Práctica No. 3a
Mecanismos de búsqueda no informada en Common Lisp

Introducción

Uno de los primeros paradigmas de la Inteligencia Artificial fue la representación de problemas como un espacio de estados, lo que permitió definir la solución de los mismos como la búsqueda de un estado que represente una solución aceptable al problema partiendo de un estado que represente la situación inicial del problema. Los algoritmos de búsqueda se clasifican en: *No informados* e *Informados*. Dentro de los algoritmos no informados se encuentran: *i*) *Búsqueda Primero en Profundidad* y *ii*) *Búsqueda Primero en Amplitud*. El espacio de estados puede estar representado explícitamente o implícitamente. La representación explícita es aquella en la que se definen, previo al inicio de la búsqueda, todos los estados posibles y las conexiones entre ellos. En la representación implícita normalmente se utiliza un sistema de producción para generar sobre la marcha los posibles estados siguientes de un estado dado.

Búsqueda primero en profundidad

Esta búsqueda se realiza bajando dentro del árbol tan pronto como sea posible. Se lleva a cabo generando siempre un nodo hijo del nodo más recientemente expandido, después se genera el nodo hijo de este nodo y así sucesivamente hasta que el nodo meta es encontrado o hasta que se llega a una profundidad tope predefinida. Si el llegar a este punto todavía no se encuentra el nodo meta, el programa debe regresar al nodo anterior y generar otro de sus hijos. Este proceso continúa hasta que se encuentra el nodo meta o ocurre una falla.

La estrategia general consiste en manejar una lista de trayectorias parciales desde el nodo inicial a nodos intermedios, hasta que una de las trayectorias parciales se pueda extender lo suficientemente lejos para llegar a ser una trayectoria completa aceptable. Para implementar la estrategia de búsqueda primero en profundidad se utiliza una estructura de cola (una lista de listas en Lisp) para almacenar las trayectorias parciales no exploradas. El algoritmo recursivo es el siguiente:

1.- Si la función se llama con nodo inicio y nodo final solamente, coloca el nodo inicial *s* en la cola como una primera trayectoria.
2.- Si la cola está vacía, regresa `nil` y termina
3.- Si la primera trayectoria de la cola es una trayectoria completa, regresa esa primera trayectoria de la cola
Sino
4.- Extiende la primera trayectoria de la cola a todos los nodos vecinos del nodo terminal de esta trayectoria parcial, luego reemplazamos la primera trayectoria parcial con las trayectorias recién extendidas y **las colocamos al frente de la cola** (en cualquier orden).
5.- Continuamos la búsqueda hasta encontrar una trayectoria completa aceptable o fracasar.

Búsqueda primero en amplitud

Esta búsqueda se realiza explorando todos los nodos de un nivel de profundidad dado antes de pasar al siguiente nivel de profundidad. Esto significa que todos los hijos directos de un nodo se exploran antes de explorar los nodos hijos de estos hijos. El algoritmo para esta estrategia de búsqueda es el siguiente:

1.- Si la función se llama con nodo inicio y nodo final solamente, coloca el nodo inicial *s* en la cola como una primera trayectoria.
2.- Si la cola está vacía, regresa `nil` y termina
3.- Si la primera trayectoria de la cola es una trayectoria completa, regresa esa primera trayectoria de la cola
Sino
4.- Extiende la primera trayectoria de la cola a todos los nodos vecinos del nodo terminal de esta trayectoria parcial, luego reemplazamos la primera trayectoria parcial con las trayectorias recién

extendidas y **las colocamos al final de la cola** (en cualquier orden).

5.- Continuamos la búsqueda hasta encontrar una trayectoria completa aceptable o fracasar.

Actividades

1.- Captura el siguiente código de Common Lisp en un archivo `busquedas.lsp` que implementa las búsqedas primero en amplitud y primero en profundidad.

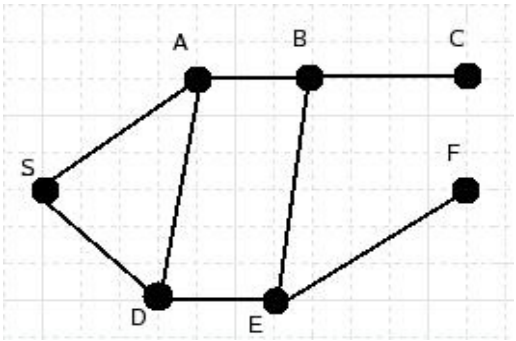
```
;; -----------------------------------------------------------------------
;; Archivo: busquedas.lsp
;; Implementa las busquedas no informadas primero en profundidad (bpp) y
;; primero en amplitud (bpa). Es una implementacion recursiva.
;; -----------------------------------------------------------------------
(defun espacio ()
  (setf (get 's 'vecinos) '(a d)
        (get 'a 'vecinos) '(s b d)
        (get 'b 'vecinos) '(a c e)
        (get 'c 'vecinos) '(b)
        (get 'd 'vecinos) '(s a e)
        (get 'e 'vecinos) '(b d f)
        (get 'f 'vecinos) '(e) )
)

(defun extiende (trayectoria)
      (mapcar #'(lambda (nuevo-nodo) (cons nuevo-nodo trayectoria))
                  (remove-if #'(lambda (vecino) (member vecino trayectoria))
                  (get (first trayectoria) 'vecinos))
      )
)

(defun bpp (inicial final &optional (cola (list (list inicial))))
      (cond       ((endp cola) nil) ; ¿esta vacia la cola?
            ((eq final (first (first cola))) ; trayectoria completa?
                        (reverse (first cola)) )       ; Devuelve trayectoria.
            (t (bpp inicial final (append (extiende (first cola)) (rest cola))))
      )
)

(defun bpa (inicial final &optional (cola (list (list inicial))))
      (cond ((endp cola) nil) ; ¿esta vacia la cola?
            ((eq final (first (first cola))) ; trayectoria completa?
                        (reverse (first cola)) )       ; Devuelve trayectoria.
            (t (bpa inicial final (append (rest cola) (extiende (first cola)))))
      )
)
```

La función `espacio()` define el siguiente espacio de estados:



2.- Carga el archivo en el intérprete de Common Lisp tecleando `(load "z:/busquedas.lsp")`, genera la representación explícita de estados ejecutando la función `espacio` tecleando `(espacio)`, ejecuta las siguientes búsquedas y reporta la ruta determinada por el algoritmo en cada caso:

Búsquedas primero en amplitud

`(bpa 'a 'f)` Ruta: _____
`(bpa 's 'f)` Ruta: _____
`(bpa 'f 's)` Ruta: _____
`(bpa 's 'e)` Ruta: _____

Búsquedas primero en profundidad

`(bpp 'a 'f)` Ruta: _____
`(bpp 's 'f)` Ruta: _____
`(bpp 'f 's)` Ruta: _____
`(bpp 's 'e)` Ruta: _____

3.- Define un nuevo espacio de estados que contenga por lo menos 10 nodos y especifícalo dentro de una nueva función `espacio2()`, que debes agregar al archivo `busquedas.lsp` para que genere el nuevo espacio de estados, dibuja el espacio que definiste, escribe el código de la función `espacio2()` que lo describe, recarga el archivo `busquedas.lsp`, ejecuta varias búsquedas tanto en amplitud como en profundidad y reporta los resultados.

3.1.- Dibujo del espacio de estados que definiste

3.2.- Código de la función `espacio2()`

3.3.- Búsquedas primero en amplitud

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

3.4.- Búsquedas primero en profundidad

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

Nodo Inicial: _____ Nodo Final: _____
Ruta: _____

4.- Comentarios y conclusiones

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____