Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores

Programa de Licenciatura en Ingeniería en Computadores

Curso: CE-3104 Lenguajes, Compiladores e intérpretes

Documentación Proyecto Tic-Tac-Toe

Realizado por:

Alexis Gavriel Gómez	2016085662
Emmanuel Aguilar Sánchez	2016009338
Crisptofer Fernández Fernández	2016048367

Profesor:

Marco Rivera

Fecha: Cartago, Mayo 12, 2018

Índice

Introducción

Tic Tac Toe o tres en línea, es un juego entre dos jugadores en donde los jugadores se turnan para colocar X y O (cada uno selecciona el símbolo que utilizará) en una matriz. La victoria de un jugador ocurre cuando uno de estos logra llenar una fila, columna o diagonal con su símbolo.

1.1. Manual de Usuario

El software es una versión del popular juego Tic-Tac-Toe el cual, consiste; en su versión clásica, de completar ya sea, una columna, una fila o una diagonal, con el símbolo que se esté utilizando para jugar. En el juego participan 2 jugadores generalmente, uno juega con “X” y otro con “O”.

Esta versión del Tic-Tac-Toe, cuente con la peculiaridad de que se puede jugar en un tablero de hasta 10 columnas y 10 filas, dicho tablero puede tener todas la combinaciones en las que se desee jugar.

El juego cuenta además con una inteligencia artificial la cual, tratará de ganarle al jugador en todo momento eligiendo la mejor posición disponible en el momento. La computadora siempre usará la “X” como su símbolo mientras que, el jugador usará el “O”.

● Requerimientos para usar el software:

Tener instalado el intérprete DrRacket, y tenerlo configurado para el lenguaje “The Racket Lenguaje”. A continuación, se muestra una guía de cómo configurarlo.

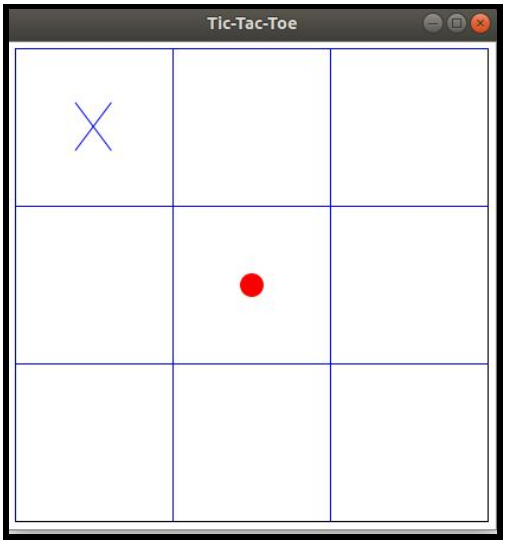
- ❒ En la barra superior de DrRacket, seleccionar el apartado “Lenguaje”.
- ❒ Posteriormente, elegir la opción “Seleccionar Lenguaje”.
- ❒ Finalmente, elegir “The Racket Lenguage”.

● Guía de uso:

En la terminal de DrRacket, se debe escribir el comando (TTT m n) donde, m es la cantidad de filas con las que se va a jugar y n las columnas. Luego se debe presionar la tecla enter. En el ejemplo se va a usar un tablero de 3 filas y 3 columnas.

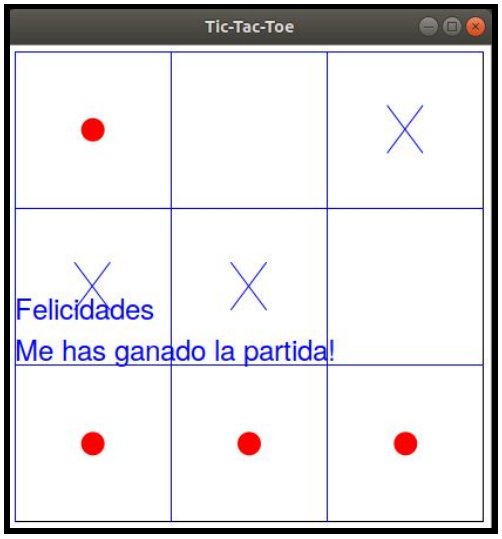
```
Bienvenido a DrRacket, versión 6.12 [3m].
Lenguaje: racket, with debugging; memory limit: 128 MB.
> (TTT 3 3)
```

Una vez abierta la ventana de juego, se debe seleccionar la casilla en la que se va a realizar el primer movimiento. Una vez que se realiza el movimiento, la computadora realizará el suyo.

