Inteligencia Artificial

Resumen

Dorian Wozniak

Índice

1 AGENTE DE RESOLUCIÓN DE PROBLEMAS Y BÚSQUEDAS **2**
1.1 **Agentes** . . . 2
1.2 **Modelos conceptuales y matemáticos** . . . 2
1.3 **Lenguaje de representación** . . . 3
1.4 **Inteligencia artificial** . . . 3
1.5 **Algoritmos básicos de búsqueda** . . . 3

2 ESTRATEGIAS NO INFORMADAS (BÚSQUEDA CIEGA) **5**
2.1 **Primero en anchura (BFS)** . . . 5
2.2 **Coste uniforme (UCS)** . . . 5
2.3 **Primero en profundidad (DFS)** . . . 5
2.4 **Profundidad limitada (DLS)** . . . 6
2.5 **Profundidad iterativa (IDS)** . . . 6
2.6 **Bidireccional (BS)** . . . 6

Capítulo 1

AGENTE DE RESOLUCIÓN DE PROBLEMAS Y BÚSQUEDAS

1.1 Agentes

Un **agente** es una entidad que percibe su entorno a través de sensores y actúa sobre el a través de actuadores.

- **Agente reflejo**: Reacciona al estado actual del entorno sin tener en cuenta su historia. Simples pero poco viables.
- **Agente reflejo basado en modelo**: El agente contiene un modelo del entorno para representar su estado interno.
- **Agente reflejo basado en objetivo y modelo**: Toma decisiones en consecuencia de sus acciones, con una serie de objetivos deseables (objetivos). Elige y organiza acciones anticipando cual será su resultado.

1.2 Modelos conceptuales y matemáticos

Un **modelo conceptual** es la forma de describir los elementos de un problema. Bueno para delmiitar restricciones al modelo. No tiene en cuenta aspectos computacionales.

Pasos generales:

- Formulación del **objetivo** (estado del mundo a alcanzar)
- Formulación del **problema** (estado y acciones a considerar)
- **Búsqueda** (Secuencia de acciones que llevan al objetivo)
- Ejecución

Matematicamente se puede representar como un grafo. **Modelo de estados restringido (Σ)**:

- **Espacio de estados** finito y discreto: S
- **Estado**: $s \in S$
- Estado **inicial** : $s_0 \in S$
- Conjunto de estados **objetivos**: $G \subseteq S$
- **Acciones aplicables** en estado: $A(s) \subseteq A$
- **Función transición**: $f(s, a), s \in S, a \in A(s)$
- **Función de coste**: $c(a, s) > 0$
- **Una solución**: Una secuencia de acciones a_i que lleve del estado inicial s_0 al estado objetivo $s \in SG$.
- **Una solución óptima** que minimiza el coste: $\sum_{i=0}^{n} c(a_i, s_i)$

Restricciones:

- Σ es completamente **observable** (se conoce bien el estado), **determinista** (una acción solo devuelve un único estado nuevo) y **estático** (no hay eventos que cambien el estado del sistema).
- La solución es una secuencia ordenada de acciones.
- Las acciones no tienen duración

1.3 Lenguaje de representación

Los **agentes de resolución de problemas** son agentes dirigidos por objetivos que utilizan **representaciones atómicas**, donde a cada estado le corresponde un símbolo.

Los agentes dirigidos por objetivos con representaciones más sofisticadas se les conocen como agentes de planificación.

1.4 Inteligencia artificial

La IA consiste en el modelado de sistemas con espacios de estados de gran escala. No se puede representar todo el espacio de estados en memoria, por lo que el modelo va generando el espacio de estados durante la exploración.

1.5 Algoritmos básicos de búsqueda

Una búsqueda simula la exploración del espacio de estados y genera sucesores de estados ya explorados.

Un **grafo del espacio** de estados representa el problema de búsqueda:

- Cada **nodo** es un estado posible
- Los **arcos** representan transiciones a partir de acciones
- El **test de objetivo** es el conjunto de uno o más nodos
- Suelen ser demasiado grandes para ser almacenados

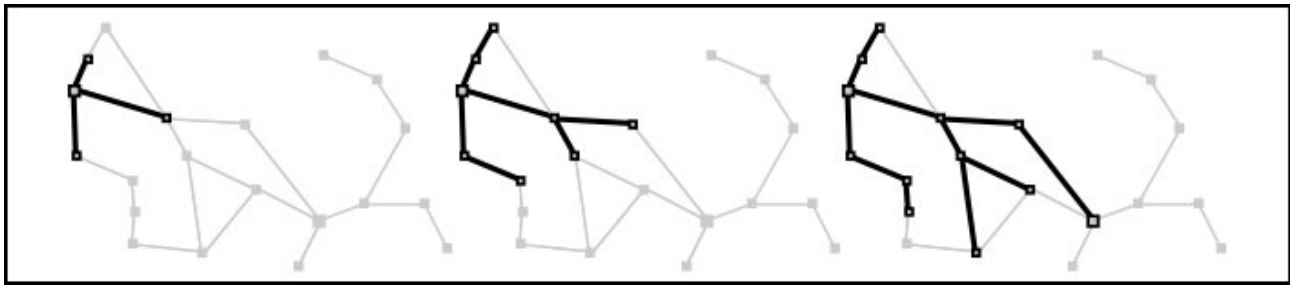

Figura 1.1: Busqueda en grafo

Las búsquedas generan un **árbol de búsqueda**:

