

Algoritmos de Búsqueda

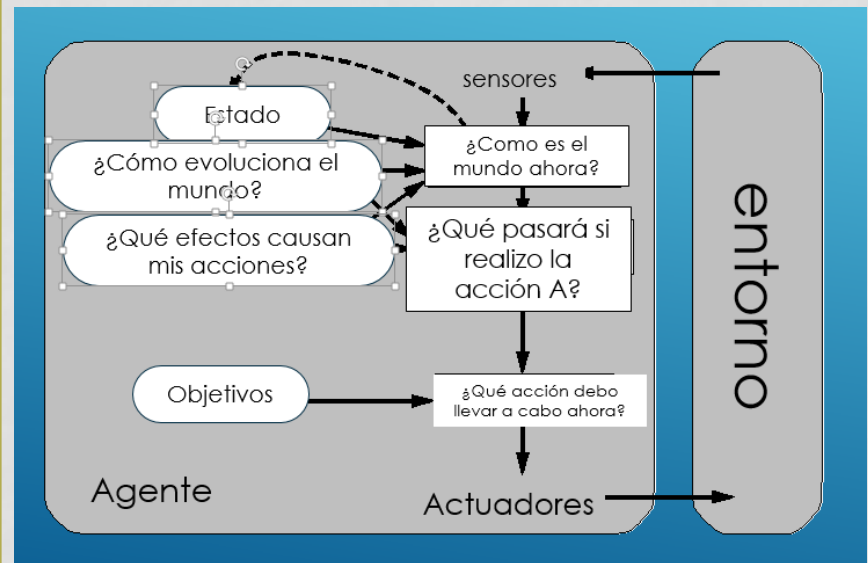
“En donde veremos cómo un agente puede encontrar una secuencia de acciones que alcance sus objetivos, cuando ninguna acción simple lo hará.” Inteligencia Artificial – Un enfoque Moderno



Dra. Ing. SELVA S. RIVERA
PROFESORA TITULAR

AGENTES QUE RESUELVEN PROBLEMAS

- Es un agente basado en objetivos
- Deciden qué hacer para encontrar secuencias de acciones que conduzcan a los estados deseables
- Como todo agente inteligente debe maximizar su medida de rendimiento (esto se puede simplificar si el agente puede elegir un objetivo y trata de satisfacerlo)



PASOS PARA SOLUCIONAR UN PROBLEMA



Formulación del objetivo

A partir de la situación actual, definir los estados objetivo y los factores que puedan influir en el grado de satisfacción de las distintas maneras de conseguirlo.



- **Formulación del problema**

Proceso de decidir qué acciones y estados tenemos que considerar.



- **Búsqueda**

Decidir qué hacer examinando diferentes secuencias de acciones que llevan a estados objetivo y escoger la mejor.



- **Ejecución**

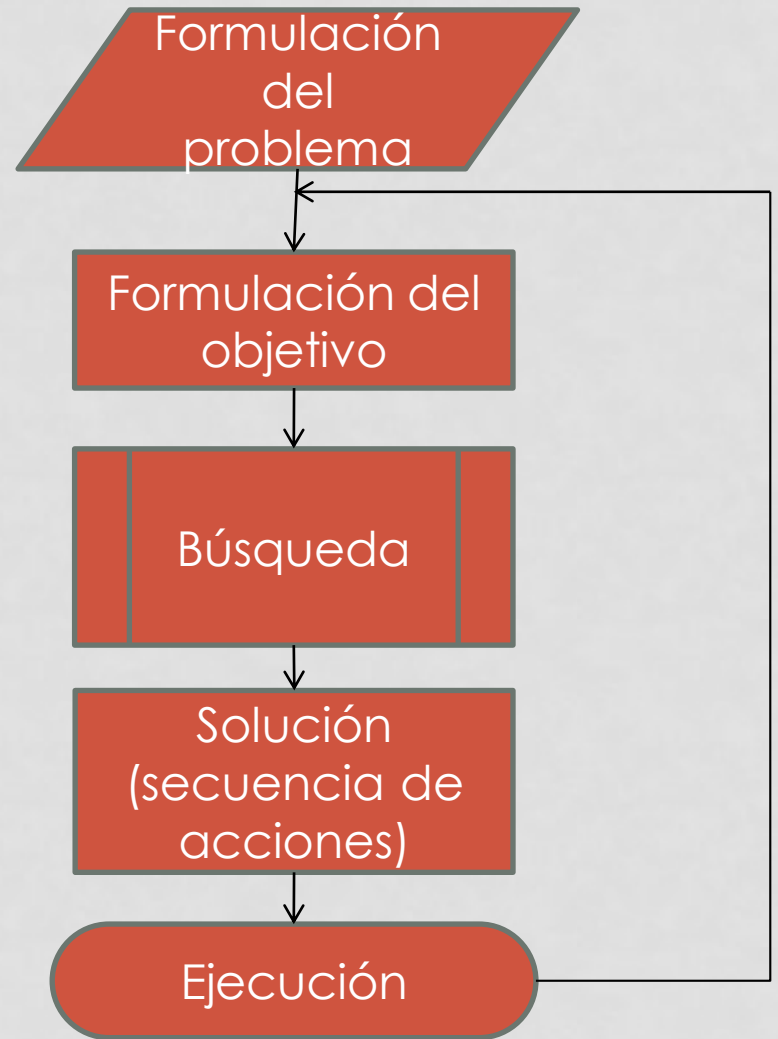
Ejecutar las acciones recomendadas.

BÚSQUEDA



- Un agente con distintas opciones inmediatas con valores desconocidos puede decidir qué hacer; examinando las diferentes secuencias posibles de acciones que le conduzcan a estados de valores conocidos y entonces escoger la mejor secuencia.
- **El proceso de hallar esta secuencia se llama búsqueda**

«FORMULAR, BUSCAR, EJECUTAR»



Entorno:
-estático
-observable
-discreto
-determinista



DEFINICIÓN FORMAL DE PROBLEMA

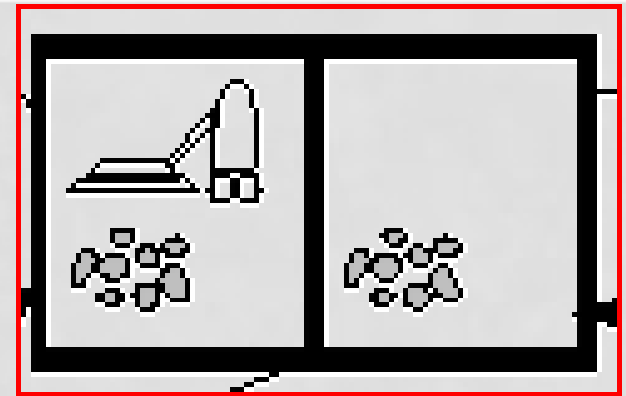


DATOS

- **Estado inicial**
es el estado en que comienza el agente
- **Acciones**
 - Función sucesor
describe las posibles acciones disponibles por el agente
 - Espacio de Estado
definido por el Estado inicial y la función sucesor
- **El test objetivo**
determina si un estado es un estado objetivo
- **Costo del camino**
es una función que asigna un costo numérico a cada camino. Refleja la medida de rendimiento.
- **Solución**
- una solución es un camino desde el estado inicial al estado objetivo
- **Solución óptima**
una solución óptima tiene el menor costo del camino entre todas las soluciones

MUNDO DE JUGUETE Y MUNDO REAL

- **Un problema de juguete** se utiliza para ilustrar o ejercitar los métodos de resolución de problemas. Éstos se pueden describir de forma exacta y concisa.

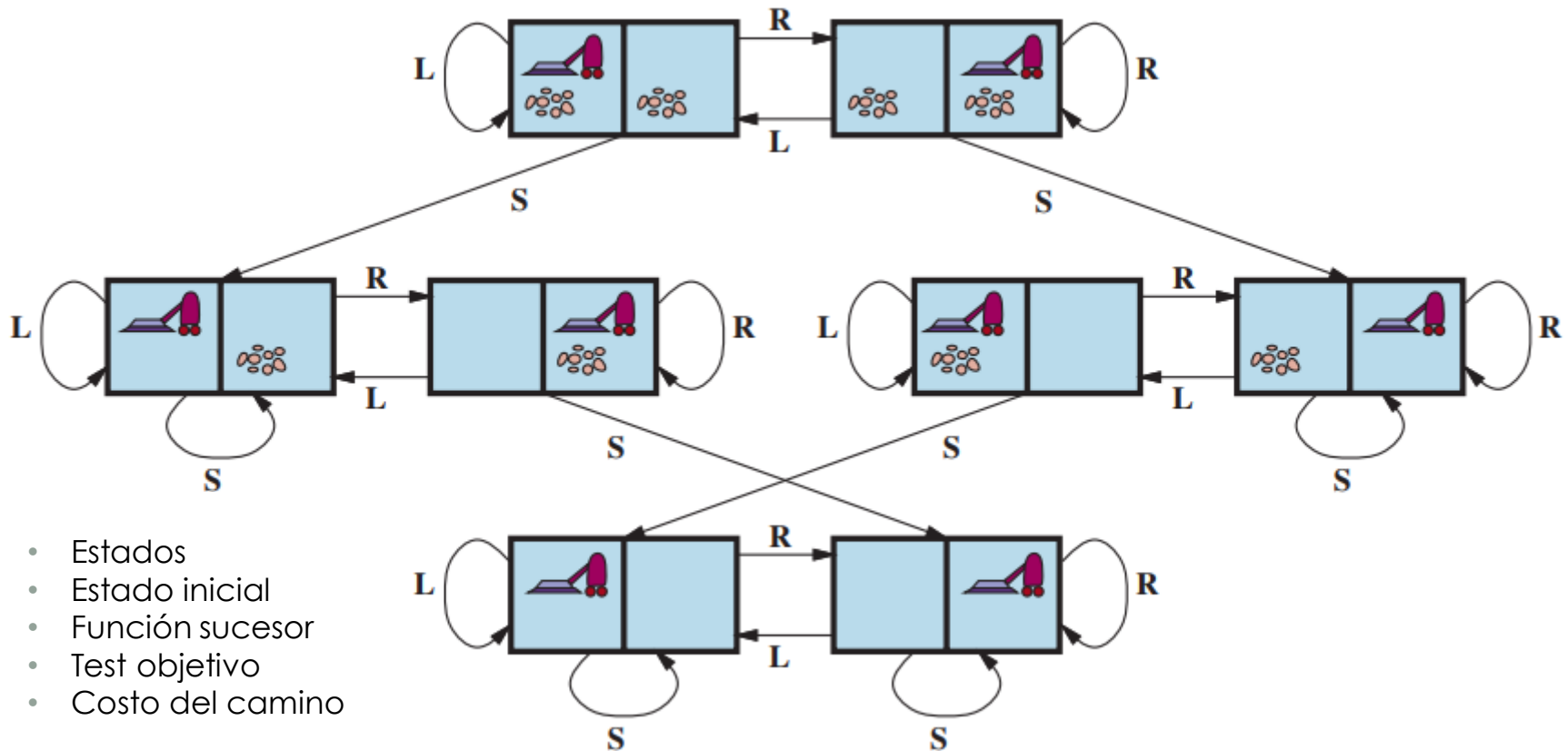


Un **problema del mundo-real** tiende a no tener una sola descripción. En general no se puede describir en forma exacta.

