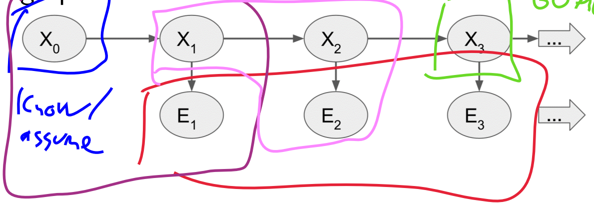
Oct 26 HMM Wrapup and MDP

Opening Example: Sketch the FORWARD algorithm to find $P(X_3|E_{1:3})$ on the given graph.

t=0→1 t=1→2 GOAL.



known/assume

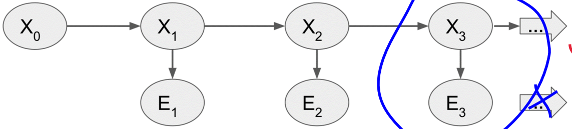
observe

What about $P(X_5|E_{1:3})$?

Oct 26 HMM Wrapup and MDP

Opening Example: Sketch the FORWARD algorithm to find $P(X_3|E_{1:3})$ on the given graph.

forward to here



$X_4 \rightarrow X_5$

go along

$p(X_4|X_3)$

$P(X_5|X_4)$

We use Bayes', and sum over the prior X value (iteratively):

$$P(X_3|E_3) = \frac{P(E|X)P(X)}{\cancel{P(E)}}$$

Bayes thm.

$$\underset{\alpha}{\propto} P(E|X) \boxed{\sum_{prior\,X} P(X|prior\,X)P(prior\,X)}$$

$P(X)$ based on prior steps

What about $P(X_5|E_{1:3})$?

Announcements and To-Dos

Announcements:

1. Skip 1a for now but it's worth a bit of E.C. if you get A^* working. I'll add a few edges to hard code in an addendum.

input (start node) (end node)) disconnect?

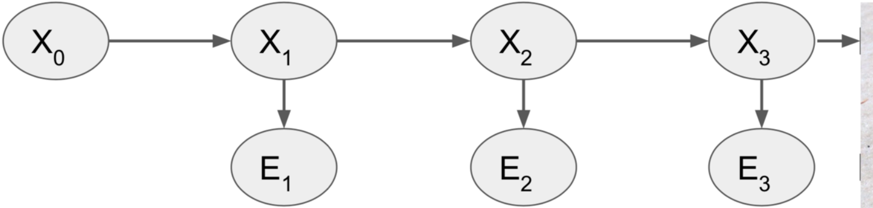
return shortest path

Last time we learned:

1. Stationary distributions to Markov Models.

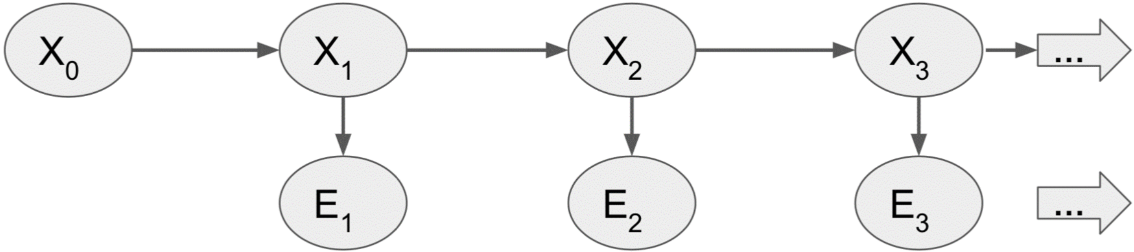
Hidden Markov Models

Example:
Suppose you are a graduate student in a basement office. You are writing your dissertation, so you don't get to leave very often.
You are curious if it is raining, and the only contact you have with the outside world is through your advisor. If it is raining, she brings her umbrella 90% of the time, and has it just in case on 20% of sunny days. You know that historically, 40% of rainy days were followed by another rainy day, and 30% of sunny days were followed by a rainy day.



