Agentes y busqueda - Busqueda no informada

Sunday, August 27, 2023 6:15 PM

Agentes y busqueda

Algoritmos de Busqueda

Propiedades que deben cumplir los algoritmos de busqueda

Branching Factor: Cantidad maxima de hijos que puede tener cada nodo

Completos: Encuentran una solucion si esta existe

Optimal: Encuentran el camino de menos costo

Complejidad temporal: Cantidad de tiempo que requiere un algoritmo para resolver un problema en funcion de la entrada

Complejidad espacial: Cantidad de memoria que requiere un algoritmo para resolver un problema en funcion de la entrada.

Depth First Search

- Puede llegar a recorrer todo el arbol
- Siendo m la profundidad del arbol, la complejidad al recorrerlo es $O(b^m)$ → complejidad temporal
- El espacio en memoria maximo que puede tomar la frontera es $O(mb)$ → Frontera de busqueda: conjunto de nodos que se encuentran aun en la pila
- Es completo cuando el espacio de estados es finito. De lo contrario es incompleto
- No es optimo: No se garantiza una solucion optima

Breadth First Search (Busqueda en amplitud)

Es igual que la busqueda en profundidad, pero la frontera de busqueda ya no es una pila, sino una cola

- Realiza busqueda por niveles (Recorre el primer nivel, luego el segundo, etc)
- Su complejidad es $O(b^s)$ (con s el nivel donde se encuentra el estado objetivo)
- La complejidad espacial seria $O(b^s)$, ya que en la cola estan almacenados los nodos del nivel s
- Es completo: si la solucion existe, tarde o temprano la hallara
- Si es optimo si todas las acciones tienen costos iguales, de lo contrario no necesariamente

Summary of algorithms

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening
Complete?	Yes*	Yes*	No	Yes, if $l \geq d$	Yes
Time	b^{d+1}	$b^{\lceil C^*/\epsilon \rceil}$	b^m	b^l	b^d
Space	b^{d+1}	$b^{\lceil C^*/\epsilon \rceil}$	bm	bl	bd
Optimal?	Yes*	Yes	No	No	Yes*

Interactive Deepening

Es otro algoritmo que combina los mejores atributos del Depth First y el Breadth First

Es completo: Se acota la exploracion a una profundidad determinada. Ya que recorre toda la zona acotada, entonces es completo

Uniform cost search

Algoritmo bastante parecido al Breadth Search

- En vez de buscar por niveles, busca por zonas del mismo costo
- La pila (Depth First Search) o la cola (Breadth First Search) son reemplazadas por una cola de prioridad

- La clase hija debe sobreescribir los metodos de "acciones" y "resultados" y depende otros metodos
- Cuando se crea una instancia de una clase hija, el constructor recibira el estado inicial, y el objetivo (O una funcion goal test) y se crea, otros argumentos.

3,5

Estados: $\langle A, B \rangle$

Acciones: 6 estados

- Llenar completamente A,B => $3 - A > 0$ (1) | $5 - B > 0$ (2)
- Vaciar completamente A,B => $A > 0$ (3) | $B > 0$ (4)
- Depositar A en B hasta llenar B => $A > 0$ ∧ capacidad $B - B > 0$ 5
- Depositar B en A hasta llenar A => $B > 0$ ∧ capacidad $A - A > 0$ 6

Llenar B (0,5)
Depositar B en A (3,2)
Vaciar A (3,2)
Depositar B en B (

- Vaciar completamente A,B => A>0 ∧ B>0
- Depositar A en B hasta llenar B => A>0 ∧ capacidad B - B>0 5
- Depositar B en A hasta llenar A => B>0 ∧ capacidad A - A>0 6

Depositar A en B L

A - [sizeB - B]
1 5 1

Estado inicial y goal Test

- A y B estan vacios
- contenido A == 1 o contenido B == 1