Group 6 Term Project III: Milestone Report #1

Tiny Islands (2019) https://dr-d-king.itch.io/tiny-islands

Github Repository: https://github.com/xuhongkang/tinyislands

Alex Ma, Andrew Sayegh, Diego Valdivia Cox, Hongkang Xu

Current Progress:

- Implemented game model with all scoring and moving features, including a game state representation, board representation, tile representation and terrain type representation. https://github.com/xuhongkang/tinyislands/blob/main/src/board.py

- Have received initial feedback and guidance from TAs on possible approaches, models and algorithms. Also received approval on current progress and milestone requirements.

- Decided as a group to simplify and focus on solving the reduced and relaxed problem, which in essence is a turn-based tile generation simulator.

Meeting Notes:

Removing island drawing means removing water, beach and ship terrains.
We could just have 1 choice for the tile for the next submission and maybe expand on it if we have time.
It may be generally useful for procedurally generated games.
It could definitely be adapted to many things.
Do we want to have tiles just to worry about adjacent tiles or the whole map?

Proposed Changes:

- Smaller Board, 5x5 or 7x7
- Less Number of Turns - 10 for 5x5 or 16 for 7x7
- Simple Terrain Generator given a fixed number of randomized choices (limited positions (1 for each), two terrain types from which to choose)
- Change square tiles to hexagon tiles, increase the amount of adjacent and corner tiles.

- Create a relationship between scoring and number of tile sides?
- Abstract Tile (types of terrain tiles)

Proposed New Tile Types and Associated Scoring Logic:

1. Type A
    a. Gives +2 score if there's not any other Type A sharing its row and column
2. Type B
    a. Gives +1 score for each Type B adjacent to it
3. Type C
    a. Gives +1 score for each Type A or Type B near it
    b. Gives -2 score for each Type C adjacent/near it

Explanation: After careful consideration, we believe that the above 3 types of tile design best capture the many features of terrain tiles in Tiny Islands, and will be the candidates for the tiles in our simplified model. This simplified model, whose mechanics will be introduced below, will become the environment in which our agent will be trained for Milestone 2.

Proposed New Rules and Mechanics:

1. The **Board** will consist of 25 (initially empty) **Tiles** ordered in a 5x5 grid.
2. A **Tile** is a unit located on the grid with a unique 2d **Position**. Tiles are initially empty, but may be assigned specific **Types**. If a Tile has already been assigned a Type, it cannot be reassigned. (This feature might conflict with MDP principles since the Seed will have to offer alternative Choices if the tile at the initial target position has already been assigned. This rule might be relaxed further in the future.)
3. **Position** is a unique identifier for Tiles denoting its 2D position on the **Board**. For a 5x5 Board, Positions may range from (0, 0) to (4, 4) where (x, y) represents the Tile in column x on row y on the Board.
4. A **Type** is a specific typing for a Tile that represents a unique piece of scoring logic. Scoring logic computes the individual score of a Tile, based on information about adjacent (directly connected in at most 4 directions) or near (directly connected in at most

4 directions) Tile to the reference Tile. Tiles with different Types score differently individually.

5. At the start of the **Turn**, the agent chooses from a **Choice**. A **Choice** consists of two sets of Tile **Type**-**Position** pairs. By selecting one of the sets, the state progresses to the next state by changing the **Type** of the **Tile** at the paired **Position** to its target **Type**.
6. A **Game** of ~~*Tiny*~~ *Mini Islands* consists of 10 turns. The final score is calculated by adding up all the individual scores of **Tiles** based on their **Type**.

Proposed Searching and Optimization Algorithms:

The below searching and optimization algorithms may be extensively tested to train our agent for Milestone 2's objectives.

1. Breadth First Search to generate full search space and evaluate performance.
2. Simple Pruning via devaluating negative rewards to optimize performance.
3. Depth Limited Search via our built in evaluation function for testing purposes.
4. Prospects for model based reinforcement learning with scaled up models.