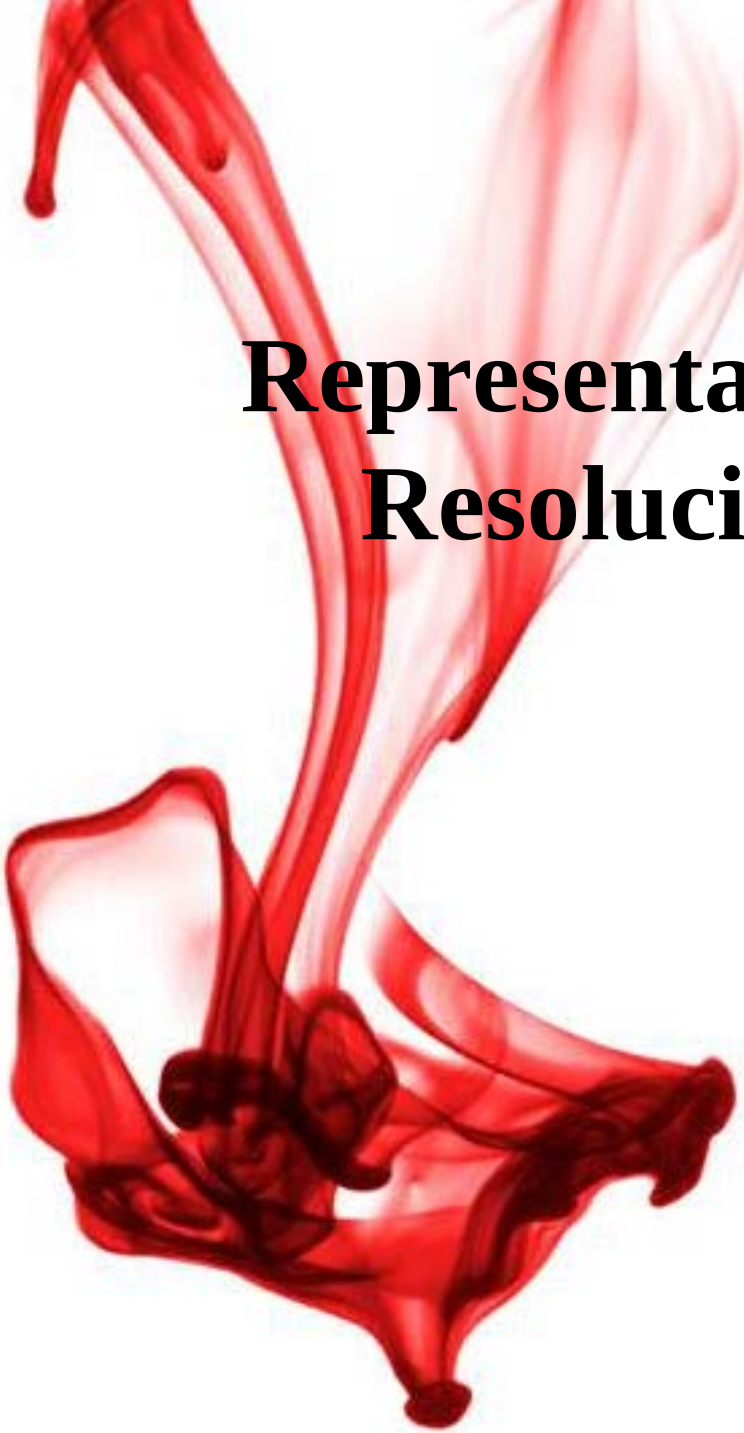


Representación y Control en la Resolución de Problemas.

IV Parte

Mario Hernández

BÚSQUEDA INFORMADA

Prefacio

Las Búsquedas Ciegas o No Informadas presentan problemas de ineficacia en la práctica por la Explosión Combinatoria

¿Cómo realizar búsquedas reduciendo los costos computacionales e intentando asegurar una cierta completitud en la solución?

Reduciendo el número de nodos a visitar del árbol expandido utilizando para ello CONOCIMIENTO ESPECÍFICO expresado como PRINCIPIOS HEURÍSTICOS

Heurísticas

Criterios, métodos o principios para decidir cuál de entre diversos cursos de acción alternativos "promete" ser el más efectivo para alcanzar algún objetivo

Representan compromisos entre dos requerimientos:

1. Las necesidad de hacer que esos criterios, métodos o principios sean sencillos
2. El deseo de que discrimine correctamente entre elecciones buenas y malas

Heurísticas

Objetivo: guiar el proceso de búsqueda visitando algunos nodos seleccionados según las siguientes estrategias:

- Exploración Ordenada: ordenar las preferencias sobre los nodos en base a expectativas heurísticas de éxito, explorando primero los caminos más prometedores
- Poda: eliminación de ciertas ramas heurísticamente poco adecuadas para la exploración

Características

1. Se basa en la experiencia y el sentido común
2. Es un principio, más o menos general, y aproximadamente correcto pero que no se ha demostrado científicamente que sea preciso
3. No garantiza que se vaya a encontrar la solución
4. Si la encuentra no asegura que sea óptima
5. En casos puede encontrar una buena solución en tiempo aceptable

(-) Pierden completitud y/o optimalidad

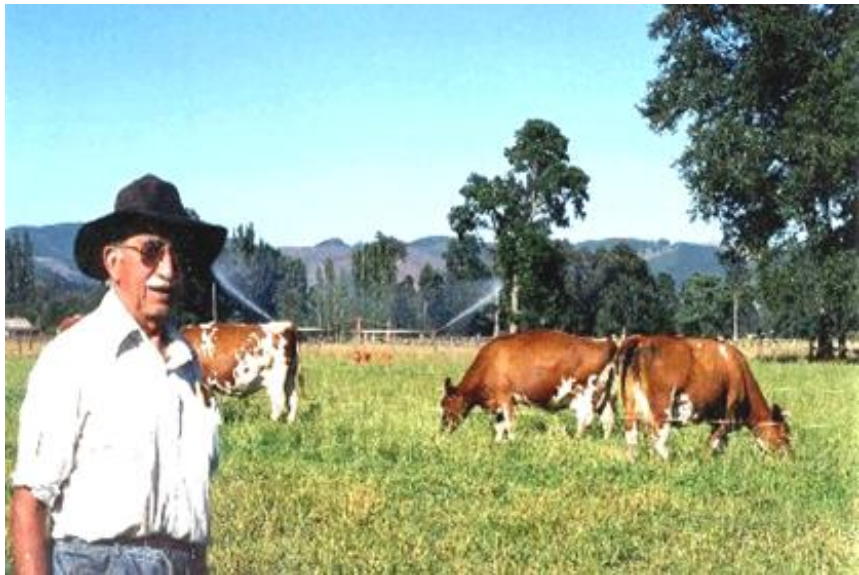
(+) Aumentan eficiencia

Ejemplo 1: Agricultor

Una heurística puede ser una regla derivada de la experiencia (rule of thumb)

La predicción meteorológica del viejo agricultor que observa la naturaleza frente a la predicción científica que realiza el meteorólogo

"Meteorología Popular"

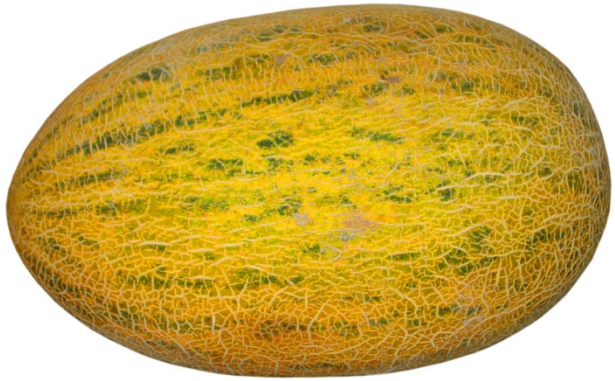


Ejemplo 2: Elección de Melones

Método popular para elegir los melones en el mercado → presionar el botón por donde el melon estuvo unido a la planta:

Si no esta rígido y si al olerlo huele a melón, entonces lo escogemos como maduro para abrirlo y comerlo.

