گزارش پروژه یک - هوش مصنوعی

سیدمحمد حسینی - ۸۱۰۱۹۴۵۴۱

توضیح مسأله

در این سؤال میخواهیم با استفاده از الگوریتم های سرچ به جواب مساله n وزیر برسیم

فضای حالت و عمل

در این مسأله فضای حالت، محل هر کدام از وزیر هاست. این فضای حالت در یک آرایه دو بعدی نامپای قرار میگیرد. فضای اکشن نیز به این صورت است که هر وزیر میتواند ۸ جهت حرکت کند. به همین منظوری یک فضای عمل ۸ حالت است. برای نگه داری این فضای عمل از یک آرایه دوبعدی استفاده کردم که ضرب دکاری (۱،۰،۱-) در خودش است که (۰،۰) از آن حذف شده است.

توابع تعریف شده

خواندن ورودی:

ابتدا فایل را با استفاده از تابع get_input میخوانیم. محل قرار گیری فایل را در متغیر اول برنامه قرار میدهیم. همچنین در اول برنامه تعداد خطوط ورودی را به سیستم میدهیم. (میتوان با استفاده از تعداد ورودی هم این کار را کرد. یک روش رو به کار گرفتم و تا آخر رفتم) ورودی ها در این تابع بر اساس استاندارد داده شده خوانده شده و سپس به آرایه numpy تبدیل می‌شود و به ایندکسینگ پایتون تبدیل می‌شود. به این صورت که از تمامی مقادیر ۱ مقدار کم میکنیم.

بررسی وزیرها:

برای بررسی وضعیت وزیر ها از تابع check_state استفاده میکنم. در این تابع ابتدا وجود وزیر دیگری در سطر و ستون را بررسی میکنیم و سپس به صورت قطری بررسی میکنیم. برای بررسی سطر و ستون، دو ستون آرایه دوبعدی را به set تبدیل میکنم و اندازه آنرا با اندازه اصلی ستون میسنجم؛ اگر برابر بود نشان میدهد که عدد تکراری نداریم و اگر برابر نبود یعنی عدد تکراری وجود دارد. برای بررسی قطری نیز اختلاف محل دو به دوی وزیرها را در نظر میگیرم و به بیشینه مقدار ارایه ۱ بعدی بدست آمده تقسیم میکنم و قدر نطلق آن اگر [۱و ۱] بود نشان میدهد که دو وزیر به صورت قطری همدیگر را تحدید میکنند.

تبدیل مختصات وزیر ها به صفحه:

تابع make_grid نوشته شده است که محل وزیر ها را گرفته و سپس به یک آرایه دو بعدی ۸*۸ تبدیل میکند.

رسم صفحه:

تابع print_grid نوشته شده است که ورودی یک ارایه دو بعدی ۸*۸ گرفته و شکل صفحه را آنگونه که در صورت سؤال گفته شده است رسم میکند.

پیاده‌سازی الگوریتم‌های سرچ:

BFS

برای سرچ به صورت BFS به این صورت در نظر گرفتم که یک صف کلی وجود دارد که frontier برگ‌ها را در خود نگه‌میدارد. به همین منظور ابتدا اولین گره که همان ورودی است در صف اضافه می‌شود. سپس با استفاده از حلقه تمام حالاتی که میتوانند در این گره وزیر ها حرکت کنند را بررسی میکنیم. به این صورت که هر وزیر را با توجه به فضای حالت گفته شده حرکت میدهیم. ۸ وزیر داریم و ۸ فضای حالت پس هر گره ما ۶۴ عدد گره بچه دارد. به همین صورت درخت تشکیل می‌شود و میتوان رو آن جست‌وجوی مورد نظر را پیاده‌سازی کرد.

این الگوریتم را برای ورودی اول شبیه سازی کردم. بعد از مدت ۳ دقیقه و ۱۳۲۰۰۰۰۰ تعداد حلقه و اندازه frontier 10034917 به جواب درست نرسید و بدلیل کم بودن فضای حافظه مجبور به قطع برنامه کردم. بدلیل اینکه frontier در صف بزرگ و بزرگ‌تر می‌شود حافظه زیادی برای انجام محاسبات لازم دارد.

```
13200000
o o x o o o o o
o o x o o x o o
o o o o o o x o
o o o o o o o o
o o o o o o o o
o o o o o o o o
x o o o o o x o
x o o o o o o x
184.97992944717407
10034917
```

دومین تست کیس نیز باگذشت ۳ دقیقه به جواب نرسید و سیستم OOM داد.

```
Number of Iteration :  11800000
o o o x o o o o
x o o o o o o o
o o o o x o o o
o o o o o o x o
o o o o o o o o
x x o o o o o o
o o o o x o o o
o o o o o o x o
Process Time :  167.74322700500488
Fontier size :  9838482
----------------
```

سومین تست کیس نیز جواب نداد:

```
Number of Iteration :  11800000
o o o x o o o o
x o o o o o o o
o o o o x o o o
o o o o o o x o
o o o o o o o o
x x o o o o o o
o o o o x o o o
o o o o o o x o
Process Time :  167.74322700500488
Fontier size :  9838482
----------------
```

دلیل جواب ندادن از نظر من این هستش که اول اینکه خیلی از مراحل تکراری دارد این درختی که تشـکیل شـده است. دوم اینکه تعداد حالاتی که دارد خیلی زیاد است. سوم اینکـه چـون در هـر مرحلـه از درخت یـک جابجـایی داریم،جابجایی هایی که خیلی بیشتر است در مراحل پایین‌تر قرار میگیرند. مثلاً فرض کنید یک وزیر را اگر ۴ خلنه با راست ببریم چیدمان درست بدست میاید. این جابجایی در عمق ۴ درخت است که بعد از $۶۴^۴$ بار جسـت‌وجو بـه آن میرسد. برای همین سرعت پایینی دارد و حافظه زیادی نیز لازم دارد.

IDS

برای IDS تابعی با همان نام نوشته شده است. یک تابع دیگر که به صورت بازگشتی است به نام lds_ نوشته شده است که تا عمق معینی پایین می‌رود و شرایط را چک میکند.
این الگوریتم برای سه ورودی بعد از گذشت ۲۰ دقیقه به جواب نرسید.

A*

برای این الگوریتم، بدلیل کمبود زمان نتوانستم الگوریتم را implement کنم. ایده کلی باری تابع heuristic به این صورت است که تعداد تهدید های کلی جدول را بدست میاوریم و با تعداد تهدید های قبلیی که بوده جمع میکنیم.