TEAM SHREK

COMP 472

Our Heuristic Function - Explained

- Wanted to prioritize tokens and their positions
- Developed through some trial and error
- Computes board ‘snapshots’ based on token positioning at certain depths
- Quick to compute and evaluate
- Can be improved by taking move consideration (more playtesting), or…
- ...an idea we had: create a database (file) that has the absolute best possible moves compiled after computing at depth 30, for example, and build a graph/structure to get the moves back given a board state
- Heuristic retrospect: content! But room to improve

Our Heuristic Function - Pseudocode

Black Cells = 2x // Diagonal moves - count twice in importance

White Cells = 1x // No diagonal moves - count once in importance

For each cell, check if it has adjacent cells of the same color.

If R has R beside it, +2 for each

If G has G beside it, -2 for each

IF it's a winning board, and winning for Green, result is 999999

IF it's a winning board, and winning for Red, result is -999999

Our Search Function - Explained

- Standard alpha-beta pruning
- Initially was minimax algorithm implementation
- Alpha-beta truly helps getting faster results, allowing for more complex heuristics
- Depth of ‘n’, for the tournament we set it to depth 4
    - Originally wanted to go to depth of 5, but search time averaged around 2.7s, sometimes slightly over 3s
- Search retrospect: content!

Tournament Results

- 3 wins, 1 loss, 4 ties
  - Overall 5 points result
- 3 wins - enemy AI taking over 3s
- 1 loss - error in our code to evaluate winning conditions
- 4 ties - equally matched AI heuristic and search capability

Finally...

- Pros:
  - Tournament result was good
  - Tournament flowed nicely
  - Heuristic & search were good
  - Team workflow was good
  - C++ definitely a positive
- Cons:
  - Error in code - costed us wins
  - Lack of time to improve AI & heuristic (database idea)
  - Poorly structured code (all in one file)