From ants to safe Artificial Intelligence: Reinforcement Learning Part I: Markov Reward Processes

Leon Lang

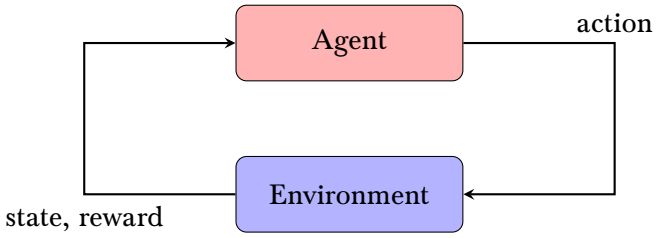
Abstract

This series of three articles aims at providing an overview for Reinforcement Learning, a modern framework in Artificial Intelligence. The first part deals with markov reward processes, which can be imagined as "reinforcement learning without learning". In the second part, we introduce markov decision processes, building on what we learn in part I. Finally, in the third part we will look into AI Safety work that is specifically targeted to making reinforcement learning agents safe.

Contents

1 Introduction

In reinforcement learning (RL), one usually deals with an agent in an environment. The agent finds itself in a specific world state and performs some action. Subsequently, the environment will put the agent in a new state and will reward the agent with some numerical reward. The goal of the agent is to maximize the sum of its future rewards. Interestingly, many problems arising in artificial intelligence, like playing board games, can be formalized this way, which makes it a fruitful framework for thinking about intelligence. The interaction between the environment and the agent is often illustrated as follows:



For those interested in learning more, I can recommend the second edition of the Introduction to Reinforcement Learning by Richard Sutton and Andrew Barto, which is so new that I can unfortunately only cite the first edition [SB98]. For people interested in a more rigorous mathematical treatment – though also with less guidance and intuition – one can recommend [Sze10]. My text probably lies in between in terms of mathematical sophistication. For those who like to watch lectures on YouTube, I can highly recommend David Silver, who is one of the creators of the infamous AlphaGo [Sil15].

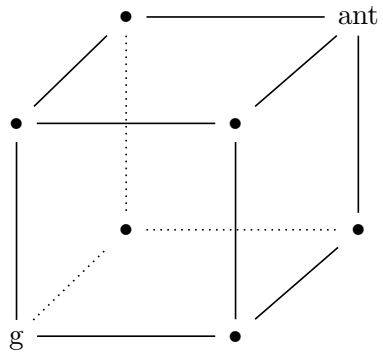
Please note the prerequisites needed to understand my article: You should be acquainted with a basic understanding of linear algebra (linear functions, matrices, linear systems of equations), calculus (sequences, series, limits) and probability theory (probabilities and expectations of discrete random variables) at a first year undergraduate level in mathematics. Section 2 is probably easier to read since it deals with an explicit example, whereas Section 3 deals with the theory underlying this example.

In this first part of the series, we take a step back from RL and view the behaviour of the agent as "fixed", so that the agent does not try to improve its behaviour. This is what I called "reinforcement learning without learning" in the abstract. What remains is called a markov reward process (MRP), and the mathematics dealing with these processes is mainly concerned with computing the future rewards, given the behaviour of the agent. Building on these techniques, we will have an easy time developing the theory of RL in the next article.

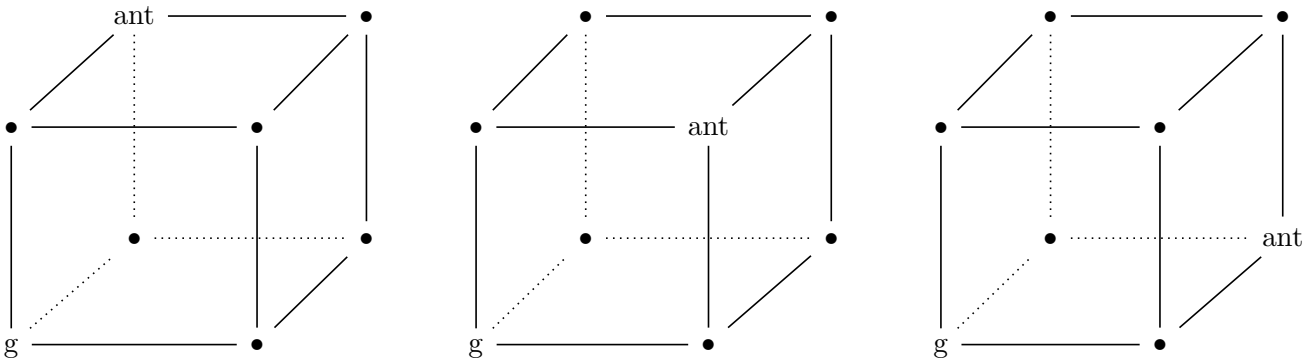
2 An ant on a cube

2.1 Problem formulation

To illustrade MRPs, I invite you to look at the following problem: An ant sits at the top right corner of a cube. In every time step, the ant will, completely randomly, walk to one of the three adjacent corners. At the bottom left corner, diagonally opposite to the ant, there is the goal state the ant is supposed to reach, and once it's there, it will stay there forever, because there is food the ant can eat. The question is: How many time steps will the ant, on average, need to reach this goal state? Here is an illustration for the starting position, where the letter "g" indicates the goal state:



