaning that we're going to be following  $\pi$ ? >> Mm-hm. So these are the, the sequence of states we're going to see in a world where we follow  $\pi$ . And it's an expectation because, things are non-deterministic. Or may be non-deterministic. And do we know which state we started? >> It doesn't matter, it's whatever  $s_0$  is. >> I see. Whatever  $s_0$  is, but isn't that random? I mean  $s_1$  and  $s_2$ ,  $s_3$ ; those are all random. >> Well, we start at some state, it doesn't matter, so  $t$  is starting out a zero. And going to infinity. Okay? So does this make sense? >> Yes, so then, so we're saying, I would like to know the policy that maximizes the value of that expression. So it gives us the highest expected reward. Yeah, that's the kind of policy I would want. >> Exactly. So, good, we're done, we know what the optimum policy is. Except that it's not really clear what to do with this. All we've really done is written down what we knew it was we were trying to solve. But it turns out that we've defined utility in such a way that it's going to help us to solve this. So let me write that down as well. I'm going to say that the utility of a particular seque, of a particular state okay. Well it's going to depend upon the policy that were following. So I'm going to rewrite the utility (U) that takes the superscript  $\pi$ . And thats simply going to be the expected set of states that I'm going to see from that point on given that I've followed the policy. There, does that make sense? >> It feels like the same thing. I guess the difference now is that you're saying the utility of the policy out of state is what happens if we start running from that state. >> Yep. And we follow that policy. >> Got it. >> Right. So, this answers the question you asked me before about, well, what's  $s_0$ ? Well, we talk about that in terms of the utility of the state. So how good is it to be in some state? Well, it's exactly as good to be in that state as what we will expect to see from that point on. Given that we're a following a specific policy where we started in that state. >> Hm,. >> Does that make sense? >> Kay. Yeah. >> Very important point here, Michael, is that the reward for

entering a state is not the same thing as the utility for that state. Right? And in particular. What reward gives us is immediate gratification or immediate feedback. Okay? But utility gives us long term feedback. Does that make sense? So when reward for a state is the actual value that we get for being in that state. The utility for a state is both the reward we get for that state. But also, all the reward that we're going to get from that point on. >> I see. So yeah. That seems like a really important difference. Like, if I say, here's a dollar. You know? Would you poke the president of your university in the eye? You'd be, like, okay. The immediate reward for that is one. But the long term utility of that could be actually quite low. >> Right. On the other hand, I say, well, why don't you go to college? And you say, but that's going to cost me \$40,000. Or better yet, why don't you get a masters degree in computer science from Georgia tech, but you can say that's going to cost me \$6600. Yes, but at the end of it you will have a degree. And by the way it turns out the average starting salary for people who are getting a masters degree or undergraduate degree is about \$45000. >> So is it considered product placement if you. Plug your own product within the product itself? >> No, I'm just simply stating fact Michael. This is all I'm doing. Just facts. >> Alright. >> This is called fact placement. >> Alright. >> The point is, there's a, an immediate negative reward, of say, \$6,600 for, I'm going through a degree. Or maybe it's \$10,000 by the time, the 15th person sees this. But anyway, it's some cost. But, presumably it's okay to go to college, or go to grad school, or whatever. Because at the end of it you are going to get something positive out of it. So it is not just that it prevents you from taking short term positive things if that is going to lead to long term negative things. I also always you to take short term negatives if it will lead to long term positives. That makes sense. What this does is this gets us back to what I mentioned earlier, which is this notion of delayed reward. So we have this notion of reward, but utilities are really about accounting for all delayed rewards. And if you think about that, I think you can begin to see how, given you have a mathematical expression delayed rewards, you will be able to start dealing with the credit assignment problem. >> Cool. >> Okay, so let's keep going and write more equations.

## POLICIES

$$\pi^* = \operatorname{argmax}_{\pi} E \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi \right]$$

$$\underbrace{R(s)}_{\text{immediate}} \neq \underbrace{U(s)}_{\text{long term}} = E \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, s_0 = s \right] \quad \text{delayed reward}$$

$$\pi^*(s) = \operatorname{argmax}_a \sum_{s'} T(s, a, s') U(s')$$

$$\left[ \underbrace{U(s) \equiv U(s)}_{\pi^*} \right] \quad U(s) = R(s) + \gamma \max_a \sum_{s'} T(s, a, s') U(s')$$

## Bellman Equation

$T(s, a, s')$  就是第 10 段中定義的, 將其抄過來, 即: Model:  $T(s, a, s') \sim \Pr(s' \mid s, a)$

Bellman Equation 就是指它上面那一行, 記憶: URgTU, 即 Ukraine Russian give to you.

