Minesweeper
AI Project

By Justin Chung, John Lu, Sui Ting Yeung, Jian Li
Directed By Abdullah Younis

For

CS-171

Introduction to Artificial Intelligence

Student Booklet

Table of Contents

Introduction

In this programming assignment, you will be tasked with implementing a Minesweeper AI Agent, which should be able to play and solve the Minesweeper game. You will have the choice of programming in C++, Java, or Python. Much of the code has already been written for you, however, you will be asked to edit one or more files from whichever shell you decide to use. Your agent should be able to take in percepts and act accordingly. Your grade will depend on your agent's performance measure. At the end of the quarter, your agent will compete against your peers' agents in a class-wide tournament.

Minesweeper Game Mechanics

This version of Minesweeper is based on the classic computer game. However, there are a few differences designed to better evaluate your agent which will become apparent as you play with and familiarize yourself with the game. In Minesweeper, you are given a board that is set up as a 2D grid of tiles. Each tile covers either: (1) a hint number or (2) a mine. Ultimately, your agent's goal is to uncover all tiles which do not contain a mine. A more concrete definition of the game is given by the following PEAS description.

Performance Measure

- The performance measure of your agent will be a score calculated based on number of boards your agent has completed. Points are awarded to your agent only if it successfully solves the entire board:
    - Beginner: +1
    - Intermediate: +2
    - Expert: +3
- The game ends when your agent chooses to leave the game or if your agent uncovers a mine. Your agent does not lose any points for not finishing a board.

Environment

- Each difficulty has a different board dimension and number of mines:
    - Beginner: 8x8 with 10 mines
    - Intermediate: 16x16 with 40 mines
    - Expert: 16x30 with 99 mines
- The board will begin with one random tile already uncovered and presumably safe.
- Mines are randomly placed throughout the board.
- Your agent dies when it uncovers a mine.

    An example 8x8 Minesweeper run by option –rd attached at the end of this document.

Actuators

- Your agent has 4 moves:
    - (1) The action UNCOVER reveals a covered tile.
    - (2) The action FLAG places a flag on a tile.
    - (3) The action UNFLAG removes a flag from a tile if that tile has a flag.
    - (4) The action LEAVE ends the game immediately.
- The actions UNCOVER, FLAG, and UNFLAG are to be coupled with a pair of coordinates which allows the agent to act on a single tile.

Sensors

- Your agent will receive only one percept:
    - Following an UNCOVER action, your argent will perceive the hint number associated with the previous UNCOVER action. This number represents how many mines are within that tile's immediate neighbors.
    - Following a FLAG or UNFLAG action, your agent will perceive -1.

Tasks to Complete

Setup Your Environment

In this section, you will find help setting up your coding environment. This project will take advantage of UCI's openlab; any other coding environment is not supported.

Install Required Applications

To connect to openlab, you will need to use SSH. SSH stands for Secure Shell. It is a program designed to allow users to log into another computer over a network, to execute commands on that computer and to move files to and from that computer. A Mac user can use the terminal application, whereas, a Windows user will need to install PuTTY. You can download PuTTY from here. Download the MSI installer for Windows, and run the installer for PuTTY.

Connect to Openlab

Connecting to openlab is as easy as SSHing into the open lab server. If you are on Windows and using PuTTY, type “openlab.ics.uci.edu” into the Host Name box; make sure the port is 22 and the SSH flag is ticked. Click open, and login using your ICS account info. If you are using Mac, open the terminal found under Application -> Utilities. Enter ‘ssh yourICSusername@openlab.ics.uci.edu’ and login using your into ICS account.

Download the shells on Openlab

To download the shells on Openlab, you will use Git. On openlab, whether through PuTTY or terminal, execute the following git clone command:

Git clone addrees….

Extra Information about Openlab:

- http://www.ics.uci.edu/~lab/students/#unix
- https://www.ics.uci.edu/computing/linux/hosts.php

Extra Information about UNIX:

- https://cgi.math.princeton.edu/compudocwiki/index.php?title=Documentation_and_Information:Getting_started_with_Linux

Program Your AI

Once you have your environment setup, you can start to program your agent. In the ‘src’ folder of your shell you will find the source code of the project. You are only allowed to make changes to the MyAI class.

