

Inteligencia Artificial

Resolución de Problemas Mediante Búsqueda:
Búsqueda no Informada

Temas fundamentales

1. Introducción
2. Agentes para la resolución de problemas
3. Formulación de problemas
4. Problemas ejemplo
5. Búsqueda de soluciones
6. Estrategias básicas de búsqueda
 - Búsqueda primero en anchura
 - Búsqueda de coste uniforme
 - Búsqueda primero en profundidad
 - Búsqueda limitada en profundidad
 - Búsqueda por profundización iterativa
 - (Búsqueda bidireccional)

Introducción

- Los agentes reflejos simples tienen muchas limitaciones
- Vamos a ver un tipo de agente basado en objetivos: agentes para resolución de problemas.
- Los agentes para resolución de problemas funcionan buscando secuencias de acciones que conduzcan a estados deseados.

Agentes para Resolución de Problemas

Los agentes inteligentes deben actuar de forma que se maximice su medida del desempeño => introducción de objetivos.

Los objetivos ayudan a dirigir el comportamiento del agente limitando las acciones que intenta realizar.

Las **etapas** de la resolución de problemas con objetivos:

1. Formulación de objetivos: a partir de la situación actual, definir los estados objetivo y los factores que pueden influir en el grado de satisfacción de las distintas maneras de conseguirlo.
2. Formulación del problema: decidir qué acciones y estados considerar.
3. Búsqueda: decidir qué hacer examinando diferentes secuencias de acciones que llevan a estados objetivo y escogiendo la mejor.
4. Ejecución: ejecutar las acciones recomendadas.

Sencillo agente resolvedor de problemas

función AGENTE-SIMPLE-RESOLVEDOR-PROBLEMAS(*percepción*) devuelve una acción

entradas: *percepción*: una percepción

estático: *sec*: una secuencia de acciones, vacía inicialmente

estado: una descripción del estado actual del mundo

objetivo: un objetivo inicialmente nulo

problema: una formulación del problema

estado \leftarrow ACTUALIZAR-ESTADO(*estado*, *percepción*)

si *sec* está vacía **entonces hacer**

objetivo \leftarrow FORMULAR OBJETIVO(*estado*)

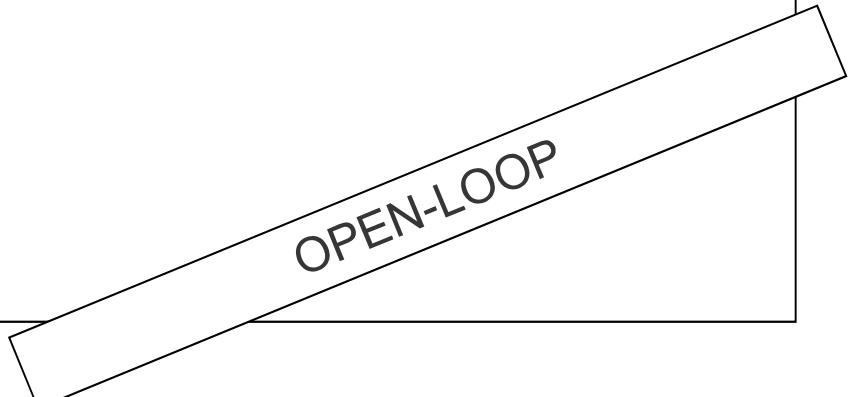
problema \leftarrow FORMULAR-PROBLEMA(*estado*, *objetivo*)

sec \leftarrow BÚSQUEDA(*problema*)

acción \leftarrow PRIMERO(*sec*)

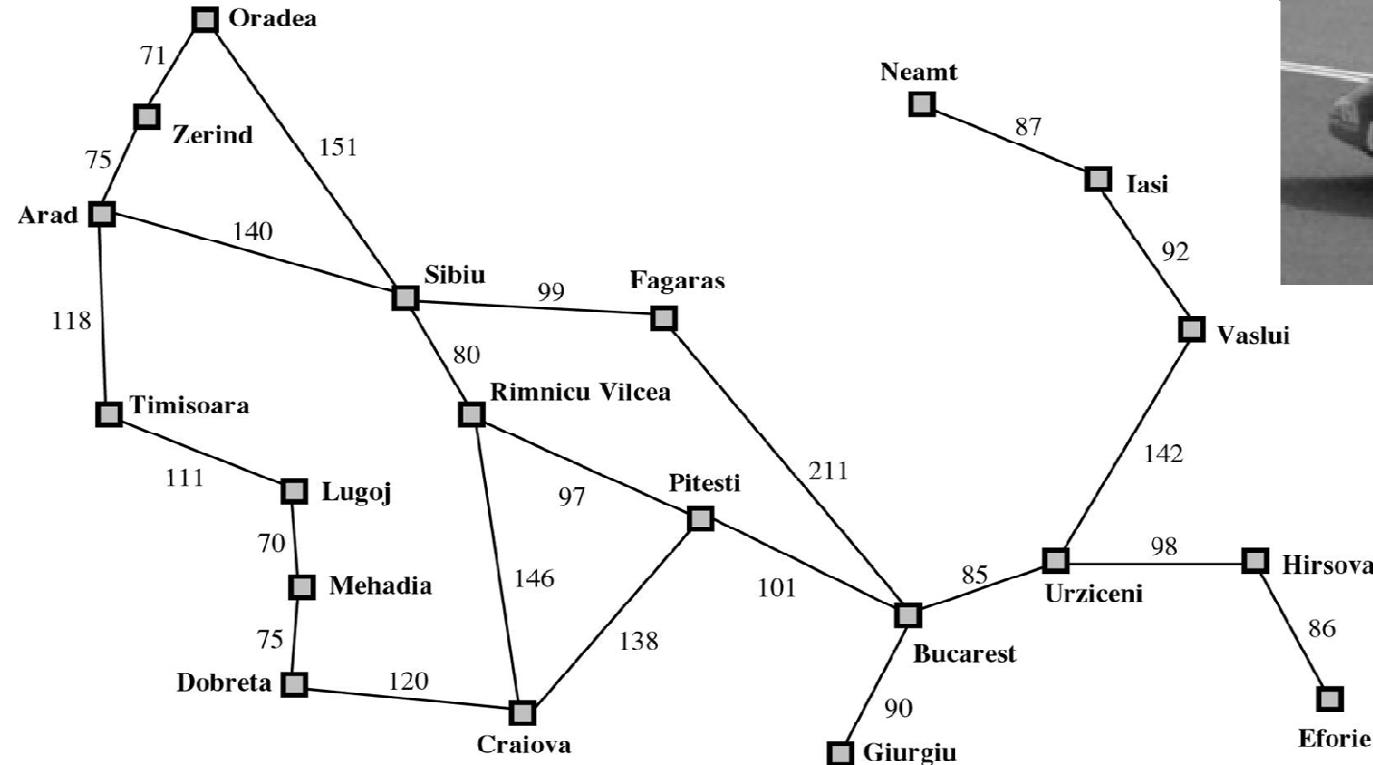
sec \leftarrow RESTO(*sec*)

devolver *acción*


 OPEN-LOOP

Ejemplo: Hallar rutas en Rumanía

Un agente en la ciudad de Arad (Rumanía), que dispone de un coche y un mapa de carreteras entre las principales ciudades de Rumanía. Mañana sale un vuelo a Bucharest.



Ejemplo: Hallar rutas en Rumanía



Un agente en la ciudad de Arad (Rumanía), que dispone de un coche y un mapa de carreteras entre las principales ciudades de Rumanía.
Mañana sale un vuelo a Bucharest.

Formulación del objetivo:

?