22. So, now that we've got utility fine, and we've got this  $\pi^*$  fine, we can actually do an even better job of writing out  $\pi^*$ . And let me do that. All right, so does this equation make sense, Michael? >> Let's see, so the policy, is that a star again, or is that a K. >> That's a star. So it's the optimal policy. >> All right. The optimal action to take at a state is, well, look over all the actions, and sum up overall the next states, the transition probability, so that's the probability we end up in state  $s'$ . And now we have the utility of  $s'$ , the problem being that that's not defined. >> Well, it sort of is, we defined it immediately above, at least with respect to some policy. >> But that's concerning because we don't know. The policy that you want to put in there is gotta be the policy that you're trying to find. >> Right, so in fact implicitly what I mean here is  $\pi^*$ . So, in fact, let me write that down that whenever you see me write from now on, the utility of a state, I'm almost always going to actually mean the utility of the state if I follow the optimal policy. We might call this the true utility of the state. >> I see. >> So I'm just going to write this off to the side here as something for you to remember. So this this says then that the optimal policy is the one that, for every state, returns the action that maximizes my expected utility. >> With regard to the optimal policy, it feels rather circular. >> It is rather circular, but you're a computationalist. You're a big fan of recursion. We just went through a whole exercise where we figured out the geometric series by effectively doing recursion. >> It's a similar kind of situation, for this? >> It kind of is. So, let me write one more equation down and then you'll be one step closer to actually seeing it. Of course, if we're in an infinite horizon with a discounted state, even though you're one step closer you won't actually be any closer. Well let's worry about that when we get there. So let me write one more equation down. >> We're never going to get there. It's infinitely long. >> [LAUGH] Yeah. >> Wait are you demonstrating something with this lesson by making it infinitely long? >> [LAUGH] I'm certainly demonstrating something with this lesson. I don't know what it is. So let me write this next equation down. So then the true utility of a state  $s$  is then, I'm just basically going to unroll the equation for utility. It's the reward that I get for being in that state, plus I'm now going to discount all of the reward that I'm going to get from that point on. Got it? >> All right, so once we go to our new state  $s'$ , we're going to look at the utility of that state. Okay, that's sort of fine, modular recursion. We're going to look at overall actions, which action gives us the highest value of that. Oh I see, that's kind of like the  $\pi^*$  expression just above. >> Yup. All right, so once we figure that out, we know what action we're going to take in state  $s'$ . We're going to discount that, because why? Because I guess that just kind of ups the gamma factor on all the rewards in the future. >> Right. >> And then we're going to add to that our immediate reward. Yes, okay I think I follow that. >> In some sense all I've done is I kept substituting pieces back into one another. So the true utility being in a state is the reward you get in that state, plus the discount of all the reward you're going to get at that point, which, of course, is defined as the utility you're going to get for the states that you see, but each one of those is defined similarly. And so the utility you will get for  $s''$  say will also be further discounted but since it's multiplied by gamma that will be gamma squared and then  $s'''$  will be gamma cubed, and so that's basically just unrolling this notion of utility up here. >> Okay so now it seems like all the pieces are in one place. >> Right. And so it would be nice if we were done. And I'm going to say that we're not just one step closer, but you can see an oncoming light and it is not an oncoming train, okay. So >> Yeah, [this \(最後一行黑字\) seems like a really important equation.](#) >> It is, in fact, [it's so important, it's got a name.](#) You want to guess what the name is? >> Bill >> That's actually very close. [It's Bellman Equation.](#) >> Bellman equation >> Esquire. This equation was invented by a guy named Bellman, and [it turns out to be in some sense the key equation for solving MDPs and reinforcement learning.](#) >> Wow. >> And it's actually even more [INAUDIBLE] than it looks. But basically, this is the fundamental recursive equation that defines the true value of being in some particular state. And [it accounts for everything that we care about in the MDP. The utilities themselves deal with the policy that we want to have, the gammas are discount, and all the rewards are here. The transition matrix is here, and the actions or all the actions we're going to take. So basically the whole MDP is referenced inside of here and allows us by](#)

determining utilities, to always know what's the best action to take, what's the one that's going to maximize the utility. So if we can figure out the answer to this equation, the utilities of all the states, we per force know what the optimal policy is. It becomes very easy. So we've sort of taken all that neat stuff about NDPs and stuck it in a single equation. Bellman was a very smart guy. >> So **was he the same Bellman from the curse of dimensionality?** >> Yes. >> Cool. >> There can be only one Bellman. >> [LAUGH] >> Actually, are there any more Bellmans? I don't think so, I think that they retired, like retiring a jersey. They retired his name. I could've sworn that I saw one at the last hotel that I went to. >> It was probably the same one. Oh, I get it. Hotel, Bellman, that's really good. >> Very good. >> Okay good. Well, so now that we've killed that as much as we could, let's see if we can actually solve this equation, which since this is clearly the key equation since it has a name, okay? >> Yeah, that would be cool. Especially because it looks like, if you could solve this, you could solve it, right? because then you have u. You could just plug the u in and get the u out. >> Right. And once you have the u in, and you get the u out, then you got the policy. >> Right. >> For u. >> It's always been for u. >> [LAUGH] It's for us, Michael, it's for us.

