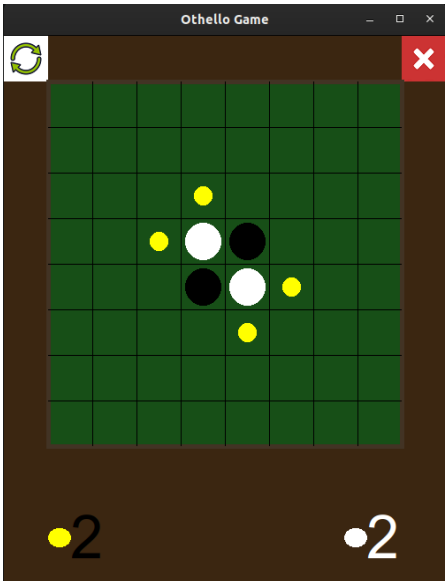
شهریار مربی | ۹۷۲۴۳۰۶۴
نرگس قاسمی | ۹۷۲۴۳۰۵۰

گزارش پروژه ی هوش مصنوعی :

بازی Othello

فاز اول :

فاز یک را با کتابخانه ی tkinter و پکیج pillow به صورت گرافیکی انجام دادیم. برای هر بازیکن حرکت های ممکن او روی صفحه (با رنگ زرد) می شود.



فاز دوم:

Minimax algorithm with Alpha Beta pruning

درخت مینیمکس را به این شکل می سازیم هر راس نشان دهنده ی یک حالت board است و بچه های آن board های ممکن پس از انجام یک حرکت از طرف شخصی که نوبت اوست بدست می آیند.

شهریار مربی | ۹۷۲۴۳۰۶۴
نرگس قاسمی | ۹۷۲۴۳۰۵۰

اعمال ساده سازی با معیار های تنوع:

علاوه بر هرس آلفا بتا با استفاده از هایپر پارامتر BRANCHING_FACTOR بیشترین تعداد بچه های هر راس را محدود می کنیم. برای این هرس کردن ثانویه هدفمان این است که زیر مجموعه ای از فرزندان یک راس با بیشترین تنوع انتخاب شوند. برای این کار ۵ معیار در نظر گرفتیم که هر کدام مربوط به تنوع یکی از ویژگی های خانه ای است که قرار است مهره در آن قرار بگیرد (زیرا هر کدام از فرزندان از انجام یک action از وضعیت فعلی حاصل شده اند. در نتیجه تفاوت action های انتخابی باعث تفاوت در مرحله ی بعدی بازی می شود) و به ازای هر کدام متناسب با اختلاف آن ویژگی هر فرزند از دیگر فرزندان به او امتیاز دادیم. و امتیاز هر راس به ازای معیار های مختلف را با یکدیگر جمع کردیم سپس به تعداد BRANCHING_FACTOR فرزندهایی را که بیشترین امتیاز را داشتند انتخاب می کنیم و بقیه هرس می شود.

به ازای یک معیار مشخص مانند x اگر مقدار نسبت داده شده به فرزند j-ام برابر $children[j].x$ باشد. امتیاز فرزندان بدین شکل محاسبه می گردد:

$$Let\ sorted\_children\ be\ the\ sorted\ form\ of\ children\ based\ on\ their\ x\ attribute.$$

$$For\ each\ index\ i\ in\ sorted\_children:\ sorted\_children[i].score\ =$$

$$(sorted\_children[i].x\ -\ sorted\_children[i-1].x)\ +\ (sorted\_children[i+1].x\ -\ sorted\_children[i].x)$$

زیرا بیشتر بودن اختلاف ویژگی x بین فرزند i-ام و فرزند قبلی و بعدی او در این آرایه به این معنی است که در نزدیکی ویژگی x فرزند i-ام فردی وجود ندارد و i با تعداد بیشتری از دیگر فرزندان تفاوت دارد.

در واقع این گونه انتخاب کردن باعث می شود که فرزندان از همسایگی های متفاوت انتخاب شوند و از هر همسایگی تعداد کمتری انتخاب شوند.

