رویکردی برای پیاده‌سازی بازی اتللو مبتی بر الگوریتم‌های تکاملی

محمدمهدی تیموریان و هانیه ملائی

دانشگاه شهید بهشتی

دی 99

چکیده

در طی این تحقیقات رویکردی برای پیاده‌سازی بازی معروف اتللو مبتی بر الگوریتم‌های تکاملی و برنامه‌نویسی ژنتیک ارائه شده است. پیاده‌سازی طی سه مرحله کلی منطق بازی، درخت مینیماکس و تمرین مدل بر اساس الگوریتم های تکاملی صورت گرفته است.

مقدمه

در هوش محاسباتی، یک الگوریتم تکاملی به عنوان زیرمجموعه‌ای از محاسبات تکاملی، یک الگوریتم عمومی جمعیت پایه و فرامکاشفه‌ای با هدف بهینه‌سازی می باشد. الگوریتم ژنتیک نوع خاصی از الگوریتم‌های تکاملی است که از تکنیک‌های زیست‌شناسی تکامل مانند وراثت، جهش و انتخاب برای یافتن راه‌حل بهینه‌سازی و یا جستجو استفاده می‌کند. الگوریتم ژنتیک از تکامل ژنتیکی به عنوان یک الگوی حل مسئله استفاده می‌کند. مسئله‌ای که باید حل شود دارای ورودی‌هایی می‌باشد که طی یک فرایند الگوبرداری شده از تکامل ژنتیکی به راه‌حل‌ها تبدیل می‌شود؛ سپس راه حل‌ها به عنوان کاندیداها توسط تابع ارزیاب مورد ارزیابی قرار می‌گیرند و چنانچه شرط خروج مسئله فراهم شده باشد الگوریتم خاتمه می‌یابد. به‌طور کلی یک الگوریتم مبتنی بر تکرار است که اغلب بخش‌های آن به صورت فرایندهای تصادفی انتخاب می‌شوند که این الگوریتم‌ها از بخش‌های تابع برازش، نمایش، انتخاب و تغییر تشکیل می‌شوند.

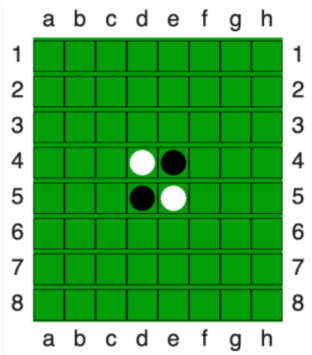
پیاده‌سازی

منطق بازی

بازی اتللو یک صفحه مربع شکل با 64 خانه می باشد. خانه‌ها توسط دو بازیکن با مهره روشن برای یک بازیکن و مهره تیره برای بازیکن دیگر، طبق قوانین خاصی پر میشوند. در انتها هر بازیکن که تعداد مهره‌های بیشتری درون صفحه داشته باشد، برنده است.

شروع بازی

ابتدای بازی، چهار مهره مطابق شکل در وسط صفحه به صورت ضربدری قرار می‌گیرند.



مهره تیره بازی را آغاز می‌کند. هر یک از دو بازیکن به نوبت یک حرکت انجام می‌دهند. هر بازیکن باید مهره خود را جایی قرار دهد که یک یا چند مهره حریف را محاصره کند. انجام حرکت به معنی گذاشتن یک مهره (از رنگ خود) در صفحه و محصور کردن یک یا چند مهره حریف در یک یا چند راستا است. در نتیجه، مهره‌های محاصره شده برگردانده می‌شوند و به رنگ مهره بازیکن در می‌آیند.