Compile Your AI

Compiling your program is easy as executing the command make from the shell’s root directory (the directory with the makefile in it).

Test Your AI

To run your program after you have compiled it, navigate to the bin folder. You should find the compiled program inside. Refer to the Shell Manual Appendix for help running it. To generate large amounts of worlds to use with the folder option, refer to the World Generator. If you are using the Python Shell make sure you are using Python 3.5.2. On openlabs, run the command module load python/3.5.2 to load Python 3.5.2.

Write Your Project Report

Write a report according to your Professor's instructions. Make sure your report is in pdf format and place it inside the 'doc' folder.

Submit Your Project

At this point you should have your most up-to-date source code in the 'src' folder, your report in pdf format in the 'doc' folder, and your compiled project in the 'bin' folder. Navigate to your shell's root directory and execute the command make submission. It will ask you for some information and create a zip file inside the folder. Submit this zip file to EEE or Canvas.

Understanding the Tournament

After you submit your project and the deadline passes, you will be entered into a tournament with your classmates. The tournament checks to make sure you followed all the instructions correctly, then runs your agent across 3000 worlds where three different worlds of difficulty level are randomly distributed with 1000 each. Every agent is run on the same 10000 worlds to ensure fairness. Your agent's total score is calculated and a scoreboard is constructed that will be made available. Your agent will be timed-out if it hangs for longer than 2 hours. After the scoreboard is constructed, scores are checked for any illegal submissions. These include two agents with the same score.

Deadlines

For this project, you will have three deadlines throughout the quarter, each of which build on top of each other. With each deadline, your agent should become smarter as you implement more strategies to solving the game. The deadline breakdown is as follows:

- Deadline 1:
    - Complete 200 out of 1000 worlds of size 5x5 with 1 mine
- Deadline 2:
    - Complete 300/1000 beginner worlds (8x8 with 10 mines) and 150/1000 intermediate worlds (16x16 with 40 mines) for a cumulative score of 600 points
- Deadline 3:
    - Complete 500/1000 beginner worlds (8x8 with 10 mines) and 350/1000 intermediate worlds (16x16 with 40 mines) for a cumulative score of 1200 points

For deadlines 2 and 3, every agent will be tested on the same set of worlds to ensure fairness. This set will contain 1000 beginner, 1000 intermediate, and 1000 expert worlds. Should your agent exceed 2 hours while running on the tournament set, you will lose points for that deadline.

Notes

Although each of deadline requirements measures only on the completion level, the ranking of tournament is based on the score that AI gets from world sets.

Appendix: Shell Manual

Name

The command line name used to invoke this program will change depending on the shells:

python3 Main.pyc	if using python shell
java -jar mine.jar	if using java Shell
./Minesweeper	if using cpp shell

Synopsis

Minesweeper_World [Options] [InputFile] [OutputFile]

Options

-m	Use the ManualAI instead of MyAI. If both –m and –r specified, ManualAI will be turned off.
-r	Use the RandomAI instead of MyAI.
-d	Debug mode, which displays the game board after every move.
-v	Verbose mode, which displays name of world files as they are loaded.
-f	Depending on the InputFile format supplied, this operand will trigger program **1)** Treats the InputFile as a folder containing many worlds. The program will then construct a world for every valid world file found. The program to display total score instead of a single score. The InputFile operand must be specified with this option **2)** Threats the inputFile as a file. The program will then construct a world for a single valid world file found. The program to display a single score.

Operands

InputFile	A path to a valid Minesweeper World file, or folder with –f. This operand is optional unless used with –f or OutputFile.
OutputFIle	A path to a file where the results will be written. This is optional.