And here is an illustration of the three situations the ant might find itself in after one time step, each with probability $1/3$:



Before delving into the solution, I want to spent a few sentences explaining how this problem relates to what I wrote about reinforcement learning. The ant can be imagined as the agent in this problem. The cube is the environment. The corners are the states. And the reward, most subtly, the ant gets in each time-step is the negative of the time that passed, but only as long as the ant is not already at the goal. Once it is there, the reward will be zero indefinitely, since there will be no harm at a place where the ant has food. Therefore, maximising the reward amounts to minimising the time the ant needs in

order to reach its goal. But since we engage, as I said, only with MRPs in this article – i.e. Reinforcement Learning without learning – the ant does not learn and will forever walk randomly. Therefore there is no need to put everything in the "maximising"-framework and I will just use the passed time itself as the reward, not its negative.

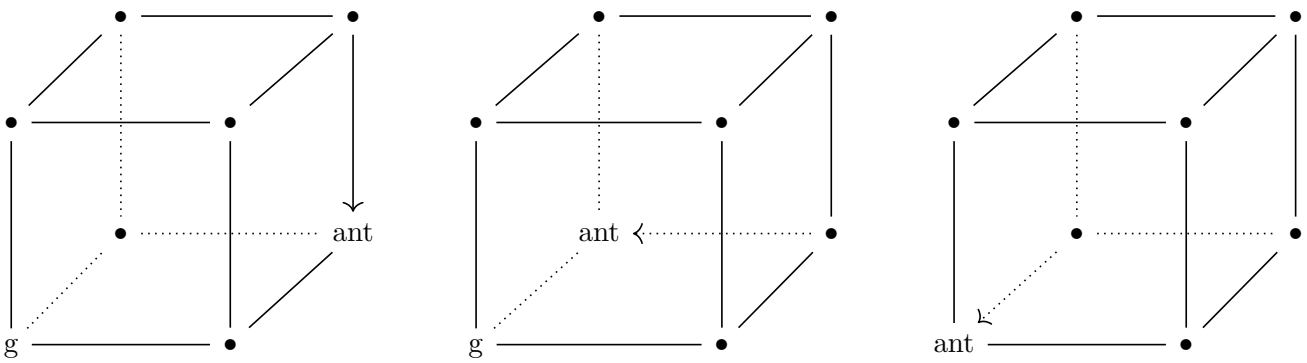
I would advice you to not read on for some time and to try to solve the problem on your own.

2.2 Naive solution

Naively, one might try to solve the question as follows, and with enough pupil-dilating effort this really leads to a success [1]: The question is to find the average number of time steps the ant needs to reach the goal. With "average number", what is really meant is the expected value, so the sum of all the possibilities for the number of time steps, each multiplied with its probability. We call the requested quantity V:

$$V = \sum_{t=1}^{\infty} \Pr(t) \cdot t$$

We quickly see that it is not possible for the ant to reach the goal within only one or two time steps. But for three steps, there are several ways, one indicated in the following series of pictures, where the arrow illustrates which way the ant walked in the preceding time frame:



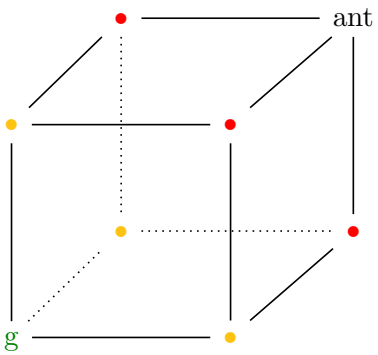
What is the probability of reaching the goal within three time-steps? In the first step, every move will result in a position that is nearer to its goal (look back at the second illustration in this document). After that, every step except the one back to the home state will move the ant further to its goal. Since these are two out of three possible movements, the ant has a $2/3$ probability of doing the right thing. After that, the ant will be in one of the three vertices adjacent to the goal and precisely one out of the three possible movements will move it to the goal. So this probability is $1 \cdot 2/3 \cdot 1/3$. Thus, we get:

$$V = \Pr(3) \cdot 3 + \sum_{t=4}^{\infty} \Pr(t) \cdot t = 2/3 + \sum_{t=4}^{\infty} \Pr(t) \cdot t$$

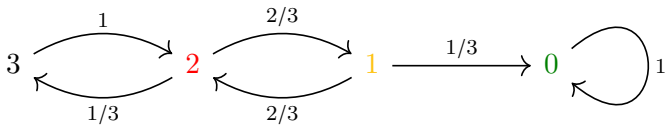
This has brought us a little bit closer to showing what we want to show, but in order to see what the whole series looks like, we have to find more regularities; especially considering the fact that, once the number of time-steps increases, there are more and more fundamentally different ways for the ant to reach its goal.