Ojo: no se asegura que el melón escogido sea el que este maduro en su punto siempre, pero es bastante eficaz frente a otras estrategias más complejas



Ejemplo 3: Gran Maestro de Ajedrez

- Un gran maestro se enfrenta con la elección entre varias posibilidades de movimiento al decidir jugada
- Puede decidir que un movimiento en particular es más efectivo porque su resultado es una disposición de tablero tal que le deja en situación que "aparentemente" es más fuerte que la de otros movimientos
- *El criterio de "aparentemente" más fuerte es más sencillo de aplicar para el gran maestro que, por ejemplo, determinar rigurosamente que movimientos provocan jaque-mate*
- El hecho de que el gran maestro no siempre gane indica que su heurística no siempre garantiza que se ha seleccionado el movimiento más efectivo.
- *Si se le pregunta al gran maestro que describa su heurística, solo podrá dar una descripción rudimentaria y parcial de lo que el parece aplicar sin esfuerzo*

Funciones de Evaluación Heurísticas en las Estrategias de Búsqueda

- Idea: Aglutinan el conocimiento acerca del dominio sobre el que se apoyará la decisión

- Fundamento: asociar a cada nodo ***e*** ligado a cada estado del problema un número *h(**e**)*, que indica lo prometedor, o no, que es ese nodo ***e*** de cara a alcanzar el estado objetivo óptimo

Interpretaciones

1. Estima la “calidad” del estado E

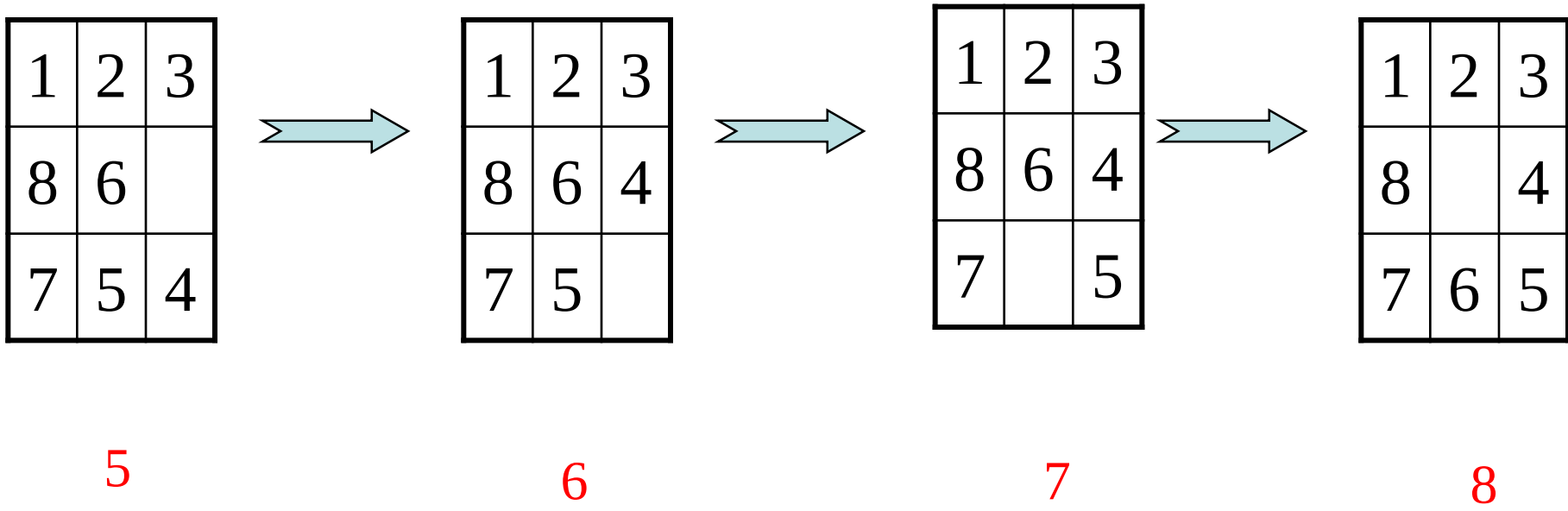
   Objetivo de las estrategias heurísticas: buscar los nodos con mayor calidad, es decir con mayor valor heurístico

2. Estima la “proximidad” a un estado final

   Objetivo de las estrategias heurísticas: buscar los nodos menos discrepantes, es decir menos distantes o más cercanos con el estado final, o de otra manera, con menor valor heurístico

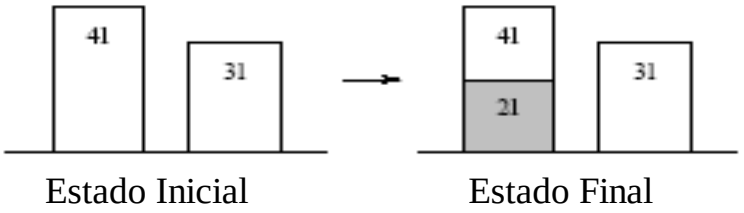
Ejemplo 1

- 8-puzzle: número de casillas bien colocadas (*de calidad*)

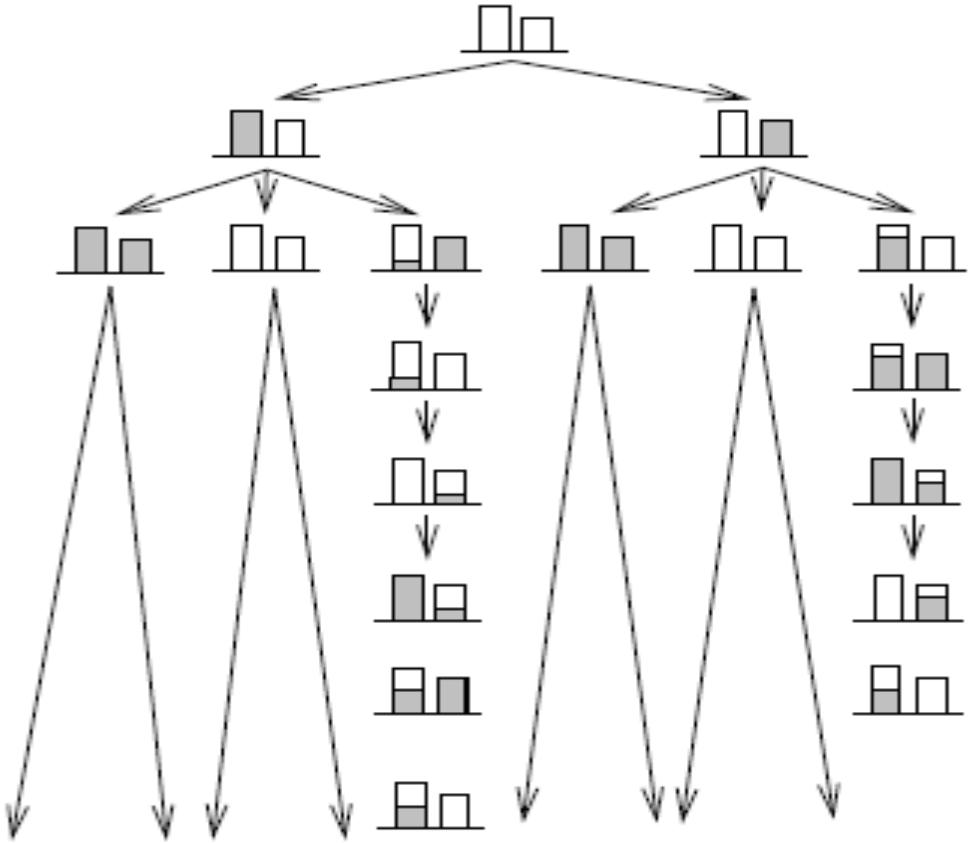


Ejemplo 2

- Jarras: cantidad en J4 – cantidad en J3 – 2 (*de proximidad*)

