Q-Learning Algorithm

by Kardi Teknomo (../../index.html)

(purchase.html)

< Previous (Q-Learning.htm) | Next (Q-Learning-Example.htm) | Contents (index.html) >

Q-Learning Algorithm

Our virtual agent will learn through experience without teacher (this is called unsupervised learning). The agent will explore state to state until it reaches the goal. We call each exploration as an *episode* . In one episode the agent will move from initial state until the goal state. Once the agent arrives at the goal state, program goes to the next episode. The algorithm below has been proved to be convergence (See references for the proof (Resources.html))

Q Learning

Given : State diagram with a goal state (represented by matrix R)

Find : Minimum path from any initial state to the goal state (represented by matrix Q)

Q Learning Algorithm goes as follow

1. Set parameter γ , and environment reward matrix R
2. Initialize matrix Q as zero matrix
3. For each episode:
    - Select random initial state
    - Do while not reach goal state
        - Select one among all possible actions for the current state
        - Using this possible action, *consider* to go to the next state
        - Get maximum Q value of this next state based on all possible actions
        - Compute $\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$
        - Set the next state as the current state

End Do

End For

The above algorithm is used by the agent to learn from experience or training. Each episode is equivalent to one training session. In each training session, the agent explores the environment (represented by Matrix R), get the reward (or none) until it reach the goal state. The purpose of the training is to enhance the 'brain' of our agent that represented by Q matrix. More training will give better Q matrix that can be used by the agent to move in *optimal* way. In this case, if the Q matrix has been enhanced, instead of exploring around and go back and forth to the same room, the agent will find the fastest route to the goal state.

Parameter γ has range value of 0 to 1($0 \leq \gamma < 1$). If γ is closer to zero, the agent will tend to consider only immediate reward. If γ is closer to one, the agent will consider future reward with greater weight, willing to delay the reward.

To use the Q matrix, the agent traces the sequence of states, from the initial state until goal state. The algorithm is as simple as finding action that makes maximum Q for current state:

Algorithm to utilize the Q matrix

Input: Q matrix, initial state

1. Set current state = initial state
2. From current state, find action that produce maximum Q value
3. Set current state = next state
4. Go to 2 until current state = goal state

The algorithm above will return sequence of current state from initial state until goal state.

< Previous (Q-Learning.htm) | Next (Q-Learning-Example.htm) | Contents (index.html) >

Preferable reference for this tutorial is

Teknomo, Kardi. 2005. Q-Learning by Examples. http://people.revoledu.com/kardi/tutorial/ReinforcementLearning/index.html