به نام خدا

درس هوش مصنوعی و سیستم‌های خبره

تمرین سری سوم

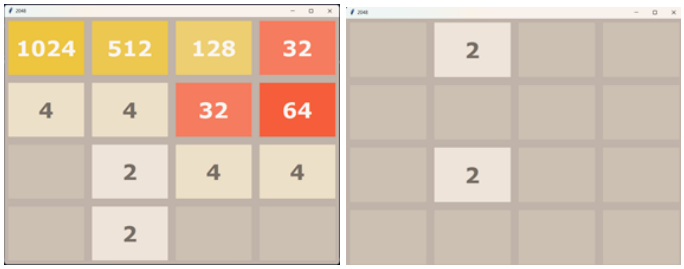
طراحان:
کامیار مرادیان زه آب

مدرس درس:
جناب آقای دکتر محمدی

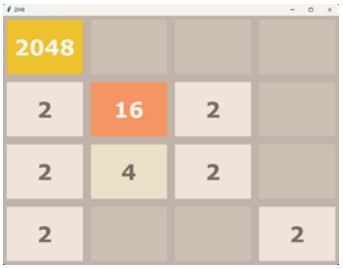
مهلت ارسال: ۱۴۰۲/۰۹/۱۰

۱ مقدمه و طرح مسئله

هدف این تمرین ساخت عاملی هوشمند برای حل کردن بازی ۲۰۴۸ است. تصاویری از محیط این بازی را می‌توانید در تصویر ۱، مشاهده کنید. محیط این بازی یک مربع متشکل از ۱۶ کاشی است که به صورت ۴ در ۴ قرار گرفته‌اند. هر کاشی آن می‌تواند مقدار اختیار کند یا مقداری نداشته باشد. در صورتی که یک کاشی مقداری داشته باشد، مقداری از توان ۲ خواهد داشت. در این بازی ۴ عمل وجود دارد که عبارتند از: حرکت صفحه به سمت چپ، حرکت صفحه به سمت راست، حرکت صفحه به سمت بالا و حرکت صفحه به سمت پایین. با اعمال هر یک از این عملیات، تمام مقادیر موجود در صفحه در جهت گفته شده و در صورت امکان ــ اگر سدی در مسیر وجود نداشته باشد: مانند مقدار نامساوی یا لبهٔ صفحهٔ بازی ــ حرکت می‌کنند. در صورتی که دو کاشی با مقدار برابر، با یکدیگر برخورد داشته باشند، مقدار آن دو جمع شده و مقدار حاصل به کاشی‌ای که تلاقی در آن اتفاق افتاده است، داده می‌شود و کاشی دیگر در صورتی که کاشی‌ای حاوی مقدار جای آن را نگرفته باشد، مقداری نخواهد داشت. همچنین با اعمال هر حرکت یک کاشی از کاشی‌های خالی موجود در کنارهٔ مربع به طور تصادفی، مقداردهی می‌شوند؛ به گونه‌ای که مقدار ۲ با احتمال ۹۰ درصد و مقدار ۴ با احتمال ۱۰ درصد در کاشی خالی قرار خواهد گرفت. در پایان درصورت نبود هیچ کاشی خالی و نیز عدم امکان برای قرارگیری دو کاشی با مقدار برابر بر روی یکدیگر و یا رسیدن به مقدار ۲۰۴۸، بازی خاتمه می‌یابد.



(ب) (آ)



(ج)

تصویر ۱: (آ) شروع بازی (ب) اواسط بازی (ج) پایان بازی (برد ــ رسیدن به ۲۰۴۸)

۲ پکیج‌های مورد استفاده و ساختار برنامه

پکیج‌های استفاده شده عبارتند از: tkinter، numpy، matplotlib

همچنین فایل‌های زیر جهت پیاده‌سازی بازی و نیز پیاده‌سازی الگوریتم‌های مورد نظر توسط شما، از پیش پیاده‌سازی شده‌اند. game_gui، game_functions، game_ai، game_console

جهت مشاهدهٔ توضیحات تکمیلی در رابطه با این پکیج‌ها و توابع موجود در هر یک از فایل‌های عنوان شده، به بخش پیوست‌ها مراجعه کنید.