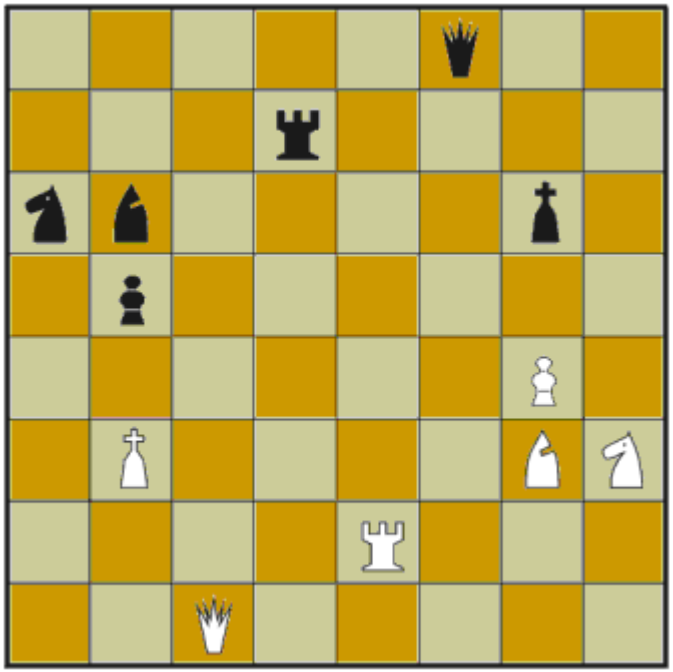


Estado Inicial Estado Final



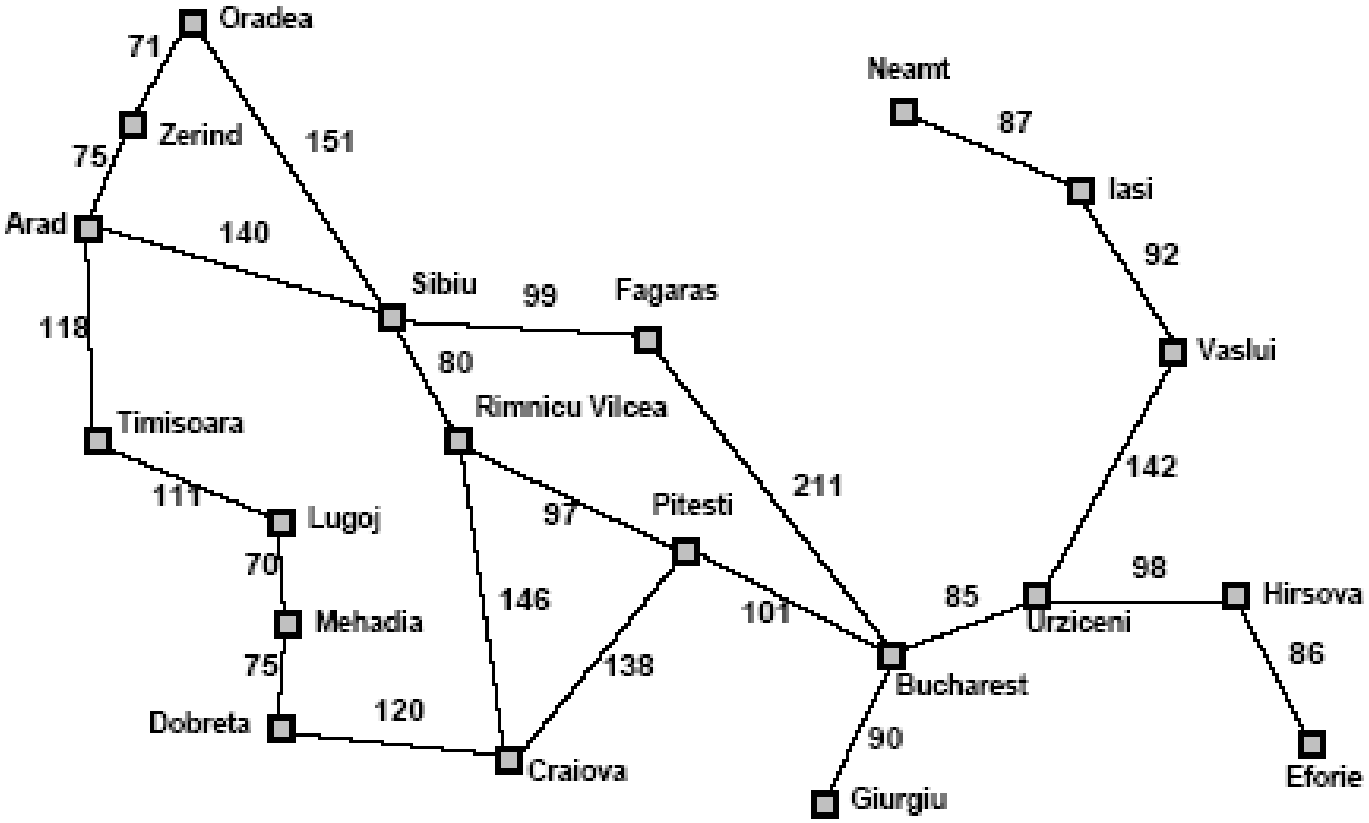
Ejemplo 3

- Ajedrez: nº de piezas de ventaja (*de calidad*)



Ejemplo 4

- Búsqueda del camino mínimo en las carreteras de Rumanía entre Arad y Bucarest
- Heurística: valor de la distancia en línea recta



Distancia en línea recta a Bucarest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Clasificación

- Heurísticas Generales: son adecuadas para múltiples dominios
  - Pe: el vecino más próximo para medir distancias
- Heurísticas de Propósito Especial: usan conocimiento exclusivo de un dominio

Heurísticas Bien Fundadas

1. Si estima la "calidad" del estado E
   - *h(E)* está bien fundada si los estados finales tienen el valor heurístico máximo posible → la estrategia debe buscar el máximo
   - El estado inicial suele tener valor heurístico 0
2. Si estima la "proximidad" a un estado final
   - *h(E)* está bien fundada si los estados finales tienen el valor 0

Búsqueda Informada

- Una *estrategia de búsqueda informada* utiliza conocimiento que va más allá de la definición del problema
- El conocimiento se “empotra” en la *función de evaluación* del nodo f(n)

Estrategia en Escalada

- Idea: Ordenar los hijos en base a evidencias heurísticas y elegir en cada paso a uno de los descendientes del estado actual que mejore el valor heurístico de su padre

- Sentido de la Escalada:
  - El mejor el valor más alto: Ascensión a la Colina
  - El mejor el valor más bajo: Descenso según el Gradiente

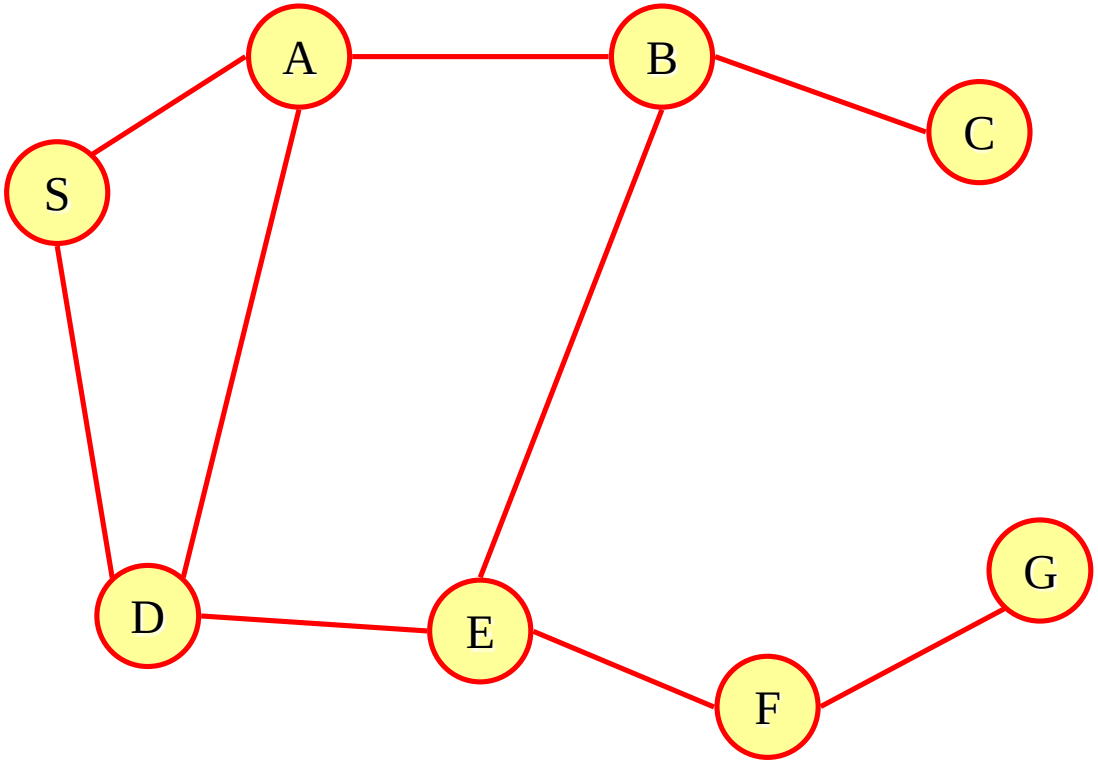
Variantes

- Escalada Simple:
  - Se generan los hijos uno a uno calculando su valor heurístico
  - El primer hijo que sea mejor que el estado actual se elige como sucesor y se convierte en el nuevo estado actual
- Escalada por Máxima Pendiente:
  - Se generan todos los hijos y se calcula su valor heurístico
  - Se escoge al mejor de los hijos
    - Si es mejor o igual que estado actual, pasa a ser el nuevo estado actual
    - Si no, se detiene el proceso

Procedimiento con Estrategia en Escalada por Máxima Pendiente

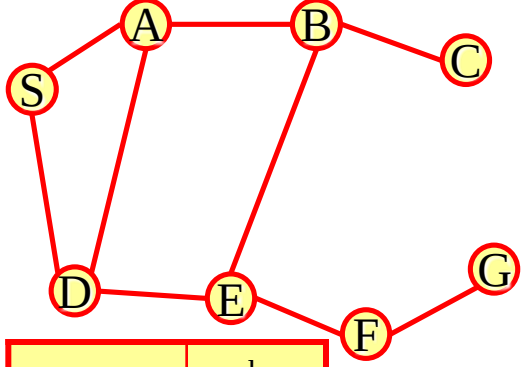
1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   - Si el primer nodo es el Objetivo, entonces salir del bucle
   - Si en primer nodo no el Objetivo, eliminar esta trayectoria de ABIERTA y añadir sus trayectorias descendientes, si existen, *al principio de la lista* ordenadas en base a sus expectativas de éxito, colocando en primer lugar la trayectoria más prometedora
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*