HMM: Filtering

Filtering: The goal is to predict X_{t+1} *given* all the evidence available $E_{1:t+1}$.
At $t = 0$:
$P(X_1|E_1) = P(E_1|X_1) \sum_{X_0} P(X_1|X_0)P(X_0)$
At $t = 1$:
$P(X_2|E_{1:2}) = P(E_2|X_2) \sum_{X_1} P(X_2|X_1)P(X_1|E_1)$
We continue FORWARD through the graph.



HMM: Prediction

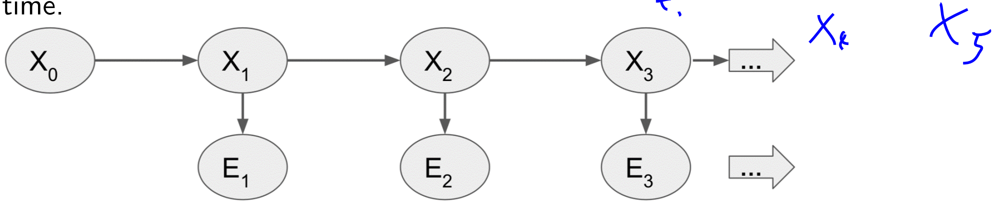
Prediction: The goal is to predict X_{t+k+1} *given* all the evidence available $E_{1:t+1}$.
The prediction of X two $(k=1)$ time steps beyond where our evidence ended was:

$$P(X_{t+2}|E_{1:t}) = \sum_{X_{t+1}} P(X_{t+2}|X_{t+1}) \sum_{X_t} P(X_{t+1}|X_t) P(X_t|E_{1:t})$$

steps of t. Markov Markov from forward

Making a k-step prediction just means doing FORWARD-steps up until we're out of evidence, and then following the Markov process to evolve $X_{t+1}|X_t$ until we reach the desired future time.

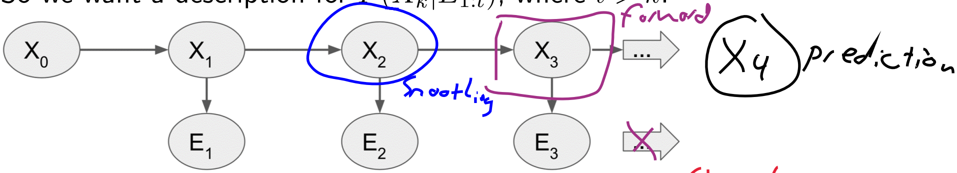
t.



HMM: Smoothing

Our final task is **smoothing**, where we try to update probabilities of prior states X based on current evidence.

So we want a description for $P(X_k|E_{1:t})$, where $t > k$.



past ↓ present ↓ past after k. past

$$\underbrace{P(X_k|E_{1:t})}_{P(X_2|E_{1:3})} = P(X_k|\boxed{E_{1:k}}, \boxed{E_{k+1:t}})$$

$$= \alpha P(E_{k+1:t}|X_k, \cancel{E_{1:k}})\underbrace{P(X_k|E_{1:k})}_{\text{Bayes}}$$

$$= \alpha P(E_{k+1:t}|X_k)\underbrace{P(X_k|E_{1:k})}$$

We can find the last term by the FORWARD algorithm for filtering.

HMM: Smoothing

This leaves the $P(E_{k+1:t}|X_k)$ term, which we denote by $b_{k+1:t}$, which is the probability of future *measurements* given the current state of our system, which is just the combination of our transition and sensor models! Imagine taking *one* time step and asking about the new evidence: we need to describe X_{k+1}.