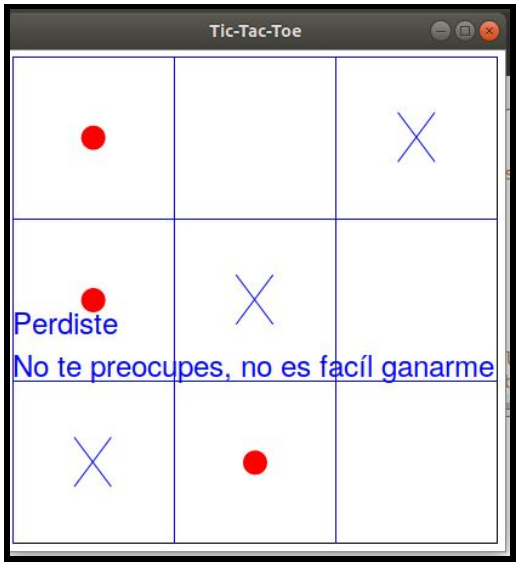


Desde este punto el jugador es libre se realizar cualquier estrategia para intentar ganarle a la computadora, existen tres posibles finales para una partida, a continuación se muestran.

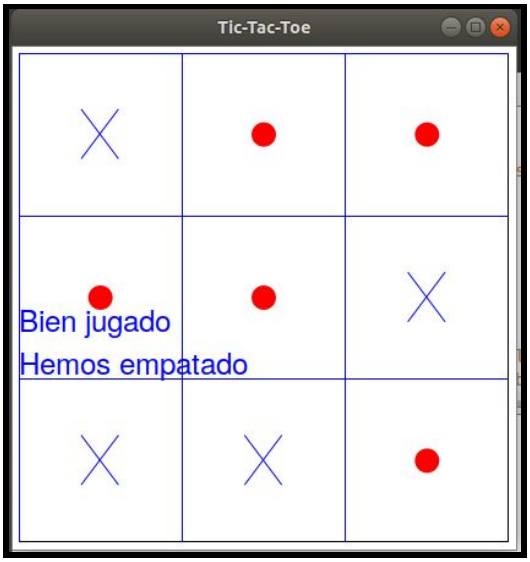
El jugador gana si logra completar una fila, una columna o una diagonal con el símbolo “O”. Para este caso, se necesitó de tres “O” para llenar la fila y ganar.



EL jugador pierde si la computadora logra completar una fila, una columna o una diagonal con el símbolo “X”. Para este caso, se necesitó de tres “X” para llenar la fila y que la computadora ganara.



Para que se dé un empate, todas las casillas del tablero deben estar llenas y no debe existir ninguna fila columna o diagonal, completa con el símbolo de la computadora o del jugador.



Finalmente, cada vez que se desea volver a jugar, se debe seleccionar la opción “ejecutar” el DrRacket. Una vez de reinicia todo, se repiten los pasos anteriores para volver a tener una partida.

1.2. Descripción de las funciones implementadas.

- (greedyAlgorithmn matrix): Esta función provee la estructura de los pasos a seguir para llegar a una solución óptima, la cual es ganar el juego. Ella recibe la matriz de juego y luego se realiza el llamado a las funciones que verifican si el usuario o la pc pueden ganar colocando una ficha más y si no, hace el llamado a la función que se encarga de “pensar” donde colocar la ficha de juego.
- (greedyAlgorithmnAux matrix lista): Su unica función es sustituir la ficha en la matriz de juego.
- (someoneWin? matrix num): Su función es verificar si el usuario o la pc están a una ficha de ganar, esto lo hace sustituyendo una ficha vacía y llamando a la función que verifica si existe un ganador, si no funciona en esta posición repite el ciclo con la siguiente.
- (someoneWinAux i j matrix num): Es la que realiza el procedimiento anteriormente descrito.
- (mejorPos lista): se encarga de buscar en la lista de tripletas (tripleta, corresponde a una lista que contiene probabilidad y posiciones en i y j en la matriz) cual de ellas contiene la probabilidad total menor.
- (mejorPosAux ele lista): realiza el proceso anteriormente descrito.
- (listaProb matrix): Crea la lista de tripletas, sustituyendo cada posición y llamando a una función que calcula las probabilidades.

