

Ejercicios 3: Búsqueda no informada

Departamento de Informática / Department of Computer Science
Universidad Carlos III de Madrid

Inteligencia Artificial
Grado en Ingeniería Informática
2019/20

~ Problema que se puede
representar en un grafo.
Consiste en encontrar la manera de
llegar a la solución de la forma
más óptima, coste reducido.
No informada porque no tenemos
info de que supone una opción
u otra, arcos con coste.

Notas

Representación de un problema de búsqueda

Búsqueda en grafos

Representacion y búsqueda

Ejercicios adicionales

- ▶ Para definir un **estado** hay que escribir toda la información que lo caracteriza. Aunque en los grafos todo el estado se representa con una sola letra (A, B), en otros ejercicios tenemos que usar tuplas, listas, o objetos con TODA la información que los operadores pueden usar.
- ▶ Para **representar** un problema tenemos que codificar **estados** y **operadores**, además del estado inicial y final para un problema concreto. Esto se puede hacer usando la misma sintaxis de **un sistema de producción**: hechos para los estados, y reglas para los operadores.
- ▶ El formato de los estados y operadores tiene que escribirse de forma genérica (con **variables**); mientras que el estado inicial o final se escriben con **valores** particulares para estas variables.
- ▶ Para indicar cuál es la **solución** obtenida por un algoritmo, hay que expresar la secuencia de **acciones** (si tienen nombre), así como el **número** de nodos generados y expandidos.

Convenciones sobre la simulación de los algoritmos

- ▶ Para **simular la ejecución** de un algoritmo **no basta indicar la solución**, sino que hay que dejar claro el orden de expansión de los nodos. Se puede usar uno cualquiera de los siguientes formatos (en algún ejercicio se especifica cuál):
 - ▶ **Una tabla** en la que cada línea muestre la acción tomada, nodo expandido, y el estado del algoritmo (lista abierta con todos los detalles necesarios).
 - ▶ **Un árbol** donde cada nodo contiene un estado, etiquetado con el orden en que ha sido generado y expandido. Si es posible, cada arco tiene que etiquetarse con el nombre (y parámetros si hay) de la acción utilizada para expandir el nodo.
- ▶ En cualquier caso, cuando escribimos un estado, hay que escribir **toda su información**. Ejemplo: si es un móvil que puede llevar una carga, tendremos su posición y la carga que lleva. No puede ocurrir que dos situaciones **distintas** estén representadas igual en el árbol de búsqueda.
- ▶ **Control de estados repetidos** (convención):
 - ▶ En Amplitud no se generan nodos para estados que ya hayan sido generados antes.
 - ▶ En el resto, no se generan nodos con estados que estén en la rama que conduce al nodo que se está expandiendo.
 - ▶ Para explicar cómo funciona, en estas soluciones a veces escribimos estos nodos, pero no es necesario en una respuesta de examen.

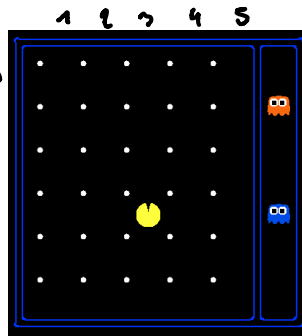
Ejercicio 1: PAC-MAN

~ Los estados son todas las posibles casillas que puede ocupar.

Mira solución, para otra interpretación.
Possible comida 30 estados los 2 posib. comb.

12 pos. para los fantasmas, hay 2 (a lo mas 10x12 = 12 estados)

1. ¿Cuáles son los posibles estados en el entorno del Pacman mostrado en la imagen? ¿Cuántos hay? $5 \times 6 = 30$ estados
2. Supongamos que en este entorno se define un problema en el que el Pacman comienza en una posición inicial y debe llegar a una posición final



Debo encontrar el camino. Si

Casilla (x,y)
 $0 < x < 6$ $0 < y < 7$

- ▶ ¿Es este problema un problema de búsqueda?
- ▶ ¿Cuál es el espacio de estados? ¿Qué tamaño tiene?
- ▶ ¿Cuáles son el estado inicial y el/los estados finales?
- ▶ ¿Cuáles y cómo son las acciones?

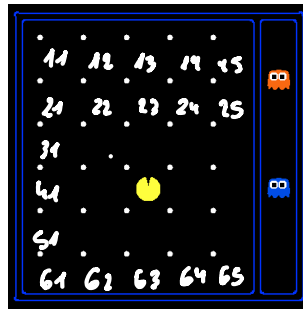
Moverse arriba, abajo, derecha, izquierda pero sin salir del tablero.

