Tema 2:
Resolución de Problemas con Técnicas de Búsqueda

Carlos García Martínez. Dept. IAN

Tema 2: Búsqueda

- Ejemplo
- Condiciones necesarias
- Búsqueda ciega
- Búsqueda heurística
- Otros problemas
- Comentarios Finales

Problema del Enrutado Web

Google Maps

Condiciones Necesarias

- Modelo de representación de los estados (S)
- Estados iniciales (s0)
- Representación de acciones (A: s ->s')
- Estados / Función objetivo (s* / O: s -> T|F)
- Función de coste (F: {a1, ..., aN} -> R)
- Función mejor padre conocido (P: s -> s')
- **Objetivo**: Encontrar un camino {a1, ..., aN} que conecte el estado inicial con el estado final

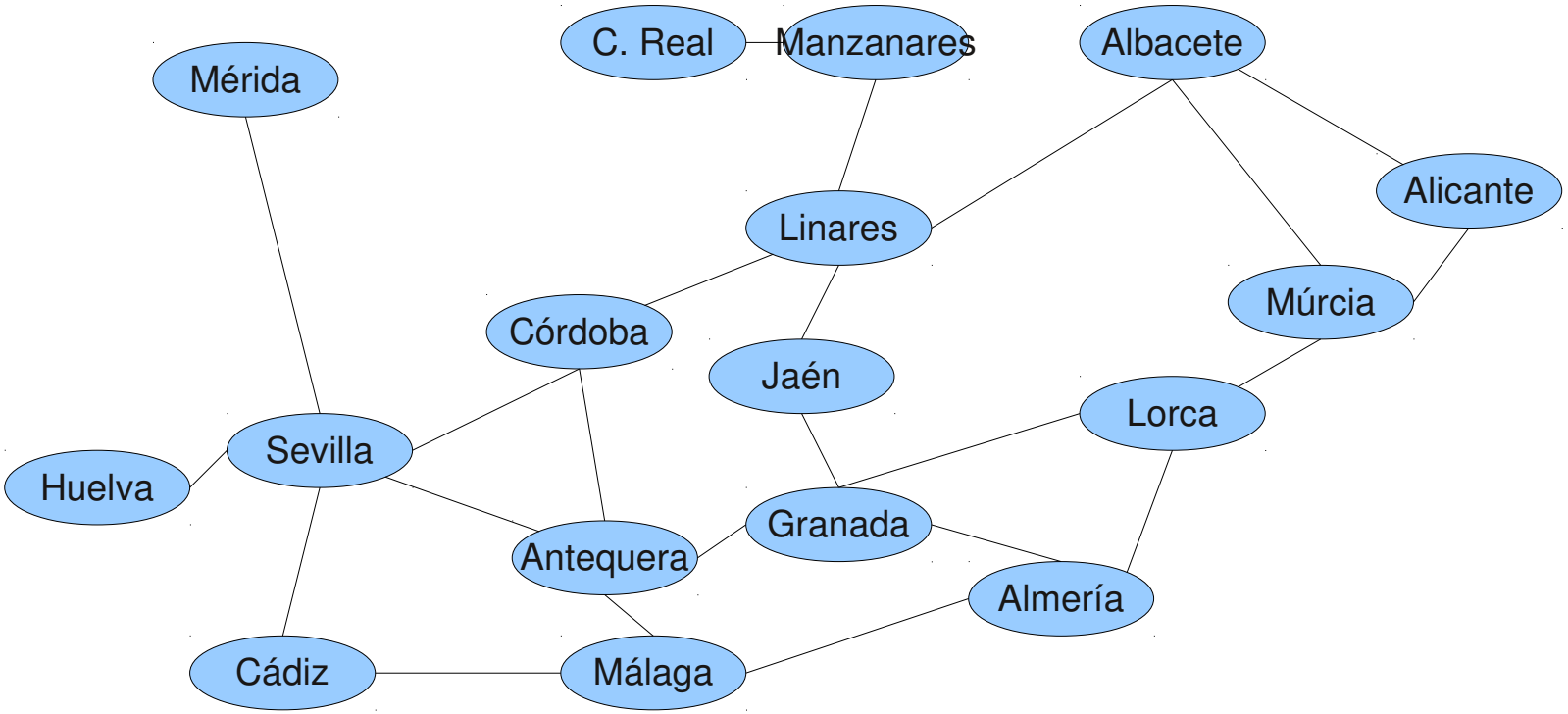
