EL2805 Lab 1: The Great Escape

Shuyuan Zhang
`shuyuanz@kth.se`

Zhenghong Li
`zhenghli@kth.se`

Abstract

This report answers the questions and includes relevant figures according to the lab instruction of lab1.

1 The Maze and the Random Minotaur

1.1 Modelling the MDP

State space $\mathcal{S}$: $\mathcal{S} = \{s_0, s_1, ..., s_{2239}, s_{exit}, s_{dead}\}$, $s_i = \{P_{x,y}, M_{x,y}\}$,$0 \le x \le 6$, $0 \le y \le 7$, where $s_i = \{P_{x,y}, M_{x,y}\}$ represents the positions of the person and the monster. In the maze, there are $7 * 8 = 56$ grids. However, since the person cannot go through the wall, the person could only stand on 40 positions (out of 56). Therefore, the number of non-terminal states in the state space is $56 * 40 = 2240$. In addition, there are 2 terminal states which represent the person exits the maze or the person is eaten by the monster.

Actions $\mathcal{A}$: $\mathcal{A} = \{up, down, left, right, stay\}$. Actions are the possible actions taken by the person, i.e., move up, move down, move left, move right, or stay. If the action cause the people to hit the wall, then the person stay at the grid.

Rewards $\mathcal{R}$:

$r(s = s_{exit}, a = \cdot) = 0, r(s = s_{dead}, a = \cdot) = 0$
$r(s = s_i, a = a_{wall}) = -100$
$r(s = s(P_{x,y} = M_{x,y}), a = \cdot) = -100$
$r(s = s(P_{x,y} = E_{x,y}), a = \cdot) = 1$
$r(s = s_i, a = a_{not\_wall}) = -1$

$P_{x,y}$ and $M_{x,y}$ represent the positions of the person and the monster. $E_{x,y}$ represents the location of the exit. a_{wall} means actions at a certain state that can cause the person to hit the wall, a_{not_wall} means actions at a certain state that don't lead to the wall.

Transition Probability $\mathcal{P}$

For terminal states:

$P(s' = s_{exit}|s = s_{exit}, a = \cdot) = 1$
$P(s' = s_{dead}|s = s_{dead}, a = \cdot) = 1$
$P(s' = s_{exit}|s = (P_{x,y} = E_{x,y}), a = \cdot) = 1$
$P(s' = s_{dead}|s = (P_{x,y} = M_{x,y}), a = \cdot) = 1$

For non-terminal states:
If the Minotaur cannot stand still:

$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(1,0)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.25$

$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(-1,0)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.25$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(0,1)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.25$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(0,-1)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.25$

If the Minotaur can stand still:
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(1,0)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.2$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(-1,0)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.2$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(0,1)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.2$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(0,-1)}\}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.2$
$P(s' = \{P_{(x,y)+a}, M_{(x,y)+(0,0)}|s = \{P_{x,y}, M_{x,y}\}, a = [a\_x, a\_y]) = 0.2$

$a = [a\_x, a\_y]$ is the vectorized version of actions. For example, $\mathcal{A} = \{left\}$ correspond to $a = [-1, 0]$. $P_{(x,y)+a}$ and $M_{(x,y)+(dx,dy)}$ represent positions of the person/Minotaur if they are at initial positions (x,y) and take action a or (dx, dy). Note that if the action cause the person/Minotaur hit the wall or boundary of the maze, then $P_{(x,y)+a} = P_{(x,y)}$ and $M_{(x,y)+(dx,dy)} = M_{(x,y)}$
All other elements in the transition matrix are 0.

Time horizon $\mathcal{T}$: This is a finite horizon problem. So there is a horizon T.

1.2 Is there a difference between the Minotaur can/cannot stand still?

First, we solved the DP problem and ran the simulation for different values for the time horizon, then calculated the probability of exiting the maze:

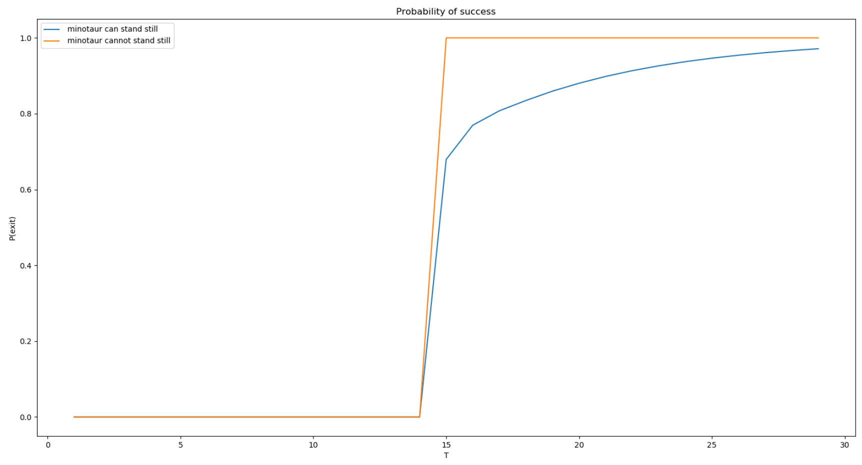


Figure 1: Success probability of different T

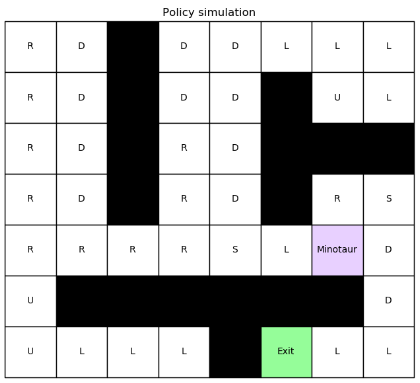
From the figure we may notice that there is a clear difference between the Minotaur can/cannot stand still.

When $T < 15$, the probabilities of successfully exit for both settings are all 0. This is because the length of the shortest path from the initial state to the exit is 15 so it cannot exit with a time horizon less than 15.

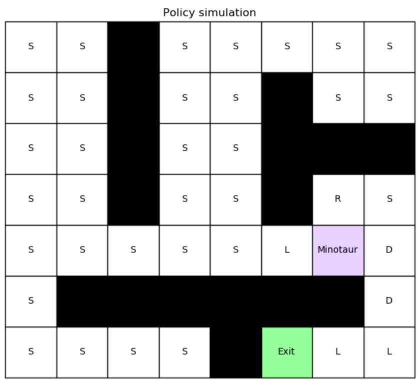
When T exceeds 15, the exit probability when the Minotaur cannot stand still rises immediately to 1, telling us that the policy can always help the player go out of the maze. While if the Minotaur can stand still, the probability rises slowly. One possible explanation is that the Minotaur may stay on a grid and blocks the way to the exit for several rounds and the player can do nothing but wait. Also, when the Minotaur cannot stand still, then the player is always safe if he stands next to the Minotaur. But this is not the case when Minotaur can stand still so the player needs to dodge it.

1.3 Showing policies for $T = 20$

Optimal policies when the Minotaur is at (4,6):



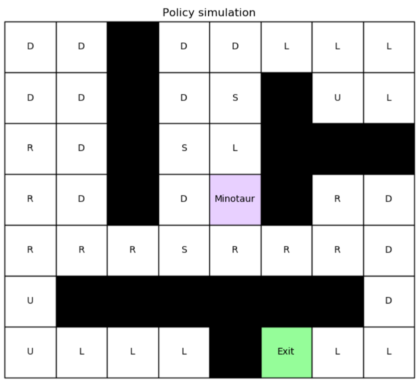
(a) $t = 1$



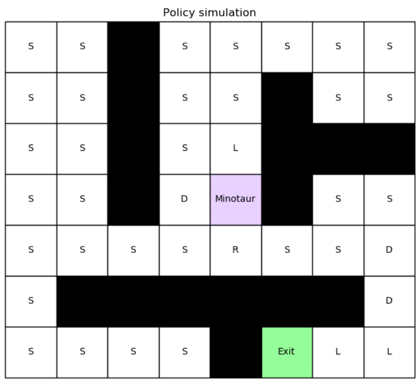
(b) $t = 15$

Figure 2: Optimal policy for Minotaur at (4,6)

Optimal policies when the Minotaur is at (3,4):



(a) $t = 1$



(b) $t = 15$

Figure 3: Optimal policy for Minotaur at (3,4)

Figures above show the optimal policy for the player when the Minotaur is at (4,6) or (3,4). For optimal policies when t=1, we can see that the player tries to be closer to the exit, and tries to dodge or wait for the Minotaur to leave when it's near the monster. When t=15, the player choose to stay when it is too far away from the exit because it only have 5 steps left. Staying is the best choice as moving has a minus reward.

1.4 Geometrically distributed life

Since that now the player's life is geometrically distributed with mean 30, we do not have a clear number for the time horizon $\mathcal{T}$. We then tend to model this problem as an infinite horizon MDP with:

$$\mathbb{E}[T] = \frac{1}{1-\lambda} = 30$$

State space $\mathcal{S}$, **Actions** $\mathcal{A}$, **Rewards** $\mathcal{R}$ and **Transition Probability** $\mathcal{P}$ are the same as before.