- (listaProbAux matrix i j lista): realiza el proceso anteriormente descrito.
- (posibilidades matriz num): Es la función principal del conjunto de funciones que extraen las posibilidades, suma los resultados de cada función.
- (transposex matriz): Traspone la matriz.
- (remove_colx matriz): Remueve la primera columna de la matriz.
- (horizontalx matriz num posibilidad): Toma una fila de la matriz y verifica si todos los elementos de ella son el mismo, en este caso que sean iguales a num y si es así, suma 1 a la la entrada llamada posibilidad.
- (verticalx matriz num posibilidad): llama la función de horizontal con la matriz traspuesta para verificar si una columna está llena de num.
- (get_columnx matriz): Devuelve la primera columna de la matriz.
- (linea_hx lista num): verifica si todos los elementos de una lista son iguales.
- (invertirx lista): invierte una lista.
- (diagonalx matriz num): Verifica si la diagonal “\” de la matriz está llena de num o es una posibilidad de ganar a futuro.
- (diagonalesx1 matriz num posibilidad): llama a la función anterior para comprobar si es posibilidad la diagonal “\”.
- (diagonalesx2 matriz num posibilidad): Llama a la función anterior pasandole la lista invertida para comprobar si es posibilidad.
- (diagonalesx matriz num posibilidad): función de análisis de diagonales principal, se encarga de contar las posibilidades que se hallan en cada diagonal.

1.3. Descripción de las estructuras de datos desarrolladas.

Matriz: la matriz del juego que contiene las posiciones en donde cada jugador coloca su símbolo correspondiente. Dependiendo del valor en la casilla se determina de quién es dicha casilla.
Lista de mejores opciones: Esta lista contiene sublistas con 3 números que indican la probabilidad total, posición en i y posición en j en la matriz.

1.4. Descripción detallada de los algoritmos desarrollados.

A. Greedy Algorithm

En resumen, el algoritmo opera utilizando una definición de probabilidad dada por el número de filas y columnas disponibles que pueden ser rellenadas a futuro para generar un gane; de forma detallada, el algoritmo ocupa temporalmente cada posición vacía de la matriz para contar el número de posibilidades de juego que posee él y el usuario. Para cada posición, se genera una resta de la posibilidad del usuario con la posibilidad del algoritmo y este número es guardado en una lista con su respectiva posición en i y j de la matriz y esta sublista es guardada en una lista .
El número resultante de la resta de estas probabilidades provee la información necesaria para que el algoritmo realice la evaluación de cuál casilla vacía es más conveniente ocupar en el momento, dado que se busca que este número sea lo menor posible para que de esta forma, tenga mayores posibilidades de ganar. Seguidamente, la lista de tripletas es parseada en busca de la tripleta que contenga el número de probabilidad total menor y de esta forma, extraer la posición en i y j conveniente para el algoritmo para que pueda colocarse la ficha en esa posición y acercarse cada vez más al gane. También se implementó dos funciones extra para que el algoritmo sea capaz de tener en cuenta cuando el usuario está a una ficha de ganar y así poder repeler este movimiento ocupando esa casilla y de forma casi análoga, es capaz de ver cuando está él a una ficha de ganar y colocar en esa casilla para obtener el gane.

De manera resumida, el algoritmo funciona bajo los siguientes pasos:

1. Tomar la matriz entrante.
2. Verificar si existe un posible gane para colocar una ficha en esa posición.
3. Verificar si existe un posible gane del usuario para cerrar esa jugada.
4. Colocarse temporalmente en cada posición vacía de la matriz para contar el número de posibilidades del usuario y propias asociadas a esa posición y así crear una lista de tripletas.
5. Parsear la lista de tripletas para buscar el número menor de posibilidad total (el resultado más bajo en la resta de la posibilidad del usuario con la del algoritmo) y tomar los índices de posición asociados a ese número.
6. Sustituir por un 2 en la posición encontrada.
7. Devolver la matriz con esa posición sustituida.

1.5. Problemas conocidos.

Existen casos de juego en donde el algoritmo no es capaz de evaluar y es vencido fácilmente.

