

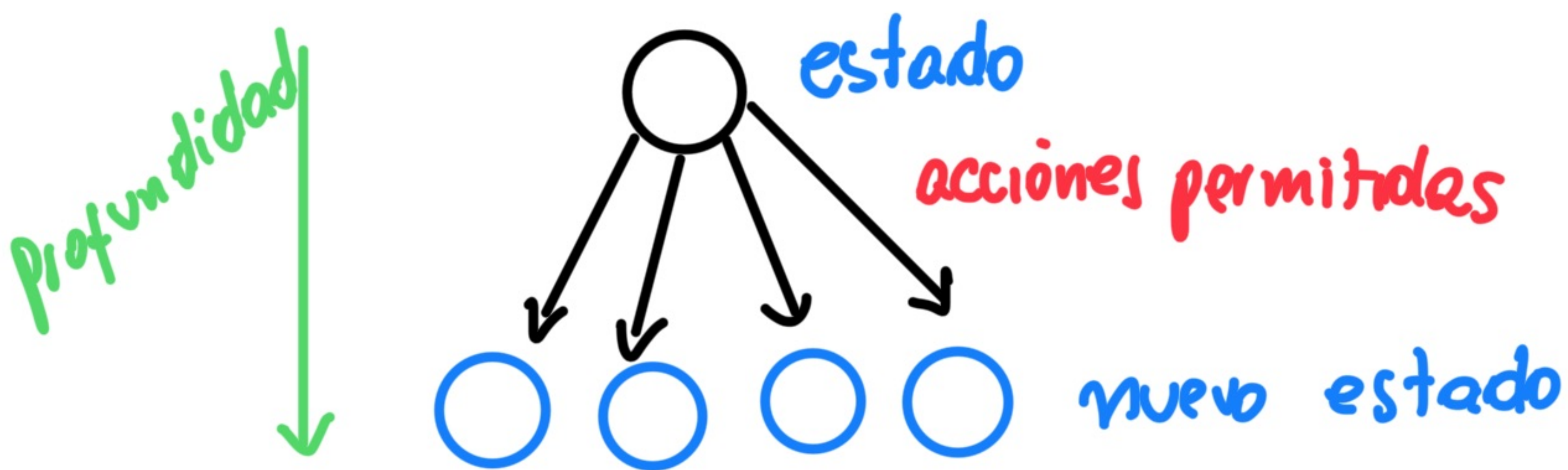
Problemas $\begin{cases} \rightarrow \text{Juguete} \\ \rightarrow \text{Mundo real} \end{cases}$

Problema $\left\{ \begin{array}{l} \text{Estado} \\ \text{Operadores} \\ \text{Test de objetivo} \\ \text{Costo del camino} \end{array} \right.$

8 reinas
 \hookrightarrow backtracking

Problema del
vendedor viajero

Búsqueda de soluciones
estado inicial \longrightarrow estado final



acciones tienen un costo asociado

algun nodo hoja es el estado objetivo y
con un costo asociado (Σ costos asociados)

* Profundidad árbol 8 reinas $h=8$

* Anchura hacia el lado

1. Realizar test objetivo (al principio falso)
2. Expandir estado actual (abrir hacia el lado)
Luego en el nuevo estado volver a realizar el test objetivo
3. Seleccionar estado
4. Moverse al estado seleccionado

Búsqueda → No informada (la ciegas)
 (solo test objetivo, estrategia no depende del objetivo)
 Informada (heurística)
 (tiene estimación de la distancia al objetivo)

Criterios de evaluación → Encuentra una solución?
 Encuentra la mejor solución?
 Costo computacional (tiempo y espacio)

Búsqueda en anchura (BFS?)