Formulación del problema:

estados: ?

acciones: ?

Búsqueda de la solución:

secuencia: ?

Ejecución:

?

Ejemplo: Hallar rutas en Rumanía



Un agente en Arad dispone de un coche y un mapa de carreteras para llegar a Bucharest.

Formulación del objetivo:

estar en Bucharest

Formulación del problema:

estados: cada estado es estar en una ciudad, #estados=#ciudades

acciones: conducir entre las ciudades siguiendo los tramos de carretera

Búsqueda de la solución:

secuencia de rutas entre ciudades, p.ej.:, ir de Arad a Sibiu, ir a Fagaras, ir a Bucharest.

Ejecución:

conducir por la secuencia de ciudades (i.e., las rutas entre ciudades)

Ejemplo: Hallar rutas en Rumanía



Un agente en Arad dispone de un coche y un mapa de carreteras para llegar a Bucharest

Formulación formal del problema:

Ciudades = {Oradea, Zerind, Arad, Sibiu, Timisoara, Lugoj, Mehaia, Drobeta, Craiova, Rimicu Vilcea, Fagaras, Pitesti, Bucharest, Giurgiu, Irrziceni, Hirsova, Eforie, Vaslui, Iasi, Neamt}

Estados: cada estado es estar en una ciudad,

Estados = { En(x) | x ∈ Ciudades}

Estado inicial = En(Arad)

Estado objetivo = En(Bucharest)

