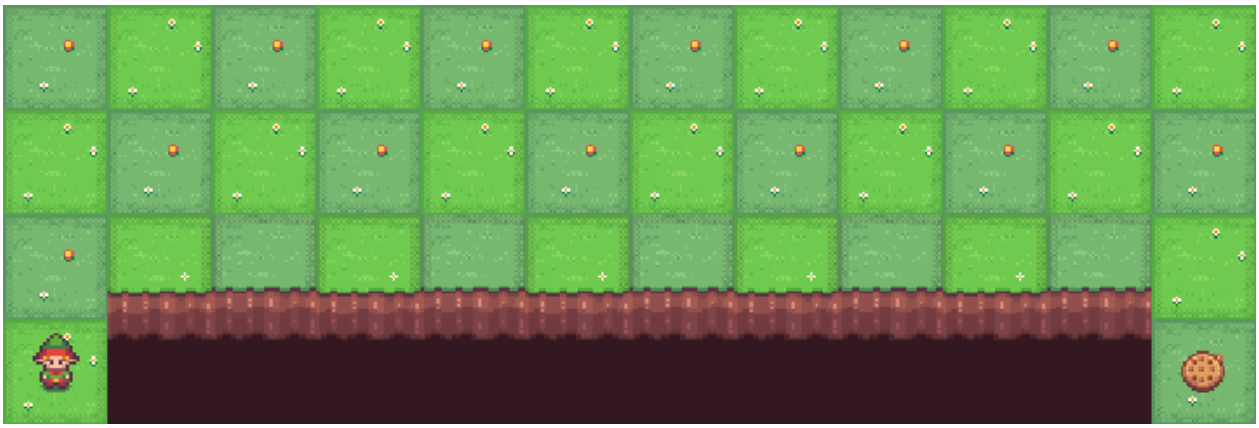
Cliff Walking



توضیح کلی

هدف این پروژه پیاده‌سازی یک عامل هوشمند است که بتواند در یک محیط غیرقطعی به فعالیت پرداخته و اهداف محیط را برآورده کند. برای درک هرچه بهتر کنش‌های عامل از محیط‌های فریمورک `gymnasium` استفاده می‌کنیم. هدف این فاز از پروژه پیاده‌سازی تابعی برای انتخاب مناسب کنش‌ها با کمک مواردی است که از فرآیند تصمیم مارکوف آموخته‌اید.

```
pip install 'gymnasium[all]'
```

محیط

در این فاز از پروژه از محیط Cliff Walking استفاده می‌کنیم که متناسب با نیاز پروژه ما شخصی‌سازی شده است. پیشنهاد می‌شود برای درک بهتر پروژه و نحوه پیاده‌سازی قسمت خواسته شده حتما صفحه این محیط را مطالعه کنید. این محیط شامل خانه شروع (`[3, 0]`)، خانه هدف (`[3, 11]`)، خانه‌های صخره (به صورت تصادفی در هر اجرا تعیین می‌شود) و عامل است. وظیفه شما هدایت عامل از خانه شروع به خانه هدف بدون ورود به خانه صخره‌هاست. در صورتی که عامل به داخل خانه‌های صخره وارد شود تعامل با محیط (اپیزد) با پایان می رسد.

کنش‌های ممکن (action space)

عامل می‌تواند کنش‌های بالا (0)، راست (1)، پایین (2) و چپ (3) را درصورتی که حرکت آن مجاز باشد (در گوشه‌ها نباشد) انتخاب نماید. نکته بسیار مهم اینکه کنش‌های عامل غیرقطعی هستند و به صورت زیر عمل می کنند: درصورتی که عامل قصد داشته باشد به هر سمتی حرکت کند تنها به احتمال یک سوم می تواند به آن سمت برود و به احتمال یک سوم به هر یک از کنش‌های همسایه‌اش می‌رود. در واقع اگر عامل حرکت به سمت بالا را انتخاب کند، تنها به احتمال یک سوم به سمت بالا رفته و به احتمال یک سوم به چپ و یک سوم به راست می‌رود. با توجه به اینکه کنش پایین در همسایگی کنش بالا قرار ندارد و دقیقا روبه روی آن می باشد، احتمال انتخاب آن کنش صفر خواهد بود.

