

Facultad de Ciencias Empresariales

Proyecto de Inteligencia Artificial

(Laberinto 2014 IECI)

**INTEGRANTES:** - Francisco Barra

* Andrés Garcés
* Héctor Poblete

**ASIGNATURA:** Inteligencia Artificial

**PROFESORA:** Clemente Rubio Manzano

**CARRERA:** IECI

**FECHA:** 27 de Junio del 2014

**Introducción**

En este proyecto de inteligencia artificial demostraremos como implementar algoritmos de búsquedas, vistos en clases. Se desarrolló un programa que consiste en generar un laberinto en donde participan dos jugadores: Jugador y Enemigo.

Jugador tiene la particularidad de buscar monedas en el laberinto para llegar a la meta y completar el nivel, si este logra completar los 10 mapas posteriormente ganara el juego.

Enemigo se encargara de intentar atrapar al jugador para que no logre ganar

En este proyecto presentaremos un programa en JAVA utilizando algoritmos de búsqueda como A\* y Coste uniforme.

**Análisis**

En este proyecto el problema consiste en buscar objetivos en un mapa aplicando técnicas de búsquedas de IA. El mapa donde se implementan los algoritmos tiene un tamaño de 480x480.

El algoritmo implementado es Búsqueda A\* donde la IA del Personaje principal y el enemigo ocupan este algoritmo.

El Personaje principal tiene que buscar la ruta más corta para encontrar todas las monedas y luego poder llegar a la meta sin que fuese tocado por el enemigo, para lo cual calcula la poción de la moneda más cercana al punto actual del personaje y calcula la ruta en la no choque con el enemigo, cada paso que da calcula la nueva ruta verificando si las posiciones siguientes existe el enemigo cambia la ruta para evitarlo.



**Diseño**

**Diseño gráfico.**





**Estados.**

Los Estado está definido por la siguiente estructura.

Estado (Punto(x,y),distancia, movimientos,antecesor).

* Donde Punto(x,y) la poción del objeto.
* Distancia del objeto hacia la meta ósea F.
* Movimientos son String de los movimientos originarios del estado.
* Antecesor es el estado original del nuevo estado.

Estado inicial Jugador ((1,1),calcularF(Moneda),null,null).

Estado final Jugador ((14,14),0,null,null).

Estado inicial Enemigo ((1,14),calcularF(Jugador),null,null).

Estado final Enemigo ((1,1),0,null,null).

**Movimientos;**

Jugador.

MoverAriba el objeto de tipo jugador se mueve hacia arriba si la posición “Y” nueva es mayor que cero, distinta de P (Pared) y distinta de E (enemigo),

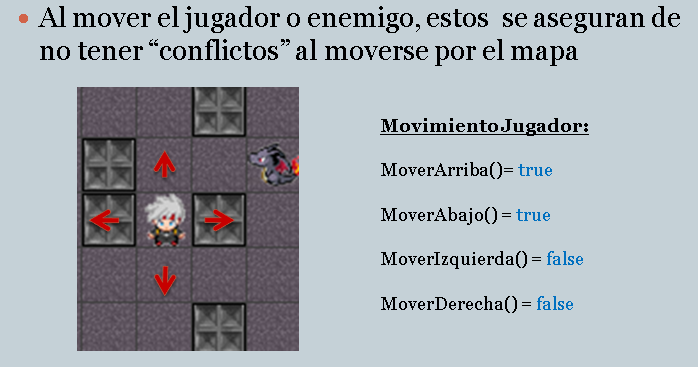
MoverAbajo el objeto de tipo Jugador se mueve hacia abajo si la posición “Y” nueva es menor que 16, distinta de P (Pared) y distinta de E (enemigo),

MoverIzquierda el objeto de tipo Jugador se mueve hacia la izquierda si la posición “x” nueva es mayor que cero, distinta de P (Pared) y distinta de E (enemigo),

MoverDerecha el objeto de tipo Jugador se mueve hacia la derecha si la posición “X” nueva es menor que 16, distinta de P (Pared) y distinta de E (enemigo),

Enemigo.

Son los mismo movimientos que el jugador exceptuando la verificación de la nueva posición sea distinta de E (Enemigo).



**Implementación**

Primero, estructuras.

Tipo ESTADO = {POSICION\_X , POSICION\_Y, COSTO\_F, “MOVIMIENTO”, ANTECESOR}

Tipo LISTAPRIORIDAD = [CONJUNTO DE ESTADOS]

Tipo LISTA = {CONJUNTO DE ESTADOS}

Segundo, debemos declarar las variables que voy a utilizar.

ABIERTOS: tipo LISTAPRIORIDAD

CERRADOS: tipo LISTA

ACTUAL: tipo ESTADO

INICIAL: tipo ESTADO

FINAL: tipo ESTADO

Tercero. Inicialización de variables.