ESPACIO DE ESTADOS

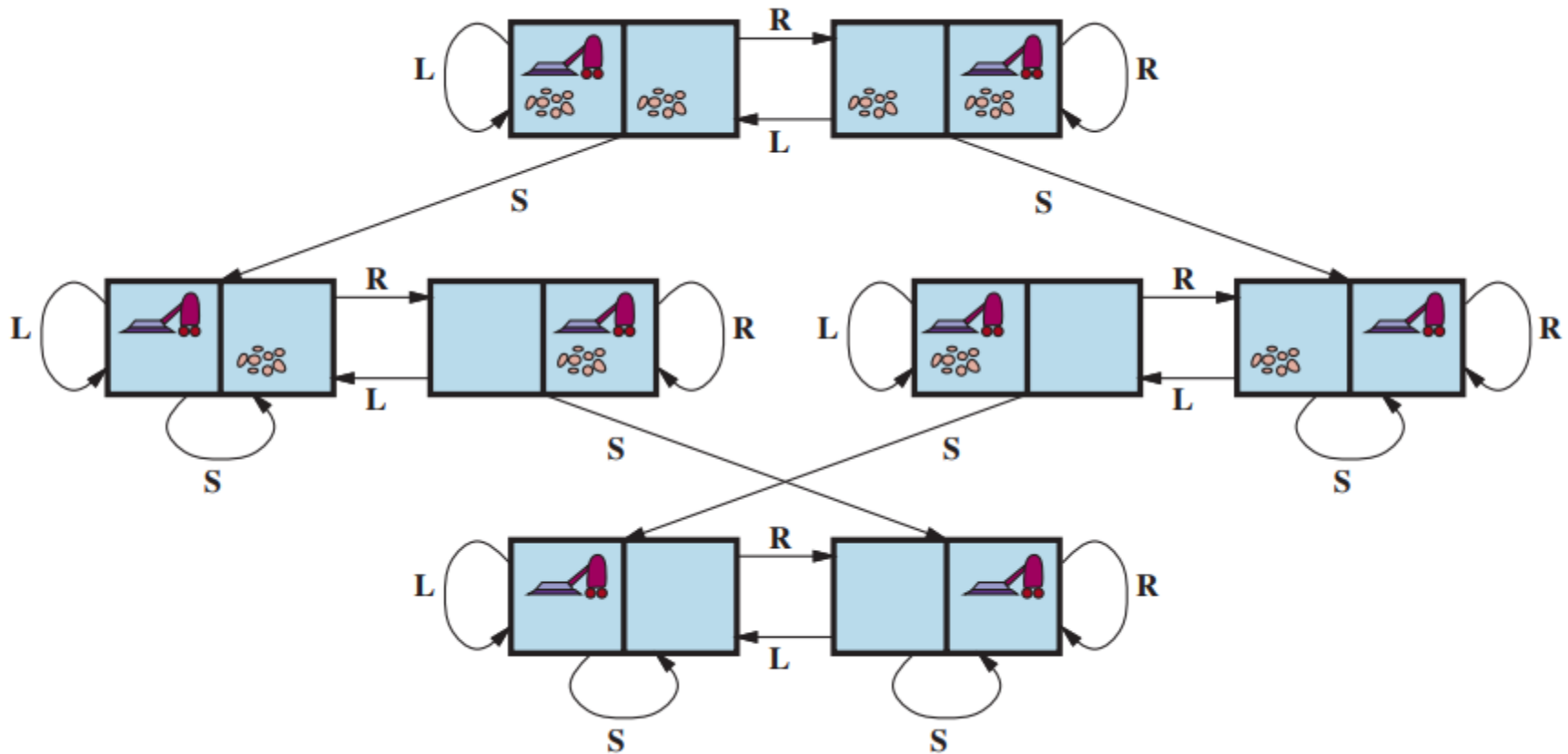
EL ESTADO INICIAL Y LA FUNCIÓN SUCESOR DEFINEN EL ESPACIO DE ESTADOS

mundo de la aspiradora



ESPACIO DE ESTADOS

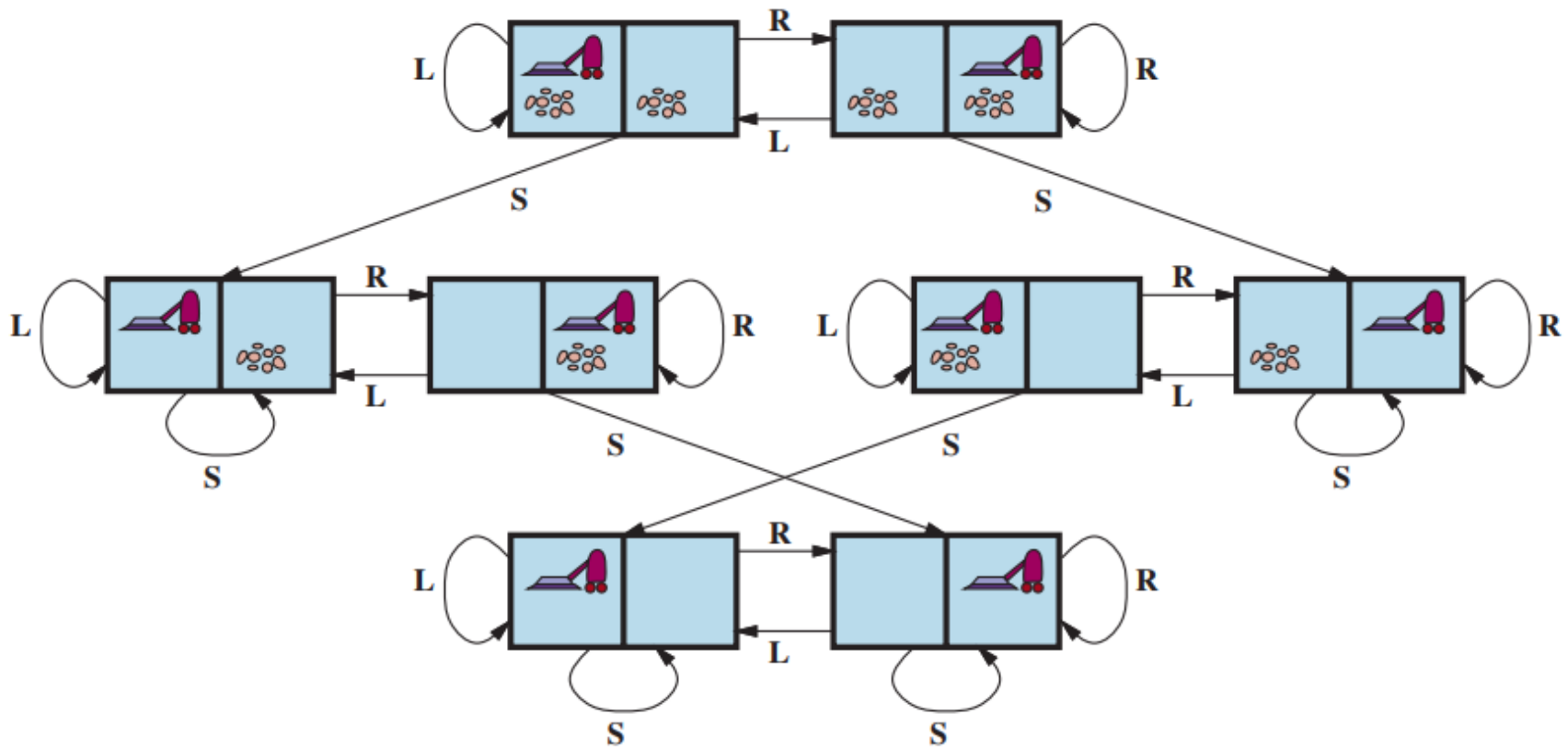
mundo de la aspiradora



Estados: el agente está en una de sus dos localizaciones, cada una de las cuales puede o no contener suciedad. Así, hay $2 \times 2^2 = 8$ estados posibles.

ESPACIO DE ESTADOS

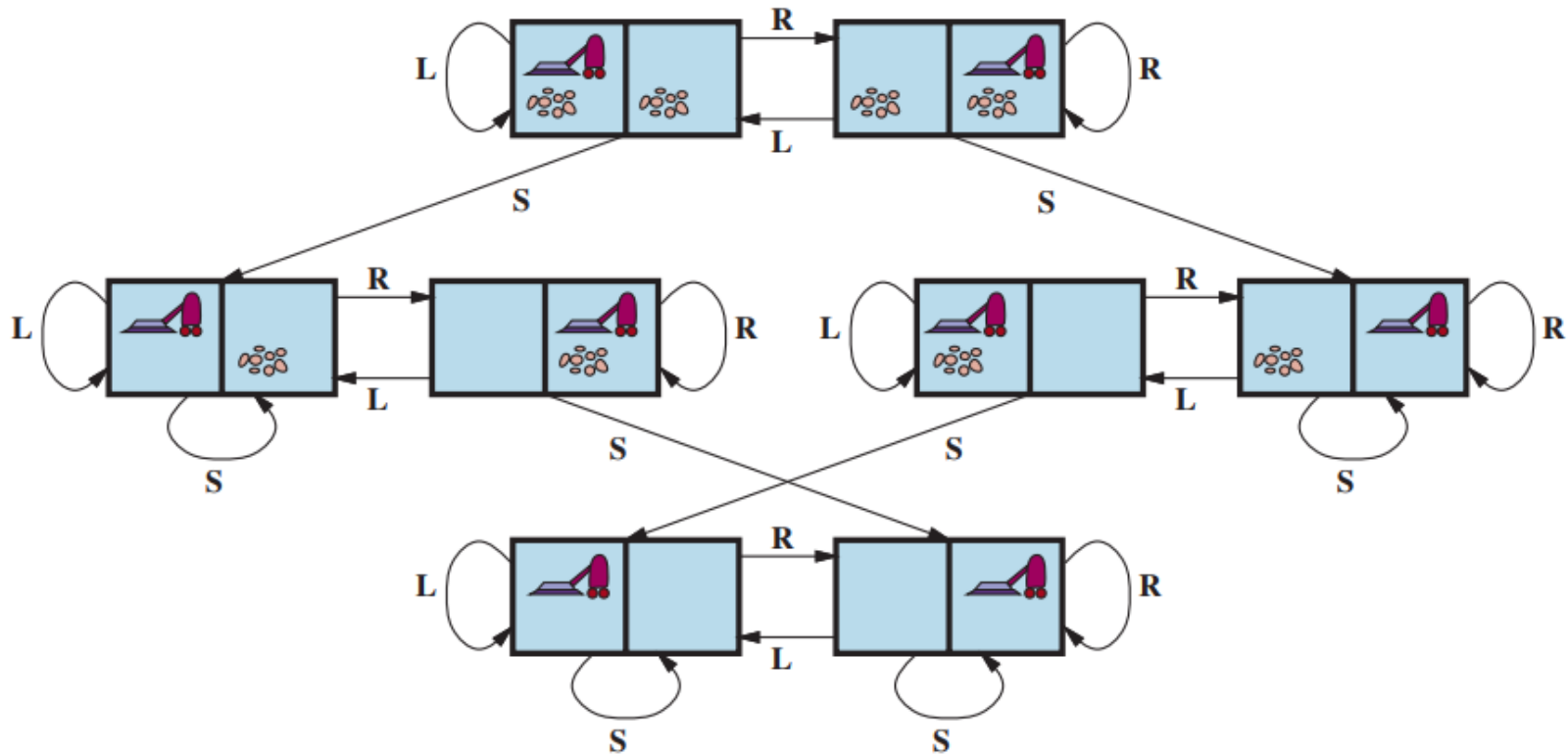
mundo de la aspiradora



Estado inicial: cualquier estado puede designarse como estado inicial

ESPACIO DE ESTADOS

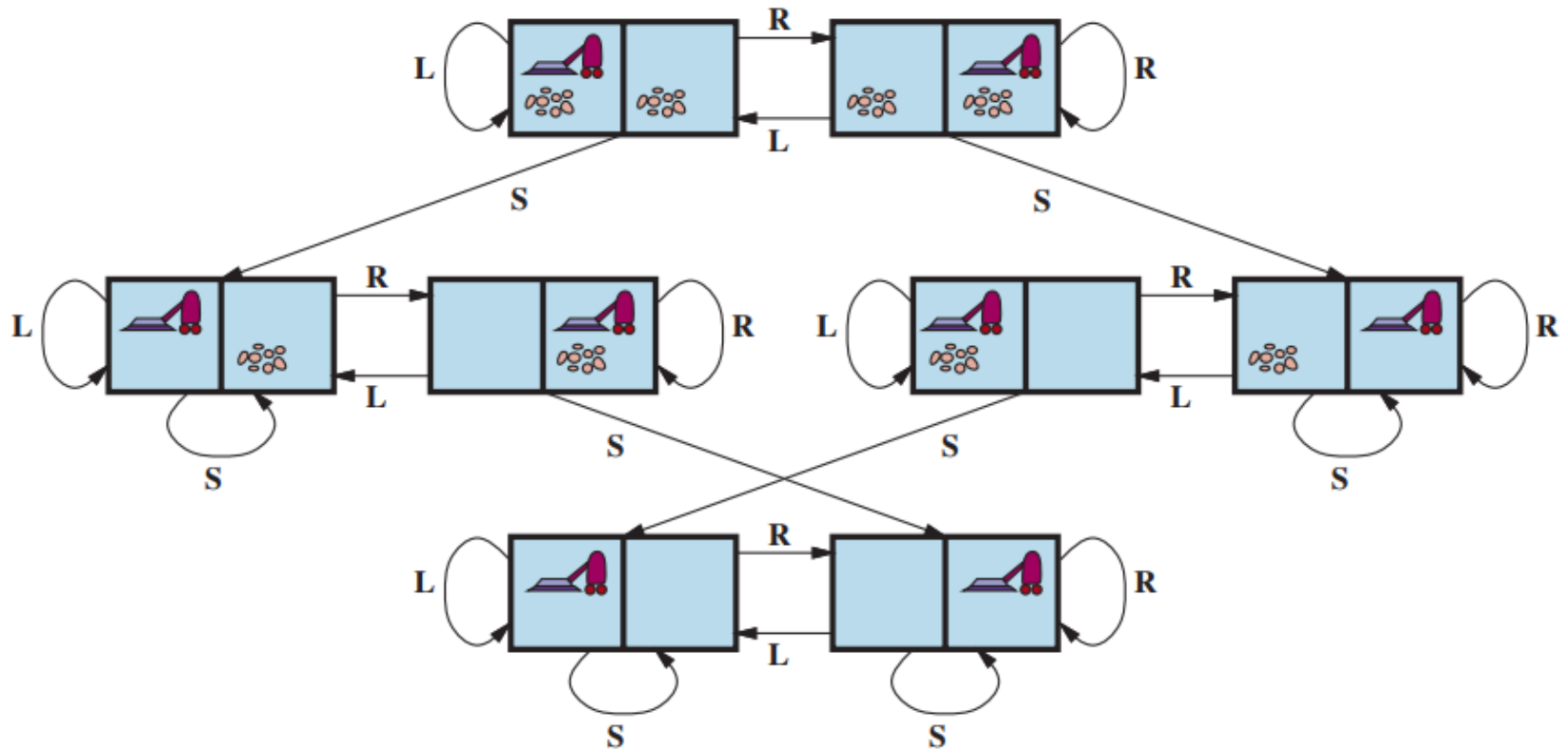
mundo de la aspiradora



Función sucesor: ésta genera los estados legales que resultan al intentar las tres acciones (*Izquierda, Derecha y Aspirar*)