Ejemplo



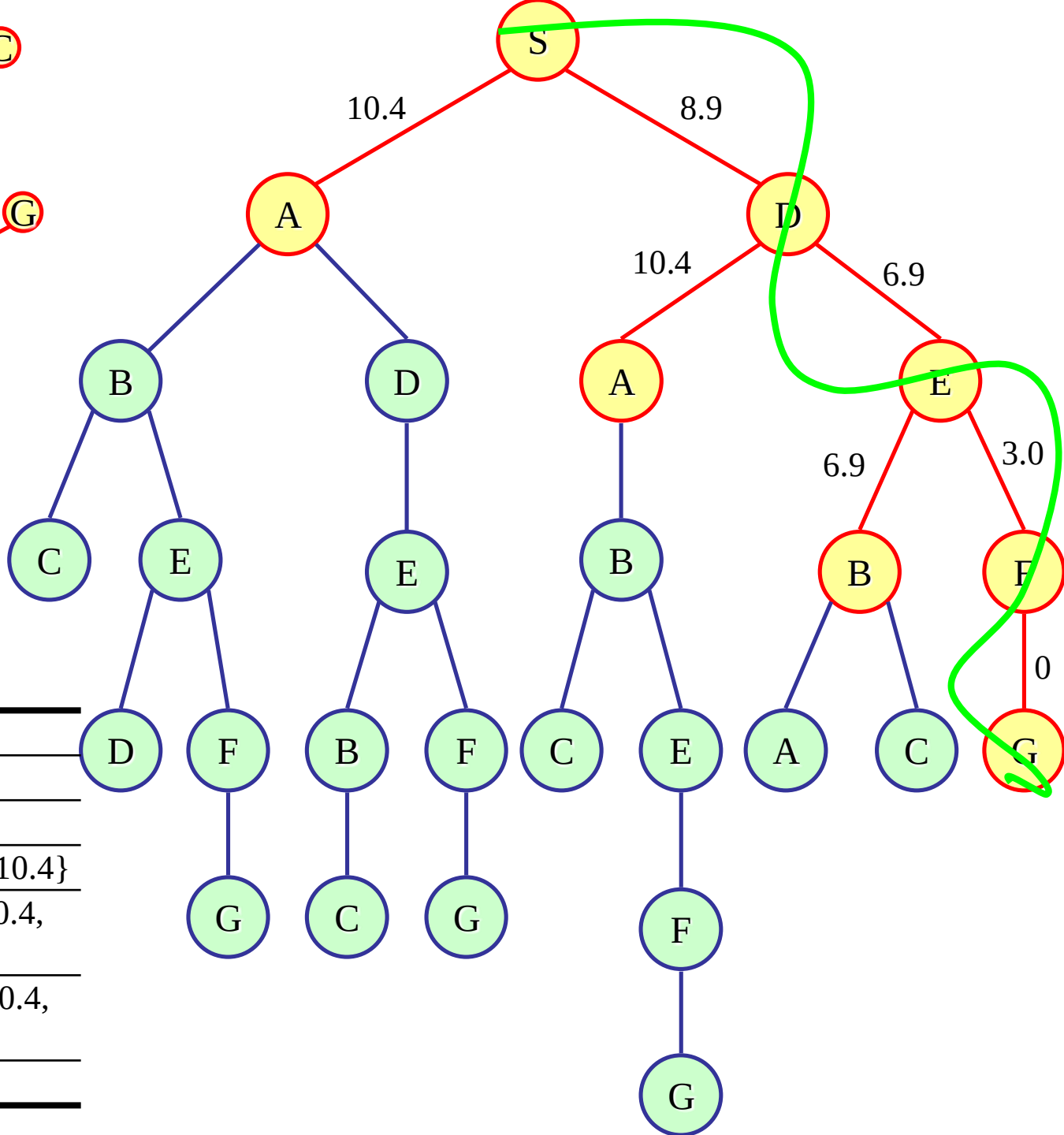
Ruta de evaluación del nodo indicado al G	Valor heurístico h
h(A)=h(A,G)	10,4
h(D)= h(D,G)	8,9
h(E)= h(E,G)	6,9
h(B)= h(B,G)	6,7
h(C)= h(C,G)	4,0
h(F)= h(F,G)	3,0

Como el valor de h es menor cuanto más próximos estamos del objetivo, el proceso es de minimización del coeficiente heurístico, es decir, el proceso será de descenso según el gradiente



	h
h(A,G)	10,4
h(D,G)	8,9
h(E,G)	6,9
h(B,G)	6,7
h(C,G)	4,0
h(F,G)	3,0

actual	ABIERTA
	{S}
S	{D/8.9, A/10.4}
D	{E/6.9, A/10.4, A/10.4}
E	{F/3.0, B/6.7, A/10.4, A/10.4}
F	{G/0.0, B/6.7, A/10.4, A/10.4}
G	Objetivo



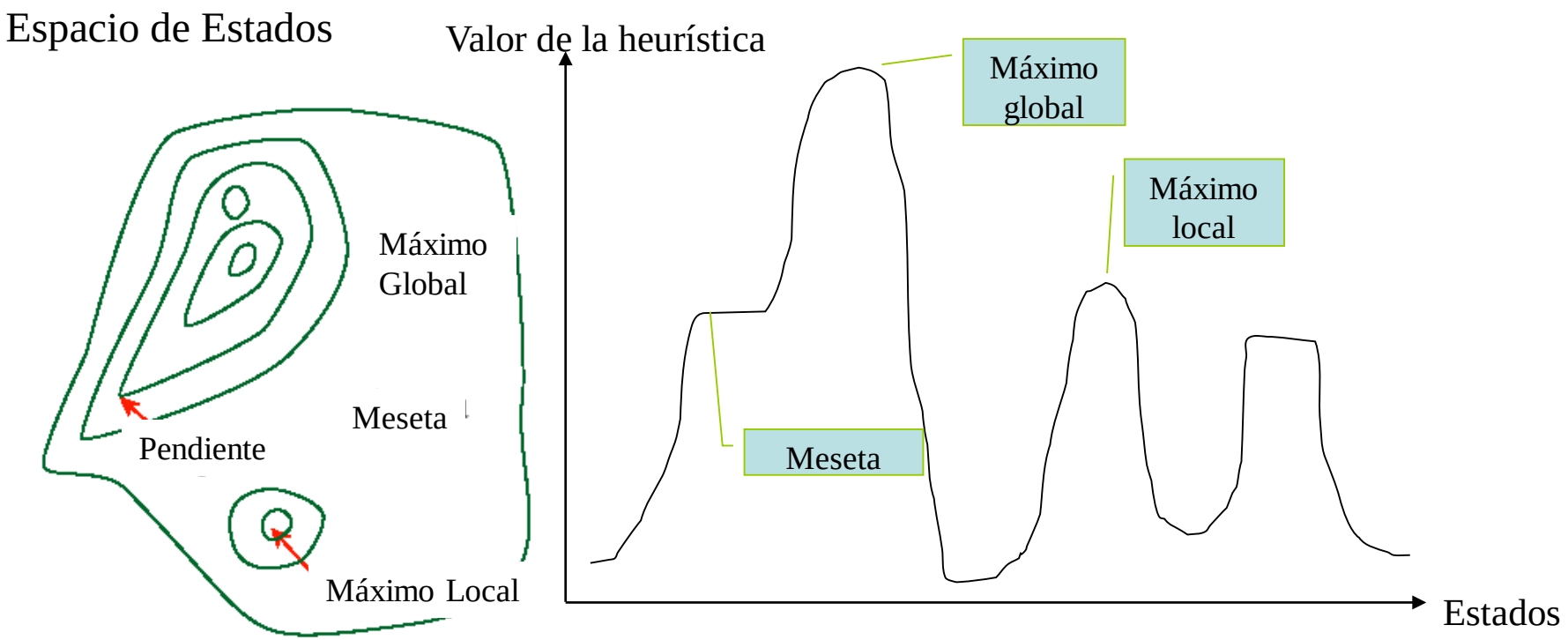
Características

- Ventajas
  - Muy poco consumo de espacio
  - Complejidad Espacial: O(1) (basta guardar un estado)
- Inconvenientes
  - Complejidad Temporal: exponencial en el peor caso (puesto que hay que revisar todos los nodos)
  - No es óptimo ni Completo
    - Pueden no encontrar solución aunque exista (ver Problemas)
    - No garantizan el camino más corto

Problemas

Existen situaciones en las que el procedimiento se estanca:

- Máximos Locales
- Mesetas
- Crestas



Máximos Locales

- Síntoma: Todos los hijos de un estado padre, que no es el objetivo, son peores que él
- Definición: un máximo local es un estado mejor que cualquier otro vecino, pero peor que otros más lejanos
- Consecuencia: el algoritmo para sin dar solución
- Solución: volver al nodo anterior y probar estado hijo distinto