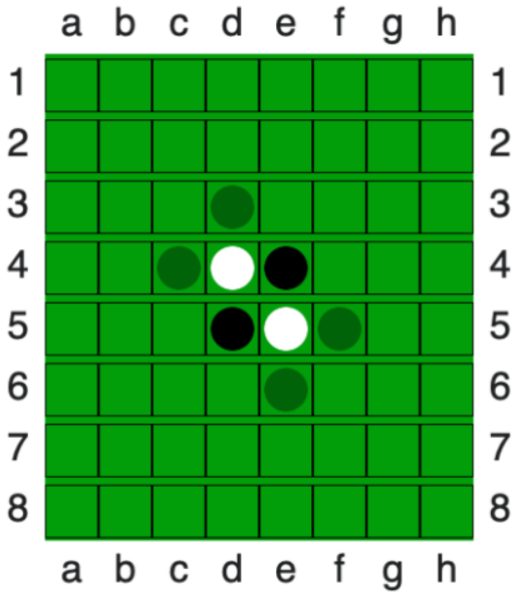
قوانین بازی

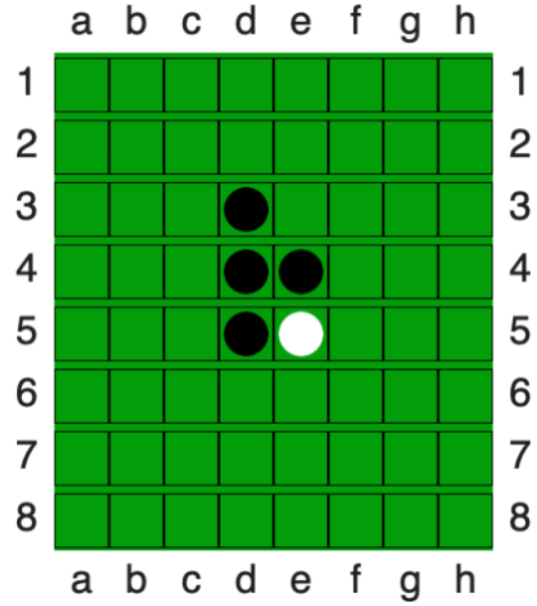
- اولین نوبت بازی برای مهره تیره می‌باشد.

- مهره جدید را فقط در محلی می‌توان قرار داد که مهره‌ای از حریف محاصره شود. به عبارتی قرار دادن یک مهره درون صفحه به صورت آزاد غیرمجاز است. و همه مهره‌ها در مجاورت یکدیگر هستند.
- خط محاصره می‌تواند افقی، عمودی، مورب یا ترکیب از موارد اشاره شده باشد.
- در هر نوبت بازیکن باید حتما مهره‌ای درون صفحه قرار دهد و نمیتواند نوبت خود را به حریف بدهد.
- در صورتی که یکی از بازیکن‌ها در نوبت خود، خانه‌ای برای انتخاب نداشته باشد، نوبت به بازیکن حریف می‌رسد.

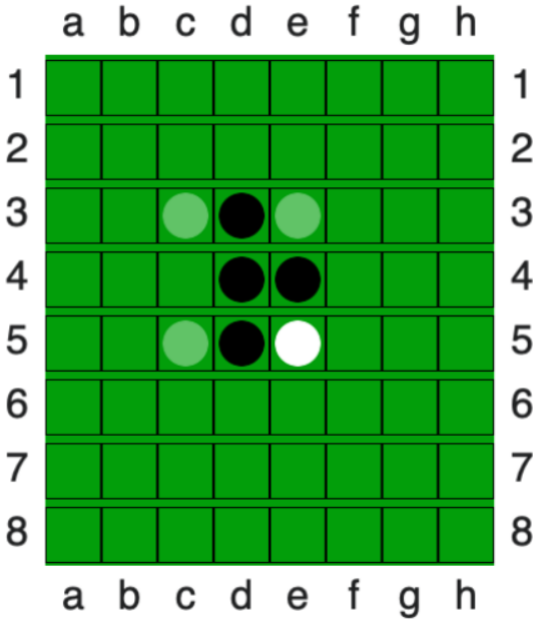
نمونه قرار گرفتن مهره ها

اولین حرکت توسط مهره تیره انجام می‌شود. خانه های 3d، 4c، 5f و 6e مهره‌های روشن را محاصره میکنند و بازیکن با مهره تیره میتواند از بین این حرکت‌ها انتخاب کند.



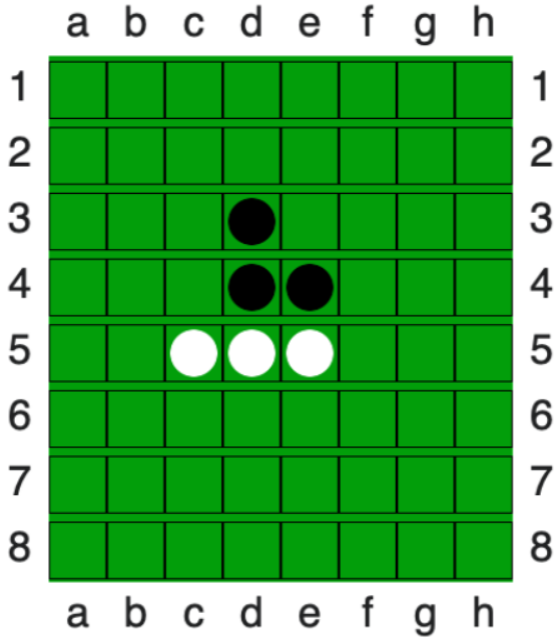


بازیکن با مهره تیره خانه 3d را انتخاب میکند. به عنوان نتیجه، خانه 4d که مهره روشن درون آن بین مهره‌های تیره محاصره شده بود، برگردانده می‌شود.



