Representación de problemas y Búsqueda sin información.

Introducción a la Inteligencia Artificial

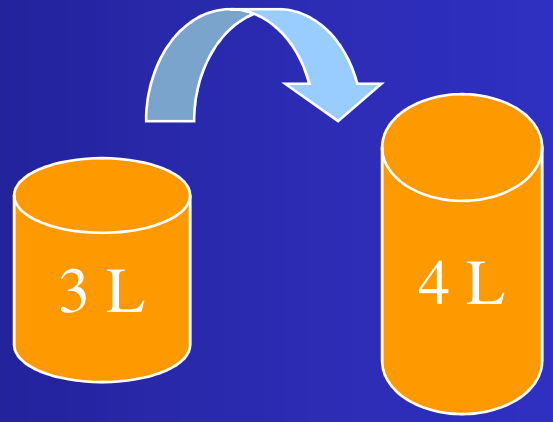
Ana Casali

Cómo desarrolar un programa que encuentre la solución al siguiente problema????

EL PROBLEMA DE LAS JARRAS DE AGUA (DE LOS GALONES)

Se dan dos jarras una de 3 litros y otra de 4 litros sin ningún tipo de marca. Hay una canilla donde se las puede llenar y se puede derramar agua al piso.

Como se puede llegar a tener exactamente 2 litros de agua en la jarra de 4 litros???



INTELIGENCIA ARTIFICIAL

SOLUCION DE CIERTOS PROBLEMAS

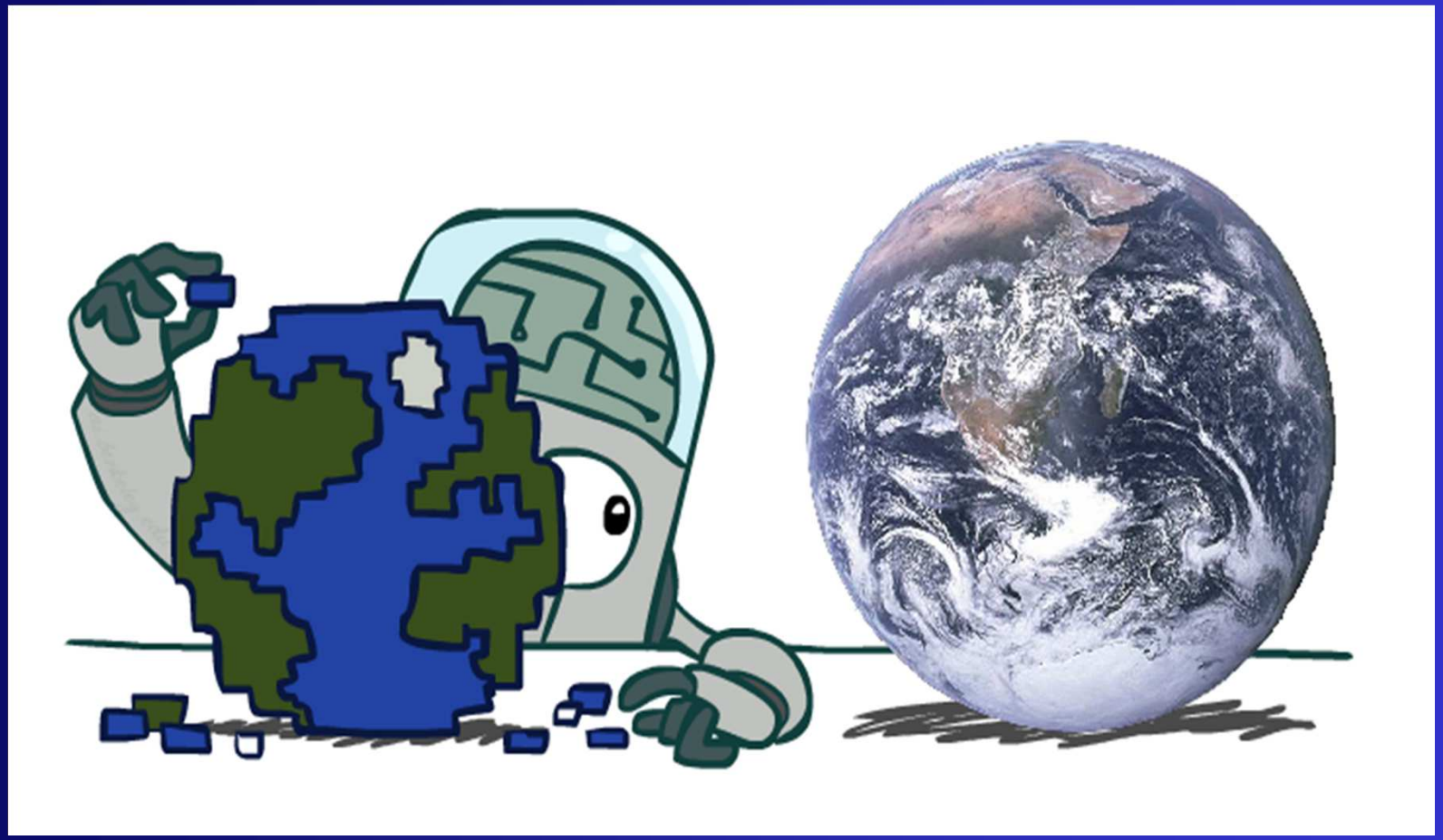
UTILIZANDO LA BÚSQUEDA EN UN ESPACIO DE ESTADOS

Problemas de Búsqueda

Problemas de Búsqueda

* Cómo formular un problema de modo que permita la construcción de un proceso para la búsqueda de soluciones???

Los problemas de búsqueda son Modelos



RESOLUCION DE PROBLEMAS

- FORMULACION DE PROBLEMAS MEDIANTE ESPACIO DE ESTADOS

Representación Formal.

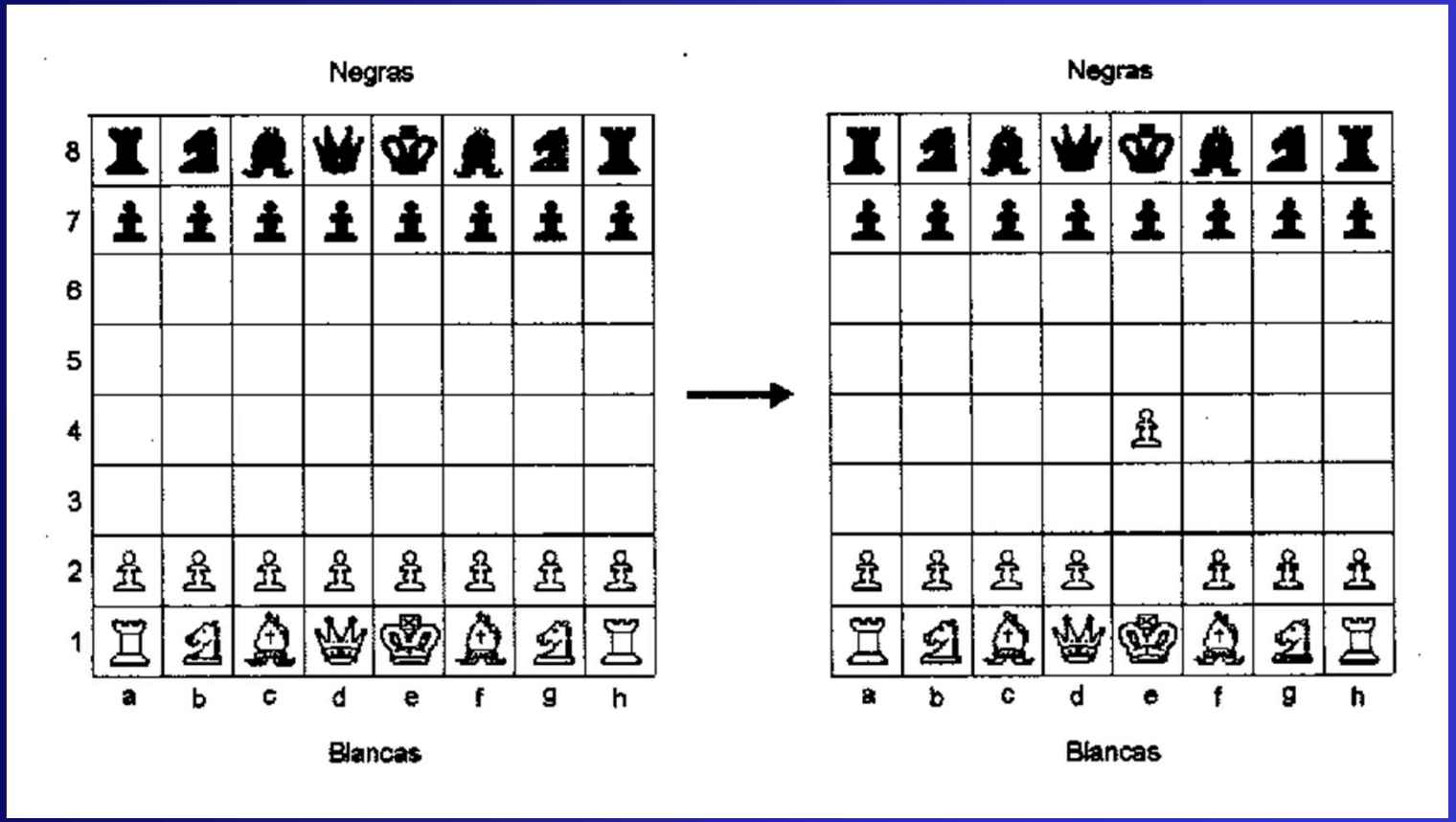
- APLICACIÓN DE DISTINTAS TECNICAS DE BUSQUEDA

Secuencias de operadores.