ESPACIO DE ESTADOS

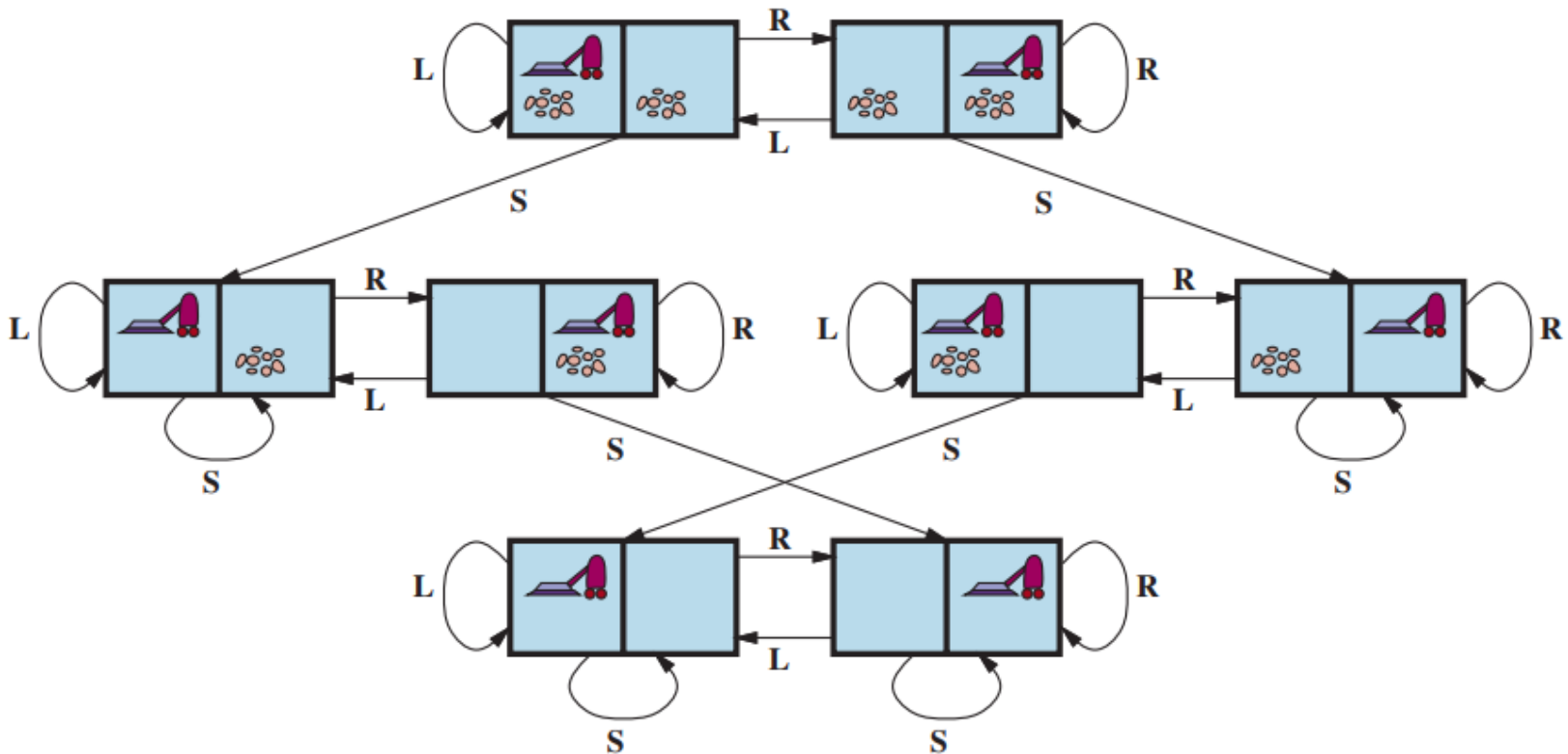
mundo de la aspiradora



Test objetivo: comprueba si todos los cuadrados están limpios.

ESPACIO DE ESTADOS

mundo de la aspiradora



Costo del camino: si cada costo individual es 1, el costo del camino es igual al número de pasos que lo componen



8-PUZLE

7	2	4
5		6
8	3	1

Estado Inicial

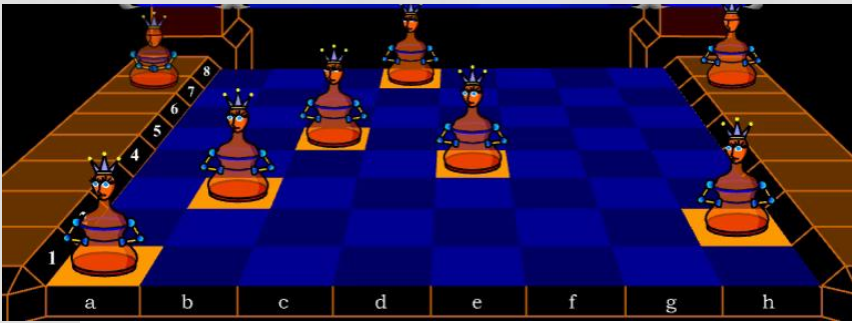
	1	2
3	4	5
6	7	8

Estado Objetivo

Formulación del problema

- Estados:** localización de cada ficha y el espacio
- Estado inicial:** cualquier estado puede ser inicial
- **Func. Sucesor:** mover el espacio a la *Izquierda, Derecha, Arriba* o *Abajo*.
- Test objetivo:** comprueba si el estado coincide con la configuración objetivo de la figura.

Costo del camino: N° de pasos



8-REINAS

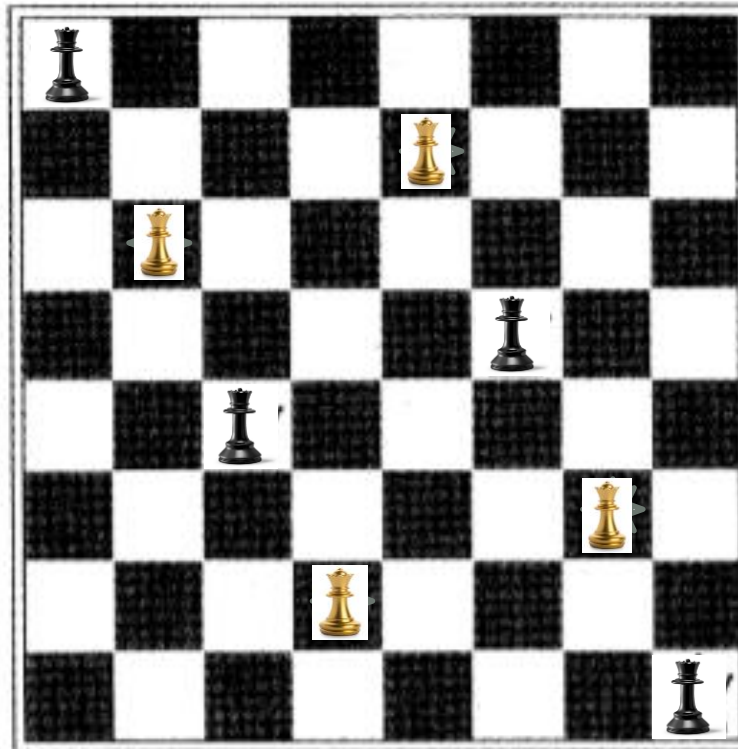


Figura 3.5 Casi una solución del problema de las 8-reinas. (La solución se deja como ejercicio.)

PROBLEMAS DEL MUNDO REAL

BÚSQUEDA DE UNA RUTA

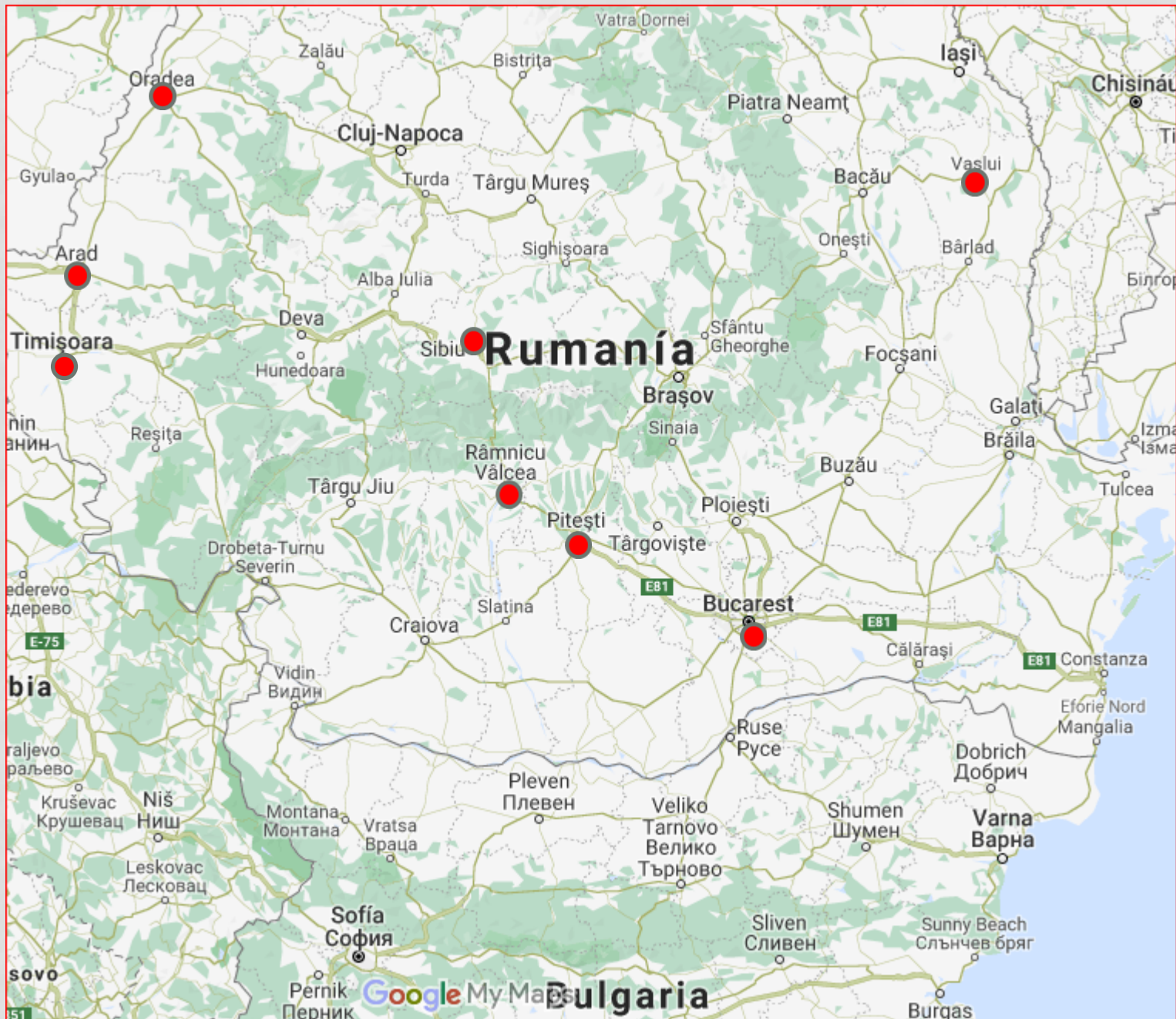


- Problemas turísticos
- Viajes de líneas aéreas
- El problema del viajante de comercio
- Planificación de los movimientos de un taladro para fabricar circuitos impresos.
- Distribución VLSI (Very Large Scale Integration)
- Navegación de un robot
- Secuencia para ensamblaje automático
- Robot software para búsquedas en internet