توجه: هیچ یک از فایل‌های شرح داده شده در بالا را تغییر ندهید.

فایل‌های expectimax، mcts و evaluation به منظور پیاده‌سازی توسط شما قرار گرفته است.

۳ پیاده‌سازی

۱.۳ آماده‌سازی محیط برنامه

در ابتدا با استفاده از دستور زیر، تمام پکیج‌های مورد نیاز برای تمرین را نصب کنید.

```
pip install -r requirements.txt
```

تصویر ۲: قطعه کد جهت نصب پکیج‌های مورد نیاز برای برنامه

۲.۳ پیاده‌سازی تابع ارزیابی[۱]

برای تعریف یک تابع ارزیابی، نیاز است که در ابتدا تعدادی ویژگی[۲] را تعریف کرد. هر یک از این ویژگی‌ها نشان‌دهندهٔ مورد خاصی است که باعث رسیدن به امتیاز بالا می‌شود. از طرفی به منظور تعیین اهمیت هر یک از این ویژگی‌ها، به هر کدام از آن‌ها وزن نسبت داده می‌شود؛ به طوری که وزن بالاتر برای هر ویژگی، به معنای پر اهمیت‌تر بودن و در نتیجه تاثیرگذارتر بودن آن ویژگی برای عامل هوشمند است.

به منظور پیاده‌سازی این تابع نیاز است که تابع evaluate_state از فایل evaluation.py را کامل کنید. این تابع صفحهٔ بازی را به عنوان ورودی دریافت می‌کند و پس از محاسبهٔ امتیاز صفحهٔ ورودی، آن را به عنوان خروجی باز می‌گرداند.

۱. دو ویژگی زیر را در این تابع پیاده‌سازی کنید.

[۱] Function Evaluation
[۲] Feature

آ. یکنوایی: عامل هوشمند را تشویق می‌کند تا یک الگوی یکنواخت را در کاشی ها حفظ کند؛ به این صورت که کاشی با بالاترین ارزش را در یک گوشه نگه دارد و چیدمان کاشی های دیگر به ترتیب نزولی در امتداد لبه‌ها شکل بگیرد.

ب. همواری: تفاوت بین کاشی های مجاور را به حداقل می‌رساند و ادغام آن‌ها را آسان تر کنید.

۲. به هر یک از ویژگی‌ها وزنی اختصاص دهید.

۳. در پایان به منظور ترکیب ویژگی‌ها و وزن‌هایشان تابع خطی زیر را تعریف کنید و مقدار حاصل از آن را به عنوان خروجی تابع بازگردانید.

$$Score = F_1.w_1 + F_2.w_2 + \ldots + F_n.w_n; \quad w_i \in \mathbb{R}\ ,\ F_i \in \mathbb{R} \qquad (۱)$$

۳.۳ پیاده‌سازی الگوریتم expectimax

در این الگوریتم، با درختی با دو نوع گره[۳] مواجه هستیم؛ گرهٔ بیشینه‌ساز[۴] و گرهٔ شانس[۵]. ریشهٔ درخت در این الگوریتم را گرهٔ بیشینه‌ساز تشکیل می‌دهد. گرهٔ بیشینه‌ساز گره‌ای است که منتسب به بازیکن می‌باشد که نیاز دارد امتیاز خود را بیشینه کند. در حالی که گرهٔ شانس مربوط به محیط می‌باشد. به عبارتی گره‌ای است که به واسطهٔ آن، محیط با احتمال‌های گوناگون به حالات دیگر گذار می‌کند.
برای بدست آوردن امتیاز هر یک از گره‌ها به صورت زیر عمل می‌کنیم:

۱. گرهٔ بیشینه‌ساز: امتیاز فرزندی با بیشترین امتیاز را به این گره اختصاص می‌دهیم.

۲. گرهٔ شانس: نیاز است که میانگین وزن‌دار امتیاز فرزندان این گره را بدست آوریم و امتیاز حاصل را به این گره اختصاص دهیم. منظور از میانگین وزن‌دار، حاصل ضرب امتیاز هر یک از گره‌های فرزند در احتمال رسیدن به آن گره است.

به منظور پیاده‌سازی این الگوریتم، شما بایستی کلاس Expectimax در فایل expectimax.py را کامل کنید.