POICIES : FINDING POICIES

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

n equations in n unknowns

not-linear

- start w/ arbitrary utilities
- update utilities based on neighbors
- repeat until convergence

$$\hat{U}_{t+1}(s) = R(s) + \gamma \max_a \sum_{s'} T(s, a, s') \hat{U}_t(s')$$

23. Okay Micheal, so I've erased the screen and kept Bellman's equation, the most important equation ever and we are going to solve it. >> Oh. >> So, how are we going to make that work? Okay, so how many, so we wrote this down as a utility of s, **how many S's are there?** >> s, n, m, I don't know? >> Pick a letter. >> N. >> **n, so we n states which means this isn't really one equation.** This is how many equations? >> N. >> **Yes it's n equations. How many unknowns are there?** >> Well we have U, **the Rs are known, the Ts are known, the only things that's missing is the Us.** And there's, oh and **there's n of those as well.** >> **Right there's n equation in n unknown.** >> So we're done. Yes because we know how to solve n equations in n unknowns, right? >> If the equations were linear. >> If the equations were linear. Are these equations linear? >> I'm going to go with no. >> Why not? >> because the max is problematic. >> That's right. **This max operation makes it nonlinear.** So, it looks really good for a moment there. We've got n equations, n unknowns. We know how to solve that. But max makes for a very very very weird non linearity. And there's kind of no nice way to deal with this. Actually, one day, Michael, if you ask me, there is a cute little aside you can do where you can turn max into something



that's differentiable. >> Oh. >> But. That doesn't actually help us here so I'm not going to go off on that aside yet. >> But even differentiable wouldn't quite be linear. >> That's right. And it wouldn't help us in this case. Yeah that's exactly right. But the fact that you can have differentiable maxes I think actually [UNKNOWN]. But also unimportant for what we're talking about now. So, we've got  $n$  equations,  $n$  unknowns, they're nonlinear, which means we can't solve it the way we want to by like inverting matrices or like. You people are in the regression would normally do but it turns out that we can in fact, do something fairly similar, something that actually allows us to solve this even though it is nonlinear. And here's the equation, or the equation, here's, [here's the algorithm you use that sort of works](#), okay? So it's really simple. Just simply start with some arbitrary utilities. Declare those the answer and you're done. >> That would be one way of doing it but it turns out we can do even better. >> Wait. Start off with the correct utilities. >> That would work except we don't know what they are. >> Oh right. >> So we're going to [start with some arbitrary utilities, but then we're going to update them based on their neighbors. And then we'll simply lather, rinse, repeat.](#) Alright, so [what does that mean based on neighbors?](#) >> So, it means, based on the states, you're going to update the utility for a state based on all of the states that it can reach. So, [let me write down an equation that tells you how to update them and then maybe it'll be clear](#), okay? >> Yep. >> So we're going to have a bunch of utilities since we're going to, we're going to be looping through this. And [let's just say every time we loop through, that's time  \$t\$ .](#) Okay? So we know what the equation for utility is where the utilities are. It's just the equation that's written up here. So it's  $r$  of  $s$  plus  $\gamma$  times the mass over a of. The expected utility. Right? Except we have some estimate of the utility at time  $t$ . Okay? and, probably the right thing for me to do would be to write this like as you had or something like that. This is my estimate of the utility. And so now using this, I'm going to want to sort of update the, you know all of my utilities to make them better. And it turns out I can do that by simply doing this. [So I'm going to update at every iteration update utilities based on neighbors I'm going to update at every iteration, at every iteration my estimate of the utility of some state  \$S\$  by simply recalculating it to be the actual reward that I get for entering state  \$S\$ , plus the discounted utility that I expect given the original estimates of my utility.](#) Does that make any sense at all? >> Yeah, I guess so, so. So the  $s$ , okay, so, [so we need the whole  \$U\$  hat  \$t\$ .](#) >> Mm-hm. >> [Like at all states, because we're not just using the values that state  \$s\$  to update the values that state  \$s\$ . We're using all the values, at the, at the previous time step to update all the values to the current time step.](#) >> Yep. >> So, so all these  $n$  equations, they're all tangled together. >> Right, because of this This, expectation. So really, just to make it clear, this should be a summation over  $s$  prime.

## POLICIES : FINDING POLICIES

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

n equations      in      n unknowns

not-linear

- start w/ arbitrary utilities
- update utilities based on neighbors
- repeat until convergence

Value Iteration

$$\hat{U}_{t+1}(s) = R(s) + \gamma \max_a \sum_{s'} T(s, a, s') \hat{U}_t(s')$$

truth

此圖即為 value iteration 算法

24. So, I'm going to update the utility of  $s$  by looking at the utilities of all the other states, including itself, as prime. And weight those based on my probability of getting there given that I took an action. And what action am I going to take? Well, I'm going to take the one that maximizes my, expected utility. >> So it's sort of like figuring out what to do in a state assuming that this what really is the right answer for the future. >> Right. Now **why is that going to work? So I just made up the uhats, right? They started out as arbitrary utilities.** The next little step about updating utilities based on neighbors makes some sense because effectively is any state that you can reach. Which is determined by the transition function. But, all I'm doing is reaching states that are also made up of arbitrary utilities so, arbitrary values. **Why should that help me? Well, it's because of this value right here ( $R(s)$ ). This is actual truth.** The actual word I get for entering into a stage, is a true value. **So, effectively what's going to happen is I'm going to be propagating out the true value, or true reward,** the true immediate reward that I get for entering into a state, say. Through all of the states that I'm going to see and propagating that information back and forth. Until ultimately I converge. >> Still not obvious why we should converge, because we start off with an arbitrary function and it seems like that could be really wrong. So we're like adding truth to wrong. >> Yes but then, the next time around I'm going to, I've been adding truth twice to wrong, and then more truth to wrong, and more truth to wrong. And eventually, I'm going to be adding, more and more and more and more and more and more and more and more and more truth, then it will overwhelm the original wrong. >> And is that, does it help that the wrong is being discounted? >> Yes, it helps that the wrong is being discounted. Does it help that the wrong is being discounted? I actually don't know that matters. I'll have to think about that for a moment. It certainly doesn't hurt. But I, I guess the way that I think about this, I mean there's an actual proof you can look it up but in the end I think, I, **I tend to think of this as a kind of simple contraction proof that effectively at every step you have to be moving towards the answer. Because you've added in**

