پروژه اول (Maze Problem with A* search)

نام و نام خانوادگی : احسان رضایی

شماره دانشجویی : ۹۷۲۰۲۳۰۱۵

استاد درس: دکتر پدرام

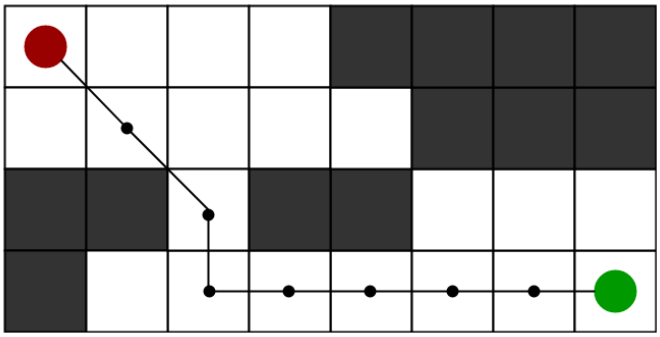
مقدمه

مسئله هزارتو یا Maze Problem مسئله ای است که از باید از یک ورودی داخل و به سمت خروجی حرکت کنیم.

پ.ن: در این مسائل بعضی راه ها توسط بلاک ها یا دیوار ها بسته شده است.

الگوریتم A*

از الگوریتم A* برای تخمین کوتاه‌ترین مسیر در مسائل جهان واقعی مانند نقشه‌ها و بازی‌هایی که امکان دارد موانع زیادی در آن‌ها باشد، استفاده می‌شود. می‌توان یک شبکه ۲ بعدی را در نظر گرفت که دارای چندین مانع است؛ کار جستجو در این شبکه، از سلول منبع (به رنگ قرمز رنگ‌آمیزی شده است) برای رسیدن به سلول هدف آغاز می‌شود (به رنگ سبز، رنگ‌آمیزی شده است). الگوریتم جستجوی A* یکی از بهترین و محبوب‌ترین روش‌های مورد استفاده در مسیریابی و پیمایش گراف است.



ساختار کد

کد شامل یک کلاس به نام Maze می‌باشد که شامل توابع و متغیرهای زیر می‌باشد.

- دیکشنری heuristics
  - در این متغیر heuristic هر خانه ذخیره می‌شود (در شروع الگوریتم توسط تابع make_heuristics محاسبه می گردد)
- دیکشنری adjacency_test
  - در این دیکشنری خانه های مجاور هر خانه ذخیره می‌شود (در شروع الگوریتم توسط تابع maek_adjacency محاسبه می‌شود)
- متغیر های maze_rows و maze_cols

- در این متغیر ها به ترتیب تعداد سطر و ستون جدول هرازتو ما مشخص می‌شود
- لیست path
  - در این متغیر توسط تابع بازگشتی find_path مسیر رسیدن به goal ذخیره می‌شود.
- متغیر های start و goal
  - در این مغییر ها به ترتیب مختصات شروع و مقصد ذخیره می‌شود.
- تابع make_adjacency
  - در این تابع ماتریس هم جواری خانه های هزار تو ساخته می شود و خروجی آن در دیکشنری adjancency_list ذخیره می‌شود.
- تابع make_heuristics
  - در این تابع فاصله‌ی mahattan هر خانه تا Goal محاسبه می‌شود.
- تابع solve
  - در این تابع که شامل تو متغیر fringe و explored می‌باشد در ابتدا node شروع ساخته شده و بعد از بررسی آن با استفاده از ماتریس همجواری، بلاک های همسایه آن که امکان رفتن به آن ها وجود دارد اضافه می‌شود.
    بعد از اتمام بررسی و یا شرط پیدا کردن goal ، تابع find_path فراخوانی می‌شود
- تابع find_path
  - این تابع در صورتی که به goal برسیم، مسیر جواب را به عنوان خروجی به ما می‌دهد.

هم چنین کد دارای کلاس Node هم می‌باشد که مختصات هر خانه ای که در مسیر موجود است را ذخیره می‌کند.

نحوه ی اجرا و خروجی برنامه

در انتها برای اجرای برنامه از main.py استفاده کرده و با ورودی های زیر

```
maze_cols = 3
maze_rows = 3
path = [[0, 1, 0],
        [0, 0, 0],
        [0, 1, 0]]
start = "0 0"
goal = "2 2"
```

به خروجی زیر می‌رسیم.

```
C:\Users\Ehsan\AppData\Local\Programs\Python\Python38-32\python.exe C:/Users/Ehsan/Desktop/project/ai_final_projects/MazeProblem/main.py
Cost is: 4
Path is: ['0 0', '1 0', '1 1', '1 2', '2 2']

Process finished with exit code 0
```