۱. تابع get_depth را به طریقی پیاده‌سازی کنید که عمق جست‌وجو را با توجه به تعداد حرکاتی که تا کنون عامل هوشمند طی کرده است، برگرداند؛ به طوری که در ابتدا مقدار عمق بازگردانده شده، پایین بوده و برابر با DEPTH_BASE_PARAM باشد و سپس به ازای هر افزایش در تعداد حرکات اتخاذ شده، به اندازهٔ مطلوب، عمق مورد جست‌وجو به میزان دلخواه (SCALAR_PARAM) افزایش پیدا کند.

۲. حالت پایه برای الگوریتم expectimax را در تابع epectimax بنویسید؛ به این صورت که بررسی کنید آیا عمق مورد جست‌وجو به پایان رسیده است یا خیر، و همچنین آیا به استیت نهایی رسیده‌ایم یا خیر.

---
[۳] Node
[۴] Node Maximizer
[۵] Node Chance

۳. با بررسی کردن متغیر turn تعیین کنید که آیا نوبت گرهٔ بیشینه‌ساز است یا نوبت گرهٔ شانس. بر حسب نتیجهٔ بدست آمده، تابع maximizer_node یا chance_node را صدا بزنید و خروجی آن‌ها را به عنوان خروجی تابع بازگردانید.

۴. در تابع maximizer_node، از میان تمام اکشن‌ها، اکشنی را به عنوان خروجی برگردانید که باعث رسیدن به صفحه‌ای از بازی شود که بیشترین امتیاز را رقم بزند. برای این منظور، در ابتدا لیست اکشن‌ها را دریافت کنید. سپس هر کدام از آن‌ها را بر روی محیط بازی اعمال کرده و با دریافت محیط جدید بازی، برای باری دیگر تابع expectimax را صدا بزنید. امتیاز بازگردانده شده از این تابع را به مجموع امتیازاتی که تا کنون از اعمال این اکشن به دست آورده‌اید، بیفزایید. در پایان، اکشنی که بیشترین امتیاز را حاصل کرده است را به همراه امتیاز منتسب به آن بازگردانید.

۵. در تابع chance_node، میانگین وزن دار فرزندان را محاسبه کنید. برای این منظور، در ابتدا تمام خانه‌های خالی صفحه را پیدا کنید و سپس هر خانهٔ خالی را با استفاده از مقادیر ۲ و ۴، مقداردهی کنید. در مرحلهٔ بعد با صدا زدن تابع expectimax، امتیاز را در صورت مقداردهی خانه با مقدار مورد نظر، بدست آورید. سپس امتیاز بدست آمده را در احتمال مقداردهی خانه با هر یک از مقادیر ۲ و ۴ ضرب کرده و حاصل را به مجموع امتیازاتی که تا کنون بدست آمده است، بیفزایید. در پایان خروجی را به صورت تاپلی متشکل از مجموع امتیازات به همراه مقدار None به عنوان اکشن، برگردانید.

توجه: مقدار اولیه برای DEPTH_BASE_PARAM حداکثر برابر با ۳ بوده و حداقل مقدار برای SCALAR_PARAM برابر با ۴۰۰ است.

۴.۳ پیاده‌سازی الگوریتم MCTS

این الگوریتم ورژن‌های مختلفی دارد که دو مورد از آن‌ها را برای این بازی پیاده‌سازی می‌کنیم. مورد اول، ورژن صفر این الگوریتم است. در این نسخه، به ازای هر یک از اکشن‌ها، یک بار بازی را به تعداد مشخص و به صورت تصادفی بازی می‌کنیم. به این عملیات شبیه‌سازی گفته می‌شود. عملیات شبیه‌سازی را برای هر اکشن به تعداد مشخص انجام می‌دهیم و تمام مقادیر بدست آمده را با همدیگر جمع می‌زنیم. در پایان حرکتی با بیشترین امتیاز را به همراه امتیاز آن باز می‌گردانیم.