Mesetas

- Síntoma: Todos los hijos de un estado padre, que no es el objetivo, tienen el mismo valor heurístico que el padre
- Definición: una meseta es una región del espacio de estados donde todos los estados tienen el mismo valor heurístico
  Consecuencia: el algoritmo para sin dar solución, y si sigue, la heurística no informa, por lo que la búsqueda se vuelve ciega
- Solución:hacer un “salto” grande para “salir” de la meseta

Crestas

- Síntoma: Mezcla de los dos anteriores donde se llega a un conjunto de máximos locales contiguos
- Definición: región del espacio de estados que tiene algunos estados con mejor valor heurístico que los colindantes, pero a los que no se puede llegar por transiciones simples, es decir, aplicando un único operador
- Consecuencia: el algoritmo para sin dar solución
- Solución: Dar un paso más → generar sucesores de sucesores y “ver que pasa”

Otra solución para todos

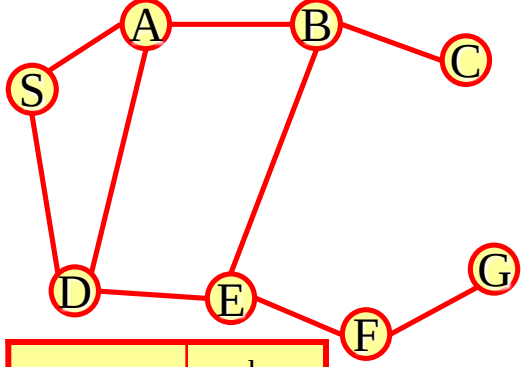
- Reiniciar toda o parte de la búsqueda (implica no iniciar en el mismo estado inicial)

Estrategia Primero el Mejor

- Sinónimos: El Mejor Nodo, Best-First
- Idea: si la ordenación tras la inserción de los nodos hijos se hace con todo el contenido de la pila en vez de sólo con los nodos hijos
- Resultado: un procedimiento que intenta combinar anchura y profundidad guiándonos por la heurística, de manera que se sigue un camino y se pasa a otro cuando deja de ser prometedor
- Diferencia con la Estrategia de Escalada: los descendientes del estado actual “compiten” con todos los demas nodos no expandidos

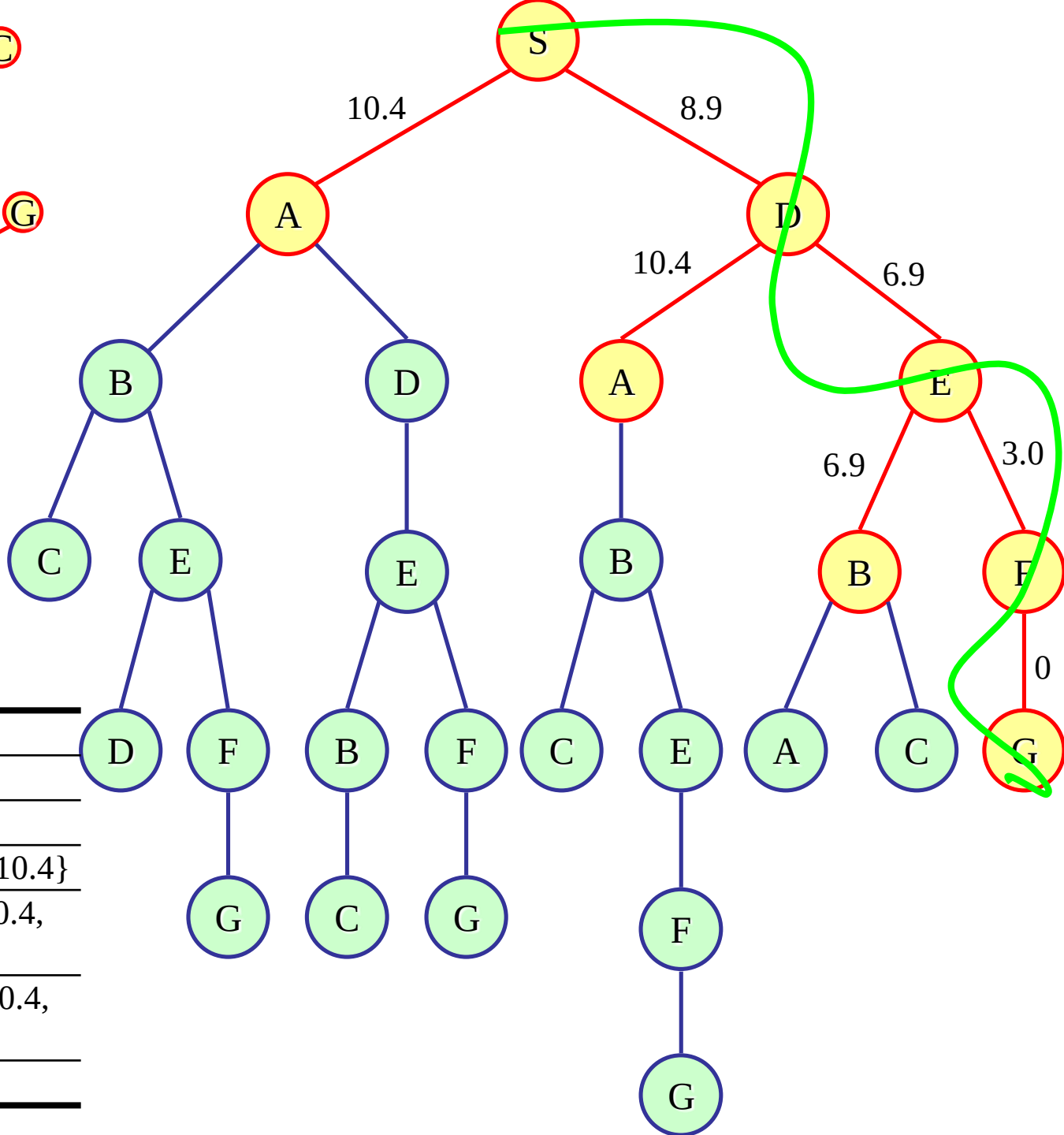
Procedimiento con Estrategia Primero el Mejor

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   - Si el primer nodo es el Objetivo, entonces salir del bucle
   - Si en primer nodo no el Objetivo, eliminar esta trayectoria de ABIERTA y añadir sus trayectorias descendientes, si existen, *a la lista* ordenando todos los elementos de ABIERTA en base a sus expectativas de éxito, colocando en primer lugar la trayectoria más prometedora
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*



	h
h(A,G)	10,4
h(D,G)	8,9
h(E,G)	6,9
h(B,G)	6,7
h(C,G)	4,0
h(F,G)	3,0

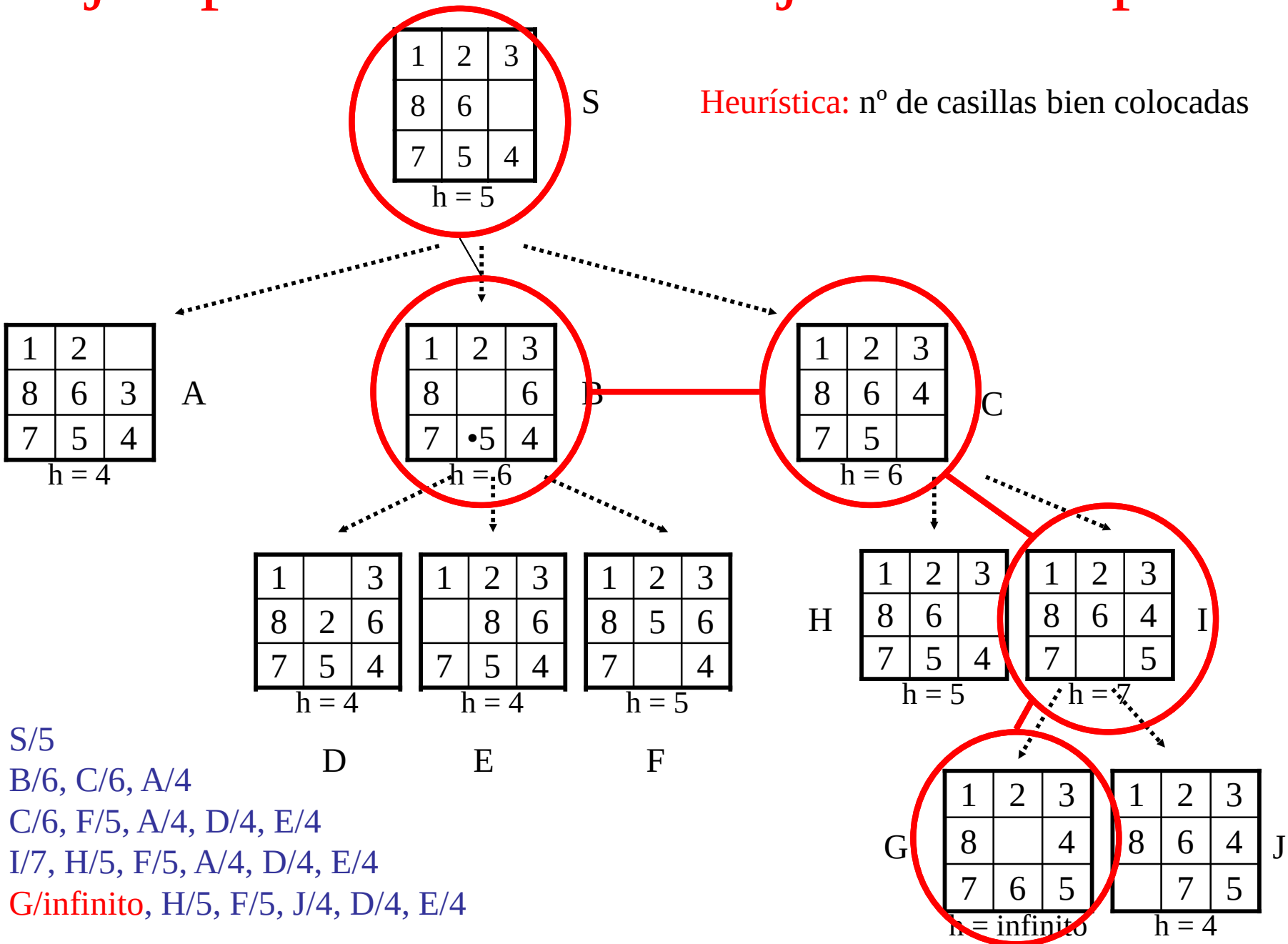
actual	ABIERTA
	{S}
S	{D/8.9, A/10.4}
D	{E/6.9, A/10.4, A/10.4}
E	{F/3.0, B/6.7, A/10.4, A/10.4}
F	{G/0.0, B/6.7, A/10.4, A/10.4}
G	Objetivo