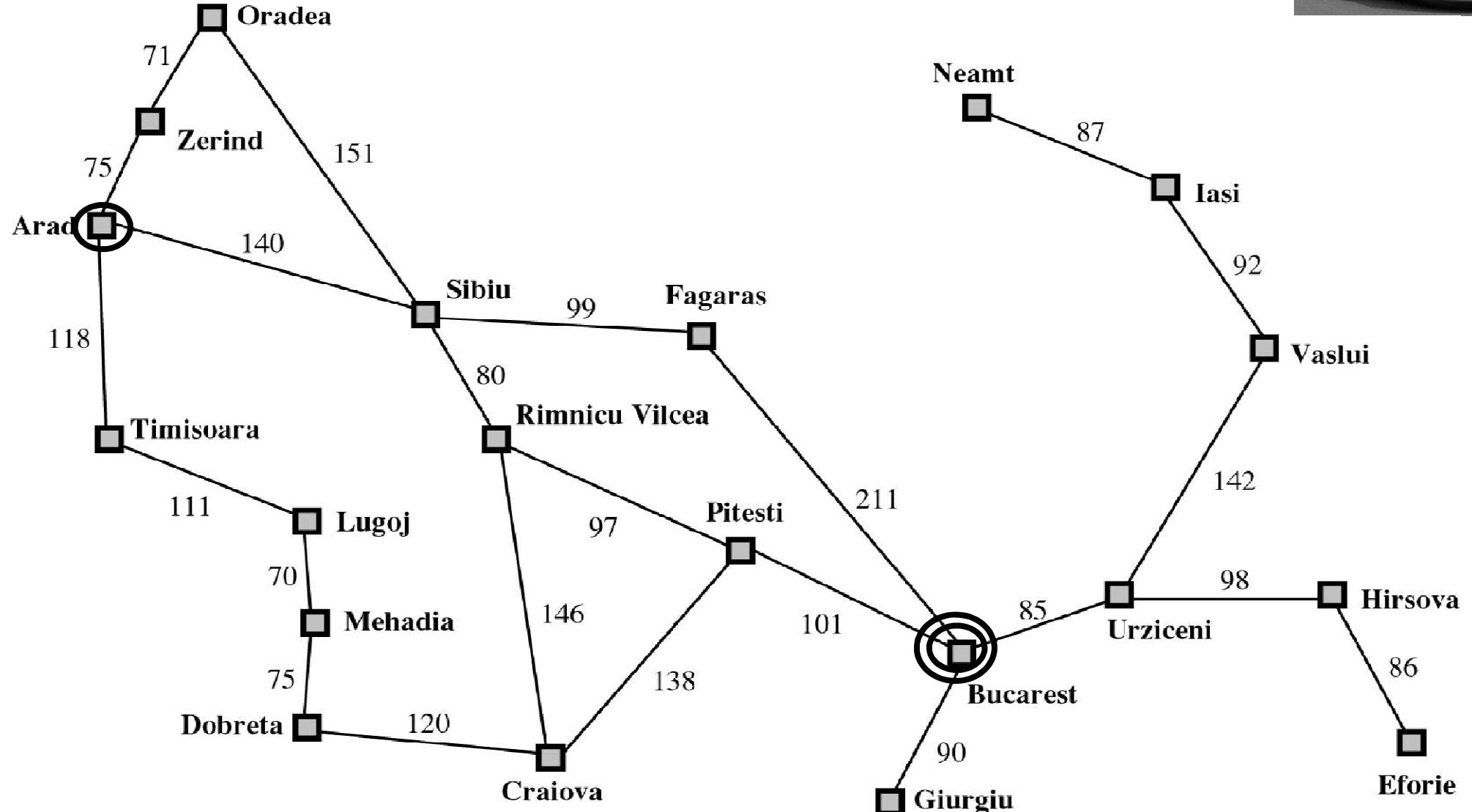
Acciones: conducir de una ciudad a otra

Acciones= { Ir(x,y) | x,y ∈ Ciudades, x≠y, TramoMapa(x,y)}

Función sucesor : S(En(x))= { < Ir(x,y), En(y) > }

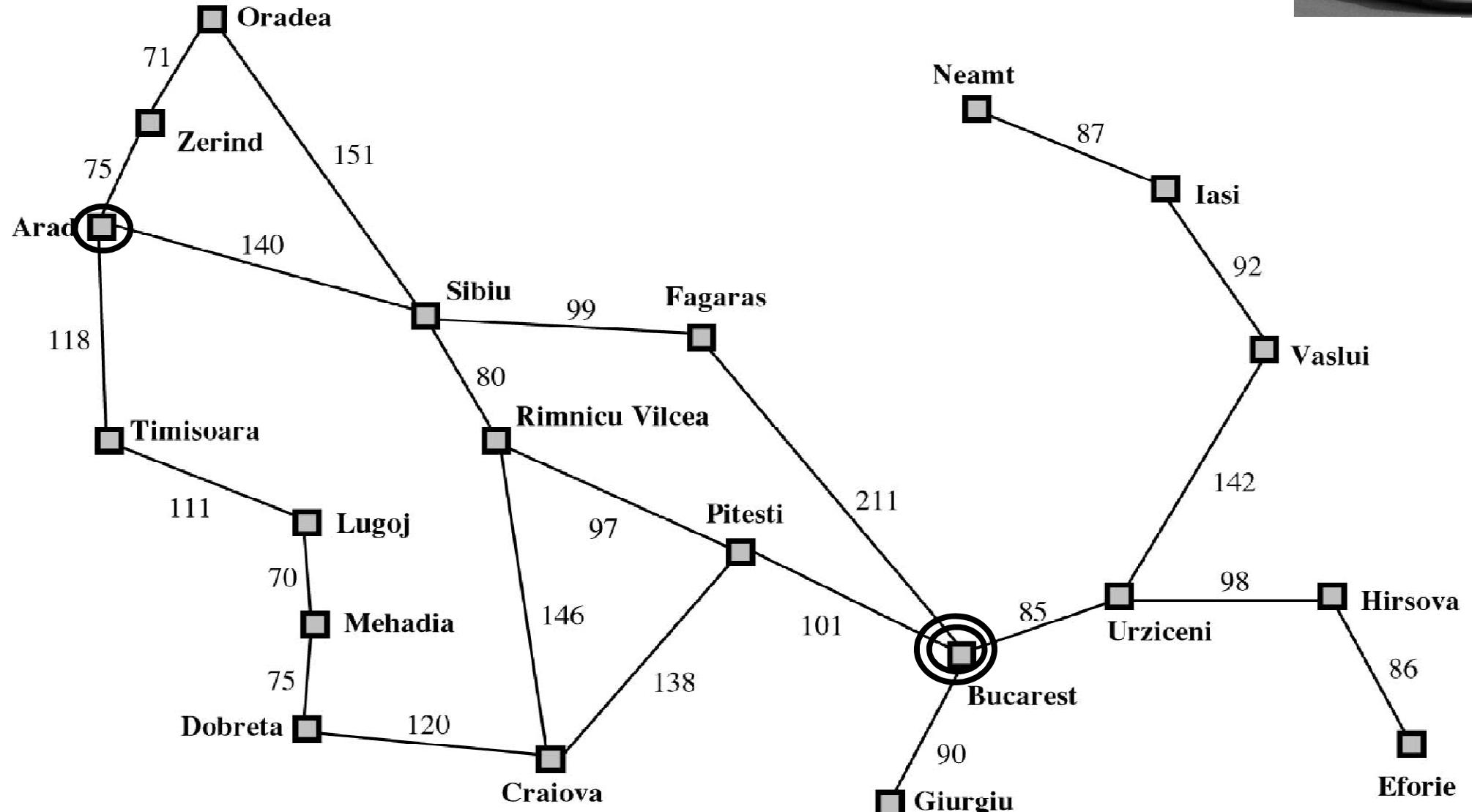


Ejemplo: Hallar rutas en Rumanía



$S(\text{En}(Arad))$?

Ejemplo: Hallar rutas en Rumanía



$S(En(Arad)) = \{ \langle Ir(Sibiu), En(Sibiu) \rangle, \langle Ir(Timisoara), En(Timisoara) \rangle, \langle Ir(Zerind), En(Zerind) \rangle \}$
 $S(En(Sibiu))$?
 $S(En(Neamt))$

Tipos de Problemas

Deterministas, completamente observables \Rightarrow problemas de estado simple

El agente sabe exactamente en qué estado estará; la solución es una secuencia.

No observables \Rightarrow problemas conformados (conformant or sensorless)

El agente puede no saber dónde está; la solución (si la hay) es una secuencia.

No deterministas y/o parcialmente observables \Rightarrow problemas de contingencia

Las percepciones del agente proporcionan información nueva sobre el estado actual.

La solución es un árbol o una política.

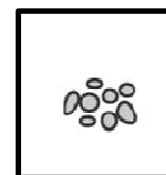
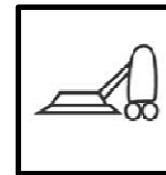
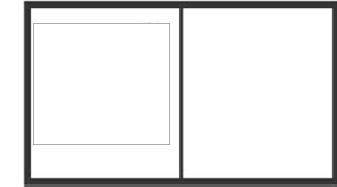
A menudo intercalan la búsqueda y la ejecución.

Problemas de espacio-estado desconocido \Rightarrow problemas de exploración (“en línea”).

Ejemplo: el mundo de la aspiradora



- Un mundo discreto de dos posiciones (celdas) adyacentes habitado por un robot
- El robot puede estar situado en cualquiera de las dos posiciones
- Las celdas pueden contener suciedad
- Se pretende llegar a una situación en la que el mundo esté limpio
- El robot aspiradora es el agente que puede cumplirlo
- Las acciones que el robot aspiradora puede hacer son:
 - Moverse a la derecha (si hay una celda a su derecha se mueve, si no se queda igual)
 - Moverse a la izquierda (equivalente a der)
 - Aspirar la suciedad de su celda (si no hay suciedad, se queda igual)
- Las percepciones del robot le permiten observar su posición y su entorno



Ejemplo: el mundo de la aspiradora



-
- Mundo discreto de dos celdas adyacentes habitado por un robot que puede aspirar o moverse a derecha o izquierda
 - Las celdas pueden contener suciedad, que se quiere eliminar
 - Las percepciones del robot le permiten observar su posición y su entorno

Formulación del objetivo:

Formulación del problema:

estados:

acciones:

función sucesor

Búsqueda de la solución:

secuencia

Ejecución:

Ejemplo: el mundo de la aspiradora



- Mundo discreto de dos celdas adyacentes habitado por un robot que puede aspirar o moverse a derecha o izquierda
- Las celdas pueden contener suciedad, que se quiere eliminar
- Las percepciones del robot le permiten observar su posición y su entorno

Formulación del objetivo:

mundo limpio (estados 7 u 8)

Formulación del problema:

estados: ? →

acciones: der, izq, asp

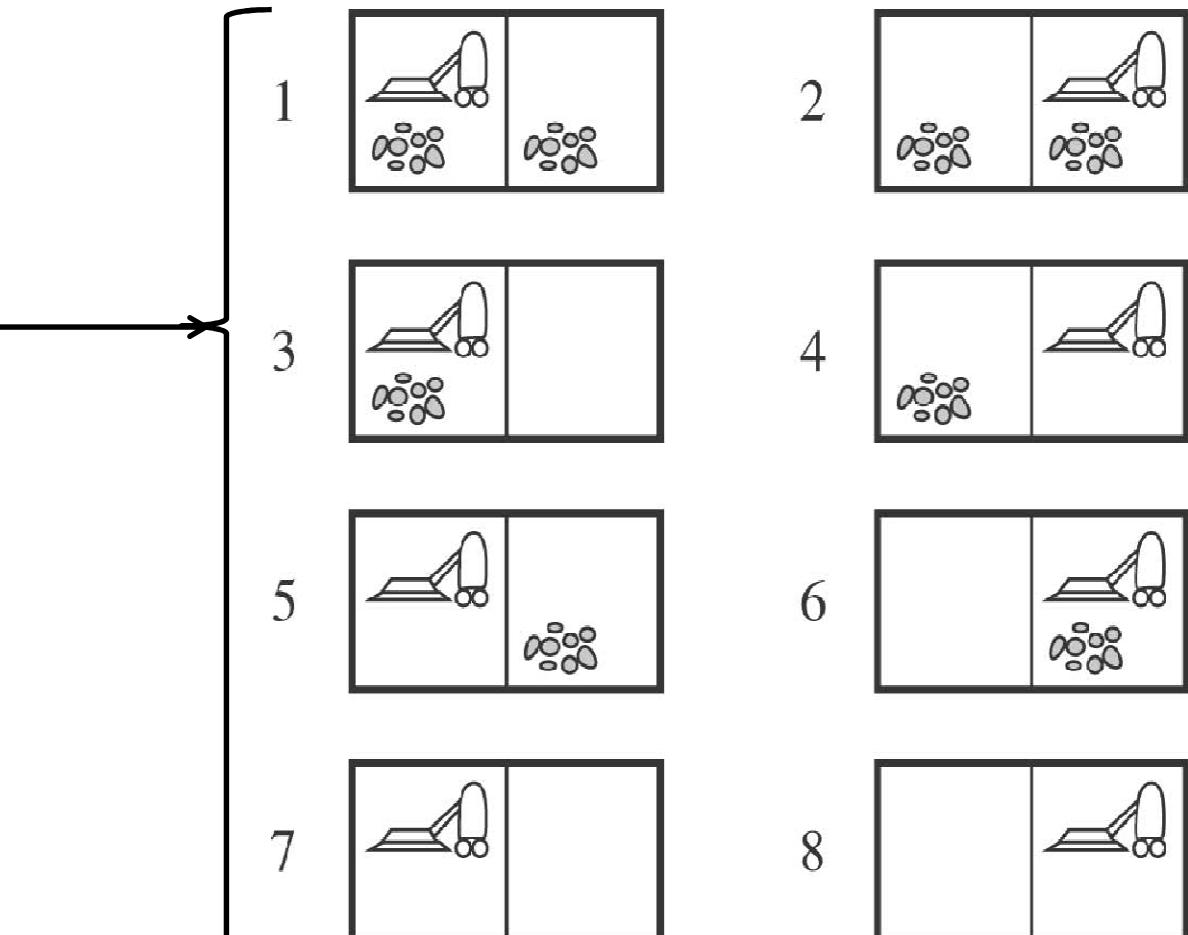
función sucesor ?