Examples

Minesweeper_World	Constructs a random 8x8 with 10 mines world, runs the MyAI agent on the world, and prints output to console.
Minesweeper_World -m	Constructs a random 8x8 with 10 mines world, runs the ManualAI agent on the world, and prints output to console.
Minesweeper_World -d	Constructs a random 8x8 with 10 mines world, runs the MyAI agent on the world, and prints output to console. After every turn, the game pauses and prints the current game state to the console.
Minesweeper_World -r	Constructs a random 8x8 with 10 mines world, runs the RandomAI on the world, and prints output to console.
Minesweeper_World -rd	Constructs a random 8x8 with 10 mines world, runs the RandomAI agent on the world using debug mode, and prints output to console. After every turn, the game pauses and prints the current game state to the console.
Minesweeper_World -f /path/to/world/file.txt	Constructs the world specified in the file, runs the MyAI agent on the world, and prints output to console.
Minesweeper_World -f /path/to/world/files/	Constructs all the worlds specified in the folder, runs the MyAI agent on all the worlds, and prints output to console.
Minesweeper_World -f /path/to/world/files/ /path/to/outputfile/yourscores.txt	Constructs all the worlds specified in the folder, runs the MyAI agent on all the worlds, and write output to txt file.

Minesweeper_World -fv /path/to/world/files/	Constructs all the worlds specified in the folder, runs the MyAI agent on all the worlds, and prints output to console. Before running every world, print out a message indicating which specific world is running.

Notes

The Python shell uses Python version 3.5.2.
When using debug mode or ManualAI, the board will printed to the console. Each tile is represented as a full stop potentially followed by a series of characters. Every board that get displayed starts with 1-indexing and bottom left

This example is demonstrated by random AI of debug mode running in a beginner world with three sequential actions:

```
---------------- Game Board ------------------

8   |       .       .       .       .       .       .       .       .

7   |       .       .       .       .       .       .       .       .

6   |       .       .       .       .       .       .       .       .

5   |       .       .       .       .       .       .       .       .

4   |       .       .       .       .       .       .       .       0

3   |       .       .       .       .       .       .       .       .

2   |       .       .       .       .       .       .       .       .

1   |       .       .       .       .       .       .       .       .

            _       _       _       _       _       _       _       _
            1       2       3       4       5       6       7       8

------------------ Percepts -------------------
Tiles Covered: 63 Flags Left: 10    Last Action: Uncover on tile 8 4
Press ENTER to continue...
```

```
---------------- Game Board ------------------

8   |       .       .       .       .       .       .       .       .

7   |       .       .       .       .       .       .       .       .

6   |       .       .       .       .       .       .       .       .

5   |       .       #       .       .       .       .       .       .

4   |       .       .       .       .       .       .       .       0

3   |       .       .       .       .       .       .       .       .

2   |       .       .       .       .       .       .       .       .

1   |       .       .       .       .       .       .       .       .

            _       _       _       _       _       _       _       _
            1       2       3       4       5       6       7       8

------------------ Percepts ------------------
Tiles Covered: 63 Flags Left: 9    Last Action: Flag on tile 2 5
Press ENTER to continue...
```

```
---------------- Game Board ------------------

8   |       .       .       .       .       .       .       .       .

7   |       .       .       .       .       .       .       .       .

6   |       .       .       1       .       .       .       .       .

5   |       .       #       .       .       .       .       .       .

4   |       .       .       .       .       .       .       .       0

3   |       .       .       .       .       .       .       .       .

2   |       .       .       .       .       .       .       .       .

1   |       .       .       .       .       .       .       .       .

            _       _       _       _       _       _       _       _
            1       2       3       4       5       6       7       8

------------------ Percepts ------------------
Tiles Covered: 62 Flags Left: 9    Last Action: Uncover on tile 3 6
Press ENTER to continue...
```

```
---------------- Game Board ------------------

8   |       0       0       0       1       1       1       1       *

7   |       0       0       0       1       *       1       1       1

6   |       1       1       1       1       1       1       0       0

5   |       3       *       2       0       1       1       1       0

4   |       *       *       2       1       2       *       1       0

3   |       2       2       1       1       *       2       1       0

2   |       1       1       1       1       2       2       2       1

1   |       1       *       1       0       1       *       2       *

            -       -       -       -       -       -       -       -
            1       2       3       4       5       6       7       8

------------------ Percepts ------------------
Tiles Covered: 61 Flags Left: 9    Last Action: Leave
```

```
WORLD INCOMPLETE

Process finished with exit code 0
```