Dynamic Programming

Hisu-Wen Yen A59010599 hsyen@ucsd.edu

I. Introduction

Dynamic programming is a method to solve problem by breaking it into subproblem. The optimal solution for each subproblem will be the optimal solution for original problem. It'll be more efficient than brute force approach. Therefore, in robotics dynamic programming can help us solve problems involving states transition in a faster way.

II. Problem Formulation

A. Part A

Formulate this problem into a Markov Decision Process to help us find the optimal control policy.

To define state space X, we need to contain all possible variables. Therefore, X will be

$$X = \begin{bmatrix} carring \\ door \\ y_{cor} \\ x_{cor} \\ direction \end{bmatrix} \in R^5 \tag{1}$$

$$carring = \begin{cases} 0, & robot\ has\ key \\ 1, & robot\ don't\ have\ key \end{cases} \tag{2}$$

$$door = \begin{cases} 0, & door\ isn't\ open \\ 1, & door\ is\ open \end{cases} \tag{3}$$

$$y_{cor} = y\ axis\ coordinate\ in\ grid\ map\ \in R \tag{4}$$

$$x_{cor} = x\ axis\ coordinate\ in\ grid\ map\ \in R \tag{5}$$

$$direction = \begin{cases} 0, & robot\ face\ up \\ 1, & robot\ face\ right \\ 2, & robot\ face\ down \\ 3, & robot\ face\ left \end{cases} \tag{6}$$

Then control space U will be

$$U = \begin{cases} 0, & move\ forward \\ 1, & turn\ left \\ 2, & turn\ right \\ 3, & pickup\ key \\ 4, & unlock\ door \end{cases} \tag{7}$$

Motion model can be used to estimate state x_{t+1} by x_t and u_t. This will be elaborated in III.

$$x_{t+1} = f(x_t,\ u_t) \tag{8}$$

Initial state x_0 can be defined by how we put the robot before it takes any actions.

Final state x_{goal} is our expected state where the robot should be at the end.

To get the optimal control, we stop calculating once the shortest path is created. Therefore, the planning horizon T will be

$$T = \min\left(number\ of\ steps\ from\ x_0\ to\ x_{goal}\right) \tag{9}$$

Therefore, under the optimal policy x_T should be

$$x_T = x_{goal} \tag{10}$$

Stage cost l and terminal cost q are other criterion to estimate optimal control policy. To make the problem simple, I define in the following way

$$l = \begin{cases} 1, & for\ all\ reasonable\ actions \\ \infty, & for\ all\ not\ reasonable\ actions \end{cases} \tag{11}$$

$$q = \begin{cases} 0, & if\ x_T = x_{goal} \\ \infty, & if\ x_T \neq x_{goal} \end{cases} \tag{12}$$

B. Part B

The only difference between part A and part B is the environment. In part B, we now have two door. Therefore, we only need to rewrite the state space X as

$$X = \begin{bmatrix} carring \\ door\ 1 \\ door\ 2 \\ y_{cor} \\ x_{cor} \\ direction \end{bmatrix} \in R^6 \tag{13}$$

$$door\ 1 = \begin{cases} 0, & door\ 1\ isn't\ open \\ 1, & door\ 1\ is\ open \end{cases} \tag{14}$$

$$door\ 2 = \begin{cases} 0, & door\ 2\ isn't\ open \\ 1, & door\ 2\ is\ open \end{cases} \tag{15}$$

The rest is still same with part B.

III. Thectival Approches

A. Part A

To get optimal control policy $\pi_{0:T-1}$, we use dynamic programming algorithm. The truncated policy $\pi_{t:T-1}$ will be minimum π for the value function $V_t^{\pi}(x)$ at time t. Therefore, we compute the value function from T to 0.

$$V_t^{\pi}(x) = E\left[q(x_T) + \sum_{\tau=t}^{T-1} l\big(x_\tau, f(x_\tau,\ u_\tau)\big) \middle| x_t = x\right] \quad (16)$$

At each time stamp, construct two ndarray to store value function and control policy for corresponding states. Ndarray for value function called MV, ndarray for control policy called MM. Argument order as following

$[carring, door, y_{cor}, x_{cor}, direction]$

At Time T:

The robot should be in the goal area without any action. Therefore

All states in MM = -1 to represent no action applied.

$$MM_T[:,:,:,:,:] = -1 \quad (17)$$

For MV, only states in goal area are 0, other are ∞ .

$$MV_T[:,:,:,:,:] = \infty \quad (18)$$

Except

$$MV_T\left[:,:,y_{goal},x_{goal},:\right] = 0 \quad (19)$$

At Time $T-1$:

According to (16), the value function now needs to add stage cost l. As defined in (11) all reasonable action will make stage cost = 1. To define reasonable action as following

To simplify, call the cell in front of robot front cell.

