Section overview

Lets go Markov

Markov Process

Definition

A Markov Process is a tuple $(\mathcal{S}, \mathcal{P})$,
$\mathcal{S}$ is a set of states
$\mathcal{P}_{ss'}$ is a state transition probability matrix,

$$\mathcal{P}_{ss'} = P\left[S_{t+1} = s' | S_t = s\right] \tag{2}$$

A Markov process generates a chain of Markov states governed by probabilistic transitions.

Markov Property

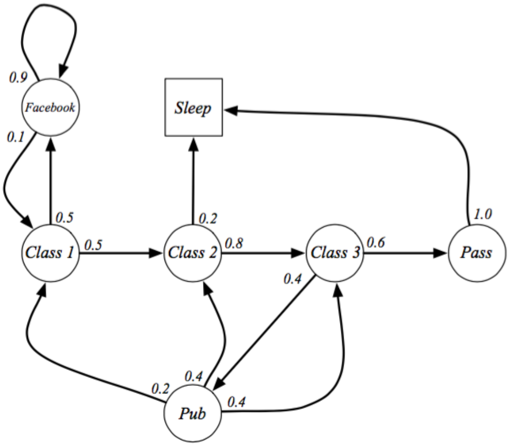
Definition

A state s_t is Markov if and only if $P\left[s_{t+1} \middle| s_t\right] = P\left[s_{t+1} \middle| s_1, \ldots s_t\right]$

The future is independent of the past given the present.

- The present state s_t captures all information in the history of the agent's events.
- Once the state is known, then any data of the history is no longer needed.

Example: University life Markov Process



Round states (transient states), Square states (terminal states).
[2]

[2] Example by David Silver (UCL)

State Transition Probabilities

For a Markov state s and successor state s', the state transition probability is defined by $\mathcal{P}_{ss'} = P\left[s_{t+1} = s' \middle| s_t = s\right]$. The state transition matrix defines transition probabilities from all states s to all successor states s'.
Because all transition probabilities have to be accounted for to give a total probability of 1, we have $\sum_{s'} \mathcal{P}_{ss'} = 1$ (we need to end up somewhere after leaving s, including returning to s). We choose the matrix row notation so that the rows have to sum to one.

Definition (Stationarity)

If the $P\left[s_{t+1} \middle| s_t\right]$ do not depend on t, but only on the origin and destination states, we say the Markov chain is stationary or homogenous.

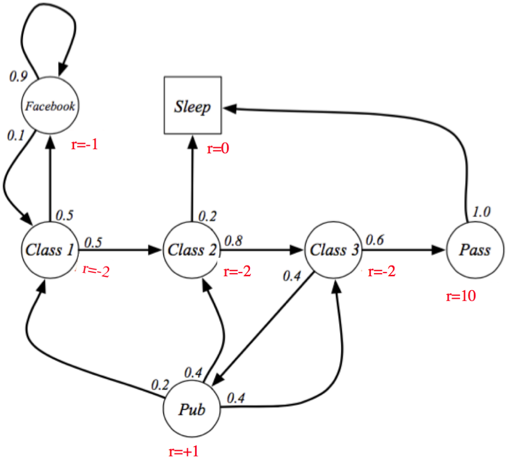
Markov Reward Process

A Markov Reward Process (MRP) is a Markov chain which emits rewards.

Definition (Markov Reward Process)

A Markov Reward Process is a tuple $(\mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma)$
$\mathcal{S}$ is a set of states
$\mathcal{P}_{ss'}$ is a state transition probability matrix
$\mathcal{R}_s = \mathbb{E}\left[r_{t+1} | S_t = s\right]$ is an expected immediate reward that we collect upon departing state s, this reward collection occurs at time step $t + 1$
$\gamma \in [0, 1]$ is a discount factor.

Example: University life Markov Reward Process



What do samples from this process look like?

Return

Definition (Return)

The return R_t is the total discounted reward from time-step t.

$$R_t = r_{t+1} + \gamma r_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{3}$$

The factor $\gamma \in [0, 1]$ is how we discount the present value of future rewards.
The value of receiving reward r after $k+1$ time-steps is $\gamma^k r$.
The discount values immediate reward higher than delayed reward:

- γ close to 0 leads to "myopic" (short-sighted) evaluation.
- γ close to 1 leads to "far-sighted" evaluation.

University life MRP returns

Sample returns for Student MRP:
Starting from $S_1 = C1$ with $\gamma = \frac{1}{2}$

$$R_1 = r_2 + \gamma r_3 + \cdots + \gamma^{T-2} r_T$$

T is for the time it takes to reach the terminal state.

