Artificial intelligent Coursework

1 Introduction

The artifical intellgent is extensively studied in many fields of research. One of the branches that begins to have more boom nowadays worldwide in Artificial Intelligence, is the field of Intelligent Agents.The agents can be classified according to different enviornments. In the problem of FrozenLake, we could define the intelligent agent in following aspects:

- Performence measure: the agent need to find the solution path from start point to goal point. The less iteration and episode the agent use, the better agent would perform.
- Enviornment: the agent need to work on the frozen lake which includes four kinds of states, "G" means goal state, "F" means ice surface which could be slippery for agent to take action and "S" represents the start state and "H" means the holes on the frozenlake. The whole enviornment is a 8*8 or 4*4 text maze.
- Actuator: The agent would take actions like left, right , up and down to interact with enviornment for solving the frozenlake maze.These actions are represented in an array of [0,1,2,3].
- Sensors: In the construction of agent, the agent need to sensor the reward after taking each step and make the next move towards the goal of problem. The goal state would give reward of 1.0 and hole state would return the reward of -0.01 while other state would give 0.0.

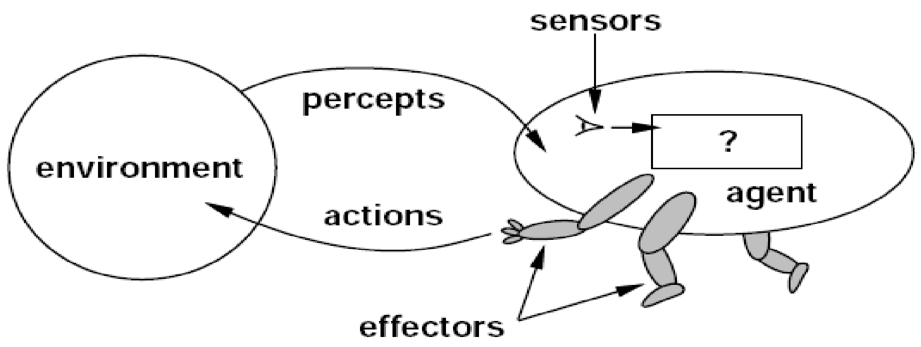

Figure 1: Intelligent agent

In the following part, three methods would be introduced to solve the problem.They are random agent, simple agent and reinforcement learning agent.

2 Method

- Senseless/random agent
  The random agent would take any possible action in any state until the agent reach the state of goal or hole.The agent can not sensor the external enviornment and have no knowledge of state-space and prior reward.The only thing agent can do is taking random action while it running.

- Simple agent
  The astar algorithm is applied in building the simple agent. The agent requires to sense all the details of enviornment and know about the prior state and goal state.
  The priciple of A* algorithm is starting from a specific starting node of a graph, it aims to find a path to the given goal node having the smallest cost (least distance travelled, shortest time, etc.) It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied. At each iteration of its main loop, A* needs to determine which of its paths to extend. It does so based on the cost of the path and an estimate of the cost required to extend the path all the way to the goal. Specifically, A* selects the path that minimizes following equation:

  $$f(n) = g(n) + h(n)$$

  In the equation, $g(n)$ represents the cost of start state to current state and $h(n)$ represents the cost of goal state to current state.The agent would terminates when the path it chooses to extend is a path from start to goal or if there are no paths eligible to be extended.

- Reinforcement Learning agent
  we applied the Q-learning algorithm for the reinforcement agent. The agent requires to gain perfect knowledge about the current state and available actions in that state.However,no prior knowledge about the state-space in general.Since the agent would take action based on the Q table possibility, the action could be noisy.
  The basic idea of Q-learning is to train the action-state policy table(Q-table) to guide the agent what to do in each state.Therefore, the algorithm has a function to calculate the quality of the policy table. Before learning stage, the Q table is initialised to arbitrary values. Then, at each time t the agent select an action a and observe the reward r and Q table is updated.
  To update the possibility of policy table, the agent applied formula based on Bellman Equation:

  $$Q^{max}(s_t, a_t) = Q(s_t, a_t) + \alpha * (r_t + \gamma * max(Q(s_{t+1}, a_t)) - Q(s_t, a_t))$$

  where r_t is the reward received when moving from the state s_t to the state s_{t+1} , and α is the learning rate ($0 < \alpha < 1$).

