Sarsa

May 6, 2021

[22]:

```python
import matplotlib
import numpy as np
import gym
import matplotlib.pyplot as plt
import random
```

[23]:

```python
policy = np.zeros([9, 6])
policy = [[-1, 1, -1, -1, -3, 50],
     [1, 1, -1, 1, 5, -10],
     [1, -1, -1, -1, 5, -10],
     [-1, 1, -1, -1, 5, -10],
     [1, 1, 1, 1, 5, -10],
     [1, -1, -1, -1, 5, -10],
     [-1, 1, -1, -1, 5, -10],
     [1, 1, 1, -1, 5, -10],
     [1, -1, -1, -1, 5, -10]]
```

[24]:

```python
class OfficeEnv(gym.Env):

    def __init__(self):

        self.action_space = 6
        self.state_space = 9
        self.observation_space = 72
        reward = 0
        state = random.randint(0,self.state_space-1)

    def step(self, action):


        state = random.randint(0,self.state_space-1)
        reward = policy[state][action]

        done = True

        info = {}

        return state, reward, done, info
    def reset(self):
        state = random.randint(0,env.state_space-1)
        reward = 0
        return state
```

[25]:

```
env = OfficeEnv()
```

[61]:

```
import gym
import itertools
from collections import defaultdict
import numpy as np
import sys
import time
from multiprocessing.pool import ThreadPool as Pool



from collections import defaultdict


def make_epsilon_greedy_policy(Q, epsilon, nA):

    def policy_fn(observation):
        A = np.ones(nA, dtype=float) * epsilon / nA
        best_action = np.argmax(Q[observation])
        A[best_action] += (1.0 - epsilon)
        return A
    return policy_fn

def sarsa_lambda(env, num_episodes, discount=0.9, alpha=0.01, trace_decay=0.9,␣
 ↪epsilon=0.1, type='accumulate'):

    Q = defaultdict(lambda: np.zeros(6))
    E = defaultdict(lambda: np.zeros(6))

    policy = make_epsilon_greedy_policy(Q, epsilon, 6)

    rewards = [0.]
    r_vals = []
    for i_episode in range(num_episodes):

        print("\rEpisode {}/{}. ({})".format(i_episode+1, num_episodes,␣
 ↪rewards[-1]), end="")
        sys.stdout.flush()

        state = env.reset()
        action_probs = policy(state)
```