- Cada **nodo** representa un estado.
- Un estado (nodo padre) se **expande** al generar nuevos estados aplicando acciones permitidas sobre el estado original (nodos hijos).
- El conjunto de nodos no expandidos se denomina **frontera**.
- Se suelen repetir estructuras en un árbol, mientras que en un grafo un estado solo aparece una vez

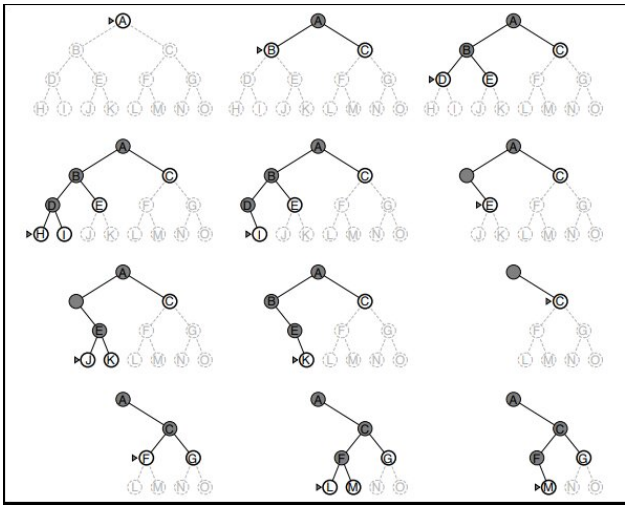

Figura 1.2: Busqueda en grafo

Una **estrategia de búsqueda** se define según el órden de expansión de los nodos. Se evalúan según su:

- **Completitud** (si encuentra la solución si existe).
- **Optimalidad** (si encuentra la solución menos costosa)
- **Complejidad temporal/espacial** (cuantos nodos expande/almacena)

La **complejidad** se mide según la dificultad del problema:

- **Factor de ramificación máximo** (b) (número máximo de sucesores de un nodo)
- **Profundidad de la solución de menor coste** (d)
- **Máxima profundidad de cualquier camino del espacio** (m)

Capítulo 2

ESTRATEGIAS NO INFORMADAS (BÚSQUEDA CIEGA)

Las **búsquedas ciegas** solo utilizan la información de la definición del problema.

2.1 Primero en anchura (BFS)

Se expande el nodo no expandido menos profundo. La frontera es una cola FIFO.

Completitud	**SI** cuando el nodo objetivo está a una profundidad finita d y el número de nodos b es finito
Optimalidad	**SI** siempre que el coste no decrezca con la profundidad
Complejidad temporal	$\mathbf{O(b^{d+1})}$ (se generan todos los nodos del último nivel)
Complejidad espacial	$\mathbf{O(b^{d+1})}$ (determinado por nodos en la frontera)

Aunque puede garantizar encontrar el resultado óptimo, tiene **complejidad exponencial**.

2.2 Coste uniforme (UCS)

Modificaciónd de BFS donde se expande el nodo con menor coste (cola ordenada).

Completitud	**SI** cuando cada paso tiene al menos un coste mayor o igual a una constante ϵ
Optimalidad	**SI** siempre que no se reduzca el coste de un camino al expandirlo
Complejidad temporal	$\mathbf{O(b^{1+(C*/\epsilon)})}$ donde $C*$ es el coste de la solución optima
Complejidad espacial	$\mathbf{O(b^{1+(C*/\epsilon)})}$ donde $C*$ es el coste de la solución optima

2.3 Primero en profundidad (DFS)

Se expande el nodo más profundo primero. La frontera es una pila LIFO.

Completitud	**SI** para búsquedas en grafo (en el pero caso expandirá cada nodo). **NO** para búsquedas en árbol (se debe comprobar que no haya bucles infinitos).
Optimalidad	**NO**
Complejidad temporal	$\mathbf{O(b^m)}$ (La profundidad máxima m puede ser mucho mayor que la de la solucióón d e incluso infinita)
Complejidad espacial	$\mathbf{O(bm)}$ para búsquedas en árbol. $O(m)$ con algoritmo de *backtracking*

La DFS es la base para otras técnicas de búsqueda por su **eficiencia espacial**.

2.4 Profundidad limitada (DLS)

Se aplica un límite de profundidad l al algoritmo DFS.

Completitud	**SI** cuando $l \geq d$
Optimalidad	**SI** cuando $l \leq d$
Complejidad temporal	$\mathbf{O(b^l)}$
Complejidad espacial	$\mathbf{O(bl)}$

Puede paliar el mal rendimiento de la DFS, pero o es incompleta o poco poco óptima según la profundidad del camino menos costoso.

2.5 Profundidad iterativa (IDS)

Intenta combinar el coste espacial de la DFS con la optimalidad del BFS. Realiza búsquedas en profundidad sucesivas con un nivel de profundidad máximo acotado que crece con cada iteración.

Completitud	**SI** (sin caminos infinitos)
Optimalidad	**SI** siempre que siempre que el coste no decrezca con la profundidad
Complejidad temporal	$\mathbf{O(b^d)}$
Complejidad espacial	$\mathbf{O(bd)}$

2.6 Bidireccional (BS)

Se busca simultaneamente desde el nodo inicial y desde el objetivo. Se comprueba que un nodo pertenezca a la otra frontera antes de expandir.

Completitud	**SI** si ambas búsquedas son de anchura
Optimalidad	**SI** si ambas búsquedas son de anchura
Complejidad temporal	$\mathbf{O(b^{d/2})}$
Complejidad espacial	$\mathbf{O(b^{d/2})}$