شهریار مربی | ۹۷۲۴۳۰۶۴
نرگس قاسمی | ۹۷۲۴۳۰۵۰

معیار ها:

1. $action.x$
2. $action.y$
3. $action.x + action.y$
4. $action.x - action.y$
5. $a\ random\ number$

تابع هیوریستیک:

به هر وضعیت بر اساس معیار های مختلف مربوط به تعداد و موقعیت مهره های هر بازیکن مقداری نسبت داده می شود که به شرح زیر است:

۱- قابلیت تحرک(Mobility) :اختلاف نسبی تعداد حرکت های ممکن برای هر بازیکن

۲- اختلاف مجموع وزن های نسبت داده شده به خانه های اشغال شده توسط مهره های هر فرد

وزن خانه های گوشه ای از تمام خانه ها بیشتر است به دلیل اینکه اگر فردی این خانه ها را بگیرد آن ها را از دست نمی دهد و باعث میشود که اگر همان فرد در خانه های کناری board مهره ای گذاشت تمام خانه های این بین را بگیرد.

وزن خانه های مجاور خانه های گوشه ای منفی هستند زیرا یک حرکت احتمالی با وزن زیاد برای حریف ایجاد میکند.

دیگر خانه های کناری جدول دارای وزن نسبتا بالایی هستند زیرا اگر بازیکن در این خانه ها مهره قرار دهد و در گوشه هم مهره ای داشته باشد، تمام خانه های بین را می تواند بدست بیاورد.

شهریار مربی | ۹۷۲۴۳۰۶۴
نرگس قاسمی | ۹۷۲۴۳۰۵۰

جدول وزن ها:

20	-3	11	8	8	11	-3	20
-3	-7	-4	1	1	-4	-7	-3
11	-4	2	2	2	2	-4	11
8	1	2	-3	-3	2	1	8
8	1	2	-3	-3	2	1	8
11	-4	2	2	2	2	-4	11
-3	-7	-4	1	1	-4	-7	-3
20	-3	11	8	8	11	-3	20

۳- اختلاف تعداد خانه های گوشه ای تسخیر شده توسط هر بازیکن

٤- اختلاف تعداد خانه های مجاور گوشه ی تسخیر شده توسط هر بازیکن(تاثیر معکوس)

۵- اختلاف نسبی تعداد مهره های هر بازیکن

٦- اختلاف مجموع تعداد خانه های خالی مجاور هر خانه ی اشغال شده توسط هر بازیکن زیرا احتمال از دست دادن هر خانه متناسب با تعداد خانه های خالی اطراف آن است. (تاثیر معکوس)

شهریار مربی | ۹۷۲۴۳۰۶۴
نرگس قاسمی | ۹۷۲۴۳۰۵۰

فاز سوم:

یک جمعیت اولیه با ۲۰ کروموزوم که هر کروموزوم شامل ژنها هستند که به در ابتدا به صورت کاملا رندوم مقدار دهی شده اند ایجاد میکنیم.

سپس یک تورنمنت بین آنها برگزار میکنیم که در هر تورنمنت اگر بازی مساوی نشد به برنده یک امتیاز اضافه میکنیم.

در مرحله ی بعد بین هر دو عضو با احتمال بازترکیبی ۰/۸، crossover انجام میشود . به این صورت که هر ژن از میانگین وزن دار والدین بدست می آید به طوری که وزن ژن هر والد برابر با نسبت امتیاز او به مجموع امتیاز هر دو والد است.

سپس عمل جهش با احتمال ۰/۱ انجام میشود که اگر باید جهش انجام شود، یک ژن از کروموزوم به صورت رندوم انتخاب شود و مقدارش با عدد رندومی در بازه ی ۰ تا ۱۰۰۰ میانگین گرفته شود.

برای انتخاب اعضای نسل بعدی ، والدین را بر حسب امتیازشان مرتب میکنیم و یک سوم کروموزوم ها با fitness بیشتر را برمیداریم . برای انتخاب بقیه اعضای نسل بعدی از بین فرزندان ایجاد شده به صورت رندوم دو سوم تعداد جمعیت را برمیداریم.

سپس بین این افراد تورنمنت برگزار میکنیم و برای هر کدام fitness را بدست می آوریم.

به تعداد ۹ epoch این کار را تکرار میکنیم و در نهایت بهترین AI ای که با جمعیت اولیه بدست آمده را که ماکسیمم fitness را دارد برمیگردانیم.

یکبار این الگوریتم ژنتیک را برای بدست آوردن نحوه ی هرس کردن یعنی وزنهای معیار های تنوع action ها ی AI و یکبار برای بدست آوردن وزن های تابع هیوریستیک AI اجرا شد.

*برای انجام سریع تر تورنمنت ها از پکیج multiprocessing در پایتون استفاده کردیم تا هر مسابقه به عنوان یک subprocess مجزا اجرا شود زیرا مسابقات بین اعضای یک نسل از هم مجزا هستند و سپس از نتیجه ی این مسابقات برای تولید نسل بعدی استفاده می شود.