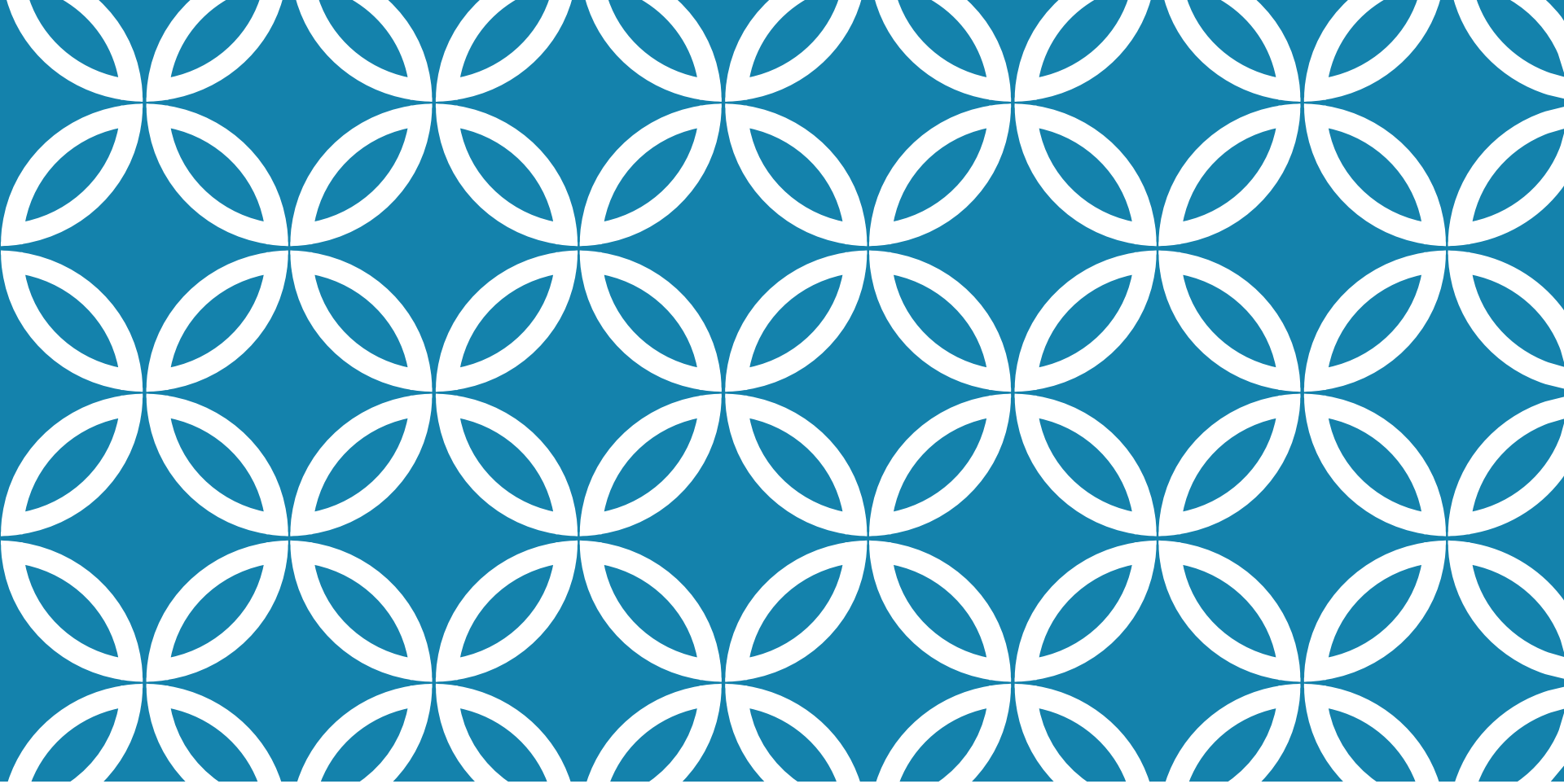


BÚSQUEDA HEURÍSTICA

José Manuel Molina López
Grupo de Inteligencia Artificial Aplicada

CONTENIDO

Búsqueda III. Búsqueda Heurística

- El concepto de Heurística
- Algoritmo de Escalada
- Algoritmo de Primero el Mejor