Representación

Google Maps

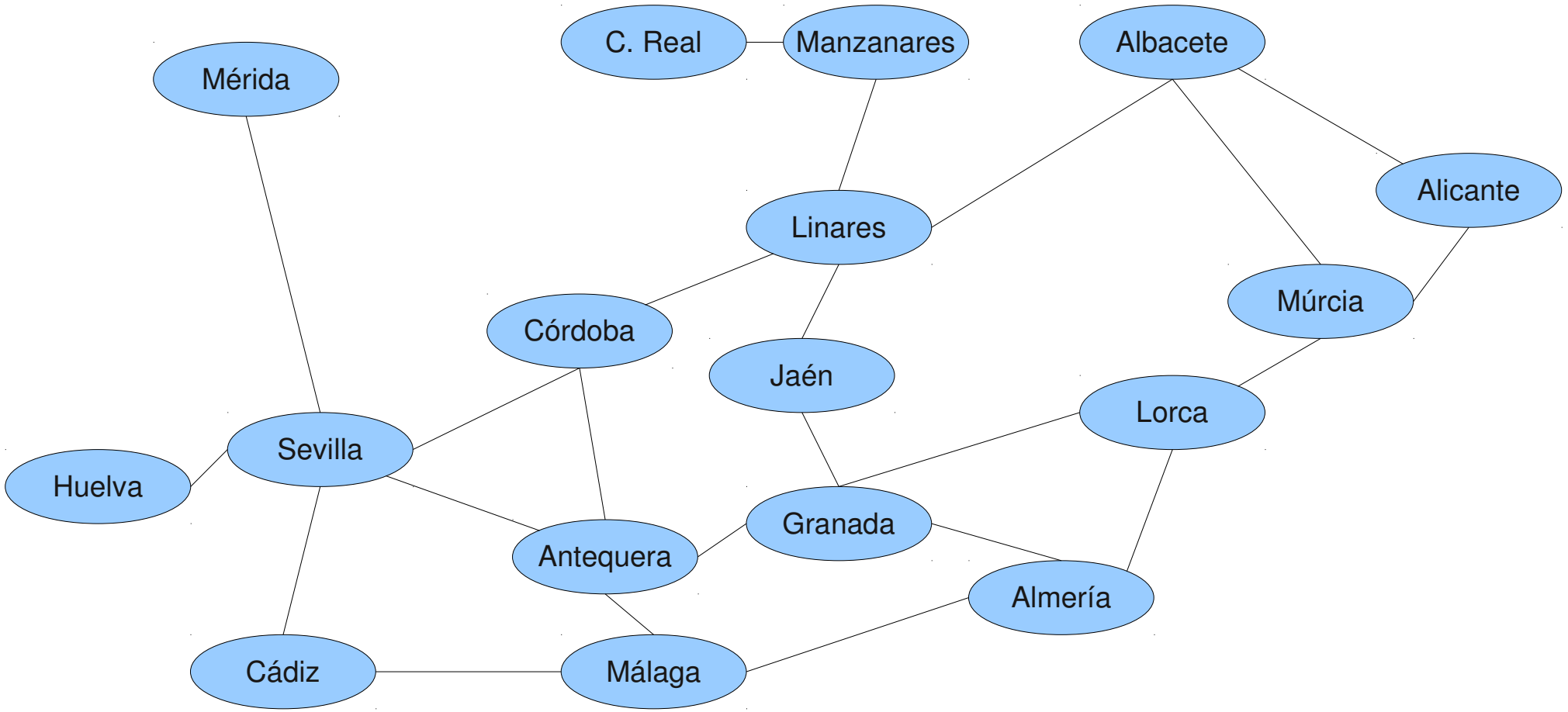
Representación

Google Maps

Representación



Representación



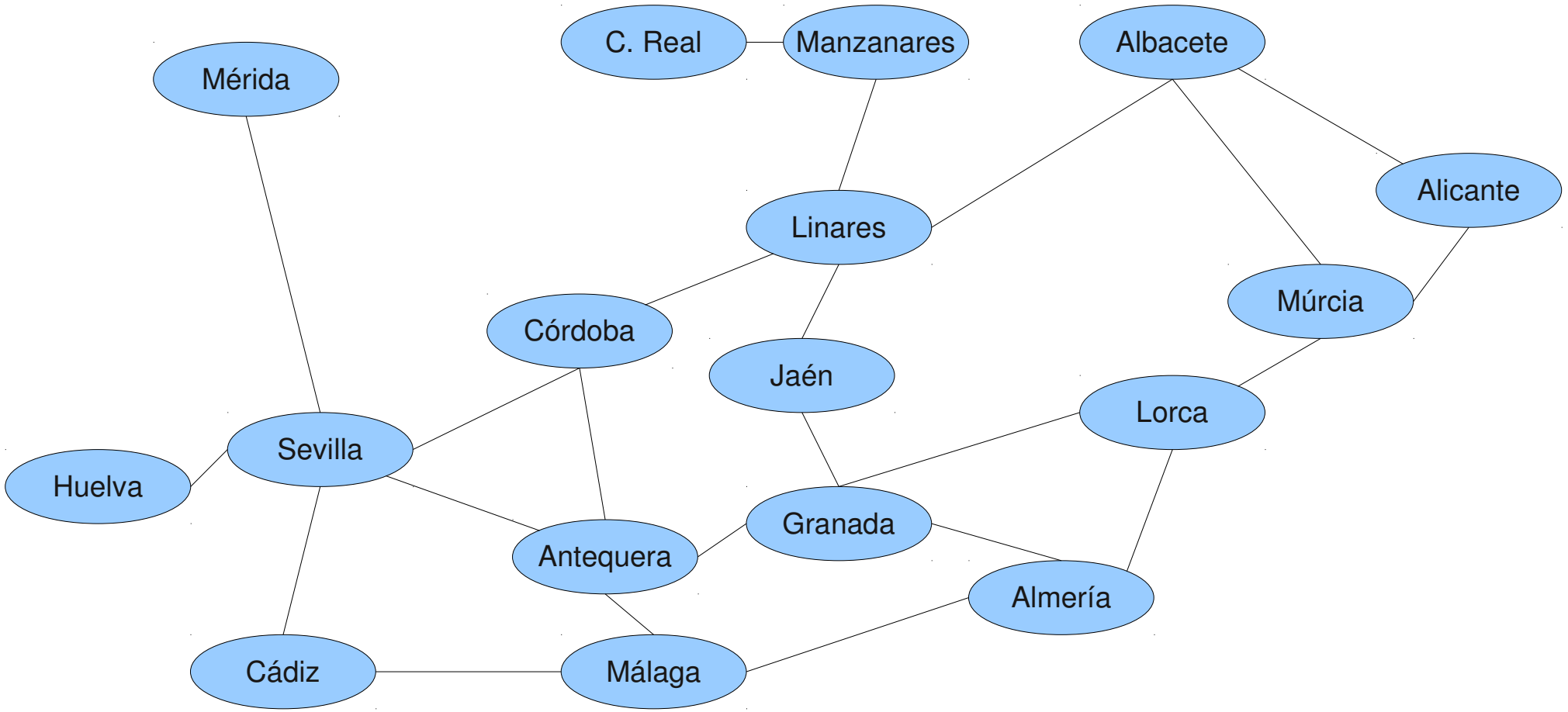
Búsqueda sobre árboles

1) Frontera = {s0}
2) Mientras la Frontera no esté vacía
    1) actual = extraer_estado(Frontera);
    2) Si O(actual), devolver camino;
    3) Para cada posible acción ai desde actual
        1) Añadir resultado ai(actual) en Frontera;
3) Devolver Fallo

Búsqueda en Amplitud

- La frontera se implementa como una ***cola FIFO***
- Se selecciona el estado que antes se introdujo en la frontera
- Se selecciona uno de los estados correspondiente al camino de menor ***"longitud"***