3 Implementation

To implement the Q-learning agent, we need to construct the learning policy of agent in the part of choose step.In order to make agent explore most state of problem, we generate the random number between 0 and 1 and set epsilon at 0.9 to make the agent take random step in most time.

```
def choose_action(state, epsilon, env, Q):
        action = 0
        if np.random.uniform(0, 1) < epsilon:
                action = env.action_space.sample()
        else:
                action = np.argmax(Q[state, :])
        return action
```

The learning policy is defined based on the formula stated before. So the details are as following:

```
        def learn(state, state2, reward, action, Q,
         gamma, lr_rate):
                predict = Q[state, action]
                target = reward + gamma * np.max(Q[state2, :])
                Q[state, action] = Q[state, action] + lr_rate *
                 (target - predict)
```

The A* agent is implemented in function "my_best_first_graph_search" and "my_astar_search" as following:

```
def my_best_first_graph_search(problem, f, initial_node_colors):
def my_astar_search(initial_node_colors, problem, h=None):
```

4 Evaluation

To evaluate the performance of each agent, we take random agent as first to exam. The random agent can hardly reach the final goal,especially in 8*8 frozen-lake problem. For 2000 episode and 500 iterations in problems, the agent only could have less than 10 times success to the goal. The following table shows the performance of random agent on each problem id:

problem_id	0	1	2	3	4	5	6	7
success times	0	0	0	6	0	3	0	11

For A* agent, since the agent know the entire enviornment, the agent would success to reach the goal very time.We evaluate the iteration step the agent would take in the following table:

problem_id	0	1	2	3	4	5	6	7
iterations	155	148	141	146	146	152	146	146

For Q-learning agent, we take 50000 episode to train the policy table for non-stochastic problem. During the learning phase, the comparison of iterations between Q-learning and A* agent is shown in following figures:

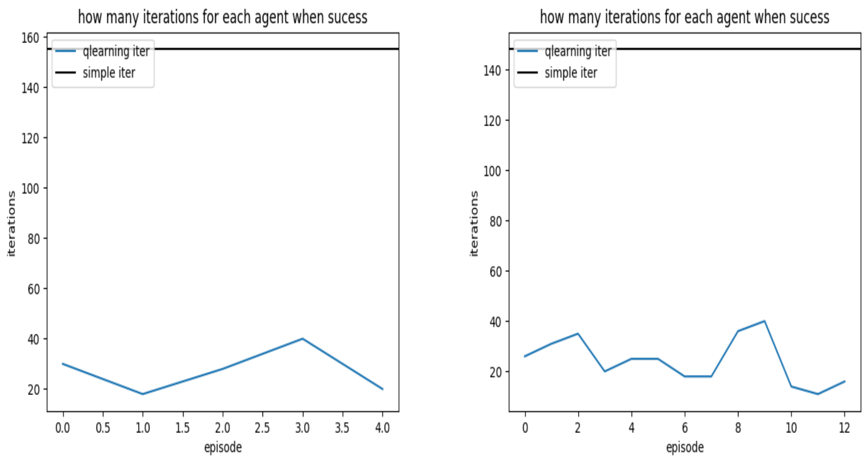

Figure 2: Iterations on problem id 0 & 1

5 Discussion

The Q-learning agent could successfully reach the goal state based on the guidance of policy table after learning phase within 5 episodes and max steps of 100(in rl_test.py).Even though the simple agent(A* agent) also could manage to solve the problem, it requires to know the whole enviornment before the agent could take action, which is rarely seen in the real world situation.However, the reinforcement learning agent could learn the better policy to improve the performance. It still requires much more time to learn the policy table, which remain a huge field to study.