Búsqueda IV. Búsqueda de dos Agentes

Concepto de Heurística

Escalada

Primero el Mejor

BÚSQUEDA HEURÍSTICA

CONCEPTO DE HEURÍSTICA

Definición de Heurística
Problemas Relajados
Heurística Admisible

CONCEPTO DE HEURÍSTICA

Heur´ıstica: (del griego “heurisko”: “yo encuentro”) conocimiento parcial sobre un problema/dominio que permite resolver problemas eficientemente en ese problema/dominio

Posibilidades:

Si se tiene conocimiento perfecto: aplicamos inferencia	Si no se tiene conocimiento: búsqueda sin información	En la mayor parte de los problemas que resuelven los humanos, se está en posiciones intermedias

HEURÍSTICAS

El estado puede darnos información parcial de cómo resolver el problema

Representación de las heurísticas:

- Funciones numéricas h(n)
- Meta-reglas

Las funciones heurísticas se descubren resolviendo modelos simplificados/relajados del problema real

PROBLEMAS RELAJADOS.

DISTANCIA ENTRE DOS PUNTOS DE UN MAPA 2D

La distancia real debe considerar el trazado de las calles

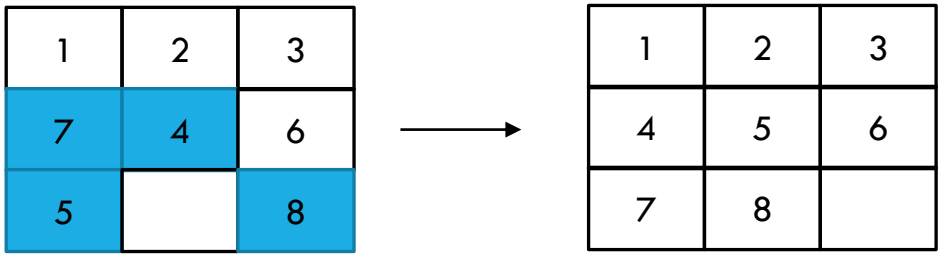
Una línea recta es una relajación del camino real que podemos seguir

Como mínimo tenemos que recorrer la distancia directa entre dos puntos

PROBLEMAS RELAJADOS. 8-PUZZLE

Casillas mal colocadas

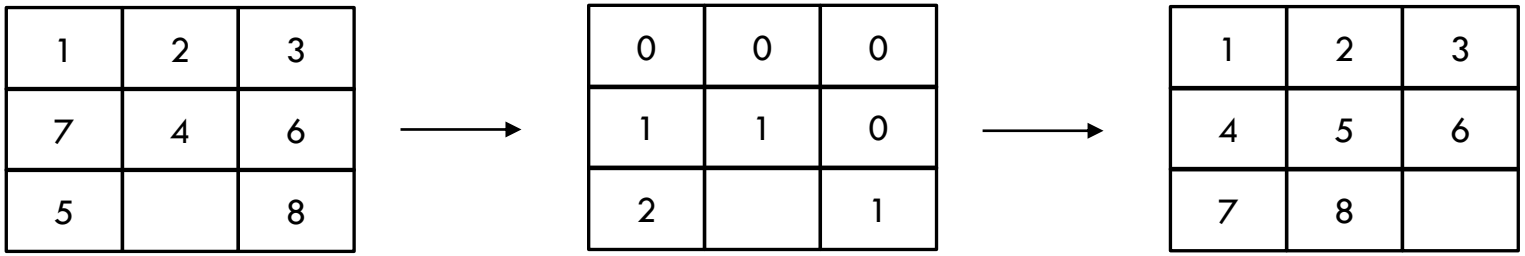
- No tiene en cuenta la distancia a la que están
- Cuatro casillas mal colocadas: h1(n) = 4