In order to procede, we first notice that some positions on the cube are in some sense similar to each other, as indicated by having the same color in the following picture:

[1] In his book "Thinking, fast and slow" Daniel Kahneman calls mental work half-jokingly "pupil-dilating" if it is very mentally straining, since it can be shown experimentally that such work will dilate ones pupils [Kah11]. Note that you do not need to dilate your pupils by reading this subsection completely. You will be just as smart going on with subsection 2.3 after reading up to image 1.



The black position – i.e. the position the ant sits on in the beginning – is the unique corner with the property of being three steps away from the goal. The red corners are all two steps away, the yellow corners are one step away, and the green corner is the goal itself – so zero steps away from the goal. Notice that the corners of the same color do not differ from each other in any relevant way. The probabilities from reaching a black or yellow color, for example, are independent from the specific red corner the ant sits at. This leads to a possibility to illustrate the situation in a more abstract way, capturing all essential information. The following picture is an example of a Markov process, as we will define it later - but don't bother about that term for now:



(1)

This picture reads as follows: the number indicates the shortest distance of the ant to the goal state, and their colors match those in the cube picture. The arrows indicate the possibilities for where the ant finds itself in the next time step, and the numbers attached to the arrows are the transition probabilities. For instance, when the ant is one step away from the goal (the yellow 1), then it has a probability of 1/3 to reach the goal in the next state and a probability of 2/3 to move on to state 2. As I said, the goal-state is "absorbing" in the sense that, once the ant is there, it will stay there (i.e. at the food) forever. Absorbing states will be defined and examined in detail in Section 3.

What we immediately see from this is that the ant does not have a chance to reach the goal within an even number of time steps. Since, every time the ant does not move one step towards the goal, it actually has to move one step in the opposite direction, so that it has to do a second step just to be in the same situation as before. Therefore, $\Pr(t) = 0$ whenever t is even.

Now, how can we, in general, compute $\Pr(t)$ when $t \geq 3$ is odd? First of all, we can write the path the ant walks as a sequence of numbers indicating the positions the ant occupies in the different time points. For the trivial path in three steps, this sequence is just $(3, 2, 1, 0)$, which we read as "At time 0, the ant is at position 3, at time 1, the ant is at position 2, and so on". The two possible paths for $t = 5$ are easily seen to be $(3, 2, 1, 2, 1, 0)$ and $(3, 2, 3, 2, 1, 0)$. We can observe several things that hold in general (for every odd $t \geq 3$) and are verified easily:

(i) All sequences start with $(3, 2, \ldots$.

(ii) All sequences end with $\ldots, 2, 1, 0)$.

(iii) At every odd time-point except the very last (where the time-point of the first 3 counts as 0), there is a 2 in the sequence.

(iv) At all other $(t - 3)/2$ places not examined so far, there can be a 3 or a 1.

(v) There are no further restrictions on how the sequences might look like.

Please take a moment to convince yourself that all these rules are in fact true.

The crucial point is the combination of (iii) and (iv): The 3's and 1's mentioned there have to be enclosed by two 2's, and that makes it actually quite easy to compute the probability of such an occurence: The probability to move from a 2 to a 1 is $2/3$, and the probability to move back to a 2 is $2/3$ as well, which means that there is a $4/9$ chance to have a 1 at such a place. In the same way, we see that there is a $1/3 \cdot 1 = 1/3$ chance to have a 3 at such a place. Together, there is a $4/9 + 1/3 = 7/9$ chance that there will be a 1 or 3. Since these are $(t-3)/2$ places and since the end of the sequence, $\ldots, 2, 1, 0)$ has a probability of $2/3 \cdot 1/3 = 2/9$, we see that

$$\Pr(t) = \frac{2}{9} \cdot \left(\frac{7}{9}\right)^{(t-3)/2}.$$

In order to clear our mind, we let q denote the fraction in the brackets: $q \coloneqq \frac{7}{9}$. As a reality check, you might want to show that the computed probabilities do indeed sum to 1, which can be shown using the limit of the geometric series (this limit will reappear in the proof of next lemma, hopefully serving as a reminder):

$$\sum_{t \geq 3 \text{ odd}} \frac{2}{9} \cdot q^{(t-3)/2} = 1.$$

Before we can finish our proof, we need the following lemma:

Lemma 2.1. *Let $|q| < 1$. Then we have*

$$\sum_{t=0}^{\infty} t \cdot q^t = \frac{q}{(1-q)^2}.$$

Proof. We define two functions $f, g : (-1, 1) \to \mathbb{R}$ by

$$f(q) = \sum_{t=0}^{\infty} q^t, \; g(q) = \sum_{t=0}^{\infty} t \cdot q^t.$$

We assume that the reader already knows that the geometric series f can be written as $f(q) = \frac{1}{1-q}$. Now we perform the following trick:

$$\begin{aligned} g(q) &= \sum_{t=0}^{\infty} t \cdot q^t = \sum_{t=1}^{\infty} t \cdot q^t = \sum_{t=1}^{\infty} q^t + \sum_{t=1}^{\infty} (t-1) q^t \\ &= f(q) - 1 + q \cdot g(q) = \frac{q}{1-q} + q \cdot g(q). \end{aligned}$$