missing X_{k+1}

Past $\to X_k \to X_{k+1} \to X_{k+2}$

E_k Given E_{k+1} E_{k+2}

sum over the unobserved

$$
\begin{aligned}
b_{k+1:t} &= P(E_{k+1:t}|X_k) \\
&= \sum_{X_{k+1}} P(E_{k+1:t}|\underbrace{X_k, X_{k+1}}_{indep})P(X_{k+1}, X_k) \\
&= \sum_{X_{k+1}} P(\underbrace{E_{k+1:t}}_{split\ up}|X_{k+1})P(X_{k+1}, X_k) \\
&= \sum_{X_{k+1}} P(\underbrace{E_{k+1}, E_{k+2:t}}_{conditional}|X_{k+1})P(X_{k+1}, X_k) \\
&= \sum_{X_{k+1}} \underbrace{P(E_{k+1}|X_{k+1})}_{sensor\ model}\underbrace{P(E_{k+2:t}|X_{k+1})}_{similar\ to\ LHS}\underbrace{P(X_{k+1}, X_k)}_{Markov\ model}
\end{aligned}
$$

HMM: Smoothing

The middle term is a *backwards* model, since we need the future value of $P(E|X)$ rather than the past, like $\textsc{Forward}$ did.

$$
\begin{aligned}
b_{k+1:t} &= P(E_{k+1:t}|X_k) \\
&= \sum_{X_{k+1}} \underbrace{P(E_{k+1}|X_{k+1})}_{sensor\,model} P(E_{k+2:t}|X_{k+1}) \underbrace{P(X_{k+1}, X_k)}_{Markov\,model} \\
&= \textsc{Backward}(b_{k+2:t}, E_{k+1})
\end{aligned}
$$

recursive

All told, then, we have:

$$
\begin{aligned}
P(X_k|E_{1:t}) &= P(X_k|E_{1:k}, E_{k+1:t}) \\
&= \alpha P(E_{k+1:t}|X_k, E_{1:k}) P(X_k|E_{1:k}) \\
&= \alpha P(E_{k+1:t}|X_k) P(X_k|E_{1:k}) \\
&= \alpha \textsc{Backward} \times \textsc{Forward} \\
&= \alpha(b_{k+1:t}) \times (f_{1:k})
\end{aligned}
$$

past X

current/future E

forward to k

Back to k

HMM: Smoothing

What does this actually look like?

$$P(X_1|E_{1:3}) = \text{What's the probability it rained on Day 1 given 3 days of evidence (TFT)?}$$
$$= \alpha f_{1:1} b_{2:3}$$

use of $E_1 = T$

use of $E_2 = F$ $E_3 = T$

$$f_{1:1} = \text{What's the probability it rained on Day 1 given evidence through day 1?}$$
$$= P(X_1|E_{1:1}) = \alpha P(E_1|X_1) \sum_{X_0} P(X_1|X_0) P(X_0|E_{null})$$
$$= \begin{pmatrix} .708 \\ .292 \end{pmatrix}$$

time 0?

HMM: Smoothing

We have to run BACKWARDS for $k = 1$ and $k = 2$. ($t = 3$ for both!)

$$b_{2:3} = \text{What's the probability of the evidence on days 2 and 3, given X at day 2?}$$

evidence @ time 2 & 3 tells us...

$$= P(E_{2:3}|X_2)$$

$$= \sum_{X_2} P(E_2|X_2)\boxed{P(E_{3:3}|X_2)}P(X_2|X_1)$$

evidence at time 3 given X_2

$$= \sum_{X_2} P(E_2|X_2) b_{3:3} P(X_2|X_1)$$

$$b_{3:3} = \text{What's the probability of the evidence on days 3-3, given X at day 2?}$$

$$= P(E_{3:3}|X_2)$$

$$= \sum_{X_3} P(E_3|X_3)P(E_{4:3}|X_3)P(X_3|X_2)$$

$$= \sum_{X_3} P(E_3|X_3)\boxed{b_{4:3}}P(X_3|X_2) \text{ but } b_{4:3} = 1 \text{ by independence!}$$

last location of evidence

HMM: Smoothing

We have to run BACKWARDS for $k = 1$ and $k = 2$. ($t = 3$ for both!)