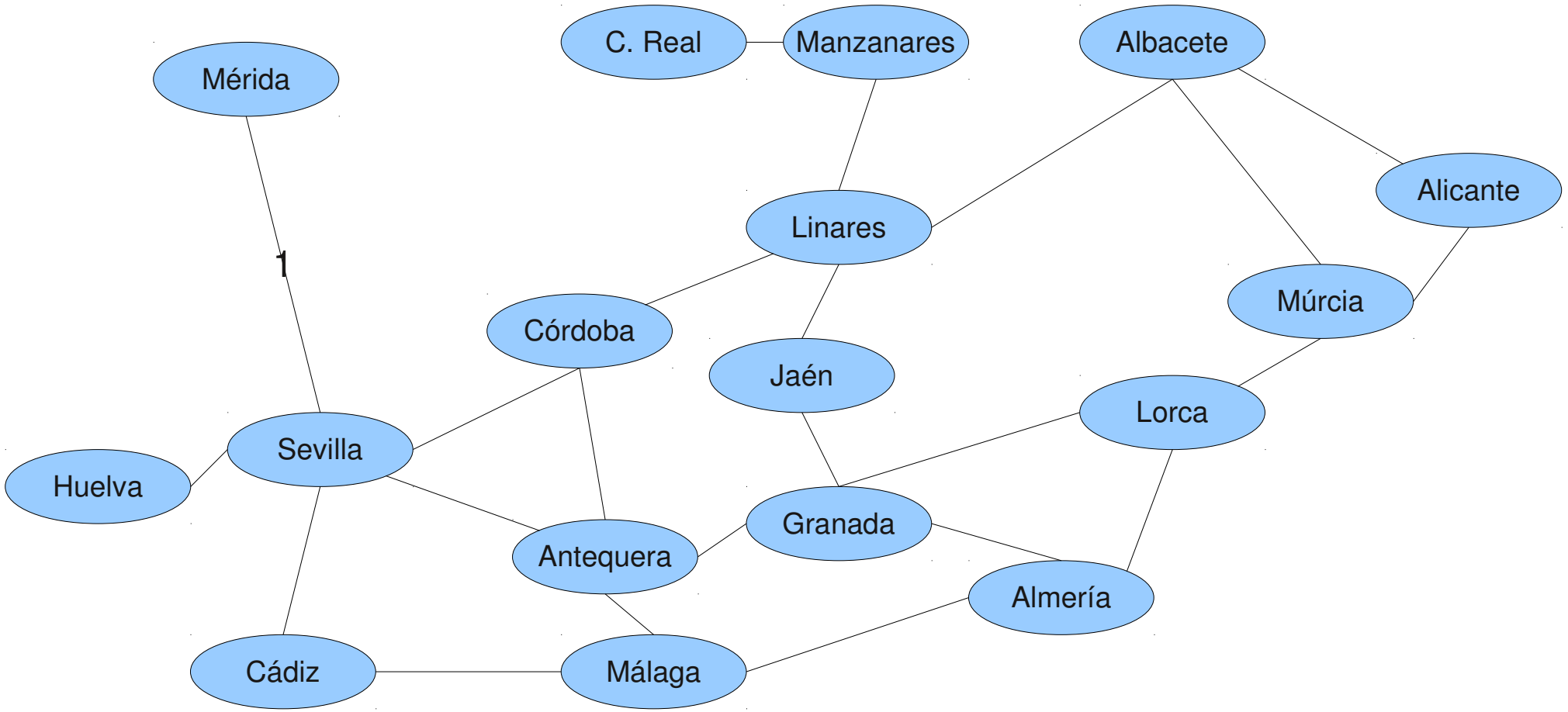
Búsqueda en Amplitud



Búsqueda sobre grafos

1) Frontera = {s0}; Explorados = {};
2) Mientras la Frontera no esté vacía
    1) actual = extraer_estado(Frontera);
    2) Añadir actual a Explorados;
    3) Si O(actual), devolver camino;
    4) Para cada posible acción ai desde actual
        1) Si ai(actual) está en explorados, actualizar recursivamente coste, padre y descendientes
        2) Si no, si está en Frontera, actualizar coste y padre
        3) Si no, añadir ai(actual) a Frontera;
3) Devolver Fallo