PROBLEMAS RELAJADOS. 8-PUZZLE

Manhattan

- Suma de las distancias de cada pieza a su destino
- h2(n) = 5



PROBLEMAS RELAJADOS. 8-PUZZLE

Problema: El puzzle funciona mediante intercambios

Restricciones (precondiciones de las acciones):

- sólo se puede mover el blanco
- el movimiento sólo se puede hacer a casillas adyacentes horizontal o vertical
- en cada paso, se intercambian los contenidos de dos casillas

Relajaciones:

- si quitamos las dos primeras restricciones, generamos una heurística del tipo “número de casillas mal colocadas”
- si quitamos la primera restricción, generamos otra del tipo “distancia al destino”

HEURÍSTICAS ADMISIBLES

Una función que nunca sobreestima el coste real $h^*(n)$ de alcanzar la solución desde el nodo n:

$$h(n) \leq h^*(n)$$

Condición necesaria para garantizar una solución óptima de ciertos algoritmos heurísticos como el A*.

Las funciones obtenidas al resolver un problema relajado de forma óptima producen heurísticas admisibles.

Ejemplos: la recta entre dos puntos en un plano

ALGORITMO DE ESCALADA

Idea general
Algoritmo
8-puzzle
Problemas
Propiedades

IDEA GENERAL

Elegir a cada momento el mejor nodo entre los sucesores directos

Técnica avariciosa, no reconsidera el camino elegido

Necesita una función de evaluación f (n) que indique el mejor nodo

El valor más alto (maximizar $f(n)$): El número de casillas bien colocadas

El valor más bajo (minimizar $f(n)$) utilizando una heurística que nos estime el camino hasta la meta: el número de casillas mal colocadas. En este caso $f(n) = h(n)$

ALGORITMO

N=Estado-inicial

EXITO=Falso

Hasta que Camino-Sin-Salida(N) O EXITO

    Generar los sucesores de N

    SI algún sucesor es Estado-final

    ENTONCES EXITO=Verdadero

    SI NO

        Evaluar cada nodo con f (n)