$$b_{2:3} = \sum_{X_2} P(E_2|X_2) b_{3:3} P(X_2|X_1)$$

$$b_{3:3} = \sum_{X_3} P(E_3|X_3) P(X_3|X_2)$$

$P(E_3 = T | X_3)$

$X_2: T/F$ $X_2: T/F$

$$= \alpha \left[\underbrace{\begin{pmatrix} 0.4 \\ 0.3 \end{pmatrix} (0.9)}_{X_3=T} + \underbrace{\begin{pmatrix} 0.6 \\ 0.7 \end{pmatrix} (0.2)}_{X_3=F} \right] = \begin{pmatrix} 0.48 \\ 0.41 \end{pmatrix}$$

Sensor Model

X_t	$P(E_t|X_t)$
T	.9
F	.2

Transition Model

X_t	$P(X_{t+1}|X_t)$
T	.4
F	.3

6

7

Initializations and Evidence

$E_{1:3} = [T, F, T]$,

$X_0 = [.5, .5]$

HMM: Smoothing

We have to run BACKWARDS for $k = 1$ and $k = 2$. ($t = 3$ for both!)

$$b_{2:3} = \sum_{X_2} P(E_2|X_2) b_{3:3} P(X_2|X_1)$$

$b_{3:3}$ top

$b_{3:3}$ bottom

$$= \alpha \left[\underbrace{\begin{pmatrix} 0.4 \\ 0.3 \end{pmatrix} (0.48)(0.1)}_{X_2=T} + \underbrace{\begin{pmatrix} 0.6 \\ 0.7 \end{pmatrix} (0.41)(0.8)}_{X_2=F} \right] = \begin{pmatrix} 0.68 \\ 0.32 \end{pmatrix}$$

$P(E=T)$

?

Sensor Model

X_t	$P(E_t\|X_t)$
T	.9
F	.2

$E_2 = F$

.1

.8

Transition Model

X_t	$P(X_{t+1}\|X_t)$
T	.4
F	.3

Initializations and Evidence

$E_{1:3} = [T, F, T]$,
$X_0 = [.5, .5]$

HMM: Smoothing

Sensor Model

X_t	$P(E_t \mid X_t)$
T	.9
F	.2

Transition Model

X_t	$P(X_{t+1} \mid X_t)$
T	.4
F	.3

Initializations and Evidence

$E_{1:3} = [T, F, T]$,
$X_0 = [.5, .5]$

Day 2:

Evidence = no umbrella

We had two calculations:

P(rain)

$$P(X_1 \mid E_1) = \begin{pmatrix} 0.708 \\ 0.292 \end{pmatrix} \qquad P(X_1 \mid E_{1:3}) = \begin{pmatrix} 0.682 \\ 0.318 \end{pmatrix}$$

Sanity check? Why is the $P(X_1 = T)$ smaller with more evidence?

... maybe sun on day 2

... maybe sun on day 1?

HMM: Smoothing

Sensor Model

X_t	$P(E_t \vert X_t)$
T	.9
F	.2

Transition Model

X_t	$P(X_{t+1} \vert X_t)$
T	.4
F	.3

Initializations and Evidence

$E_{1:3} = [T, F, T]$,
$X_0 = [.5, .5]$

We had two calculations:

$$P(X_1|E_1) = \begin{pmatrix} 0.708 \\ 0.292 \end{pmatrix} \qquad P(X_1|E_{1:3}) = \begin{pmatrix} 0.682 \\ 0.318 \end{pmatrix}$$