For action 0: Move forward

It's reasonable if front cell is walkable.

Only these two conditions will be not walkable.

If front cell is "Wall", it isn't walkable.

If front cell is "Door" and the state "door" = 0 (meaning the door is close), it isn't walkable.

To compute $V_{T-1}{}^{0}(x_{T-1})$ (value function at time T-1 with action 1), we still need to know the $f(x_{T-1},\ \pi = 0)$.

$$x_T = f(x_{T-1},\ \pi = 0) = \text{front cell} \quad (20)$$

Only the y_{cor}, x_{cor} will change to $y_{front\ cell}, x_{front\ cell}$.

We can utilize MV_T to get $V_T(x_T)$.

$$V_{T-1}{}^{0}(x_{T-1}) = \begin{cases} 1 + V_T(x_T), if\ it's\ walkable \\ \infty,\ if\ it's\ not\ walkable \end{cases} \quad (21)$$

For action 1: turn left

It will be reasonable for all states.

To compute $f(x_{T-1},\ \pi = 1)$, only the $direction$ state will change by the following order.

$$3 \rightarrow 2 \rightarrow 1 \rightarrow 0 \rightarrow 1 \quad (22)$$

$$V_{T-1}{}^{1}(x_{T-1}) = 1 + V_T(x_T) \quad (23)$$

For action 2: turn right

It will be reasonable for all states.

To compute $f(x_{T-1},\ \pi = 2)$, only the $direction$ state will change by the following order.

$$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 0 \quad (22)$$

$$V_{T-1}{}^{2}(x_{T-1}) = 1 + V_T(x_T) \quad (23)$$

For action 3: pickup key

It's only reasonable if the front cell is "Key".

To compute $f(x_{T-1},\ \pi = 3)$, only the $carring$ state will change to 1.

$$V_{T-1}{}^{3}(x_{T-1}) = \begin{cases} 1 + V_T(x_T), if\ front\ cell\ is\ "Key" \\ \infty,\ if\ front\ cell\ isn't\ "Key" \end{cases} \quad (24)$$

For action 4: unlock door

It's only reasonable if the front cell is "$door$" and "$carring$" = 1.

To compute $f(x_{T-1},\ \pi = 4)$, only the $door$ state will change to 1.

$$V_{T-1}{}^{4}(x_{T-1}) = \begin{cases} 1 + V_T(x_T), if\ it\ satifies\ both\ conditions\ above \\ \infty,\ if\ it\ fails\ one\ of\ conditions\ above \end{cases} \quad (25)$$

Find the min between $V_{T-1}{}^{0} V_{T-1}{}^{1}, V_{T-1}{}^{2}, V_{T-1}{}^{3}, V_{T-1}{}^{4}$ and record it to $MV_{T-1}[x_{T-1}]$, record the corresponding action to $MM_{T-1}[x_{T-1}]$.

$$MV_{T-1}[x_{T-1}] = \min\,[V_{T-1}{}^{0}, V_{T-1}{}^{1}, V_{T-1}{}^{2}, V_{T-1}{}^{3}, V_{T-1}{}^{4}] \quad (26)$$

$$MM_{T-1}[x_{T-1}] = argmin_{\pi}\ (MV_{T-1}[x_{T-1}]) \quad (27)$$

However, if $MV_{T-1}[x_{T-1}] = \infty$ means no action is reasonable, then

$$MM_{T-1}[x_{T-1}] = -1 \quad (28)$$

Repeat doing this until $MM_{t^*}[x_0] \neq -1$.

This is because the cost for all action is 1. Once the connection between x_0 and x_{goal} is created, it must be the shortest path. Therefore t^* is our time stamp 0.

Now, we can move forward to get the control policy by MM.

Procedure:

$$MM_0(x_0) = \pi_0 \tag{29}$$

$$x_1 = f(x_0, \pi = \pi_0) \tag{30}$$

$$MM_0(x_1) = \pi_1 \tag{31}$$

$$x_2 = f(x_1, \pi = \pi_1) \tag{32}$$

$$\vdots$$

$$MM_{T-1}(x_{T-1}) = \pi_{T-1} \tag{33}$$

This will give us optimal control policy $\pi_{0:T-1}$.

B. *Part B*

Basically similar to part A, but we need to add one more dimension.

MM and MV will be

$$[carring, door\ 1, door\ 2, y_{cor}, x_{cor}, direction]$$

The only big difference is in motion model for action 4 "unlock door".