Then, solving the equation for $g(q)$ finishes the proof. □

Using this lemma, we can finally answer our question. Remember that we substituted $q = \frac{7}{9}$. In what follows, we also use the notation $f(q)$ and $g(q)$ from the lemma:

$$\begin{aligned} V &= \sum_{t \geq 3 \text{ odd}} \Pr(t) \cdot t = \sum_{t \geq 3 \text{ odd}} \frac{2}{9} \cdot q^{(t-3)/2} \cdot t \\ &= \frac{2}{9} \sum_{t \geq 0 \text{ even}} q^{t/2}(t+3) = \frac{2}{9} \sum_{t \geq 0} q^t (2t+3) \\ &= \frac{2}{9} \left(2 \cdot g(q) + 3 \cdot f(q)\right) = \frac{2}{9} \left(2 \cdot \frac{q}{(1-q)^2} + 3 \cdot \frac{1}{1-q}\right) \\ &= 10, \end{aligned}$$

where in the last step, we just used that $q = \frac{7}{9}$ and did the computation.

So, we finally have the answer: On average, the ant will require ten steps to reach its goal. What a complicated way to write the number 10.

2.3 Smart way of solving the problem

The preceding solution was possibly the most elegant solution following the initial idea, but we should not have pursued it in the first place; there is a better, way more illuminating and generalizable approach. The solution I present now involves the so-called Bellmann equation and will also lead to interesting theory that is essential for understanding reinforcement learning. So, our efforts will definitely pay out this time.

Basically, there is only one observation necessary, namely the following: Imagine we are not only interested in finding out how many time steps the ant will need on average for reaching the goal when it is in state 3, but we are also interested in this number when the ant starts in 2 or 1. The crucial observation is that the expected value of the number of time steps the ant has yet to walk does not depend on how many steps the ant already walked; It only depends on the current position of the ant. This is due to the fact that the transition probabilities do not depend on the elapsed time. So, imagine the ant sits on one of the corners labeled 2. Then after one time step the ant will be in a corner labeled 3 with probability $1/3$ and will need as many time steps as an ant on average takes being in corner 3. And the ant will be in a corner labeled 1 with probability $2/3$ and need as many time steps as an ant will need on average when it starts in position 1 (Believe me, these almost tautological observations are useful!). In formulas, we can express this as follows: Let $V(3)$, $V(2)$ and $V(1)$ be the average number of time steps the ant needs to reach its goal when it begins in a corner labeled 3, 2 and 1 respectively. Then the equations relating these numbers are:

$$\begin{aligned} V(3) &= 1 + V(2), \\ V(2) &= 1 + 1/3 \cdot V(3) + 2/3 \cdot V(1), \\ V(1) &= 1 + 2/3 \cdot V(2) + 1/3 \cdot 0. \end{aligned}$$

If we set $V = (V(3), V(2), V(1))$ and $r = (1, 1, 1)$ and define the matrix of transition probabilities P by

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 1/3 & 0 & 2/3 \\ 0 & 2/3 & 0 \end{pmatrix},$$

then this set of equations can be written as just one equation, reading

$$V = r + P \cdot V.$$

This is the famous Bellmann equation of our problem, and the general properties of this equation will be examined in Section 3. By matrix invertion, writing Id for the identity matrix, we obtain:

$$\begin{aligned} V = (\mathrm{Id} - P)^{-1} \cdot r &= \begin{pmatrix} 1 & -1 & 0 \\ -1/3 & 1 & -2/3 \\ 0 & -2/3 & 1 \end{pmatrix}^{-1} \cdot r \\ &= \begin{pmatrix} 5/2 & 9/2 & 3 \\ 3/2 & 9/2 & 3 \\ 1 & 3 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 \\ 9 \\ 7 \end{pmatrix}. \end{aligned}$$

What a fast and reassuring success! Not only did we find a solution in less than half the space compared to Section 2.2, we also managed to answer a more general question: We not only know that the ant will, on average, need 10 steps for reaching the goal starting in position 3, but we also know that it will need on average 9 steps starting in 2 and 7 steps starting in 1. What is less reassuring is that we needed to invert a matrix in the process of doing so. If you anticipate what useful RL looks like, you can imagine that the problems arising there are much larger in size, and consequently the vector V will have way more entries, as will the matrix P. Inverting the matrix $\mathrm{Id} - P$ will then be infeasible algorithmically and one needs a smarter way.

Therefore, we now desribe an efficient algorithm for finding the solution of the Bellmann equation $V = r + P \cdot V$. We can read this equation as follows: Even if we forget for a moment what the expected

value $V(i)$ is, we know it consists of the "reward" $r(i)$ for leaving i and then getting the expected values of whereever we land – weighted with the appropriate transition probabilities. But instead of stopping there, we could look further and procede from the new states in the same way and replace their expected values by the rewards plus the expected values of the states that follow. The intuition now is that, the further we look, the less it matters for the original value $V(i)$ what the expected values of the end-states are. This is since, the longer we observe the ant moving, the higher the probability that it already reached its goal, and so we care less and less about the values in the end of this process – they are zero anyway. Thus, the rewards in the process already capture everything about the expected value we began with.