For **Time horizon** $\mathcal{T}$, now it is a infinite horizon MDP with the discount factor $\lambda = \frac{29}{30}$

Now, we do not have a terminal state, and we use a stationary policy, which means we applying the same one-step decision every time step. Also, to find the optimal policy, we need to solve Bellman's equations using a fixed point iteration algorithm - Value iteration.

We set $\epsilon = 0.001$ to check for convergence. After the training process, we ran the simulation for 10000 times on this stationary policy and escaped 10000 times. The escape rate is 1(100%)

2 Robbing Banks

Skipped

3 Bank Robbing (Reloaded)

We can model the problem similarly as question 1 and 2. Note that now it is a reinforcement learning problem so the agent does not know the transition probabilities and rewards, as well as the position of the bank, the starting point and the movement strategy of the police beforehand.

State space $\mathcal{S}$: $\mathcal{S} = \{s_0, s_1, ..., s_{255}\}$, $s_i = \{R_{x,y}, P_{x,y}\}$, where $s_i = \{R_{x,y}, P_{x,y}\}$ represents the positions of the robber and the police. $0 \leq x, y \leq 3$.

Actions $\mathcal{A}$: $\mathcal{A} = \{up, down, left, right, stay\}$.

Transition Probability $\mathcal{P}$**:**

$P(s' = \{R_{(x,y)+a}, P_{(x,y)+(1,0)}\}|s = \{R_{x,y}, P_{x,y}\}, a = [a\_x, a\_y]) = 0.25$
$P(s' = \{R_{(x,y)+a}, P_{(x,y)+(-1,0)}\}|s = \{R_{x,y}, P_{x,y}\}, a = [a\_x, a\_y]) = 0.25$
$P(s' = \{R_{(x,y)+a}, P_{(x,y)+(0,1)}\}|s = \{R_{x,y}, P_{x,y}\}, a = [a\_x, a\_y]) = 0.25$
$P(s' = \{R_{(x,y)+a}, P_{(x,y)+(0,-1)}\}|s = \{R_{x,y}, P_{x,y}\}, a = [a\_x, a\_y]) = 0.25$

$a = [a\_x, a\_y]$ is the vectorized version of actions. $R_{(x,y)+a}$ and $P_{(x,y)+(dx,dy)}$ represent positions of the robber/police if they are at initial positions (x,y) and take action a or (dx, dy). Note that if the action cause the robber/police hit the wall, then $R_{(x,y)+a} = R_{(x,y)}$ and $P_{(x,y)+(dx,dy)} = P_{(x,y)}$
All other elements in the transition matrix are 0.

Rewards $\mathcal{R}$:

$r(s = s(R_{x,y} = P_{x,y}), a = \cdot) = -10$
$r(s = s(R_{x,y} = P_{x,y} = B_{x,y}), a = \cdot) = -10$
$r(s = s(R_{x,y} = B_{x,y}), a = \cdot) = 1$

$R_{x,y}$, $P_{x,y}$ and $B_{x,y}$ means the position of the robber, police and bank. Other rewards are all 0.

3.1 Q Learning

In this part, we generated learning samples using random policy. That is, the robber just randomly choose its action at every state.

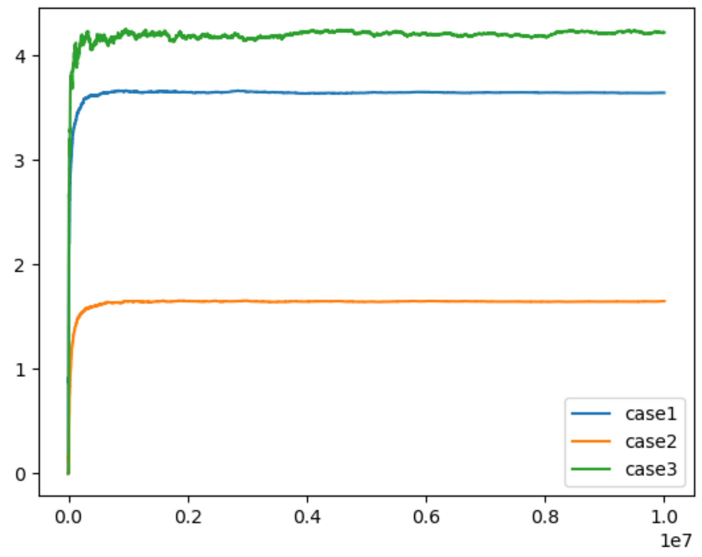


Figure 4: Value function of 3 states

We plotted the value function of the initial state and two other arbitrarily chosen states over time. In figure 1, case1 represents state (0,0,3,3), which is the initial state. Case2 represents state (3,3,2,2), which means the robber is at the down-right corner (3,3), and the police (2,2) is between the robber and the bank. Case3 represents state (1,2,1,3), which means the robber (1,2) is just next to the bank (1,1), and the police is at (1,3).