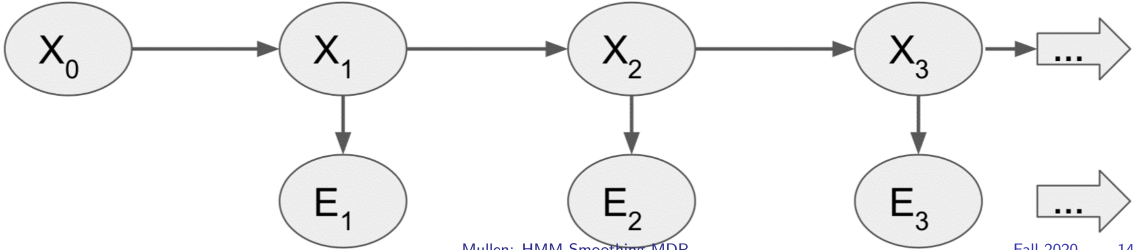
Sanity check? Why is the $P(X_1 = T)$ smaller with more evidence?
Solution: we saw evidence on rain on day 2, which in turn means it was likely to have rained on day 2... and a rainy day more likely preceded by another rainy one!

HMM: All at Once

So for any given observation X_k, we tend to have to run both a forward algorithm to ask what the evidence *up to time* k did, then a backwards algorithm to ask what the evidence afterwards did. To solve the whole chain, we do *both*. Given evidence up to time t, we:

- Run the FORWARD algorithm to filter it.
- then run the BACKWARD algorithm to smooth it

We use $f_{1:k}$ in the backwards algorithm, so we'll save them: the main tenet of dynamic programming is to not solve the same problem twice!



HMM: Wrapup

There's a final question that often is asked: what's the *most likely* sequence of X values that gave rise to our evidence.

- Lazy way: compute the $P(X|E)$ values and pick the most likely one for each time individually.
- Rigorous way: compute a maximization over all the nodes of $P(X_0, X_1, \dots X_t, E_0, E_1, \dots E_t)$

It turns out the rigorous problem can heavily exploit our independence assumptions, as usual! The joint density of the HMM will factor into

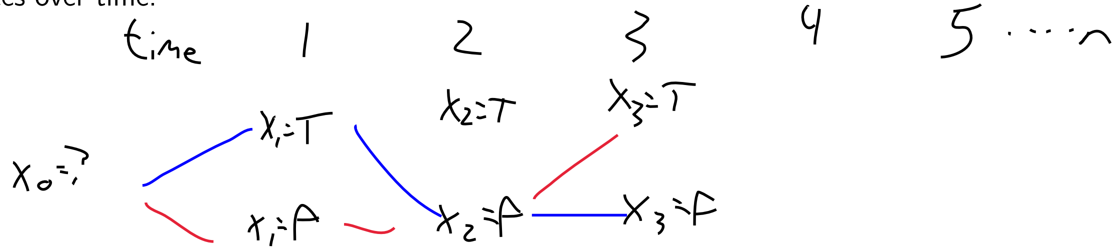
$$\Pi_{all\,nodes} P(Z_i|\text{parents}(Z_i))$$
$$= P(X_0)P(X_1|X_0)P(E_1|X_1)P(X_2|X_1)P(E_2|X_2)\dots$$

HMM: Paths

AND

$$P(X_0)P(X_1|X_0)P(E_1|X_1)P(X_2|X_1)P(E_2|X_2)\dots$$

This is the probability of a specific sequence or *path* through the graph of X states-over-time.



Path: TFF

Path: FFT

HMM: Viterbi Overview

The recursive algorithm for this is called the *Viterbi* algorithm. But lets think about the calculation by hand, briefly. Suppose we have an initial probability of $P(rain) = 1$. Then we observe over 2 days: *Umbrella*, *None*.

$E_1 = T$ $\quad E_2 = F$ $\qquad X_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Sensor Model

X_t	$P(E_t \mid X_t)$
T	.9
F	.2

Transition Model

X_t	$P(X_{t+1} \mid X_t)$
T	.4
F	.3

Initializations

$E_{1:2} = [T, F]$, $X_0 = [1, 0]$

Our task now is to compute and compare:

1. $P(X_1 = rain|E)\, vs.\, P(X_1 = sun|E)$ Forward to X_1
2. $P(X_2 = rain|E)\, vs.\, P(X_2 = sun|E)$ " to X_2
3. The full *chains* where one of (1) is true and one of (2) is true. We have to compare all of RR, RS, SR, SS and choose the *most likely* sequence.