Una solución es una secuencia de acciones (un plan) que permite transformar el estado inicial en un estado meta

AJEDREZ:

Estado inicial y representación de una jugada



AJEDREZ:

- OBJETIVO: Ganar al oponente: Jaque Mate
- CONJUNTO DE ESTADOS : distintas posiciones legales de las piezas en el tablero
- ESTADO INICIAL: posición inicial
- ACCIONES U OPERADORES: posibles movimientos:

peón-B en (e,2) y vacío (e,3) y vacío (e,4) 
mueve peón-B desde (e,2) hasta (e,4)

EL PROBLEMA DE LAS JARRAS DE AGUA (DE LOS GALONES)

Se dan dos jarras una de 3 litros y otra de 4 litros sin ningún tipo de marca.

Hay una canilla donde se las puede llenar y se puede derramar agua al piso.

Como se puede llegar a tener exactamente 2 litros de agua en la jarra de 4 litros???

EL PROBLEMA DE LAS JARRAS DE AGUA FORMULACIÓN

- CONJUNTO DE ESTADOS : (x, y)

x: representa el contenido de la jarra de 4 lts

y: representa el contenido de la jarra de 3 lts

- ESTADO INICIAL: (0, 0)

- ESTADO FINAL: (2, n), $n \leq 3$

- ACCIONES U OPERADORES: posibles reglas.
  - $(x, y) \wedge x < 4 \rightarrow (4, y)$

FORMULACION DEL PROBLEMA:

- ESTABLECER LA META U OBJETIVO
  (o función evaluadora de meta)

- CONJUNTO DE ESTADOS
  (estado/s inicial, estructura de estados)

- ACCIONES U OPERADORES
  (ESTADO i $\rightarrow$ ESTADO j)

1	(x, y) si $x < 4$	$\rightarrow (4, y)$	Llenar la jarra de 4 litros
2	(x, y) si $y < 3$	$\rightarrow (x, 3)$	Llenar la jarra de 3 litros
3	(x, y) si $x > 0$	$\rightarrow (x - d, y)$	Vaciar un poco la jarra de 4 litros
4	(x, y) si $y > 0$	$\rightarrow (x, y - d)$	Vaciar un poco la jarra de 3 litros
5	(x, y) si $x > 0$	$\rightarrow (0, y)$	Vaciar la jarra de 4 litros en el suelo
6	(x, y) si $y > 0$	$\rightarrow (x, 0)$	Vaciar la jarra de 3 litros en el suelo
7	(x, y) si $x + y \geq 4$ e $y > 0$	$\rightarrow (4, y - (4 - x))$	Verter agua desde la jarra de 3 litros a la jarra de 4 litros hasta que la jarra de 4 litros esté llena
8	(x, y) si $x + y \geq 3$ y $x > 0$	$\rightarrow (x - (3 - y), 3)$	Verter agua desde la jarra de 4 litros a la jarra de 3 litros hasta que la jarra de 3 litros esté llena
9	(x, y) si $x + y \leq 4$ e $y > 0$	$\rightarrow (x + y, 0)$	Verter todo el agua de la jarra de 3 litros en la jarra de 4 litros
10	(x, y) si $x + y \leq 3$ y $x > 0$	$\rightarrow (0, x + y)$	Verter todo el agua de la jarra de 4 litros en la jarra de 3 litros
11	$(0, 2)$	$\rightarrow (2, 0)$	Verter 2 litros de la jarra de 3 litros en la jarra de 4 litros
12	$(x, 2)$	$\rightarrow (0, 2)$	Vaciar la jarra de 4 litros en el suelo

EL PROBLEMA DE LAS JARRAS DE AGUA

Cuestiones destacables:

1. Hay operadores que nos conducen a estados que resultan más útiles para lograr la solución (operador 1).

2. Pueden surgir operadores que nunca nos acerquen a una solución (operadores 3 y 4).

3. Es importante capturar conocimiento sobre casos especiales que conduzcan a la solución. (operadores 11 y 12).

DEFINICIÓN DE LOS OPERADORES

Aspectos a tener en cuenta:

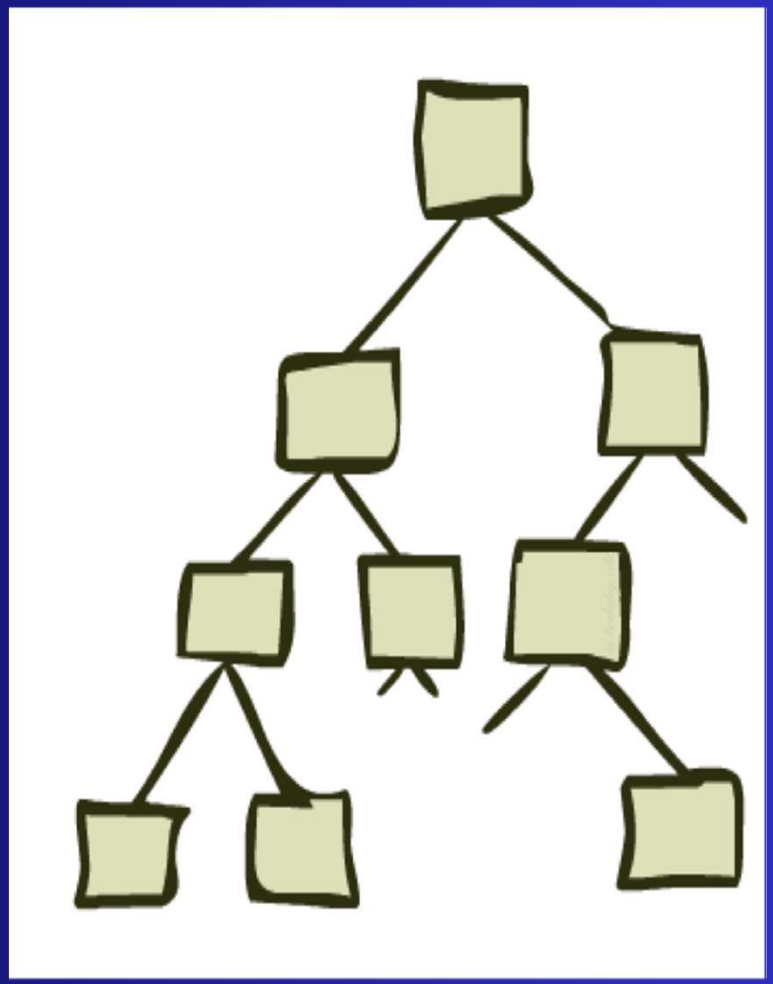
1. Suposiciones presentes en la descripción informal del problema que no están expresadas como tales (es útil representarlas).

2. Nivel de generalidad de las reglas (recordar representar casos especiales de utilidad).

- no incluir reglas inútiles

- pensar en un conjunto que resuelva más de un caso

Grafos de espacio de estados y Árboles de búsqueda



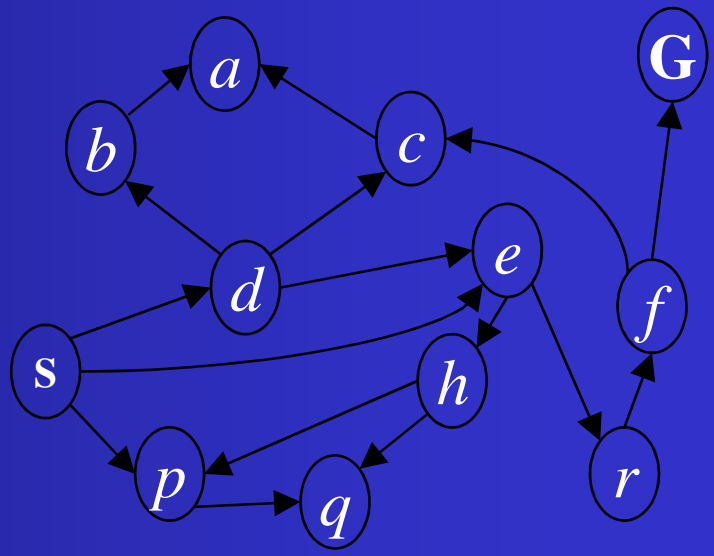
Grafos de espacios de estados

Grafo de Espacio de Estados: es una rep. matemática de un problema e búsqueda

- Nodos son configuraciones del mundo (abstracciones)
- Arcos representan sucesores (resultados de las acciones)
- El test de goal es un conjunto de nodos meta (puede ser uno sólo)

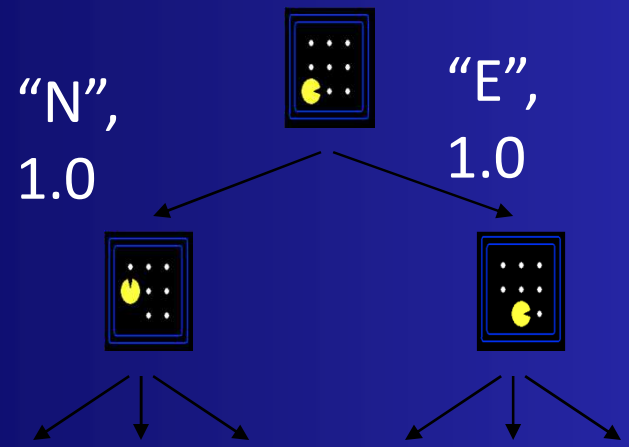
En un grafo de búsqueda, cada estado ocurre una sola vez!