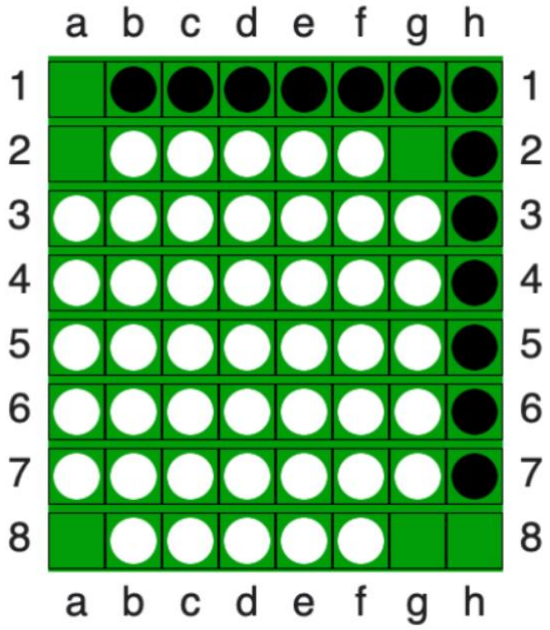
نوبت بعدی به مهره روشن میرسد. همانند مرحله قبل حرکت‌های قابل انتخاب، در صفحه مشخص شده است.

در نهایت صفحه بازی بعد از انتخاب بازیکن با مهره روشن به حالت روبه‌رو تبدیل می‌شود. بازی به همین ترتیب ادامه می‌یابد تا به حالت پایانی بازی رسیده شود.



حالت پایانی

وقتی تمام صفحه پر شود و یا هیچ کدام از دو طرف حرکتی نداشته باشند، بازی به پایان می‌رسد. در شکل زیر نمونه‌ای از وضعیتی که هیچ کدام از بازیکن‌ها حرکتی نداشته باشند، قابل مشاهده است.



مدل و درخت مینیماکس

مدل با استفاده از درخت مینیماکس و همچنین هرس آلفا و بتا، تا مرحله‌ای از بازی را ادامه می‌دهد و زمانی که به یک عمق خاص رسید، با استفاده از یک تابع یوتیلیتی به آن حالت از بازی امتیاز می‌دهد. در نهایت حرکتی را انتخاب می‌کند که به امیتاز بیشتر و حالت بهتر برسد.

به صورت کلی، میانگین تعداد حرکات مجاز در هر مرحله بالا می باشد. به طوری که با یک پردازنده معمولی، درخت مینیماکس به همراه هرس آلفا و بتا، نهایتا تا عمق 10 در زمان قابل قبولی پیش می‌رود. در ادامه با ارائه رویکردی مکاشفه‌ای عرض درخت مینیماکس را کاهش می‌دهیم.

کاهش عرض درخت

مدل در نوبت خود، با استفاده از تعدادی ویژگی به هر حرکت مجاز خود امتیازی می‌دهد و به صورت حریصانه، 3 حرکت که امتیاز بیشتری دارند را انتخاب میکند. با این رویکرد عرض درخت به 3 یا کمتر کاهش می‌یابد. حال، با استفاده از یک پردازنده معمولی به عمق 15 دست می‌یابیم. قابل ذکر است که عرض درخت در عمق 0 به 75٪ تعداد حرکات مجاز کاهش می‌یابد و انتخاب 3 حرکت بهتر برای عمق بیشتر از 0 می باشد.

ویژگی‌های کاهش عرض درخت

بازه همه ویژگی‌ها بین [1,1-] می باشد و هر کدام ضریبی دارد که با استفاده از آن به ویژگی وزن میدهیم. در مرحله تکامل راجع به تعیین مقدار ضرایب توضیح می‌دهیم. مجموع حاصل‌ضرب ویژگی‌ها در ضریب‌شان به عنوان امتیاز یک حرکت قلمداد می‌شود. در ادامه این ویژگی‌ها را بررسی می‌کنیم.

- گوشه بودن: در صورتی که حرکت بازیکن، یکی از چهار خانه گوشه‌ها باشد، مقدار 1 و در غیر این صورت مقدار 0 در نظر گرفته می‌شود.
- حرکت در وسط: در ابتدای بازی به حرکات نزدیک به وسط صفحه امتیازی بیشتری میدهد. برای 30 حرکت آخر بازی این ویژگی مقدار 0 برمیگرداند.

$$score = \frac{distance\ to\ closest\ edge}{Max\ distance\ to\ a\ edge}$$

- محاصره کمتر: در ابتدای بازی به حرکاتی که تعداد کمتر و در انتهای بازی به حرکاتی که تعداد مهره‌های بیشتری را محاصره می‌کند امتیاز بیشتری می‌دهد. بیشینه کردن محاصره در 20 حرکت اخر اتفاق می‌افتد.

$$score = \begin{cases} \dfrac{Flips}{Max\ Flips} & L < 20 \\ \dfrac{Max\ Flips - Flips}{Max\ Flips} & L > 20 \end{cases} ;\ left\ moves = L$$