Búsqueda de la solución:

secuencia acciones

Ejecución:

ejecutar las acciones



Ejemplo: el mundo de la aspiradora



Formulación formal del problema:

$$Estados = \{ (\text{posRobot}, (\text{suc1}, \text{suc2})) \}$$

$$\text{t.q. } \text{posRobot} \in \{ \text{I}, \text{D} \}, \text{ suc1, suc2} \in \{0, 1\}$$

$$\text{estado inicial : } 5 = (\text{I}, (0, 1))$$

$$\text{estados finales: } 7 = (\text{I}, (0, 0)) \text{ y } 8 = (\text{D}, (0, 0))$$

acciones: der, izq, asp

función sucesor:

$$S(1) = \{ \langle \text{der}, 2 \rangle, \langle \text{izq}, 1 \rangle, \langle \text{asp}, 5 \rangle \}$$

$$S(2) = \{ \langle \text{der}, 2 \rangle, \langle \text{izq}, 1 \rangle, \langle \text{asp}, 4 \rangle \}$$

$$S(3) = \{ \langle \text{der}, 4 \rangle, \langle \text{izq}, 3 \rangle, \langle \text{asp}, 7 \rangle \}$$

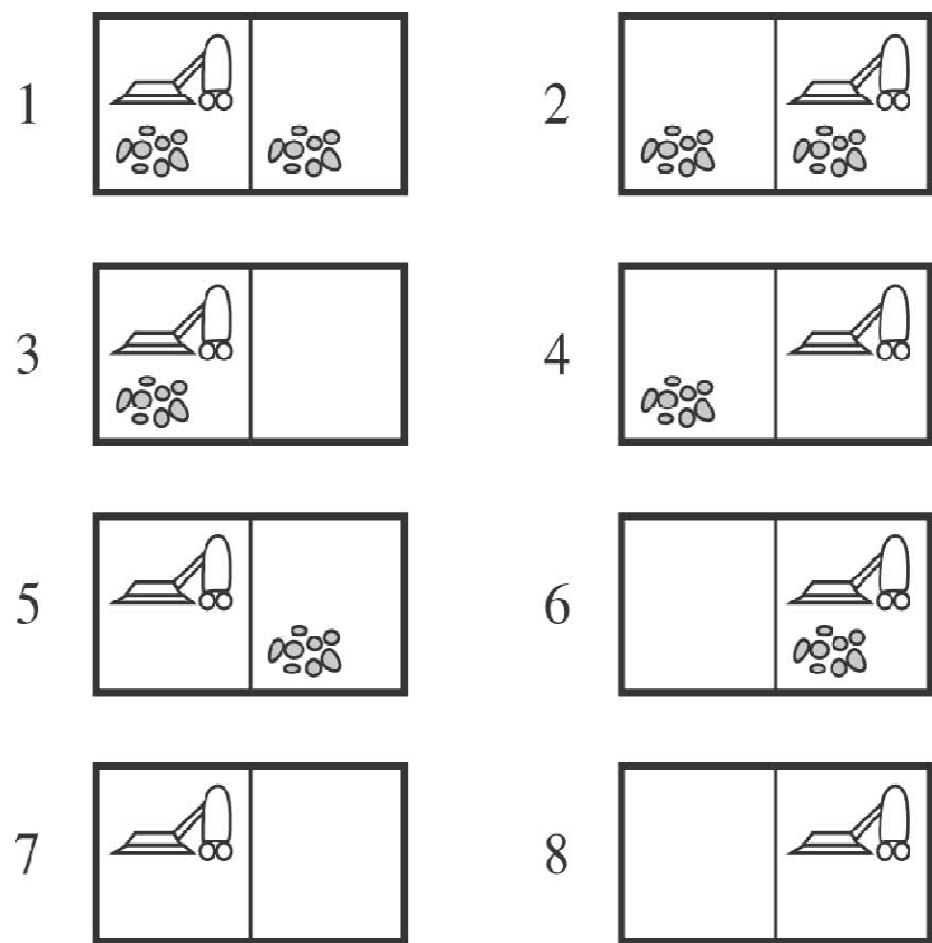
$$S(4) = \{ \langle \text{der}, 4 \rangle, \langle \text{izq}, 3 \rangle \}$$

$$S(5) = \{ \langle \text{der}, 6 \rangle, \langle \text{izq}, 5 \rangle \}$$

$$S(6) = \{ \langle \text{der}, 6 \rangle, \langle \text{izq}, 5 \rangle, \langle \text{asp}, 8 \rangle \}$$

$$S(7) = \{ \langle \text{der}, 8 \rangle, \langle \text{izq}, 7 \rangle \}$$

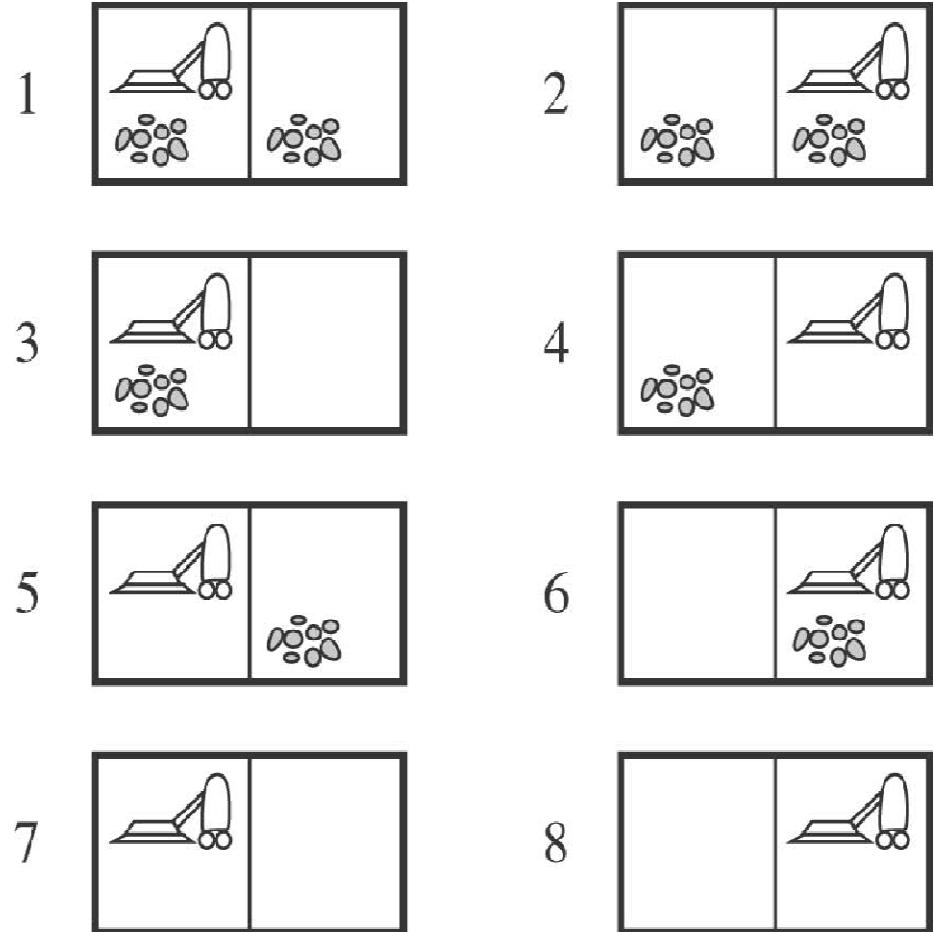
$$S(8) = \{ \langle \text{der}, 8 \rangle, \langle \text{izq}, 7 \rangle \}$$



Ejemplo: el mundo de la aspiradora



Estado simple (det, compl. obs.)
estado inicial 5. ¿Solución?



