Instituto Tecnológico de Costa Rica

Escuela de Computación

II Proyecto Programado

Lenguajes de Programación

Eliomar Rodríguez Arguedas

Sede San Carlos

Setiembre 2017

Tabla de contenidos

1. Introducción

La búsqueda en profundidad es un algoritmo que permite recorrer un grafo de manera ordenada pero no uniforme. Dicho algoritmo consiste en ir expandiendo los caminos posibles a partir del nodo actual y eligiendo uno de los nodos hasta que no exista más camino, sino encuentra lo buscado debe regresar a elegir un nodo hermano del nodo procesado en la llamada recursiva anterior y así sucesivamente hasta recorrer todos los posibles caminos para encontrar el objetivo deseado. (Wikipedia, 2017)

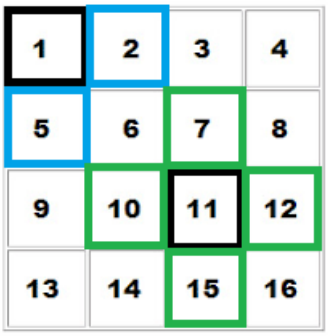
Para el siguiente proyecto se propone la creación de un programa que simule el juego Snake. Dicho juego consiste en que una serpiente debe desplazarse hasta las frutas las cuales son su comida. Para dar solución a dicha propuesta, se hace uso del algoritmo de búsqueda en profundidad el cual se explica su uso en este proyecto más adelante para poder detectar la ruta más cercana a las frutas y cuales movimientos hacer (más cercanos a la fruta más cercana).

2. Análisis de problema

Como se mencionó anteriormente, la propuesta de proyecto planteada es la realización de un programa que simule el juego Snake. Este juego consiste en una serpiente que debe moverse a ciertas posiciones en las cuales se encuentran frutas, la culebra debe poder llegar a comer esas frutas sin la ayuda de un usuario, es decir; debe ser inteligente y detectar en donde hay frutas para desplazarse hasta esa posición y así "comerse" la fruta. Además, se debe tomar el tiempo y la cantidad de movimientos realizados mostrando todo en una ventana gráfica.

Para poder avanzar se debe ir obteniendo los vecinos de una posición x por ejemplo y eligiendo el vecino que de acuerdo a un criterio de orden el cual es que el vecino que se va a utilizar para llegar al objetivo debe ser el mejor y que mediante él se llegue más rápido a la meta. A continuación, se tiene el siguiente ejemplo:

Lo encerrado en color negro es la posición x mencionada anteriormente, y lo encerrado en color azul son los vecinos de esa posición.



El mejor vecino para ir desde la posición 1 hasta la 9 es mediante el vecino 5 porque está a 2 movimiento del objetivo, en cuando el otro vecino (2) requiere de al menos 3 movimientos (2,6,10,9) para poder llegar al objetivo dado.

Los vecinos del número 11 encerrado en negro, sin 7, 10, 12, 15.

Luego de llegar al primer objetivo se debe repetir el mismo procedimiento hasta llegar al próximo objetivo y así sucesivamente hasta no tener más objetivos. Como se debe ir buscando la mejor ruta se requiere implementar un método de búsqueda en profundidad para así poder tomar la mejor decisión y llegar rápido al objetivo final (comer todas las frutas), dicho método debe ser recursivo para que realice el mismo procedimiento todas las llamadas, pero con los nuevos datos (nueva ubicación, nuevos vecinos, etc), además deben ordenar los vecinos y las frutas para saber cuál es la opción más viable y así llegar rápido al objetivo.

Por último, es necesario utilizar una librería (pygame en este caso) que permita mostrar en pantalla el proceso llevado a cabo por la serpiente para poder comer todas las frutas, así como la cantidad de tiempo que tardó en hacerlo y la cantidad de movimientos que le tomó llegar hasta la meta.

3. Solución del problema

Para poder dar solución al problema planteado, se utilizaron distintos métodos los cuales se hablarán a continuación.