more of the reality, more of the truth. And if you remember the utility of a state is just all of the rewards that you're going to see. So, basically at every time step you have added the reward that you're going to see, and then all of the rewards you're going to see after that. And so you've gotten a better estimate of actually the sequence of rewards you're likely to see from the state. And if that gets better for any of the states, then eventually that betterness will propagate out to all the other states that they can reach or can reach them. That will keep happening and you'll keep getting closer and closer and closer for the true utility of the states until you eventually run out of closeness. >> Cool. >> Does that make sense at all? >> Well does it also help that the gamma is less than one. >> Yeah it does. the, the way I like to think of this as, is as a sort of contraction proof, that makes, if you've heard of those. So, the basic idea here is that you start out with some noise, but at every step, you're getting some truth, and that truth gets added, and then, the next iteration, more truth gets added, and more truth get, gets added. So, as you have some estimate of some particular state S, you get to update it based on truth. It's actual reward. And you bring in more truth from the other utilities, as well. As this particular utility gets better. That closeness to the true utility then gets spread to all, from, to all the states that can reach it. And because the gamma is less than one. You basically get to overwhelm the past in this case which is the original arbitrary utilities. And so as you keep iterating through this, the latest truth becomes more important than the past less truth. And so you are always getting closer and closer and closer to the truth until you eventually you do. At least that's the intuition that I like to think of. >> Yeah I, I kind of get that as an intuition, though I'd probably be happier going through the math, but. >> Well, we could do that, and by we, I mean the students can do that by actually reading the proof. >> All right. >> Okay, so cool. So, this right here is a an easy way to find the true value of states. And you do it by iterating and it kind of has a name. >> It kind of has a name. >> Yeah, what do you think the name could be? >> Bellman. Bellman's algorithm. >> No. No though that's probably reasonable. >> Utility iteration. >> Yes, except utility sounds better if you say value, so it's value iteration. And it works. Remarkably well. So, and it doesn't, doesn't give you the answer but it gets you something that is closer and closer to the answer. >> Right. And, eventually it will converge. You'll get so close to converging it doesn't matter. And, once you have the two utilities, if you recall. We know how to define the optimal policy in terms of utilities. So, if I give you the utilities, then the optimum policy is just, well I'm in a state. Look at all the states I might get to. Figure out the expectation that I'm going to get for a given action. Pick whichever one is maximum and I'm done. So solving for the utilities are the true value of a state is effectively the same thing as solving for the optimal policy. >> Hm. >> Excellent. >> That's cool. >> I think so.

(為了方便看, 本段將圖放在文字後面. 本段文字是 quiz 的題目. 下一段是 quiz 的解法, 下一段有少量標藍的)

25. >> Okay, Michael, so you seemed like you knew what you were doing so I thought I would verify that by giving you a very easy [quiz](#). >> This doesn't look so easy. >> It is easy, so here's the quiz. To help you I've written up both the Bellmans equation and the update that we would give to utilities. I neglected to write the little hats and everything because I just don't think you need that but these are the two equations that you need to be able to think about. This is just what we've written up before. And I've written down the grid world we've been playing with the entire time. And [I want you to figure out how value iteration would work from the first iteration and the second iteration for this particular state here that I marked with an X](#), okay? >> Okay. >> Alright, and there are a little more information you need to go. [Gamma in this case is going to be equal to one half](#). Mainly because it's easy to do the math if you do it that way. The rewards, just to remind you, [for all of the states except for the two goal or absorbing states is going to be minus 0.04](#). And my initial [arbitrary](#) utilities for all of the states, is going to be 0, except in the two absorbing states where I already know the utilities are 1 and minus 1 respectively, Okay? >> Okay. >> You got all that? >> I think so. >> You sure? >> [LAUGH] I feel confident that I'm going to slip up, but Okay, yeah, I think I can take a stab at this. >> Alright, so gamma's one half, rewards are minus 0.04, arbitrary starting utilities at times 0 is 0, except here at the absorbing states. Tell me how the utility here will evolve after one step, or one iteration, and two steps, or two iterations. Okay then, go.

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!



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

$$U_2(x) = \boxed{\phantom{.36}}$$

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

$T(s, a, s')$ 就是第 10 段中定義的, 將其抄過來, 即: Model:  $T(s, a, s') \sim \text{Pr}(s' | s, a)$ , 即從  $s$  到  $s'$ 之概率.