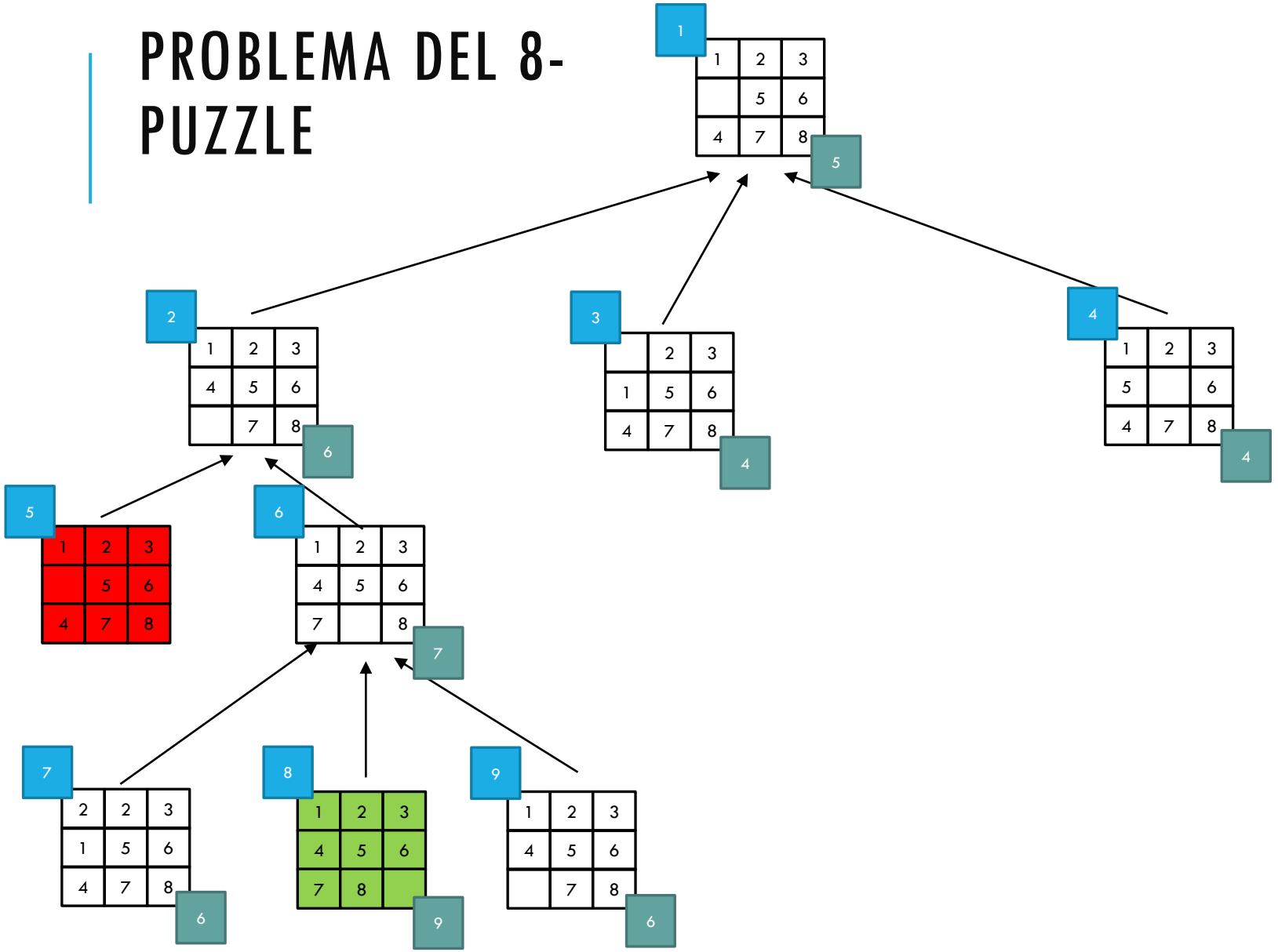
        N=mejor sucesor

Si ÉXITO Entonces

    Solución=camino desde nodo del Estado-inicial al nodo N por los punteros

Si no, Solución=fracaso

PROBLEMA DEL 8-PUZZLE



PROBLEMAS

Problemas de los métodos avariciosos con la Función Heurística:

- Máximos (o mínimos) locales: valor que es más alto (o más bajo) que cada uno de sus estados vecinos, pero más bajo (alto) que el máximo (mínimo) global
- Mesetas: zona del espacio de estados con función plana
- Crestas: zona del espacio de estados con varios máximos (mínimos) locales

Soluciones

- Retroceso
- Dar más de un paso
- Reinicio aleatorio

PROPIEDADES

Completitud: no tiene porqué encontrar la solución

Optimalidad: no siendo completo, tampoco se puede garantizar que sea óptimo

Eficiencia: rápido y útil si la función es monótona (de)creciente

ALGORITMO DE PRIMERO EL MEJOR

Idea general
Algoritmo
Cálculo de Rutas
Propiedades
Consideraciones de la Heurística

IDEA GENERAL

Elegir a cada momento el mejor nodo entre todos los sucesores que hayan sido generados y no explorados

Búsqueda global, reconsidera los caminos

Se necesita una función de evaluación $f(n)$

que indique el mejor nodo

ALGORITMO

Crear árbol de búsqueda A, con el nodo inicial, I (Estado-inicial)

ABIERTA=I, CERRADA=Vacío, EXITO=Falso

Hasta que ABIERTA esté Vacía O EXITO

- Quitar el primer nodo de ABIERTA, N y meterlo en CERRADA
- SI N es Estado-final ENTONCES EXITO=Verdadero
- SI NO Expandir N, generando el conjunto S de sucesores de N, que no son antecesores de N en el grafo
- Generar un nodo en A por cada s de S
- Establecer un puntero a N desde aquellos s de S que no estuvieran ya en A
- Añadirlos a ABIERTA
- Reordenar ABIERTA según f (n)
- Para cada s de S que estuviera ya en ABIERTA o CERRADA decidir nodo eliminar