مورد دوم، ورژن ۲ این الگوریتم است. در این نسخه تمام عملیات مشابه نخسهٔ اول است با این تفاوت که عملیات انتخاب اکشن جدید، به صورت هدفمند انجام می‌شود. به عبارتی عملیات شبیه‌سازی برای تمامی اکشن‌ها به تعداد برابر صورت نمی‌گیرد؛ بلکه به واسطهٔ امتیاز UCB، اکشن‌ها انتخاب می‌شوند و در نتیجه ممکن است تعداد دفعات شبیه‌سازی برای هر اکشن به صورت نامساوی حاصل شود.

به منظور پیاده‌سازی این الگوریتم، بایستی کلاس MCTS موجود در فایل mcts.py را کامل کنید.

۱. تابع get_serach_params را به گونه‌ای پیاده‌سازی کنید که حداکثر تعداد حرکات در عملیات شبیه‌سازی و تعداد دفعات انجام شبیه‌سازی را به صورتی محاسبه کند که در ابتدا

برابر با یک مقدار پایه باشد و سپس به ازای هر افزایش در تعداد حرکات صورت گرفته، مقدار این دو متغیر بیشتر شود. برای این منظور با توجه به اینکه حالت الگوریتم بر روی کدام نسخه است، مقدار پایه برای این دو مورد را انتخاب کنید. در پایان این دو مقدار را به عنوان خروجی بازگردانید.

۲. عملیات شبیه‌سازی را در تابع simulate_move پیاده‌سازی کنید. به این طریق که برای صفحهٔ وارد شده به تابع، به ازای عمق ذخیره شده در متغیر search_depth، عملیات انتخاب اکشن تصادفی را انتخاب کرده و امتیاز صفحهٔ بدست آمده را به مجموع امتیازات بدست آمده در طول شبیه‌سازی، بیفزایید. در پایان مجموع امتیازات حاصل را به عنوان خروجی بازگردانید.

۳. در تابع ucb، تابع زیر را که به عنوان UCB1 شناخته می‌شود پیاده‌سازی کنید. این تابع، مجموع تعداد دفعات انجام عملیات شبیه‌سازی را به همراه آرایه‌ای متشکل از تاپلی با سه عضو (اکشن، امتیاز آن اکشن، تعداد دفعات اتخاذ اکشن) به عنوان ورودی دریافت می‌کند. سپس امتیاز UCB هر چهار عمل به عنوان خروجی بازگردانده می‌شود. در صورتی که تا کنون عملی وجود داشته باشد که انتخاب نشده است، مقدار منتسب به آن اکشن را مثبت بی‌نهایت در نظر بگیرید.

$$UCB1(n) = \frac{U(n)}{N(n)} + C \times \sqrt{\frac{\log N(Parent(n))}{N(n)}}$$

برای محاسبهٔ جذر و لگارتیم، از توابع آماده log و sqrt از پکیج numpy استفاده کنید.

۴. در تابع mcts_v0، ورژن صفر الگوریتم را پیاده‌سازی کنید. در این تابع به ازای تمام اکشن‌ها، عملیات شبیه‌سازی را به عمق search_depth و به تعداد ذخیره شده در متغیر total_moves انجام دهید. در پایان اکشن با بیشترین امتیاز را بازگردانید.

۵. در تابع mcts_v2، ورژن دو الگوریتم را پیاده‌سازی کنید. برای این منظور نیاز است که لیستی متشکل از تاپل‌های سه عضوی (اکشن، امتیاز آن اکشن، تعداد دفعات اتخاذ اکشن) را به ازای هر اکشن ایجاد کنید. سپس به تعداد ذخیره شده در متغیر total_moves، عملیات شبیه‌سازی را به عمق search_depth و برای اکشنی با بیشترین امتیاز UCB، انجام دهید. برای یافتن بیشترین امتیاز UCB از میان امتیازات دریافتی از تابع ucb، از تابع argmax از پکیج numpy استفاده کنید. سپس اکشن مورد نظر را بر روی محیط بازی اعمال کنید. در صورت موفقیت در اعمال حرکت، مقدار امتیاز صفحهٔ بدست آمده را به مجموع امتیازات بیفزایید و در غیر این صورت مقدار امتیاز آن را به مقدار دلخواه کاهش دهید. همچنین نیاز است که تعداد دفعات اتخاذ اکشن را نیز یک واحد اضافه کنید. در پایان اکشنی با بیشترین امتیاز را بازگردانید.

