عماد آقاجان زاده

پروژه گیم فاز نهایی:

کاهش نرخ انشعاب

مقدمه:

همانطور که در درس داشتیم نرخ انشعاب در درخت تصمیم بسیاری از بازی ها بزرگ است و لذا پیاده سازی الگوریتم های تصمیم گیری نظیر minimax برای این درخت ها باید با محدود سازی هایی صورت گیرد تا قابل اجرا باشند.

به عنوان نمونه یک عملی که پیش تر انجام داده ایم تعیین یک level off برای بررسی است به این مفهوم که برای تعیین امتیاز گره تا پایان درخت (برگ) پیش نرویم و تا یک عدد مشخصی عمق های بعدی را بررسی کنیم.

حال در این قسمت از پروژه یک روش دیگر را برای بالابردن توان محاسباتی مدل ارائه شده بررسی کنیم که در آن نرخ انشعاب گره ای درخت کاهش یابد و بتوانیم عمق های بیشتری را مورد بررسی قرار دهیم.

هرس کردن راس های کم وزن

پیشتر در محتوای درس روشی مبتنی بر هرس کردن بخش ها اضافی درخت بازی در نسخه آلفا بتا الگوریتم minimax گفته شد.

می دانیم که اگر بتوانیم راس ها طوری مرتب کنیم که همواره اولین فرزند یک راس min کمترین مقدار را داشته باشد و بالعکس یعنی اولین فرزند یک راس max بیشترین مقدار را داشته باشد، در این صورت الگوریتم بسیار کارارتر عمل خواهد کرد.

اما مشکلی که وجود دارد آنکه در ابتدای کار از مقدار وزن راس ها خبر نداریم و ناچارا باید راس را تا یک عمق مشخص گسترش دهیم تا در نهایت وزن مربوط به آن تعیین شود.

به عنوان یک راهکار خوب برای ترتیب دهی گره ها می توان اینگونه عمل کرد که در هر سطح مستقل از عمق های پایینتر یک روش ارزیابی و وزن دهی گره ها را پیش گرفت و تنها گره هایی را گسترش داد که وزن مشخصی را دارا هستند. بدین صورت دیگر نیازی به گسترش همه گره های آن سطح نداریم و تنها آن هایی را گسترش می دهیم که با توجه به وزن دهی مناسب تر هستند.

اما خود اینکه چه میزان از راس های وزن دهی شده را جهت گسترش انتخاب کرد یک پارامتر ای هست که باید به صورت تجربی و با آزمایش کردن روش تعیین نمود.

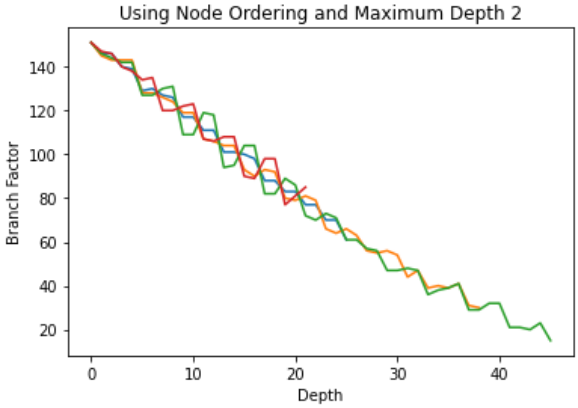
محاسبه بار محاسباتی اضافه شده

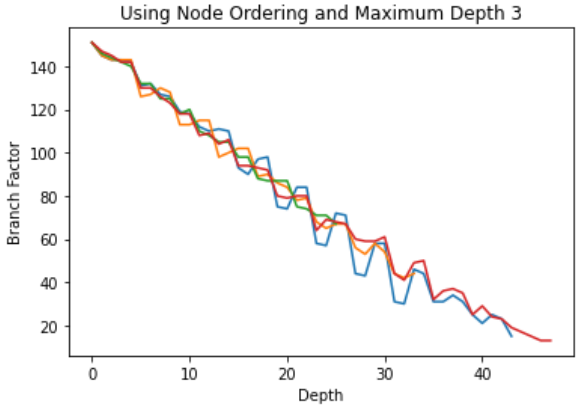
۱. تاثیر بر روی نرخ شاخه شاخه شدن:

برای بررسی تاثیر ترتیب دهی گره ها بر روی نرخ انشعاب درخت بازی نمودار های زیر را ترسیم کردم.