Si EXITO Entonces Solución=camino desde I a N a través de los punteros de A

Si no Solución=Fracaso

ALGORITMO A* (HART,NILSSON Y RAFAEL, 1968)

Función de ordenación de nodos

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

- f (n): función de evaluación
- g(n): función de coste de ir desde el nodo inicial al nodo n
- h(n): función heurística que mide la distancia estimada desde n nodo meta

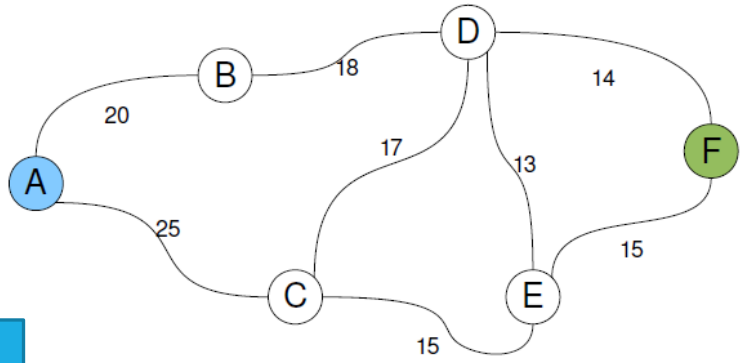
g(n) se calcula como la suma de los costes de los arcos recorridos, $k(n_i ; n_j)$

Los valores reales sólo se pueden conocer al final de la búsqueda

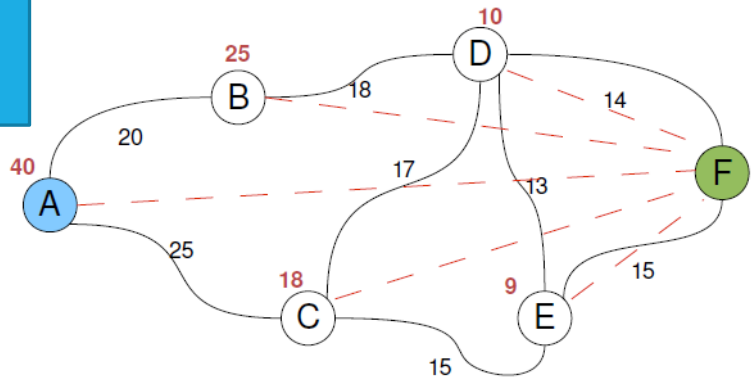
- f*(n): coste real para ir desde el nodo inicial a meta a través de n
- g*(n): coste real para ir desde el nodo inicial al nodo n
- h*(n): coste real para ir desde el nodo n a meta

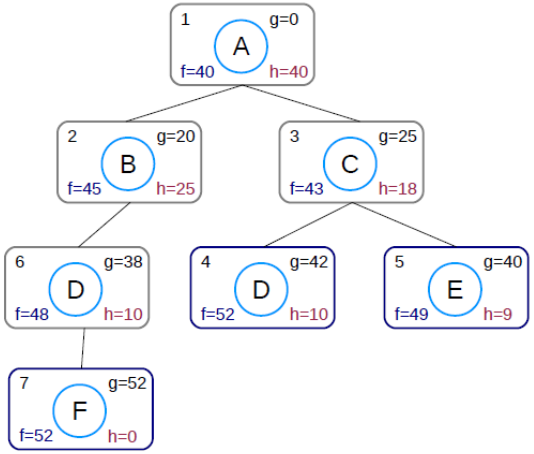
PROBLEMA DE RUTAS. SE QUIERE DETERMINAR EL CAMINO MÁS CORTO DE LA CIUDAD A A LA CIUDAD F

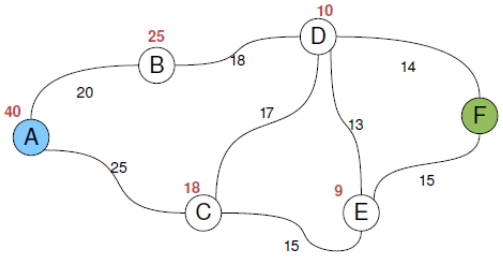
COSTE



ESTIMACIÓN DE LA DISTANCIA A LA META







Se continúa, pues el algoritmo termina cuando se expande el nodo que contiene la solución (nodo 7).