- از دست دادن گوشه: در صورتی که حرکتی باعث دادن یکی از گوشه ها در نوبت بعدی به حریف شود امتیاز -1 و در غیر این صورت امتیاز 0 تعلق می‌گیرد.
- کاهش حرکات مجاز رقیب: در صورتی که بعد از انجام حرکت مورد بررسی، تعداد انتخاب مجاز کمتری در نوبت بعدی به حریف داده شود، امتیاز بیشتری دریافت میکند.

$$score = \frac{Max\ Mobility - Opponent\ Mobility}{Max\ Mobility}$$

- حرکت در حاشیه صفحه: این ویژگی حرکات در حاشیه صفحه بازی را بررسی میکند. امتیاز دهی به صورت زیر می باشد.

موقعیت	وضعیت	امتیاز
کنار یکی از گوشه‌ها	گوشه نزدیک هم رنگ	1
	گوشه نزدیک رنگ مخالف	0
	گوشه نزدیک خالی و سمت دیگر مهره خالی	0
	گوشه نزدیک خالی و سمت دیگر مهره، مهره بازیکن مقابل	-1
در حاشیه ها	قرار گرفتن بین دو مهره بازیکن مقابل	1
	عدم وجود یکی از شرایط بالا	0
در میانه صفحه		0

محاسبه یوتیلیتی

با رسیدن به عمق 15 یا اتمام بازی، به صفحه امتیاز می‌دهیم که آیا با انجام دادن این سلسله از حرکات، موقعیت خوبی داریم؟ مبنای تصمیم درخت مینیماکس نیز بر اساس امتیاز محاسبه شده است.

ویژگی‌های محاسبه یوتیلیتی

بازه همه ویژگی‌ها بین [1,1-] می باشد و هر کدام ضریبی دارد که با استفاده از آن به ویژگی وزن میدهیم. در مرحله تکامل راجع به تعیین مقدار ضرایب توضیح می‌دهیم. مجموع حاصل‌ضرب ویژگی‌ها در ضریب‌شان به عنوان امتیاز آن حالت قلمداد می‌شود.

- اختلاف تعداد مهره‌ها:

$$score = \frac{Current\ Player\ Disks - Opponent\ Disks}{Max\ Disks\ Difference}$$

- تعداد حرکات مجاز بازیکن مقابل: هر چه تعداد حرکات مجاز بازیکن مقابل کمتر باشد، در وضعیت بهتری قرار داریم.

$$score = \frac{Max\ Mobility - Opponent\ Mobility}{Max\ Mobility}$$

- تعداد دیسک‌های مرزی: قبل تر ذکر شد که همه مهره‌ها در کنار یکدیگر هستند. هرچه مهره‌های مدل داخل‌تر باشد یا به عبارت دیگر مرزی نباشند در وضعیت بهتری قرار داریم.

$$score = \frac{Max\ Frontier - Frontier\ Count}{Max\ Frontier}$$

- تعداد گوشه‌های تسخیر شده: هرچه تعداد گوشه های بیشتری تسخیر شده باشد، وضعیت بهتر است. بهترین حالت برای مدل این است که 4 گوشه را تسخیر کند.

$$score = \frac{captured\ corners}{corner\ counts}$$

- تعداد مهره‌های تثبیت شده: مهره تثبیت شده به مهره‌ای گفته می‌شود که در طول بازی دیگر برگردانده نمی‌شود و تا پایان بازی ثابت است. هرچه تعداد این مهره بیشتر باشد، بهتر است.

$$score = \frac{stable\ disks\ count}{64}$$

- اتمام کننده بازی: در اکثر شرایط آخرین حرکت بازی حیاتی و مهم است. در صورتی که مدل آخرین حرکت را انجام دهد امتیاز مثبت برای آن در نظر گرفته میشود.

بهینه‌سازی درخت از لحاظ حافظه و سرعت

در ابتدا برای سادگی پیاده‌سازی، در هر عمق و برای هر حرکت، از صفحه بازی یک کپی ایجاد و تغییرات بر روی آن اعمال شد. راه‌حل جایگزین آن، بازگرداندن تغییرات اعمال شده بر روی صفحه بازی به صورتی که به حالت اولیه برگردد. در این روش، بعد از انجام محاسبات مدل، تغییرات اعمال شده بازگردانده میشوند. بدین صورت تنها با استفاده از یک صفحه فرآینده جستجو انجام میشود. این بهینه‌سازی در عمق های 10 و 15 مورد بررسی قرار گرفته است. و کاهش مصرف در هر دو عمق به خوبی قابل مشاهده می‌باشد.

