Reinforcement Learning and Autonomous Systems (CS4122)

Lecture 11 (04/09/2024)
Lecture 12 (09/09/2024)
Lecture 13 (10/09/2024)

Instructor: Gourav Saha

1 Introduction

In lecture 10 we learned that a Markov decision process (MDP) is defined by:

1. State and state space.
2. Action and action space.
3. Reward.
4. State transition probability.
5. Reward probability.

We now discuss that given a problem, how to formulate it as an MDP. Formulating a problem as an MDP essentially means to find the above five components for that particular problem.

Formulating a problem as an MDP is perhaps one of the most important steps from application point of view because while there might be existing solvers online to find the optimal policy for an MDP, the process of parsing a problem and formulating it as an MDP is still something we have to do ourselves.

We already discussed a simpler problem called "fishing in gridworld" in lecture 10 that shows how to formulate a problem as an MDP. Please refer to that first for a gentler introduction to this topic. In what follows, we discuss how to formulate the following problems as an MDP:

1. Automatic scheduler for WiFi and mobile data.
2. Investing in Stock Market.

If you want more examples of modeling real-world problems as MDP, here are two "old-school" papers that documents the application of MDP (many of these applications are non-intuitive):

it.uu.se/edu/course/homepage/aism/st11/MDPApplications3.pdf

it.uu.se/edu/course/homepage/aism/st11/MDPApplications1.pdf

2 Automatic Scheduler for WiFi and Mobile Data

2.1 Problem Statement

Many a times we have to switch between WiFi and mobile data based on availability. This problem is motivated by this broad idea. Whenever we are using WiFi or mobile data, we are selecting a wireless channel to transmit on. Let's say that WiFi and mobile data are associated with channels 1 and 2 respectively.

Let $i \in \{1, 2\}$ denote the channel index. Channel i is either available for not available in a time slot. Let $s_{i,t} \in \{0, 1\}$ denote whether channel i is available ($s_{i,t} = 1$) or not available ($s_{i,t} = 0$) in time slot t. If channel i is not available in time slot t, then the probability that it is not available in next time slot $t + 1$ is α_i. If channel i is available in time slot t, then the probability that it is available in next time slot $t + 1$ is θ_i. If a channel is not available, the number of data packets that we can send through it is obviously zero. If channel i is available, we can set

$d \in \{0, 1, 2, \ldots, D\}$ packets through it where the probability of sending d packets is $p_{i,d}$.

If we switch to channel i, we can't start to use it even if it is available. We have to wait for τ_i time slots for the device to establish connection with the channel. For this τ_i time slots, the number of data packets that we can send is zero. The objective is to maximize the total number of data packets transmitted over an infinite horizon. Formulate this problem as a Markov decision process.

2.2 MDP Formulation

2.2.1 State and State Space

The states of the system are:

1. $s_{1,t}$: It describes is a channel 1 is available $(s_{1,t} = 1)$ or not available $(s_{1,t} = 0)$ in time slot t.
2. $s_{2,t}$: It describes is a channel 2 is available $(s_{2,t} = 1)$ or not available $(s_{2,t} = 0)$ in time slot t.
3. z_t: The channel which the scheduler is using in time slot t.
4. c_t: It is a counter used to capture the time after which a channel can be used after switching to it. We are using a count down timer. c_t will be set to either $\tau_1 - 1$ or $\tau_2 - 1$ depending on whether the channel was switched to channel 1 or 2 respectively. From there c_t decrease to 0. When c_t is 0, the channel can be used for data transmission.

Finally, the state of the system is,

$$x_t = (s_{1,t}\ ,\ s_{2,t}\ ,\ z_t\ ,\ c_t)$$

Now, we have to find the state space of x_t. Now:

1. $s_{i,t} \in \{0, 1\}\,,\ \forall i \in \{0, 1\}\,,\ \forall t.$
2. $z_t \in \{1, 2\}\,,\ \forall t.$
3. $c_t \in \{0, 1, \ldots, \max(\tau_1, \tau_2) - 1\}\,,\ \forall t.$ $\max(\tau_1, \tau_2) - 1$ because the maximum value of c_t is either $\tau_1 - 1$ or $\tau_2 - 1$.