Example (Sample Runs)

C1 C2 C3 Pass Sleep	$R_1 = -2 + \frac{1}{2} \times -2 + \frac{1}{2}^2 \times -2 + \frac{1}{2}^3 \times 10$
C1 FB FB C1 C2 Sleep	$R_1 = -2 + \frac{1}{2} \times -1 + \frac{1}{2}^2 \times -1 + \frac{1}{2}^3 \times -2 + \frac{1}{2}^4 \times -2$
...	
C1 FB FB FB ...	$R_1 = -2 + \frac{1}{2} \times -1 + \frac{1}{2}^2 \times -1 + \frac{1}{2}^3 \times -1 + \dots$

What is the value of being in C1, C2, C3?

Why discounting is a good idea?

Most Markov reward processes are discounted with a $\gamma < 1$. Why?

- Mathematically convenient to discount rewards
- Avoids infinite returns in cyclic or infinite processes
- Uncertainty about the future may not be fully represented
- If the reward is financial, immediate rewards may earn more interest than delayed rewards
- Human and animal decision making shows preference for immediate reward
- It is sometimes useful to adopt undiscounted processes (i.e. $\gamma = 1$), e.g. if all sequences terminate and also when sequences are equally long (why?).

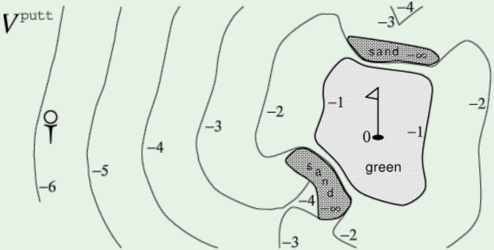
State Value Function

Definition (State value function)

The state value function $v(s)$ of an MRP is the expected return R starting from state s at time t.

$$v(s) = \mathbb{E}\left[R_t | S_t = s\right] \tag{4}$$

Example (Golf)



Bellman Equation for MRPs

$$
\begin{aligned}
v(s) &= \mathbb{E}\left[R_t | S_t = s\right] && (5)\\
&= \mathbb{E}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots | S_t = s\right] && (6)\\
&= \mathbb{E}\left[r_{t+1} + \gamma (r_{t+2} + \gamma r_{t+3} + \ldots) | S_t = s\right] && (7)\\
&= \mathbb{E}\left[r_{t+1} + \gamma R_{t+1} | S_t = s\right] && (8)\\
&= \mathbb{E}\left[r_{t+1} + \gamma v(S_{t+1}) | S_t = s\right] && (9)
\end{aligned}
$$

Value equation decomposes into 2 terms:

- Immediate reward r_{t+1}
- Discounted return of successor state $\gamma v(S_{t+1})$

Note for the mathematically orthodox: Between the two last derivation steps we sweeped under the carpet that we are using the Tower rule/Law of Total expectation to replace the expected return for state with that of the successor state, which is beyond what we expect for this course.

Forms of the Bellman Equation for MRPs

- Expectation notation:

$$v(s) = \mathbb{E}\left[R_t | S_t = s\right]$$

- Sum notation (expectation written out):

$$v(s) = \mathcal{R}_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'} v(s') \tag{10}$$

  We have n of these equations, one for each state.

- Vector notation:

$$\mathbf{v} = \mathcal{R} + \gamma \mathcal{P} \mathbf{v} \tag{11}$$

  The vector $\mathbf{v}$ is n-dimensional.

Direct solution

The Bellman equation is a linear, self-consistent equation:

$$\mathbf{v} = \mathcal{R} + \gamma \mathcal{P} \mathbf{v}$$

we can solve for it directly:

$$\begin{aligned} \mathbf{v} &= \mathcal{R} + \gamma \mathcal{P} \mathbf{v} && (12) \\ (\mathbb{1} - \gamma \mathcal{P}) \mathbf{v} &= \mathcal{R} && (13) \\ \mathbf{v} &= (\mathbb{1} - \gamma \mathcal{P})^{-1} \mathcal{R} && (14) \end{aligned}$$

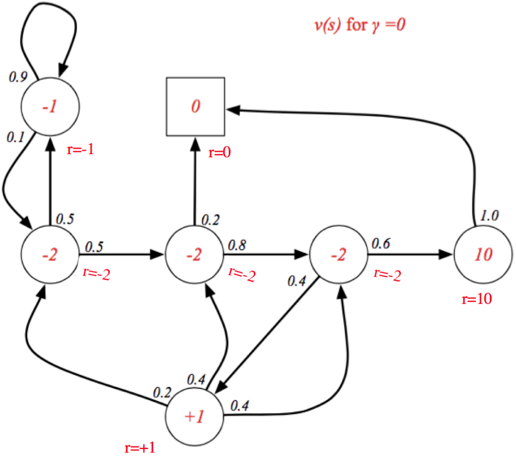
Matrix inversion is computational expensive at $\mathcal{O}(n^3)$ for n states (e.g. Backgammon has 10^{20} states), so direct solution only feasible for small MRPs. Fortunately there are many iterative methods for solving large MRPs:

- Dynamic programming
- Monte-Carlo evaluation
- Temporal-Difference learning

These are at the core of Reinforcement learning, we will learn all 3 algorithms. By the way you have met the solution of self-consistent equations before, whenever you solved for a set of n linear equations in n unknowns you exploited that the equations and the unknowns had to be self-consistent (i.e. related to each other by the common structure of the problem).