Tipos de Problemas

Deterministas, completamente observables \Rightarrow problemas de estado simple

El agente sabe exactamente en qué estado estará; la solución es una secuencia.

No observables \Rightarrow problemas conformados (conformant or sensorless)

El agente puede no saber dónde está; la solución (si la hay) es una secuencia.

No deterministas y/o parcialmente observables \Rightarrow problemas de contingencia

Las percepciones del agente proporcionan información nueva sobre el estado actual.

La solución es un árbol o una política.

A menudo intercalan la búsqueda y la ejecución.

Problemas de espacio-estado desconocido \Rightarrow problemas de exploración (“en línea”).

Ejemplo: el mundo de la aspiradora



Estado simple, estado inicial 5.

¿Solución?

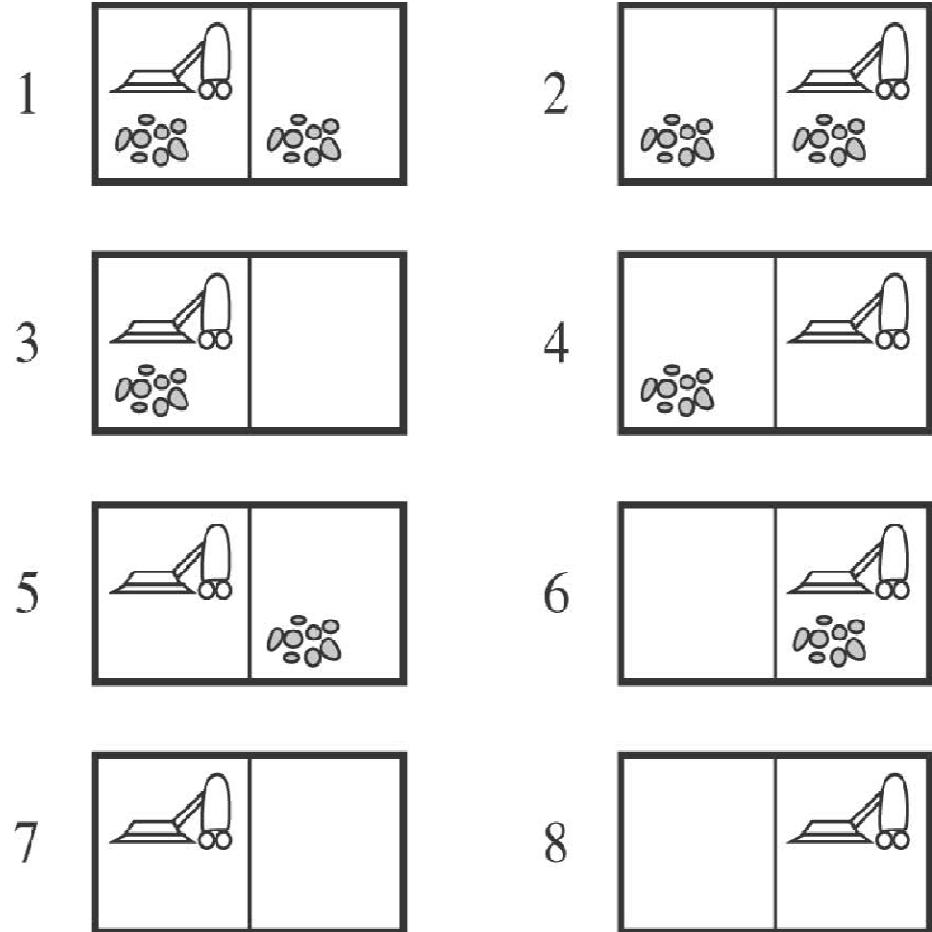
Sec= [Derecha, Aspirar] (5 -> 6 ->8)

No observable, estado inicial
 $\{1, 2, 3, 4, 5, 6, 7, 8\}$

Por ejemplo:

Derecha produce $\{2, 4, 6, 8\}$.

¿Solución?



Ejemplo: el mundo de la aspiradora



Estado simple, estado inicial 5.

¿Solución?

Sec= [[Derecha, Aspirar] (5 -> 6 ->8)

No observable, estado inicial

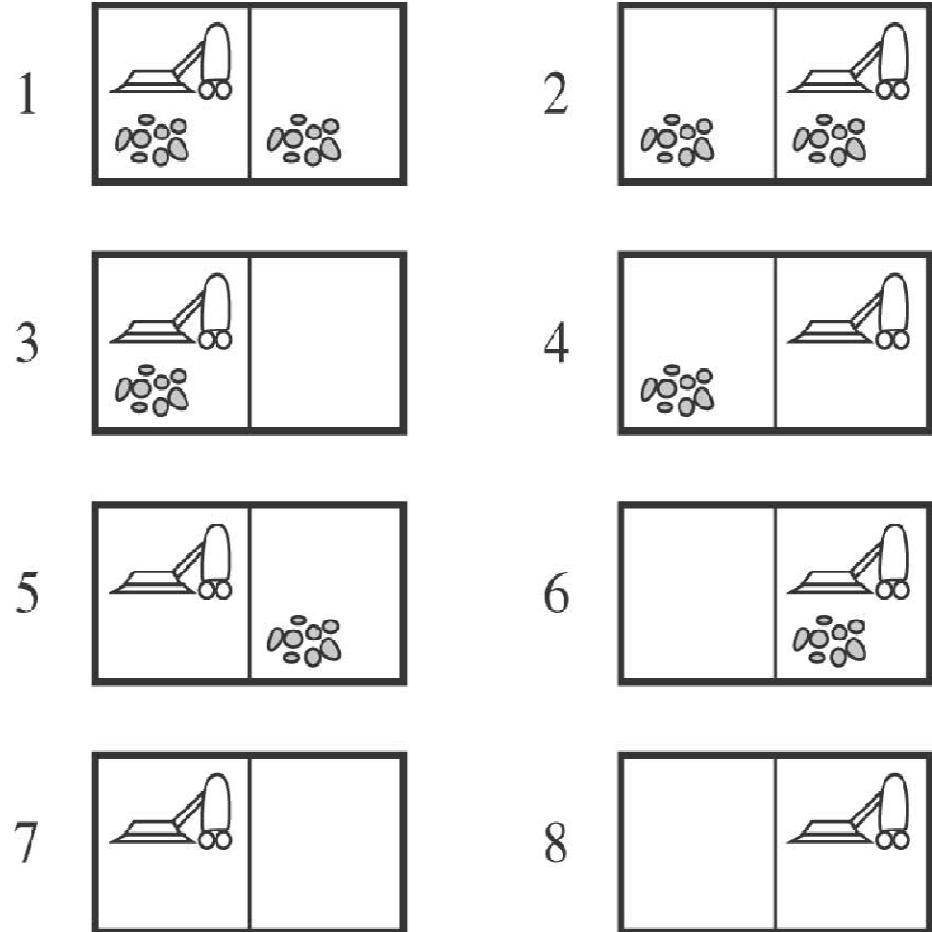
{1, 2, 3, 4, 5, 6, 7, 8}

Por ejemplo:

Derecha produce {2, 4, 6, 8}.

