CONCEPTOS DE INTELIGENCIA ARTIFICIAL – PROYECTO 2

Búsqueda Informada – A*

Búsqueda de Planes en una Isla

Francisco Brunner
brunner.francisco874@gmail.com – LU: 102242

Introducción

El objetivo de este proyecto es la implementación del método de búsqueda informada A*, en PROLOG para la obtención de planes de desplazamiento de un agente aventurero a través de una isla. Esta isla está representada como una grilla de celdas y el aventurero debe desplazarse por las celdas transitables en búsqueda de alguna posición determinada como meta que será la posición donde, según un rumor, se encuentra un tesoro. Para esto, debe escoger inteligentemente el camino a transitar, ya que debe ser el de menor costo posible desde su posición inicial hasta alguna meta.

La isla cuenta con distintos tipos de suelo (firme, resbaladizo o lava), existen elementos que están tirados en la isla (palas o llaves), que el aventurero puede levantar y en las celdas puede haber refugios que pueden requerir una llave de acceso con accesos habilitados, o puede haber obstáculos el aventurero puede saltar siempre que la altura lo permita.

El aventurero puede realizar seis acciones sobre una celda:

1. **Avanzar**: esta acción le permite al aventurero desplazarse a una celda contigua. El costo asociado a esta acción es 1 si la celda hacia donde se desplaza es firme y 2 en caso que sea resbaladizo.
2. **Girar**: esta acción permite al aventurero cambiar la dirección en la que esta mirando. El costo asociado a esta acción es 1 si gira 90º y 2 si gira 180º.
3. **Saltar Lava**: esta acción le permite al aventurero desplazarse por encima de las celdas que tengan lava. El costo asociado a esta acción es 2 como costo base mas el costo asociado a avanzar sobre suelo firme.
4. **Saltar Obstáculo**: esta acción le permite al aventurero desplazarse por encima de las celdas que contengan un obstáculo con una altura menor a 5.
5. **Levantar Llave**: esta acción le permite al aventurero obtener una llave que se encuentre en una celda y guardarla junto con sus posesiones. Esta acción no tiene costo asociado.
6. **Levantar Pala**: esta acción le permite al aventurero obtener una pala que se encuentre en una celda y guardarla junto con sus posesiones. El costo asociado a esta acción es de 1.

Todas las acciones poseen restricciones detalladas en el enunciado.

Estrategia de método de búsqueda A*

Como el objetivo del proyecto es obtener planes para un agente que dado un estado inicial desea alcanzar una meta, el programa implementa el predicado buscar_plan/5:

buscar_plan (+EstadoInicial, +Metas, -Destino, -Plan, -Costo).

que dado un *estado inicial* y un conjunto de *posiciones meta*, devuelve la secuencia de acciones que lleven al agente desde la posición inicial a la posición meta que menos le cueste llegar, junto con la posición de la meta alcanzada y el costo asociado a ese plan.

En caso de que las posiciones meta no puedan ser alcanzadas el predicado falla indicando lo sucedido.

La búsqueda del plan se realiza siguiendo el método de búsqueda A* por lo tanto, se implementó el predicado buscarAE/3:

buscarAE (-Destino, -Solución, -Costo).

este predicado sigue la siguiente estrategia:

- **Caso Base**:
  - Se Selecciona un nodo de la frontera
  - El nodo seleccionado corresponde a un Estado Meta
  - Destino devuelve la posición del Estado Meta
  - Solución devuelve Lista de acciones asociado al camino de la búsqueda desde el Estado Inicial hasta el Estado Meta
  - Costo devuelve el costo asociado al camino luego de aplicar el camino de acciones
- **Caso Recursivo**:
  - Se selecciona un nodo de la frontera
  - El Nodo seleccionado NO corresponde a un Estado Meta
  - Se generan los Potenciales Vecinos del Estado asociado al Nodo
  - Se agregan los Potenciales Vecinos a la Frontera haciendo el control de visitados.

Representación de Estados

Los **estados** suponen una situación puntual del aventurero en un momento determinado.

Se representan mediante una lista de tres elementos, donde el primer elemento corresponde a la posición, el segundo elemento corresponde a la dirección en la cual se encuentra mirando y el tercer elemento corresponde a la lista de posesiones.

A su vez, la posición se representa como una lista de dos elementos donde el primer elemento corresponde a la fila y el segundo elemento corresponde a la columna de la grilla. La lista de posesiones se representa como una lista de elementos donde cada elemento puede ser una Llave = [l, NombreL, Accesos], o una Pala = [p, NombreP].

Por lo tanto, un estado posee la siguiente estructura:

Estado = [Posición, Dirección, Posesiones].

Representación de Nodos

Los **nodos** suponen una estructura en la cual almacenar la información de los estados a medida que se realiza la búsqueda de la meta. Por ello se necesita almacenar ciertos datos adicionales para determinar el camino recorrido junto con el costo y la heurística asociada al estado.