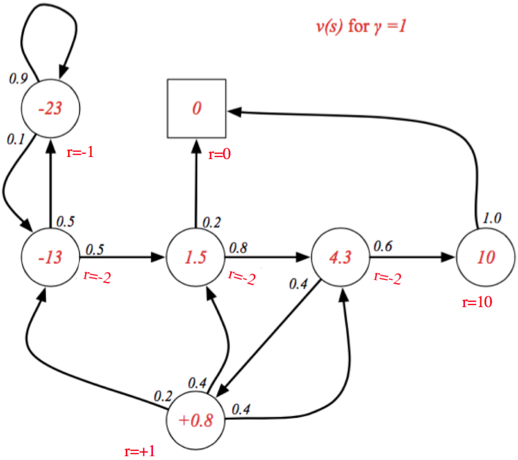
Myopic MRP value function: $\gamma = 0$

What is the value of being in a state?



Immediate reward and total return match, but what if $\gamma > 0$?

Far sighted MRP value function $\gamma = 1$ I

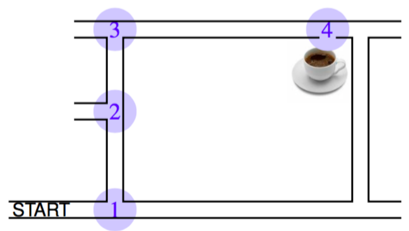


Far sighted MRP value function $\gamma = 1$ II

- Check for self-consistency $v(C3) =?$
  Using Bellman Equation:
  $v(C3) = -2 + 1 \times 0.6 \times 10 + 1 \times 0.4 \times 0.8 = 4.32 \approx 4.3$
  There are cycles in the graph, why is the value of some states not infinite?
  This is self-consistent (numbers in figure are rounded to one digit for space reasons)
- Check for self-consistency $v(C1) =?$
  Using Bellman Equation: $v(C1) =$
  $-2 + 1 \times 0.5 \times -23 + 1 \times 0.5 \times 1.5 = -12.75 \approx -13$
  This is self-consistent (numbers in figure are rounded to one digit for space reasons)

From state to action: policy

Example (Coffee Process)

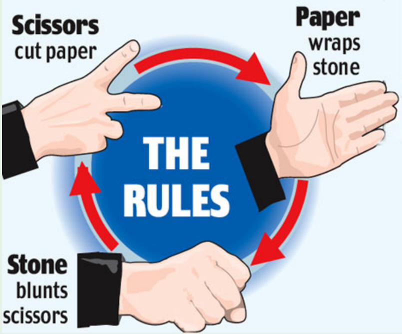


States are $1, 2, 3$ and 4. What to do where?

$$\begin{aligned}
\pi(1) &= \text{turn left and walk} \\
\pi(2) &= \text{go straight and walk} \\
\pi(3) &= \text{turn right and walk} \\
\pi(4) &= \text{turn right and walk}
\end{aligned}$$

Rock, Paper & Scissors Process

Example (Rock, Paper & Scissors Process)



Following a rigid policy can be disadvantageous and exposes the agent to being systematically exploited (see "Dutch Book" argument).

Definition (Policy)

A policy $\pi_t(a, s) = P\left[A_t = a | S_t = s\right]$ is the conditional probability distribution to execute an action $a \in \mathcal{A}$ given that one is in state $s \in \mathcal{S}$ at time t.

The general form of the policy is called a probabilistic or stochastic policy, so π is a probability. If for a given state s only a single a is possible, then the policy is deterministic: $\pi(a, s) = 1$ and $\pi(a', s) = 0, \forall a \neq a'$. A shorthand is to write $\pi_t(s) = a$, implying that the function π returns an action for a given state.
Now we "only" need to work out how to choose an action ...

Lottery decision making

Example

Optimal decision maximises our expected return

Actions	Reward
a_1: play	s_1: Win the lottery
a_2: save	s_2: Lose the lottery

$$a^* = \arg\max_{a_i} \sum_{j=1}^{2} \mathcal{R}_{s_j}^{a_i} P\left[s_j | a_i\right] \tag{15}$$

Were a_i are the actions available in state s_i, i.e. $a_i \ \in \mathcal{A}(s_i)$.

$$\begin{aligned} P\left[s_1|a_1\right] &= 10^{-7} & \mathcal{R}_1^1 &= 500,000 \text{ USD} \\ P\left[s_2|a_1\right] &= 1 - 10^{-7} & \mathcal{R}_1^2 &= -1 \text{ USD} \\ P\left[s_1|a_2\right] &= 0 & \mathcal{R}_2^1 &= 0 \text{ USD} \\ P\left[s_2|a_2\right] &= 1 & \mathcal{R}_2^2 &= 0 \text{ USD} \end{aligned}$$

What is the optimal action for this decision problem?