Raramente se construye todo el grafo en memoria (es muy grande) pero es una buena idea



Grafo pequeño para un problema pequeño

Árboles de búsqueda



Comienzo/nodo actual

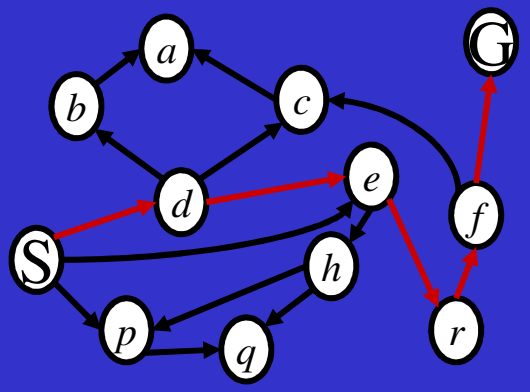
Posibles movimientos

Un árbol de búsqueda (search tree):

- Es un árbol de planes "what if" y sus salidas
- El nodo de inicio es el nodo raíz
- Los hijos corresponden a sucesores
- Los nodos representan estados , pero corresponden a PLANES para alcanzar esos estados
- Para la mayoría de los problemas no se construye todo el árbol

Espacio de estados vs. Árboles de búsqueda

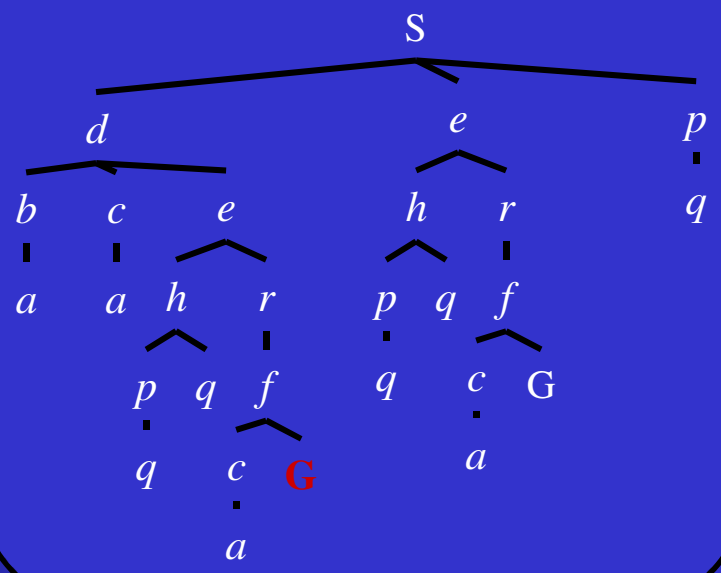
State Space Graph



Cada NODO en un árbol de búsqueda representa todo un camino en el grafo

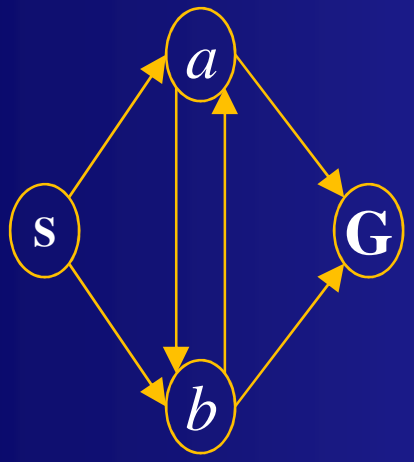
Se construyen ambos según demanda – y se los construye tan pequeños como sea posible

Search Tree



Espacio de estados vs. Árboles de búsqueda

Consideremos este grafo de 4 estados



Cuán grande es el árbol de búsqueda (S estado inicial)?

Importante: Hay mucha estructura repetida en un árbol de búsqueda!

Búsqueda:

- Es el proceso de evaluar las distintas secuencias de acciones (planes) para encontrar las que me lleven

  DEL ESTADO INICIAL ⇨ AL ESTADO META.

- Es muy importante en la solución de problemas de IA, si no existen técnicas mas directas.

Estrategias de Búsqueda:

- Deben ocasionar cambios.

Una estrategia que no cause cambios nunca alcanzará una solución del problema.

- Deben ser sistemáticas.

Es necesario un cambio global (en el curso de varios pasos), esto evita la reiteración de secuencias de operadores poco apropiados.

- Deben ser eficientes.
  - *Completitud
  - *Complejidad (espacial y temporal)
  - * Buena solución, óptima?

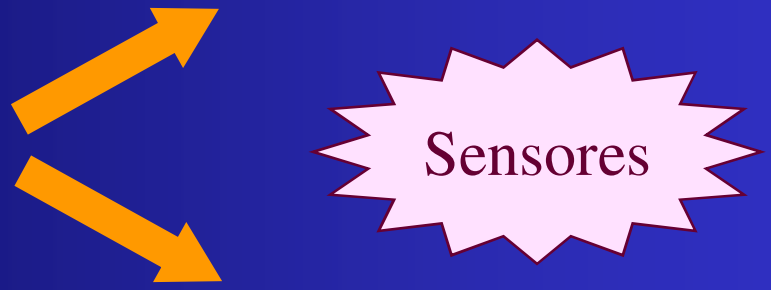
ENFOQUE DE AGENTES:

AGENTE PARA LA SOLUCIÓN DE PROBLEMAS

- Agente formula una *META*
- Agente tiene una *META* y debe esforzarse para alcanzarla.

¿ Conoce su estado actual?

Formulará el problema a encarar dependiendo de:



¿Conoce el resultado de sus acciones?

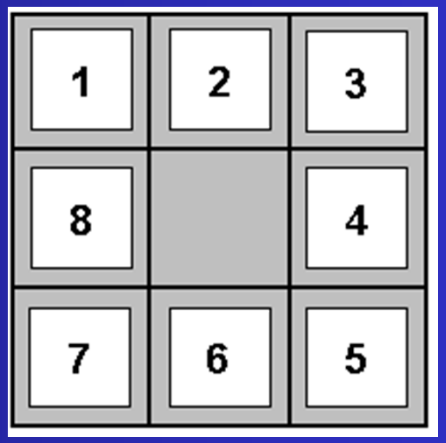
ENFOQUE DE AGENTES:

AGENTE PARA LA SOLUCIÓN DE PROBLEMAS

- FORMULACIÓN DE METAS **(ESTADO FINAL)**
- ESTADO ACTUAL DEL MUNDO **(ESTADO INICIAL)**
- *ACCIONES POSIBLES* (y por ende estados futuros)

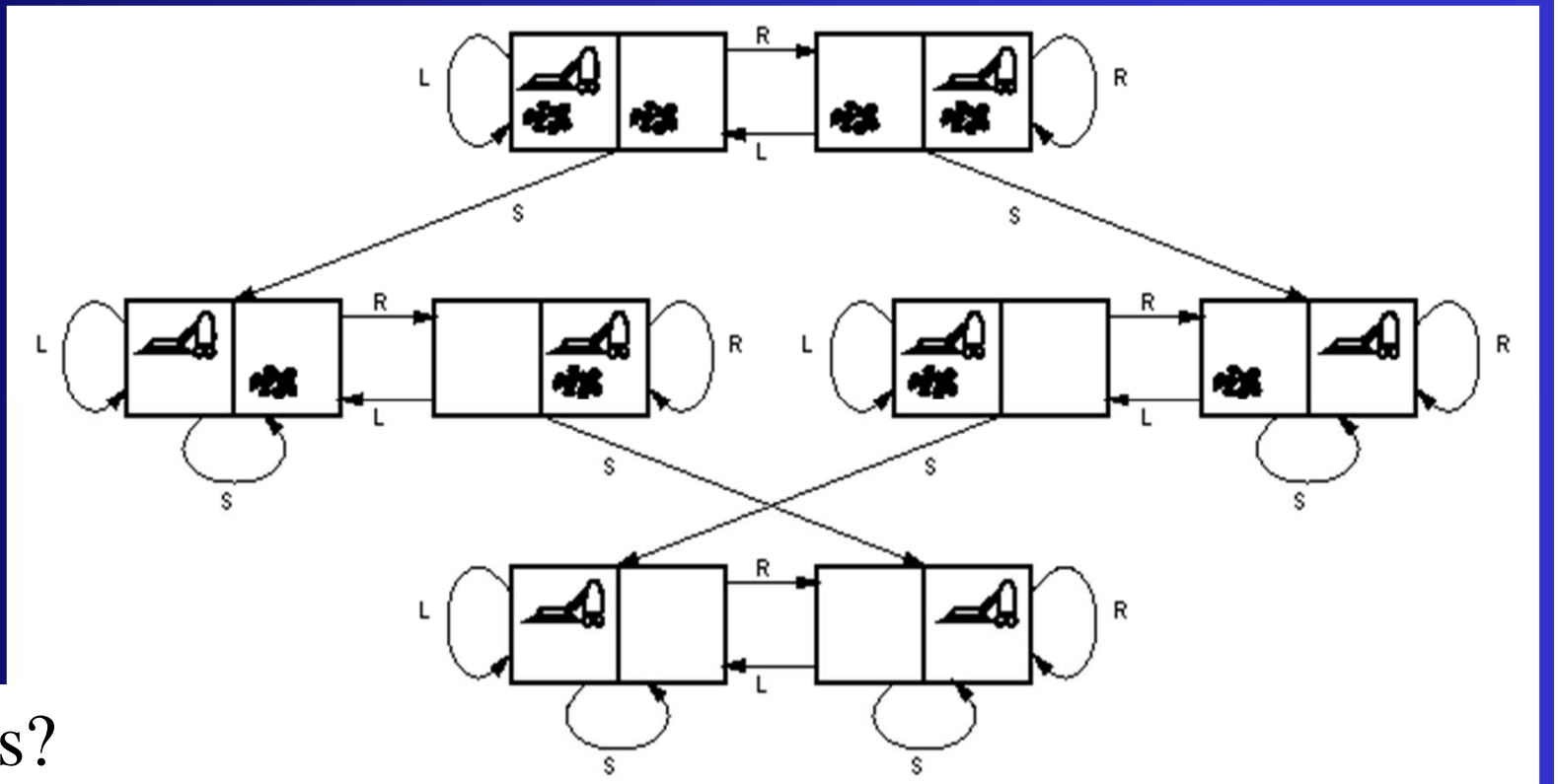
FORMULACIÓN DEL PROBLEMA

Ejemplo - el juego de las 8 fichas



- Estados?
- Operadores?
- Test de Meta?
- Costo de Ruta?