¿Solución?

Sec= [Derecha, Aspirar, Izquierda, Aspirar]



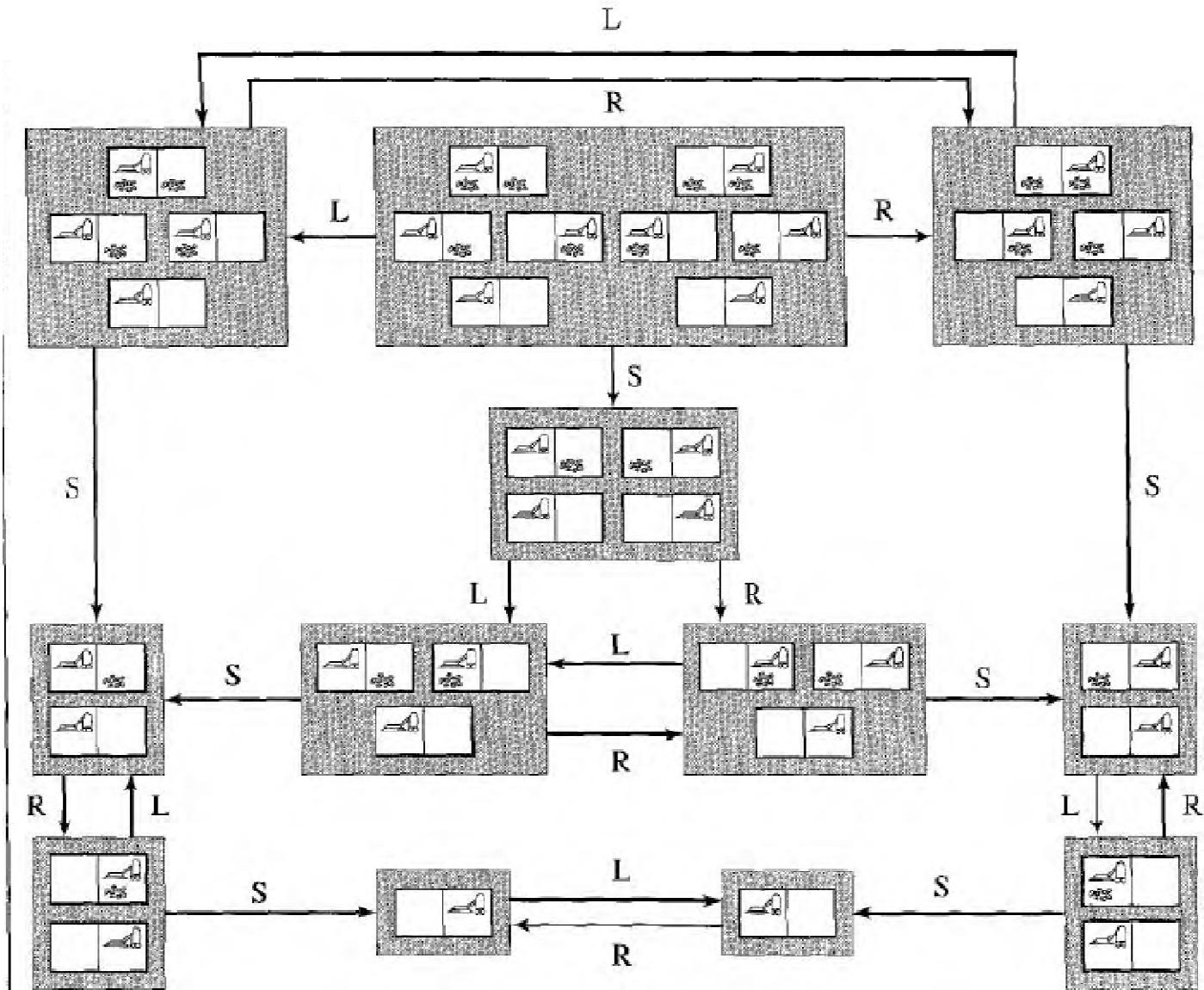
Ejemplo: el mundo de la aspiradora



- Problema

No observable :

Espacio de estados



Ejemplo: el mundo de la aspiradora



Estado simple, estado inicial 5.

¿Solución?

Sec= [Derecha, Aspirar] (5 -> 6 ->8)

No observable, estado inicial

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

Por ejemplo:

Derecha produce {2, 4, 6, 8}.

¿Solución?

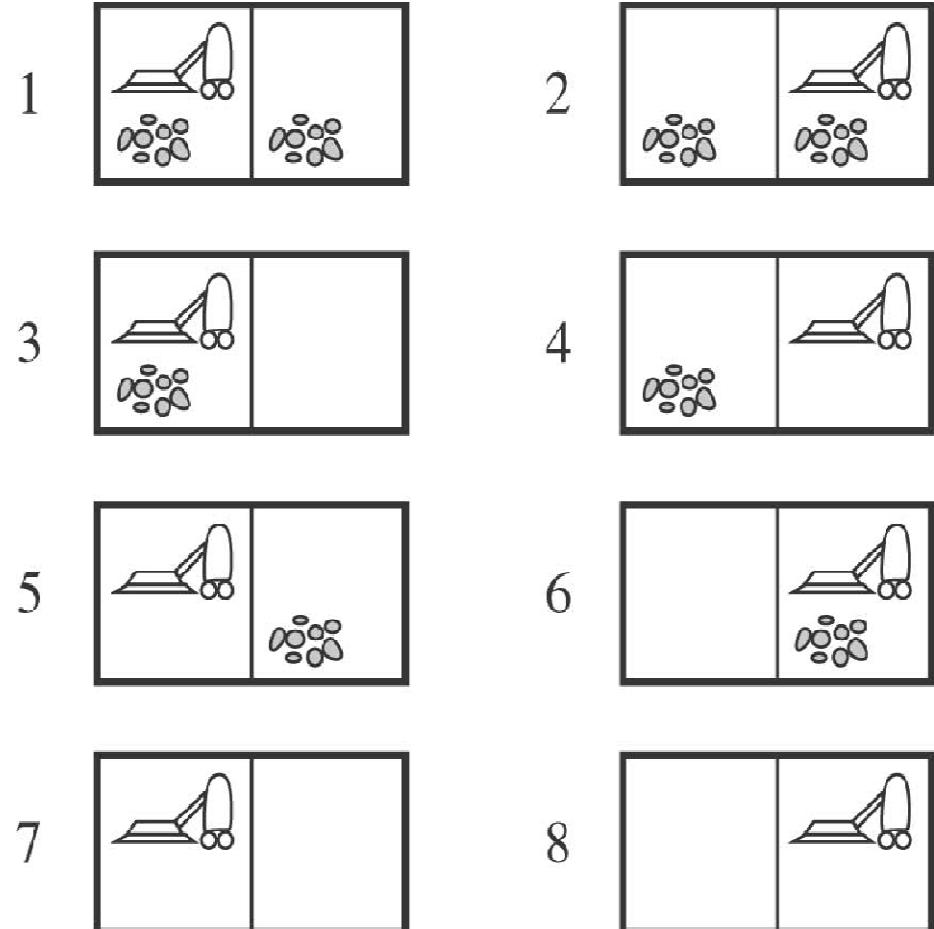
Sec= [Derecha, Aspirar, Izquierda, Aspirar]

Contingencia, estado inicial 5.