توجه: مقدار اولیه برای هر دو متغیر SD_SCALE_PARAM و TM_SCALE_PARAM حداکثر برابر با ۸ بوده و حداقل مقدار برای $SCALAR\_PARAM$ برابر با ۲۰۰ است. همچنین

مقدار اولیه برای $UCB\_TM\_SCALE\_PARAM$ حداکثر برابر با ۲۰ و حداکثر مقدار برای $UCB\_SD\_SCALE\_PARAM$ برابر با ۱۰ است. همچنین حداقل مقدار برای متغیر UCB_SCALAR_PARAM برابر با ۳۰۰ است.

توجه: پس از اعمال اکشن مورد نظر و اطمینان از انجام شدن عملیات، نیاز است که تابع add_new_tiles را صدا بزنید تا محیط جدید بازی را دریافت کنید.

توجه: تنها توابع گفته شده را تغییر دهید و در هیچ یک از توابع یا فایل‌های دیگر تغییری اعمال نکنید.

توجه: ورودی و خروجی هیچ یک از توابع را تغییر نداده و توابع را به همان صورت که توضیح داده شده است پیاده‌سازی کنید.

توجه: برای هر یک از الگوریتم‌های گفته شده، محدودیت‌های اعلام شده برای عمق مورد نظر برای جست‌وجو را حتما لحاظ کنید.

۴ پرسش‌ها

به پرسش‌های زیر پاسخ دهید و به صورت یک فایل پی‌دی‌اف به همراه کدهای نوشته شده، در سایت بارگذاری کنید.

۱. به چه منظور، دو الگوریتم MCTS و Expectimax برای حل این بازی انتخاب شده‌اند؟

۲. چرا اجرای الگوریتم را به عمقی مشخص، محدود کردیم؟

۵ پیوست

۱.۵ پکیج‌های موردنیاز

پکیج‌های مورد استفاده و فایل‌های آماده پکیج‌های استفاده شده به همراه دلیل استفاده از آن‌ها در زیر لیست شده‌اند:

- tkinter ایجاد رابط کاربری
- numpy تسریع و تسهیل محاسبات
- matplotlib رسم نمودار جهت مشاهده و بررسی نحوهٔ عملکرد الگوریتم نوشته شده

برای نصب هر یک از پکیج‌ها می‌توان از دستور زیر استفاده کرد. (به احتمال زیاد در صورت نصب کردن python، پکیج tkinter نیز به همراه آن بر روی سیستمتان نصب شده باشد و نیازی به نصب مجدد آن وجود نداشته باشد.)

۲.۵ فایل‌های آماده‌شده

فایل‌های زیر نیز از پیش آماده شده‌اند:

- game_gui: شامل توابع و کلاس‌های لازم برای ساخت محیط گرافیکی بازی است. با اجرای این فایل می‌توانید محیط بازی را تماشا کنید. با فشردن کلیدهای زیر نیز می‌توانید عملیات معادل نوشته شده برای آن‌ها را در محیط بازی اعمال کنید:
  - W: حرکت صفحه به سمت بالا
  - A: حرکت صفحه به سمت چپ
  - D: حرکت صفحه به سمت راست
  - S: حرکت صفحه به سمت پایین
  - E: اجرای الگوریتم Expectimax — دیدن نتیجهٔ این عمل، نیاز به پیاده سازی شما را دارد.
  - M: اجرای الگوریتم MCTS v0 — دیدن نتیجهٔ این عمل، نیاز به پیاده سازی شما را دارد.

- U: اجرای الگوریتم MCTS v2 – دیدن نتیجهٔ این عمل، نیاز به پیاده سازی شما را دارد.