算  $U_1(x)$ :

$$U_1(x) = -.04 + \frac{1}{2} [0.1*0 + 0.1*0 + 0.8*1] = .36,$$

上式之解釋:

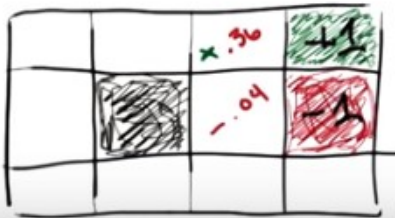
1. 已知在 x 處,  $R(s) = -.04$ .  $\gamma = 1/2$ .
2. 最先要做的, 就是選一個最好的 a. 在 x 處有四個不同的方向可以走, 即有四個不同的 a, 我們要選一個 a 使得我們得到最大值, 但由前面已知最好的 a 就是往右走, 故就直接選 a 為 往右走.
3. 然後就算  $a = \{\text{往右走}\}$  時的  $T(s, a, s')U_t(s')$  (即從 s 往右走到  $s'$ ). 由前面知, 我們要求往右走, 實際上走的方向有三種可能: 往上走(概率為 0.1), 往下走(概率為 0.1), 往右走(概率為 0.8). 往上走的結果是不動, 即  $s'$  仍為 s, 根據假設,  $U_t(s) = 0$ , 即  $T(s, a, s')U_t(s') = 0.1 * 0$ . 同理, 往下走的  $T(s, a, s')U_t(s') = 0.1 * 0$  (注意下面那一格的 reward = -0.04, 但 utility = 0). 往右走的  $T(s, a, s')U_t(s') = 0.8 * 1$ , 因為走到的是 +1 那裡.

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!



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

$$-.04 + \frac{1}{2} [.036 + -.004 + .8]$$

$$U_2(x) = \boxed{.376}$$

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

算  $U_2(x)$ :

x 下面那格 utility 為 -.04, 算 -.04 的方法跟上面一樣. 首先  $a = \{\text{往左走}\}$ , 這是為了防止落入 -1 中. 然後  $U_1(x \text{ 下面那格}) = -.04 + \frac{1}{2} [0.1*0 + 0.1*0 + 0.8*0] = -.04$ .

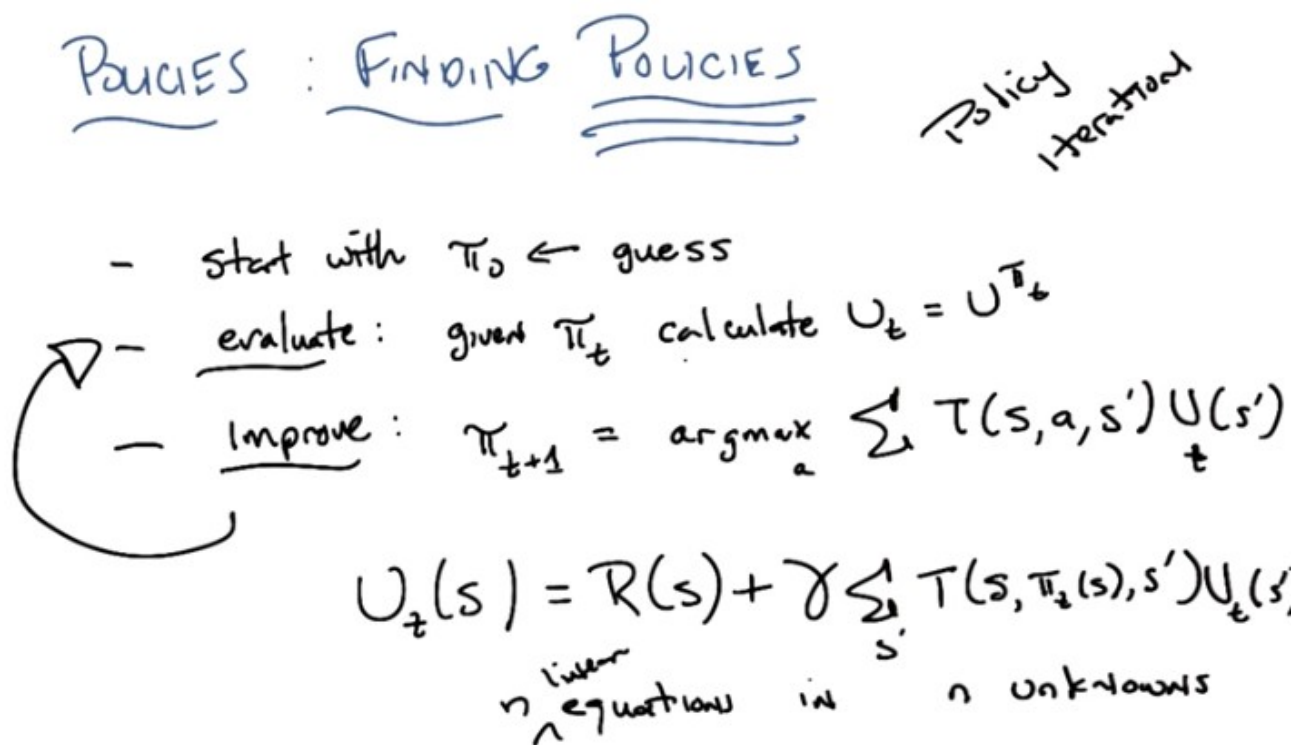
算  $U_2(x)$  的方法也一樣, 此時在 x 處, a 依然為  $a = \{\text{往右走}\}$ ,  
 $U_2(x) = -.04 + \frac{1}{2} [0.1*0.36 + 0.1*(-.04) + 0.8*1] = .376$ ,