Ocho: casilla (x,y) :- casilla (x,y), $x < 5$

¿Dónde comienza PacMan la casilla (3,4) inicial
Los estados finales son — donde termina

1. Supongamos que ahora se define un problema en el que **el Pacman debe comer todas las bolas de comida**
 - ▶ ¿Es éste problema un problema de búsqueda?
 - ▶ ¿Cuál es el espacio de estados? ¿Qué tamaño tiene?
 - ▶ ¿Cuáles son el estado inicial y el/los estados finales?
 - ▶ ¿Cuáles y cómo son las acciones?

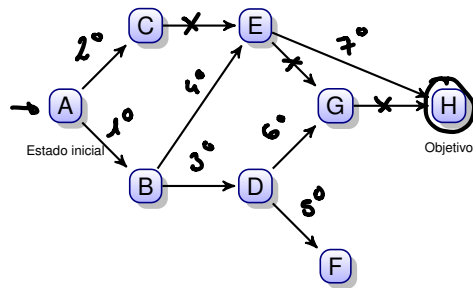
Dcha: casilla $(x+1, y)$:- casilla (x, y) , $x < 5$
Izq : casilla $(x-1, y)$:- casilla (x, y) , $x > 1$
Arriba : casilla $(x, y-1)$:- casilla (x, y) , $y > 1$
Abj : casilla $(x, y+1)$:- casilla (x, y) , $y < 6$



Ejercicio 2: Búsqueda en Grafo

Hacer Búsqueda en Amplitud y Búsqueda en Profundidad para el grafo, donde los arcos representan acciones y los vértices estados, suponiendo que el estado inicial es A y que la meta es llegar al estado representado como H. Mostrar el árbol de búsqueda indicando en qué orden se expande cada nodo, así como el contenido de la lista abierta en cada paso. Considérese que al expandir cada nodo, sus sucesores se generan en orden alfabético. Indique si hay que añadir retroceso para encontrar la solución.

Ampli tud



Lista abierta

0. (A)
1. (B, C)
2. (C, D, E)
3. (D, E)
4. (E, F, G)
5. (F, G, H)
6. Fin

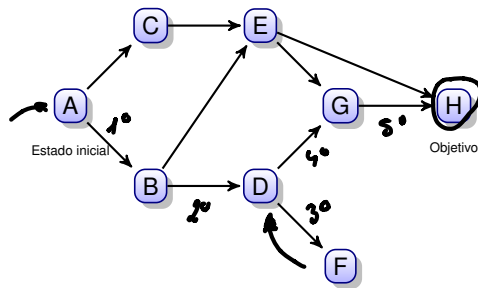
A → B → E → H
Ho encontrado H, el objetivo

Ejercicio 2: Búsqueda en Grafo

Hacer Búsqueda en Amplitud y Búsqueda en Profundidad para el grafo, donde los arcos representan acciones y los vértices estados, suponiendo que el estado inicial es A y que la meta es llegar al estado representado como H. Mostrar el árbol de búsqueda indicando en qué orden se expande cada nodo, así como el contenido de la lista abierta en cada paso. Considérese que al expandir cada nodo, sus sucesores se generan en orden alfabético. Indique si hay que añadir retroceso para encontrar la solución.

Profundidad. lista ^{alfabetico} abierta:

0. (A)
1. (BC)
2. (DEC)
3. (FGE)
 Retrocedo a D
4. (GEC)
5. (HEC) ~
6. Fin


$$A \rightarrow B \rightarrow D \rightarrow G \rightarrow H$$

Ejercicio 3: Búsqueda en Grafo con costes

Aplicar búsqueda en amplitud, profundidad y Dijkstra en el problema descrito en el grafo de la figura, donde los vértices representan estados, los arcos son acciones con sus costes asociados, el estado inicial es S y el objetivo es llegar a uno cualquiera de los estados objetivo {G1,G2}. Para resolver empates, usar el orden alfabético.

Amplitud

(S) 0

(B,C) 1 4

(C,A,E) 4 2 8

(A,E,D,G1) 2 8 9 25

(E,D,G1) 8 9 25

(D,G1,F) 9 25 18

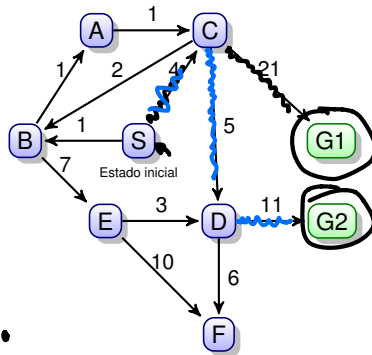
(G1,F,G2) 25 18 20

(F,G2) 18,20 ?