Ejemplo: el espacio del mundo de la aspiradora



- Estados?
- Operadores?
- Test de meta?
- Costo de trayectoria?

AGENTE PARA LA SOLUCIÓN DE PROBLEMAS

Se agrega a la definición de ***PROBLEMA***

* ***PRUEBA DE META:*** Es el estado actual un estado META?

* ***COSTO DE RUTA (g):*** Suma de las acciones individuales (operadores) que se emprendan al recorrerla.

MEDICION DE LA EFICIENCIA DE LA ESTRATEGIA

* ¿Permite encontrar una solución?

* ¿ Es una buena solución? (Bajo costo de ruta)

* ¿Cuál es el costo de búsqueda en tiempo y memoria para encontrar una solución?

AGENTE PARA LA SOLUCIÓN DE PROBLEMAS

- El agente está en ARAD (RUMANIA), al fin de un viaje.
- Mañana abordará un vuelo en BUCAREST (RUMANIA).
- El pasaje no es reembolsable.
- Su visa está a punto de vencer.
- Si pierde el vuelo no hay lugar en otros en seis semanas.

Factores que intervienen:

- ✓ Costo pasaje
- ✓ No ser arrestado
- ✓ Otros...

META

Manejar de Arad a Bucarest.

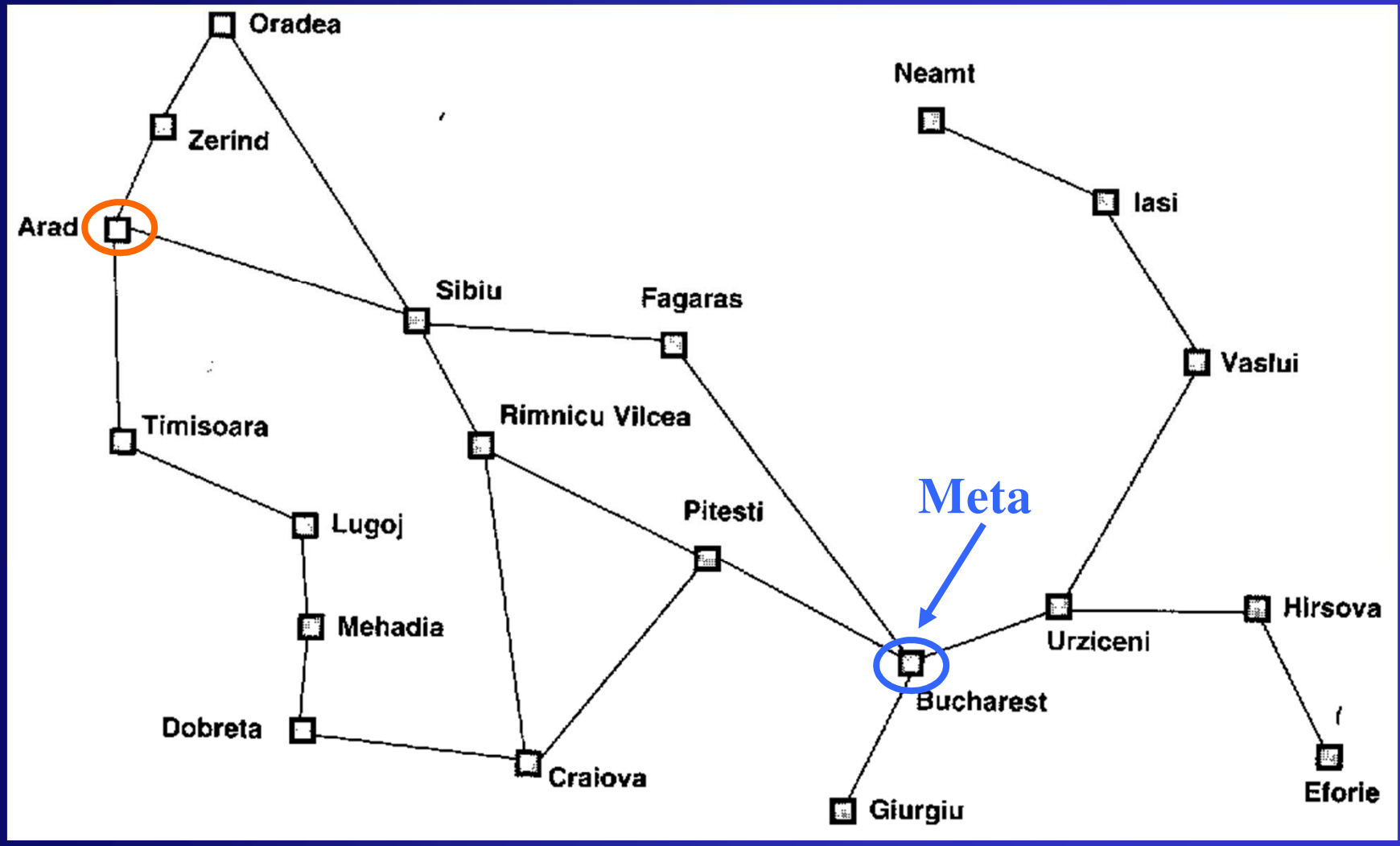
COMO ESCOGER ESTADOS Y ACCIONES ???

El verdadero arte de la solución de problemas consiste en decidir que es lo que servirá para representar los estados y operadores, y que no.

Hay que realizar un proceso de eliminación de los detalles que sean innecesarios
(representación de estado - acciones)

ABSTRACCION

MAPA DE RUMANIA.



ESTADOS Y ACCIONES del problema del agente que viaja de Arad a Bucarest.

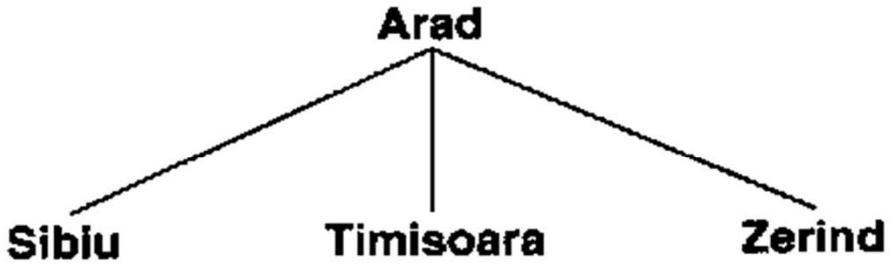
ESTADOS: Ubicación del agente en alguna de las veinte ciudades del mapa.

OPERADORES O ACCIONES: Conducir en el tramo de ruta que lleva de una ciudad a otra según indica el mapa.

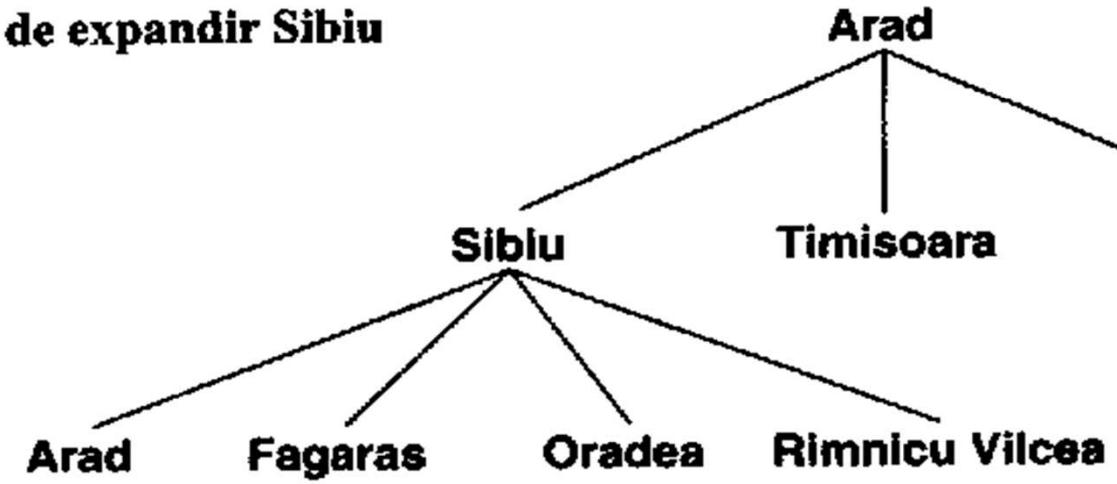
ÁRBOL DE BÚSQUEDA PARCIAL (Arad a Bucarest).

(a) Estado inicial

(b) Después de expandir Arad



(c) Después de expandir Sibiu



GENERACIÓN DE SECUENCIAS DE ACCIONES

EXPANSIÓN DE UN NODO

Se aplican operadores al nodo elegido → **Se genera un nuevo conjunto de nodos.**

ÁRBOL DE BÚSQUEDA

- ✓Su raíz corresponde al estado inicial.
- ✓Sus hojas son nodos sin sucesores.
- ✓El algoritmo de búsqueda elige el nodo a expandir.

NODOS Y ESTADOS

ESTADO → Conjunto de configuraciones del mundo

NODO → Estructuras de datos que sirven para representar un estado en el ***árbol de búsqueda*** de un problema específico.

- ✓ El estado que le corresponde.
- ✓ Nodo padre.
- ✓ Operador que lo generó.
- ✓ Profundidad del nodo.
- ✓ Costo de ruta.

ALGORITMO DE BÚSQUEDA GENERAL.