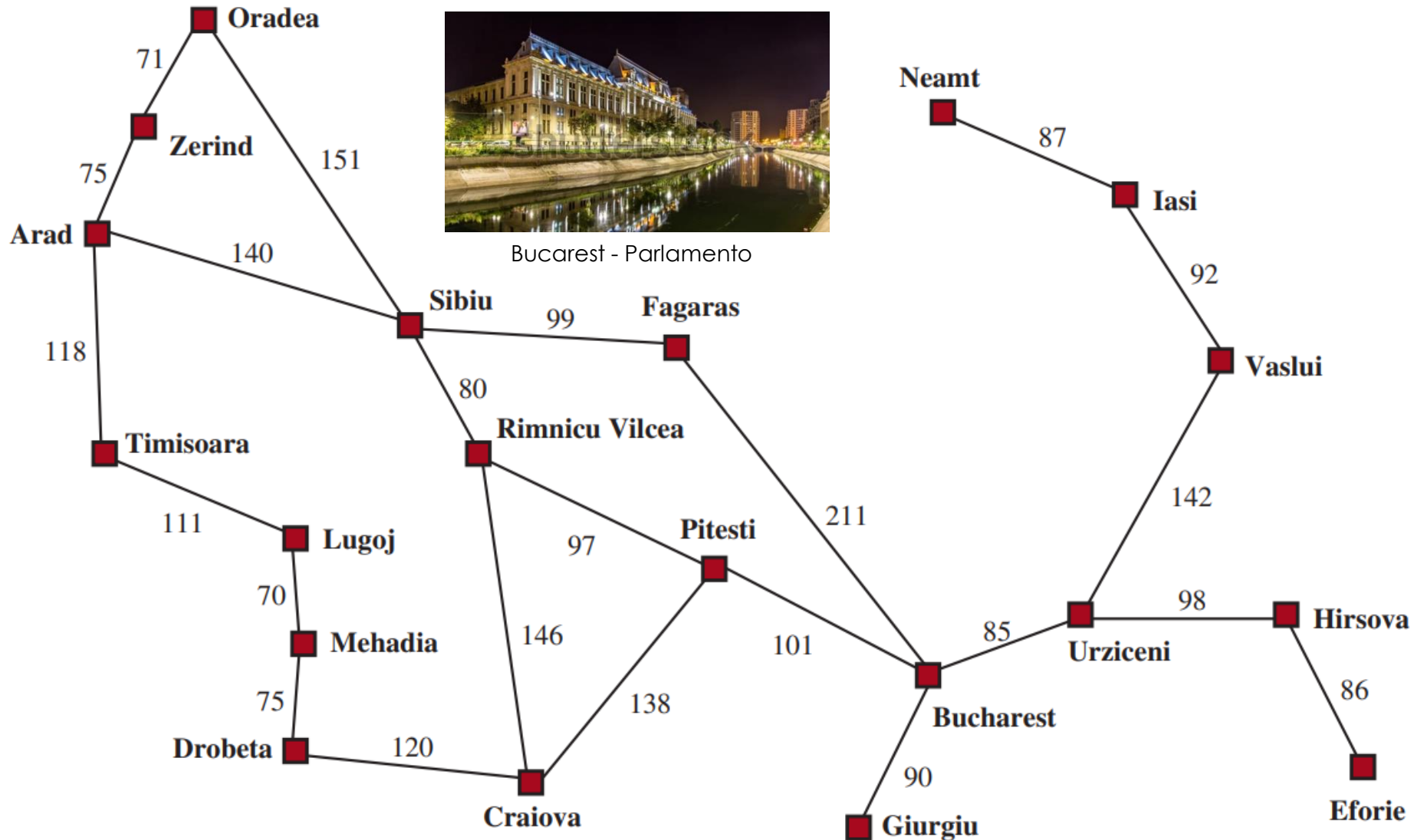


FORMULACIÓN VIAJE DE LÍNEAS AÉREAS

- **Estados:** una localización (aeropuerto) y la hora actual.
- **Estado inicial:** especificado por el problema (un aeropuerto específico a una hora determinada)
- **Función sucesor:** devuelve los estados que resultan de tomar cualquier vuelo programado desde el aeropuerto actual a otro, que salgan a la hora actual más el tiempo de tránsito del aeropuerto.
- **Test objetivo:** ¿Está disponible nuestro destino para una cierta hora especificada?
- **Costo del camino:** costo en dinero, tiempo de espera, tiempo de vuelo, costumbres y procedimientos de la inmigración, calidad del asiento, hora, tipo de avión, kilometraje, etc.



BÚSQUEDA DE SOLUCIONES

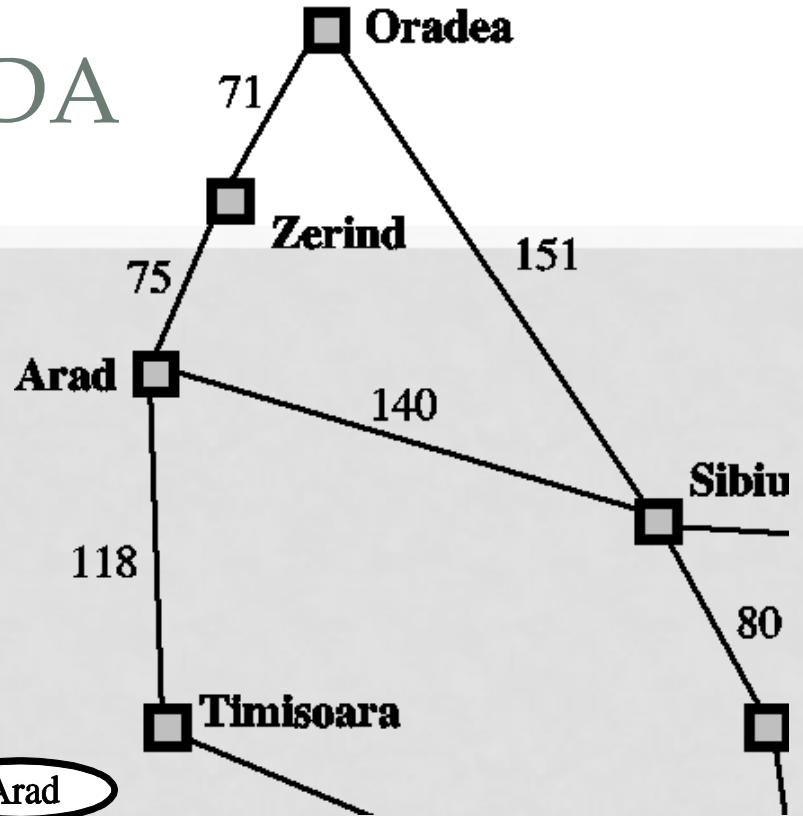


Mapa de carreteras simplificado de parte de Rumania

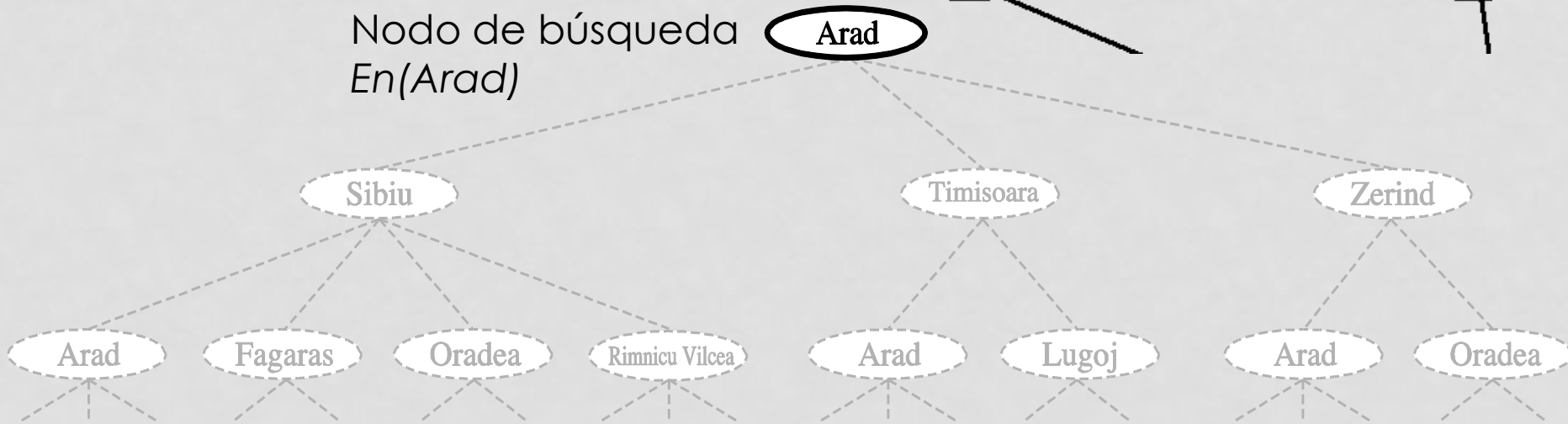
ÁRBOL DE BÚSQUEDA



Bucarest – Palacio de Cultura de Arad



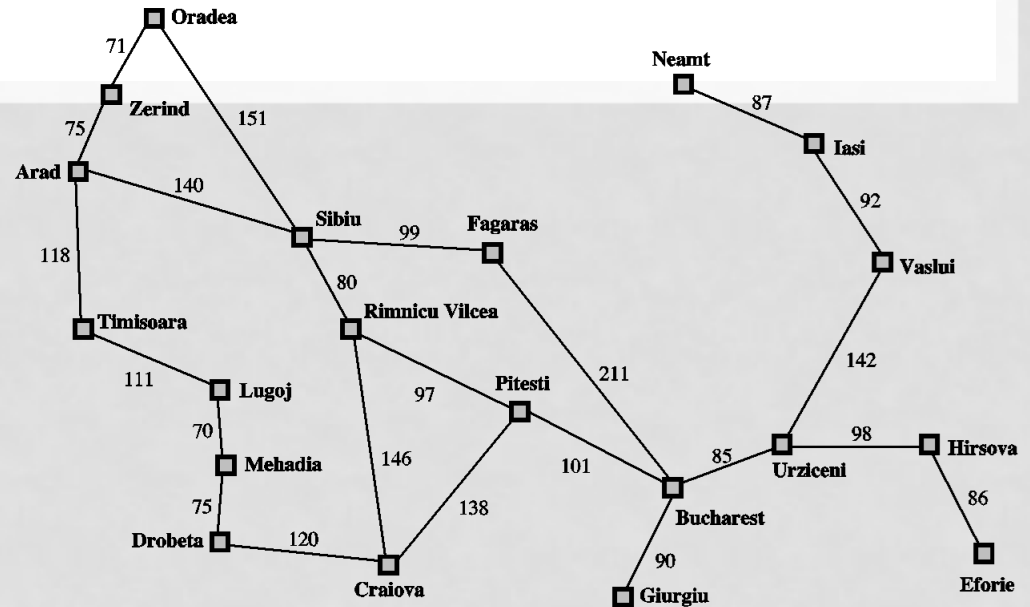
Nodo de búsqueda
En(Arad)



