WEC 2019 Competition

Mitch Adam, Kyle Hennig, Nayan Prakash, and Ryan Shukla.
Team Osborne Village.

Catch Basins

18th January 2019

OVERVIEW

Our intent is to create a game similar in style to Minesweeper capable of simulating random distributions of catch basins. It will then allow a user to attempt locating all the empty spaces on the grid without accidentally hitting a catch basin.

Finally, the game will be played by a bot capable of solving the game.

GOALS

- Level 1: Backend
- Level 2: Create a User Interface
- Level 3A: Accept User Input (Game Play)
- Level 3B: Accept User Input (Customizable)
- Level 4: Play the Game

SPECIFICATIONS

Technologies

- Tornado web server
  - We chose Tornado because it is modern, scales well, and has good support for websockets
- JavaScript client using Pixi.js
  - We chose Pixi.js because it is fast at rendering 2D graphics
- JSON protocol over websockets
  - We chose the websocket protocol because it is well suited to persistent connections

- We choose to use web technologies so that the game could be cross-platform and even mobile-friendly

Client (client -> server)

- JOIN
  - Requests to join the game.
  - Params: { size: number, seed: number }
  - Returns: { success: boolean }
- BOARD
  - Requests a copy of the board.
  - Params: N/A
  - Returns: { success: boolean, board: [[]] }
  - board is a multidimensional array indexed [x][y].
  - Each element of board is a JSON object:
  - {visited: true/false, basin: true/false, adjacent: <number>}
- MOVE
  - Makes a move.
  - Params: { x: number, y: number }
  - Returns: { success: boolean }

Server (server -> client)

- DONE
  - Used to return data after an operation has completed.
  - See the “Returns” section for each client request.

Division of Work

- Backend (Python tornado server and game logic)
  - Kyle
  - Nayan
- Frontend (Javascript client and graphical interface)
  - Mitch
  - Ryan
- Networking
  - Kyle
- Bot
  - Nayan

MILESTONES

Create the web server and basic game logic
Done!

Create the user interface using web technologies
Done!

Establish communications between the client and the server
Done!

Develop bot
Done!

Data Structures and Algorithms

- Wrapper Objects
    - We used wrappers over the Pixi rectangle object to allow us to add metadata such as which nodes had been accessed, basins, etc
- Arrays
    - Used 2-D arrays to keep track of the board
- Space Selector Algorithm
    - Upon selecting a space, that space is now set as checked and its counter of amount of adjacent catch basins is set to zero
    - Then, all adjacent spaces are checked if they are catch basins, and if they are, a counter of adjacent catch basins is increased and a flag is set to prevent recursion
    - If the counter of the amount of adjacent catch basins is still zero, the algorithm will recurse upon adjacent spaces that are not set as previously checked
- Bot Algorithm
    - The bot will check if the amount of adjacent catch basins is equal to the amount of adjacent unchecked spaces. If this is the case, the bot will know all adjacent unchecked spaces ARE catch basins
    - Then, the bot will iterate through all spaces and find spaces where the amount of known adjacent catch basins is equal to the amount of adjacent catch basins. On these spaces, if there are any adjacent unchecked spaces that are not known catch basins, the bot will know this space is NOT a catch basin
    - Then, the bot will iterate through all spaces and find the last (most bottom-right) space where the space is known to not be a catch basin and select it
        - If there are no known catch basins, the bot will guess the most bottom-right space

Please feel free to contact us for more information about our project or if you need any assistance making it run. Instructions can be found in the README at https://github.com/kylehennig/wec2019.