Universidad del Valle de Guatemala

Curso: Estructura de datos

Catedrática: Lynette García

Integrantes: Robin Woods,Dieter de Wit y Gerardo Cardoza

Fecha de entrega: 08/08/16

Algoritmos para la resolución de laberintos

A continuación se encontrarán algunos algoritmos investigados:

- **Wall follower:** Este es un simple algoritmo de resolución de Maze. Se centra en ti, siempre es muy rápido, y no utiliza memoria adicional. Comience a seguir pasajes, y cada vez que se llega a un cruce siempre gire a la derecha (o izquierda). Equivalente a un ser humano para resolver un laberinto, poniendo su mano sobre la derecha (o izquierda) de la pared y dejándolo allí, ya que caminar a través. Si lo desea, puede marcar lo que las células que ha visitado, y lo que las células que ha visitado dos veces, donde al final se puede volver sobre la solución siguiendo esas células visitadas una vez. Este método no necesariamente va a encontrar la solución más corta, y no funcionar en absoluto cuando el objetivo está en el centro del laberinto y hay un circuito cerrado que lo rodea, a medida que va a ir por el centro y, finalmente, se encuentran de vuelta al principio.
- **Pledge algorithm:** Esta es una versión modificada de la pared después de eso es capaz de saltar entre islas, para resolver pared laberintos siguiente no se puede. Es una forma garantizada para llegar a una salida en el borde exterior de cualquier laberinto 2D desde cualquier punto en el medio, sin embargo no es capaz de hacer lo contrario, es decir, encontrar una solución dentro del laberinto. Es muy bueno para su ejecución por un robot de escapar del laberinto, ya que puede salir de cualquier laberinto sin tener que marcar o recordar el camino de ninguna manera.Comience eligiendo una dirección, y siempre se mueven en esa dirección cuando sea posible. Cuando se golpea una pared, pared de comenzar siguiente hasta que su dirección elegida está disponible de nuevo. Nota usted debe comenzar la pared siguiente a la pared del fondo que ha golpeado, en la que si el paso se convierte en una esquina allí, puede hacer que la vuelta en el medio de un pasaje y volver por el mismo camino. Cuando la pared siguiente, cuente el número de vueltas que usted hace, por ejemplo, un giro a la izquierda es -1 y un giro a la derecha es 1. Sólo detener la pared siguiente y tome su dirección elegida cuando el número total de vueltas que usted ha hecho es 0, es decir, si te has convertido en torno a 360 grados o más, mantener la pared siguiendo hasta que distorsiona mismo. El conteo asegura que está finalmente poder llegar al otro lado de la isla que está actualmente en, y salta a la siguiente isla en su dirección elegida, donde se va a mantener en la isla a isla en esa dirección hasta llegar a la pared limítrofe , en la que la pared punto siguiente le lleva a la salida. Nota algoritmo Prenda puede hacer que usted visita un pasaje o el inicio más de una vez, aunque los tiempos posteriores siempre estarán con diferentes totales de giro. Sin marcar su camino, la única manera de saber si el laberinto es imposible de resolver es si su giro total sigue aumentando, aunque el total a su vez puede llegar a un gran número de laberintos que tienen solución en un pasaje en espiral.
- **Chain algorithm:** El algoritmo de la cadena resuelve el laberinto tratando de manera efectiva como una serie de laberintos más pequeños, como los eslabones de una cadena, y la solución de ellos en secuencia. Se tiene que especificar el inicio y

ubicaciones finales deseados, y el algoritmo siempre encontrará un camino de principio a fin si es que existe, donde la solución tiende a ser un tiempo razonablemente corto, si no la solución más corta. Entonces sólo tiene que seguir la línea de principio a fin. Si te encuentras con una pared, no se puede pasar por ella, así que hay que dar la vuelta. Enviar dos paredes siguiente "robots" en ambas direcciones a lo largo de la pared se golpea. Si un robot se encuentra con la línea de guía de nuevo, y en un punto que está más cerca de la salida, y luego se detiene, y siga esa pared a sí mismo hasta que llegue allí también. Siga la línea y repitiendo el proceso hasta que se alcanza el final. Si ambos robots vuelven a sus lugares y direcciones originales, los puntos más lejanos a continuación, a lo largo de la línea son inaccesibles, y el laberinto es irresoluble.