4 chains of length 2.

HMM: Viterbi Overview

Sensor Model

X_t	$P(E_t \mid X_t)$
T	.9
F	.2

Transition Model

X_t	$P(X_{t+1} \mid X_t)$
T	.4
F	.3

Initializations

$E_{1:2} = [T, F]$, $X_0 = [1, 0]$

We can compute two probabilities at time $t = 1$, just as in forward-stepping.

1. Joint probability that evidence at $t = 1$ was U and the true value was $rain$. This is

$$P(X_1 = T, E_1 = T) = P(E_1 = T \mid X_1 = T)\boxed{P(X_1 = T)}$$

$$P(X_1 = T, E_1 = T) = P(E_1 = T \mid X_1 = T) \sum_{X_0} P(X_1 = T \mid X_0)$$

split/sum over

2. Joint probability that evidence at $t = 1$ was U and the true value was sun. This is F

$$P(X_1 = F, E_1 = T) = P(E_1 = T \mid X_1 = F)P(X_1 = F)$$

$$P(X_1 = F, E_1 = T) = P(E_1 = T \mid X_1 = F) \sum_{X_0} P(X_1 = F \mid X_0)$$

HMM: Viterbi Overview

We then *classify* what was the most likely outcome at the first time step. This is also a way to ignore the denominator (α) from Bayes that occurred in forward-stepping.

The key to the Viterbi is to then use the joint probabilities through time $t = 1$ to compute probabilities up to time $t = 2$. There are two ways to get a "rain" at time $t = 2$: the ones that came from $X_1 = rain$ and the ones that came from $X_1 = sun$, and we previously computed each of those.

Writing down the full probability for $P(X_2 = rain)$ is messy, but instead we just break down using the cases on X_1 we already have.

HMM: Viterbi Sketch

We can update this recursively, as well.

$t=1$ $\qquad$ $t=2$ $\qquad$ END $\qquad$ Pick larger

$P(X_0)$ $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ = more likely than $x_0=F$

$P(X_1=T \mid \text{evidence})$ $.3$ $\rightarrow$ $P(X_2=T)$ $.05$ $\rightarrow$ $P(X_3=T)$ $.002$

$P(X_1=F \mid \text{evidence})$ $.1$ $\rightarrow$ $P(X_2=F)$ $\qquad$ $P(X_3=F)$ $.001$

$X_2=F \mid X_1=T$ $\qquad$ $X_2=F \mid X_1=F$

Store:

$P(X_t=T)$

$P(X_t=F)$

(Forward...ish)

$E=T$ $\qquad$ $E=F$

Store more likely parent of each i

$(X_3=F) \leftarrow \text{parent: } (X_2=F)$

HMM: Most Likely Sequence

$$P(X,E) = P(X_0)P(X_1|X_0)P(E_1|X_1)P(X_2|X_1)P(E_2|X_2)\ldots$$

So we want to maximize this thing... but maximizing products is harder than sums, so we hit with a $\log$ for numerical stability, which keeps the max in the same place and changes products to sums.

$$\log P(X,E) = \log\left(P(X_0)P(X_1|X_0)P(E_1|X_1)\right) + \sum \log\left(P(X_k|X_{k-1})P(E_k|X_k)\right)$$

The recursive algorithm for this is called the *Viterbi* algorithm and is computed in linear time. It's just the calculations above where, for each time step and for each state (T/F, umbrella or not), we store the summed up probabilities of "all the ways we could get to this state."

To choose the best path at the end, we just work backwards. We choose the best *state* at the end, and then go backwards along the graph asking what that states' most likely predecessor was.