- nodos alcanzables del nodo actual
- no profundiza hasta expandir todos los nodos
- factor de ramificación: cuantos nodos hijos
- crecimiento exponencial (tabla ppt)

Criterios

Encuentra solución? Si

Encuentra mejor solución? Si, todas con igual costo

Costo computacional Muy alto 😞

Búsqueda con costo uniforme

busca nodo con menor costo de camino.

expande y evalúa en la frontera (muestra los nodos en competencia)

Complejidad ($\epsilon > 0$) para que no quede en un loop haciendo nada

Complejidad temporal $O(b^{1+(C^*/\epsilon)})$

Complejidad espacial mismo que BFS

Optimalidad

Búsqueda en profundidad (DFS)

expande forma iterativa

Encuentra solución? No siempre

mejor solución? No necesariamente

Costo computacional? Bajo 😊

DFS → Limitada Busca con un límite (actúa como final árbol)
 Iterativa con límite que se incrementa gradualmente
 ↓
 Se ejecuta varias veces cambiando el límite

Aplicaciones

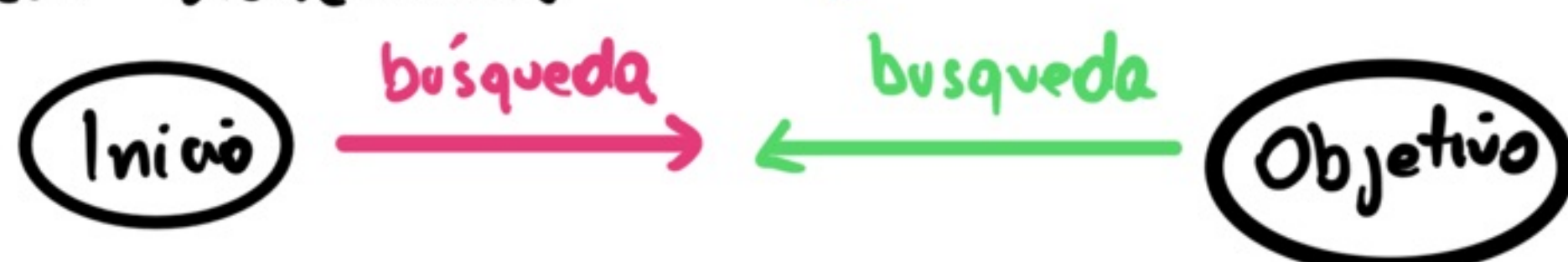
Calendarización tareas, recursos

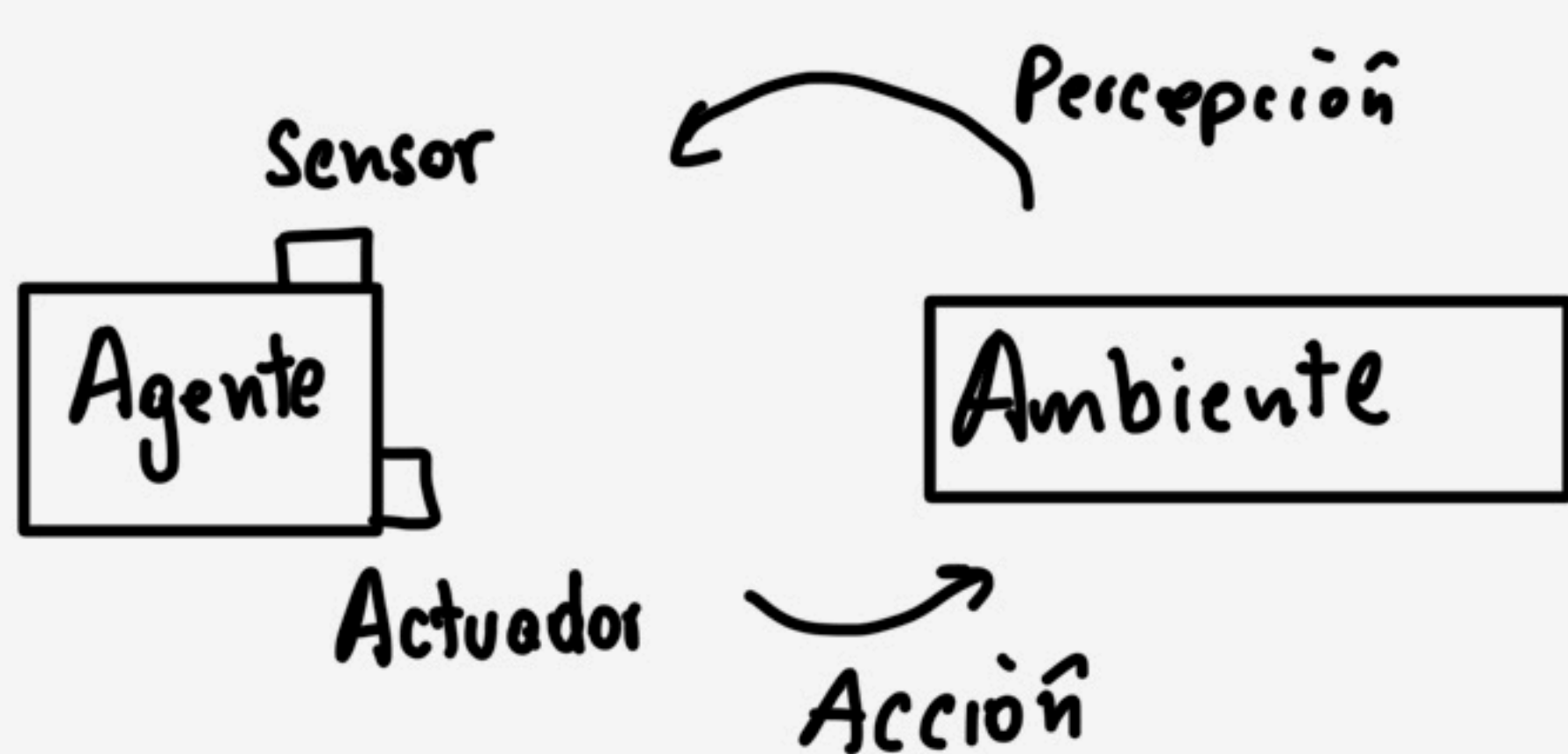
Navegación robots ambiente desconocido

↳ Aspiradora robot, no conoce habitación
BFS no sirve

Búsqueda bidireccional

Cuando sabemos como es el objetivo





- Agente humano
- Robot
- Software

Comportamiento

$$f: P^* \rightarrow A$$

if → then
Secuencia → acción

Ejemplo Robot aspiradora
Función racional

Racionalidad → hacer "lo correcto"