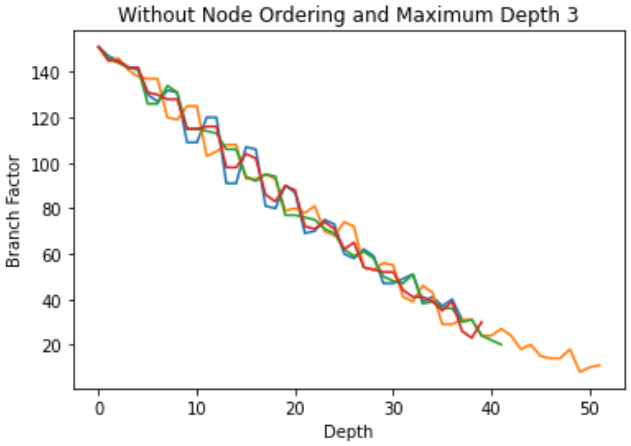
همچنین در زیر هر نمودار میانگین نرخ انشعاب در آن تنظیمات بازی را آورده ام.

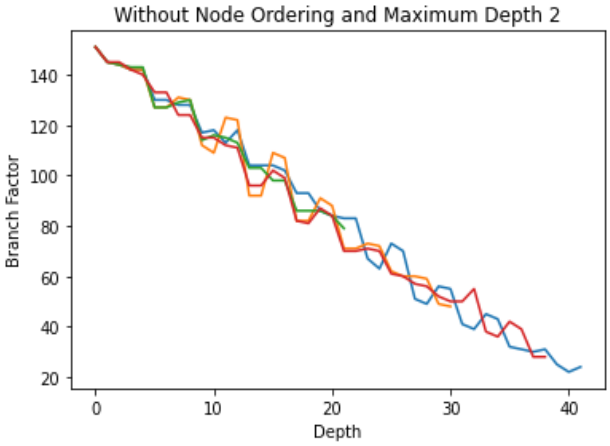
همچنین برای داشتن نتایج پایدار تر از چندین بار اجرای بازی استفاده شده است.





در دو نمودار بالا نرخ انشعاب درخت به ازای اجرای بازی با ترتیب دهی گره ها رسم شده است. در نمودار سمت راست میانگین نرخ انشعاب برابر ۹۱.۵۴ است و در نمودار سمت چپ ۱۰۲.۵۵ است.

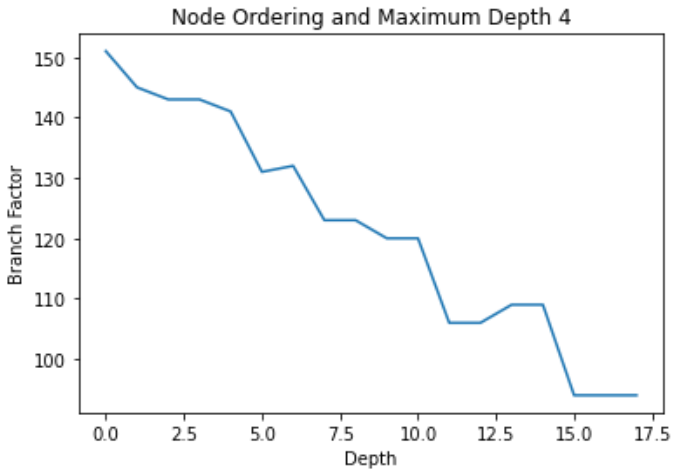




همچنین در این دو نمودار نیز، برای نمودار راست میانگین برابر ۸۶.۵۶ و برای نمودار چپ برابر ۹۹.۲۷ است.