26. Alright, Michael. What's the answer? >> I think I've already made a mistake. >> Okay. What's the mistake? >> Suggesting that we do a quiz. >> [LAUGH] I'm pretty sure that's always true. Unless Michael, by doing a quiz now and suffering pain, you, in the future, are a better person. In which case, you have made the right long term decision. >> I see. You're, this is a MDP metapoint you're making. >> Mm-hm. >> Alright. So let, but let's, let's just buckle down and do this thing. So. >> Okay. >> Alright, so at that state x, we have to consider, according to the equation, we're going to do  $U_{sub} one$  at x. And that's going to be the reward at x, which is minus .04. Mm-hm. >> Plus gamma, which is a half. >> Yes. >> Feel free to write these things down. >> Okay, so, let's see. It's going to be minus .04 plus one half times. >> Alright. So now we need to do four different actions. >> Right. >> So, I don, I



would make like a brace-y thing at this point. Not a bracket, but a brace. Alright? >> Or I could do a bracket, because you're going to notice immediately that it's obvious what the right action is. >> Okay, alright. Well, we know that the, the right action's going to be to go to the right. >> Yeah, but even then, you know you don't have to do the rest of the computation because my first guess at all the utilities is that they're 0. Which means you're always going to want to take the action that gets you to +1 with the highest chance. >> Right. >> So there's just no point in. >> I see. Okay. Fair enough. Thank you for. >> Okay. >> The shortcut. So, so we only have to do the one action which is to the right. >> Mm-hm. >> And so if we go to the right, there's three possible next states we could go in. >> Yeah. >> One is back to x, which has a value of zero. >> Mm-hm. >> One is to the thing underneath of x, which has a value of zero. And then the last one with probability 0.8 needs to go to the plus one. >> Which is. >> 0.8 times plus one. >> Which is 0.8. >> Okay. And that is? >> Okay. So 0.8 times a half is 0.4 minus 0.04 is 0.36. >> Yep. And that is correct. >> Okay, so to do the same thing for U<sub>2</sub>, we're going to need the U<sub>1</sub> value for a bunch of other states, seems to me. >> Maybe, so let's right that down. So we know for now the utility here is .36 right? >> Yeah. >> And you're saying that in order to do U two, I'm now going to have to, you know, I was able to avoid doing some of the math before because, these are all zeroes. So it was just easy to do. But when I went right, I either stayed where I was, went to the plus one, or I ended up going down. Well I think by the same argument that allowed us to cheat our way out of to cheat our way out before. >> Okay, >> It's still going to be best to go to the right, so. >> Yeah but I know that but the value itself is going to depend on the value in several other states. >> Yeah well how many other states? >> Oh, just that one. >> Just this one, so what was U one of this state? I see. So, presumably, we want to avoid falling into the pit, so the, the best thing we can do is bash our head against the wall, >> Mm-hm >> Which will get us a -.04 for that statement. >> Right. >> Okay, alright, so maybe this isn't so, so bad. >> Mm-hm. >> So now for our u<sub>2</sub> values we need -.04 plus a half times point, point one times point 36. Plus. >> Mm-hm. >> 0.1 times negative 0.04, so that's minus 0.004, plus 0.8 times one. Oh, which is 0.8 again, just like it was last time. >> Yep. >> And I get 0.376. >> Which is what I get by also getting out your calculator. Okay, so 0.376, and you can imagine how we would do that on and on. (關於算 U<sub>2</sub>) I want to point something out, Michael, which is that. You decided to figure what the true utility was for 「the state under X」 by bashing your head into the wall (由後面知, 此處的意思不是往上走). But you know that based on the discussion we had earlier that actually the optimal policy would involve going up instead of bashing your head into the wall. What you did at that point was in fact right because everything else in this utilities were all zero. The best thing you could do is avoid getting, avoid ever falling into minus one, so the policy of the very first of bashing your head into the wall, is infact the right thing to do at that point. But what you'll notice is that next time around the utility for the X state is bigger than zero. In fact, will keep getting bigger and bigger than zero. As you can see, it went from .36 to 0.376. Which means that at some point it's going to be worthwhile to try to go up instead of bashing your head into a wall. >> I see. So this, so this is kind of cool that it works. But, it does seem like a really roundabout way of getting there. I mean, is there some way that we could some, I don't know, maybe take advantage of the fact that there's not that many policies? >> Yeah, so you actually said something, fairly subtle there, so let's see if we can unpack it. So, lemme point out two things, which I think will get us to what you're answering. The first is. Do you realize that the reason that the value iteration works is because eventually value propagates out from its neighbors, right? The first time we can calculate the .36 without really worrying about anything around it because the utilities are all zero, and in fact, based on the initial utilities. Even for this state here, we do the wrong thing. But after some time, this state becomes a truer representation of its utility and, in fact, gets higher and higher. The right thing to do here will go up. You'll also notice that after another time step I'm going to need to start looking at the value of this state as well. Alright. So, eventually I'm going to have to figure out the value, the utilities or the values of all of these states and this plus one is going to propagate out towards the other states. Where this minus one will propagate out less because you're going to try to avoid falling in there. So that makes sense, right? But