- game_functions: شامل توابع لازم برای ساخت بازی و نیز نوشتن الگوریتم‌ها است. توابع مورد نیاز شما از این فایل:

  - initialize_game: محیط بازی را ایجاد و صفحهٔ ابتدایی بازی را باز می‌گرداند.
  - move: صفحه را در جهت درخواست شده حرکت می‌دهد. به منظور مشخص کردن جهت، باید پارامتر direction را به طریق زیر مقداردهی کنید:
    - ۰: حرکت به سمت بالا
    - ۱: حرکت به سمت راست
    - ۲: حرکت به سمت پایین
    - ۳: حرکت به سمت چپ
  - get_moves: تمام چهار عملیات مورد نظر را به صورت لیستی از توابع باز می‌گرداند.
  - get_all_possible_moves: تمام حرکت‌های قابل اعمال بر روی صفحه، به همراه حالت جدید صفحه که از اعمال آن‌ها بر روی صفحه به وجود می‌آید را بر می‌گرداند.
  - get_empty_cells: کاشی‌های خالی موجود در صفحه را باز می‌گرداند.
  - random_move: در صورتی که همچنان عملی را بتوان بر روی صفحه انجام داد، یکی از این اعمال به صورت تصادفی انتخاب شده و بر روی صفحه انجام می‌شود.
  - add_new_tile: در صورت وجود کاشی خالی در صفحه، یک کاشی را به صورت تصادفی به مقدار ۲ یا ۴ مقداردهی کند.
  - check_for_win: بررسی می‌کند که آیا در صفحه مقدار ۲۰۴۸ وجود دارد یا خیر.
  - check_for_loss: بررسی می‌کند که آیا دیگر حرکتی وجود دارد که بتوان استفاده کرد یا خیر.
  - terminal_state: بررسی می‌کند که آیا بازی به اتمام رسیده است یا خیر.
  - within_bounds: برای بررسی آن است که آیا مختصات طول و عرض وارد شده برای یک کاشی در محدودهٔ طول و عرض صفحهٔ بازی صدق می‌کند یا خیر.

همچنین تعدادی متغیر در این فایل تعریف شده است که می‌توانید در صورت لزوم از آن‌ها استفاده نمایید.

- game_ai: در این فایل تابع ai_move قرار گرفته است که با صدا زدن آن و پاس دادن نوع الگوریتم مورد نظر می‌توانیم الگوریتم را بر روی بازی اجرا کنیم. agent_name برای این منظور در پارامترهای این تابع قرار گرفته شده است. همانطور که پیش‌تر ذکر شد، سه الگوریتم زیر برای حل کردن بازی، مورد استفاده قرار می‌گیرند:

- expectimax: الگوریتم Expectimax
- mcts: الگوریتم MCTS بدون استفاده از UCB (ورژن صفر الگوریتم MCTS)
- ucb: الگوریتم MCTS با استفاده از UCB (ورژن ۲ الگوریتم MCTS)

- game_console: در این فایل، دو تابع جهت اجرای بازی در کنسول و نیز بررسی عملکرد الگوریتم‌های نوشته شده، قرار دارند.
  - ai_play: برای اجرای بازی به تعداد یک بار با استفاده از الگوریتم مورد نظر است.
  - ai_plot: برای اجرای بازی به تعداد دلخواه با استفاده از الگوریتم مورد نظر و نیز کشیدن نمودارهای بازی‌های اجرا شده است. همچنین اطلاعاتی در رابطه با میانگین امتیاز به دست آمده و مدت زمان صرف شده برای اجرای الگوریتم را به شما می‌دهد.

توجه: هیچ یک از فایل‌های شرح داده شده در بالا را تغییر ندهید. در صورت نیاز به توابع جدید، یک فایل تحت عنوان utils.py بسازید و توابع خود را داخل آن بنویسید.

قوانین:

۱. در این تمرین می‌توانید با یک نفر دیگر نیز همفکری داشته باشید. نیاز است که نام این فرد را در گزارش تمرین قید کنید.

۲. نمره شما بر اساس گزارش راه طی شده برای حل مسئله و پاسخ صحیح خواهد بود لذا از هرگونه اطناب در گزارش پرهیز و به موارد خواسته‌شده به صورت کامل پاسخ دهید.

۳. برای تمرین از شما ارائه شفاهی گرفته خواهد شد بنابراین تسلط لازم را بر کدی که پیاده می‌کنید داشته باشید.

۴. در صورتی مشاهده شباهت غیرعادی بین پیاده سازی‌ها نمره طرفین طبق قوانین درس محاسبه خواهد شد.

۵. برای تحویل تمرین یک فایل zip شامل گزارش حل سوالات، با نام زیر در سامانه LMS بارگذاری کنید.

[HW3_ID_NAME1_ID_NAME2]