بسم الله الرحمن الرحیم

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

فاز چهارم پروژه در مبانی هوش و کاربردها

منطق درجه اول

استاد درس: دکتر حسین کارشناس

دستیار استاد: پوریا صامتی

دانیال شفیعی

مهدی مهدیه

سید امیررضا نجفی

بهمن ۱۴۰۳

فهرست مطالب

۱ پرولوگ

برنامه از الگوریتم جستجوی هزینه یکنواخت برای پیداکردن کوتاه‌ترین مسیر استفاده می‌کند و مسیر را به یک لیست از اقدامات تبدیل می‌کند در کد پرولوگ عملیات زیر انجام می‌شود

۱.۱ بارگذاری و تجزیه نقشه (Map Loading and Parsing)

- read_map(File) : این بخش از کد مسئول خواندن فایل نقشه است. فایل نقشه شامل یک شبکه (گرید) است که موقعیت پرنده، خوک‌ها، سنگ‌ها و سلول‌های خالی را مشخص می‌کند. اگر در حین خواندن فایل خطایی رخ دهد، پیام خطا نمایش داده می‌شود.
- read_grid(Stream, Row) : این بخش هر خط از فایل را می‌خواند و آن را به‌عنوان یک‌رشته پردازش می‌کند. اگر خط به پایان فایل نرسیده باشد، خط پردازش می‌شود و شماره سطر افزایش می‌یابد.
- process_line(Line, Row, Col) : این بخش هر کاراکتر از خط را پردازش می‌کند. اگر کاراکتر معتبر باشد (مانند B برای پرنده، P برای خوک، R برای سنگ، و T برای سلول خالی)، موقعیت آن در شبکه ثبت می‌شود. اگر کاراکتر ناشناخته باشد، پیام خطا نمایش داده می‌شود.

۲.۱ پردازش سلول‌ها (Cell Processing)

- process_cell : این بخش کاراکترهای مختلف را پردازش می‌کند و موقعیت آن‌ها را در شبکه ثبت می‌کند:
  - B : موقعیت پرنده را تنظیم می‌کند.
  - P : موقعیت خوک را اضافه می‌کند.
  - R : موقعیت سنگ را اضافه می‌کند.
  - T : سلول خالی را نشان می‌دهد و هیچ عملی انجام نمی‌دهد.
  - کاراکتر ناشناخته: پیام خطا نمایش داده می‌شود.

۳.۱ مدیریت وضعیت بازی (State Management)

- dynamic bird_pos/2, dynamic pig_pos/2, dynamic rock_pos/2 : این بخش از کد، موقعیت‌های پرنده، خوک‌ها و سنگ‌ها را به‌صورت پویا تعریف می‌کند تا در طول اجرای برنامه

تغییر کنند.

- set_bird :موقعیت پرنده را تنظیم می‌کند و موقعیت قبلی را پاک می‌کند.
- add_pig : موقعیت خوک را اضافه می‌کند، اما فقط اگر قبلاً در آن موقعیت خوکی وجود نداشته باشد.
- add_rock : موقعیت سنگ را اضافه می‌کند، اما فقط اگر قبلاً در آن موقعیت سنگی وجود نداشته باشد.

۴.۱ تعریف حرکات (Movement Definition)

- Move : این بخش حرکات ممکن در گرید را تعریف می‌کند
  - up: حرکت به بالا
  - Down : حرکت به پایین
  - Left : حرکت به چپ
  - Right : حرکت به راست
- valid_pos : این بخش بررسی می‌کند که آیا یک موقعیت در گرید معتبر است یا نه. یک موقعیت معتبر باید در گرید ۸ در ۸ باشد و روی سنگ قرار نگیرد.

۵.۱ جستجوی مسیر بهینه (Pathfinding)

- find_path : این بخش مسیر بهینه را برای رسیدن پرنده به تمام خوک‌ها پیدا می‌کند. ابتدا موقعیت پرنده و خوک‌ها را پیدا می‌کند، سپس مسیر بهینه را برای هر خوک محاسبه می‌کند.
- find_optimal_path : این بخش مسیر بهینه را برای رسیدن به هر خوک محاسبه می‌کند. از الگوریتم جستجوی هزینه یکنواخت (Uniform Cost Search) استفاده می‌کند.
- Ucs : این بخش الگوریتم جستجوی هزینه یکنواخت را پیاده‌سازی می‌کند. این الگوریتم کوتاه‌ترین مسیر را از موقعیت فعلی پرنده به موقعیت خوک پیدا می‌کند.
- ucs_search :این بخش جستجوی هزینه یکنواخت را انجام می‌دهد و مسیر بهینه را برمی‌گرداند.

۶.۱ تبدیل مسیر به اقدامات (Convert Path to Actions)

- convert_path_to_actions : این بخش مسیر پیدا شده را به یک لیست از اقدامات (حرکات) تبدیل می‌کند. هر حرکت با یک عدد مشخص می‌شود:
  - ۰ : حرکت به بالا.
  - ۱ : حرکت به پایین.
  - ۲ : حرکت به چپ.
  - ۳ : حرکت به راست.
- get_direction_number : این بخش جهت حرکت بین دو سلول را مشخص می‌کند و آن را به یک عدد تبدیل می‌کند.

۲ پایتون

کد آی پایتون از کتابخانه‌های pyswip و استفاده pygame می‌کند. در این گزارش، هر بخش از کد به طور کامل توضیح داده شده است.