what's propagating out, Michael? What's propagating out is the true utilities, the true values of these states. But what's a policy? A policy is a function from what to what? >> States to actions. >> Right, a policy is the mapping from state to action. It is not a mapping from states to utilities. >> No, that's what U is. >> That's what U is. >> Or that's what you are. >> So, [LAUGH], so if we have U, we can figure out pi, but U is actually much more information than we need to figure out pi. If we have a U that is not the correct utility, but say, has the ordering of the actions correct, then we're actually doing pretty well, right? It doesn't matter whether we have the wrong utilities. >> I see. Uh-huh. >> You'll remember we did this in the, the first third of the class as well when we noticed that we were computing in the Bayesian learning case actual probabilities. But we don't really care about actual probabilities. We just care that the labels are right. There's a very similar argument here. We don't care about having the correct utilities, even though by having the correct utilities, we have the right policy. All we actually care about is getting the right policy. And order matters there rather than absolute value. Does that make sense? >> Yeah, that's interesting. It's almost kind of like pi is more of like a classifier, right? It's mapping. Inputs to discrete classes and the user kind of like, more like regression where its mapping these states to continuous values. >> Right and given one, given the utilities, we can find pi, given pi there's an infinite number of utilities that are. Consistent with it. So, what you end up wanting to do is get a utility that's good enough to get you to your pie. Which is one reason why you don't have to worry about getting the absolute convergence in [UNKNOWN]. But it gives us a hint of something we might do that's a little bit better with the policies that might go faster in practise. So, I'm just going to take three seconds to give you an example of that. Okay? >> Awesome.



此圖即為 policy iteration 算法

27. Okay so Michael let's see if we can do something that might be **faster** and just as good as what we've been doing with value iterations. And what we're going to do is **we're going to emphasize the fact that we care about policies, not values.** Now it's true given the true utilities we can find a policy, but **maybe we don't have to find the true utilities** in order to find the optimal policy. So, here's a little sketch

of an algorithm okay, so it's going to look a lot like value iterations. We are going to start with some initial policy, let's call it  $\pi_0$  and that's just going to be a guess, so it's just going to be an arbitrary setting of actions that you should take in different states. Then we're going to evaluate how good that policy is, and the way we're going to do it at time  $T$ . Is to calculate its utility, which I'm going to call  $U_t$ . Which is equal to the utility you get by following that policy. Okay? And I, I'll show you in a moment exactly how you do that, alright? But I just want to make certain that you, that you, you kind of buy that maybe we can do that. So, We have, given a policy, we ought to be able to evaluate it by figuring out what the utility of that policy is. And, again, we'll talk about that in a second. And then, after we know what the utility of that policy is, we're actually going to improve that policy in a way similar to what we did with value iteration, we're going to update our policy, Time  $T$  plus one, to be the policy. That takes the actions that maximizes the expected utility based of what we just calculated for  $\pi_t$ . Now notice it will allow us to change  $\pi$  over time because imagine that we discover that in some state that actually there is an action that is very good in that state. That gets you some place really nice gives you a really big reward and that went on and you do fairly well. Well then all other states that can reach that state. Might end up taking a different action than they did before, because now the best action would be to move towards that state. So these two steps will actually, or can actually lead to some improvement of the policy over time. But the key thing here is we have to figure out exactly (evaluate 那步中) how to compute  $U_t$ . Well, the good news is we know how to do that, and it's actually pretty easy. And it boils down to our favorite equation, Bellman's equation. >> [LAUGH] >> So our utility at time  $t$ , that is the utility that we get by following a policy at time  $t$ , is just well, the true reward that we're going to get by entering that state plus  $\gamma$  times the expected utility, which now looks like this. There, does that make sense? Do you see that? >> Okay, hang on, it looks a little different from the other equation. So did you mean for it to have  $U_t$  is defined in terms of  $U_t$  and not  $U_{(t-1)}$ ? >> Yes. >> Okay, that's interesting. And the max is gone but instead of max, there's a policy stuck into the transition function. >> Yep. >> A choice of action is determined by the policy. >> Right. And that's actually the only difference between what we were doing before is that rather than having this max over actions, we already know what action we're going to take. It's determined by the policy we're currently following. >> Okay, but isn't this just as hard as solving? The thing with the max, you said? >> Well, what was the problem that we were solving before with the max? >> That was the Bellman equation. >> Yes, but we were solving a bunch of equations. How many of them? >>  $N$ . >> So we were solving  $n$  equations, and how many unknowns? >>  $N$ . >> What's the difference between this,  $n$  equations and  $n$  unknowns, and the other  $n$  equations and  $n$  unknowns? >> Well,  $n$  is the same. Mm hm. >> There's no max, though. >> There's no max. And what was it that made solving that hard before? >> It made it, the max made it non linear. The max is gone now. You're saying this is, this is a set of linear equations? >> Yeah. Because, well, there's, there's just a bunch of sums. And the  $\pi$  is not like some weird function. This is just effectively a constant. >> I see. >> So now, I have  $n$  equations, and  $n$  unknowns. But it's in linear equations. And now that I have in linear equations and unknowns, I can actually compute this is a reasonable amount of time, by doing matrix inversions, and regression, and other magic hand wavey things. >> That's very slick. >> Yeah. >> It's seems, it's still more expensive than doing the valued iteration, I guess. >> Yeah, but you don't have to, perhaps, do as many iterations as you were doing before. So once you've evaluated it, which we now know how to do, and you've improved it, you just keep doing that until your policy doesn't change. >> Very cool. >> Mmhm. And this does look a lot like value iteration to you, doesn't it? >> Yeah, though it seems like it's making bigger jumps somehow. >> It is, and that's because instead of making jumps. In value space, it's making jumps in policy space. Which is why we call this class of algorithms Policy Iteration. >> Cool. >> Right. Now, this inversion can still be fairly painful, it's, you know, if we don't worry about being highly efficient, you know, it's roughly  $n^3$ , and if there are a lot of states, this can be kind of painful. But it turns out there's little tricks you can do, like do a little step evaluate iteration here for a while to get an estimate and then, you know, kind of cycle through. So there's all kinds of clever