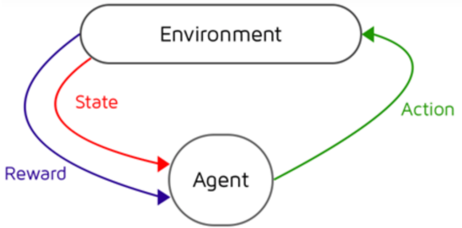
Markov Decision Process

A *Markov Decision Process* (MDP) is a sequential decision problem that is the combination of a Markov Model and a decision-making agent. It asks the question: how do we maximize utility if there are uncertainties associated with the successor states of each action? To do this, we require:

1. A fully observable, stochastic environment
2. A Markov transition model that gives probabilities of states *given* decisions
3. An additive reward structure

They are often used for

1. Inventory management
2. Routing/logistics
3. Games
4. Planning under uncertainty



MDPs

Consider an agent-based game. We win if we reach the treasure. We lose if we run into the internet troll or the dragon.

Goal: describe the appropriate set of moves from our current location to the treasure.

MDPs

Consider an agent-based game. We win if we reach the treasure. We lose if we run into the internet troll or the dragon.

Goal: describe the appropriate set of moves from our current location to the treasure.

Suppose the Dragon and Troll can't move. What do we do?

MDPs

Consider an agent-based game. We win if we reach the treasure. We lose if we run into the internet troll or the dragon.

Goal: describe the appropriate set of moves from our current location to the treasure.

Suppose the Dragon and Troll can't move. What do we do?

Twist: suppose, as in our trip to Taco Bell, we sometimes get disoriented, and move in a *different* direction than the one we choose!

MDP Uncertainty and Choice

The MDP is meant to describe a real world process where actions are not perfectly reliable. Suppose we describe a transition model:

1. *Given* our intended action, the probability we move where we intend is .8.
2. *Given* our intended action, the probability we move to either side (90°) is .1 each.

Definition: A *policy* is what we would tell our agent to do in any given possible state s. Denote $\pi(s)$ by the policy chosen at state s.

MDP Rewards

The MDP is meant to describe a real world process where actions are not perfectly reliable. Suppose we describe a transition model:

1. *Given* our intended action, the probability we move where we intend is .8.
2. *Given* our intended action, the probability we move to either side (90°) is .1 each.

Definition: A *policy* is what we would tell our agent to do in any given possible state s. Denote $\pi(s)$ by the policy chosen at state s.

MDP Rewards

To choose a policy we need a notion of what makes a move good or bad. Suppose that:

1. Moving to the dragon or troll achieves a reward of -1, and ends the game.
2. Moving to the treasure achieves a reward of 1, and ends the game.
3. We can define a reward $R(s)$ (or maybe $R(s \to s')$) associated with moving to any state.

We may even encode a reward for the non-movement $s \to s$. For example, $R(s \to s) > 0$, an agent will rarely move, whereas a reward of $R(s \to s) = -2$ will create a frenetic, always-moving agent.

MDP Utility

Since rewards may not exist on all actions, we need to conceptualize an *expected* rewards or a *long-run* rewards. These like in the *utility* associated with each state.

Utility is our long-run gain.

1. It depends on the entire *sequence* of states visited, $[s_0, s_1, \ldots s_{50}, s_{51}, \ldots]$
2. Informal definition: utility is the *sum* of rewards achieved over a set of states/movements: $U[s_0, s_1, \ldots s_{50}, s_{51}, \ldots] = R(s_0) + R(s_1) + \ldots$

Classically, two terms are added to clarify and add tuning to the concept of utility.

MDP Utility

Definition: The *Time Horizon* of an MDP can be either:

1. *Finite Horizon*, where after a fixed time N no actions matter. Here we consider the rewards or utility $U[s_0, s_1, \ldots s_N, s_{N+1}, \ldots] = U[s_0, s_1, \ldots s_N]$. The length of the horizon may impact your decisions.
   Example: It's the first/last lap of your game of Mario Kart. Should you save your star piece for when someone tries to shoot you?

2. *Infinite Horizon*, where there is never a reason to behave differently in the same state at different times.
   Example: *When* you get the treasure doesn't matter, only whether you get there.