۱.۲ بارگذاری و اجرای کد پرولوگ (Prolog Integration)

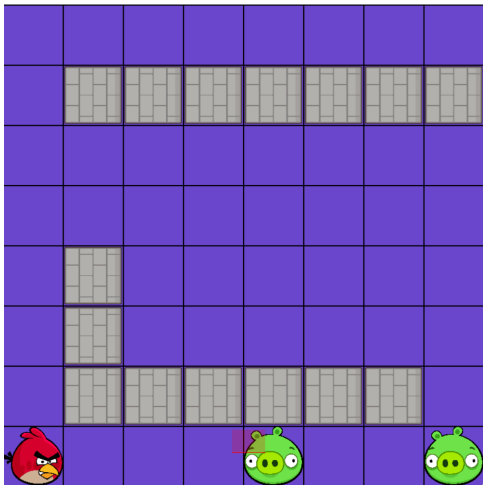
- prolog = Prolog() : یک شیء از کلاس Prolog ایجاد می‌شود که امکان اجرای کد پرولوگ در پایتون را فراهم می‌کند.
- prolog.consult("x.pl") : فایل پرولوگ x.pl که حاوی منطق بازی و الگوریتم‌های مسیریابی است، بارگذاری می‌شود.
- absolute_path = str(Path(f'Env/templates/ template.txt').resolve()).replace(' ', '/') : مسیر فایل نقشه (که موقعیت پرنده، خوک‌ها و سنگ‌ها را مشخص می‌کند) به‌صورت مطلق و با فرمت مناسب برای پرولوگ آماده می‌شود.
- list(prolog.query(f"read_map(' absolute_path ') ")) : نقشه از فایل خوانده می‌شود و موقعیت‌ها در پرولوگ ثبت می‌شوند. دریافت موقعیت‌ها از پرولوگ (Querying Prolog for Positions)

- print(list(prolog.query('pig_pos(X, Y)'))) : موقعیت تمام خوک‌ها از پرولوگ دریافت و چاپ می‌شود.
- bird_pos = env.get_bird_position() : موقعیت پرنده از محیط بازی دریافت می‌شود.
- print(list(prolog.query('bird_pos(X, Y)'))): موقعیت پرنده از پرولوگ دریافت و چاپ می‌شود.
- print(list(prolog.query('findall((PX, PY), pig_pos(PX, PY), Pigs)'))) : موقعیت تمام خوک‌ها از پرولوگ دریافت و چاپ می‌شود.

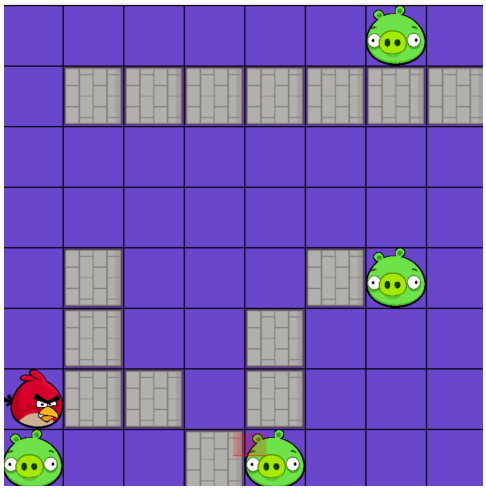
۲.۲ یافتن مسیر بهینه (Finding the Optimal Path)

- solutions = list(prolog.query(f"find_path(Actions)")) : مسیر بهینه برای رسیدن پرنده به تمام خوک‌ها از پرولوگ دریافت می‌شود.
- lengths = [len(d ['Actions']) for d in solutions] : طول هر مسیر محاسبه می‌شود.
- min_length = min(lengths): کوتاه‌ترین طول مسیر پیدا می‌شود.
- optimal_solutions = [d for d in solutions if len(d ['Actions']) == min_length] : مسیرهای بهینه (با کمترین طول) انتخاب می‌شوند.
- actions = optimal_solutions [0] ['Actions'] : اولین مسیر بهینه انتخاب می‌شود.
- print(actions) : اقدامات (حرکات) مربوط به مسیر بهینه چاپ می‌شوند.
- print(lengths) : طول‌های تمام مسیرها چاپ می‌شوند.
- print(solutions) : تمام مسیرهای پیدا شده چاپ می‌شوند.

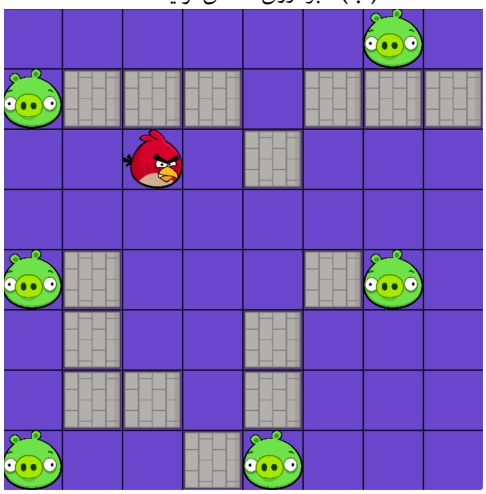
۳ اجرا روی نقشه‌ها



(آ) اجرا روی محیط ساده



(ب) اجرا روی نقشه‌ی تولید شده ۱



(ج) اجرا روی نقشه‌ی تولید شده‌ی ۲

شکل ۱: اجرا روی محیط‌ها