Ley de Murphy: Aspirar a veces deposita suciedad en la alfombra.

Sensores locales: suciedad, sólo en el lugar.

¿Solución?



Ejemplo: el mundo de la aspiradora



Estado simple, estado inicial 5.

¿Solución?

Sec= [Derecha, Aspirar]

Conformado, estado inicial

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

Por ejemplo:

Derecha produce {2, 4, 6, 8}.

¿Solución?

Sec= [Derecha, Aspirar, Izquierda, Aspirar]

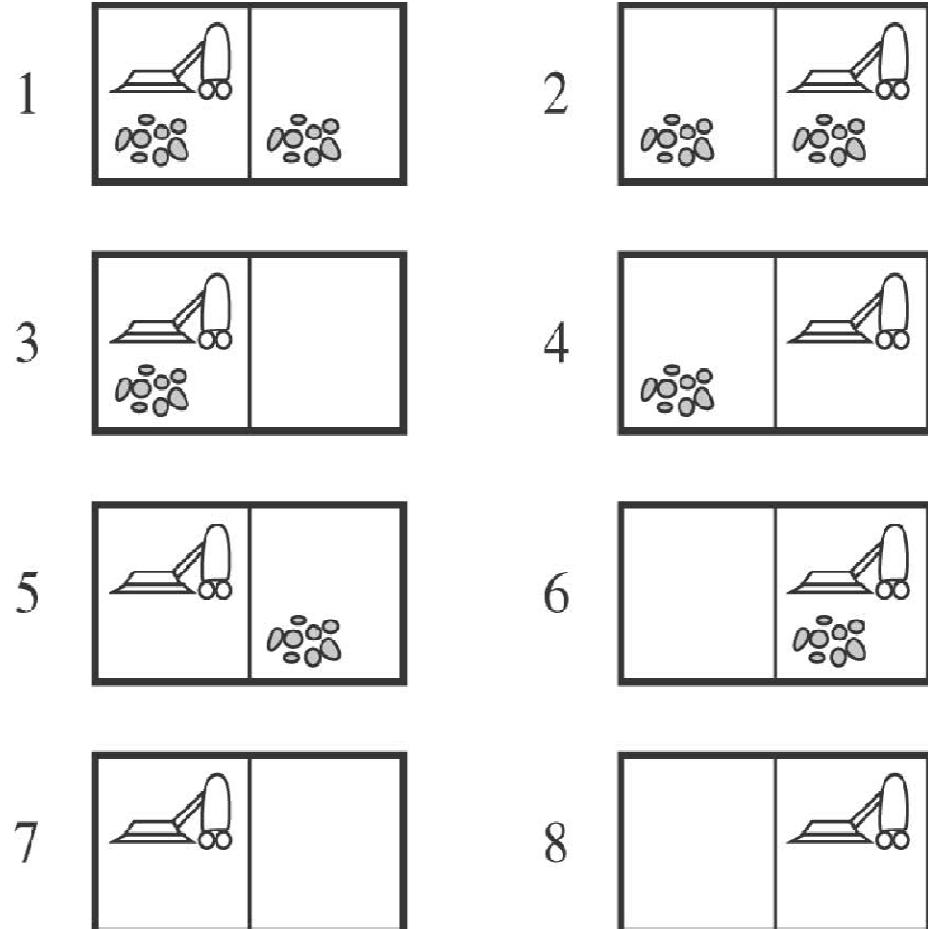
Contingencia, estado inicial 5.

Ley de Murphy: Aspirar a veces deposita suciedad en la alfombra.

Sensores locales: suciedad, sólo en el lugar.

¿Solución?

Sec= [Derecha, si suciedad entonces Aspirar]



Formulación de Problemas de Estado Simple

Un *problema* se define por cuatro elementos:

estado inicial por ejemplo, En(Arad)

función sucesor $S(x) = \text{conjunto de pares acción-estado}$
p. ej. $S(\text{En(Arad)}) = \{ \langle \text{Ir(Zerind)}, \text{En(Zerind)} \rangle, \dots \}$

test objetivo, puede ser

explícito, p. ej, $x = \text{En(Bucharest)}$
implícito, p. ej, $\text{NoSuciedad}(x)$

coste del camino (aditivo)

p.ej.: suma de distancias, número de acciones ejecutadas, etc.
 $c(x, a, y)$ es el coste individual, se supone que es ≥ 0



Una **solución** es una secuencia de acciones que parten desde un estado inicial hasta alcanzar un estado objetivo.

Selección de un espacio de estado

El mundo real es tremadamente complejo

⇒ el espacio de estado se debe *abstraer* de la solución del problema

Estado (abstracto) = conjunto de estados reales

Acción (abstracta) = combinación compleja de acciones reales

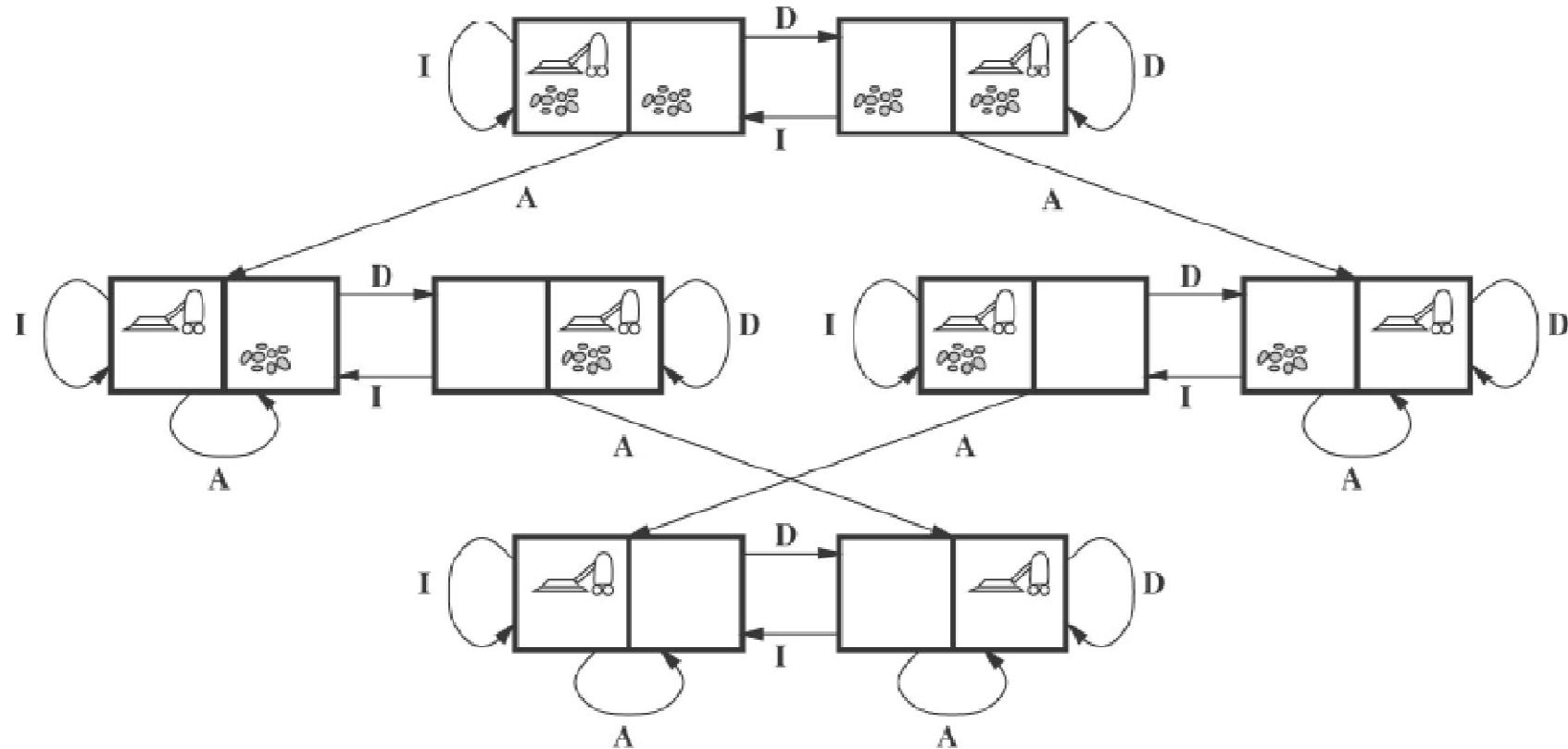
p.e.: Ir(Zerind) representa un conjunto complejo de rutas posibles, tours, paradas de descanso, etc.

