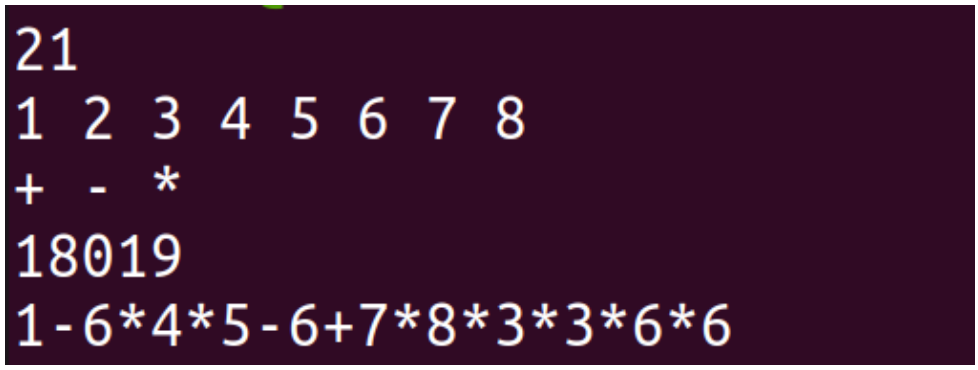
اولدوز نیساری 810199505

گزارش پروژه دوم ژنتیک و بازی

درس هوش مصنوعی

بخش اول : ژنتیک



1 – هریک از اعداد مثبتی که به عنوان ورودی به برنامه داده شده است و هر یک از عملگر هایی که به عنوان عملگر های قابل استفاده به برنامه داده شده اند ژن های ما هستند . با کنار هم قرار دادن این ژن ها و ساختن رشته ای به طول equationSize مشخص شده کروموزوم را تشکیل می دهیم .برای ساختن کروموروم در برنامه به این صورت عمل می کنیم که یک حلقه را به تعداد کف حاصل تقسیم equationSize بر 2 بار اجرا می کنیم .هر بار در ابتدا یکی از اعداد مثبت را با تابع رندوم انتخاب می کنیم و سپس یکی از عملگر ها را با تابع رندوم انتخاب می کنیم و به رشته کروموزم می افزاییم .در نهایت هم یکی از اعداد مثبت را انتخاب و به رشته اضافه می کنیم .

2- به تعداد populationSize مراحلی که در بخش طی کردیم را طی می کنیم و عملا به تعداد populationSize کروموزوم تشکیل می دهیم. مجموعه این کروموزوم ها می شود جمعیت اولیه ما .

3 – تابع معیار سازگاری را به این صورت تعریف می کنیم که حاصل رشته کروموزومی را با رعایت ترتیب و اولویت عملگر ها را محاسبه می کنیم و اختلاف آن را با goalNumber مان در نظر می گیریم . واضح است که با این تابع معیار سازگاری هر قدر خروجی تابع کمتر باشد سازگاری بیشتر است.

4- در این بخش کروموزوم ها را به تابع cross over می فرستیم . در این تابع با انتخاب دو تا از کروموزوم ها به صورت تصادفی عملیات cross over روی آن ها انجام می شود .نقطه ای به تصادف روی رشته کروموزوم انتخاب می شود .برای فرزند اول از ابتدای این رشته تا آن نقطه از کروموزوم والد اول انتخاب می شود و ادامه کروموزم از والد دوم و برای فرزند دوم به صورت برعکس .

در تابع mutation هم تعدادی از ژن ها از یک کروموزوم به تصادف توسط مجموعه ژن ها جایگزین میشود . این تعداد معمولا بسیار کوچک است ..

-5

– 6

1) اگر جمعیت اولیه بسیار کم جمعیت باشد احتمال این که از یک جایی به بعد نسل بعدی که توسط cross over تولید می شود مشابه نسل قبل خود شود زیاد است . در این حالت چون تغییر کلیت جمعیت بسیار کم و عملا ناچیز است ، در واقع در یک حالتی متوقف میشویم و ممکن است نتوانتیم به جواب برسیم . هم چنین اگر جمعیت زیاد باشد اولا زمان اجرای الگوریتم بسیار افزایش می یابد . به علاوه در اثر تنوع و گوناگونی ایجاد شده ممکن است در تولید هر نسل کروموزوم های متفاوت و با اختلاف زیاد با goal مان تولید شود که حذف و کم رنگ کردن نقش آن ها در جمعیت کلی نیاز به زمان و هزینه زیادی دارد.

