Training a Supervised Learning Agent to Play Halite II

Introduction and Game Mechanics

Halite II is an artificial intelligence programming challenge created in 2017 by Two Sigma. Halite II is simultaneous turn-based game played on a two-dimensional plane with planets. Players control ships, which can dock on planets and attack other ships. Larger planets allow for more ships to dock and provide more Halite, which is used to create more ships. Ultimately, the goal is to overwhelm the opponent by amassing Halite via docking on planets and destroying all of the opponents’ ships.

Setup

The game is played with two or four players—in the implementation of this agent, we will assume the two-player game, potentially expanding to the four-player game if time permits. The board is rectangular, with a 3:2 aspect ratio. The board size can vary from 240 x 160 units to 384 x 256 units. Planets are randomly placed, with a cluster of four guaranteed to be in the center. Other planets are placed symmetrically and can be larger than those found in the central cluster. Players start with three ships and no planets, arranged symmetrically across from the other team.

Gameplay

The game concludes when a bot wins (all enemy ships dead or all planets owned) or the turn limit is reached. Every turn, bots receive the game state and have two seconds to issue commands for every ship. Turns are calculated in the following order:

1. The status of any docked ships is updated.
2. Player commands are processed. For instance, a new thrust command will instantaneously update the ship velocity.
3. Movement, collisions, and attacks are resolved simultaneously.
4. Damage from combat is applied.
5. Planet resources/ship creation is updated.

Ships have 255 integral health points, zero to seven integral velocity in any direction, and a weapon that deals 64 damage per turn with a range of five units. Ships move with a thrust command, with an angle in integral degrees and an integral velocity. A command must be given every turn to move the ship.

Planets have a radius and health associated with it. They have unlimited resources, but a maximum number of ships can dock on it depending on its radius. Only one team can dock on a ship. A ship can dock on a planet once it is within 4 units of a planet, and it will fully dock in five turns. Once docked, the ship begins to mine the planet. Each ship docked produces six units per turn, and 72 units are required to produce a ship, which spawns two units from the planet. The ship fails to spawn if there is not enough space near the planet. Finally, undocking takes five turns and returns control of the ship to the owner. A ship that is docking, docked, or undocking cannot fire its weapon.

Ships can damage other planets and ships in a few ways.

First, a ship can collide into a planet, dealing damage to the planet equal to the amount of health the ship had. A planet has 255 health for every unit of radius it has—a full-health ship has 255 health, so a planet can withstand one collision with a full-health ship for every unit of radius it has. A ship can also collide into another ship, and both will die.

Ships can fight other ships once they come within the five-unit weapon radius, dealing 64 units of damage per turn. If multiple enemy ships are within range, the damage is split evenly among them. Ships may only fire its weapon once, which applies when there are multiple enemy ships that come within range within one turn. A ship that moves toward multiple enemy ships fires at the first one it encounters within the turn, and does not fire at others. If it encounters multiple at the same time, then it splits the damage evenly.

**Background and Motivation**

One of the greatest breakthroughs in the field of artificial intelligence is AlphaGo, which beat top players in Go, a game with immense complexity using supervised learning and reinforcement learning. This project is motivated by the
first portion of the supervised learning part of AlphaGo. With the restriction that this project is an undertaking by an undergraduate with limited resources, a full reinforcement learning agent to defeat the top bots is likely impossible. However, a supervised learning agent that can defeat an average bot is certainly possible, and useful as a setup for a potential reinforcement learning agent in the same way AlphaGo is set up.

**Project Goals**

Ultimately, the goal is to produce a supervised learning agent that plays Halite II as well as possible. The quality of the agent will be measured against the ML Starter Bot provided by Two Sigma, unless it is quickly evident that the supervised learning agent is vastly superior. In that case, it will be compared to previous versions of itself, motivated by comparisons of the improved AlphaZero to AlphaGo.

If the supervised learning agent proves to be successful, a potential stretch goal is to expand to the full AlphaGo methodology, and using the supervised learning agent to implement a reinforcement learning agent to compete with top bots.

**List of Deliverables**

- Project description
- Collection of replays
- Source code of supervised learning agent
- A final report detailing and documenting the agent’s approach and performance

**Timeline**

**Week 1 (9/19):** Submit project description

**Week 1 – 4 (9/19 – 10/10):** Amass and collect replays to train on

**Week 4 – 7 (10/10 – 10/31):** Create working version of supervised learning agent that achieves some kind of success against the ML Starter Bot

**Week 7 – 11 (10/31 – 11/28):** Improve on agent and finalize it

**Week 11 – 12 (11/28 – 12/5):** Complete final report