1.6. Actividades realizadas por estudiante.

Alexis Gavriel:

- Funciones para la indexación, edición y manejo en general de matrices. 29 de abril. 4 horas

Crisptofer Fernandez:

- Investigación, diseño e implementación del algoritmo voraz. 5 de mayo, 6 de mayo.

Emmanuelle Aguilar:

- Investigación de las herramientas necesarias para la elaboración de la interfaz gráfica en Racket, así como su posterior implementación y adaptación con el resto de módulos del programa.(debería ir la fecha pero me la volé sin querer)

1.7. Problemas encontrados.

Debido a que es un algoritmo voraz, la decisión es tomada al momento y no al final de todo. El algoritmo no necesariamente tomará posesión de una posición con la que puede ganar si encuentra una posición en donde puede evitar que el jugador gane antes.

1.8. Conclusiones y Recomendaciones del proyecto.

Conclusiones

- Los algoritmos voraces consumen muchos recursos debido a que toman muchos datos y los intentan analizar para tomar la mejor decisión en un momento dado.
- La decisión de un algoritmo voraz no necesariamente es la mejor decisión que puede ser tomada.
- Utilizar el lenguaje de programación Racket permite que las funciones matemáticas, en este caso las funciones que analizan la matriz y las funciones del algoritmo voraz, sean realizadas de una forma eficiente y eficaz utilizando recursividad.
- Se debe utilizar listas como la estructura de datos básica para almacenar los datos de la matriz del juego.

Recomendaciones

- Se recomienda buscar acerca de distintos algoritmos utilizados en el juego de tres en línea. Esto ayudará en la selección de un mejor algoritmo para que la computadora pueda tomar mejores decisiones a la hora de jugar.

1.9. Bibliografía consultada en todo el proyecto

Helo-Guzman, José Elias : Introducción a la programación con SCHEME /. Cartago, Costa Rica : Editorial Tecnológica de Costa Rica, 2005.[ISBN 9977 66 176 6] (#000038574).

https://www.quora.com/Is-there-any-algorithm-for-tic-tac-toe-that-does-not-rely-on-look-ahead-search-that-is-perfect-for-any-sized-boards.

2.1. Bitácora de trabajo

Alexis Gavriel		
Fecha	**Horas**	**Actividad realizada**
26 Abril	0.5	Repartición de tareas entre los tres miembros de trabajo.
28 Abril	0.5	Investigación sobre algoritmos que juegan el juego entre los tres miembros de trabajo.
29 Abril	3	Implementación de las funciones básicas para la verificación de los datos en la matriz. Función para determinar la victoria independientemente de si es una línea horizontal, vertical, o diagonal.
5 Mayo	1	Reunión Alexis y Emmanuelle: Se unió la lógica del juego y la interfaz gráfica. Se corrigieron unos errores en la verificación de las diagonales.
9 Mayo	2	Finalización de la documentación.

Crisptofer Fernández		
Fecha	**Horas**	**Actividad realizada**
26 Abril	0.5	Repartición de tareas entre los tres miembros
28 Abril	0.5	Investigación sobre algoritmos que juegan el juego entre los tres miembros.
5 Mayo	3	Investigación, análisis y diseño del algoritmo voraz a implementar.
6 Mayo	6	Implementación y pruebas en varios casos del algoritmo voraz implementado.
9 Mayo	2	Finalización de la documentación

Emmanuelle Aguilar		
Fecha	**Horas**	**Actividad realizada**
26 Abril	0.5	Repartición de tareas entre los tres miembros.
28 Abril	0.5	Investigación sobre algoritmos que juegan el juego entre los tres miembros.
1 Mayo	3	Investigación sobre interfaz en DrRacket, su función big-bang y uso de los clicks del mouse.
2 Mayo	1.5	Realización de la función que acomoda los cuadros del tablero en pantalla, la función que inserta un número en una posición de la matriz y modificación de la matriz según los clicks del usuario.
4 Mayo	2	Dibujado final del tablero en pantalla y mostrar los cambios hechos por el jugador.
5 Mayo	1	Reunión Alexis y Emmanuelle: Se unió la lógica del juego y la interfaz gráfica.
7 Mayo	0.5	Mostrar el mensaje de finalización de la partida.
9 Mayo	2	Finalización de la documentación.