From the figure we can see that the value function converges rather quickly after a few iterations. And we can somehow comment on the value functions of the 3 chosen states. The converged value of initial state (0,0,3,3) in case 1 is about 3.6, while the value of state in case2 (3,3,2,2) is about 1.7, which is much smaller than the value of state (0,0,3,3). This is because the police is between the robber and the bank and the robber is far away from the bank so the expected reward is low.

On the other hand, the value of the state in case3 (1,2,1,3) is higher than the initial state because the robber is just next to the bank and he only needs 1 step to get the reward, so the expected total reward over time of this state is higher than the initial state.

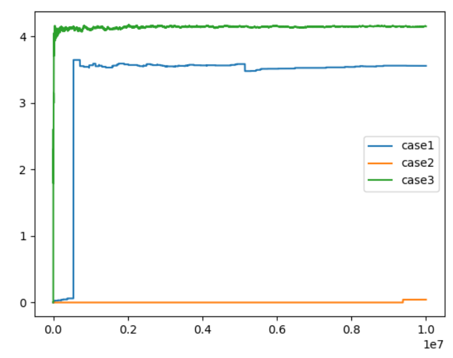
3.2 SARSA

In SARSA, the behaviour policy is an ϵ-greedy policy with regard to the Q function.

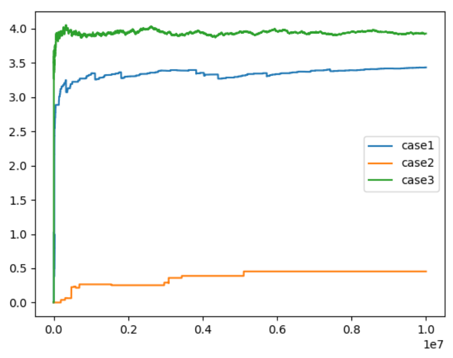
In this part we plotted the value function of the initial state and two other arbitrarily chosen states over time with different ϵ values:

In general, SARSA converges slower than Q-learning algorithm. The curve of value function still oscillates after many iterations.

In the figure, case 1,2 and 3 still represents the 3 states mentioned in the previous part, that is, (0,0,3,3), (3,3,2,2) and (1,2,1,3). When we study the relationship between the curve and ϵ values, one interesting conclusion is that the value for those states will decrease if we increase ϵ. The reason is that SARSA approximates the Q function under the ϵ-greedy policy and the value functions of

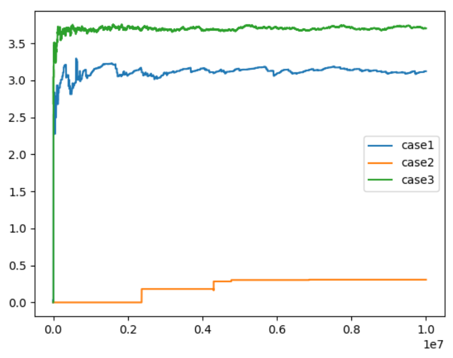


(a) $\epsilon = 0.01$

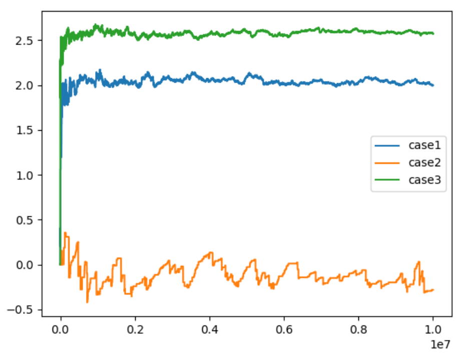


(b) $\epsilon = 0.05$

Figure 5: Value functions of different states with different ϵ (part 1)



(a) $\epsilon = 0.1$



(b) $\epsilon = 0.3$

Figure 6: Value functions of different states with different ϵ (part 2)

states are then less than the optimal value. As ϵ grows, the ϵ-greedy policy have a higher probability to make a mistake and values become smaller.