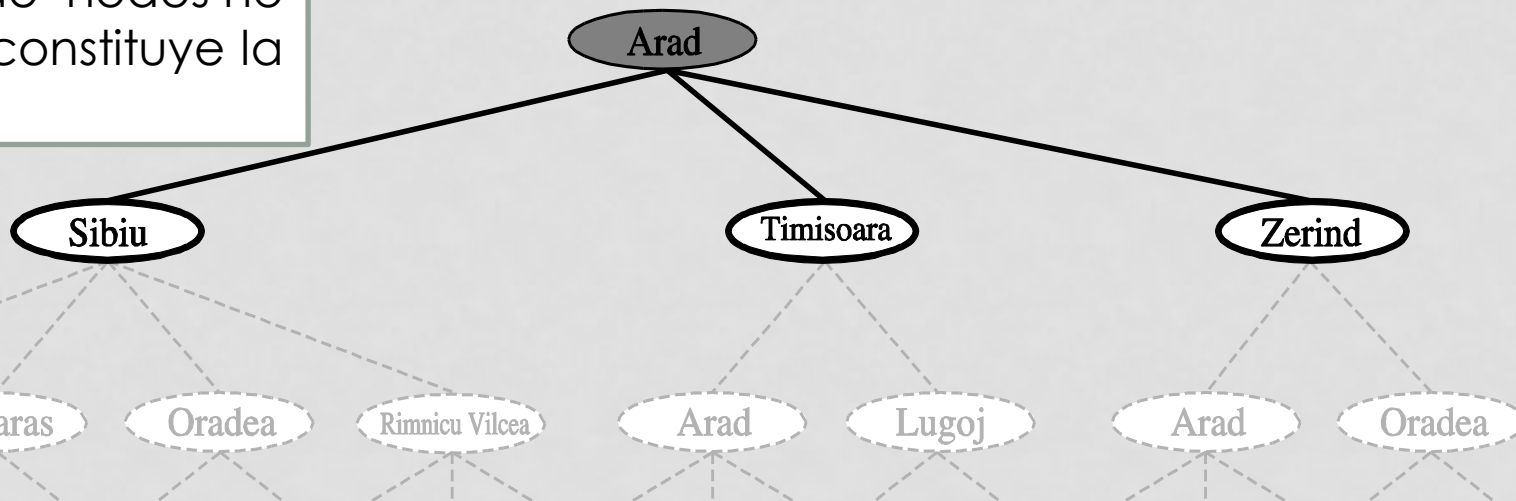
ÁRBOL DE BÚSQUEDA



Bucarest – Sibiu



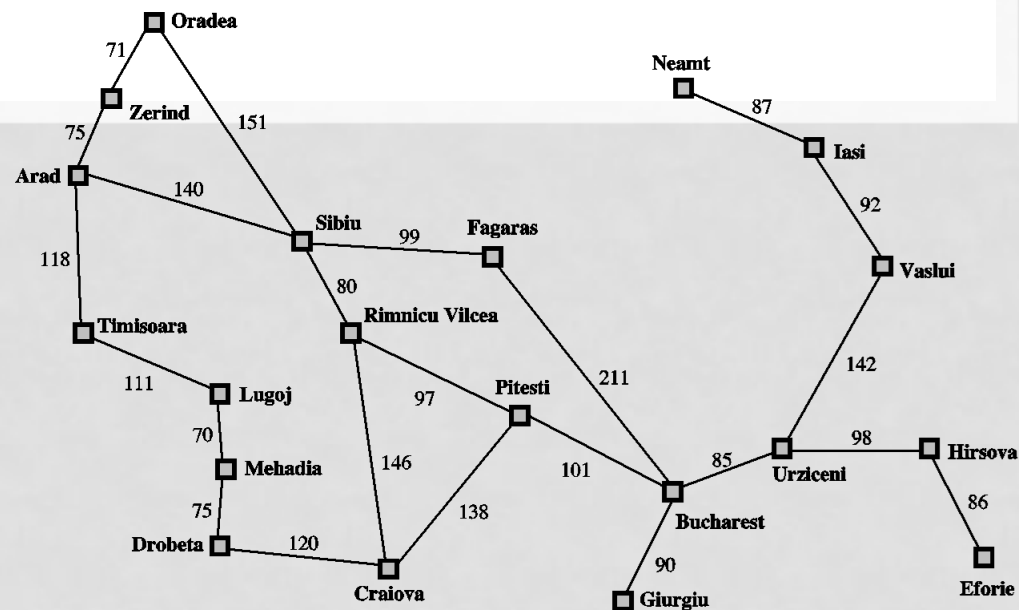
El conjunto de nodos no Expandidos constituye la **frontera**.



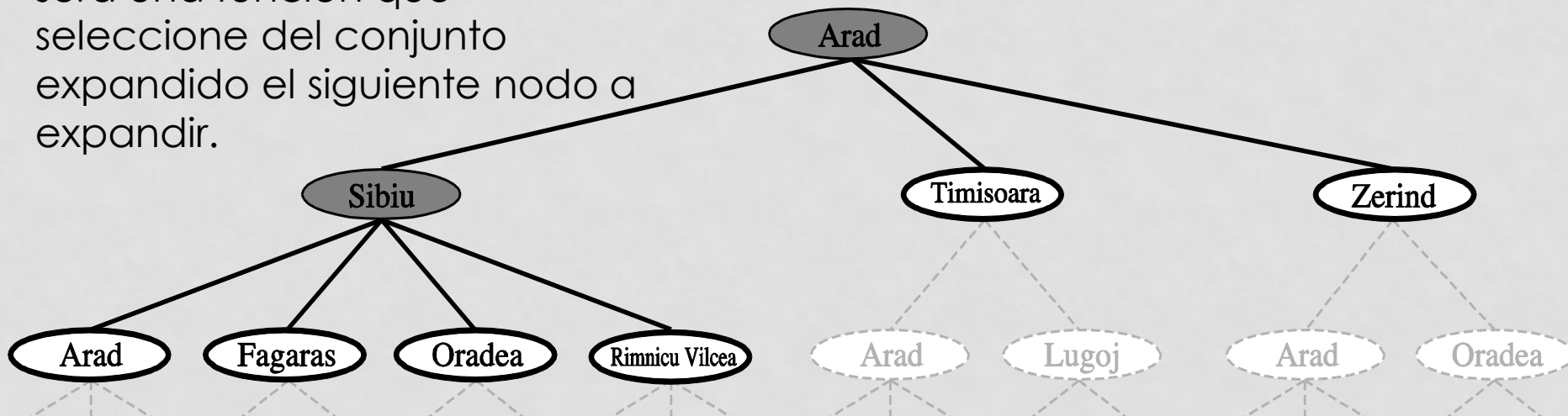
ÁRBOL DE BÚSQUEDA



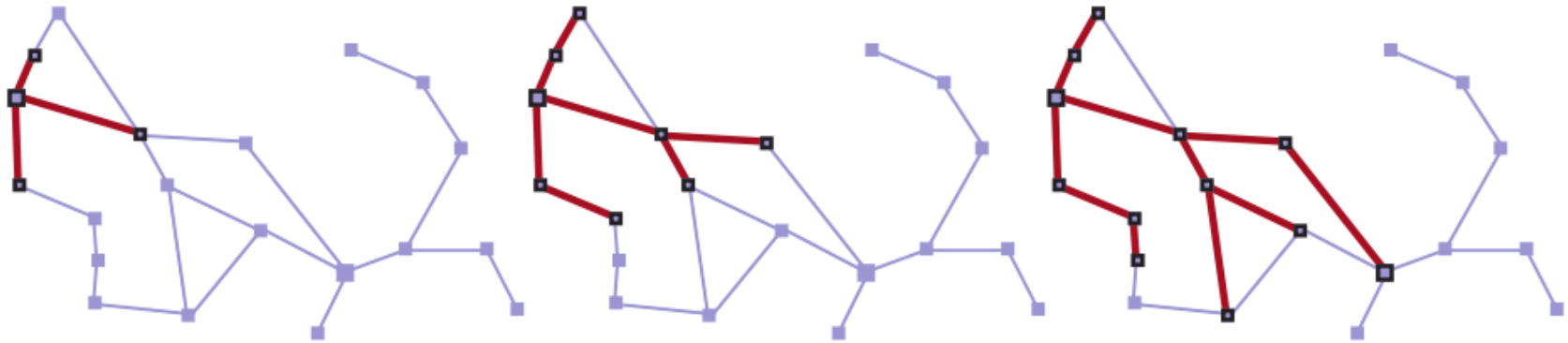
Bucarest – Timisoara



La **estrategia** de búsqueda será una función que seleccione del conjunto expandido el siguiente nodo a expandir.

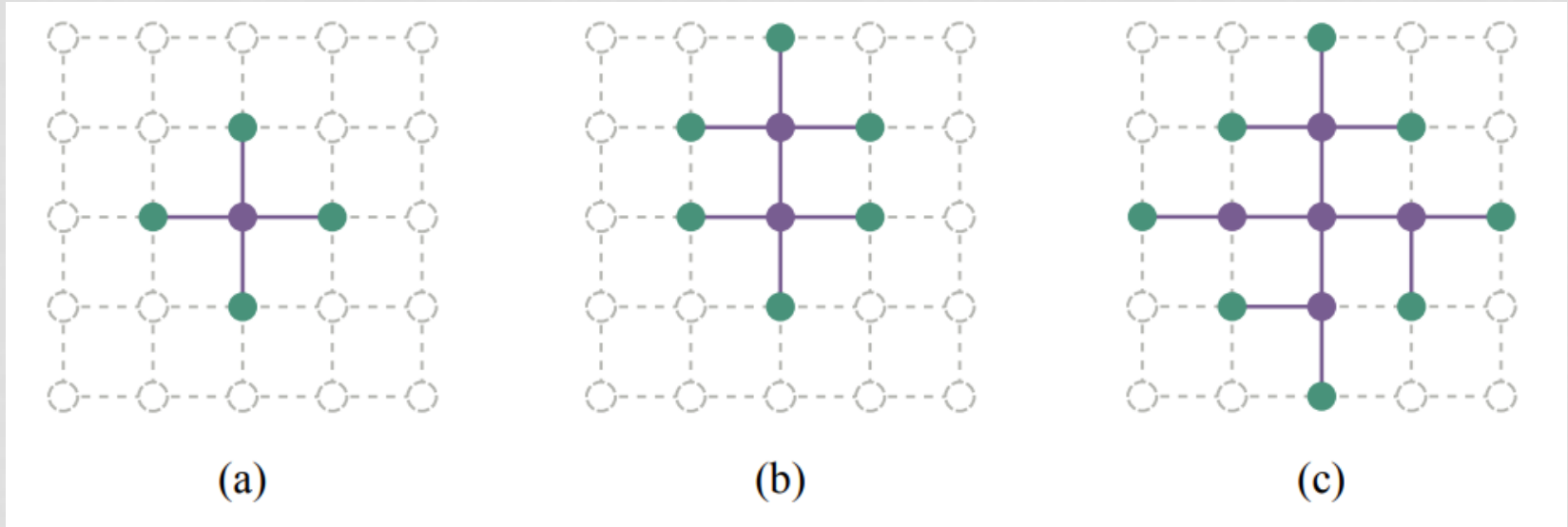


NO VOLVER POR DONDE YA ESTUVIMOS



En cada etapa, hemos expandido cada nodo en la frontera, extendiendo cada camino con todas las acciones aplicables que no dan como resultado un estado que ya se ha alcanzado. Observe que en la tercera etapa, la ciudad más alta (Oradea) tiene dos sucesores, ambos de los cuales ya han sido alcanzados por otros caminos, por lo que no se extienden caminos desde Oradea.

SEPARACIÓN DE LA BÚSQUEDA DE GRÁFICOS, ILUSTRADA EN UN PROBLEMA DE CUADRÍCULA RECTANGULAR



La **frontera (verde)** separa el **interior (lavanda)** del exterior (rayas tenues). La frontera es el conjunto de nodos (y estados correspondientes) que se han alcanzado pero aún no se han expandido; el interior es el conjunto de nodos (y estados correspondientes) que se han expandido; y el exterior es el conjunto de estados que no se han alcanzado. En (a), solo se ha expandido la raíz. En (b), se expande el nodo superior de la frontera. En (c), se expanden los sucesores restantes de la raíz en sentido horario.

RENDIMIENTO



Se evalúa de cuatro formas:

Complejidad: ¿está garantizado que el algoritmo encuentre una solución cuando esta exista?

Optimización: ¿encuentra la estrategia la solución óptima?

Complejidad en tiempo: ¿cuánto tarda en encontrar una solución?

Complejidad en espacio: ¿cuánta memoria se necesita para el funcionamiento de la búsqueda?

COMPLEJIDAD TEMPORAL Y ESPACIAL

Se miden en términos de:

b – máximo **factor de ramificación** del árbol de búsqueda (máximo n° de sucesores)

d - **profundidad** del nodo objetivo más superficial

m - **longitud** máxima de cualquier camino en el espacio de estados (puede ser ∞)



¿Por qué los astronautas no pueden mantener una relación amorosa? Porque siempre necesitan espacio.

COMPLEJIDAD TEMPORAL Y ESPACIAL

- **Tiempo:**
se mide en términos de **n° de nodos generados** durante la búsqueda.
- **Espacio:**
se mide en términos de máximo **n° de nodos que se almacena** en memoria.
- **Costo de la búsqueda** depende la complejidad en tiempo y espacio
- **Costo total** = costo búsqueda + costo del camino

ESTRATEGIAS DE BÚSQUEDA

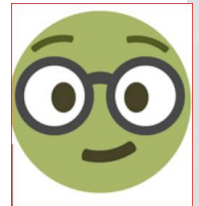
- **Búsqueda no informada o búsqueda a ciegas**
no cuentan con información adicional sobre estados que no son objetivos



- **Búsqueda con información parcial**
cuando el conocimiento de los estados o las acciones es incompleto (exploración)



- **Búsqueda informada o heurística**
cuenta con estrategias que le permiten saber si un estado no objetivo es más prometedor que otro



BÚSQUEDA NO INFORMADA

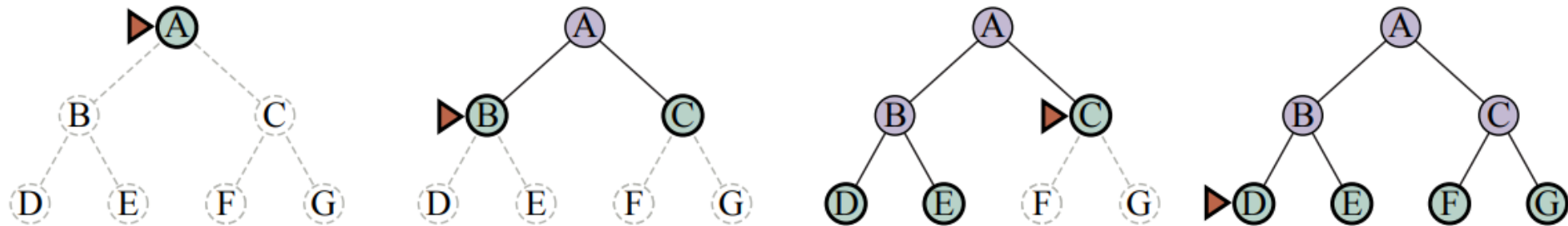
(BÚSQUEDA A CIEGAS)

- Primero en anchura
- Costo uniforme
- Primero en profundidad
- Profundidad limitada
- Primero en profundidad con profundidad iterativa



No tienen información adicional acerca de los estados más allá de la que proporciona la definición del problema. Todo lo que pueden hacer es generar los sucesores y distinguir entre un estado objetivo y uno que no lo es.