Características

- Complejidad
  - Temporal: $O(b^m)$, siendo m= profundidad de la solución más lejana
  - Espacial: $O(b^m)$
  - En el peor caso hay que recorrer todos los estados
- Completitud:
  - Es completo, ya que si hay solución la encuentra, siempre que la heurística funcione bien
- Optimalidad:
  - No es óptimo ya que puede no dar la solución más cercana
- En esencia sigue siendo un procedimiento de búsqueda en profundidad
- Da la primera solución que encuentra

Ejemplo: Primero el Mejor en el 8-puzzle



S/5
B/6, C/6, A/4
C/6, F/5, A/4, D/4, E/4
I/7, H/5, F/5, A/4, D/4, E/4
G/infinito, H/5, F/5, J/4, D/4, E/4

Estrategia por Haces

- Sinónimo: Beam-Search
- Idea: Variante de Exploración por Niveles (Anchura), pero en vez de explorarse todas las ramas sólo se explora un subconjunto seleccionado o HAZ, *podando* el resto

Criterios de Poda

1. Por Umbral en el Número de Ramas a explorar

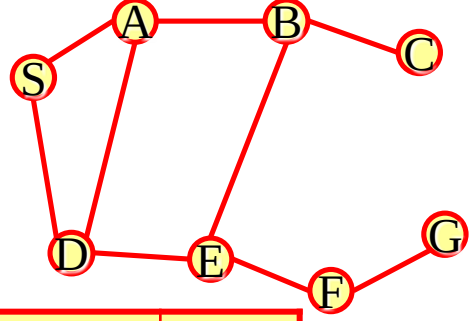
   Sólo perdura un número máximo prefijado de las ramas mejores o más prometedoras

2. Por Umbral de Parámetros

   Sólo perduran las ramas cuyos valores de parámetros heurísticos superan un cierto umbral

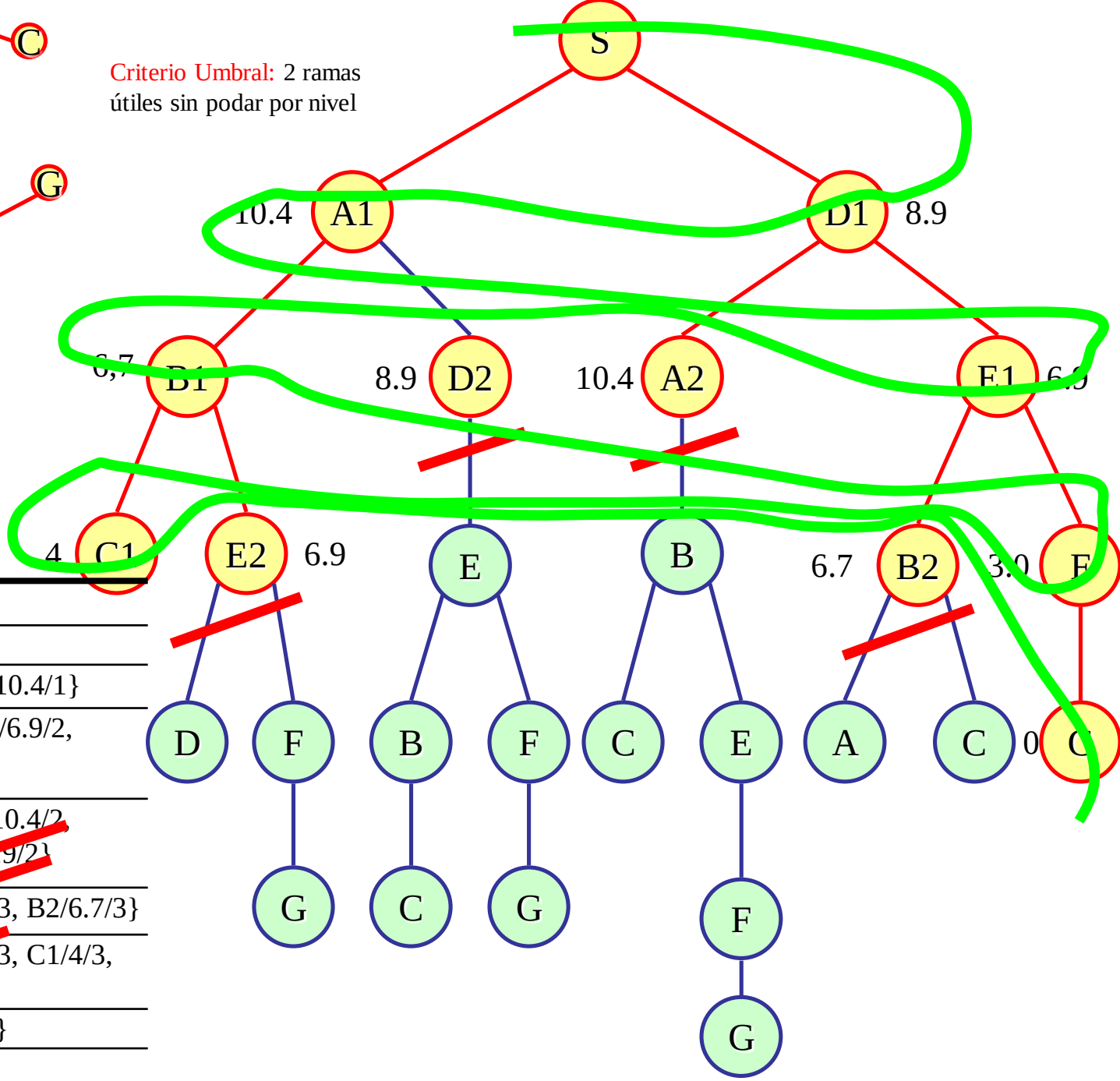
Procedimiento de la Estrategia por Haces

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   - Si el primer nodo es el Objetivo, entonces salir del bucle
   - Si en primer nodo no el Objetivo, eliminar esta trayectoria de ABIERTA y añadir sus trayectorias descendientes, si existen, ordenadas en base a sus expectativas de éxito *al final de la lista,* y *eliminando de toda la lista aquellas trayectorias que no superen el criterio de poda establecido*
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*



	h
h(A,G)	10,4
h(D,G)	8,9
h(E,G)	6,9
h(B,G)	6,7
h(C,G)	4,0
h(F,G)	3,0

Criterio Umbral: 2 ramas útiles sin podar por nivel



actual	ABIERTA
	{S}
S	{D1/8.9/1, A1/10.4/1}
D	{A1/10.4/1, E1/6.9/2, A2/10.4/2}
A1	{E1/6.9/2, A2/10.4/2, B1/6.7/2, D2/8,9/2}
E1	{B1/6.7/2, F/3/3, B2/6.7/3}
B1	{F/3/3, B2/6.7/3, C1/4/3, E2/6,9/3}
F	{C1/4/3, G/0/4}
C1	{G/0}
G	Objetivo