(G2) 20

Se ejecuta G1, y termino
coste 25

coste 20
G2



Ejercicio 3: Búsqueda en Grafo con costes

Aplicar búsqueda en amplitud, profundidad y Dijkstra en el problema descrito en el grafo de la figura, donde los vértices representan estados, los arcos son acciones con sus costes asociados, el estado inicial es S y el objetivo es llegar a uno cualquiera de los estados objetivo {G1, G2}. Para resolver empates, usar el orden alfabético.

Profundidad

(S) 0

(B) 1 ~ (E) 8 No hay más posible

(A) 2

(C) 3 ~ (G1) 24 Coste 23

(D) 8

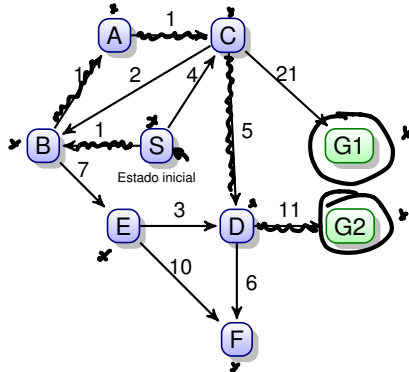
llega a G1

(F) 16

Retrocedo a D

(G2) 19 Coste 19 llegar a G2

Retro a C



Ejercicio 3: Búsqueda en Grafo con costes

Aplicar búsqueda en amplitud, profundidad y Dijkstra en el problema descrito en el grafo de la figura, donde los vértices representan estados, los arcos son acciones con sus costes asociados, el estado inicial es S y el objetivo es llegar a uno cualquiera de los estados objetivo {G1, G2}. Para resolver empates, usar el orden alfabético.

Ir recalculando los que van apareciendo y quedando de menor coste

Dijkstra

(S) 0

(B, C) 1 4

(C, A, E) 4 2 8 (A, C, E) 2 4 8

(C, E) 4 8 No puede avanzar por A

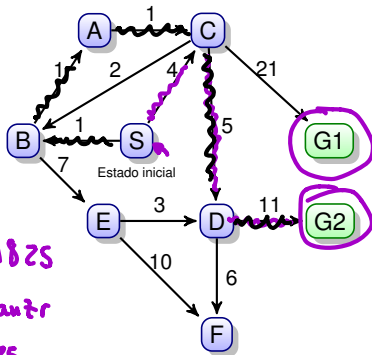
(E, D, G1) 8 9 25

(D, G1, F) 9 25 19 (D, F, G1) 9 18 25

(F, G1, G2) 18 25 20 No puede F avanzar

(G1, G2) 25 20 Llegara G1 coste 25

(G2) 20 Llegara G2 coste 20



(S) 0

(B, C) 1 4

(C, A, E) 4 2 8 (A, C, E) 2 4 8

(C, E) 3 8

(E, D, G1) 8 8 24

(D, G1, F) 8 24 18 (D, F, G1) 8 18 24

(F, G1, G2) 14 24 19 (F, G2, G1) 14 19 24

F sin salida

(G2, G1) 19 24

Coste G2 → 19
Coste G1 → 24

$\min(4, 2+1) = 1$ \rightarrow $\text{A} = \text{C}$
 insert A

	P1	P2	P3	P4	P5	P6	P7	P8	P9
\rightarrow S	0, S								
A	\rightarrow	2, B	2, B						
B	1, S	1, S							
C	4, S	4, S	3, A	3, A					
D	\rightarrow	\rightarrow	\rightarrow	8, C	8, C	8, C			
E	\rightarrow	8, B	8, B	8, B	8, B				
F	\rightarrow	\rightarrow	\rightarrow	\rightarrow	18, E	14, D	14, D		
* G1	\rightarrow	\rightarrow	\rightarrow	24, C	24, C	24, C	24, C	24, C	24, C
* G2	\rightarrow	\rightarrow	\rightarrow	\rightarrow	\rightarrow	19, D	19, D	19, D	