2) با افزایش جمعیت دقت افزایش می یابد اما سرعت الگوریتم کاهش می یابد . اما لازم است دقت کنیم که به دلایل ذکر شده در قسمت قبل ممکن است افزایش جمعیت الگوریتم را از توقف دور تر کند و سرعت را افزایش دهد .

3) cross over روشی است که کمک می کند بتوانیم به صورت معقول روی دو کروموزوم تغییر ایجاد کنیم .یعنی هم تنوع و تغییر را ایجاد کرده ایم و هم میزان خوبی از ویژگی های نسل قبل را حفظ کرده ایم .mutation اما روشی است که کمک ما می کند میزان اختلاف را از نسل قبل کمی بیشتر افزایش دهیم . cross over بدون mutation می تواند منجر به نسل های مشابه با نسل قبلشان شود که ما را به جواب اصلی نزدیک تر نمی کنند . اگر mutation به تنهایی انجام شود که تنوع ایجاد نمیشود و سرعت تغییر کروموزوم ها بسیار بسیار کم می شود.

4) میتوانیم نرخ انتقال مستقیم به نسل بعد را کم کنیم .تعداد ژن هایی که mutate می‌کنیم را زیاد کنیم.

5) میتوانیم یک متغیر داشته باشیم که اگر چند دفعه متوالی عملیات mutation , cross over انجام شد و تغییر جمعیت از یک میزانی کمتر بود جمعیت اولیه جدیدی مجدداً بسازیم.

6) می توانیم یک میزانی از اختلاف در نظر بگیریم و اگر میزان سازگاری رشته با هدف از یک حدی بیشتر بود الگوریتم را متوقف کنیم .یا می توانیم تعدادی مرحله مشخص کنیم و عملیات ساختن نسل های جدید را نهایتا به آن تعداد تکرار کنیم.

بخش دوم : بازی

سؤال ۱: هیورستیک خوب هیورستیکی است که بتواند منجر به افزایش درصد برد ما بشود . به علاوه تناسب میان هیورستیک و نوع عمل‌کرد بازیکن حریف هم مهم است . روشی که در حالتی که حریف حریصانه بازی می‌کند با روشی که وقتی حریف رندوم بازی می‌کند در پیش می‌گیریم باید متفاوت باشد.

هیورستیکی که من نوشتم در اصل به این صورت عمل می‌کند که به تعداد مثلث هایی که دو یال آن‌ها رنگ شده است توجه می‌کند .برای این نکته درجه قرمز و آبی هر راس را در نظر می‌گیرد. از هر راس به تعداد انتخاب ۲ از درجه قرمز مثلث با دو یال قرمز داریم که این‌ها برایمان خطرناک هستند لذا به آن‌ها امتیاز منفی میدهیم و به تعداد انتخاب ۲ از درجه آبی مثلث آبی داریم که این‌ها برای حریف خطرناک هستند پس به این‌ها امتیاز مثبت میدهیم.تعدادی هم مثلث هستند که یک یال آن‌ها و یک یال آن‌ها قرمز است آن‌ها هم باعث می‌شوند که ضلع سوم مثلث قابل انتخاب باشد لذا به آن‌ها امتیاز مثبت میدهیم.

البته من در روند نوشتن از چندین هیورستیک دیگر هم استفاده کردم :

تابعی به ازای هر حرکت احتمال باخت را به ازای هر یک از حرکات از مجموعه حرکت‌های قابل دسترسی می سنجید .

تابعی که به ازای تمام مثلث ها تعداد یال های قرمز آن‌ها می شمارد و...

سوال۲ : با افزایش عمق در حقیقتاً بررسی های بیشتری برای انجام یک حرکت انجام می‌شود ُ در این صورت در حالتی که حریف هم به صورت استراتژیک و به هدف باخت ما بهترین حرکت را انجام دهد شانس پیروزی افزایش می‌یابد . اما وقتی حریف از حرکات رندوم استفاده می‌کند گاهی افزایش عمق منجر به باخت بیشتر می‌شود .اتفاق که می‌افتد این است که عملاً با محاسبه عمق های بسیار جلوتر عواقب حرکاتی را پیش‌بینی و از آن‌ها خودداری می‌کند که به دلیل انتخاب تصادفی رقیب رقیب هیچ گاه آن‌ها را انجام نمیدهد و نتیجه این می‌شود که زودتر می بازد.