Finally, $x_t \in \mathcal{S}$, $\forall t$ where,

$$\mathcal{S} = \{0, 1\} \times \{0, 1\} \times \{1, 2\} \times \{0, 1, \ldots, \max(\tau_1, \tau_2) - 1\}$$

is the state space of x_t.

2.2.2 Action and Action Space

The action a_t in time slot t is whether to switch channel $(a_t = 1)$ or to not switch channel $(a_t = 0)$. So the action space of the system is $\mathcal{A} = \{0, 1\}$.

NOTE: For this problem, the action space is not a function of the state. But, in general, it is.

2.2.3 Reward and Reward Probability

The reward r_t at time slot t is the number of packets of data transmitted in that time slot,

$$r_t = \begin{cases} 0 & ; s_{z_t,t} = 0 \\ 0 & ; s_{z_t,t} = 1, c_t > 0 \\ d & ; \text{w.p. } \ p_{z_t,d} \ \text{ if } \ s_{z_t,t} = 1, c_t = 0 \end{cases} \tag{1}$$

In (1), $r_t = 0$ when $s_{z_t,t} = 0$ because $s_{z_t,t} = 0$ implies that current channel z_t is not available for data transfer. Also, $r_t = 0$ when $s_{z_t,t} = 1$ and $c_t > 0$ because there is a wait time after switching before which we can't use the channel. But when $s_{z_t,t} = 1$ and $c_t = 0$, we can send data packets. The probability of sending d packets in current channel z_t is $p_{z_t,d}$ *(note that w.p. in (1) means "with probability").*

NOTE: During lecture I wrote,

$$r_t = \begin{cases} 0 & ; s_{z_t,t} = 0 \\ 0 & ; s_{z_t,t} = 1, c_t > 0 \\ \sum_{d=0}^{D} d \cdot p_{z_t,d} & ; s_{z_t,t} = 1, c_t = 0 \end{cases} \tag{2}$$

which is INCORRECT! This is because r_t is a random variable and not expected value of a random variable.

Finding the states of a system comes with practice. I am not aware of any other way. For example, wihout practice you will be able to guess that $s_{1,t}$ and $s_{2,t}$ is likely to be the states of this system. But, it will not be intuitive that z_t and c_t are also the states of the system. During lecture 11, I tried to do the problem wihout using z_t and c_t as states to demonstrate the need for z_t and c_t as states. In short, for MDP, we have to express the reward in a time slot as a function of actions and states only. Without z_t and c_t we can't express reward of this problem as a function of actions and states only. This is because:

1. z_t determines the channel we are currently on. z_t in turn determines channel availability through the variables$s_{z_t,t}$. Also, the probability of sending d packets depends on which channel we are currently on.

2. c_t determines whether we can send data packets or not even if the channel we are currently on is available.

2.2.4 State Transition Probability

The state transition of $s_{1,t}$ and $s_{2,t}$ is governed by,

$$s_{i,t+1} = \begin{cases} 0 & ; \text{w.p. } \alpha_i \text{ if } s_{i,t} = 0 \\ 1 & ; \text{w.p. } 1-\alpha_i \text{ if } s_{i,t} = 0 \\ 1 & ; \text{w.p. } \theta_i \text{ if } s_{i,t} = 1 \\ 0 & ; \text{w.p. } 1-\theta_i \text{ if } s_{i,t} = 1 \end{cases} \tag{3}$$

where $i \in \{1, 2\}$ in equation (3). The first two cases describes the state transition when in the current time slot t, the channel is not available. The next two cases describes the state transition when in the current time slot t, the channel is available.