things you might want to do, but at a high level without worrying about, you know, the, the details of constants, this general process of moving through policy space. And taking advantage of the fact that by picking a specific policy you're able to turn your nonlinear equations into linear equations turns out to often to be very helpful. >> So, [is it guaranteed to converge?](#) >> Yes. >> Nice. >> Well there, that was easy. I'm, I'm not going to go into it but, you know, there's a finite number of policies. You're always getting better so eventually you have to converge. It's very similar to the, or at least intuitively it's very similar to the argument you might make for [UNKNOWN]. Cool.

- MDPs : states, rewards, actions, transitions (discount)
- policies
- value functions (utilities)
  - ↳ long term!
- discounting ! infinite, finite
- stationary
  - ↳ value iteration
- BELLMAN
  - ↳ policy iteration

28. Okay Michael, so, this is it. Why don't you help me remember what we learned in this one marathon session that was not all distributed over multiple months. >> [LAUGH] Sure. Or years. >> Mm hm. >> So, MDPs. So, we talked about mark-off decision processes, and we said what that meant. [LAUGH] >> Okay, so, that's the first thing we did is we talked about MDPs. >> And that, that MDP consists of states and rewards and actions and like that, transitions, discounts. >> So you said discounts and I just want to point out that there are some people who think that that's a part of the definition of the problem and there are some people who think that that's more of a parameter to your algorithm. okay, alright. So there's some people who think of it the right way. >> Like me. And some people think of it the wrong way. So, I tend to think of the discount as being something that you're allowed to fiddle with, as opposed to it being a fundamental part of the. >> Then why not fiddle with the rewards? >> Sure. You are allowed to fiddle with the rewards. >> Oh! You are very open minded! >> I am open minded, i believe that rewards should be able to marry other rewards. In any case, the important thing here is that there is some under lying process that you care about that is suppose to represent the world, states, rewards, actions, and transitions and capture that fairly well. Discount is, in some sense, an important part of the problem, because it tells you how much you want to care about the future versus the past. But it's reasonable to think of that as something that you might want to change outside of the kind of underlying physics of the world. >> I see. Okay, that's fair. Yeah, in some sense, the states and the actions and the transitions represent. The physical world and the rewards and the discount represent the kind of task description. >> Right. And of course, you say that, but if you, you could decide to define states differently, and doing that would impact both your actions and the

transition function and so on and so forth. But the basic idea is right, which is, there's some underlying process we're trying to capture, and I think it's exactly right to say, states, actions, and transitions. Sort of capture that. And rewards and discounts capture more about the nature of the task, you're trying to do in that underlying world. >> Okay. And in, in that context we talked about two really important concepts. Policies and, value functions? >> Mm-hm. >> Which we sometimes call utilities. >> Right. >> And **how do utilities differ from rewards? The utilities factor in the long term aspects, and the rewards are just telling you the moment to moment.** >> Right. >> Utilities are like a group of rewards. Like a gaggle. >> Or a murder. >> Of crows. So, that we talked about how we can assign value to an infinite sequence of rewards. Mm-hm. >> But it helps if we use discounting to do that so that we don't get infinitely large sums. >> And that allowed us to deal with infinite sequences, but treat them as if their value is, in fact, finite, thus solving the immortality problem. >> [LAUGH] And, let's see, then we kind of. And the stationarity was a really important part of that. >> Yep. In fact kind of drove everything. >> Yeah. And all these things were tied up together in the Bellman equation itself. >> Yes which is an awesome equation and deserves capital letters. [LAUGH] Is that it? And then, well then we solve the Bellman equation using value iteration and policy iteration. >> Oh, yes. >> I think that was it. Alright, so are any of these polynomial time algorithms? >> Well, the way we've been talking about them, no, but you can map these into linear programs and turn them into polynomial problems, or polynomial. So, yes these problems can be solved that way. But actually, that reminds me, of something that we haven't said and that we haven't learned today, that I think is worth mentioning. Which is we been talking about, you know, this section of the class is about over the course is about reinforcement learning. But, **we actually haven't done any reinforcement learning here because we know everything,** we know the states, we know the rewards, we know the actions, we know the transitions. We have some discount, we have been solving MDP's, but that's not quite the same thing [SOUND] as doing reinforcement learning, however, it's very important to do these things, to make it easier to think about how reinforcement learning works. So, **in reinforcement learning, the difference is, you don't necessarily know, the rewards and the transitions or even the states and the actions (所以才有了下节的 Q Learning 算法, Q Learning 才是真正的 learning), for that matter.** I see. So when are we going to get into reinforcement learning then? >> Next time. And when I say we next time, I mean you next time. That's your assignment. Learn all about reinforcement learning and talk about it next time. >> All right. I gotta go and get to work then. >> Okay. Well, you have a good one Michael. >> Thanks for all this. It's really cool. >> It is cool. Bye. >> Bye. >> Bye.