If the front cell is "*door* 1", then the reasonable action will make state "*door* 1" = 1, and keep "*door* 2" intact.

$$[\text{“}door\ 1\text{”}, \text{“}\ door\ 2\text{”}] = [1, \text{“}\ door\ 2\text{”}] \tag{34}$$

If the front cell is "*door* 2", then the reasonable action will make state "*door* 2" = 1, and keep "*door* 1" intact.

$$[\text{“}door\ 1\text{”}, \text{“}\ door\ 2\text{”}] = [\text{“}door\ 1\text{”}, 1] \tag{35}$$

Do the same procedure as part A, we will get optimal control policy.

IV. Results

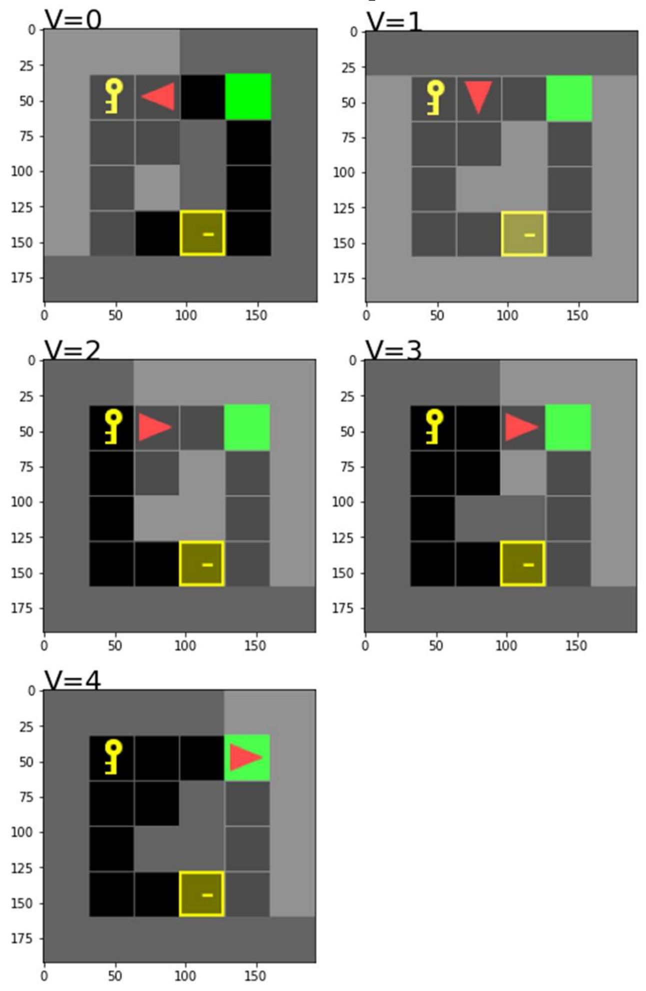
A. *Part A*

Example 1

Initial state:

$$[carring: 0,\ door: 0,\ y_{cor}: 1,\ x_{cor}: 2,\ direction: 3]$$

Time interval in picture:1

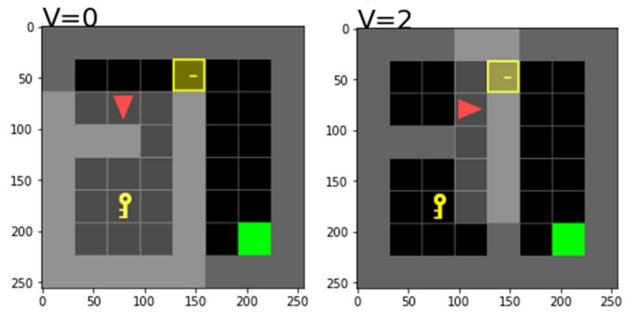


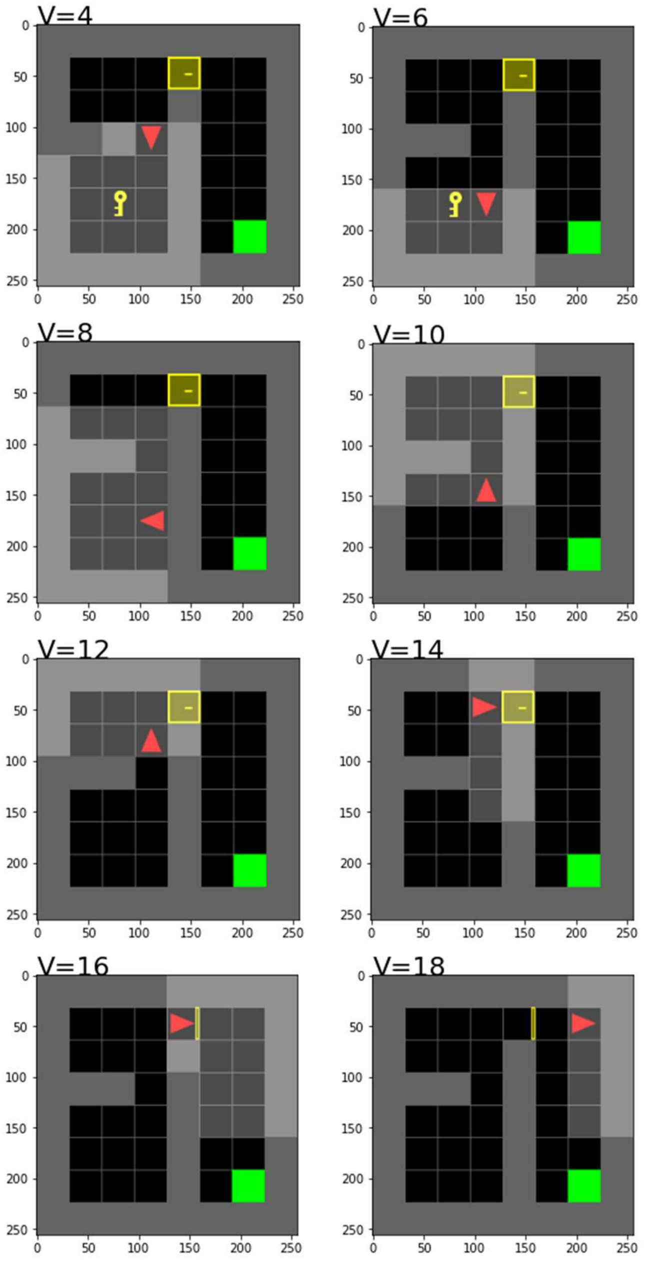
Example 2

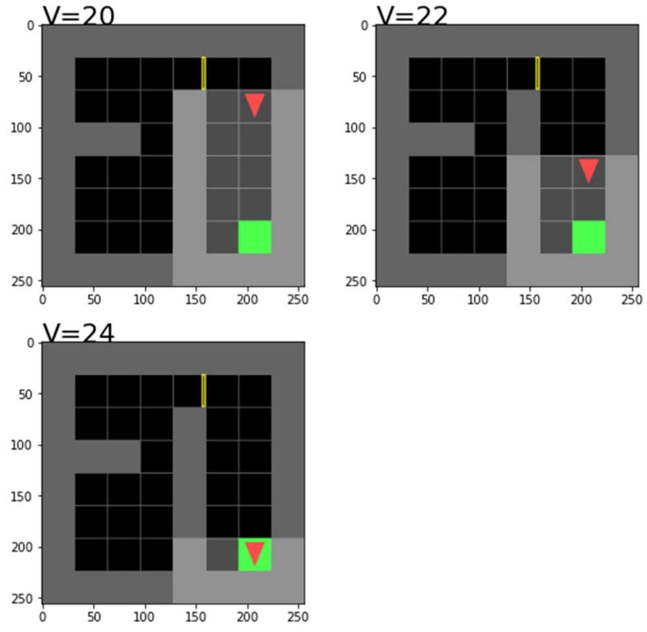
Initial state:

$$[carring: 0,\ door: 0,\ y_{cor}: 2,\ x_{cor}: 2,\ direction: 2]$$

Time interval in picture:2







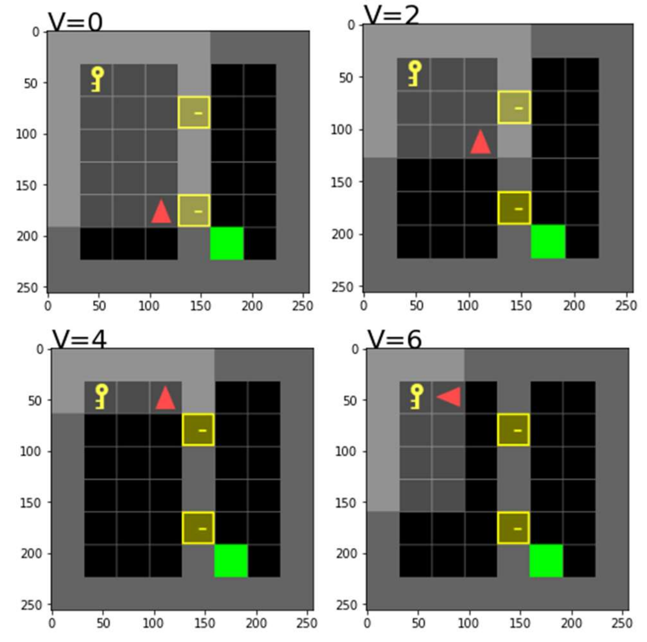
B. *Part B*

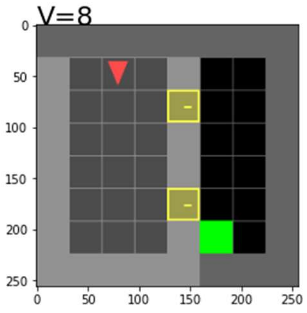
Example 3

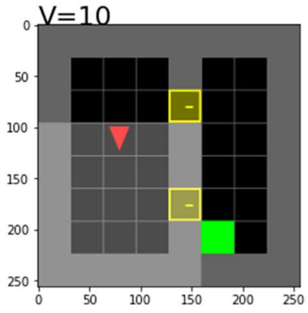
Initial state:

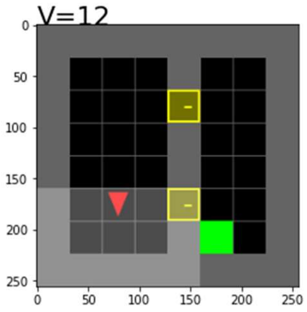
$[\,carring\!:0\,,\quad door\ 1\!:0\,,\quad door\ 2\!:0\,,\quad y_{cor}\!:5\,,\quad x_{cor}\!:3\,,\ direction\!:0]$

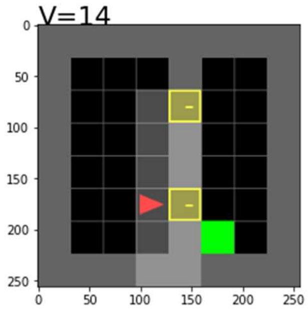
Time interval in picture:2

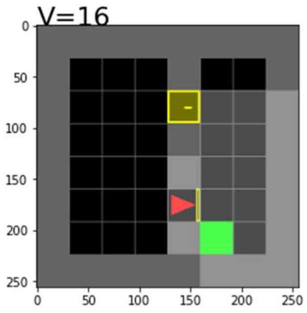


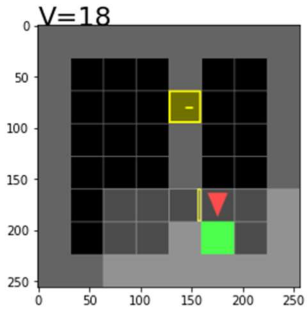










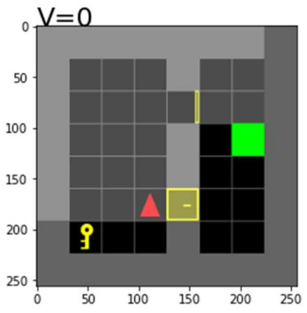


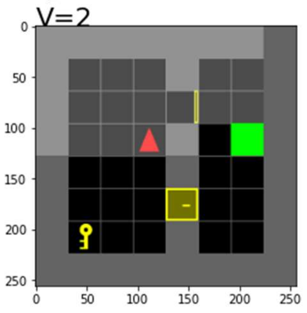
Example 4

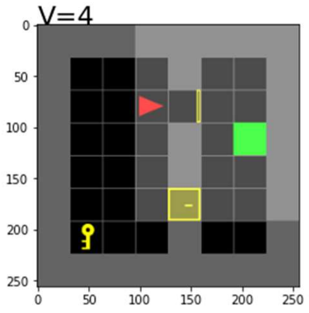
Initial state:

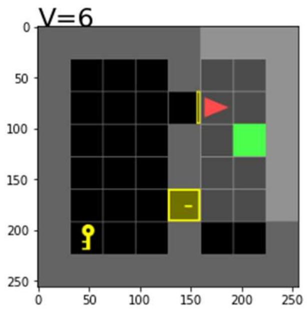
[$carring$: 0 , $door$ 1: 1 , $door$ 2: 0 , y_{cor}: 5 , x_{cor}: 3 , $direction$: 0]

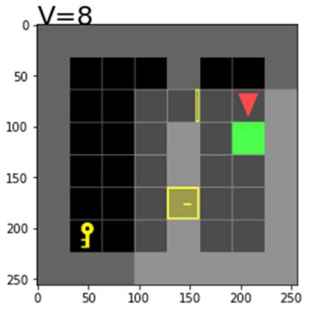
Time interval in picture:2











V. Reference

1. Professor Nikolay Atanasov's slides (lecture2~4)