BÚSQUEDA GENERAL

responde con SOLUCIÓN o FALLA

LISTA-NODOS ← ESTADO INICIAL

bucle hacer

si LISTA-NODOS está vacía contestar FALLA

tomo NODO de LISTA-NODOS

si NODO es ***meta*** contestar con NODO

LISTA-NODOS ← expansión NODO

FIN

Estrategias de Búsqueda:

BÚSQUEDA SIN INFORMACIÓN

El agente sólo puede diferenciar un nodo que es meta de uno que no lo es. No posee información respecto a cuántos pasos necesita dar, o a qué distancia está de la meta.

BÚSQUEDA RESPALDADA POR INFORMACIÓN

El agente posee información sobre el problema como para poder elegir operadores más convenientes.

Las estrategias de BÚSQUEDA SIN INFORMACIÓN se diferencian por el orden en que expanden los nodos.

Propiedades de Algoritmos de Búsqueda

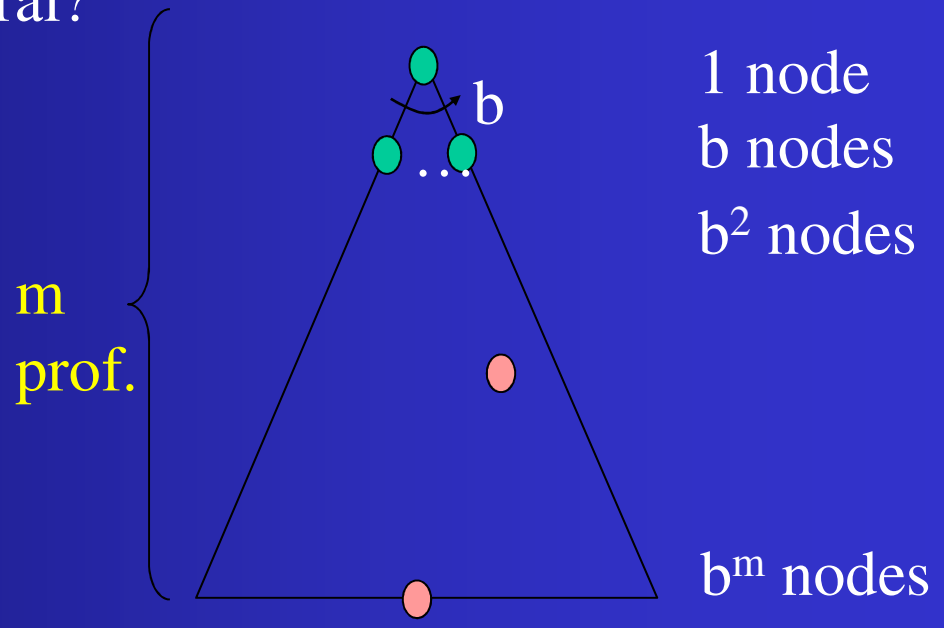
Es Completa: Garantiza encontrar una solución si existe alguna?

Es Óptima: Garantiza encontrar la solución óptima (menor costo)?

Complejidad espacial y temporal?

Nomenclatura:

- b es el factor de ramificación
- m es la profundidad maxima
- Soluciones a distintas prof.

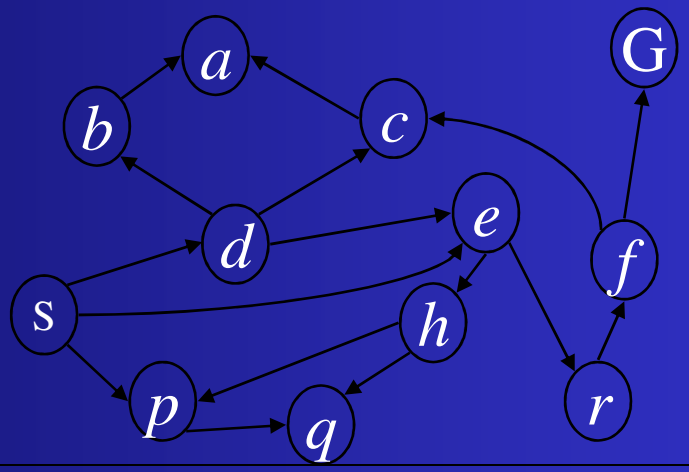


Número de nodos en el árbol?

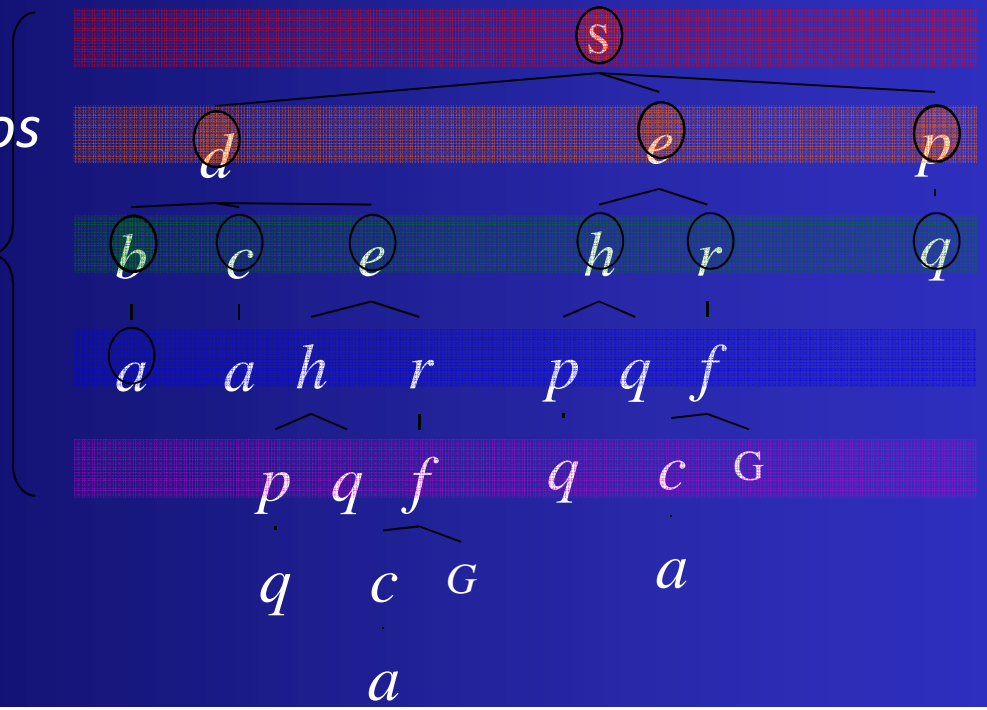
$1 + b + b^2 + \ldots. b^m = O(b^m)$

Búsqueda a lo ancho (Breadth-First)

Estrategia: expandir los nodos menos profundos primero



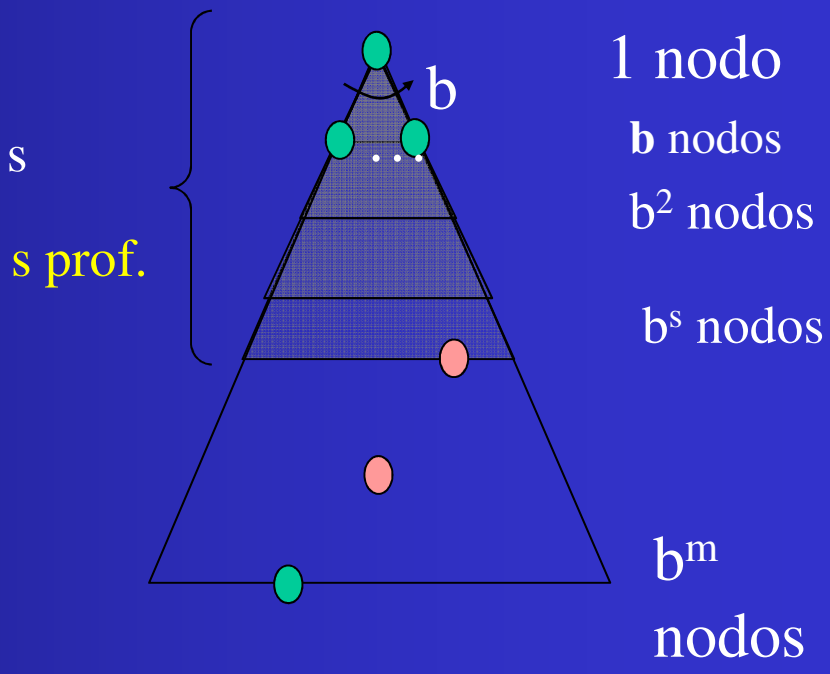
Implementa: Lista de nodos como FIFO



Búsqueda a lo ancho (BFS)Propiedades

Qué nodos expande BFS?

- Procesa todos los nodos arriba de la solución menos profunda
- Sea la solución menos profunda en nivel s
- Tiempo de búsqueda $O(b^s)$



Cuanto espacio ocupa la frontera?

- Aprox el último nivel: $O(b^s)$

Es completa? SI

- s debe ser finito si la solución existe

Es óptima? SI

- Considerando todos los costos de operadores 1

Búsqueda Primero a lo Ancho

Se utiliza el algoritmo de BÚSQUEDA GENERAL colocando los NODOS generados al expandir, al final de la LISTA-NODOS.

- Es completa: Si existe una solución la encontrará.
- Es óptima: Si hay varias soluciones encuentra la más superficial, la mejor si el costo es proporcional a la profundidad.
- Complejidad (espacial y temporal): $O(b^s)$, **dónde b** : factor de ramificación y **s** profundidad de la solución.