Consideraciones

- Completitud

  El procedimiento no asegura la correcta solución del problema, ya que la poda puede producir la mutilación de algunas o todas las ramas que conducen al objetivo

Estrategia de Ramificación y Acotación (Ramificación y Salto)

- Sinónimos: de Coste Uniforme, Branch-and-Bound Search (BBS), Uniform-Cost Search (UCS), Algoritmo de Dijkstra
- Nota: es una estrategia de búsqueda no informada, pero que introducimos aquí a efectos de claridad expositiva en relación a las estrategias siguientes que veremos
- Supuestos:
  1. Se puede medir el coste de cada trayectoria parcial según se va avanzando, o de otra forma, se conoce el coste de aplicar cada operador
  2. El coste total de una trayectoria parcial se corresponde con la acumulación de los costes de las transiciones entre estados
- Principio: utilizar la información suministrada por una función de coste de cada trayectoria parcial

Método

- Utilizar la lista ABIERTA de trayectorias parciales que contienen los nodos recorridos, *ordenadas de menor a mayor costo acumulado*, de manera que la primera trayectoria de la lista es siempre la de menor coste acumulado

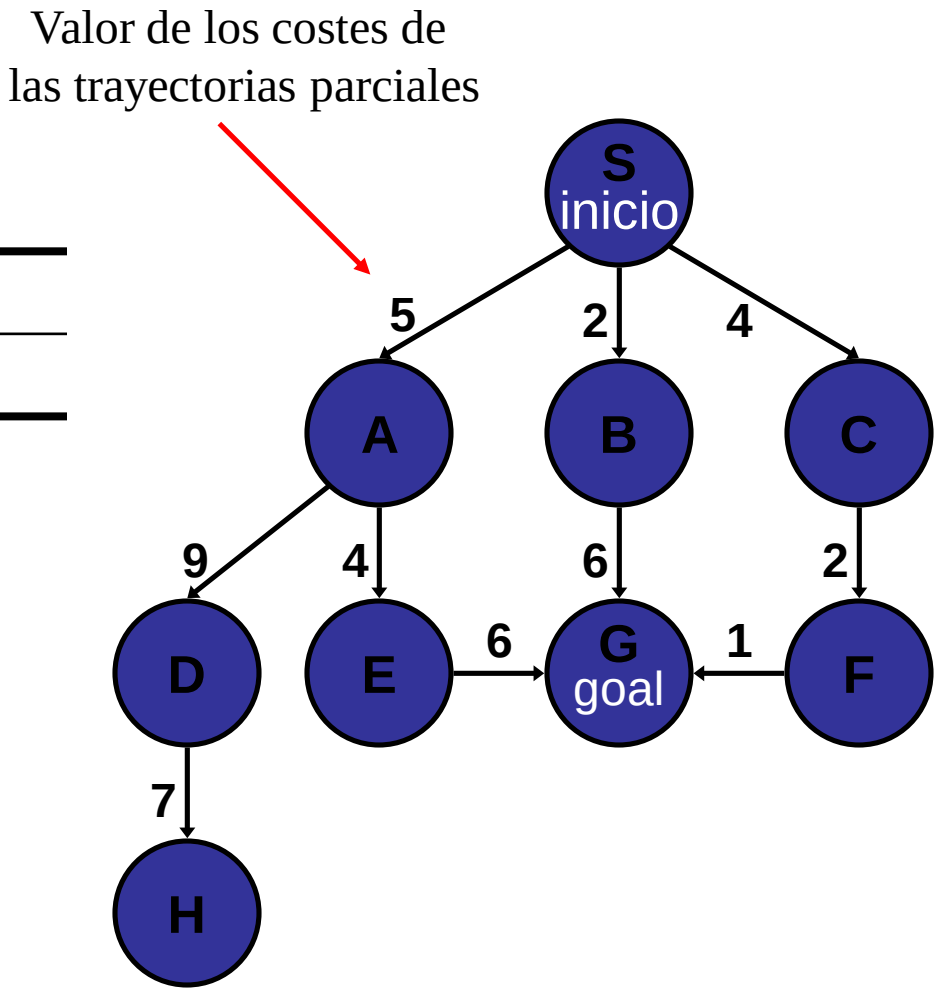
- La idea es que la primera trayectoria de la lista ABIERTA genere, mientras sea posible su expansión, nuevas trayectorias por ramificación de sus hijos

Procedimiento por Ramificación y Acotación

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   - Si el primer nodo es el Objetivo, entonces salir del bucle
   - Si en primer nodo no el Objetivo, eliminar esta trayectoria de ABIERTA y añadir sus trayectorias descendientes, si existen, *a la lista* ordenando la lista ABIERTA entera en base al coste acumulado, colocando en primer lugar la trayectoria de menor coste
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*

Estrategia de Ramificación y Acotación. Ejemplo

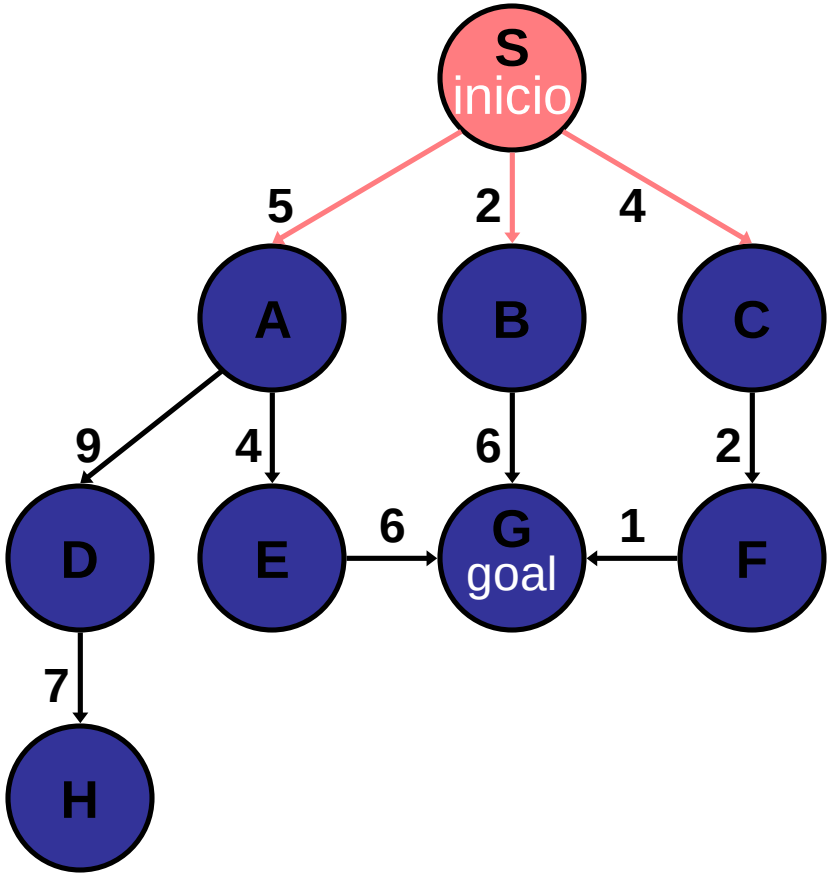
actual	Lista de nodos
	{S}



Ejemplo 2

nodos comprobados: 1, expandidos: 1

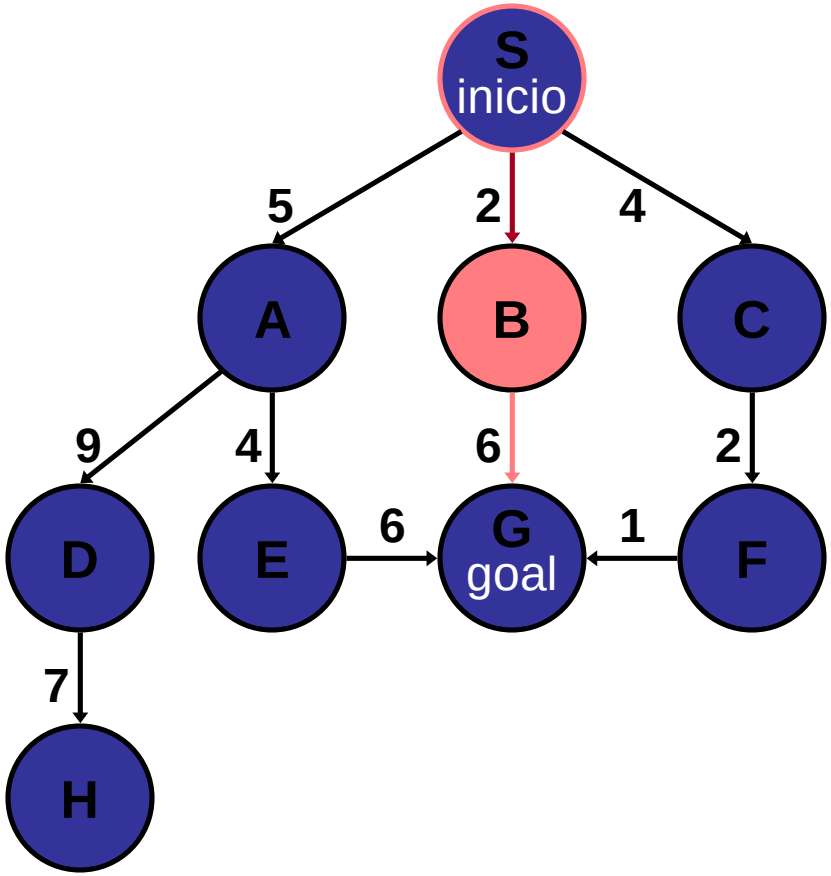
actual	Lista de nodos
	{S/0}
S no objetivo	**{B/2,C/4,A/5}**