ABIERTOS = [ ]

ACTUAL = NULL

INICIAL = { x, y, f,” ”,null }

FINAL = {x, y, f,” ”,null }

PROCESO DE BÙSQUEDA: BÙSQUEDA EN A\*

ABIERTOS.INSERTAR (INICIAL = { x, y, f,” ”,null } )

ABIERTOS = [ { x, y, f,” ”,null } ]

PRIMERA ITERACIÒN

ACTUAL = ABIERTOS.PRIMERO()={ { x, y, f,” ”,null } } //EL MENOR

ACTUAL NO ES FINAL

EXPANDIR(ACTUAL):

1.- MOVER\_ARRIBA -> (x , y-1, calcular(f) ,”U”, ACTUAL);

2.- MOVER\_ABAJO -> (x , y+1, calcular(f) ,”D”, ACTUAL);

3.- MOVER\_DERECHA -> (x +1, y, calcular(f) ,”R”, ACTUAL);

4.- MOVER\_IZQUIERDA -> (x-1 , y, calcular(f) ,”L”, ACTUAL);

ABIERTOS.QUITAR\_PRIMERO();

ABIERTOS = [ { SUCESORES{1 , 2 , 3 , 4} } ]

Un ejemplo de la traza del jugador.

Inicial((1,1),calcularF(moneda),null,null).

Meta((3,5),0,null,null).

1.-Iteración.

Abiertos (1,1)

Actual (1,1) // menor F de los abiertos

Expandir (1,1)

Cerrados (1,1)

2.-Iteración.

Abiertos {(1,2),(2,1)}

Actual (1,2) // menor F de los abiertos

Expandir (1,2)

Cerrados {(1,1),(1,2)}

3.-Iteración.

Abierto {(1,3),(1.1),(2,2),(2,1)}

Actual (1,3) // menor F de los abiertos

Expandir (1,3)

Cerrados {(1,1),(1,2),(1,3)}

4.-Iteración.

Abierto {(1,4),(2,3),(1,2),(1.1),(2,2),(2,1)}

Actual (1,4) // menor F de los abiertos

Expandir (1,4)

Cerrados {(1,1),(1,2),(1,3),(1,4)}

5.-Iteración.

Abierto {(2,4),(1,5),(1,3),(2,3),(1,2),(1,2)(1.1),(2,2),(2,1)}

Actual (2,4) // menor F de los abiertos

Expandir (2,4)

Cerrados {(1,1),(1,2),(1,3),(1,4),(2,4)}

6.-Iteración.

Abierto {(2,5),(2,3),(3,4),(1,4),(1,5),(1,3),(2,3),(1,2),(1,2)(1.1),(2,2),(2,1)}

Actual (2,5) // menor F de los abiertos

Expandir (2,5)

Cerrados {(1,1),(1,2),(1,3),(1,4),(2,4),(2,5)}

7.-Iteración.

Abierto {(3,5),(2,4),(1,5),(2,3),(3,4),(1,4),(1,5),(1,3),(2,3),(1,2),(1,2)(1.1),(2,2),(2,1)}

Actual (3,5) // menor F de los abiertos

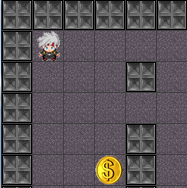
Meta (3.5) //La meta es la moneda

Cerrados {(1,1),(1,2),(1,3),(1,4),(2,4),(2,5),(3.5)}

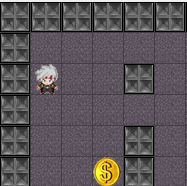
FIN

Se puede ver la secuencia en las siguientes imágenes.

(1.-Iteración.)



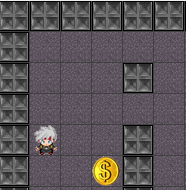
(2.-Iteración.)



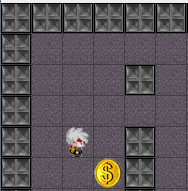
(3.-Iteración.)



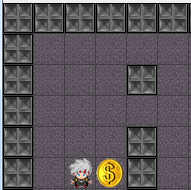
(4.-Iteración.)



(5.-Iteración.)



(6.-Iteración.)



**Conclusión**

* En este proyecto aprendimos a implementar los algoritmos vistos en clases y deducir que algoritmos usar en ciertas ocasiones para escoger el óptimo y aplicarlo a un problema.
* Una buena aplicación que puede ocupar A\*, es la ruta más óptima que calcula los mapas en base a la ubicación por GPS.

**Bibliografías**

[**http://www.face.ubiobio.cl/~clrubio/ia2014/**](http://www.face.ubiobio.cl/~clrubio/ia2014/)

[**https://github.com/laberintoIA2014/IA\_2**](https://github.com/laberintoIA2014/IA_2)