افزایش عمق به دلیل محاسبات بیشتر باعث می‌شود زمان بیشتر صرف شود .اما از طرفی دیگر به دلیل آنکه در بازی ای بیشتری موجب برد سریع‌تر می‌شود کمی از زمان از آن جمت کم می‌شود .

افزایش عمق تأثیر مستقیم روی نود های دیده شده دارد و باعث افزایش آن‌ها میشود.

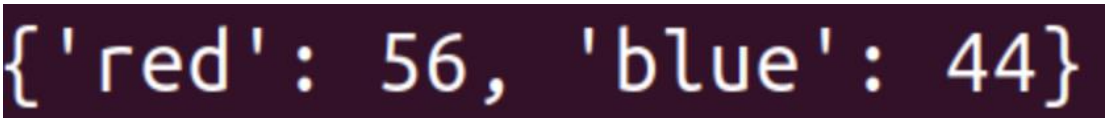
سؤال ۳:

به ترتیبی که حرکت ها در استیت ذخیره شده‌اند می‌بینیم. بله اهمیت دارد .ممکن است نودی که بناست هرس شود به عنوان آخرین نود دیده شود .واضح است که در این حالت هرس کردن فایده‌ای ندارد.این تربیت به دلیل نظمی و مرتب بودنی که دارد هرس کردن را بی‌فایده نمی کند.

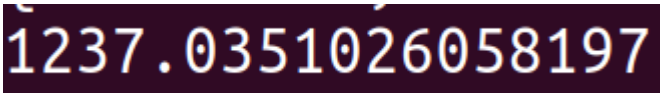
نتایج:

با عمق ۱ بدون هرس:

تعداد برد :

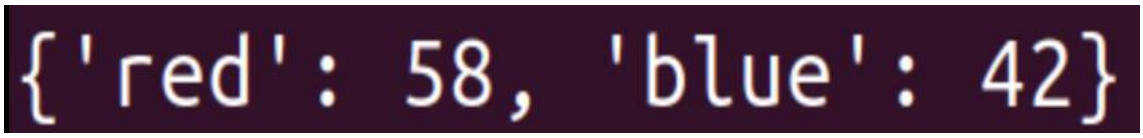


زمان :

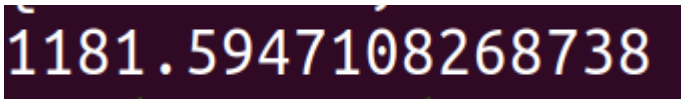


با عمق ۳ بدون هرس :

تعداد برد :



زمان :



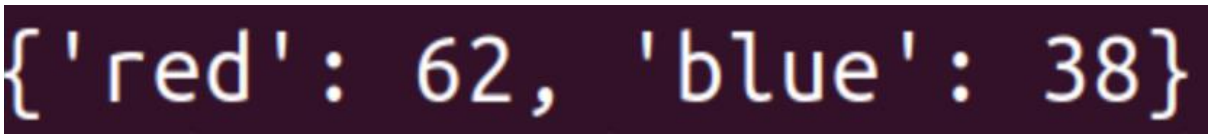
با عمق ۵ بدون هرس :

تعداد برد :

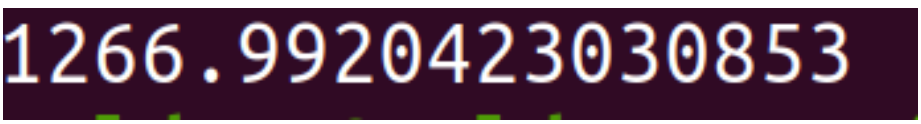
زمان :

با عمق ۱ با هرس :

تعداد برد :



زمان :



با عمق ۳ با هرس :

تعداد برد:

```
{'red': 58, 'blue': 42}
```

زمان :

```
1161.1321394443512
```

با عمق ۵ با هرس :

تعداد برد :

```
{'red': 56, 'blue': 44}
```

زمان :

```
1066.20232796669
```

با عمق ۷ با هرس:

```
{'red': 51, 'blue': 49}
```

زمان :


1007.6254913806915