Primeramente, se parte ubicando a la serpiente en la posición (0,0) el cual será el punto de partida cuando inicie el juego. La serpiente debe tener vecinos para poder elegir uno de ellos y así poder avanzar hasta la manzana más cercana, para poder obtener los vecinos de la serpiente, es necesario tener la posición de la fila y la columna (x,y) de la cabeza de la serpiente ya que dependiendo de la ubicación de la cabeza, va a tener ciertos vecinos, los cuales pueden ubicarse arriba, abajo, derecha e izquierda dependiendo de la posición en fila y columna de la serpiente.

Para poder elegir el mejor vecino, se debe restar a la x más grande la x más pequeña de manera que el número sea positivo (x quiere decir la ubicación en fila del vecino o de la fruta que se está evaluando), de igual manera con la y, a la y más grande se le resta la y más pequeña (misma lógica, y puede ser o la ubicación en columna de la fruta o del vecino que se está evaluando) y luego se suman ambos resultados obteniendo la cantidad de movimientos que se deben hacer desde la ubicación actual hasta la ubicación de una fruta, se ordenan las frutas de acuerdo a la cantidad de movimientos que se deben hacer para poder llegar a la fruta partiendo desde la cantidad más baja a la más alta de movimientos de modo que en la lista de vecinos va a quedar en la posición 0 el vecino más eficiente, de igual manera con la lista de frutas, en la posición 0 va a quedar la fruta que esté más cerca de la ubicación actual de la cabeza de la serpiente.

De esta manera es posible saber que vecino utilizar para poder avanzar hasta llegar a las frutas deseadas. Para la parte grafica simplemente se reutilizó código de un proyecto que se tuvo que realizar para el curso Taller de Programación el cual se imparte en el primer semestre de la carrera, solamente se tuvieron que cambiar algunas cosas como recibir una lista de movimientos e ir recorriéndola mostrando a su vez la ubicación actual de la serpiente, para ubicar en pantalla se multiplica la posición por una cantidad especifica de pixeles de manera que en la parte visual quede bien ubicada la serpiente la cual será una imagen que se carga mediante comandos propios de la librería pygame.

4. Análisis de resultados

A continuación se muestra una tabla con las actividades a realizar para la correcta elaboración del proyecto, en algunas actividades se tardó más tiempo que en otras debido a la diferencia de complejidad como por ejemplo la parte visual, como se tiene tiempo de no utilizar pygame, los conocimientos en dicha librería estaban un poco deteriorados pero gracias a que se había creado un proyecto para Taller de programación y además se había documentado para la entrega de la misma, se logró entender nuevamente cómo funcionaba la misma. A continuación, la tabla de actividades.

Actividad	Descripción	% de avance
Ubicación de la cabeza en la posición (0,0)	Desde ahí inicia el juego	100
Muestra tiempo y cantidad de movimientos realizados		100
Cantidad de movimientos mínima es 50		100
Ventana donde se logre apreciar la serpiente avanzando hasta los objetivos		100
La velocidad ingresada de actualización, permite apreciar el movimiento de la serpiente desplazándose hacia los objetivos		100
La serpiente será descrita por un solo elemento particular que pueda ser identificado claramente.	En este caso el dibujo es vegeta	100
Los objetivos desaparecen cuando la serpiente se los come		100
El juego termina cuando la serpiente se come todos los objetivos		100
El tamaño de la matriz será de 18 x 20		100

Las frutas se ubican en las posiciones indicadas	(9,5) - (2,11) – (8,18) – (17,4) – (13,19)	100
La serpiente se puede mover solamente en 4 direcciones	Arriba, abajo, izquierda, derecha	100
La serpiente no puede acceder los márgenes del campo de acción		100
Implementación del juego en modo automático.	Solo se ejecuta y el juego se resuelve solo	100
Manejo correcto de memoria correcto	No se hace uso de estructuras ya que eso implicaría un gasto extra de memoria.	100
Funciona bien el proyecto	https://drive.google.com/drive/folders/0Bxde4gIUnnxlOEZpa05ZTDZ6cEk?usp=sharing Carpeta donde se encuentra video que prueba la correcta ejecución del proyecto	100
Incluye todo lo necesario para que el proyecto se ejecute sin ningún problema	Para la ejecución es necesario tener instalada la librería pygame la cual se puede instalar con el tutorial mencionado en la parte de recomendaciones, no es nada complicado	20