If this was to hand-wavy for you, then you will soon see how this can be made precise. For that, we create a new symbol T, which we call the Bellmann operator. We set $T : \mathbb{R}^3 \to \mathbb{R}^3$, $x \mapsto r + P \cdot x$. Then, for the true vector of expected values, $V = (V(3), V(2), V(1)) \in \mathbb{R}^3$, what it means to look further is to perform the operation $T(V) = r + P \cdot V$, thereby replacing the simple expression for V by a complicated expansion (remember that $T(V) = V$). We do this recursively with the inner V again and get $T(T(V)) = r + P \cdot T(V) = r + P \cdot r + P^2 V$. We can do this arbitrarily often and see that

$$\begin{aligned} V = T(V) = \cdots = T^k(V) &= r + P \cdot r + \cdots + P^{k-1} r + P^k \cdot V \\ &= \left(\sum_{j=0}^{k-1} P^j \right) \cdot r + P^k \cdot V. \end{aligned}$$

Now you see more precisely why the values on the right side of this expression matter less and less: The $i - j$ entry of P^k is the probability to reach state j from state i within k time-steps (this will be proven in Lemma 3.11), and this probability will get really small when k increases, since there will be more and more chances to reach the goal. Therefore, the term $P^k \cdot V$ will move closer and closer to zero, and since this is the only part on the right hand side that depends on V, we expect that we can plug in pretty much anything into T^n and obtain something close to the true vector of expected values V.

More precisely, the following holds: For every vector $x \in \mathbb{R}^3$, the sequence $x, T(x), T^2(x), \ldots$ will converge to the vector V. Doing a composition of affine-linear mappings is actually pretty easy for computers to perform, so this gives rise to a decent algorithm for finding V without inverting a matrix. In order to increase memory efficiency, the composition should be evaluated from the right to the left, contrary to what we did for explaining the algorithm. The code in Python then looks as follows:

```
from __future__ import division
from numpy import *

def update(R, P, V, n):
    if (n == 0):
        return V
    else:
        return update(R, P, add(R, matmul(P, V)), n-1)

print(update([[1], [1], [1]],
             [[0, 1, 0],[1/3, 0, 2/3],[0, 2/3, 0]],
             [[-45],[23],[13]],
             50))
```

The result of this operation is the vector $[[9.96557453], [9.02615269], [6.97704968]]$, so relatively near to the true solution $[[10], [9], [7]]$, considering how far away the input vector $[[-45], [23], [13]]$ is.

Finally, we remark one more interesting fact that we will prove in Lemma 3.12. In the beginning of this subsection, we found the vector V by performing the operation $(\mathrm{Id} - P)^{-1} \cdot r$. But now we have an algorithm converging to the same vector, and looking closely we see that V can thereby also be written as $V = \left(\sum_{k=0}^{\infty} P^k\right) \cdot r$. The obvious question then is if $\sum_{k=0}^{\infty} P^k$ is maybe the inverse of $\mathrm{Id} - P$. Similarities to the usual geometric series suggests so, and in the aforementioned lemma we will indeed prove that this is true.

3 Markov Reward processes and the Bellmann Equation

In this section, we describe the general theory underlying the example of the ant in order to capture its true essence. We will define so-called absorbing Markov Reward Processes, which are random processes with a finite number of states and certain properties. The most important property is that they are absorbing: they have a state such that, with probability 1, the process will end up in that state evantually. This is like the ant that reached the goal-state in the end. Also, in every state-transition, there is a reward that the "agent" receives (although we will not talk about agents here: Since there is no learning, there is no need to talk about agency), and this reward matches the elapsed time in the ant-example. Even in this generality, it will be possible to prove that for every state there are well-defined expected values for the sum of future rewards. This quantity will match the expected time the ant needed to reach its goal.

This section is more mathematical in nature and will serve as a basis for defining Reinforcement Learning in the next article. I hope that the long example was motivating enough for you to read on and enjoy the full theory.

3.1 Basic definitions

Definition 3.1 (Markov Reward Process). Let (S, R, P) be a tuple where

(i) $S = \{s_1, \dots, s_{n+1}\}$ is a set of states,

(ii) $R = \{r_1, \dots, r_{n+1}\} \subseteq \mathbb{R}$ is a set of rewards,

(iii) $P \in \mathbb{R}^{(n+1)\times(n+1)}$ is a matrix of transition probabilities, where $p_{ij} \in [0, 1]$ and $\sum_{j=1}^{n+1} p_{ij} = 1$ for all $i \in \{1, \dots, n+1\}$.