Búsqueda de Costo Uniforme

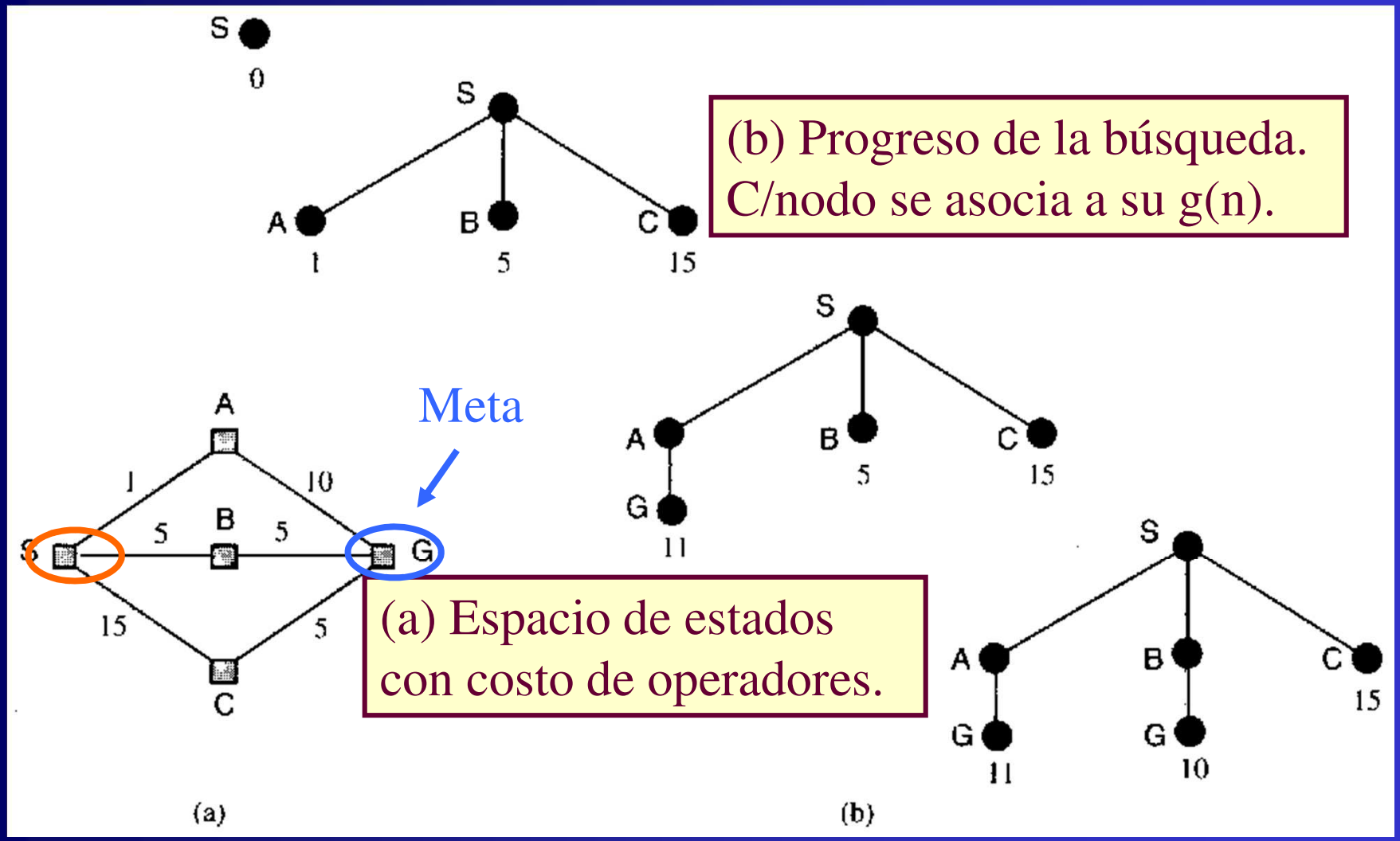
- ✓Expande siempre el nodo de menor costo.
- ✓ El costo de ruta asociado a cada nodo n: g(n).

Si g (n) = profundidad (n) → Búsqueda preferente por amplitud (a lo ancho).

Garantiza obtener la ***solución más barata*** si el costo de ruta nunca disminuye al avanzar. → g(Sucesor(n)) >= g(n)

Búsqueda de Costo Uniforme.

Determinación de ruta de S a G.



Búsqueda de Costo Uniforme: Conclusiones

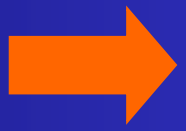
- Es completa: Si existe una solución la encontrará.
- Es óptima: Encuentra la ruta de menor costo, siempre que dicho costo no sea decreciente con la profundidad.
- Complejidad (espacial y temporal): $O(b^s)$, dónde **b** es el factor de ramificación y **s** la profundidad de la solución.

Búsqueda preferente por profundidad – Primero en profundidad

- Se expande siempre uno de los nodos que se encuentra en el nivel más profundo.
- Sólo cuando un nodo no tiene expansión, se revierte la búsqueda y se expanden nodos de niveles menos profundos.

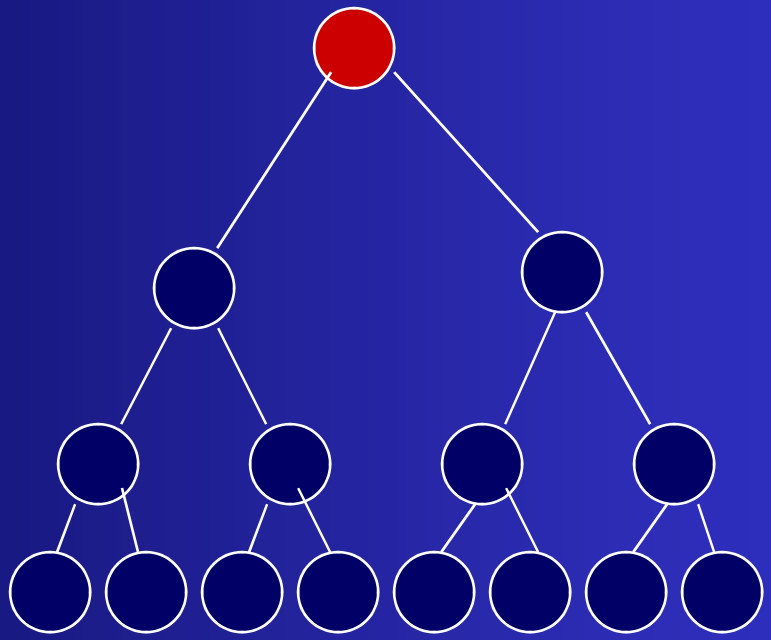
Se utiliza el algoritmo de BÚSQUEDA GENERAL colocando los NODOS generados al expandir, al comienzo de la LISTA-NODOS.

Sólo debe guardarse la ruta y los nodos no expandidos

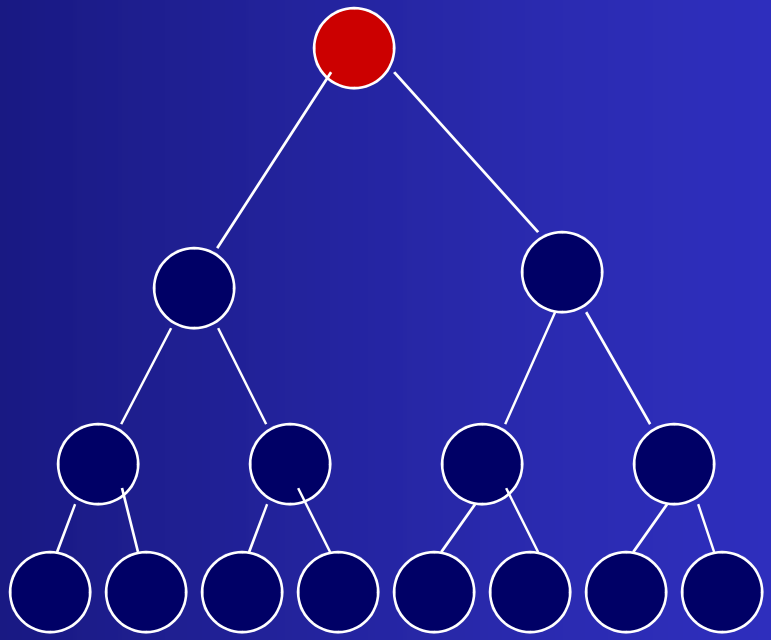


Necesita un volumen menor de memoria.

