Intro2AI: Project 01 - SEARCH

Course `CSC10003`: Introduction to Artificial Intelligent
`FIT, HCMUS, 27/07/2020`

This is a group assignment of 4 members:

- `18127231`: Đoàn Đình Toàn (GitHub: @t3bol90)
- `18127195`: Hỷ Phú Quyền (GitHub: @HPQuyen)
- `18127170`: Dương Trung Nhật (GitHub: @zedeus33)
- `18127244`: Bùi Tạ Đức Tuấn (GitHub: @ductuan15)

Have a good day

About this assignment:

You are given a file which describe Pac-man World. Propose or apply learned algorithms to help Pac-Man to find foods without dying by monsters.

Pacman or monsters only moves in 4 direction: left, right, bottom, up and cannot move over or through the wall. The game has four levels:

- **Level 1**: Pac-man know the food's position in map and monsters do not appear in map. There is only one food in the map.
- **Level 2**: monsters stand in the place ever (never move around). If Pac-man pass through the monster or vice versa, game is over. There is still one food in the map and Pac-man know its position.
- **Level 3**: Pac-man cannot see the foods if they are outside Pacman's nearest three-step. It means that Pac-man just only scan all the adjacent him **(8 tiles x 3)**. There are many foods in the map. Monsters just move one step in any valid direction (if any) around the initial location at the start of the game. Each step Pacman go, each step Monsters move.
- **Level 4** (difficult): map is **closed**. Monsters will seek and kill Pac-man. Pac-man want to get food as much as possible. Pacman will die if at least one monster passes him. It is ok for monsters go through each other. Each step Pacman go, each step Monsters move. The food is so many.

Game points is calculated as following rules:

- Each moving step, your point will be decreased by 1.
- Each food you take, 20 points will be given for you.

You may need to run your algorithm on many different graphs to make a comprehensive comparison of these algorithms' performance regarding the following aspects:

- Time to finished
- The length of the discovered paths

Assignment Plan:

In three weeks:

Week	Week 1	Week 2	Week 3
Team plan	Looking solution for level 1 & 2 Simple Graphic display	Done level 1 & 2, apply level 1 & 2 to graphic display Brainstorm solution for level 3 & 4	Apply level 3 & 4 to graphic Generate random map to testing and fix bug

After a long day at Master Le Ngoc Thanh's Seminar (22/07/2020), we decided to work on graphic first. We split into 2 different parts: graphical module and built-in algorithms.

Graphic

Base on Stanford homework [1] (but we are not clone from it - because it write on `python2` :< but we want to build in `python3`). So we decide to rewrite our own graphic module with 3 following parts:

- Class GraphicUtils to connect with `tkinter` (GUI python) use canvas to create many kind of general shape(circle,square,dot,polygon...), display a test,load image.
- Class PacmanGraphics(module graphic pacman) use module above(GraphicUtils) to combine,calculate the shape,color,size for things relevant to pacman game graphic like pacman,monster,foods,aura,... or panel to display score.
- Class GameController to control game flow,the rule,all kind of movement of entities even algorithms.

The graphic's module will seprate from algorithms module. It's call algorithms module as a built-in fuction in `algorithms.py`.

Algorithms

Level 1

> We used A* algorithm for Pacman to find the food to optimize the path with shortest length. It also persure that expanded fewer node than other optimal search algorithms. A* with Manhattan distance [2] can be optimal in grid search - cost of path between 2 nodes in grid by using A* with Manhattan distance is minimum cost.

Because there is only one food on the map, there are 2 cases:

- Eat the food: if the cost < food value (**20**).
- Do not eat the food and surrender: if the cost > food value (**20**).

It is desicion tree with 2 branchs.

Level 2

> Because there's only one food and one monster without moving, we treat that monster as a wall, and use A* just like Level 1 to find the path to that food. But monster is not wall, it can bite you as a monster. Applied same method with level but if you are touched by the monster, you will be bitten and lose the game.