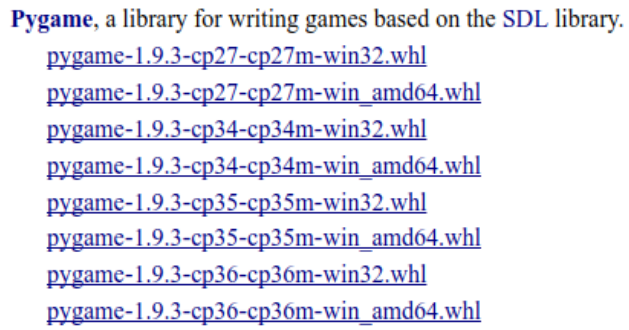
5. Recomendaciones

Para la correcta ejecución del programa, es necesario tener instalada la librería Pygame para Python ya que en caso de no contar con dicha librería el programa no se ejecutará de manera correcta y no es posible incluir dicha librería dentro de la carpeta del proyecto, en caso de no contar con dicha librería (pygame), puede instalarla siguiendo los pasos del siguiente video: https://www.youtube.com/watch?v=ki_5uS4bOgQ.

Para descargar la librería debe acceder al siguiente link: http://www.lfd.uci.edu/~gohlke/pythonlibs/#pygame y descargar la versión que necesita, ya sea

1) **pygame-1.9.3-cp36-cp36m-win32.whl** para **32 bits**

2) **pygame-1.9.3-cp36-cp36m-win_amd64.whl** para **64 bits**


Pygame, a library for writing games based on the SDL library.
pygame-1.9.3-cp27-cp27m-win32.whl
pygame-1.9.3-cp27-cp27m-win_amd64.whl
pygame-1.9.3-cp34-cp34m-win32.whl
pygame-1.9.3-cp34-cp34m-win_amd64.whl
pygame-1.9.3-cp35-cp35m-win32.whl
pygame-1.9.3-cp35-cp35m-win_amd64.whl
pygame-1.9.3-cp36-cp36m-win32.whl
pygame-1.9.3-cp36-cp36m-win_amd64.whl

6. Conclusión

Al concluir este proyecto se llegó a determinar que no era necesario el uso de estructuras. Como por ejemplo para almacenar las posiciones de los elementos (vecinos o frutas) ya que mediante la recursión propia del programa se puede obtener en cada llamada recursiva todos los vecinos y ordenarlos dependiendo de la cantidad de movimientos que se lleve hasta el objetivo más cercano desde un vecino determinado, así como las frutas ordenadas de manera que siempre se eligiese el mejor objetivo.

También gracias al desarrollo de este proyecto, se obtuvo gran conocimiento en el paradigma funcional, además de poder implementar a una escala menor un juego inteligente (IA) el cual se llega a resolver eligiendo el mejor camino y decidiendo que opción es mejor para cumplir un objetivo x.

Además, se llegó a determinar muchas ventajas del paradigma funcional como lo es el paso de parámetros, modularidad en el código lo cual ayuda a tener un código ordenado y a la hora de buscar posibles errores simplifica mucho el proceso, recursividad como en el paradigma imperativo no se puede hacer esto.

También mejoré mis conocimientos en el uso de librerías gráficas para el manejo de ventanas en Python como lo es en este caso pygame.

7. Bibliografía

Wikipedia. (11 de Mayo de 2017). Obtenido de

https://es.wikipedia.org/wiki/B%C3%BAsqueda_en_profundidad