The state transition of z_t is governed by,

$$z_{t+1} = \begin{cases} z_t & ; a_t = 0 \\ 2 & ; a_t = 1, z_t = 1 \\ 1 & ; a_t = 1, z_t = 2 \end{cases} \tag{4}$$

In (4), when the action is to not switch ($a_t = 0$), $z_{t+1} = z_t$ because we will be using the same channel on the next time slot. When the action is to switch channel ($a_t = 1$), then z_{t+1} is 2 or 1 depending on whether the current channel we are using is 1 or 2 respectively.

The state transition of c_t is governed by,

$$c_{t+1} = \begin{cases} \tau_2 - 1 & ; a_t = 1, z_t = 1 \\ \tau_1 - 1 & ; a_t = 1, z_t = 2 \\ c_t - 1 & ; a_t = 0, c_t > 0 \\ 0 & ; a_t = 0, c_t = 0 \end{cases} \tag{5}$$

In (5), the first two cases set the countdown timer state to $\tau_2 - 1$ or $\tau_1 - 1$ depending on whether channel switched from channel 1 to 2 or 2 to 1 respectively. The last two cases of (5) is to keep decreasing c_t to 0 and hold it at 0 until the next switch of channel.

3 Investing in Stock Market

We discussed this topic during lecture 12. I could not finish it completely. So here is a YouTube video where I explain this topic:

https://youtu.be/_TPu-GoDpN8

In the above video, I explain the topics that I have written in the subsequent sections.

Please note: I did not conduct any lecture on Tuesday (10/09/2024) because of minor 1. I have recorded this YouTube video as a compensation of Tuesday's lecture. It is MANDATORY to watch the video and also to read thiese notes.

3.1 Problem Statement

I gave the broad idea of this problem before lecture 12. During lecture 12, we discussed in class and came up with a problem statement that accounts for:

1. Trading with multiple stocks.
2. Price of one stock can effect the price of another stock.
3. Rise in stock prices under too much demand.
4. The overhead fees required for buying stocks. The overhead fees has a "bucket structure".
5. The objective is to earn a certain pre-specified amount of money as soon as possible.

NOTE: I only wrote the key points above. To understand the details of the setup, check the YouTube video.

3.2 MDP Formulation

3.2.1 State and State Space

The states of the system are:

1. $s_t = \begin{bmatrix} s_{1,t} & s_{2,t} & \cdots & s_{K,t} \end{bmatrix}$: $s_{i,t}$ is the number of i^{th}stocks the investor owns at time slot t.
2. $p_t = \begin{bmatrix} p_{1,t} & p_{2,t} & \cdots & p_{K,t} \end{bmatrix}$: $p_{i,t}$ price of the i^{th}stock at time slot t.
3. b_t: Bank balance of the investor at time slot t.

So, the state of the system is,

$$x_t = (s_t \ , \ p_t \ , \ b_t)$$

where s_t and p_t are themselves vectors.

Now, we have to find the state space of x_t. Now:

1. $s_{i,t} \in \{0, 1, \cdots, M_i\}, \ \forall i \in \{1, 2, \cdots, K\}, \ \forall t$. We assume that the investor will not have more than M_i stocks of the i^{th}stock in any time slot t. We have to put and upper bound M_i on $s_{i,t}$ because finding the optimal policy for an MDP (which is not discussed yet) is not straightforward if we don't upper bound $s_{i,t}$. That said, upper bounding $s_{i,t}$ is not completely impractical. It is possible that the company to which the i^{th}stock belong will not want any investor to own more than a certain number of stocks. Otherwise, there might be ownership issues!
   - This implies that $s_t \in \mathcal{S}_s$, $\forall t$ where $\mathcal{S}_s = \{0, 1, \cdots, M_1\} \times \{0, 1, \cdots, M_2\} \times \cdots \times \{0, 1, \cdots, M_K\}$.
2. $p_{i,t} \in \{0, 1, \cdots, P_i\}, \ \forall i \in \{1, 2, \cdots, K\}, \ \forall t$. We have to upper bound $p_{i,t}$ by P_i because of the same reason as $s_{i,t}$. Also, in reality, price is not discrete. However, finding the optimal policy for an MDP is not straightforward if price is continuous. Hence, we assume that $p_{i,t}$ can be only discrete values. *NOTE: Deep RL does not rely on such assumptions. So, these assumptions about the state space of $s_{i,t}$ and $p_{i,t}$ can be removed when we do Deep RL.*

- This implies that $p_t \in \mathcal{S}_p$, $\forall t$ where $\mathcal{S}_p = \{0, 1, \cdots, P_1\} \times \{0, 1, \cdots, P_2\} \times \cdots \times \{0, 1, \cdots, P_K\}$.

