CSE4705 - Introduction to Artificial Intelligence December 4, 2019

Project - Kōnane

I have written a server application for you to use to test your game player and compete with your peers. The server is written in Java and runs on a Linux workstation. **Your game player *must* be able to communicate with the server in order to compete in the end-of-semester tournament.**

Kōnane Game Overview

Kōnane is a native Hawaiian game played in preliterate Hawaii. Kōnane was popular among all classes and genders unlike some other games that were taboo to one class or gender. A game sometimes lasted an entire day and often matches consisted of a large number of games.

The Game Board

Kōnane is played on a rectangular game board divided into a grid of squares or indentations for game pieces. Boards of different sizes and dimensions existed. The number of rows ranged from 8 to 13, and the number of columns from 8 to 20. Each position on the game board had a hole or indentation for the game pieces. The center of the board was called piko (navel). The row along the borders of the board was termed kaka'i . Before starting play, the board positions were filled with alternating black and white stones. Local beaches provided basalt and coral pebbles for game pieces, whose preferred size was under an inch in diameter and slightly flattened rather than spherical.

About notation: the "standard" notation for checkers and similar board games numbers the dark squares of the checker board from 1 to 32. Samuel used a different one (1-35, skipping 9, 18, and 27) for internal representation. The Kōnane uses yet another representation for the 18×18 board to your implementation simpler, indexing board positions by row and column. The board rows are numbered from 0 to 17 and columns denoted by the numbers 0 through 17 as shown in Figure 1.

Using the Server

Connecting

The server listens for incoming connections on port 4705. It is currently running on the machine *artemis.engr.uconn.edu*.

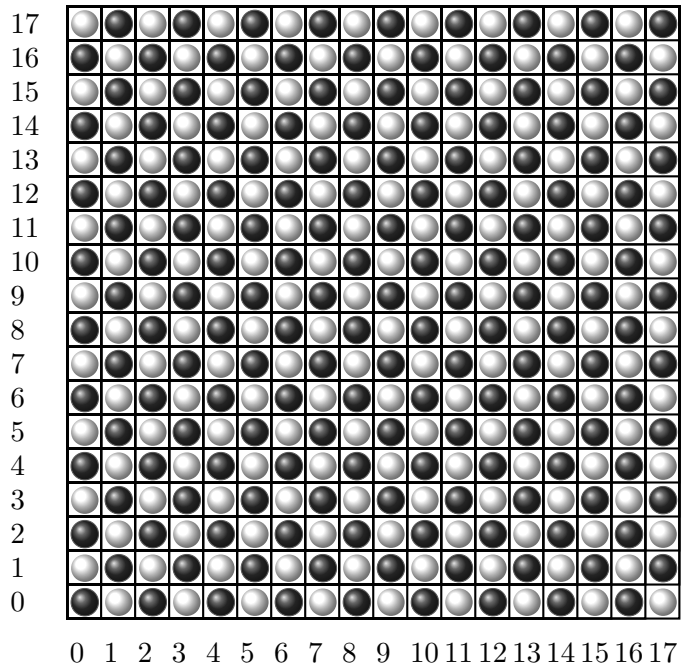

Figure 1: The starting Kōnane board.

Authentication

In order to make things simple, all messages exchanged with the server will consist of a string of characters terminated by a Ctrl M followed by a Ctrl J . (Note: Ctrl M is ASCII 13, written as `'\r'`, in C++, etc.; Ctrl J is ASCII 10, written `'\n'` in C++, etc.). Why use such strange terminators? So that you can experiment with the server using telent! If you're unsure how the server behaves, you can telnet to port 4705 on artemis and experiment.

When you initially connect to the server, it will respond with:

```
You are connected to the Artemis Konane Server v1.0
```

The version number may be different when you connect. You may ignore this line which will be followed by:

```
?Username:
```

Any time a message from the server demands a response from you, it will preface it's question with a question mark. In this case, the server is asking for your username, which is an integer you will be assigned. If your username was 13, you would respond with

```
13
```

The server will then ask for your password:

```
?Password:
```

Again, you are expected to respond with your password, another integer. If your password was 453423, you would respond with

```
453423
```

Finally, the server will ask you who you wish to play:

```
?Opponent:
```

To this you must respond with the username of an opponent. In general, the server will now wait for your opponent to authenticate itself and request *you* as an opponent. When your opponent has authenticated, you will receive a message of the form

```
Game:3812
```

which is your indication that a game is about to begin. The number is a unique tag associated with this game. Next, the server will simulate a coin toss to see which player moves first. If your player wins the toss, you will see:

```
Player:1
?Remove:
```

You must report the piece you've selected to remove in the following format. For example, if you choose to remove the piece in row 0 and column 0, you should send the following string to the server

```
[0:0]
```

The server will report back the color of the piece your player selected for removal. This will be your player's color for the remainder of the game. The server also reports back all moves made in the game, even your player's own moves! For example, if your opponent selects the piece at [0:1] for removal, your player will see the following messages from the server.

```
Color:BLACK
Removed:[0:0]
Removed:[0:1]
?Move:
```

What follows is the game.

Playing the Game

The server will report *every* move made in the game (even your own moves) to you on a new line with a message of the form

```
Move[2:0]:[0:0]
```

The first part of the message indicates the server is reporting a move. The second part of the message, consisting of a string of the form "$[i:j]:[k:l]$", which indicates a move by the piece at position $[i:j]$; the second tuple indicates the position the piece moves to. Since pieces can only move in one direction during a turn (even if executing multiple jumps) only the final position is required. In each pair the first number indicates the *row*, starting from the bottom, and the second number indicates the *column*, starting from the left. (So, the upper left cell of the board is [17:0].) Recall Figure 1.

When it is your turn to play, the server will request a move from you with a message of the form

```
?Move{180000):
```

You are expected to express your move in the same syntax that the server uses to inform you of moves: $[i : j]$... The number appearing in the parentheses is the amount of time you have remaining to play the rest of your moves, in milliseconds. If you do not play in the allotted time, you will forfeit the game.

```
You are connected to the Artemis Konane Server v1.0
?Username:
1
?Password:
1
?Opponent:
2
Game:1
Color:WHITE
Player:2
Removed:[0:0]
?Remove:
[0:1]
Removed:[0:1]
Move[2:0]:[0:0]
?Move(180000):
[2:1]:[0:1]
Move[2:1]:[0:1]
...
```

And the opposing player will see:

```
You are connected to the Artemis Konane Server v1.0
?Username:
2
?Password:
2
?Opponent:
1
Game:1
Player:1
?Remove:
[0:0]
Color:BLACK
Removed:[0:0]
Removed:[0:1]
?Move(180000):
[2:0]:[0:0]
Move[2:0]:[0:0]
Move[2:1]:[0:1]
```

```
?Move(162757):
...
```

Recall that every line is followed by a Ctrl M followed by a Ctrl J but Telnet takes care of that here.

Result reporting, Errors

At the end of the game, you will receive a message such as

```
Opponent wins!
```

or

```
You win!
```

depending on the outcome of the game. In general, errors will be reported on a line of the form

```
Error:(some error text)
```

Errors are non-recoverable, Not that if your opponent plays incorrectly (or runs out of time), the server will report the end of the game to you rather than a new query for a move.