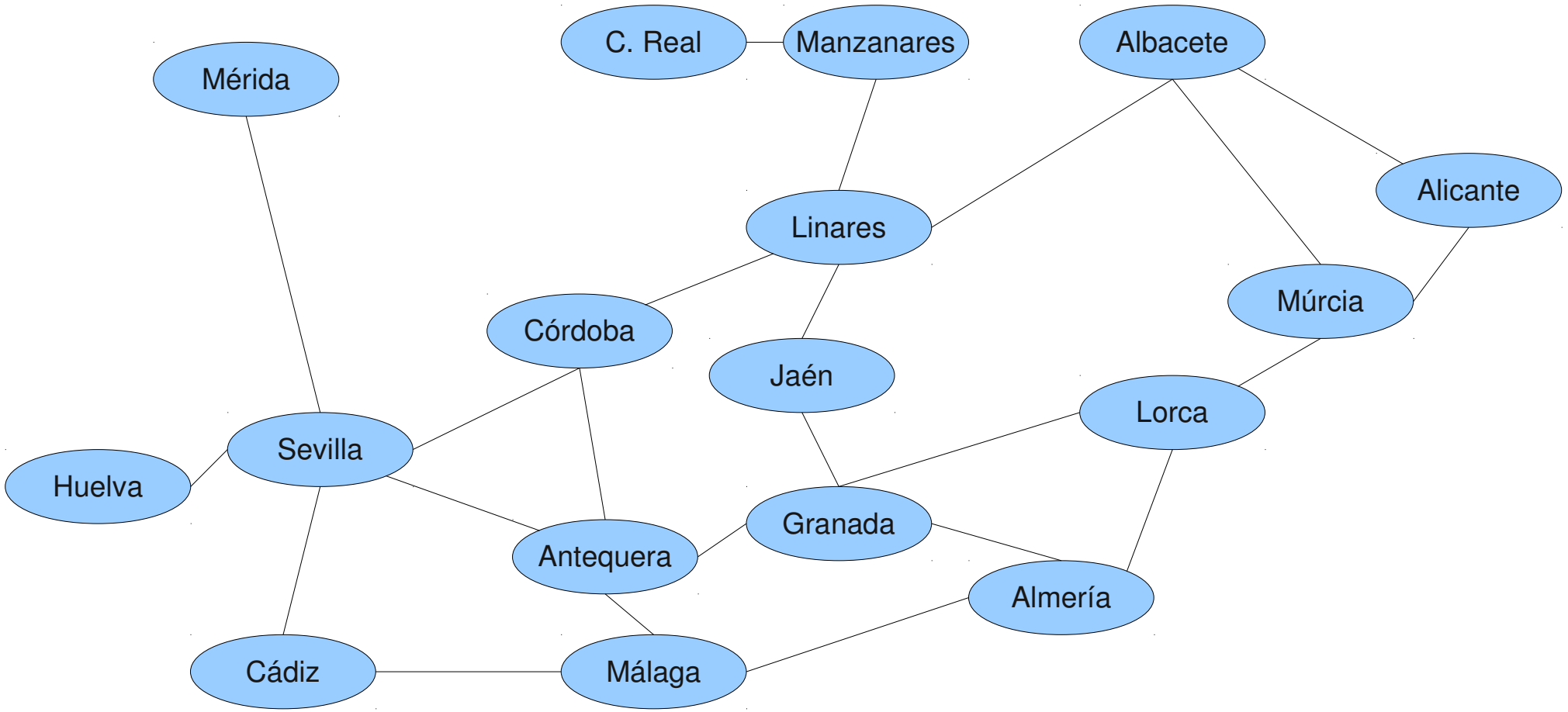
Búsqueda en Amplitud



Propiedades de la Búsqueda en Amplitud

- Ventajas:
  - Método *completo* y *óptimo*
- Desventajas
  - Gran complejidad temporal: $O(n^p)$
  - Gran complejidad espacial: $O(n^p)$

Búsqueda en Profundidad: Pila



Propiedades de la Búsqueda en Profundidad

- Ventajas:
  - Complejidad espacial reducida: $O(n \cdot p)$
- Desventajas:
  - *No es óptimo*
  - *No es completo* (puede perderse en una rama infinita sin solución).
  - Gran complejidad temporal: $O(n^p)$

Búsqueda con retroceso

- Es una búsqueda en profundidad pero generando sólo un nodo hijo de cada nodo m (al volver a m se genera otro nodo distinto)
- Ventajas:
  - Complejidad espacial muy reducida: O(n)
- Desventajas:
  - No es óptimo
  - No es completo (puede perderse en una rama infinita sin solución).
  - Gran complejidad temporal: $O(n^p)$

Búsqueda en profundidad limitada

- Igual que búsqueda en profundidad, sólo que se vuelve al alcanzar una profundidad máxima, en vez de un nodo hoja.
- Ventajas:
  - Evita quedar atrapado en una rama infinita.
  - Complejidad espacial: $O(n \cdot p)$
- Inconvenientes:
  - No es completo.
  - No es óptimo.
  - Gran complejidad temporal: $O(n^p)$

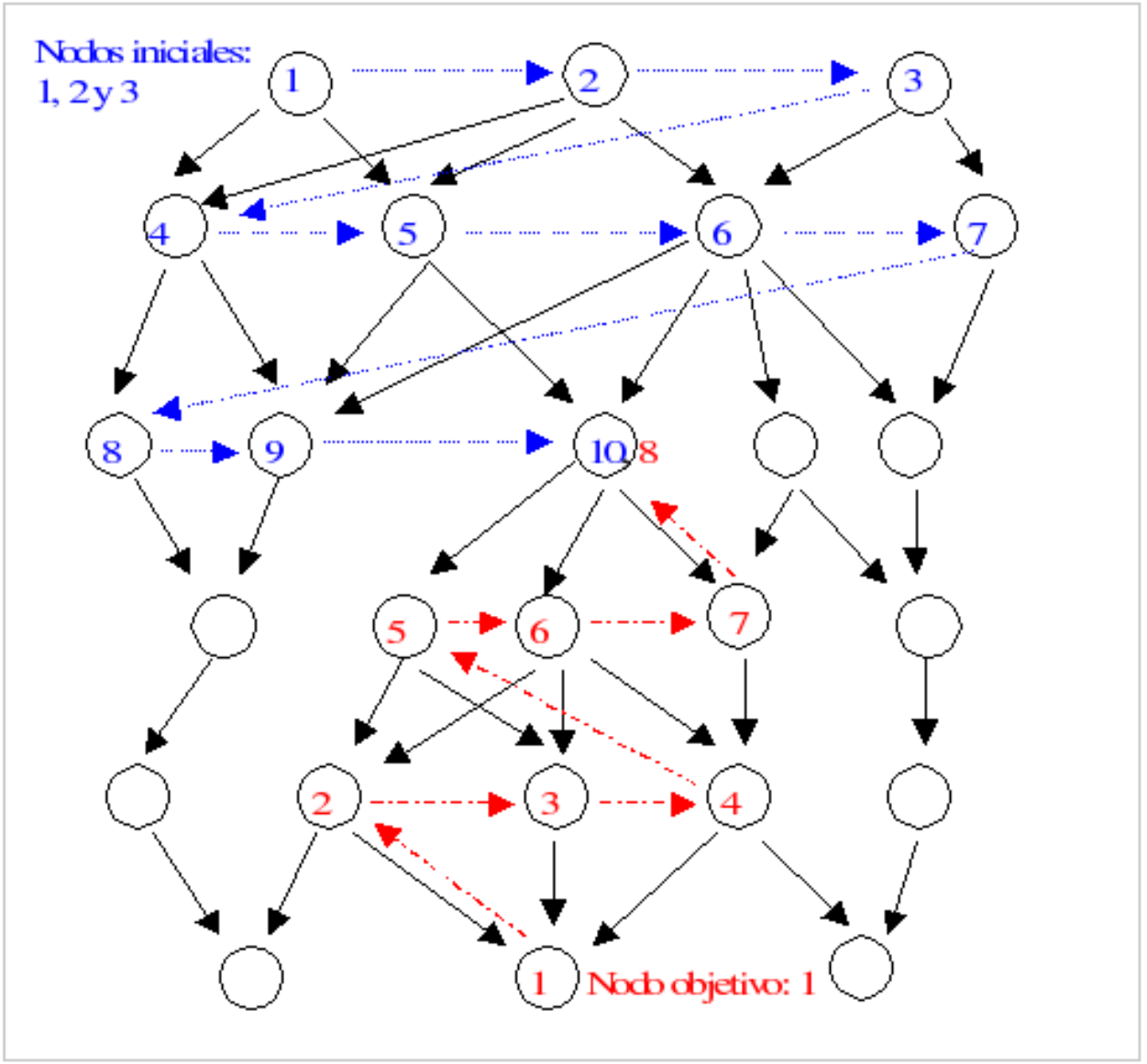
Búsqueda en profundidad iterativa

- Aplicar el método de búsqueda en profundidad limitada, aumentando progresivamente la profundidad máxima si no se encuentra la solución.
- Ventajas:
  - Evita quedar atrapado en una rama infinita.
  - Método completo y óptimo
  - Complejidad espacial: $O(n \cdot p)$
- Inconvenientes:
  - Gran complejidad temporal: $O(n^p)$