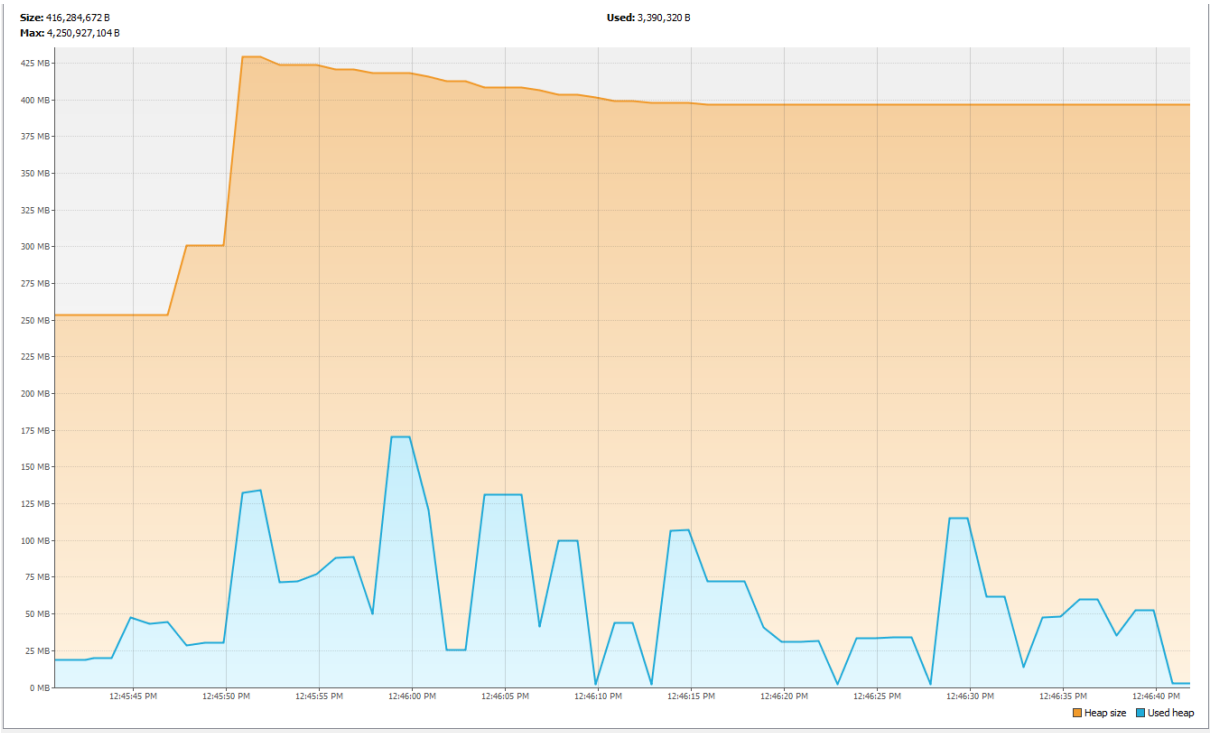


Figure 1 عمق 10 بدون بهینه سازی

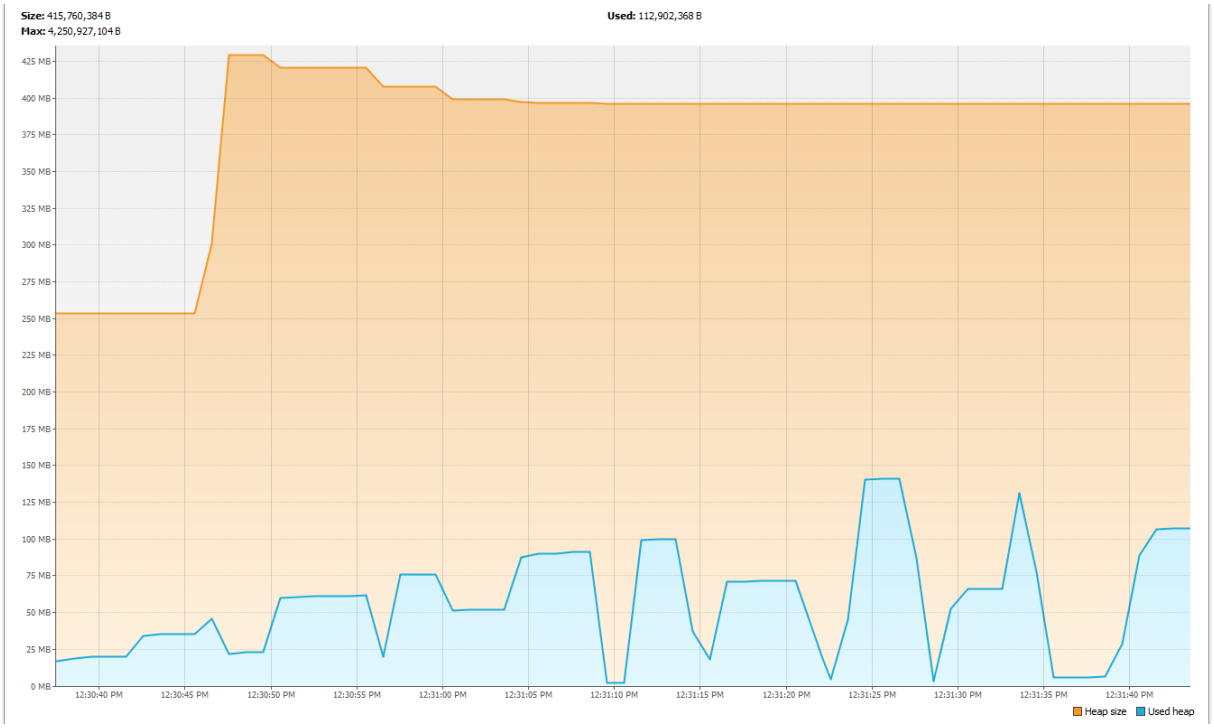


Figure 2 عمق 10 همراه با بهینه سازی

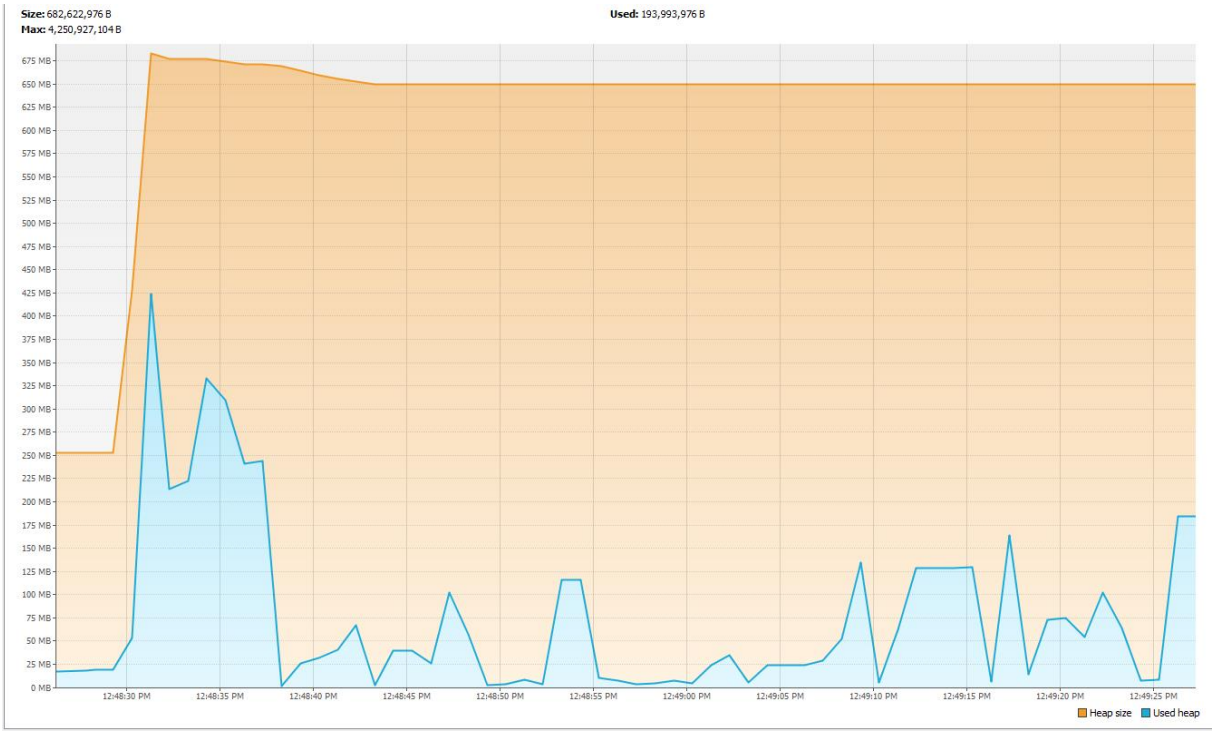


Figure 3 عمق 15 بدون بهینه سازی

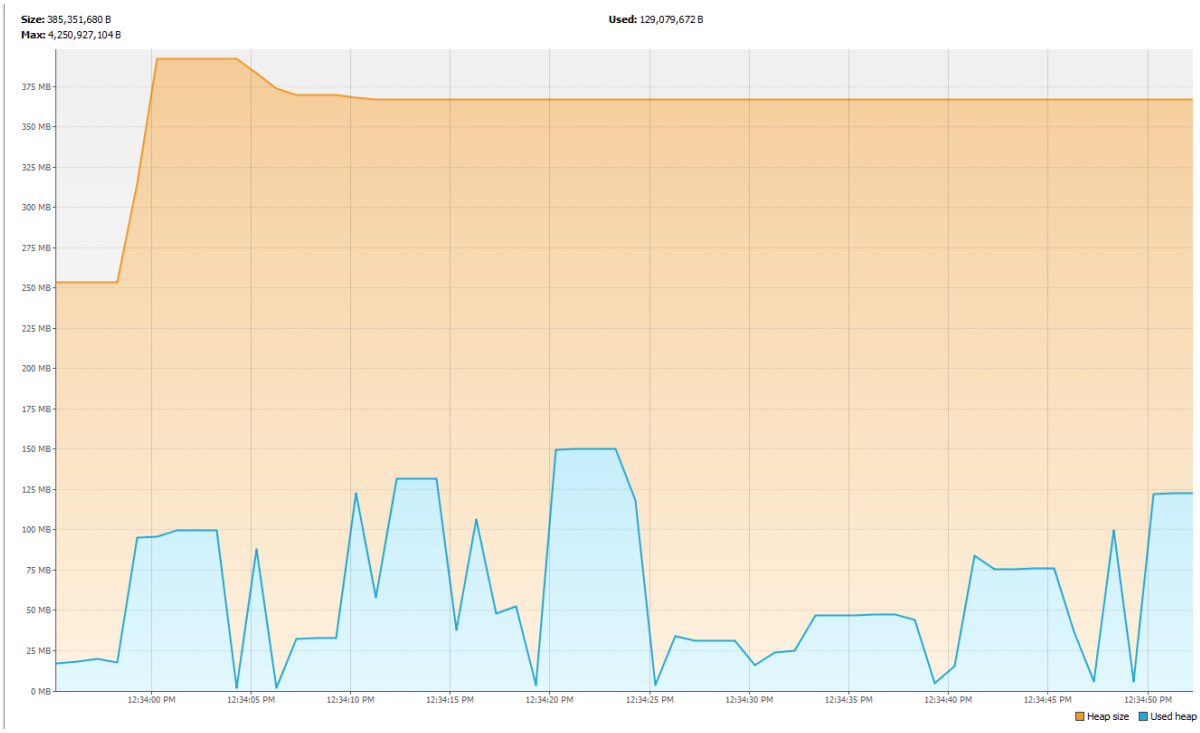


Figure 4 عمق 15 همراه با بهینه سازی

تکامل مدل

در درخت مینیماکس ویژگی هایی برای کاهش عرض درخت و ویژگی هایی برای امتیاز دادن به صفحه داشتیم که هر ویژگی ضریبی داشت. در این مرحله با تغییر این ضرایب به هر ویژگی وزن مناسبی می‌دهیم تا مدل بهترین حرکت را در بازی انتخاب کند.

ویژگی های الگوریتم ژنتیک

جمعیت اولیه	50% به صورت تصادفی و 50٪ به صورت مکاشفه‌ای
بازنمایی	حقیقی