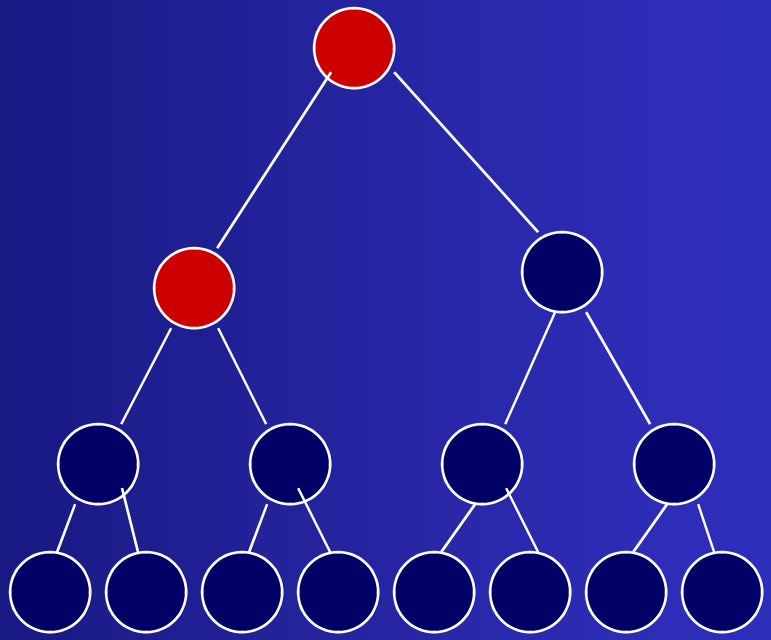
Búsqueda Primero en Profundidad



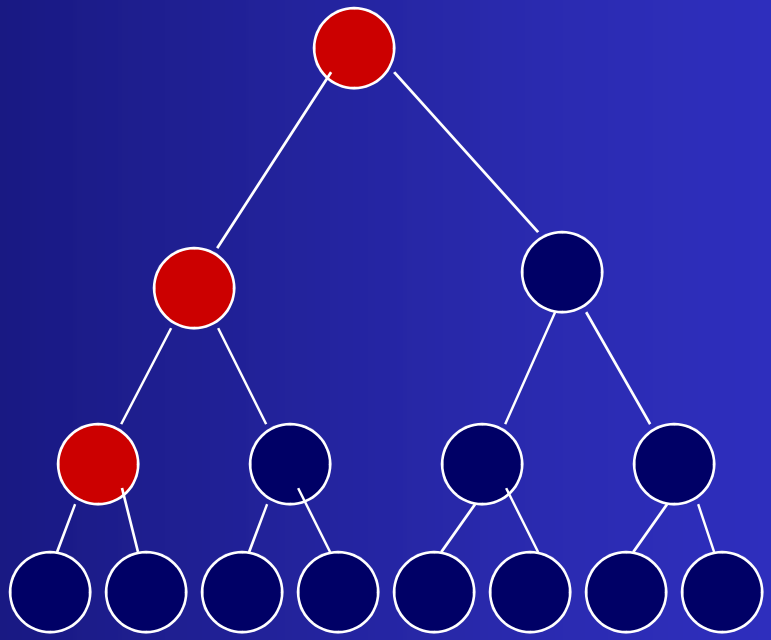
Búsqueda Primero en Profundidad



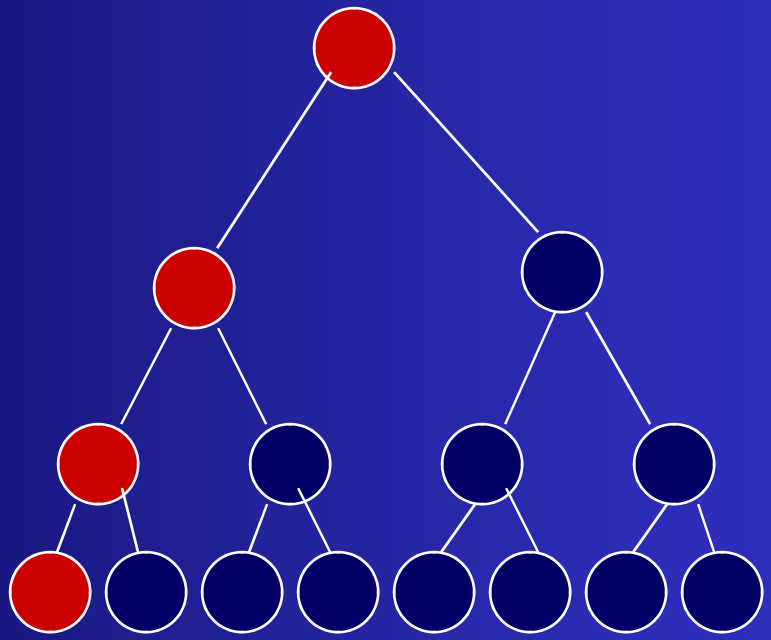
Búsqueda Primero en Profundidad



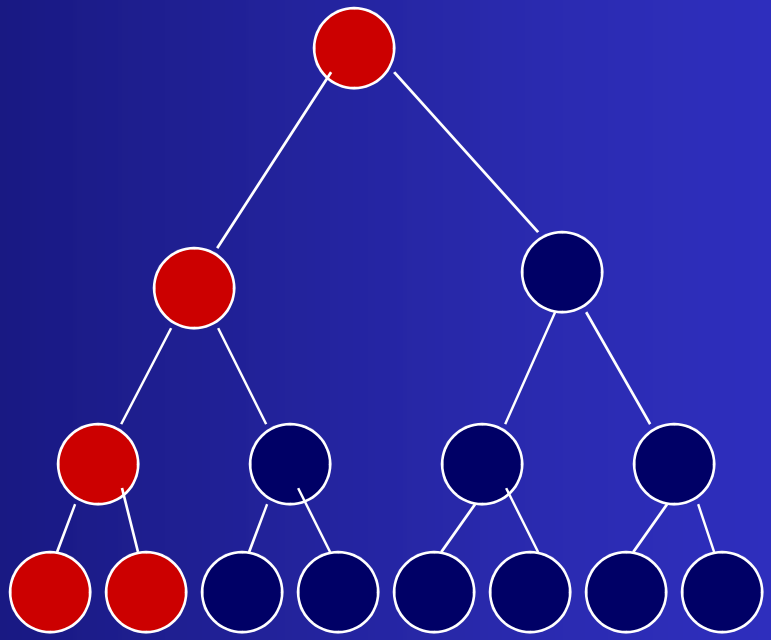
Búsqueda Primero en Profundidad



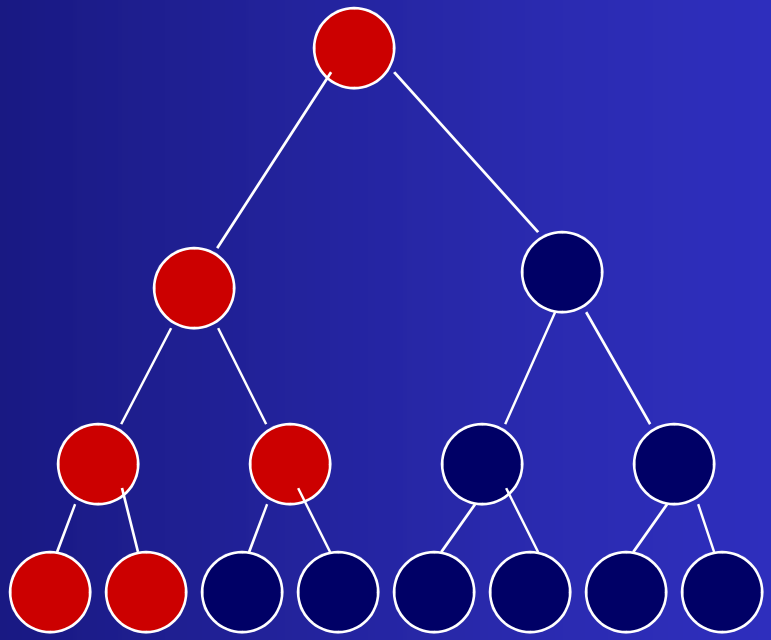
Búsqueda Primero en Profundidad



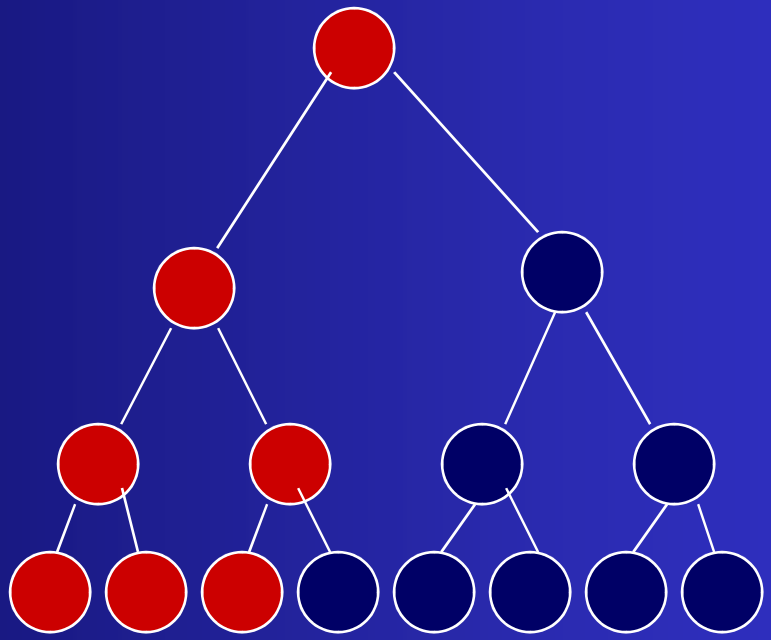
Búsqueda Primero en Profundidad



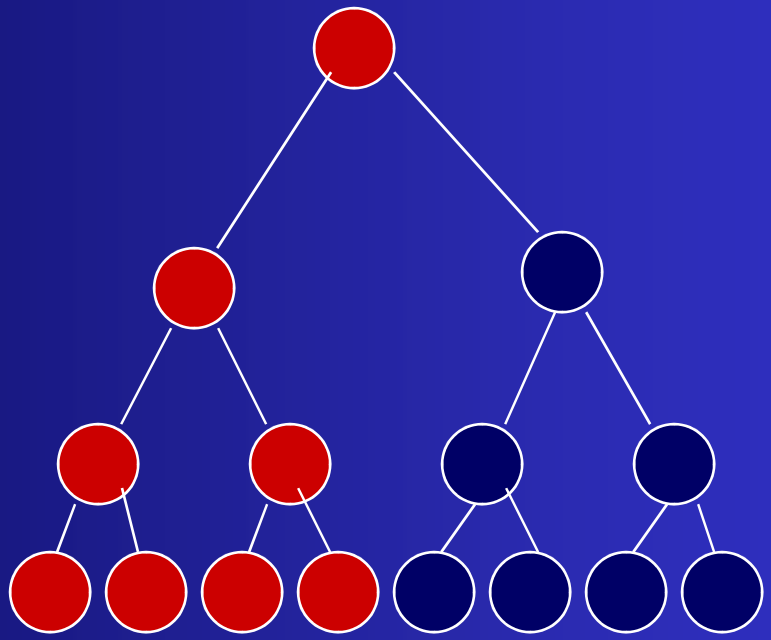
Búsqueda Primero en Profundidad



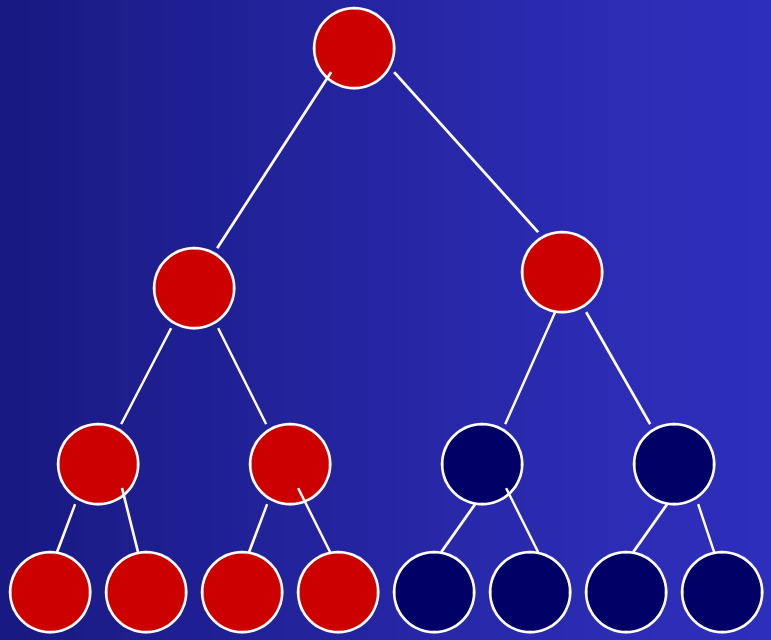
Búsqueda Primero en Profundidad



Búsqueda Primero en Profundidad



Búsqueda Primero en Profundidad



Búsqueda preferente por profundidad: Conclusiones.

* Complejidad espacial $\rightarrow$ O (b.m), m profundidad máxima del árbol de búsqueda

* Complejidad temporal $\rightarrow$ O (b^m)

* No es óptima $\rightarrow$ Puede mejorar si hay muchas soluciones.

* No es completa $\rightarrow$ Puede atascarse en bucles o caminos ∞.

No es aconsejable la búsqueda preferente por profundidad, si el árbol es de gran profundidad máxima.

Búsqueda limitada por profundidad

Se impone un límite máximo a la profundidad de la ruta.

Se elimina la posibilidad de atascamientos de la BPP.

*Se utiliza el algoritmo de BÚSQUEDA GENERAL colocando los NODOS generados al expandir, al comienzo de la LISTA-NODOS.

* Se incluyen operadores que garanticen la vuelta atrás cuando se ha alcanzado el límite, backtracking.

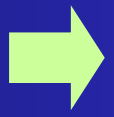
Búsqueda limitada por profundidad: Conclusiones

* Complejidad espacial $\rightarrow$ O (b.l)
* Complejidad temporal $\rightarrow$ O (b^l)
* No es óptima $\rightarrow$ No garantiza encontrar la mejor solución.
* Es completa $\rightarrow$ Si se elige el límite adecuado para el problema.

Si el límite es demasiado pequeño no puede garantizarse completitud. Es aconsejable en problemas donde se tenga idea de un límite razonable (como ir de Arad a Bucarest).

Búsqueda por profundización iterativa.

Resuelve el problema de elegir un límite adecuado.



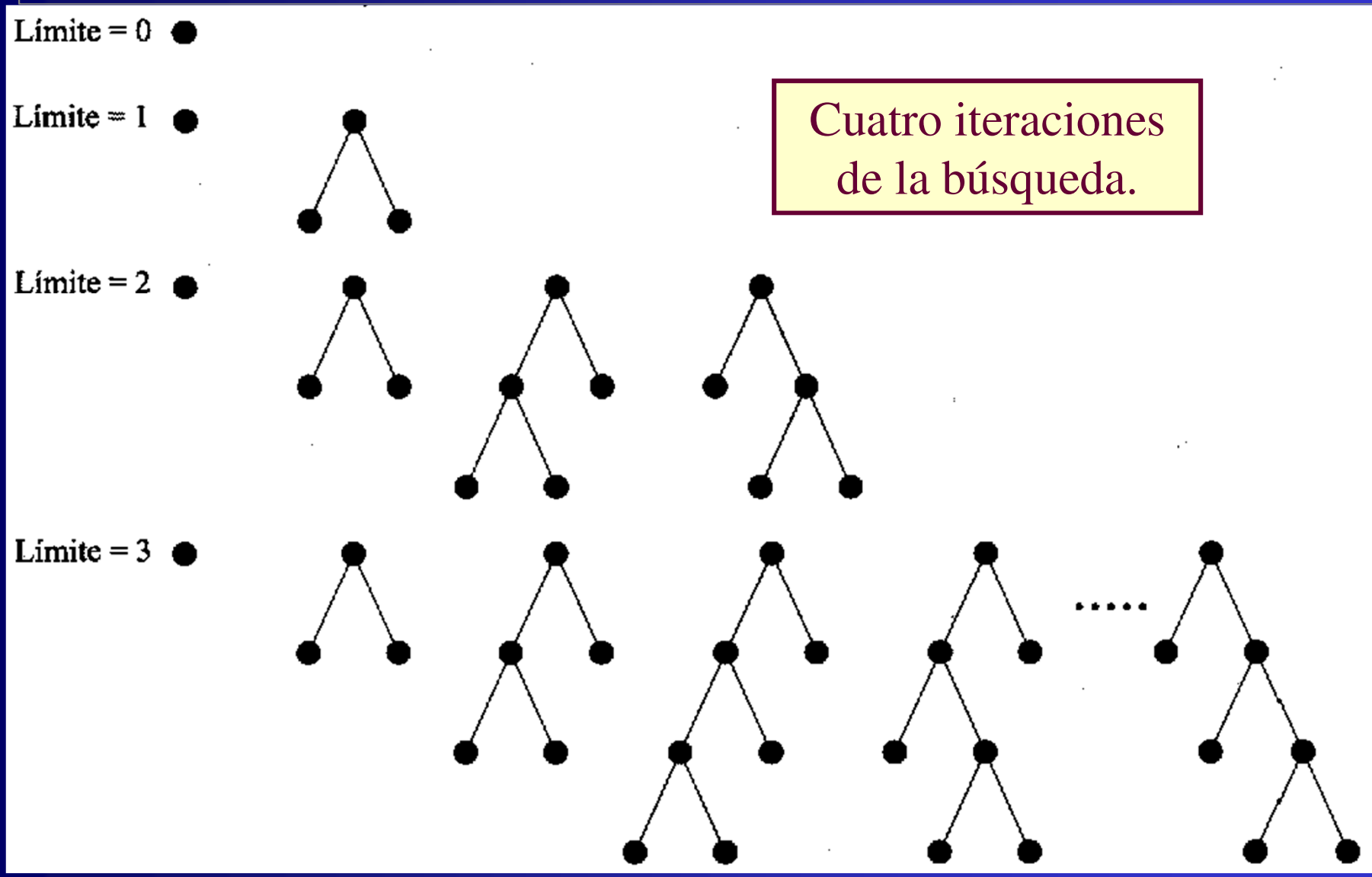
