

Homework 4 - Strategy and Tactics

COS498/598 - Video Game AI - Spring 2025

by Zachary Hutchinson

Due Date: April 11, 2025, Midnight, Brightspace

Goal

Use game AI to:

- analyze the strengths and weaknesses of force distributions.
- coordinate the movement of multiple units.
- decide force composition.

Submission:

You are required to submit the following items.

1. An AI for the game.
2. A write-up describing your AI and any other modifications.

AI Requirements

First, please submit all game files...even those you didn't modify. This includes any resources distributed with the game. In other words, I should be able to run what you submit without having to add files.

Second, you are only required to write one AI (to be used by all factions). The AI must, at a minimum, build units according to some plan and move units according to some strategy. You are welcome to differentiate the AIs of the different factions but it isn't required.

Third, you are welcome to use some degree of randomness in your AI, however the AI's build and move strategies cannot rely substantially on randomness. In other words, *why* and *how* the AI made and carried out a particular choice must be the result, in part, of decision logic.

Write-up Requirements

The write-up should contain a detailed description of your AI. Include (if applicable), the general approach, build and move decisions, and low and high level decisions.

Do you make use of terrain? How do you go on the attack? How do you defend? How do you identify targets? How do you assess the strength of an AI's position? Or enemy position?

Also, if you've made modifications to the game itself, please include those as well.

To sum up: I will read your write-up first when grading. Once I read it, I will run the game a few times. The write-up should explain what I'm seeing.

The Game

The game is a typical strategy game based (strongly) on the rock-paper-scissors concept common to so many games. The map consists of cells (default 40x30). Each cell has a terrain which impacts combat asymmetrically.

Factions possess cities which produce income. Income is used to purchase units. All cities in the default game are identical. Cities are displayed by a cell outline in the color of the owning faction.

There are three unit types (utypes) in the game:

- R: rock
- S: scissors
- P: paper

They are displayed on the map with those four capital letters. The unit types are identical except in combat. Combat involves rolling a d10 (0 to 10) for both the attacker and defender and deducting the rolls from the other unit's health. If one of the two units would be a winner in rock-paper-scissors (for example, R vs S), then the would be winner (R in this example) rolls a d20 (0 to 20) while the other still rolls the normal d10.

When a unit loses all its health, it is removed from the game. Combat is rather lethal. It can kill both units.

The game continues until one faction holds all the cities.

Additional Rules Enforced by the Engine

- Two units cannot occupy the same map cell.
- If a unit is ordered to move into a cell occupied by a friendly unit, the order will be ignored.
- Combat is initiated by a unit trying to move into a cell occupied by a unit of a different faction.

- A city cannot build a unit if the city is occupied by a friendly unit.
- A faction must have enough money to build a unit otherwise the BuildUnitCommand is ignored.
- If your AI script issues more than one BuildUnitCommand but can only afford one unit, only the first encountered will be executed.
- Units can only move once. If issue two MoveUnitCommands for the same unit, only the first encountered will be executed.
- Factions start with a pool of money (default 1000)
- Commands are shuffled before execution. This means they will not be executed in the order your AI creates them or in faction order. (NOTE: I feel this design choice is a bit controversial. You're welcome to turn this off by deleting the shuffle() call at the top of the RunAllCommands() function in main.py if you feel your AI approach would benefit from a predictable turn order.)
- Terrain does not impact movement in the default game (because there are no movement points).

Files

What follows is a description of the files associated with this project. The descriptions below are minimal. The Python3 code files contains a lot of information. I suggest reading through all the comments in all files before starting.

If you don't plan on modifying the game itself, you probably won't need to edit any files other than ai.py.

NOTE: You are welcome to create more files. For example, if you want to offload AI code to some other module.

ai.py

This is where most (if not all) of your code will go.

cell.py

Defines the map cell's (squares).

cell_terrain.py

Defines map terrain.

city.py

The city class

command.py

The Command class. You should take a look at this one. Commands are very simple (mostly data) classes. Your AI script will need to create Commands.

faction.py

The faction class.

game_map.py

The GameMap class.

main.py

A bloated file with display, engine and game loop code. Run this to start the game.

params.py

Some constants and functions which define several basic aspects of the game.

unit.py

The unit class. Take a look at this one.

vec2.py

A 2D vector class to track unit and city position.

Using the Game

Here are a list of the hot keys for the game:

- LEFT ARROW: Increases game speed.
- RIGHT ARROW:: Decreases game speed.
- Q: quits the game

Grading

- 30% Write-up
- 60% The AI

Extra Credit

I will award extra credit for substantial, interesting modifications/extensions/overhauls of the game itself. You must have a goal/reason for your modification and they should be well-reasoned. Changing one or two default values is not really interesting.

Credit will be on a sliding scale from 0 points to 10 points.

Examples: (NOTE: these are just estimates of the scope of work per points.)

- 0 pts: a very lazy, minimal change without a core idea behind it. Or you didn't mention something in your write-up.
- 1 pt: extensively modifying the values in the game to create a new experience.
- 2 pts: adding new terrain types
- 3-9 pts: new unit types, or more (and different) Commands, or a more complex resource game. (The points awarded will be based on how many you include).
- 10 pts: complete overhaul/reimagining of the game turning it into a nightmare of complexity and awesomeness.

With one requirement: whatever you add must be used by your AI. If it isn't used by your AI, the modifications don't count.

To get extra credit you must tell me in your write-up, in a separate section, exactly what you did and **why**.