Theory of Computer Games (Fall 2020) Homework 2

NTU CSIE

Due: 14:20 (UTC+8), December 24, 2020

Outline

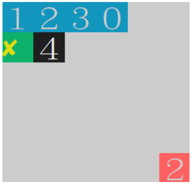
Einstein Würfelt Nicht! (Dame)

Game Rules

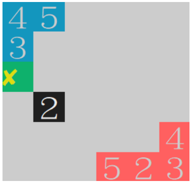
1. The game is played on a 6x6 board. Initially, there are 6 red and 6 blue pieces located at top left and bottom right consecutively.
2. The initial pieces configuration is random.
3. In a turn, player can move any piece of its own one square forward in one of the three directions closer to the opponent's corner.
4. A player can **capture other pieces** by landing on their square and then replacing them. Note that a player is allowed to capture a piece of its own.
5. A player can pass **if and if only** there is no legal move available

Terminal Condition

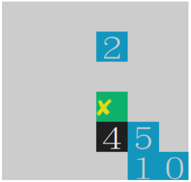
- The objective is to get **as many pieces as possible** to the opponent's corner **or** to remove all of opponent's pieces from the board.
- If the number of Red pieces at corner is equal to the number of Blue pieces at corner, player with the highest corner piece number wins
- Corner: one piece located at corner and the other pieces are connected to it



5 Blue pieces at corner Blue won

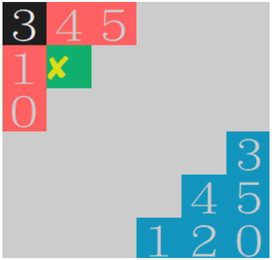


same number of pieces, $4 > 3$ Blue won

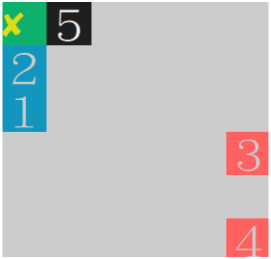


no red pieces left Blue won

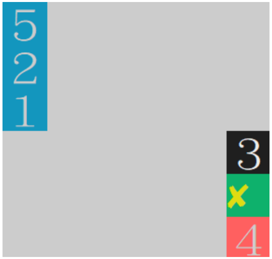
Einstein Würfelt Nicht! (Dame)



Initial board
Red: E, S, SE
Blue: W, N, NW



Blue can only capture its own piece



Blue won!

Let's Play

Compilation

- Run `make` under `hw2` directory.
- It'll generate 4 executables: `game`, `random`, `conservative` and `greedy`.
- `game` is the main gaming environment, while the others are baseline agents.

How to Play

- The game supports AI-AI, AI-human and human-human mode.
- You can choose which mode to play by specifying `[-f]` (first player) and/or `[-s]` (second player).
- For example, the following command runs random and vs human mode

```
$ ./game -f ./random
```

Outline

Requirements

HW Requirements

1. Implement an agent of modified **Einstein Würfelt Nicht! (Dame)** using Monte-Carlo Tree Search.
2. Beat the conservative AI and the greedy AI.
3. Analyze the performance of your agent

Part I: Einstein Agent

Basic Requirements

1. Write an agent that receives opponent's last move (from `game`) and sends move accordingly back.
2. We've handled most parts of the communication. Receive messages by reading from `stdin` and send messages by writing to `stdout`.
3. Read everything character-by-character: if you expect a message of length k, read one character k times instead of directly reading a string of length k.
4. Remember to flush every time after writing a message to `stdout`.

Part I: Einstein Agent (Cont'd)

Basic Requirements

1. You can assume that every move your agent receives is valid.
2. Your agent should send a valid move within 10 seconds. If `game` receives an invalid move, or doesn't receive a move within the time limit, your agent will be killed and your opponent wins immediately.

Message Format

Message Format

R: Received, S: Sent

- R_1: 12 characters that denote initial pieces configuration, e.g. 345120345120 (see illustration)
- R_2: A single character
  - `'f'`: you are the first player
  - `'s'`: you are the second player
- R_3: 2 characters, can be "`--`" (pass) or *nd* (otherwise), where
  - $n =$ number of piece to be moved
  - $d =$ direction: 0 (vertical), 1 (horizontal), 2 (diagonal)
- S: 2 characters, can be "`--`" (pass) or *nd* (otherwise) only.

Frame of an Agent

```
while $True$ do
    receive $R_1, R_2$
    $B \leftarrow$ Board($R_1$)
    $myTurn \leftarrow R_2 ==$"f" ? $True$ : $False$
    while $True$ do
        if $B$.terminal() then
            break
        end if
        if $myTurn == False$ then
            receive $R_3$
            do opponent's move $R_3$ on $B$
        else
            choose a move $S$
            do the move $S$ on $B$
            send $S$
        end if
        $myTurn \leftarrow !myTurn$
    end while
end while
```

Algorithms

1. You are required to implement the following algorithms:
   - UCT tree search with tree expansion based on UCB score
   - Add **at least one** of Progressive Pruning (PP) **or** RAVE
2. Singe core, and no more than 4GB RAM
3. You can add plug-in learning + training data, but the training needs to be done by TA in 30 minutes using hardware described above
4. Your agent will be tested by

```
$ ./game -f [your_agent] -s [our_agent] -r 5
```

Part II: Agent Performance Analysis

Report Structure

Your report should include but not limited to:

1. Implementation
   - How to compile and run your code in linux. Don't upload the compiled executable file.
   - What algorithm and heuristic you implemented.
2. Experiments
   - Results and findings of your implementation
3. Discussion
   - Observe your refinement on UCT tree search and (PP/RAVE), try to measure the improvements.

Outline

Submission

- Directory hierarchy:
  - student_id // e.g. r08922166 (lowercase)
    - Makefile // make your code
    - src // a folder contains all your codes
    - report.pdf // your report
- Compress your folder into a zip file and submit to https://www.csie.ntu.edu.tw/˜tcg/2020/hw2.php.
- Due to server limitation, the file size is restricted to 2 MB.

Grading Policy

1. Beat the baselines (10 points)
   - Beat Simple Conservative Agent (SCA) (5 points)
   - Beat Simple Greedy Agent (SGA) (5 points)
2. Report (5 points)
3. Bonus
   - Dominate Simple Greedy Agent (SGA)
   - Peer competition
   - Beat Hidden Benchmark

Beat the Baselines

- One round consists of 2 games with alternating first player.
- We will calculate the total net score of 5 rounds between your agent and the baseline agents
- You can get $S, S \in \{-2, -1, 0, 1, 2\}$ score for each round
  - Win: $+1$ point
  - Draw: 0 point
  - Lose: -1 point
- We consider total net score no less than zero as beating the baseline.

Bonus

Dominate SGA

- Get total net score of **strictly more than 2** when playing againts Simple Greedy Agent (SGA) in 5 rounds (+1 point)

Peer Competition

- $N =$ number of HW participants.
- We will host a 5-round game between each participant.
- Get net score **strictly more than** $5 \times (N - 1)$ (+1 point).
- Top K agents will be awarded more points, where K will be decided later, based on the results.

Bonus

Beat Hidden Benchmark

- You will have 3 rounds to fight the hidden benchmark.
- $W =$ net score after 3 rounds
- Additional bonus points of $\min(\max(W, 0), 3)$