Para una consecución garantizada, cualquier estado real En(Arad) debe llegar a *algún* estado real En(Zerind).

Solución (abstracta) = conjunto de caminos reales que son soluciones en el mundo real

¡Cada acción abstracta debería ser “más fácil” que el problema original!

Ejemplo: espacio de estados para el mundo de la aspiradora



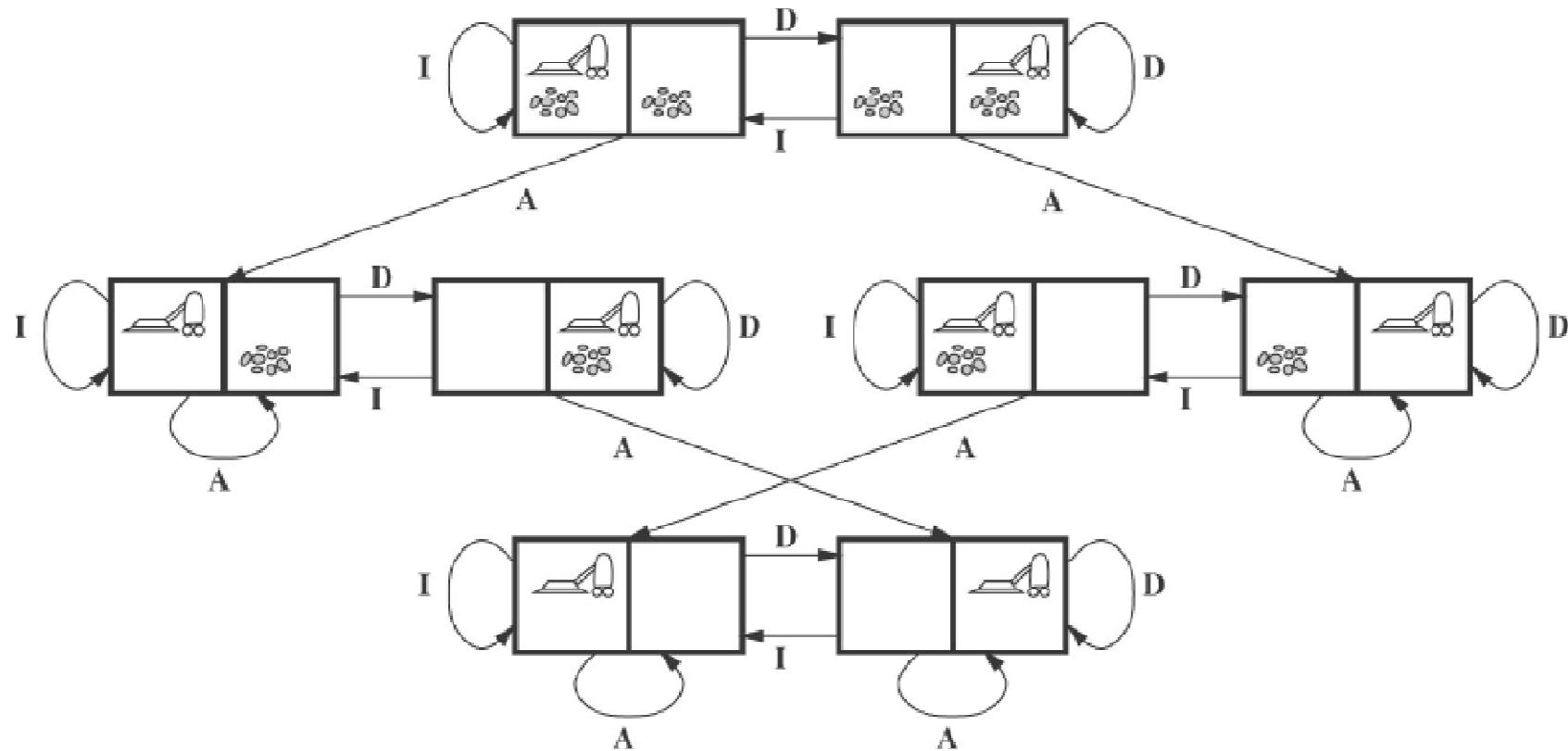
¿Estados?

¿Acciones?

¿Test objetivo?

¿Coste del camino?

Ejemplo: espacio de estados para el mundo de la aspiradora



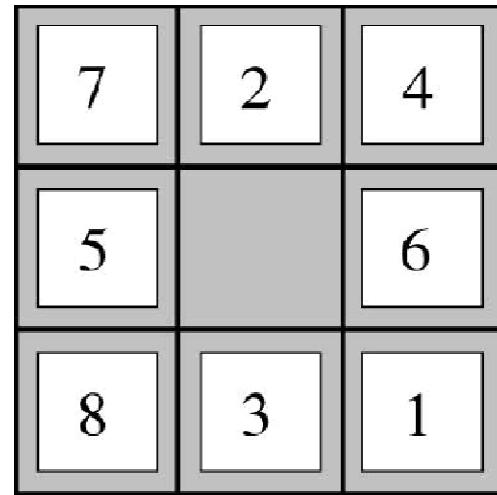
¿Estados? Suciedad completa y localizaciones de robot (ignorar *cantidades* de suciedad)

¿Acciones? Izquierda (I), Derecha (D), Aspirar (A), NoOp

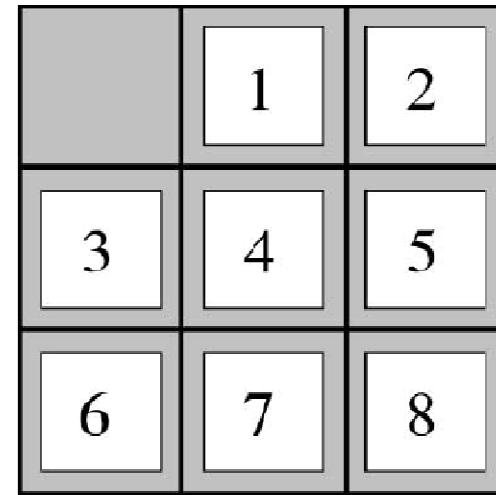
¿Test objetivo? No suciedad

¿Coste del camino? 1 por acción (0 por NoOp)

Ejemplo: el 8-puzzle <http://www.8puzzle.com/>



Estado Inicial



Estado Objetivo

¿Estados?

¿Acciones?

¿Test objetivo?

¿Coste del camino?