Búsqueda no Informada

Algoritmos de búsqueda no informada:

Primero en anchura, primero en profundidad y descenso iterativo

Primero en anchura: bpa

1. **Abiertos** $\leftarrow$ (n_0); **Cerrados** $\leftarrow$ ( )
2. Si **Abiertos** = ( ), fin devolviendo fallo
3. **n** $\leftarrow$ primer elemento de **Abiertos**; eliminar **n** de **Abiertos** y llevarlo a **Cerrados**; **Suc** $\leftarrow$ ( )
4. Si **n** es meta, fin con éxito, devolviendo el camino
5. expandir **n**, colocando sus hijos en **Suc**, como hijos de **n**
6. eliminar de **Suc** cualquier nodo cuyo estado ya esté asociado a algún nodo de **Abiertos** o **Cerrados**
7. colocar los nodos de **Suc** al final de **Abiertos**
8. Ir a 2

Consumo de tiempo y memoria Búsqueda primero en anchura

Profundidad	Nodos	Tiempo	Memoria
0	1	1 milisegundo	100 bytes
2	111	.1 segundos	11 kbytes
4	11,111	11 segundos	1Mbyte
6	10^6	18 minutos	111 Mbytes
8	10^8	31 horas	11 Gbytes
10	10^{10}	128 días	1 Tbyte
12	10^{12}	35 años	111 Tbytes
14	10^{14}	3500 años	11,111 Tbytes

Complejidad espacial y temporal $O(b^d)$
Factor ramificación medio: 10; 1000 nodos/segundo; 100 bytes/nodo

Coste uniforme

- Variante BPA
- Criterio expansión de nodos:
  - expandir primero el nodo cuyo camino tenga el menor coste
- Equivalente a BPA si todos los costes son iguales
- Comportamiento:
  - similar a BPA
  - **admisible**

Primero en profundidad (básico) bpp

1. Si $\mathbf{n_0}$ es meta, fin con éxito, devolviendo $\mathbf{n_0}$
2. **Abiertos** $\leftarrow$ ($\mathbf{n_0}$); **Cerrados** $\leftarrow$ ( )
3. Si abiertos = ( ), fin devolviendo fallo
4. **n** $\leftarrow$ primer elemento de **Abiertos**; eliminar **n** de **Abiertos** y llevarlo a **Cerrados**; **Suc** $\leftarrow$ ( )
5. expandir **n**, colocando sus hijos en **Suc**, como hijos de **n**
6. Si alguno de los hijos de **n** es un nodo meta, fin con éxito, devolviendo el camino
7. eliminar de **Suc** cualquier nodo cuyo estado ya esté asociado a algún nodo de **Abiertos** o **Cerrados**
8. colocar los nodos de **Suc** al principio de **Abiertos**
9. Ir a 3

Primero en profundidad: bpp

1. Si $\mathbf{n_0}$ es meta, fin con éxito, devolviendo $\mathbf{n_0}$
2. **Abiertos** $\leftarrow$ ($\mathbf{n_0}$); **Cerrados** $\leftarrow$ ( )
3. Si **Abiertos** = ( ), fin devolviendo fallo
4. **n** $\leftarrow$ primer elemento de **Abiertos**; eliminar **n** de **Abiertos** y llevarlo al principio de **Cerrados**; **Suc** $\leftarrow$ ( )
5. expandir **n**, colocando sus hijos en **Suc**, como hijos de **n**
6. Si alguno de los hijos de **n** es un nodo meta, fin con éxito, devolviendo el camino
7. eliminar de **Suc** cualquier nodo cuyo estado ya esté asociado a algún nodo de **Abiertos** o **Cerrados**
8. Si **Suc** = ( ), limpiar_cerrados
9. colocar los nodos de **Suc** al principio de **Abiertos**
10. Ir a 3

Primero en profundidad limitado: bppl(pmax)

1. Si $\mathbf{n_0}$ es meta, fin con éxito, devolviendo $\mathbf{n_0}$
2. **Abiertos** $\leftarrow$ ($\mathbf{n_0}$); **Cerrados** $\leftarrow$ ( )
3. Si **Abiertos** = ( ), fin devolviendo fallo
4. **n** $\leftarrow$ primer elemento de **Abiertos**; eliminar **n** de **Abiertos** y llevarlo al principio de **Cerrados**; **Suc** = $\leftarrow$ ( )
5. Si profundidad(n) < **pmax**,
   - 5.1. expandir **n**, colocando sus hijos en **Suc**, como hijos de **n**
   - 5.2. Si alguno de los hijos de **n** es un nodo meta, fin con éxito, devolviendo el camino
   - 5.3. eliminar de **Suc** cualquier nodo cuyo estado ya esté asociado a algún nodo de **Abiertos** o **Cerrados**
6. Si **Suc** = ( ), limpiar_cerrados
7. colocar los nodos de **Suc** al principio de **Abiertos**
8. Ir a 3

Descenso Iterativo

1. **pmax** $\leftarrow$ 0
2. Si bppl(pmax) devuelve éxito, fin con éxito
3. **pmax** = **pmax** + 1
4. Ir a 2

Resumen comportamiento

	BPA	BPP	BPPL l<d	BPPL l>=d	DI
¿Completo?	SI	NO	NO	SI	SI
¿Admisible?	SI	NO	NO	NO	SI
Complejidad espacial	$O(b^d)$	bm	bl	bl	bd
Complejidad temporal	$O(b^d)$	$O(b^m)$	$O(b^l)$	$O(b^l)$	$O(b^d)$

peor caso, árbol de búsqueda, igual coste
b: factor de ramificación medio
d: profundidad solución
m: profundidad máxima
l: límite profundidad