مشاهدات عامل در محیط (observation space)

موقعیت عامل یک عدد صحیح می‌باشد که از ضرب سطر فعلی در تعداد ستون‌ها به علاوه ستون فعلی بدست می‌آید. به عنوان مثال در صورتی که عامل در خانه [3,0] قرار داشته باشد، عدد برگردانده شده به صورت ضرب سه در دوازده به علاوه صفر می‌باشد (3*12+0=36). درصورتی که عامل در گوشه‌ها باشد و حرکتی انجام دهد که از زمین بخواهد خارجش کند در جای خود مانده و آن کنش اعمال نمی‌شود.

$$observation = currentrow * ncols + currentcol$$

اتمام بازی (episode end)

بازی در حالات زیر پایان می‌یابد:

- درصورتی که عامل وارد خانه‌های صخره شود
- در صورتی که عامل بتواند به هدف نهایی برسد

توجه: پس از تعیین سیاست، عامل باید بتواند در تعداد تکرارهای محدودی اپیزد را به پایان ببرد که در زمان تحویل این مرحله از پروژه مشخص خواهد شد.

پاداش در محیط (reward)

در این محیط به ازای هر حرکت امتیاز `-1` و در صورتی که عامل در خانه صخره قرار گیرد امتیاز `-100` لحاظ می‌گردد.

- **توجه: در صورت نیاز می‌توانید پاداش‌های محیط را تغییر دهید (مثلاً در قالب تعریف یک تابع پاداش جدید) تا عامل بتواند سریعتر سیاست بهینه برای محیط را پیدا کند.**

توابع استفاده شده در کد

تابع `step`

پس از انتخاب کنش مناسب با توجه به الگوریتمی که پیاده‌سازی کرده‌اید، لازم است آن را روی محیط اعمال کنید. بدین منظور از تابع step استفاده می‌شود که به عنوان ورودی یک کنش (اعداد بین `0` الی `3`) را دریافت کرده و خروجی آن نتیجه اعمال آن کنش روی محیط به صورت یک tuple به شکل زیر است.

```
(next_state: int, reward: int, done: bool, truncated: bool, info: dict)
```

نکته: در صورتی که بازی به مشکلات سرور و یا زمان اجرا بر بخورد پارامتر `truncated` برابر مقدار `True` می‌شود و درصورتی که بازی تمام شود پارامتر `done` برابر مقدار `True` قرار می‌گیرد. پارامتر 'info' در ادامه توضیح داده شده است.

تابع `reset`

از این تابع برای برگرداندن عامل به نقطه شروع استفاده می‌شود و خروجی اول آن `next_state` نقطه شروع و خروجی دوم آن دیکشنری `info` است که حاوی احتمال رفتن به خانه‌های مجاور است و نیازی به تغییر در آن نیست.

- **توجه: برای انجام این بخش از پروژه لازم است کد خود را برای انتخاب کنش مناسب در قسمت `TODO` پیاده‌سازی کنید.**
- دانلود کد جهت پیاده‌سازی

cliff_walking.py

```python
# Create an environment
env = CliffWalking(render_mode="human")
observation, info = env.reset(seed=30)

# Define the maximum number of iterations
max_iter_number = 1000

for __ in range(max_iter_number):

    # TODO: Implement the agent policy here
    # Note: .sample() is used to sample random action from the environment's act

    # Choose an action (Replace this random action with your agent's policy)
    action = env.action_space.sample()

    # Perform the action and receive feedback from the environment
    next_state, reward, done, truncated, info = env.step(action)

    if done or truncated:
        observation, info = env.reset()

# Close the environment
env.close()
```

مستندات Cliff Walking

ویژگی‌های مستندات:

۱. نحوه کار الگوریتم را برای حل این مسئله توضیح دهید.

۲. چنان چه از منبعی به غیر از اسلایدهای درس و کتاب مرجع استفاده کرده‌اید حتما نام آن منبع را ذکر کنید.

۳. چنانچه از کتابخانه‌ای استفاده کرده‌اید (مطابق با شرایط ذکر شده در اطلاعیه) نام آن را ذکر نمایید.

۴. فایل آپلود شده به فرمت PDF ، دارای مشخصات دانشجویان نظیر نام و نام خانوادگی و شماره دانشجویی و شماره تیم باشد.