بازترکیبی	با احتمال 1، میانگیری
جهش	با احتمال 0.7، گاوسی
انتخاب والدین	تصادفی
نوع انتخاب بازماندگان	انتخاب از والدین و فرزندان
روش انتخاب بازماندگان	رتبه‌بندی

بازنمایی

بازنمایی به صورت حقیقی و تعداد سلول‌های ژن 12 عدد به تعداد ویژگی‌ها می باشد. مقدار هر خانه در بازه [0.1,15] می‌باشد. هر ژن به عنوان یک مدل متفاوت در نظر گرفته می‌شود.

جمعیت اولیه

نصفی از جمعیت اولیه به صورت تصادفی در بازه گفته شده و نصف دیگر به صورت مکاشفه‌ای انتخاب می‌شوند. می‌دانیم هر ویژگی چقدر اهمیت دارد. درنتیجه بازه‌ای برای هرکدام تعریف می‌کنیم و به صورت رندوم از آن بازه انتخاب میکنیم.

ضرایب مکاشفه ای		
استفاده شده در	ویژگی	بازه
کاهش عرض درخت	گوشه بودن	[10,15]
	حرکت در وسط	[1,8]
	محاصره کمتر	[5,10]
	از دست دادن گوشه	[8,13]
	کاهش حرکات مجاز رقیب	[4,8]
	حرکت در حاشیه صفحه	[3,7]
محاسبه یوتیلیتی	اختلاف تعداد مهره‌ها	[4,8]
	تعداد حرکات مجاز بازیکن مقابل	[7,12]
	تعداد دیسک‌های مرزی	[4,8]
	تعداد گوشه‌های تسخیر شده	[12,15]
	تعداد مهره‌های تثبیت شده	[10,15]
	اتمام کننده بازی	[2,7]

بازترکیبی

بازترکیبی با احتمال 1 به روش میانگیری انجام میشود. پس از انتخاب والدین به صورت تصادفی، مطابق با فرمول زیر ژن فرزند بدست می آید :

$$New\ Gene\ =\ \alpha iG1\ +\ (1-\alpha i)G2\ ;\ \alpha i\ \epsilon\ (0,1)$$

جهش

بعد از بازترکیبی، با احتمال 0.7 جهش به روش گاوسی بر روی فرزند تولید شده اعمال می‌شود.