Then a *Markov Reward Process* (MRP) on (S, R, P) is a sequence of random variables $X_0, X_1, \dots$ with values in S such that the transition probabilities are given by P and satisfy the *Markov Condition*, i.e. for all $i, j \in \{1, \dots, n+1\}$, for all $k \geq 1$ and for all choices of indices i_l for $l = \{1, \dots, k-1\}$ and $i_l \in \{1, \dots, n+1\}$ we have

$$p_{ij} = \Pr(X_{k+1} = s_j \mid X_k = s_i) = \Pr(X_{k+1} = s_j \mid X_k = s_i, X_{k-1} = s_{i_{k-1}}, \dots, X_0 = s_{i_0}),$$

together with a sequence of received rewards $R_0, R_1, \dots$, where R_k is the reward received for leaving state X_k: $R_k = r_i$ whenever $X_k = s_i$.

For simplicity, we also call the tuple (S, R, P) itself a Markov Reward Process.

Definition 3.2 (Absorbing Markov Reward Process). Let (S, R, P) be an MRP. It is called *absorbing* if the state s_{n+1} is an absorbing state, i.e. a state with the following properties:

(i) For all $i \in \{1, \dots, n\}$ there is a sequence of indices $k_1, k_2, \dots, k_s$ in $\{1, \dots, n+1\}$ with $i = k_1$ and $n+1 = k_s$ such that $\prod_{l=1}^{s-1} p_{k_l, k_{l+1}} > 0$, i.e. s_{n+1} is reachable from s_i.

(ii) For all $j \in \{1, \dots, n\}$ we have $p_{n+1,j} = 0$ and $p_{n+1,n+1} = 1$, i.e. the absorbing state s_{n+1} cannot be left.

(iii) $r_{n+1} = 0$.

Effectively, what this means is that the Markov process $X_0, X_1, \dots$ will eventually reach the absorbing state s_{n+1} with probability 1 and that the reward will, from there on, always be zero. The absorbing state can therefore be imagined as something like an "endpoint" of the process, where nothing interesting will emerge from. We also mention that, obviously, the absorbing state does not have to be s_{n+1}; we only assume this for simplicity.

Definition 3.3 (State Values). Let (S, R, P) be an MRP. Then the *value of state* $s_i \in S$, $V(s_i) = V(i)$, is given by

$$V(i) = \mathrm{E}\left[\sum_{k=0}^{\infty} R_k \mid X_0 = s_i\right].$$

That is, $V(i)$ is the expected sum of future rewards for an MRP starting in s_i.

This expression is, as it is stated in its generality, purely formal, since this expected value does not always exist. The main goal of this section is to show that in the case of an absorbing MRP these expected values are actually well-defined numbers. First we will show that the expected values fulfil the so-called Bellmann-equations:

Lemma 3.4 (Bellmann Equation). *Let (S, R, P) be an absorbing MRP. Then we have $V(n+1) = 0$ and for all $i \in \{1, \dots, n\}$ we have*

$$V(i) = r_i + \sum_{j=1}^{n} p_{ij} V(j).$$

If we write $V, r \in \mathbb{R}^n$ for the vectors of state values and rewards (omitting the trivial cases $V(n+1) = 0$ and $r_{n+1} = 0$), and writing by abuse of notation also P for the $n \times n$ upper left corner of the full transition matrix, this can be rewritten as

$$V = r + PV.$$

This equation is called the Bellmann Equation *for V.*

Proof. $V(n+1) = 0$ is clear, since the process is captured in state s_{n+1} and always receives reward 0 from there on. We furthermore have

$$\begin{aligned}
V(i) &= \mathrm{E}\left[\sum_{k=0}^{\infty} R_k \mid X_0 = s_i\right] \\
&= r_i + \mathrm{E}\left[\sum_{k=1}^{\infty} R_k \mid X_0 = s_i\right] \\
&= r_i + \sum_{j=1}^{n+1} \Pr(X_1 = s_j \mid X_0 = s_i)\, \mathrm{E}\left[\sum_{k=1}^{\infty} R_k \mid X_0 = s_i, X_1 = s_j\right] \\
&= r_i + \sum_{j=1}^{n+1} p_{ij} E\left[\sum_{k=0}^{\infty} R_k \mid X_0 = s_j\right] \\
&= r_i + \sum_{i=1}^{n+1} p_{ij} V(j) \\
&= r_i + \sum_{i=1}^{n} p_{ij} V(j),
\end{aligned}$$

where we used in the third to last step that the process is markov, i.e. that the expected value does not depend on X_0 anymore as soon as we know X_1. In the last step we used that $V_{n+1} = 0$. This proves the claim. □

Therefore, in order to find the state values V we have to find solutions to the Bellmann equation $V = r + PV$, which is equivalent to the equation $(\mathrm{Id} - P)V = r$. Therefore, it suffices to show that the $n \times n$-matrix $(\mathrm{Id} - P)$ is invertible and writing down the inverse in order to find the state values. We will indeed do this, and by giving a precise structure of the inverse we also get an efficient algorithm, using the Bellmann update already present in the ant example, for computing the solution V.

3.2 Computing the state values in an absorbing MRP

From now on, (S, R, P) is always assumed to be absorbing.