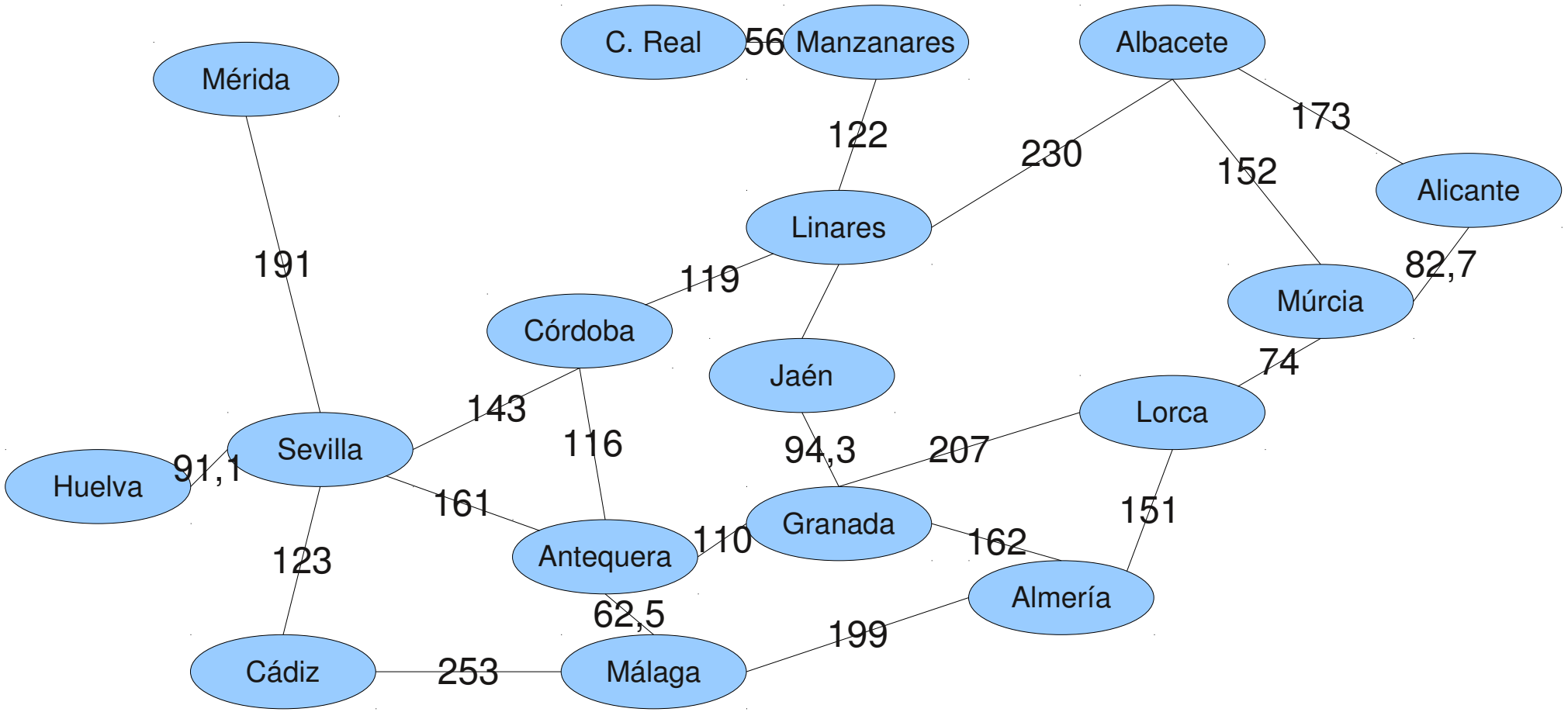
Búsqueda bidireccional

- Aplicar un método de búsqueda sobre los estados inicial y final y parar cuando llegamos a un nodo común a ambos procesos. Una de ellas debiera ser una búsqueda en amplitud.
- Necesidad de que las operaciones puedan ser *reversibles*.
- Ventajas
  - Reduce la complejidad temporal y la espacial: $O(n^{p/2})$

