

CSE 4301/5290 Homework 3

Due: 5pm, Wed, October 26; Submit Server: course = ai , project = hw3

1. CSE 4301 only

Connect 4 has a rack with 7 columns, each column has a depth of 6. Each player in turn drops a token into one of the 7 columns. The first player achieving 4 in a row/column/diagonal wins. Each move is specified by a column number.

```
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
-----
0 1 2 3 4 5 6
```

2. CSE 5290 only

4x4x4 3D tic-tac-toe. Each player in turn marks a cell, the first player achieving 4 in a row/column/diagonal wins. Each move is specified by level, row, and column numbers.

	Level 0	Level 1	Level 2	Level 3
	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3
0	X			
1	0	X 0		0
2			X	
3				X

Program requirements:

- (a) You may use Java, C, C++, or LISP. Functions (in LISP) include:

```
; perform alpha beta pruning, return an action
; all actions must be determined by alpha-beta-pruning
(defun alpha-beta-decision (state ...) ...)

; return the utility value of the state
; description of your evaluation...
(defun eval-state (state) ...)
```

- (b) By default, the computer plays with a human. The program asks if the computer or human plays first and the corresponding symbol.
- (c) Each move is specified by a column number for Connect 4, and level, row, and column numbers for tic-tac-toe as in the diagram above.
- (d) Each human move is entered from the keyboard.
- (e) Display each computer move.
- (f) After each computer/human move, display the updated rack/board.

- (g) Each move should not take more than one minute (on the computers in EC 132—CS open lab [tentative location]). Hence, you might want to have a parameter(s) that sets the limit(s) of your search.

For LISP: `compile-file` prepares a compiled version of your program. You need to load the compiled version. Running the compiled version will be faster than interpreting the source.

`get-universal-time` returns the number of seconds since Jan 1, 1970. `get-internal-real-time` returns the number of time units based on `internal-time-units-per-second` (a constant)—might need to handle the “wrap-around” issue: `end-time < start-time`.

Tournament rules:

- (a) October 26th, Wed, 5-6:15pm, EC 132 [location is tentative]
- (b) Undergraduate and graduate students compete separately.
- (c) Your program will play against two other programs, which are randomly assigned.
- (d) Your program starts in one game; your opponent starts in the other game.
- (e) You can get up to 10 points, which constitute 10% of your grade. (win: 5 points; tie: 3.5; lose: 2; non-functioning: 0)
- (f) Your program wins if it functions but your opponent's doesn't.
- (g) Each move cannot take more than one minute. Your program is considered non-functioning if it takes too long.
- (h) Your program is considered non-functioning if it suggests or allows illegal moves.
- (i) If a game takes more than half an hour, the game is a tie.

You might want to aim for a player that makes legal moves before you put in the game strategies. For example, you might have a `next-move` function that returns the first empty spot on the board; later on, you can add game strategies.

CSE 5290 only

3. Q5.9, p197, 3Ed; Q6.1, p189, 2Ed [hardcopy in class or pdf file to Submit Server]