همچنین نکته جالب این است که پیش از این قادر به انجام بازی برای عمق ۴ نبودیم (از لحاظ زمان اجرای طولانی) اما با استفاده از این روش توانستم با عمق ۴ نیز بازی را انجام دهم. اما لازم به ذکر است

که برای اجراهای بالا نرخ انتخاب گره ها را برابر ۰.۵ قرار دادم اما برای عمق ۴ با این نرخ هم قادر به اجرای بازی در زمان معقول نبودم لذا این نرخ را برای این نمودار برابر ۰.۲ قرار دادم و به نتیجه زیر رسیدم:



برای این نمودار نیز میانگین برابر ۹۳.۲۲ شد.

همانطور که انتظار میرفت برای عمق های بالا بازی زودتر و با گام های کمتری پایان می پذیرد چراکه انتخاب های بازیگر هوشمند بهتر است (به دلیل بررسی ۴ عمق) لذا می تواند با گام های کمتری حریف را مغلوب کند.

این نکته را میتوان به وضوح از نمودار های بالا استنباط کرد چراکه با عمق ۴ در ۱۷ گام بازی به پایان رسیده است در صورتی که در دیگر بازی ها این عدد در حدود ۴۰ بوده است. اما باز هم زمان اجرای بازی های قبلی بهتر است.

۲. تاثیر بر روی نرخ پیروزی:

از آنجا که هدف اصلی ما بردن بازی است به عنوان یک فاکتور بسیار مهم نرخ پیروزی ها را در هر بار بازی محاسبه کردم که گزارش آن به شرح زیر است:

۱) استفاده از ترتیب دهی و عمق ۲: ۴۰ درصد بازی ها را بردیم.
۲) استفاده از ترتیب دهی و عمق ۳: ۱۰۰ درصد بازی ها را بردیم.
۳) استفاده نکردن از ترتیب دهی و عمق ۲: ۸۰ درصد بازی ها را بردیم.
۴) استفاده نکردن از ترتیب دهی و عمق ۳: ۶۰ درصد بازی ها را بردیم.
۵) استفاده کردن از ترتیب دهی و عمق ۴: در ۱ بار اجرای بازی، آنرا بردیم.

نتایج بالا به ازای ۵ بار اجرای بازی به ازای هر پیکر بندی است و لذا ناسازگاری هایی در آن وجود دارد. اما نکته ای که انتظار میرفت و در بالا نیز مشهود است این است که با بیشتر کردن عمق درخت قطعا میانگین نرخ بردمان بیشتر خواهد شد.

در نتیجه باید طوری الگوریتم هایمان را بهبود دهیم که بررسی درخت در عمق های پایینتر عملی باشد چراکه این کار تاثیر بسزایی در نتیجه بازی خواهد داشت.

۳. تاثیر بر روی زمان اجرا:

زمان اجرا را نیز به عنوان یک فاکتور مهم در اجرای بازی مورد بررسی قرار دادم:

۱) میانگین زمان اجرا در صورت استفاده کردن از ترتیب دهی گره ها و عمق ۲: ۲۱۱۱۱
۲) میانگین زمان اجرا در صورت استفاده کردن از ترتیب دهی گره ها و عمق ۳: ۱۵۴۲۳۵
۳) میانگین زمان اجرا در صورت استفاده نکردن از ترتیب دهی گره ها و عمق ۲: ۴۳۱۳۵
۴) میانگین زمان اجرا در صورت استفاده نکردن از ترتیب دهی گره ها و عمق ۳: ۱۸۲۳۱۰
۵) میانگین زمان اجرا در صورت استفاده کردن از ترتیب دهی گره ها و عمق ۴: ۶۱۶۵۲۳

زمان اجرای های بالا بر حسب میلی ثانیه است.

همانطور که مشخص است زمان اجرا با بالا رفتن عمق بسیار زیاد خواهد شد.

و ترتیب دهی میتواند بسته به نرخ انتخاب گره ها تا حدودی زمان اجرا را نصف کند.