Definition 3.5 (Bellmann operator). The *Bellmann operator* $T : \mathbb{R}^n \to \mathbb{R}^n$ is given by $Tx := r + Px$.

Theorem 3.6. *Let (S, R, P) be an absorbing MRP. Let $x \in \mathbb{R}^n$ be arbitrary. Then the sequence $x, Tx, T^2x, T^3x, \dots$ converges to the unique solution of the Bellmann equation $V = r + PV$.*

We remark here that it is possible to prove this theorem using the Banach fixed point Theorem, applied to the operator $S := T^n$. If you know what this theorem is about, I advice you to try to work out the details. If you need a hint, then take a look at the appendix of [Sze10] where a slightly easier statement is proven with help of the Banach fixed point Theorem.

Nevertheless, I decided to present an elementary proof here in order to keep the prerequisites modest. We first need some lemmas:

Lemma 3.7. *For all $i \in \{1, \dots, n\}$ we have* $\Pr(X_n = s_{n+1} \mid X_0 = s_i) > 0$.

Proof. This follows from property (i) in the definition of an absorbing MRP. You just have to observe that, if there is a path from i to $n+1$, then you also find a path of length $\leq n$ by cutting out all subpaths which start and end in the same vertex. □

Let for the rest of this document $\gamma := 1 - \min_i \{\Pr(X_n = s_{n+1} \mid X_0 = s_i)\} < 1$. Then your chances of still being "in the game" after n time steps are $\leq \gamma$:

Lemma 3.8. *For all $i \in \{1, \dots, n\}$ we have*

$$\Pr(X_n \in \{s_1, \dots, s_n\} \mid X_0 = s_i) \leq \gamma$$

Proof. We have

$$\begin{aligned}\Pr(X_n \in \{s_1, \dots, s_n\} \mid X_0 = s_i) &= 1 - \Pr(X_n = s_{n+1} \mid X_0 = s_i) \\ &\leq 1 - \min_{i'} \{\Pr(X_n = s_{n+1} \mid X_0 = s_{i'})\} \\ &= \gamma.\end{aligned}$$

□

We can generalize the inequality of the last lemma as follows. The idea is that the absorbing state keeps on absorbing indefinitely, and that after each n time steps, at least the proportion $1 - \gamma$ of "what's left" gets swallowed:

Lemma 3.9. *For all $k \in \mathbb{N}$ and $i \in \{1, \dots, n\}$ we get*

$$\Pr(X_{k \cdot n} \in \{s_1, \dots, s_n\} \mid X_0 = s_i) \leq \gamma^k.$$

Proof. First we note a property of conditional probability that's useful here. So, whenever X, Y and Z are discrete random variables with positive probabilities for every instance, we have:

$$\Pr(Z = z \mid X = x) = \sum_y \Pr(Z = z \mid Y = y, X = x) \cdot \Pr(Y = y \mid X = x).$$

This can be proven using the definition $\Pr(Z = z \mid Y = y, X = x) = \frac{\Pr(Z=z, Y=y, X=x)}{\Pr(Y=y, X=x)}$. Together with the Markov condition, this explains the first step in the following proof:

We use the last lemma as induction start and now do the induction step $k - 1 \to k$:

$$\begin{aligned}&\Pr\left(X_{k \cdot n} \in \{s_1, \dots, s_n\} \mid X_0 = s_i\right) \\ &= \sum_{j=1}^{n} \Pr(X_{k \cdot n} \in \{s_1, \dots, s_n\} \mid X_{(k-1) \cdot n} = s_j) \cdot \Pr(X_{(k-1) \cdot n} = s_j \mid X_0 = s_i) \\ &= \sum_{j=1}^{n} \Pr(X_n \in \{s_1, \dots, s_n\} \mid X_0 = s_j) \cdot \Pr(X_{(k-1) \cdot n} = s_j \mid X_0 = s_i) \\ &\leq \gamma \cdot \sum_{j=1}^{n} \Pr(X_{(k-1) \cdot n} = s_j \mid X_0 = s_i) \\ &= \gamma \cdot \Pr(X_{(k-1) \cdot n} \in \{s_1, \dots, s_n\} \mid X_0 = s_i) \\ &\leq \gamma \cdot \gamma^{k-1} \\ &= \gamma^k.\end{aligned}$$

□

Lemma 3.10. *Let more generally $k' = k \cdot n + l \in \mathbb{N}$ be an arbitrary natural number, with $k \in \mathbb{N}$ and $l \in \{0, \dots, n-1\}$. Let again $i \in \{1, \dots, n\}$. Then we have*

$$\Pr(X_{k'} \in \{s_1, \dots, s_n\} \mid X_0 = s_i) \leq \gamma^k.$$

Proof. This follows directly using the last lemma and using the fact that, since s_{n+1} is absorbing, if $X_{k'} \in \{s_1, \dots, s_n\}$, then $X_{k \cdot n} \in \{s_1, \dots, s_n\}$ as well:

$$\begin{aligned}\Pr(X_{k'} \in \{s_1, \dots, s_n\} \mid X_0 = s_i) &\leq \Pr(X_{k \cdot n} \in \{s_1, \dots, s_n\} \mid X_0 = s_i) \\ &\leq \gamma^k.\end{aligned}$$