3. $b_t \in \mathcal{S}_b$, $\forall t$ where $\mathcal{S}_b = \{0, 1, \ldots, B\}$ and B is the bank balance after when the investor will stop investing in stock market (and retire!).

Finally, $x_t \in \mathcal{S}$, $\forall t$ where,

$$\mathcal{S} = (\mathcal{S}_s \times \mathcal{S}_p \times \mathcal{S}_b) \bigcup \{end\} \tag{6}$$

is the state space of x_t. In (6), the state *end* is required because this example is an **episodic task**, i.e. the task is over as soon as bank balance hits B.

3.2.2 Action and Action Space

The action is $a_t = \begin{bmatrix} a_{1,t} & a_{2,t} & \cdots & a_{K,t} \end{bmatrix}$ where $a_{i,t}$ is the number of stocks of the i^{th}stock that the investor buys/sells. A negative $a_{i,t}$ implies selling stocks while a positive $a_{i,t}$ implies buying stocks.

Now, about the action space. In the first glance, the action space of $a_{i,t}$ is,

$$\widetilde{A}_i\left(x_t\right) = \{-s_{i,t}, -s_{i,t}+1, \ldots -1, 0, 1, \cdots, M_i - s_{i,t}\} \tag{7}$$

where the lower bound $-s_{i,t}$ is because the investor can't sell more than the stocks it has of the i^{th}stock, and the upper bound $M_i - s_{i,t}$ is because the investor can't have more than M_i stocks at any time slot. Consequently, the action space of a_t is,

$$\widetilde{A}\left(x_t\right) = \widetilde{A}_1\left(x_t\right) \times \widetilde{A}_2\left(x_t\right) \times \cdots \times \widetilde{A}_K\left(x_t\right) \tag{8}$$

But, the above action space of a_t is WRONG because it does not account for the bank balance of the investor. $a_{1,t}, a_{2,t}, \ldots, a_{K,t}$ must satisfy,

$$\sum_{i=1}^{K} \left(p_{i,t} \cdot \max\left(0, a_{i,t}\right) + f_i\left(a_{i,t}\right)\right) \leq b_t \tag{9}$$

where $f_i\left(a_{i,t}\right)$ is function that captures the **overhead fees** required to buy $a_{i,t}$ stocks of the i^{th}stock. Note that $f_i\left(a\right) \equiv 0$ for all $a < 0$ and all i. In many cases, the overhead fees has a "bucket structure". Such a bucket structure can be modeled if $f_i\left(a\right)$ is a "staircase like" function. In (9), $p_{i,t} \cdot \max\left(0, a_{i,t}\right)$ is the cost to buy the i^{th}stock without accounting for the overhead fees. It is a positive valued quantity if the investor bought i^{th}stock and 0 is the investor sold i^{th}stock. The LHS of (9) is the cost of buying all the K stocks at time slot t. This amount can't exceed b_t, the bank balance of the investor at time slot t. Finally, the action space of a_t is,

$$\mathcal{A}\left(x_t\right) = \left\{a_t \in \widetilde{A}\left(x_t\right) : \sum_{i=1}^{K} \left(p_{i,t} \cdot \max\left(0, a_{i,t}\right) + f_i\left(a_{i,t}\right)\right) \leq b_t\right\}$$

NOTE: For this problem, the action space is a function of the state.