Los nodos se representan mediante predicado cuyo functor es el que lo identifica, los parámetros que posee son:

1. El Estado Actual donde se encuentra la búsqueda,
2. El camino desde el Estado Inicial hasta el Estado Actual,
3. El costo asociado de realizar la búsqueda siguiendo el camino, es decir g(Nodo) y
4. La función de estimación del costo total del camino, es decir: f(Nodo) = g(Nodo) + h(Nodo), donde h(Nodo) es el valor de la heurística asociada al nodo que se detalla posteriormente.

Descripción de la Heurística

Los métodos de búsqueda informada utilizan cierta información para determinar qué nodo de la frontera elegir para encontrar rápidamente una meta. Esta información es llamada heurística.

De acuerdo al contexto, la heurística adoptada se diseño para seleccionar de la frontera el nodo cuya distancia Manhattan a la meta fuera la menor. Para lograr esto, se tiene en cuenta el caso en que el aventurero tiene la pala y el caso que todavía no la tiene en su lista de posesiones. Por lo tanto, definimos los siguientes casos:

1. El aventurero **posee** la Pala en su lista de posesiones:
    a. El valor de la heurística en caso es la distancia manhattan desde la posición donde se encuentra la búsqueda hasta la meta más cercana.
2. El aventurero **NO posee** la Pala en su lista de posesiones:
    a. El valor de la heurística en este caso es la menor distancia manhattan desde la posición donde se encuentra la búsqueda hasta la Pala cuya distancia manhattan a la meta sea la menor.

Calculo de la Heurística

Para realizar el cálculo del valor de la heurística en el segundo caso, antes de comenzar la búsqueda se realiza el producto cartesiano entre Palas y Metas, esta información se almacena en hechos dinámicos de PROLOG mediante el predicado tupla/3 cuyos parámetros son: la posición de la pala, la posición de la meta y la distancia manhattan desde la pala hasta la meta. A la hora de generar los vecinos, para hallar el valor de la heurística asociada a un estado, se calculan todas las distancias desde la posición del estado hasta todas las “tuplas” y de todas esas esas distancias se selecciona la menor.

Generación de Vecinos

El predicado *generarVecinos*/2:

generarVecinos (+Nodo, -Vecinos).

es el encargado de generar los potenciales vecinos del Nodo. El proceso de generación de nodos vecinos es el siguiente:

Dado un Nodo:

- Genera todos aquellos vecinos potenciales que sean sucesores del Estado
- Calcula el costo del nuevo nodo
- Calcula la heurística para el nuevo Estado
- Calcula f(Nodo) = g(Nodo) + h(Nodo) para el nuevo Estado

Para generar los sucesores de un estado se utiliza el predicado sucesor/4

sucesor (+EstadoActual, -EstadoSiguiente, -Acción, -Costo).

que dado un EstadoActual genera un posible EstadoSiguiente luego de haber aplicado la Acción con las restricciones adecuadas a cada caso y el Costo asociado.

Control de Visitados

Se utiliza un control de nodos visitados en la implementación para lograr mayor eficiencia. Siguiendo la estrategia del método de búsqueda A*, puede que se genere un nodo con un estado que ya había sido generado, pero por otro camino con distinto costo. Para tener esto en cuenta en el programa el control se hace de la siguiente manera:

Cuando un nuevo nodo A es generado con estado E y costo $g(A)$:

- Si existe en la frontera un nodo B, con el mismo estado E y costo $g(B)$, donde $g(A) < g(B)$, entonces se reemplaza el nodo B por el nodo A.
- Si existe en el conjunto de visitados un nodo C con el mismo estado E y costo $g(C)$, donde $g(A) < g(C)$, entonces se reemplaza el nodo C por el nodo A.
- Si existe en la frontera o en el conjunto de visitados un nodo D con el mismo estado E y costo $g(D)$, donde $\mathrm{g(A)} >= \mathrm{g(D)}$, el nodo A se descarta.
- En otro caso el nodo A se agrega a la frontera como un nuevo nodo.

Nota: Los nodos visitados se manipulan con hechos dinámicos, utilizando el predicado *visitado*/1 que recibe el nodo a agregar al conjunto de visitados.

Manipulación de la Frontera

Así como el conjunto de visitados la frontera se manipula con hechos dinámicos, utilizando el predicado *frontera*/1 que recibe el nodo a agregar a la frontera.

Al momento de seleccionar un nodo de la frontera en caso de que el Estado no sea una meta entonces se remueve ese Nodo de la frontera. Al momento de agregar nodos a la frontera luego de hacer el control de visitados sobre los nodos, se agrega un hecho con la estructura *frontera* (Nodo) al programa.