Ejemplo: búsqueda bidireccional

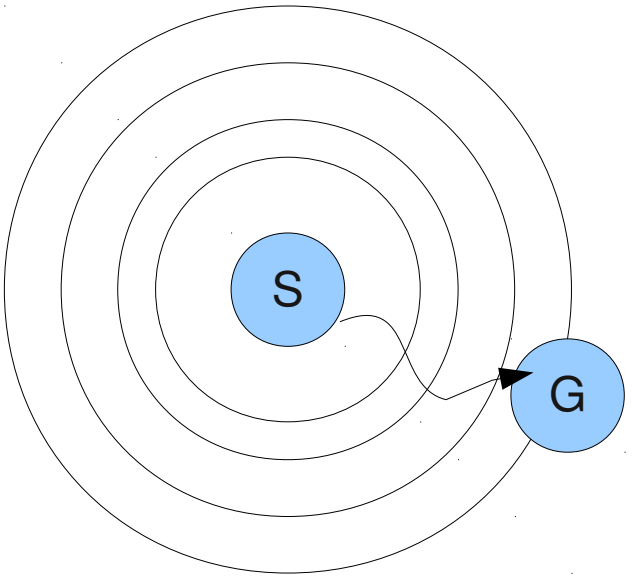


Búsqueda de Coste Uniforme



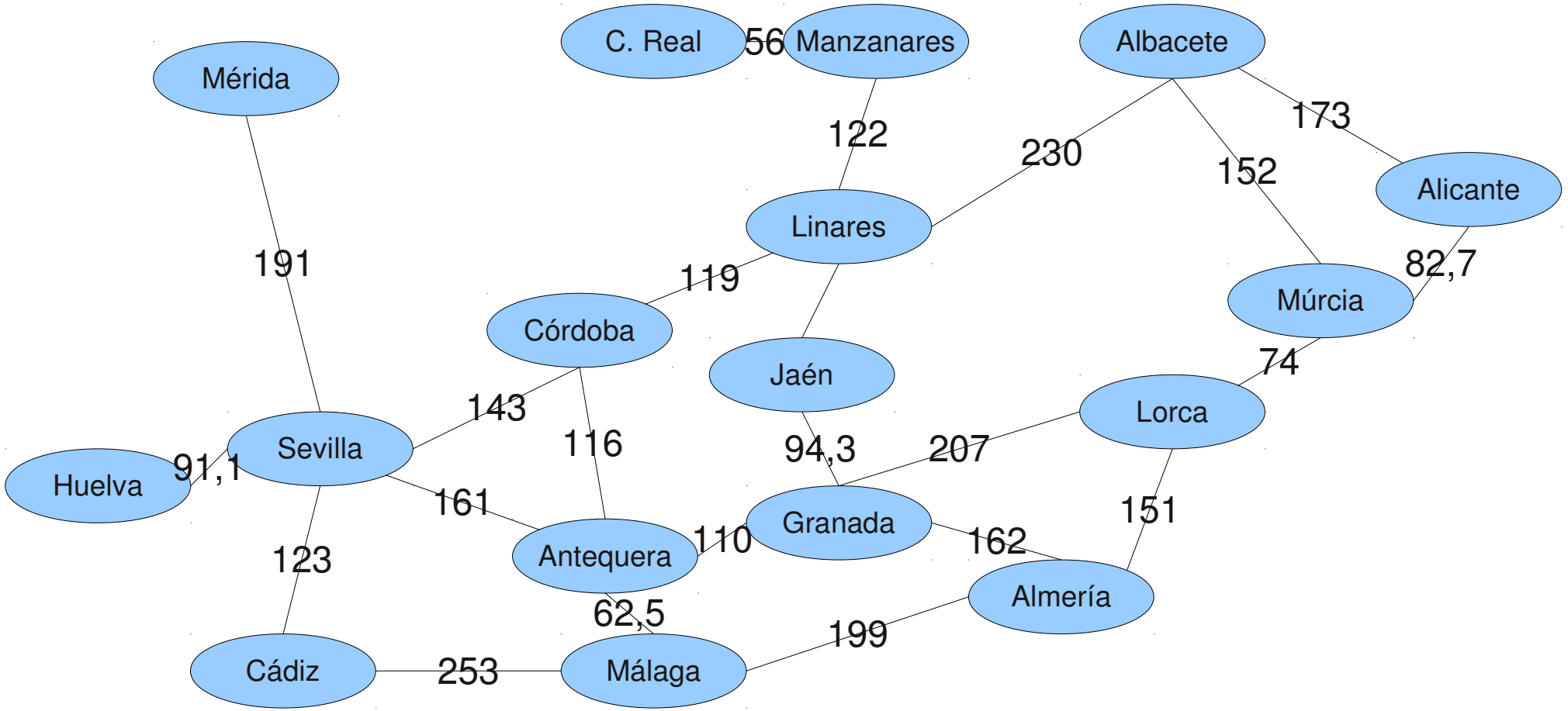
Búsqueda de Coste Uniforme

- La búsqueda empieza en un estado
- Se examina el siguiente nodo más cercano al nodo inicial
- La búsqueda no está dirigida, es ciega.