SOLUCIÓN

PROPIEDADES

Completitud: si existe solución, la encuentra

Optimalidad: si hay una solución óptima, la encuentra si:

- el número de sucesores es finito para cada nodo
- $k(n_i ; n_j) > \partial > 0$ en cada arco
- La función h(n) es admisible

SOBRE LA HEURÍSTICA

Si $h1(n) \leq h2(n)$ para todo n, h2(n) está más informada que h1(n) y servirá para expandir menos nodos

Ejemplo: distancia de Manhattan está más informada que número de casillas mal colocadas (problema de Manhattan es menos relajado que el del número de casillas)

Extremos

- h(n) = 0 para cada nodo: no se tiene información (Dijkstra)
- h(n) = h*(n) para cada nodo: se tiene información perfecta

No tiene sentido dedicar más coste computacional a calcular una buena h(n) que a realizar la búsqueda equivalente: equilibrio

RESUMEN MÉTODOS

Búsqueda no informada

- Amplitud: f (n) = profundidad(n)
- Dijkstra: f (n) = g(n)

Búsqueda informada o heurística

- A* [Hart et. al., 1968]
- Otras:
  - Búsqueda mejor primero avara (GBFS): f (n) = h(n)
  - IDA* [Korf, 1985]: f (n) = g(n) + h(n)
  - A* ponderado [Polh, 1970]

A*

https://www.youtube.com/watch?v=huJEgJ82360

Concepto

Minimax

Heurística

BÚSQUEDA DE DOS AGENTES

CONCEPTO

Suma nula: lo que gana uno, lo pierde el otro

Dos agentes (se puede generalizar)

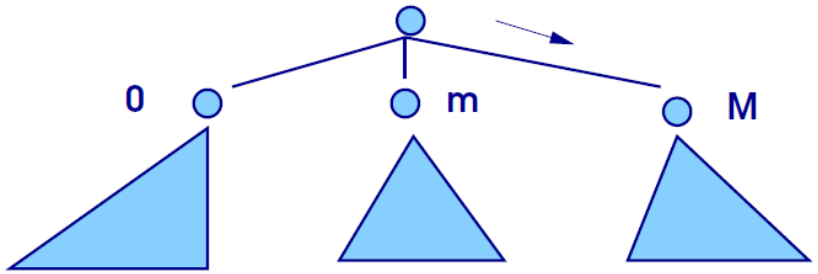
Información completa: se conoce en cada momento el estado completo del juego

Deterministas: no entra en juego el azar (se puede generalizar)

Alternados: las decisiones de cada agente se toman de forma alternada

TEOREMA MINIMAX

En cualquier juego se suma nula, o bien un jugador puede forzar una victoria, o cualquiera de los jugadores puede forzar un empate



HEURÍSTICA

MINIMAX es intratable; no se puede realizar en un tiempo razonable

Solución:

- Utilizar una función de evaluación que valore el estado actual
- Calcular las situaciones hasta una profundidad máxima

Elegir en cada momento la mejor opción según f(n) hasta una profundidad máxima, considerando que:

- Un jugador intenta maximizar f (n) (Jugador MAX)
- Un jugador intenta minimizar f (n) (Jugador MIN)

Ejemplos de Funciones de Evaluación:

- 3 en Raya: El número de posibles 3 en raya de "X" menos el número de posibles 3 en raya de "O".
- Ajedrez: Suma ponderada de las piezas del tablero. Valor piezas blancas menos valor piezas negras (9 por la reina + 5 por las torres + 3 por los alfiles + 3 por los caballos + el número de peones)