Definition: The *discount factor* of an MDP is a multiplicative punishment γ for taking longer to reach rewards. It's common in finance as it represents an increased value of immediate rewards over future rewards.

$$U[s_0, s_1, \ldots s_{50}, s_{51}, \ldots] = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \ldots$$

MDP Goal

So we have:

1. A Markov chain that gives successors *given* actions
2. Rewards associated with each state
3. A utility that may track rewards of a *sequence* of states
4. A possible discount on *when* we get rewards
5. A preference for how many total moves matter

Goal: The output of an MDP is an *optimal policy* that specifies where to move from a given starting state s. We maximize the expected utility under policy π:

$$E\left[U^{\pi}(s)\right] = E\left[\sum_{t=0}^{\infty} \gamma^{t} R(S_t)\right]$$

MDP Rewards

Suppose we start at the state $(3,1)$. What is the *expected utility* of the policy of "move to the right"?

At time $t = 0$, we have utility of $\gamma^0 R((3,1))$.

1. 80% chance we actually go right and achieve a reward of $+1$, for utility γ^1.
2. 10% chance we actually go up and achieve a reward of -1, for utility $-\gamma^1$.
3. 10% chance we actually go right and achieve a reward of... whatever the discounted *utility* of tile $(2,1)$ is.

MDP Rewards

Suppose we start at the state $(3, 1)$. What is the *expected utility* of the policy of "move to the right"?

At time $t = 0$, we have utility of $\gamma^0 R((3, 1))$.

1. 80% chance we actually go right and achieve a reward of $+1$, for utility γ^1.
2. 10% chance we actually go up and achieve a reward of -1, for utility $-\gamma^1$.
3. 10% chance we actually go right and achieve a reward of... whatever the discounted *utility* of tile $(2, 1)$ is.

MDP Rewards

Suppose we start at the state $(3, 1)$. What is the *expected utility* of the policy of "move to the right"?

At time $t = 0$, we have utility of $\gamma^0 R((3, 1))$.

1. 80% chance we actually go right and achieve a reward of $+1$, for utility γ^1.
2. 10% chance we actually go up and achieve a reward of -1, for utility $-\gamma^1$.
3. 10% chance we actually go right and achieve a reward of... whatever the discounted *utility* of tile $(2, 1)$ is.

So at $t = 1$, $U^\pi((3, 1)) =$
$\gamma\,[.1R(3, 2) + .8R(4, 1) + .1R(3, 1)]$

MDP Rewards: What to consider

Our utility gained after one attempt to move right was

$$\begin{aligned} U^{\pi}((3,1)) &= \gamma\left[.1R(3,2) + .8R(4,1) + .1R(3,1)\right] \\ &= \gamma\left[.1(-1) + .8(1) + .1R(3,1)\right] \end{aligned}$$

If we're allowed to take more moves, the $R(3,1)$ term should get it's own policy: where should we move from $(3,1)$ if it's currently $t=1$?

But the value of $U^{\pi}((3,1))$ at $t=1$ isn't necessarily the same as the left-hand side of $U^{\pi}((3,1))$ at $t=0$! This now depends on our *horizon*. If we only had one more left, it might be utility of zero!

MDP Algorithm:

Our goal with an MDP is to maximize the expected discounted utility after playing the game to its horizon. So we compute the utility associated with any given policy π and choose the best one, the *optimal policy* $\pi^*(s)$:

$$\pi^*(s) = \arg\max_{\pi} U^{\pi}(s)$$

Definition: The *true utility* of a state is its utility when associated with the optimal policy π^*. We denote it $U^{\pi^*}(s)$ or just $U(s)$.

Result: The *true utility* of a state is the expected utility gained by choosing the best successor state. This is the sum of the discounted utilities of all the possible successors of state s under optimal decision a.

Moving Forward

- Coming up:
  1. Computing stuff on Markov Decision Processes!
  2. Markov NB on Friday.