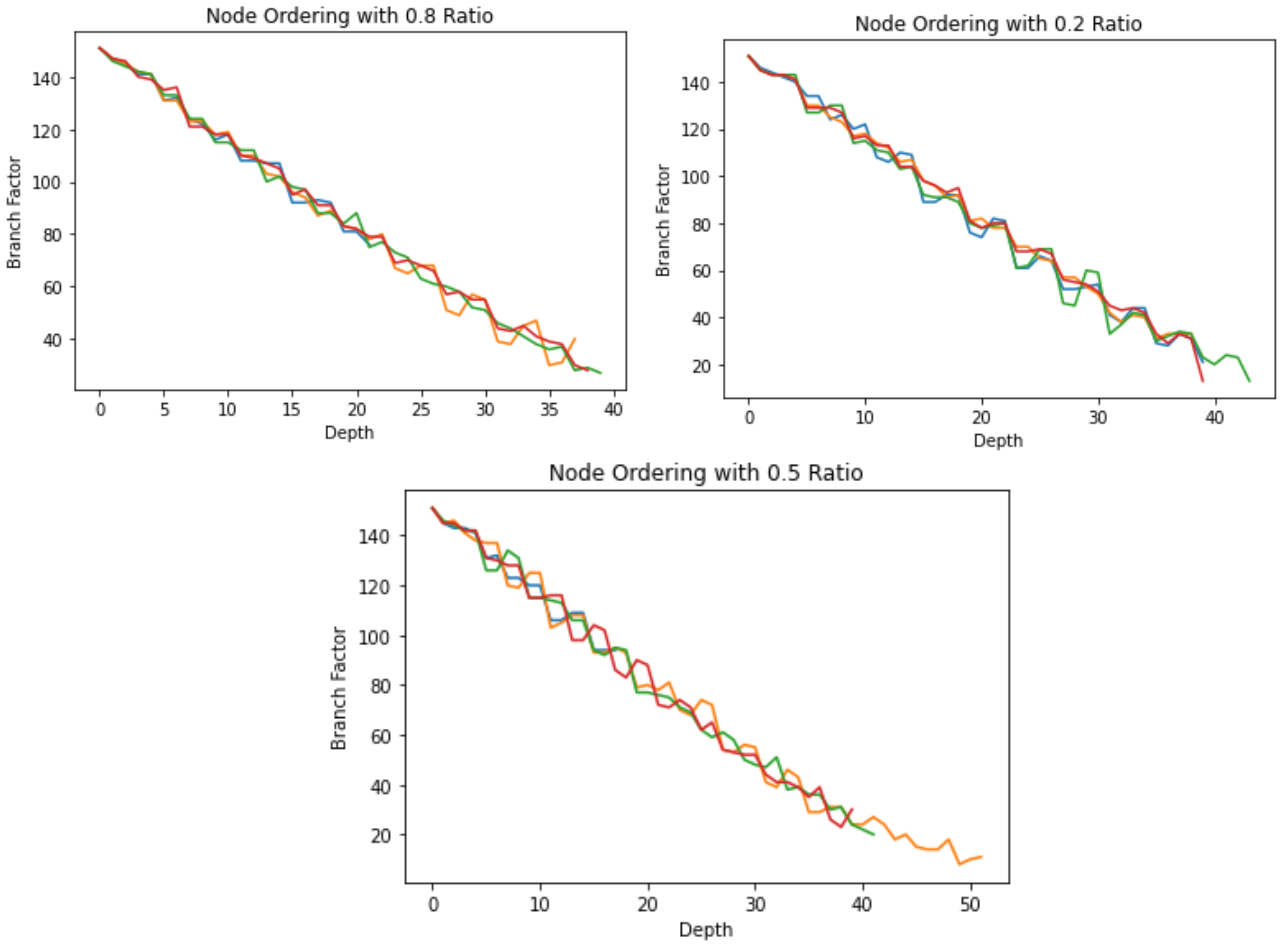
۴. تاثیر انتخاب درصد برش راس ها:

پس از آنکه راس ها را با یک تابع ارزیابی امتیاز دهی کردیم نیاز است تا با انتخاب یک پارامتر درصد برش راس های مناسب را گسترش و دیگر رئوس را هرس کنیم.