↳ Maximiza la operación

No es omnisciente

↳ desconoce aspectos del ambiente
no tiene noción del éxito

Aprendizaje y Autonomía

Agentes $\xrightarrow{\text{resuelven}}$ tareas

Desempeño
Ambientes
actuadores
sensores

destino
ganancias
tiempo

Propiedades ambientes

¿Que problemas puede resolver?

episódico: clasificar fotos no importa la decisión anterior

Secuencial: importan las decisiones anteriores

estático: ajedrez

dinámico: mundo real

Agente $\begin{cases} \nearrow \text{Arquitectura} \\ \searrow \text{Programa} \end{cases}$

① Reflejos simples if-then
(innatos) Potencia limitada

Util en ambiente observable

② Estado/modelo

③ Basado en metas

④ Utilidad → Valor numérico a cada acción

Ayuda a ver cual elegir

Agentes reflejos no útiles en ambientes complejos

Inicio
Objetivo / destino
Operadores (acciones)

Formulación de metas
Formulación de problema

metas tienen preferencias

Problema de búsqueda (ciudades)
Ambiente \rightarrow observable

Busqueda : secuencia de acciones

Camino : secuencia que lleva estado
A a un estado B
(inicial) (final)

Costo $\left\{ \begin{array}{l} \rightarrow \text{Busqueda} \\ \rightarrow \text{Camino} \end{array} \right\}$ Costo Total

Ej: Ir desde la casa a clases

estado $\left\{ \begin{array}{l} \rightarrow \text{casa} \\ \rightarrow \text{sala} \\ \rightarrow \text{camino} \\ \rightarrow \text{Uber, etc} \end{array} \right.$