3.2.3 Reward and Reward Probability

The objective is to minimize the time required to have B as bank balance. For this objective, we can set the per time slot reward (also called **immediate reward**) as follow,

$$r_t = \begin{cases} -1 & ; b_t < B \\ 0 & ; b_t = B \quad \text{or} \quad x_t = end \end{cases} \tag{10}$$

The idea behind equation (10) is to use the reward as a counter when $b_t < B$. The minus sign is because we want to **minimize** the time required to have B as bank balance. The instant bank balance is B ($b_t = B$) or the episode ends ($x_t = end$), the counter stops.

SIDENOTE: The reward defined above definitely minimizes the time required to have a bank balance of B. That said, it is possible that with the above reward finding the optimal policy might be time consuming. Intuitively this is because with the above reward function, the agent keeps getting a penalty of -1 every time the bank balance is less than B. This means that the agent does not get any feedback until the very end (when bank balance is B) on whether the action it is taking is "good" or not. In order to combat this challenge, it may be worthwile to set up a different reward function that is more sensitive to the current state and the agents action. One possible choice is the money spent/earned in a time slot,

$$r_t = \begin{cases} -\sum_{i=1}^{K} \left(p_{i,t} a_{i,t} + f_i\left(a_{i,t}\right)\right) & ; b_t < B \\ 0 & ; b_t = B \quad \text{or} \quad x_t = end \end{cases} \tag{11}$$

Even though the reward function given by equation (11) may not minimize the time to reach bank balance of B, it is likely to speed up the process of converging to a near-optimal policy.

3.2.4 State Transition Probability

We are dealing with an episodic setup. The process of writting the state transition will be divided into two parts:

1. When the task ends. This is when $b_t = B$ or $x_t = end$.
2. When the task is in progress. This is when $b_t < B$.

Case - 1 (when the task ends, i.e. $b_t = B$ or $x_t = end$)

The state transition in this case is,

$$x_{t+1} = end \tag{12}$$

Equation (12) essentially says that the next state loops back to the *end* state after the task is over.

Case - 2 (when the task is in progress, i.e. $b_t < B$)

The state transition of $s_{i,t}$ is,

$$s_{i,t+1} = s_{i,t} + a_{i,t} \ , \ \forall i \in \{1, 2, \ldots, K\} \tag{13}$$

Basically, the number of stocks in the next time slot is the number of stocks in the current time slot plus/minus the number of stocks it buys/sells in the current time slot.

The state transition of b_t is,

$$b_{t+1} = \min\left(B, b_t - \sum_{i=1}^{K} \left(p_{i,t} a_{i,t} + f_i\left(a_{i,t}\right)\right)\right) \tag{14}$$

In (14), $\sum_{i=1}^{K} \left(p_{i,t} a_{i,t} + f_i\left(a_{i,t}\right)\right)$ is the money spent/earned by buying/selling stocks in the current time slot t. This money will be accordingly subtracted from/added to the bank balance. The function $\min(\cdot)$ is to upper bound the bank balance at B (we assume that if the investor earns more than B, it give the extra amount to charity).

Finally, we have to discuss the state transition of stock price p_t. The state transition of p_t is governed by a generic probability distribution,

$$P\left[p_{t+1} | p_t, a_t\right] \tag{15}$$

It is generic because:

1. It can model situations where **big firms manipulate future stock prices** by buying/selling a lot of stocks. To capture such scenarios, we need a model like that given by (15) where the number of stocks bought/sold in the current time slot, a_t, effects the stocks price in the future time slot, p_{t+1}.
   - If a_t does not effect p_{t+1} (like for small firms), then (15) simplifies to $P\left[p_{t+1} | p_t\right]$.
2. It can model situations where **stock price of one stock can effect the stock price of another** stock. This is because according to (15), the current stock price of all the K stocks, p_t, can effect every element of p_{t+1}.
   - A special case of $P\left[p_{t+1} | p_t, a_t\right]$ is where a stock can't effect other stocks, i.e. one stock is *independent* of the other. In such situations, $P\left[p_{t+1} | p_t, a_t\right]$ is of the following form

$$P\left[p_{t+1} | p_t, a_t\right] = \prod_{i=1}^{K} P\left[p_{i,t+1} | p_{i,t}, a_{i,t}\right]$$