در نمودار های بالا به جز در حالت عمق ۴ این عدد برابر ۰.۵ بوده است که بیانگر آن است که در هر سطح تنها نصف گره های مناسب را گسترش می دهیم.

قاعدتا انتظار داریم تا با افزایش این پارامتر زمان اجرا بیشتر و نتیجه بهتر شود اما با کمتر کردن آن بار محاسباتی مدل ما کمتر شده و لذا قادر خواهیم بود تا عمق ها بیشتری را بررسی کنیم. (که همانطور

که گفته شد این خیلی تاثیر مطلوبی دارد و نرخ پیروزی ها را بسیار زیاد می کند.) همانطور که در بالاتر قادر نبودیم که با نرخ ۰.۵ تا ۴ عمق را بررسی کنیم اما توانستیم با ۰.۲ کردن مقدار این نرخ تا ۴ عمق پایینتر را هم بررسی کنیم.



حال همانطور که میبینید زمانی که نرخ انتخاب گره ها بزرگ است تصمیمات بهتر و در نتیجه بازیکن ما زودتر بازیکن حریف را مغلوب می کند اما در عوض زمان اجرای ما بیشتر خواهد شد. میانگین زمان اجرای برنامه با نرخ های متفاوت در زیر آورده شده است.

۱) میانگین زمان اجرا با نرخ ۰.۲ : ۱۰۶۵۷
۲) میانگین زمان اجرا با نرخ ۰.۵ : ۲۱۱۱۱
۳) میانگین زمان اجرا با نرخ ۰.۸ : ۳۷۴۱۳

که همانطور که گفته شد رابطه ای واضح میان نرخ هرس گره ها و زمان اجرای برنامه وجود دارد.

معیار مرتب سازی

پس از آنکه نحوه انتخاب درصد برش و تاثیر آنرا بر عملکرد مدل بررسی کردیم، حال نوبت به آن رسیده است که معیار مرتب کردن گره ها را بررسی کنیم.

نکته مهمی که باید به آن توجه داشت آن است که چون این تابع در هر عمق صدا زده می شود و باید حدود ۹۰ گره (میانگین نرخ انشعاب که در بالاتر به دست آمد) را ارزیابی کند لذا باید پیچیدگی زمانی مناسبی داشته باشد و به مراتب از تابع ارزیابی که در برگ ها درخت (حالات پایانی) استفاده می شود، ساده تر باشد.

با این تفاسیر می توان یکی از دو تابع زیر را مورد استفاده قرار داد.

۱- اختلاف تعداد حرکت های مجاز بازیکن ما و حریف (هر چه بیشتر باشد بهتر است.)

۲- اختلاف تعداد حمله هایی که می توانند بکنند. (هر چه بیشتر باشد بهتر است.)

که با آزمایش های تجربی به این نتیجه رسیدم که تابع ارزیابی دوم بهتر عمل می کند.

پیاده سازی

پس از آنکه به بررسی مباحث تئوری و توجیه استفاده از ترتیب دهی گره ها پرداختیم نوبت به آن می رسد که به پیاده سازی این ایده بپردازیم.

برای انجام این قسمت در کلاس AI در دو تابع firstAction و secondAction تمامی اکشن ها را پیش از گسترش بر اساس تابع ارزیابی نوشته شده مرتب می کنم و سپس با توجه به نرخ هرس گره ها اکشن ها را جهت گسترش دادن انتخاب می کنیم. که جزئیات بیشتر این قسمت را می توان در کد دنبال کرد.

نتیجه گیری

روش ترتیب دهی گره ها به عنوان یک روش کارآمد برای کم کردن نرخ انشعاب درخت می تواند مورد استفاده قرار گیرد. این روش از آنجا کارآمد است که می توان با بار محاسباتی کاهش یافته، عمق بررسی حرکت ها را زیاد کرد که این کار با توجه به نتایج تجربی و گزارش های هفته های گذشته می تواند به طور چشم گیری درصد برد های ما را افزایش دهد.