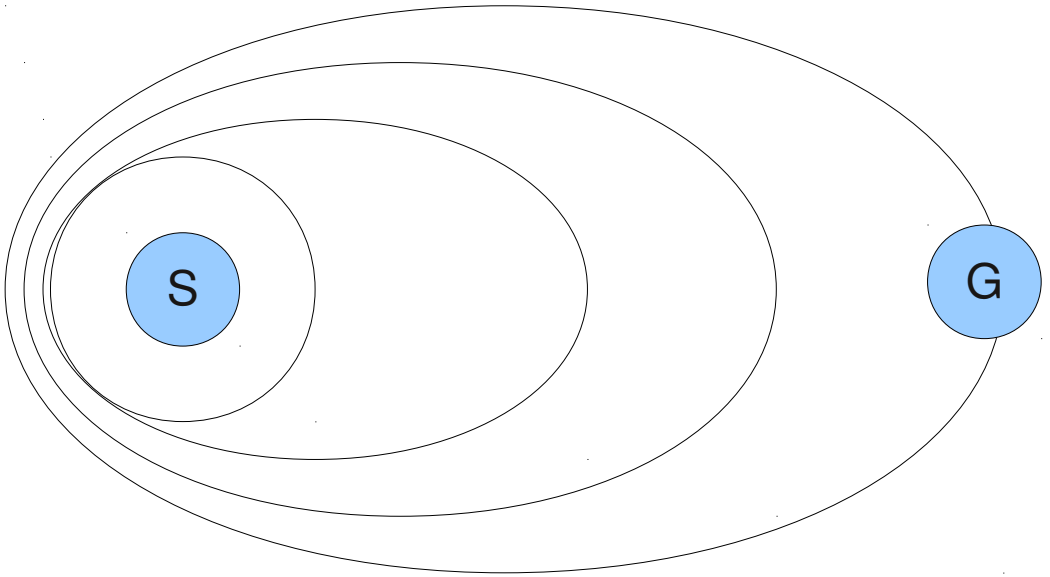
Búsqueda heurística

- **Objetivo**: Añadir conocimiento para dirigir el proceso de búsqueda

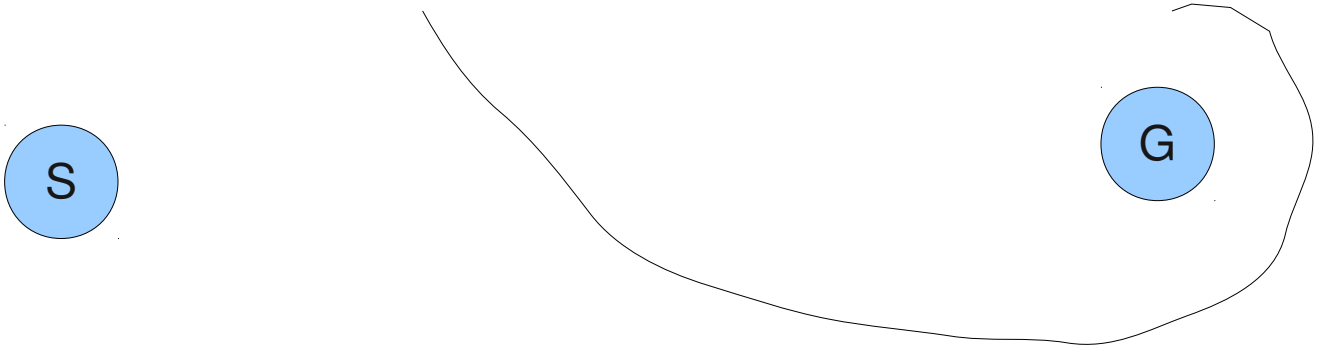


Búsqueda Primero el Mejor

- Se escoge el nodo que se cree que está más cerca del objetivo

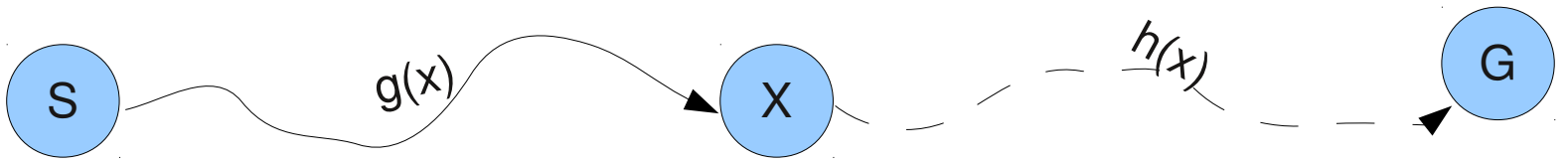


Dificultades de Primero el Mejor

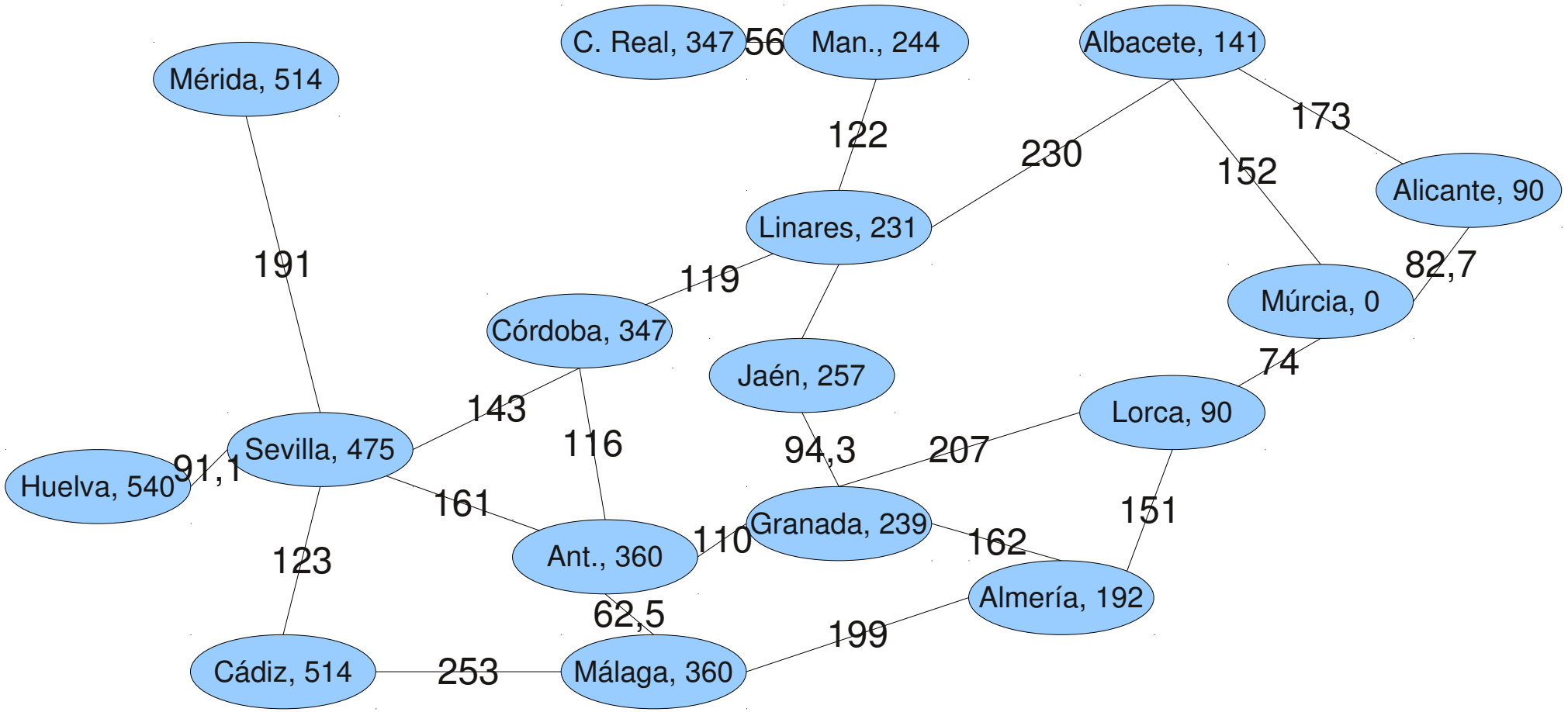


Búsqueda A*

- Se escoge el nodo que minimiza la función: $f(x) = g(x) + h(x)$
- $g(x)$ es el coste del camino desde el nodo inicial al nodo x
- $h(x)$ es el valor heurístico del nodo x, que representa una estimación hasta el nodo final



Búsqueda A*


C. Real, 347
56
Man., 244
Albacete, 141
Mérida, 514
122
173
230
152
Alicante, 90
Linares, 231
191
119
82,7
Múrcia, 0
Córdoba, 347
Jaén, 257
74
143
Lorca, 90
Sevilla, 475
116
94,3
207
91,1
Huelva, 540
161
151
Granada, 239
110
162
123
Ant., 360
Almería, 192
62,5
199
Cádiz, 514
253
Málaga, 360

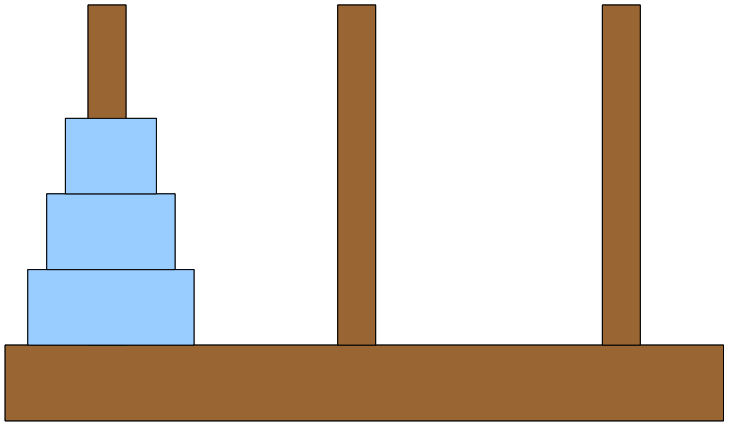
Propiedades de A*

- La elección del nodo no depende únicamente del valor heurístico
- Se elige el nodo que parece que ofrecerá la solución con mejor coste total
- A* es óptimo si h(x) <= h'(x)