BÚSQUEDA PRIMERO EN ANCHURA

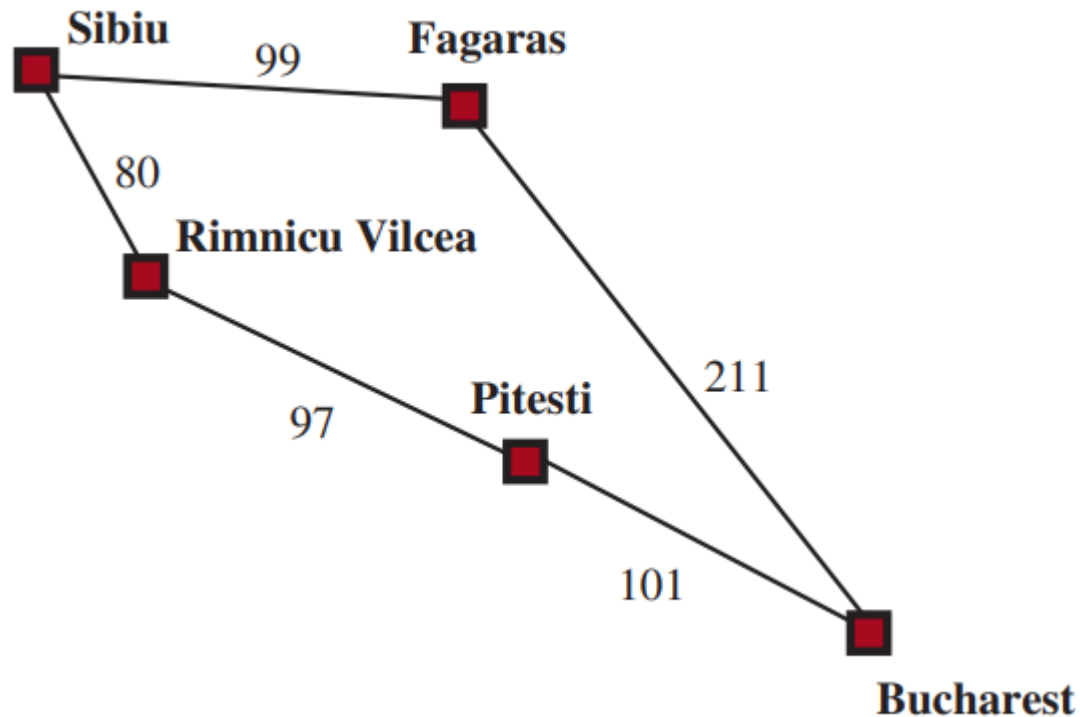


- Se expanden todos los nodos a una profundidad en el árbol antes de expandir cualquier nodo del próximo nivel. (FIFO)
- Completa
- El nodo objetivo más superficial no es necesariamente el óptimo. Es óptima cuando todos los costos son iguales.
- Complejidad $O(b^{d+1})$

BÚSQUEDA DE COSTO UNIFORME

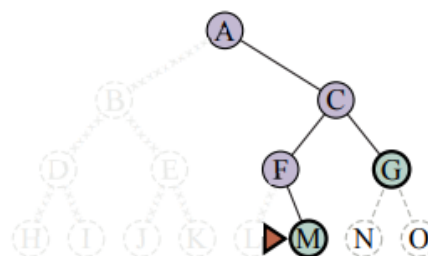
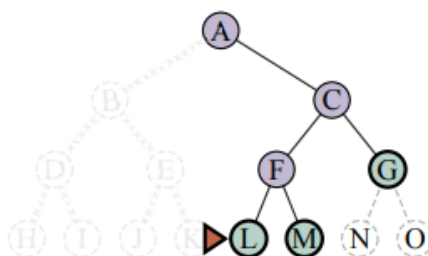
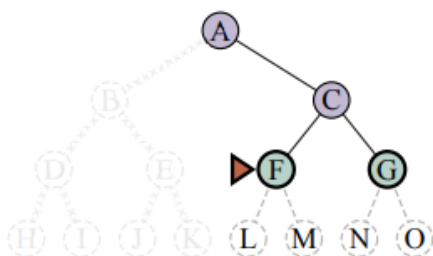
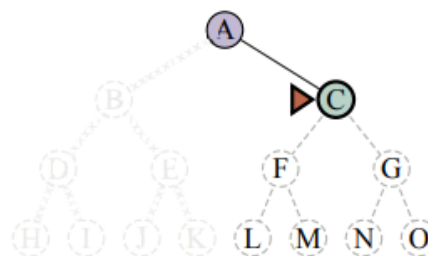
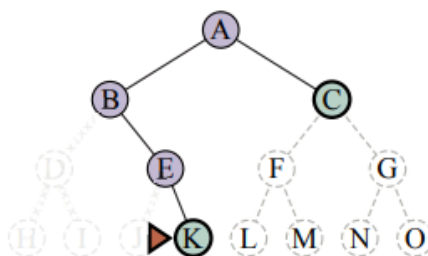
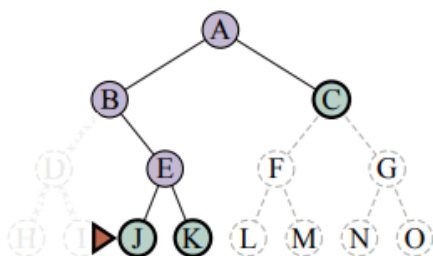
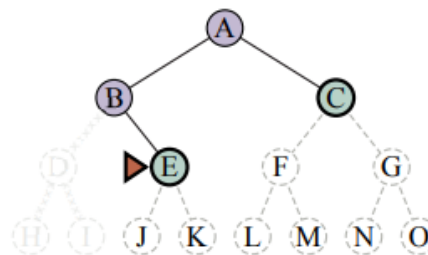
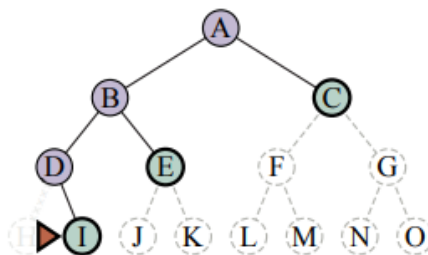
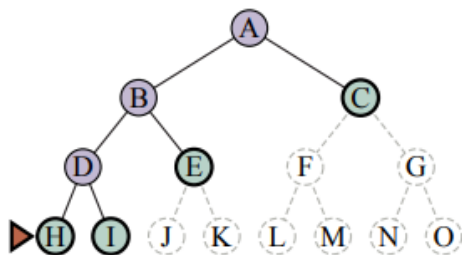
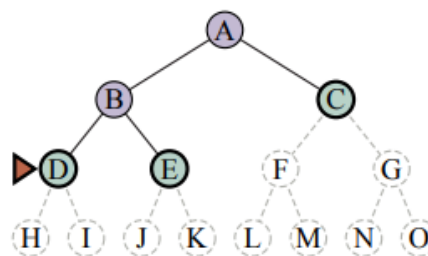
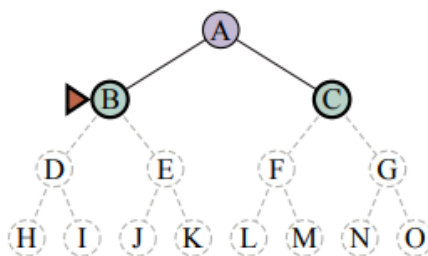
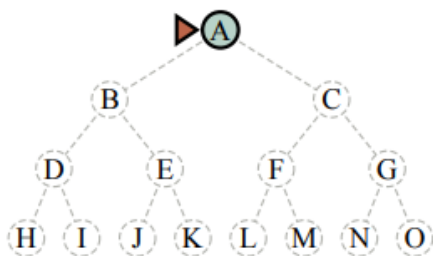
- Es óptima con cualquier función de costo.
- **Expande el nodo n con el camino de costo total más pequeño desde la raíz.**
- Si todos los costos son iguales es idéntica a la búsqueda primero en anchura.
- Bucle ∞ si expande un nodo que tiene una acción de costo cero que conduzca de nuevo al mismo estado.
- Se garantiza completitud y optimización si el costo de cada paso es mayor o igual a alguna constante positiva pequeña ϵ . **El costo de camino siempre aumenta.**
- El primer nodo objetivo seleccionado para la expansión es la solución óptima.
- Complejidad $O(b^{C^*/\epsilon})$; C^* es el costo de la solución óptima

BÚSQUEDA DE COSTO UNIFORME

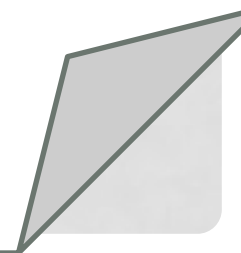


Los sucesores de Sibiu son Rimnicu Vilcea y Fagaras, con costos de 80 y 99, respectivamente. El nodo de menor costo, Rimnicu Vilcea, se expande a continuación, agregando Pitesti con un costo de $80 + 97 = 177$. El nodo de menor costo es ahora Fagaras, por lo que se expande, agregando Bucarest con un costo de $99 + 211 = 310$. Bucarest es el objetivo, pero el algoritmo solo prueba los objetivos cuando expande un nodo, no cuando genera un nodo, por lo que aún no ha detectado que este es un camino hacia el objetivo. El algoritmo continúa, eligiendo Pitesti para la expansión siguiente y agregando un segundo camino a Bucarest con un costo de $80 + 97 + 101 = 278$. Tiene un costo más bajo, por lo que reemplaza el camino anterior en alcanzado y se agrega a la frontera. Resulta que este nodo ahora tiene el costo más bajo, por lo que se considera siguiente, se encuentra como objetivo y se devuelve. Tenga en cuenta que si hubiéramos verificado un objetivo al generar un nodo en lugar de al expandir el nodo de menor costo, habríamos devuelto un camino de mayor costo (el que pasa por Fagaras).

BÚSQUEDA PRIMERO EN PROFUNDIDAD



Siempre expande el nodo más profundo en la frontera actual del árbol. (LIFO)



BÚSQUEDA PRIMERO EN PROFUNDIDAD

- No es completa. Falla en espacios de profundidad ∞
- Es completa en espacios finitos
- Es **compleja en tiempo** si $m > d$
 - m: profundidad máxima de cualquier nodo
 - d: profundidad de la solución más superficial
- La complejidad en espacio es lineal; $O(bm)$
- No es óptima
- Tiene requisitos muy modestos de memoria.

Esta estrategia debe evitarse cuando el espacio de búsqueda tiene una profundidad muy grande o incluso infinita.

BÚSQUEDA DE PROFUNDIDAD LIMITADA

- Para árboles ilimitados
- Se subdivide en búsquedas primero en profundidad con un límite de profundidad ℓ predeterminado.
- El límite de profundidad resuelve el problema de camino infinito.
- Si $\ell < d$ el objetivo está fuera del límite de profundidad.
- Si $d < \ell$ la búsqueda no será óptima.
- Complejidad en el espacio $O(b \ell)$
- Complejidad en tiempo $O(b^\ell)$

BÚSQUEDA DE PROFUNDIDAD LIMITADA

- Los límites de profundidad pueden estar basados en el conocimiento del problema. (ej: $\ell=19$ si hay 20 ciudades en total)
- Diámetro del espacio de estados da un mejor límite de profundidad. (ej: $\ell=9$ si en 9 pasos se alcanza cada ciudad).

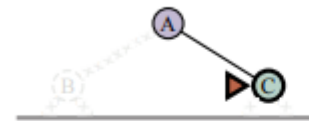
BÚSQUEDA PRIMERO EN PROFUNDIDAD CON PROFUNDIDAD ITERATIVA



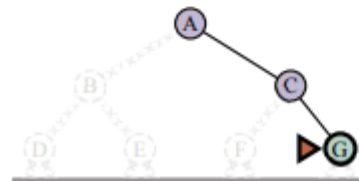
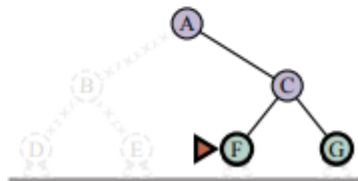
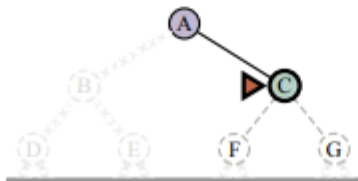
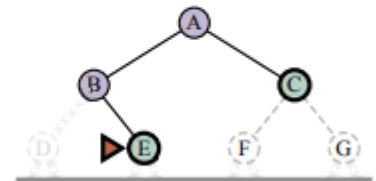
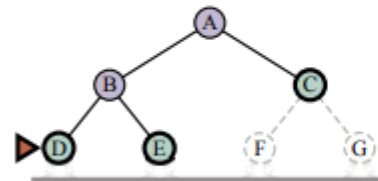
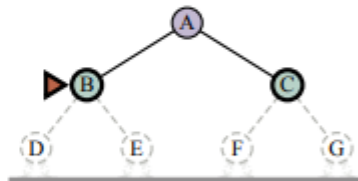
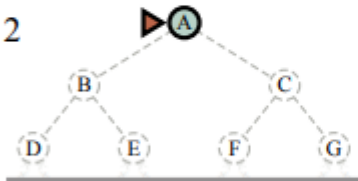
limit: 0

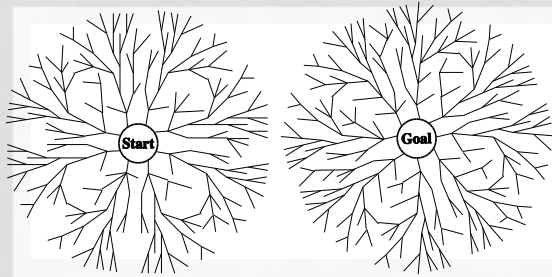


limit: 1



limit: 2





BÚSQUEDA BIDIRECCIONAL

Búsqueda simultánea que avanza desde el estado inicial y retrocede desde el objetivo.