- **Recursive backtracker:** Este será encontrar una solución, pero no necesariamente va a encontrar la solución más corta. Se centra en usted, es rápido para todo tipo de laberintos, y utiliza el espacio de pila hasta el tamaño del laberinto. Muy simple: Si usted está en una pared (o en un área que ya ha dibujaste), insuficiencia de regreso, de lo contrario si estás en la meta, el éxito de regreso, otra forma recursiva trate de mover en las cuatro direcciones. Trazar una línea cuando se intenta una nueva dirección, y borrar una línea cuando regrese fracaso, y una única solución se puntuará cuando se pulse el éxito. Cuando vuelta hacia atrás, lo mejor es marcar el espacio con un valor visitado especial, por lo que no visitan de nuevo desde una dirección diferente. En términos de Informática esto es básicamente una primera búsqueda en profundidad. Este método siempre encontrar una solución si es que existe, pero no necesariamente va a ser la solución más corta.
- **Trémaux's algorithm:** Este método de solución Maze está diseñado para ser capaz de ser utilizado por un ser humano dentro del laberinto. Es similar a la Backtracker recursiva y encontrará una solución para todos los laberintos: Al caminar por un pasillo, dibujar una línea detrás de usted para marcar su camino. Al llegar a un callejón sin salida, dar la vuelta y volver por el mismo camino. Cuando se encuentre con una unión que no ha visitado antes, elegir un nuevo pasaje al azar. Si estás caminando por un nuevo pasaje y encontrar una salida que ha visitado antes, tratarlo como un callejón sin salida y volver por el mismo camino. Si caminando por un pasaje que ha visitado antes (es decir, marcada una vez) y se encuentra con un cruce, tomar cualquier nuevo pasaje si uno está disponible, de lo contrario tomar un antiguo pasaje (es decir, uno que ha marcado una vez). Todos los pasajes serán o vacío, lo que significa que no ha visitado todavía, una vez marcada, significa que ha bajado es exactamente una vez, dos veces o marcado, lo que significa que haya pasado por ella y se vieron obligados a dar marcha atrás en la dirección opuesta. Cuando finalmente llegar a la solución, caminos marcados exactamente una vez indicarán de manera directa de nuevo al comienzo.

- **Dead end filler:** Este es un simple algoritmo de resolución del laberinto. Se centra en el laberinto, es siempre muy rápido, y no utiliza memoria adicional. Sólo búsqueda del laberinto, y rellenar en cada callejón sin salida, rellenando el paso hacia atrás desde el bloque hasta llegar a un cruce. Eso incluye el llenado en los pasajes que se convierten en partes de callejones sin salida, una vez otros callejones sin salida se eliminan. Al final solamente la solución permanecerá, o soluciones si hay más de uno. Esto siempre encontrará la solución única para una laberintos perfectos, pero no servirá de mucho en laberintos en gran medida de la trenza, y de hecho no va a hacer nada útil en absoluto para esos laberintos sin callejones sin salida.

Sacado de :http://www.astrolog.org/labyrnth/algrithm.htm

PSEUDOCÓDIGO

```
if (noParedIzqierda)
        cruce 90 grados izquierda
elseif (noParedEnfrente)
        avanzar un espacio
else
        cruce 90 grados derecha
```

¿El porqué del algoritmo?

Elegimos el algoritmo de la mano derecha porque es uno de los algoritmos que salen del laberinto sin importar cuan difícil sea. El algoritmo es fácil de implementar, ya que no tiene tanta complicación a la hora de la programación. La desventaja de este algoritmo es de que tarda mucho tiempo en salir en algunas ocasiones.