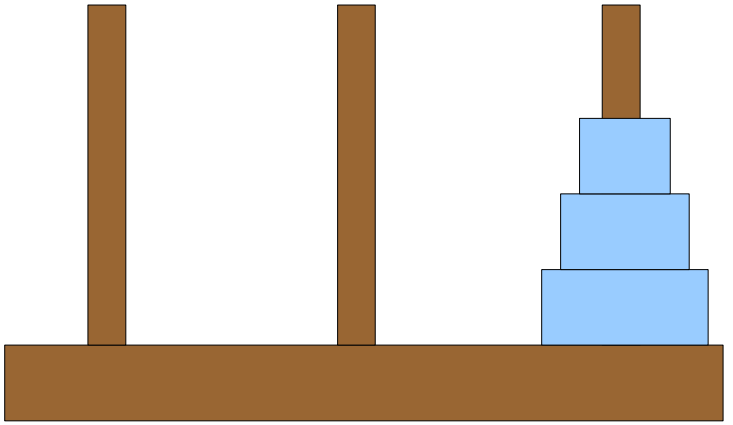
Otros Problemas

- Sistemas de Producción:
  - Un modelo de representación
  - Un conjunto de *reglas*:
    - Antecedente ⇨ Consecuente
  - Una o más *bases de datos* con:
    - Datos permanentes o datos de la situación actual
  - Una *estrategia de control*.
  - Un *agente* que aplique las reglas.

Las torres de Hanoi



Estado inicial

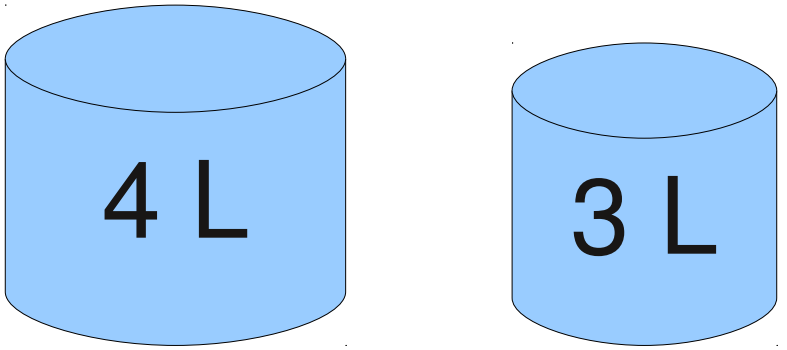


Estado final

Trabajo: Programa que resuelva las torres de Hanoi con un número *indeterminado* de discos utilizando los conceptos de este tema

Las jarras de agua

- Se tienen dos jarras, una de cuatro litros de capacidad y otra de tres. Se desea tener exactamente dos litros de aguan en la jarra de cuatro litros.



Operadores para las jarras de agua

	Estado actual		Estado nuevo	Descripción
1.-	(x,y) si $x < 4$	→	(4,y)	Llenar la jarra de 4 litros.
2.-	(x,y) si $y < 3$	→	(x,3)	Llenar la jarra de 3 litros.
3.-	(x,y) si $x > 0$	→	(x-d,y)	Vaciar un poco la jarra de 4 litros.
4.-	(x,y) si $y > 0$	→	(x,y-d)	Vaciar un poco la jarra de 3 litros.
5.-	(x,y) si $x > 0$	→	(0,y)	Vaciar completamente la jarra de 4 litros.
6.-	(x,y) si $y > 0$	→	(x,0)	Vaciar completamente la jarra de 3 litros.
7.-	(x,y) si $x + y \geq 4$ e $y > 0$	→	(4,y-(4-x))	Verter agua desde la jarra de 3 litros a la jarra de 4 litros hasta que ésta esté llena.
8.-	(x,y) si $x + y \geq 3$ y $x > 4$	→	(x-(3-y),3)	Verter agua desde la jarra de 4 litros a la jarra de 3 litros hasta que ésta esté llena.
9.-	(x,y) si $x + y \leq 4$ e $y > 0$	→	(x+y,0)	Verter por completo el agua de la jarra de 3 litros en la jarra de 4 litros.
10.-	(x,y) si $x + y \leq 3$ y $x > 0$	→	(0,x+y)	Verter por completo el agua de la jarra de 4 litros en la jarra de 3 litros.
11.-	(0,2)	→	(2,0)	Verter 2 litros de la jarra de 3 litros en la jarra de 4 litros.
12.-	(x,2)	→	(0,2)	Vaciar completamente la jarra de 4 litros en el suelo.

Generación de Heurísticas

- Se pueden generar heurísticas relajando las leyes del problema
- 8-Puzzle:
  - Un ficha A puede moverse a la posición B si:
    - A y B son adyacentes
    - B está vacía

6	1	3
4	5	7
2	8	

Marco de la IA

Niveles de diferentes problemas de la IA

Percepción, razonamiento, aprendizaje, planificación y decisión

▪ ▪ ▪

clasificación, representación y ***búsqueda***

▪ ▪ ▪

¿Cuando hay que utilizar la búsqueda?

- La búsqueda es un mecanismo general que puede utilizarse cuando no se conoce otro método más directo (*algorítmico*).
- Al mismo tiempo, proporciona un marco donde pueden empotrarse métodos más directos de resolución de partes del problema

Tipología de los procesos de búsqueda

- ***Polaridad***:
  - Unipersonales
  - Bipersonales (cooperativos, competitivos)
- ***Objetivo***:
  - Búsqueda completa.
  - Búsqueda satisfactoria.
  - Búsqueda óptima.
  - Búsqueda adaptativa.
- ***Método***:
  - Búsqueda a ciegas
  - Búsqueda heurística

Características Necesarias

- El Universo debe ser
  - Completamente observable
  - Conocido
  - Discreto
  - Determinístico
  - Estático

Algunos aspectos sobre el diseño de programas de búsqueda

- Se trata de un *recorrido* sobre el grafo *espacio de estados*.
- Se debe elegir una forma de representar los nodos y las transiciones.
- Se debe elegir la forma de seleccionar las reglas a aplicar.
- No se suele construir el árbol o grafo explícitamente. El mejor método necesitará hacer explícita la menor parte del grafo implícito.
- Reducción de la búsqueda:
  - Antes de generar los sucesores de un nodo se puede saber si ese nodo conducen o no a la solución y abandonar la búsqueda a tiempo.
  - No examinar un nodo más de una vez