Propone probar todos los límites de profundidad, o sea, $l = 0, 1, 2, 3, \ldots$

Combina ventajas de dos búsquedas anteriores:

* Búsqueda primero profundo → menor consumo de memoria.

* Búsqueda a lo ancho → completa y óptima.

Búsqueda por profundización iterativa - árbol binario.



Cuatro iteraciones de la búsqueda.

Búsqueda por profundización iterativa: Conclusiones.

* Complejidad espacial $\rightarrow$ O (b.d)
* Complejidad temporal $\rightarrow$ O (b^d)
* Es óptima y completa

Desventaja

Los nodos de niveles altos se expanden varias veces. Para b = 10 y d = 5 hay un exceso del 11% respecto a la búsqueda limitada en profundidad.

Es aconsejable en espacios grandes dónde se ignora la profundidad de la solución.

Comparación de las estrategias de búsqueda.

Criterio	Tiempo	Espacio	Óptima?	Completa?
Preferente por amplitud	b^d	b^d	SI	SI
Costo Uniforme	b^d	b^d	SI	SI
Preferente por profundidad	b^m	b.m	NO	NO
Limitada en profundidad	b^l	b.l	NO	SI, cuando l >= d
Profundización iterativa	b^d	b.d	SI	SI
Bidireccional (cuando aplica)	$b^{d/2}$	$b^{d/2}$	SI	SI

El problema de los estados repetidos.

Cómo se evita expandir estados que ya se expandieron en otra ruta?

* En algunos problemas se llega a un estado de una sola forma, y por ende es imposible repetir.

* Cuando los operadores son reversibles, entonces pueden obtenerse árboles infinitos (misioneros y caníbales, rutas).

Evitar las repeticiones disminuye sensiblemente el costo de la búsqueda:

- Los árboles infinitos pueden volverse finitos.
- En árboles finitos la reducción es muy importante.

El problema de los estados repetidos.

Formas de evitar los estados repetidos:

Costo

Eficiencia

1. No regresar al estado del que acaba de llegar.
Sucesor (n) distinto a Padre (n).

2. No generar rutas que tengan ciclos.
Sucesor(n) distinto a Ancestro(n).

3. No generar ningún estado que se haya generado alguna vez.

Hay un compromiso entre el costo de almacenar y verificar y el costo de la búsqueda adicional a realizar.

Bibliografía

- Inteligencia Artificial. Un enfoque moderno – Norvig & Russell – 2da Ed Prentice Hall 2003. (cap 3)
- Materiales curso CS188 Berkeley