Se implementa teniendo una o dos búsquedas que comprueban antes de ser expandido si cada nodo está en la frontera del otro árbol de búsqueda.

Implementación complicada!!!!

- definir sucesores y predecesores
- decidir qué tipo de búsqueda realizar en cada dirección
- contar con un procedimiento eficiente para comprobar la existencia de nodos en la otra dirección de la búsqueda

CÓMO EVITAR ESTADOS REPETIDOS

1- no generar un sucesor con un estado igual al padre

2- no generar un sucesor con un estado igual al de cualquiera de los ancestros

3- no generar un sucesor con un estado ya explorado (lista cerrada: almacena cada nodo expandido)

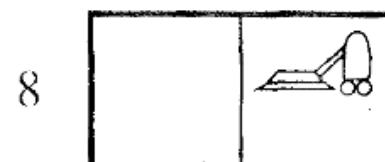
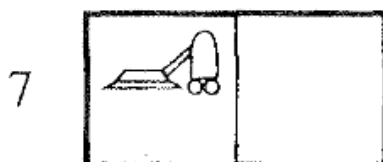
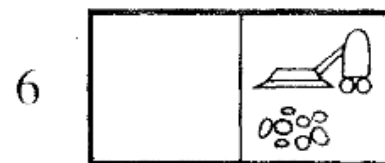
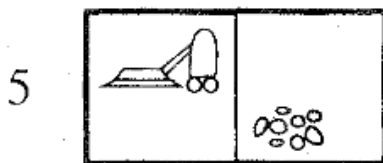
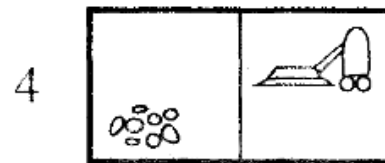
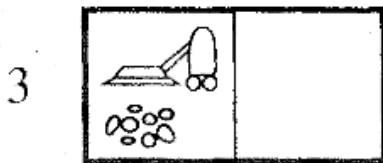
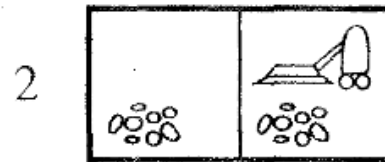
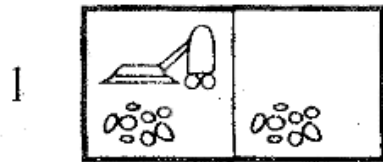
Los algoritmos que olvidan su historia están condenados a repetirla.

BÚSQUEDA CON INFORMACIÓN PARCIAL



- **Problemas sin sensores o conformados**
el agente podría estar en uno de los posibles estados iniciales posibles y cada acción puede conducir a uno de los posibles estados sucesores.
- **Problemas de contingencia**
el entorno es parcialmente observable o las acciones son inciertas. Si la incertidumbre está causada por las acciones de otro agente se llama entre **adversarios**.
- **Problemas de exploración**
se desconocen los estados y el agente debe actuar para descubrirlos.

PROBLEMAS SIN SENSORES



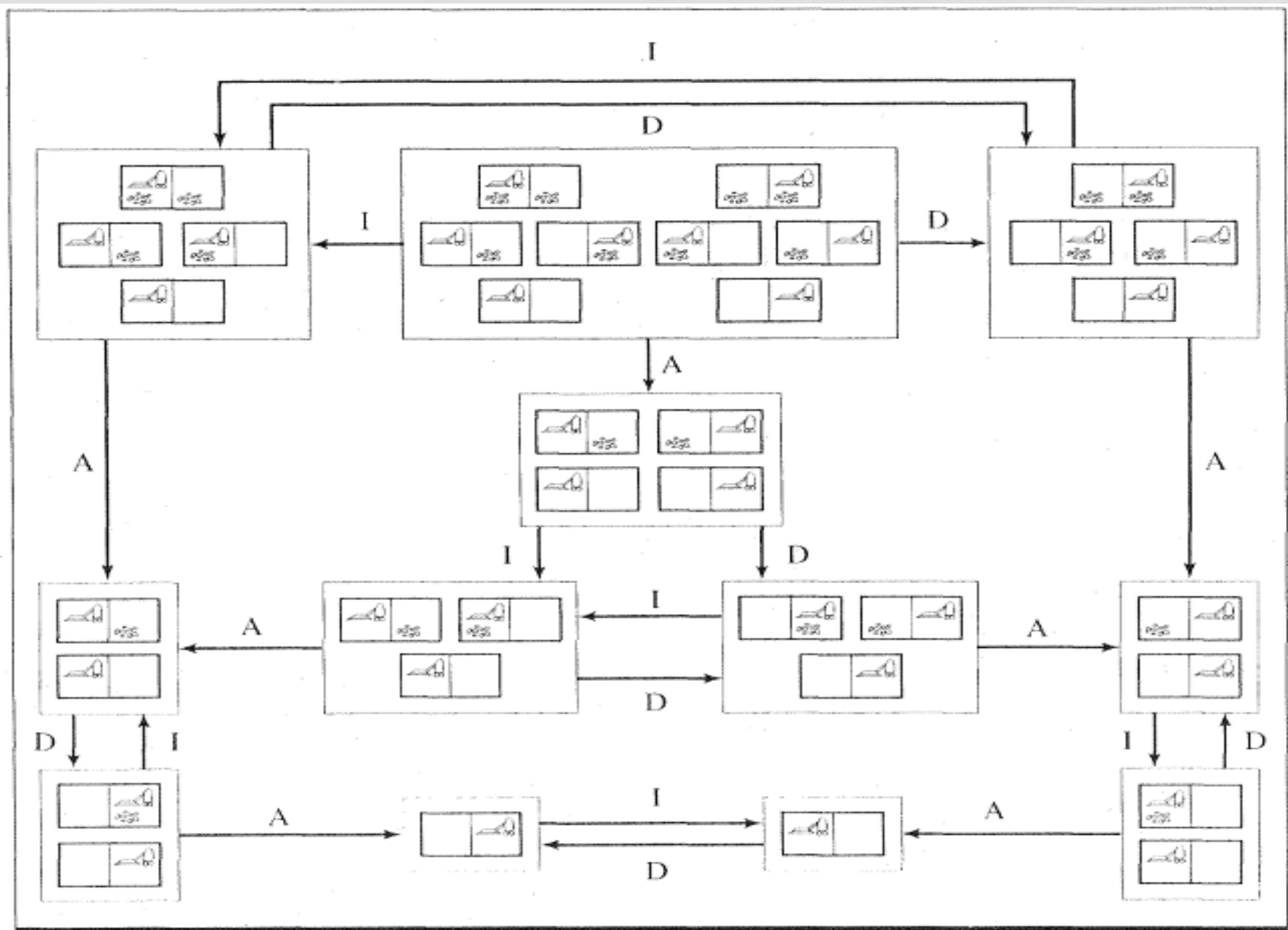
Estado inicial {1,2,3,4,5,6,7,8}

Derecha {2,4,6,8}

[Derecha, Aspirar] {4,8}

Secuencia [Der, Asp, Izq, Asp]
garantiza alcanzar el
estado 7

Estado de creencia: cada conjunto de estados posibles.



BÚSQUEDA INFORMADA

- Una estrategia de búsqueda informada puede encontrar soluciones de una manera más eficiente.
- Utiliza el conocimiento específico del problema más allá de la definición del problema en sí mismo.
- Estrategias:
 - búsqueda voraz primero el mejor
 - búsqueda A*



FUNCIÓN DE EVALUACIÓN Y FUNCIÓN HEURÍSTICA

- La selección de un nodo para su expansión se basa en una **función de evaluación $f(n)$** .
Tradicionalmente la evaluación **mide la distancia al objetivo**.



Función heurística $h(n)$: costo estimado del **camino más barato** desde el nodo n a un **nodo objetivo**.

BÚSQUEDA INFORMADA

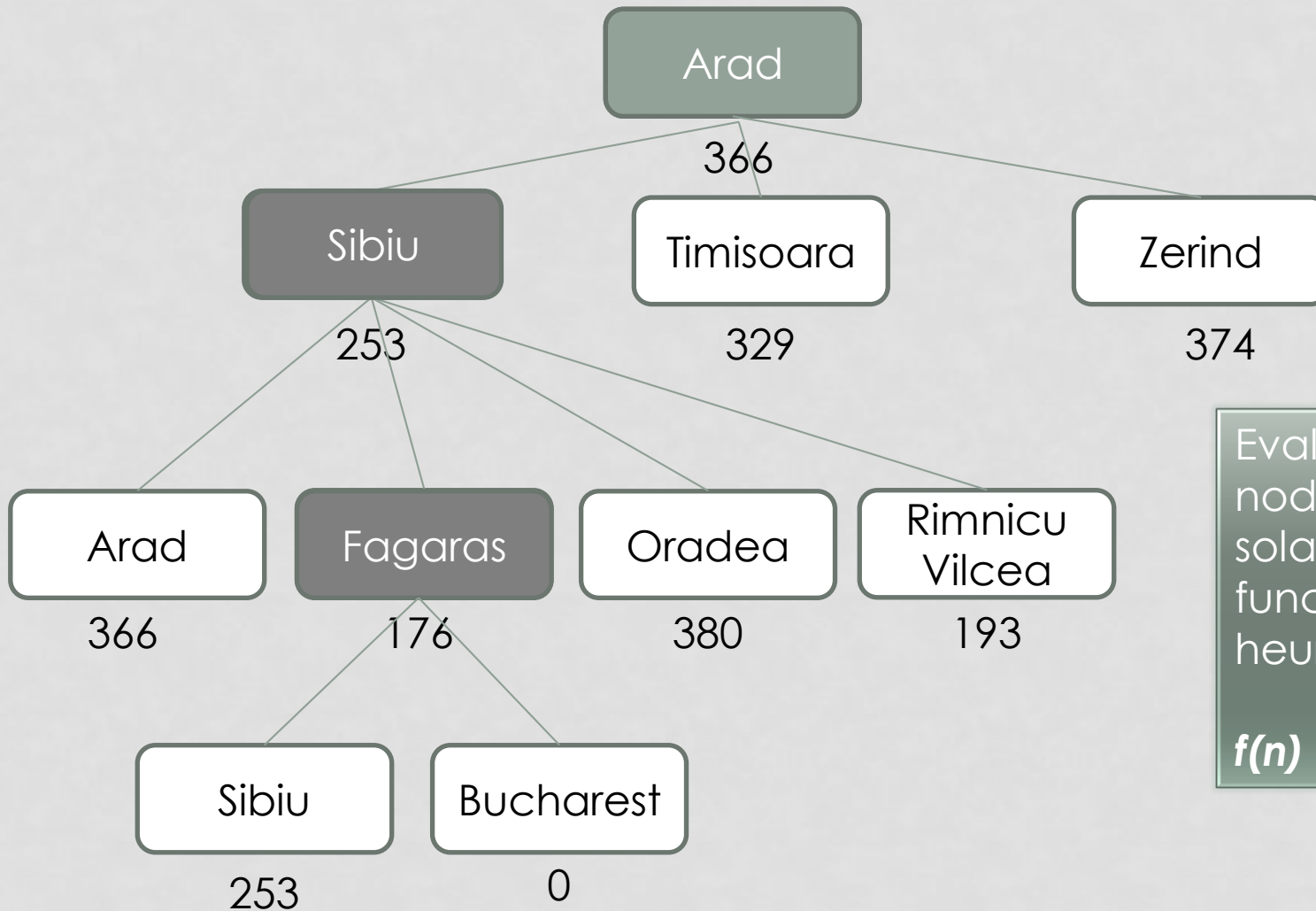
- función heurística $h(n)$

h_{DLR} : Distancia en Línea Recta hasta el **objetivo**

Ciudad	h_{DLR}	Ciudad	h_{DLR}
Arad	366	Mehadia	241
Bucarest	0	Neamt	234
Craiova	160	Oradea	380
Dobreta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgu	77	Timisoara	329
Hirsova	151	Urziceni	80
Lasi	226	Vaslui	199
Lugoj	244	Zerind	374

