Integrated System Biology Report

Department of Systems Science
Ma Zhenjia, ID: 6930-29-4499
Instructing Teacher: Ishii Shin

2017/08/18

1 Problem Description

Train an agent that automatically search the solution path in a maze using *TD0* algorithm.

2 Major Algorithm

The *TD0* method of reinforcement learning is based on a sampling method that takes sample movement in the given state space to collect samples, and improve the value function according to the observed reward. The notation *TD* refers to the full name *Temporary Difference* that use a difference between the current value and the temporary value to update the value function, which is

$$V(x, y) \leftarrow V(x, y) + \alpha(\sum_{t=0}^{T} r_t \gamma^t + \gamma^{T+1} V(x', y') - V(x, y))$$

where $\sum_{t=0}^{T} r_t \gamma^t + \gamma^{T+1} V(x_t, y_t)$ refers to the temporary value for $T + 1$ step and α refers to the training rate. *TD0* refers to the algorithm when $T = 0$ and thus

$$V(x, y) \leftarrow V(x, y) + \alpha(r + \gamma V(x', y') - V(x, y))$$

Take sufficient large quantity of samples and updates the value function following the formula above until convergent. The discount parameter γ is set to be 0.95.

3 Program Implementation

The coding is constructed by the following parts.

3.1 Environment: *Maze_ENV.py*

Environment class that describe the property and the action of the environment (maze), which is totally determined by the problem definition. It contains the query of the current state, the current reward, whether it is a terminate state or not, and the receptor of the action taken followed by the process of state dynamic and returning whether or not it is a legal action.

3.2 Agent: *Maze_AGNT.py*

Agent functions that interacts with the environment object. The function *ExecEpi()* refers to the process to take a complete episode until terminate while *TD0()* is a sub-function that returns the *TD0*-evaluated ϵ - *greedy* driven action selection result, where ϵ is a hyper-parameter controls the trade-off between the exploration and the exploitation in sampling search.

3.3 Trainer: *Maze_Train.py*

Trainer that find the optimal value function for this problem. Call the *Agent* to run for 10000 times and promise the convergence of value function.

4 Variated Studying Rate

There are two parameters, searching trade-off ϵ and the training rate α, that are vital to the proper convergence of the algorithm, especially ϵ. At the early stage of iteration, the value for most of the states are 0, which refers that their value have not been updated yet. Thus in these area, the *TD0* value can make little difference and the program is supposed to search more explorative, which require ϵ to be as large as possible to improve the chance for variance states to be reached. However, at the later stage, where according to the *TD0* update, we have already get a rough sketch of the hole state space. At this time, the value function we got plays a more important part when making decision. The algorithm is supposed to take action that is more likely to lead to the goal, and that require ϵ to be as small as possible, which stands in contrast with the constrain in early stage. To tackle this problem, in this report, a variated parameter is used that

$$\epsilon_t = \epsilon_0 c^{f(t)}$$

where c is discount constant in $(0, 1)$ and $f(t)$ is a monotonous function of times step t. So this method promise the algorithm to do more exploration at the beginning, and turn to focus on exploiting the likely states later on.

The training rate α also takes the similar idea, that at the beginning, we want a high training rate to make the value function quickly get close to the true value, and as the iteration goes on, this rate is supposed to be as small as possible to make smooth approach to the true value and prevent a large variance.

5 Result

The convergence result of value function is shown below The value for start point is

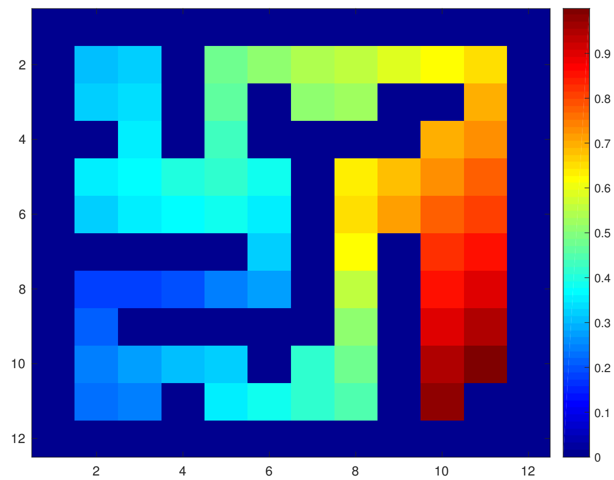

Figure 1: Value Function

arount 3.072, which that according to the definition of value function and reward, the convergence value for a certain state should be

$$V(s) = \gamma^{d(s,g)}$$

where $d(s, g)$ denotes the distant from s to goal. So the required steps from start point to the goal could be calculated by

$$d(s, g) = \frac{\ln V(s)}{\ln \gamma} \approx 23$$

which meets the human result for the maze problem.

The training curve for it is shown below

As shown in the figure, the curve started to converge after a quite long period since the discount rate is set to be large. If the discount rate is set to be smaller. the training curve would become which makes a quick convergence to the true value. However, effectiveness of algorithm in this report is promised by the selection of a sufficient long period for almost free search at the early stage, controlled by the discount process been executed every 200 complete episode. If the discount rate is set to be too low and there is no minimum undiscounted episode constrain, the algorithm may converge to a local maximum or even may not be able to get a solution.

The implement source code is uploaded on *https://github.com/mazhenjia007/Maze*

6 Reference

Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. Vol. 1. No. 1. Cambridge: MIT press, 1998.

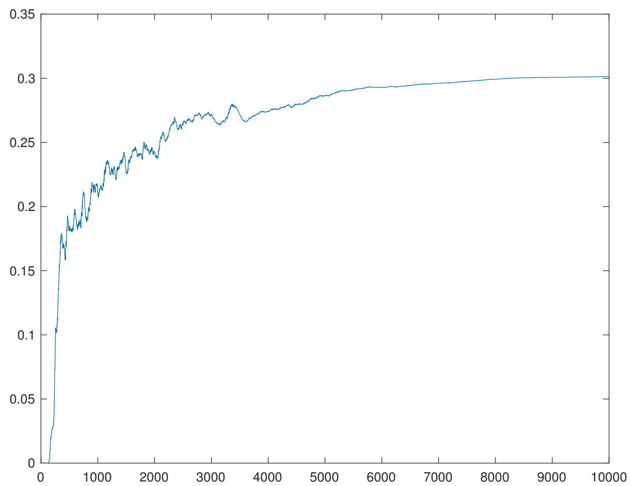

Figure 2: Training Curve with discount rate 0.95

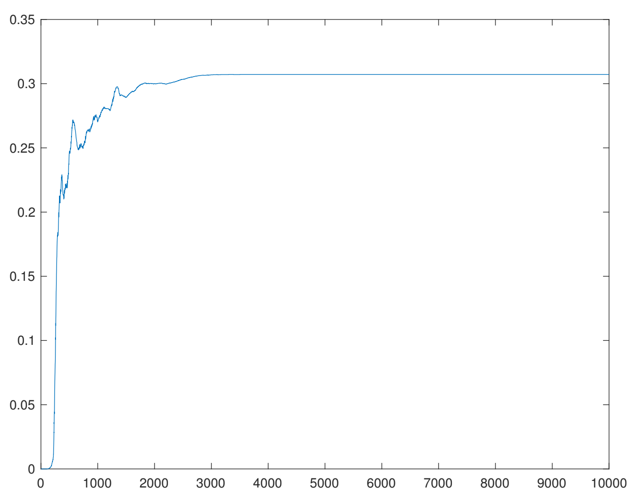

Figure 3: Training Curve with discount rate 0.8