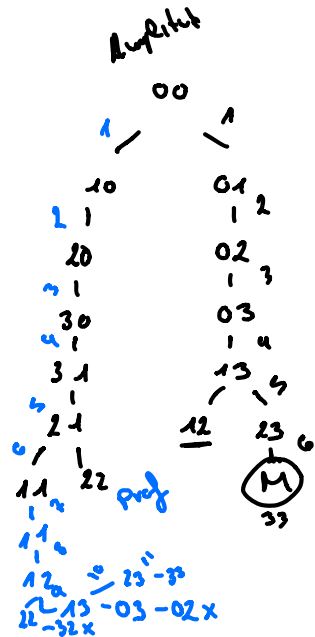
G1: S, B, A, C, G1 19

G2: S, B, A, C, D, G2 24

Cost
 a G2
 19 < 24

Cost a
 G1

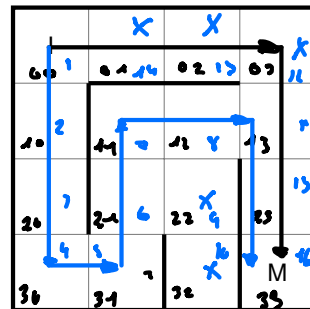
Ejercicio 4: Laberinto



- ▶ El objetivo consiste en encontrar el camino desde el origen **I** hasta la meta **M**
- ▶ Operadores (en orden):
 1. Mover arriba
 2. Mover izquierda
 3. Mover abajo
 4. Mover derecha

Se pide:

1. Representar el problema usando un sistema de producción.
2. Tomando en cuenta el orden, simule la búsqueda en amplitud. Para ello, marque en cada casilla en qué paso el nodo se expande.
3. Haga lo mismo en forma textual, de manera que en cada paso se muestre el nodo expandido y la lista abierta.
4. Hacer búsqueda en profundidad de igual forma.
5. Cambiar el orden de los operadores para que la búsqueda en profundidad encuentre la meta más rápido.



Limite Anib (op) $\underline{0.1} \underline{0.2} \underline{0.9}$
(1.1) (1.2)

Limit izq $(0,0)$, $(1,6)$, $(1,4)$..

Circle Above (3,0) (3,4) (0,4)

Limite Derech (0.9) (1.3.-

call(3,3) ← Est Fin

Castilla (v.c) e Esteri


Arriba: casilla (x, y) , $y > 0$.

Zurück Arriba (x,y) → mod coll / (x,y-1)

se

1. Derecha 3. Izq

2. Abajo 4. Arriba



Ejercicio 5: Lobo, Oveja y Col

► Problema

Un pastor tiene que pasar un lobo, una oveja y una col a la otra orilla de un río, dispone de una barca en la que sólo caben él y una de las otras tres cosas. Si el lobo se queda sin el pastor con la oveja se la come, si la oveja se queda sin el pastor con la col se la come, ¿cómo debe hacerlo?

► Preguntas

1. ¿Cómo se puede representar cada estado?
2. ¿Cuáles son los operadores para la representación escogida?
3. ¿Cuáles son los estados generados a partir del estado inicial?
4. Continuar generando los estados hasta 5 movimientos (para practicar en casa).





Suponga que se tiene el siguiente problema: dos robots situados en distintas casillas de una cuadrícula con algunos obstáculos y deben llegar a una misma casilla objetivo. Cada robot puede moverse en las cuatro direcciones (norte, sur, este y oeste), o quedarse parado en su casilla. También puede limpiar el obstáculo de cualquier casilla adyacente en esas direcciones. Ambos robots pueden coincidir en la misma casilla. No está permitido que se situen sobre un obstáculo. Ambos robots ejecutan acciones de forma simultánea.

Resuelva razonadamente las siguientes cuestiones:

- ▶ ¿Cuáles serían el espacio de estados y las acciones para resolver este problema con un algoritmo de búsqueda?
- ▶ Codifique el estado y las acciones usando la notación que prefiera de las vistas en clase. Escriba un solo ejemplo de acción de movimiento y otro de limpieza.

$m \times n$ porque ^{ocupar casilla} $obstáculos$ si no 2
 $(m \times n)(m \times n) 2^{m \times n}$
Estados tablero.

- ▶ Se tienen dos jarras de agua: una de cinco litros de capacidad y otra de tres. Ninguna de ellas tiene marcas de medición. Se tiene un grifo que permite llenar las jarras de agua (completas), también se pueden vaciar o traspasar el contenido de una jarra a otra. Si se traspasa, por ejemplo, el contenido máximo de la jarra grande a la pequeña cuando la pequeña no está vacía, la jarra grande acabará con dos litros y la pequeña se llenará. Si la pequeña contuviera ya un litro, la jarra grande acabará con tres litros.
- ▶ Inicialmente las jarras están vacías y se desea que la jarra de cinco litros de capacidad contenga exactamente cuatro litros.

Formalizar como problema de búsqueda y mostrar cómo ejecutaríamos búsqueda en amplitud para resolverlo.

