Package 'conn4R'

December 12, 2024

Type Package

Title What the Package Does (Title Case)

Version 0.1.0

Author Vicente Castellar Sebastiá

Maintainer <vic.sebastia@gmail.com>

Description More about what it does (maybe more than one line)
Use four spaces when indenting paragraphs within the Description.

License What license is it under?

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports ggplot2,
Rcpp,
methods

LinkingTo Rcpp

Contents

`bitboards` *evaluación estática de una posición representada en tablero*

Description

evalua una posición mediante criterios estáticos basados en el número de casillas conectadas del jugador en turno

Usage

```
bitboards
```

Arguments

`tablero`	matriz que representa el estado de un tablero
`turno`	qué jugador comienza pa partida: 1 para humano, 2 para IA.

Format

An object of class `matrix` (inherits from `array`) with 6 rows and 7 columns.

Examples

```
tablero <- crear_posicion_aleatoria(18)
visualizar_tablero(tablero)
evaluar_posicion(tablero)
.evaluar_turno(tablero, 1)
.evaluar_turno(tablero, 2)
```

`crear_posicion_aleatoria` *Crear posición aleatoria*

Description

crea un tablero aleatorio de cierta profundidad

Usage

```
crear_posicion_aleatoria(profundidad = 10)
```

Arguments

`profundidad`	número de jugadas que se simulan aleatoriamente

Value

Una lista con los siguientes elementos

- tablero - matriz 6 x 7 que representa el tablero generado
- turnoUltimo - turno del jugador que ha realizado la última jugada en el tablero generado

Examples

```
tablero <- crear_posicion_aleatoria(21)
visualizar_tablero(tablero)
sum(tablero == 1)
sum(tablero == 2)
```

`evaluar_posicion` *bitboarts*

Description

bitboarts

Usage

```
evaluar_posicion(tablero)
```

`hello` *Hello, World!*

Description

Prints 'Hello, world!'.

Usage

```
hello()
```

Examples

```
hello()
```

`iniciar_partida` *Iniciar una partida de conecta 4*

Description

inicia una partida de conecta 4. Se especifica la profundidad de búsqueda y el jugador que inicia la partida

Usage

```
iniciar_partida(profundidad = 5, turno = 1)
```

Arguments

`profundidad`	profundidad de búsqueda del algoritmo minimax.
`turno`	qué jugador comienza la partida: 1 para humano, 2 para IA.

Examples

```
lo siguiente inicia una partida en la que  el jugador humano es el primero en jugar
iniciar_partida(profundidad = 7, turno = 2)

para que sea la IA quien realice la primera jugada:
iniciar_partida(turno = 2)
```

`juego_terminado` *end-of-game evaluation and result*

Description

evaluates whether the game has ended and with what result ("HUMAN WINS", "AI WINS", "DRAW").

Usage

```
juego_terminado(tablero)
```

Arguments

`tablero`	a matrix representing the state of the game board

Details

- If the game is over is TRUE: result can be "WIN HUMAN", "WIN IA" or "DRAW'.
- If the game is over is TRUE: result is NA

Value

returns a list with the following contents

- finalizado: boolean representing the state of the game: TRUE game finished
- resultado: outcome of the game. If the game is over, there are three options

Examples

```
tablero <- crear_posicion_aleatoria(21)
p <- visualizar_tablero(tablero)
print(p)
juego_terminado(tablero)
```

---

jugadas_disponibles *Jugadas disponibles*

Description

dada una situacvión en el tablero de juego, devuelve las posibles jugadas existentes: columnas no completadas

Usage

```
jugadas_disponibles(tablero)
```

Arguments

tablero	matriz 6 x 7 que representa la situación del tablero de juego.

Examples

```
tablero <- crear_posicion_aleatoria(20)
visualizar_tablero(tablero)
jugadas_disponibles(tablero)
```

`minimax` *algoritmo minimax*

Description

función que mediante un algoritmo mini-max la IA decide cuál es su mejor jugada dada una cierta posición del tablero

Usage

```
minimax(
  tablero,
  profundidad,
  maximizandoIA,
  alpha = -Inf,
  beta = Inf,
  env = NULL
)
```

Arguments

`tablero`	a matrix representing the state of the game board
`profundidad`	un entero que fija la profundidad del árbol de jugadas a analizar
`maximizandoIA`	Booleano. TRUE significa que se maximiza la puntuación de la IA FALSE se minimiza la puntuación del jugador humano
`alpha`	parámetro de la poda alpha-beta. Por defecto -Inf
`beta`	parámetro de la poda alpha-beta. Por defecto +Inf

Details

- If the game is over is TRUE: result can be "WIN HUMAN", "WIN IA" or "DRAW'.
- If the game is over is TRUE: result is NA

la poda alpha-beta reduce drásticamente el número de nodos que se evaluan: a profundidad 5, con tres movimientos realizados en el tablero, el algoritmo minimax calcula:

- con poda alpha-beta: 4.677 nodos
- sin poda alpha-beta: 19.607 nodos

Value

returns a list with the following contents

- puntuacion: puntuación optenida al evaluar la posición al realizar la 'jugada'
- jugada: jugada elegida por el algoritmo

Examples

```
tablero <- crear_posicion_aleatoria(3)
kk <- minimax(tablero = tablero, profundidad = 3, maximizandoIA = TRUE)
kk
kk$env$arbol
```

---

turno_humano *solicita una jugada al humano.*

Description

la función invita al jugador humano a introducir una jugada Si la jugada es legal representeala jugada del humano en el tablero mediante un 1 en la casilla que corresponda. La jugada de la IA será representada en el tablero mediante un 2.

Usage

```
turno_humano(tablero, jugada = NULL)
```

Arguments

tablero	matriz 6 x 7 que representa la situación del tablero de juego.
jugada	un entero del 1 al 7 correspondiente a cada una de las 7 columnas si jugada = NULL (valor por defecto), entonces invita al jugador a introducir su jugada

Details

Si la jugada es legal representa la jugada del humano en el tablero mediante un 1 en la casilla que corresponda a la jugada Si la jugada introducida es ilegal, vuelve a solicitar una jugada legal

Examples

```
tablero <- reiniciar_tablero()
tablero <- turno_humano(tablero, jugada = 4)
visualizar_tablero(tablero)
# tambien se puede hacer lo siguiente (no ejecutar):
# tablero <- reiniciar_tablero()
# tablero <- turno_humano(tablero)
# visualizar_tablero(tablero)
```

`visualizar_tablero` *visualiza gráficamente la posición de un tablero*

Description

representa gráficamente el estado de un "tablero"

Usage

```
visualizar_tablero(tablero)
```

Arguments

`tablero`	matriz 6x7 que representa la posición de un tablero

Value

devuelve un objeto ggplot listo para representar gráficamente el tablero

Examples

```
tablero <- crear_posicion_aleatoria(21)
p <- visualizar_tablero(tablero)
print(p)
```

Index