BÚSQUEDA VORAZ PRIMERO EL MEJOR

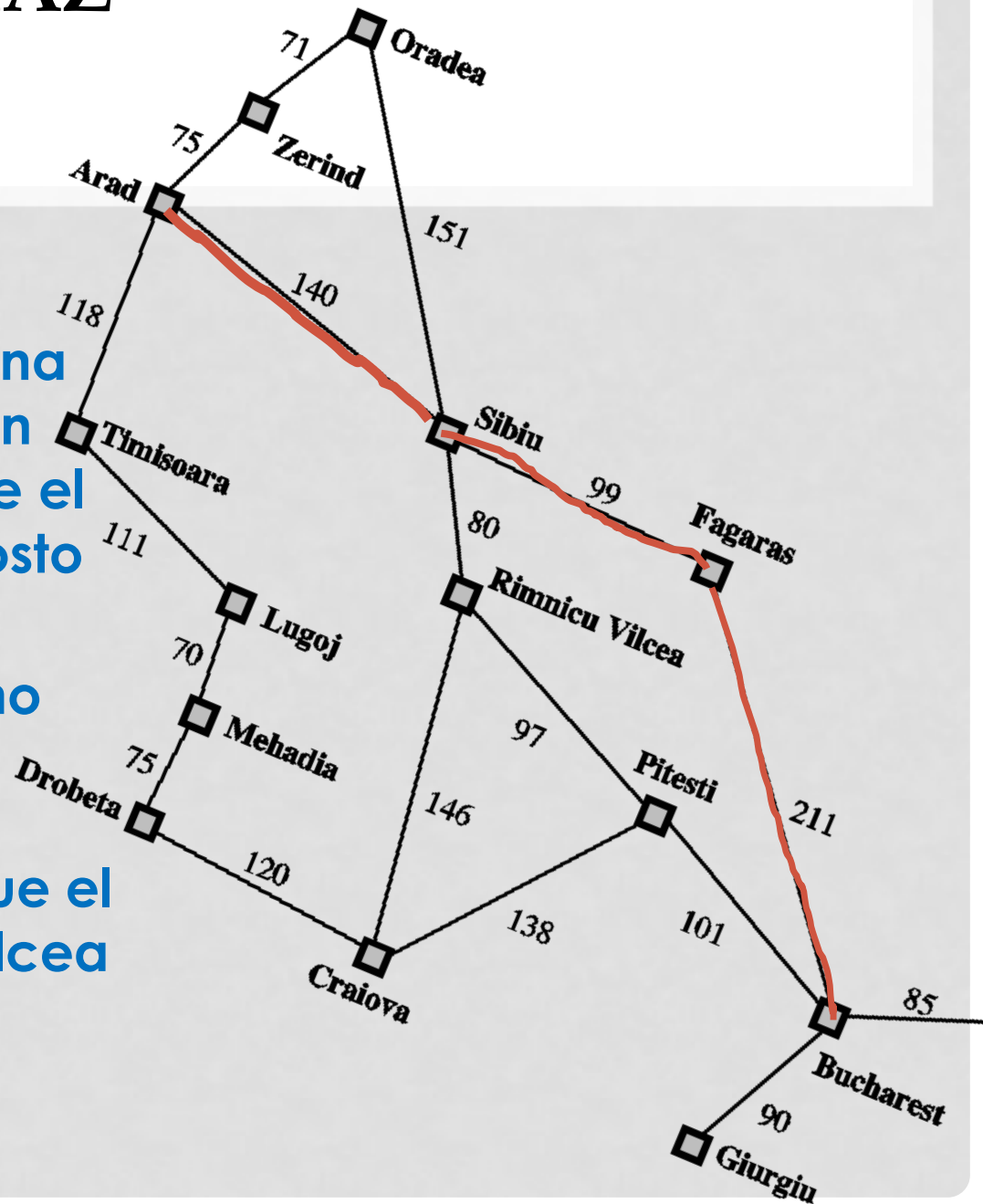
Trata de expandir el nodo más cercano al objetivo, alegando que probablemente conduzca rápidamente a una solución.



Evalúa los nodos utilizando solamente la función heurística:
 $f(n) = h(n)$

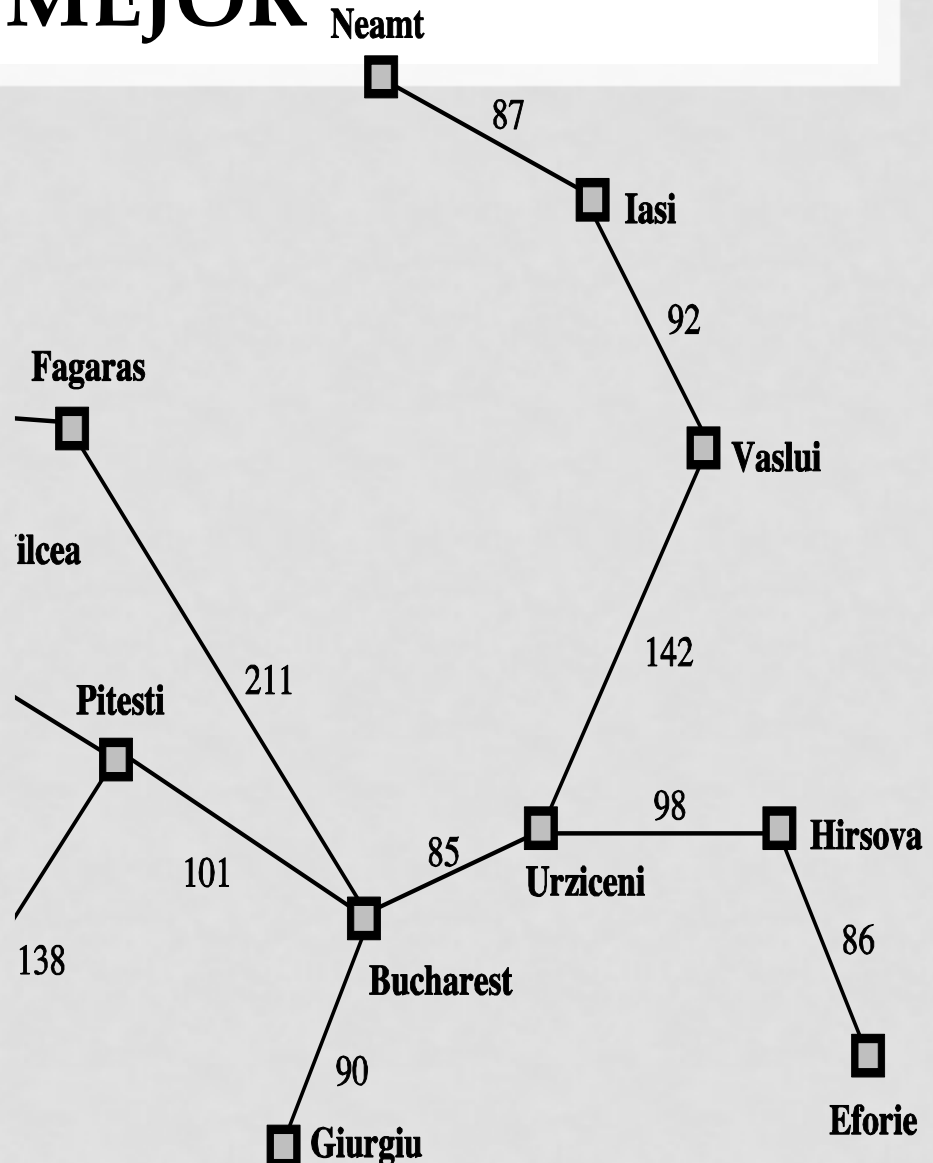
BÚSQUEDA VORAZ PRIMERO EL MEJOR

- Para este problema particular, encuentra una solución sin expandir un nodo que no esté sobre el camino solución. Su costo es mínimo.
- No es óptimo: el camino vía Sibiu y Faragas a Bucarest es 32 km más largo que el camino por Rimnicu Vilcea y Pitesti.



BÚSQUEDA VORAZ PRIMERO EL MEJOR

- Volverá atrás cuando llegue a un callejón sin salida
- La heurística puede expandir nodos innecesarios
- Es incompleta
- Puede ir hacia abajo en un camino infinito y nunca volver para intentar otras posibilidades
- La complejidad en tiempo y espacio es $O(b^m)$ con m : profundidad máxima del espacio de búsqueda.



EJEMPLO DE USO DE A*

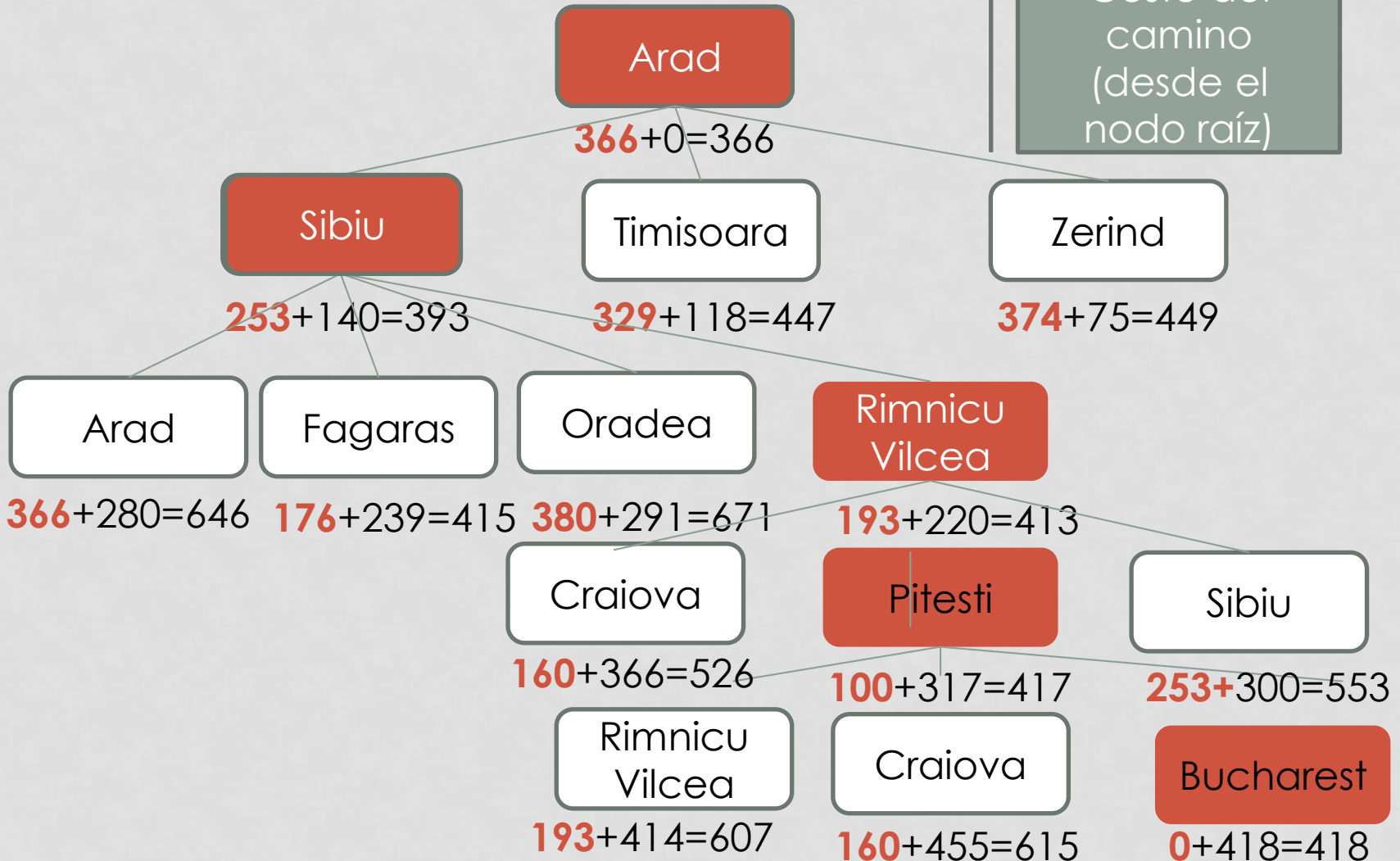


BÚSQUEDA A*

$$f(n) = h(n) + g(n)$$

Costo más barato de ir al nodo objetivo

Costo del camino (desde el nodo raíz)



BÚSQUEDA A^*

-El tiempo de cómputo es una gran desventaja de A^* .

Debido a que guarda en la memoria todos los nodos generados, por lo general A^* se queda sin memoria mucho antes de agotar el tiempo.

-No es práctico para grandes problemas.

-Es completa

-Es óptima

Si utiliza una heurística admisible (aquella que nunca sobre estima el costo de alcanzar el objetivo)