Ver online en: <http://www.mathsisfun.com/games/jugs-puzzle.html>

maximo (5,3) ~ unica

Grande 5 l / pequeña 3 litros

Capacidad (q, p) $0 \leq q \leq 5$ $0 \leq p \leq 3$

Inicial capacidad (0,0)

Objetivo capacidad (4,0)

Operaciones: Llenar: capacidad (0, y) \rightarrow modify capacidad (5, y)

Capacidad (x, 0) \rightarrow modify capacidad (x, 3)

Vaciar: capacidad (x, y) \rightarrow modify capacidad (0, y)

Capacidad (x, y) \rightarrow modify capacidad (x, 0)

Traspasar: capacidad (x, y) \rightarrow modify $C = (x+y) \bmod 5$

$x+y \geq 5$

Capacidad (5, x+y-5)

$$5 + 3 = 5 \quad 3 \quad 8$$

$$4 + 3 = 5 \quad 2 \quad 7$$

$$3 + 3 = 5 \quad 1 \quad 6$$

$$2 + 3 = 5 \quad 0 \quad 5$$

Capacidad (x, y), $x+y < 5 \rightarrow$ modify capacidad (x+y, 0)

Capacidad (x, y), $x+y < 3 \rightarrow$ modify capacidad (0, x+y)

Capacidad (x, y), $x+y > 3 \rightarrow$ modify $C = (x+y) \bmod 3$,

Capacidad (x+y-3, 3)

$$5 + 3 = 5 \quad 3 \quad 8$$

$$5 + 2 = 4 \quad 3 \quad 7$$

$$5 + 1 = 3 \quad 3 \quad 6$$

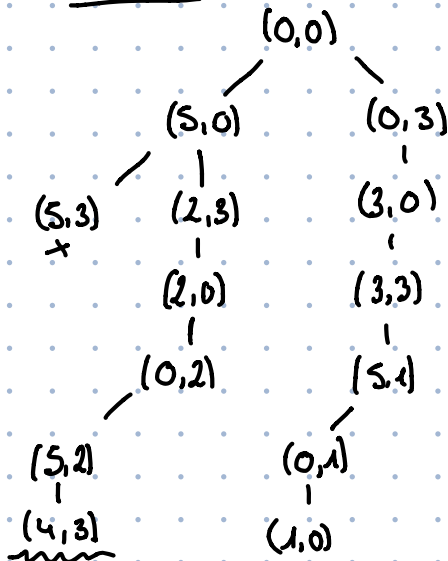
$$5 + 0 = 2 \quad 3 \quad 5$$

$$4 + 0 = 1 \quad 3 \quad 4$$

$$3 + 1 = 1 \quad 3 \quad 4$$

$$3 + 2 = 2 \quad 3 \quad 5$$

Amplitud



$$1 + 3$$

$$0$$

$$5 - 1 = 4$$



- Un grupo de personas tiene que cruzar un puente. Sólo dos personas pueden cruzar a la vez, como mucho, y una debe llevar una antorcha. Sólo disponemos de una antorcha. Además, son de distintas edades y por lo tanto cada una cruza a distinta velocidad: tardan 1,2,5 y 10 minutos respectivamente. Inicialmente están todos a un mismo lado del puente.

Formalizar como problema de búsqueda y mostrar cómo ejecutaríamos búsqueda en amplitud para resolverlo.

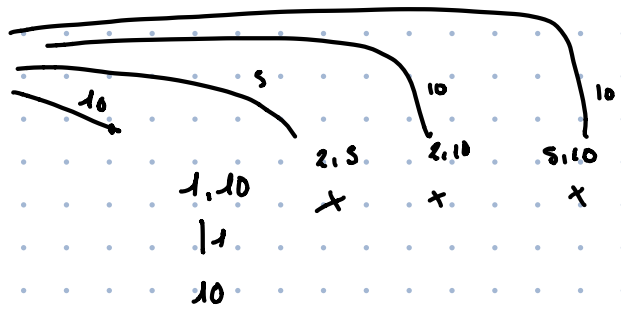
Ver online en: <https://www.inwebson.com/demo/cross-the-bridge/>

0
 $2 \{ 1, 2$
 $1 \{ 1$
 $10 \{ 5, 10$
 $2 \{ 2$
 $2 \{ 1, 2$
17 min

$5, 10 | 1, 2$
 $1, 5, 10 | 2$
 $1 | 5, 10, 2$
 $1, 2 | 5, 10$
 $0 | 1, 2, 5, 10$

2

$1, 2, 5, 10 |$
 $\downarrow 5$
 $1, 5$
 $1, 1$
 5



$1, 2$
 $1, 2$
 10
 $5, 10, 2$
 $2, 1$
 $5, 10$
 $2, 1$
 $1, 2, 5, 10$
Cost 17