تابع ارزیابی ژن

در هر نسل ژن ها دو به دو با یکدگیر بازی میکنند. مطابق با فرمول gamei,j به امتیاز ژن افزوده می شود. در نهایت مجموع امتیازی که ژن از همه بازی‌هایش بدست می‌آورد به عنوان مقدار fitness در نظر گرفته می‌شود.

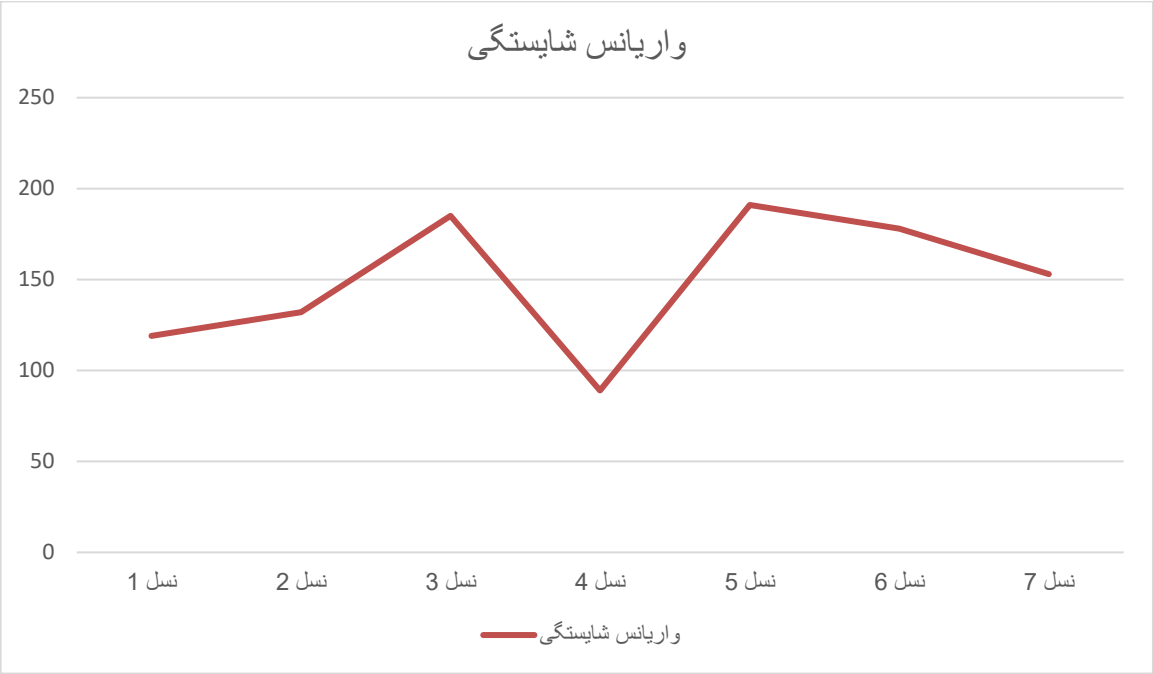
$$fitness_i = \sum_{j\ !=i}^{n} game_{i,j}\ ,\qquad game_{i,j} = \begin{cases} 3 + \dfrac{abs(disk\ difference)}{64} * 3 & win \\ 1 & draw \\ 0 & lose \end{cases}$$

شرط خاتمه

واریانس شایستگی ژن‌ها را بررسی میکنیم. و در صورتی که واریانس نسل جدید نسبت به نسل‌های قبل تغییری چندانی نداشته باشد، نشان از عدم بهبود است. در این حالت به جستجو خاتمه میدهیم.

نتیجه

با ایجاد 12 والد و 6 فرزند در هر نسل، طی 7 نسل و 1071 بازی فرآیند تکامل در حدود 18 ساعت انجام شد. تغییرات واریانس شایستگی در ابتدا داری نوسان بود اما در ادامه روند کاهشی به خود گرفت.



5 Figure نمودار تغییرات واریانس در هر نسل

ژن شماره 23 از نسل 4 به عنوان جواب برای مدل انتخاب شد. در عمل، این ژن از برترین ژن نسل های دیگر بهتر بازی می‌کند و عملکرد قابل قبولی دارد.

7.289 , 6.864 , 8.128 , 12.567 , 8.297 , 8.277 , 12.188 , 7.926 , 5.992 , 12.648 , 4.596 , 6.761

جمع‌بندی

با ادامه نسل‌ها و کاهش واریانس شایستگی می‌توانیم به مدل بهتری برسیم. فرآیند تکامل هنوز نیاز به ادامه دارد اما به علت نبود وقت و منابع کافی، این فرآیند خاتمه یافت.