Level 3

> Idea : First of all, you need to find your vision base on your current position, then You will find the way follow in these case :

- Case 1: If nothing on your vision, choose random way but possible to go (mean either not wall or visited).
- Case 2: If you find foods on your vision, put it in memory and decide which way to get the closest food as fast as posible base on the memory of position you already passed.
- Case 3: If you see monster in your vision, you will determine the way to run away or to get a food(even leave it if it is too dangerous or lots of food in the area) base on your IQ.(I mean algorithms :)).
  `Plus, just remember to check and update your memory after each step you moved.`

> Based on idea above,apply for this pacman,correspond each case we use different search strategies:

- Case 1: Random way to go if it is exist, it mean you need to look around, if position you already visited then look another way, include the wall, of course! (In fact, we usually move up, nothing personal but we like it). If there is no another way(stucked!!), it's simple, you stop the game.
- Case 2: Use A* to find the way get to the closest food. For each food you have found in your discovered vision, put it in protity queue. After each step you move, you need to update the distance from your current position to foods you have found (it mean update the priority queue) , then choose the closest food to determine next move.
- Case 3: Use minimax(limited depth on vision) to calculate possible way to get the efficent score(multiple agent). If exist the way to get foods and don't lose the game(the monster can't see you), minimax stragety will decide you can reach it. If not, minimax stragety will decide you need to leave it, remove food from priority queue and choose another way base on your current vision. Minimax has a deep attribute as a trade off. If the deep is too large, the pacman is really scared of monsters. Else, it will be more confident and sometimes lose it's mind.

After each step you have moved, check the conditions state in each case.

> Loop it until you feel tired (nothing in current vision and you visited all the path you have found or visited half of map) and wanna stop(stop the game, keep the highest score as you could If you move one more step, you will lose the score and feel can't get more)!! We have apply some bias for this, case. If you walk but no foods or monster dare you in your vision, you need to think about giving up and end the game soon. The similar case is when you go to dead-end, 2 monster is watching you in the front door and you can't escape from it -> Break up soon to relieve suffering [3]
>
> .

Level 4

> Idea : It is similar to level 3. Except the monster movement,we apply A* to monster movement to find and kill pacman. And the monster is better than pacman if there is no way to get out :). Monster's lv: 100. The behavior's movement of level 4 pacman is similar with level 3.

Team schedule:

The prepare

- Our old Vietnamese man used to say : `“Đời Xây dựng chỉ toàn vôi với vữa. Lấy bức tường làm điểm tựa tình yêu“.
- While we are young and smart, we should follow what our old man said. So we decided to build **the foundation** first. And the man we believe in this field most is ***Phú Quyền*** and ***Toàn Đoàn. Toàn Đoàn*** took care of **Direction** and ***Phú Quyền*** took care of **Graphics**
- They didn't disapointed us. The foundation they made are the best thing we've ever seen in our entire life.
- Next, we need **materials** (data - map) and we relised ***Đức Tuấn*** is born to do it. So we decided he take care of **Create map**.
- Finally, we have **foundation and materials**. The last thing we need was how to use these materials. ***Trung Nhật*** is the last member, so he decided to take the job - **Read data**

Working

After we have enough conditions (Deadlines and assignments of another course and including this course :<), we start working:

- Phú Quyền not only still working on Graphics (make the pacman, ghost run, calculate score, show the vision in level 3 and 4,...), but also joined with Toàn Đoàn and Đức Tuấn in building level 1 and 2. Trung Nhật is the tester, he runs the program and reported the issues to us. He also join us to fixed it.
- At level 3 and 4. These levels are more difficult than another so we need all members to resolve it together.
    - Phú Quyền took care of Minimax algorithm
    - Đức Tuấn decided to resolve case 3 and joined with Phú Quyền in Minimax
    - Trung Nhật joined with Toàn Đoàn to resolve the case 1 and 2.
    - Toàn Đoàn also improved the A* and Minimax.

The end is near

Our game is almost finished, the last thing we need was stabilization. So we created as many maps as possible to test the game. After tested so many maps, so many cases (and fixed so many bugs), we finally created a stability project and we decided to end here!

Environment:

Our pacman-search program was build on `Python 3.7`, with 3 IDE `Visual Studio 2019`, `Visual Studio Code` on `Windows 10` and `Ubuntu LTS 20.04`.

Using Anaconda as environment control, and pip as package-management system .
First, you need to install Anaconda (or if you are familar with another virtual environment like virtualenv).