□

Lemma 3.11. *For an arbitrary $k' \in \mathbb{N}$ let $Q_{k'} = P^{k'} \in \mathbb{R}^{n \times n}$. Then*

$$(Q_{k'})_{ij} = \Pr(X_{k'} = s_j \mid X_0 = s_i).$$

Proof. For $k' = 0$ and $k' = 1$ the statement is clear. Now we do the induction step $k - 1 \to k$. We have

$$\begin{aligned}(Q_{k'})_{ij} = (Q_{k'-1} \cdot P)_{ij} &= \sum_{l=1}^{n} (Q_{k'-1})_{il} \, p_{lj} \\ &= \sum_{l=1}^{n} \Pr(X_{k'-1} = s_l \mid X_0 = s_i) \cdot \Pr(X_{k'} = s_j \mid X_{k'-1} = s_l) \\ &= \Pr(X_{k'} = s_j \mid X_0 = s_i),\end{aligned}$$

which finishes the proof. □

Lemma 3.12. *The series $\sum_{k'=0}^{\infty} P^{k'}$ converges. The limit is the inverse of the matrix $\mathrm{Id} - P$.*

Proof. We just need to show that the series converges entry-wise. Since all summands are non-negative, it suffices to show boundedness. This is done as follows, using the last two lemmas and the fact $\gamma < 1$:

$$\begin{aligned}\left(\sum_{k'=0}^{\infty} P^{k'}\right)_{ij} &= \sum_{k'=0}^{\infty} \left(P^{k'}\right)_{ij} \\ &= \sum_{k'=0}^{\infty} \Pr(X_{k'} = s_j \mid X_0 = s_i) \\ &= \sum_{k=0}^{\infty} \sum_{l=0}^{n-1} \Pr(X_{k \cdot n + l} = s_j \mid X_0 = s_i) \\ &\leq \sum_{k=0}^{\infty} n \cdot \gamma^k \\ &= \frac{n}{1 - \gamma} \\ &< \infty.\end{aligned}$$

Then, using the usual index shift arguments, we see that $\sum_{k'=0}^{\infty} P^{k'}$ is indeed the inverse of $\mathrm{Id} - P$:

$$\begin{aligned}(\mathrm{Id} - P) \cdot \sum_{k'=0}^{\infty} P^{k'} &= \sum_{k'=0}^{\infty} P^{k'} - \sum_{k'=0}^{\infty} P^{k'+1} \\ &= \sum_{k'=0}^{\infty} P^{k'} - \sum_{k'=1}^{\infty} P^{k'} \\ &= \mathrm{Id},\end{aligned}$$

which finishes the proof. □

Now we finally have everything we need in order to proof Theorem 3.6:

Proof of Theorem 3.6. We have $T^m x = \left(\sum_{k'=0}^{m-1} P^{k'}\right) r + P^m x$, which converges by the last lemma to $\left(\sum_{k'=0}^{\infty} P^{k'}\right) r + 0 = (\mathrm{Id} - P)^{-1} r$, which is by definition the unique solution to the Bellmann equation. □

3.3 Outlook

In the next article, we look at modern reinforcement learning problems, where the agent tries its best to change the state transition probabilities in such a way that the sum of the future rewards gets maximized. The absorbing MRPs arising there will be of a special nature: The probability to reach the absorbing state will in every time step be some fixed number $0 < 1 - \gamma \leq 1$, where γ is the so-called "discount factor", which is completely independent from the current state. That is, for all $i \in \{1, \ldots, n\}$, one has

$$\Pr(X_{n+1} = s_{n+1} \mid X_n = s_i) = 1 - \gamma.$$

By doing the substitution $P \mapsto \frac{1}{\gamma} P$, the Bellmann equation then reads

$$V = r + \gamma P V.$$

This new transition matrix P has the desirable property that all probabilities in one row sum up to 1, so that this can be interpreted as the full transition probability matrix of the MRP where the absorbing state is "deleted". The goal of the agent then is not to maximize the sum of the future rewards, but the *discounted* sum of the future rewards, i.e. the expected values V satisfying the aforementioned equation are then actually defined as

$$V = \mathrm{E}\left[\sum_{k=0}^{\infty} \gamma^k R_k \mid X_0 = s_i\right]$$

in order to account for the fact that there is no absorbing state anymore. This will give a new perspective to the theory developed in this article.

References

[Kah11] Daniel Kahneman, *Thinking, fast and slow*, Farrar, Straus and Giroux, New York, 2011.

[SB98] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, IEEE Transactions on Neural Networks **16** (1998), 285–286.

[Sil15] David Silver, *RL course by David Silver - Lecture I: Introduction to Reinforcement Learning*, https://www.youtube.com/watch?v=2pWv7GOvuf0/, 2015.

[Sze10] Csaba Szepesvari, *Algorithms for Reinforcement Learning*, Morgan and Claypool Publishers, 2010.