Our tree view of `/source` folder:

```
|-- graphic
|   |-- algorithms.py
|   |-- gameController.py
|   |-- graphicsUtils.py
|   |-- main.py
|   `-- pacmanGraphics.py
|-- random_map
|   |-- __init__.py
|   |-- lv2_map_food_in_monster.txt
|   |-- lv2_simplemap1.txt
|   |-- lv3_6monsters.txt
|   |-- lv3_bestmove.txt
|   |-- lv3_longmaze.txt
|   |-- lv3_looter.txt
|   |-- lv3_map_food_in_2monster.txt
|   |-- lv3_monster_keepfoods_athome.txt
|   |-- lv3_nofoods.txt
|   |-- lv3_simplemap2.txt
|   |-- lv3_simplemap.txt
|   |-- lv4_1monster.txt
|   |-- lv4_4monsters.txt
|   |-- lvx_4conner_4monster.txt
|   |-- lvx_retangle_map.txt
|   `-- random_map.py
...
```

Then, in your console/terminal:

```
conda create -n pacman_talang_ai python=3.7
```

Then type `y` for dowload packages confirmed.
Then:

```
conda activate pacman_talang_ai
```

Install `numpy`

```
pip install numpy
```

In the repo's directory:

```
cd source/graphic
```

Run the program:

```
python main.py -i <your-input-map-dir> -l <level>
```

Or you can run `main.py -h` for help.

Packages list and their version:

```
~/Gitworkspace/I2AI-Project-01/source/graphic  master
└─ pip list
Package    Version
---------- ------------------
certifi    2020.6.20
numpy      1.19.1
pip        20.2.2
setuptools 49.6.0.post20200814
wheel      0.34.2
~/Gitworkspace/I2AI-Project-01/source/graphic  master
└─ conda list
# packages in environment at /home/t3bol90/anaconda3/envs/pacman_talang_ai:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                         main
ca-certificates           2020.6.24                    0
certifi                   2020.6.20                py37_0
ld_impl_linux-64          2.33.1                h53a641e_7
libedit                   3.1.20191231          h14c3975_1
libffi                    3.3                   he6710b0_2
libgcc-ng                 9.1.0                 hdf63c60_0
libstdcxx-ng              9.1.0                 hdf63c60_0
ncurses                   6.2                   he6710b0_1
numpy                     1.19.1                   pypi_0    pypi
openssl                   1.1.1g                h7b6447c_0
pip                       20.2.2                   py37_0
python                    3.7.7                 hcff3b4d_5
readline                  8.0                   h7b6447c_0
setuptools                49.6.0                   py37_0
sqlite                    3.32.3                h62c20be_0
tk                        8.6.10                hbc83047_0
wheel                     0.34.2                   py37_0
xz                        5.2.5                 h7b6447c_0
zlib                      1.2.11                h7b6447c_3
```

Conclusions:

Our results suggest that at least level 1&2 of Pacman can be beaten quite easily with A* (or any search strategies with same behaviour) and decision tree.

But level 3 is truly at the higher rank. For each case, we use different search strategies to decide the safeliest, smartest as next move.(case 1 : choose random but in fact we use BFS to get vision map then update it to viewed map, case 2: A* to calculate the path to closest food we have found, case 3: use minimax to play the mini game with monster (Look how brave we are!!))

So we have done level 3 so far, level 4 is quite easy. It's level 3 but the monster is chasing behind you. And minimax can handle these problem. (At the begining, we have a idea to deal with this by using Hill Climbing search but the time of this project is not enough for us to do this idea).

Estimating the degree of completion level for each requirement:

Level	Completion
1	100%
2	100%
3	89.751%
4	89.752% (Because our algorithms work better with smart monster!)

This project challenge our patience and creating an environment where we aren’t only allowed to succeed but also to fail. The result is not the best of agent pacman but it's our teamwork.

Acknowledgments

To all teacher assistants of this course,

We would like to thank you for this semester, specialy in this course. For your patience and for creating an environment where we aren’t only allowed to succeed but also to fail, for replying every questiton of us. Students succeed when students feel the freedom to imagine and trust you give [4].

This class was challenging at times but there was value in being exposed to the material. And this project is really challenging at all. But with all of our efforts, teamwork spirit and all of your supports, we did it. We wish that this project's due date is longer. It's could be a better result.

At the end, we would like to acknowledge support for this project by my teacher - Master Le Ngoc Thanh (University of Science - VNU). Special thanks to our seniors at University of Science for provided insight and expertise help us finish this project. Thanks to our teamates, we did it! Together.

References:

1. CS221 ↩

2. GameProgramming/Heuristics ↩

3. Chia tay sớm bớt đau khổ. ↩

4. Designing an Interactive Learning Environment to Support Children's Understanding in Complex Domains ↩