Ejemplo 3

nodos comprobados: 2, expandidos: 2

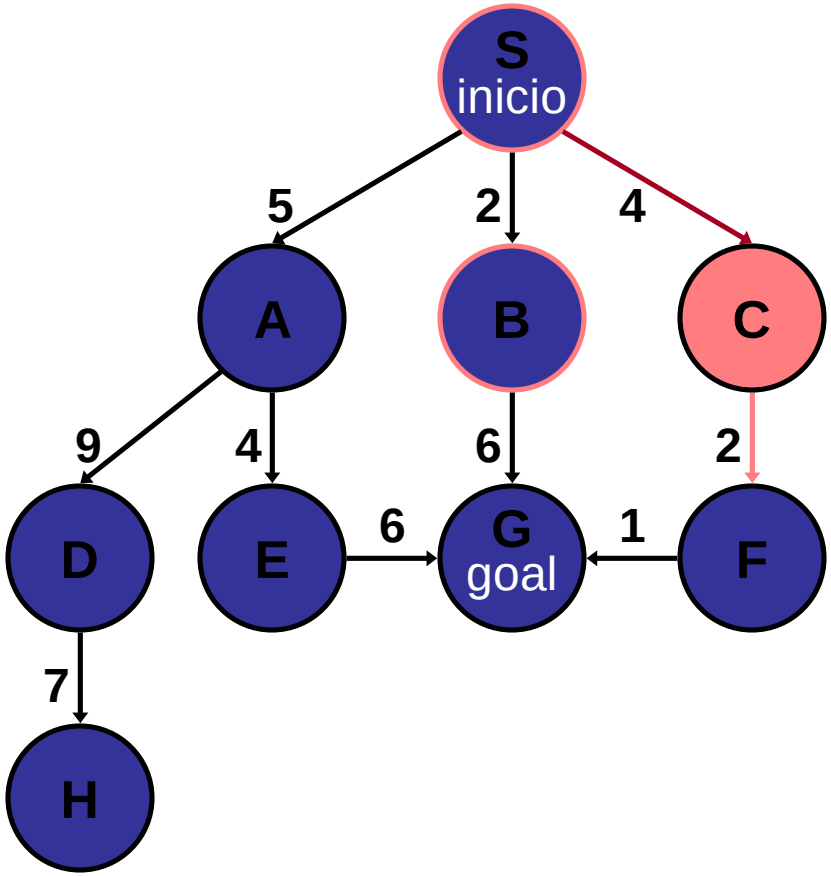
actual	Lista de nodos
	{S}
S	**{B/2,C/4,A/5}**
B no objetivo	**{C/4,A/5,G/2+6}**



Ejemplo 4

nodos comprobados: 3, expandidos: 3

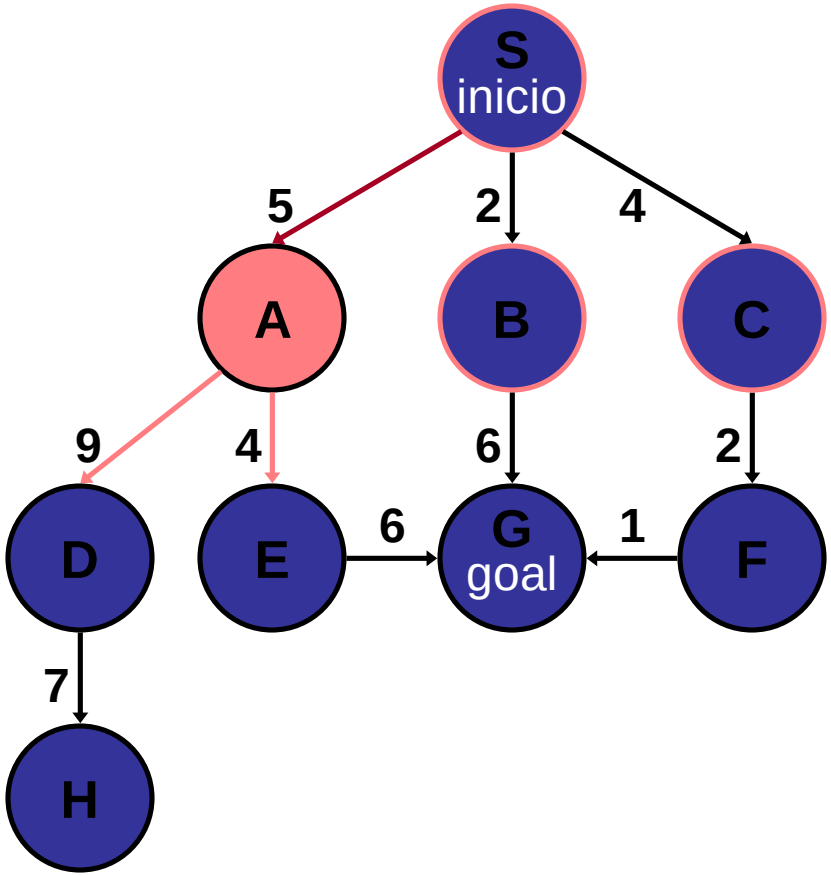
actual	Lista de nodos
	{S}
S	**{B/2,C/4,A/5}**
B	**{C/4,A/5,G/8}**
C no objetivo	**{A/5,F/4+2,G/8}**



Ejemplo 5

nodos comprobados: 4, expandidos: 4

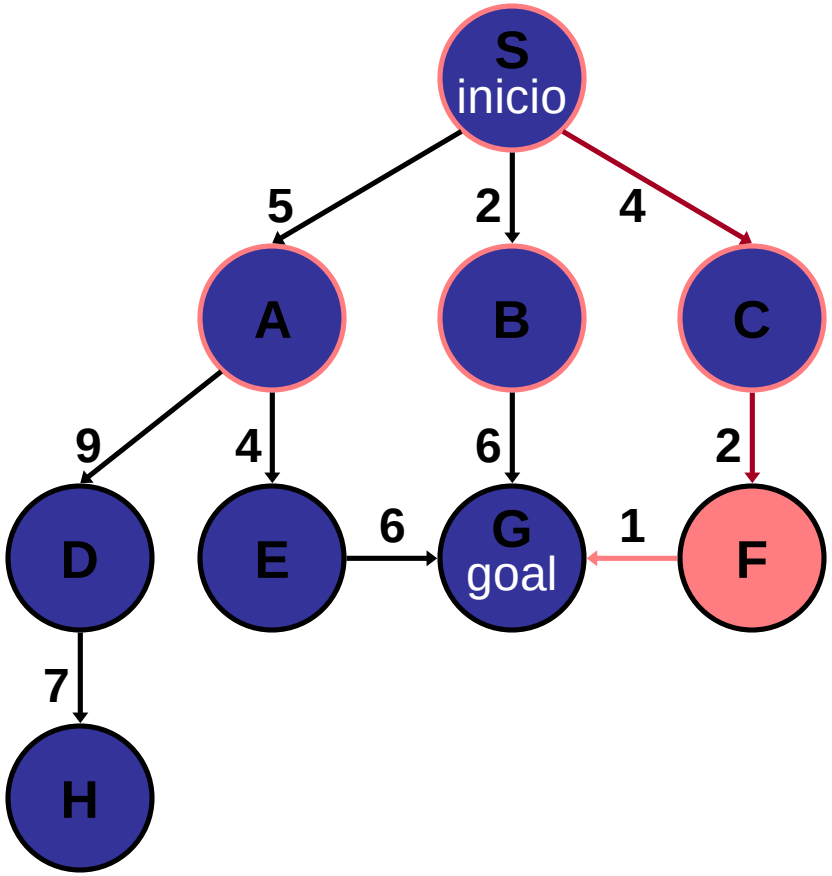
actual	Lista de nodos
	{S}
S	**{B/2,C/4,A/5}**
B	**{C/4,A/5,G/8}**
C	**{A/5,F/6,G/8}**
A no objetivo	**{F/6,G/8,E/5+4, D/5+9}**



Ejemplo 6

nodos comprobados: 5, expandidos: 5

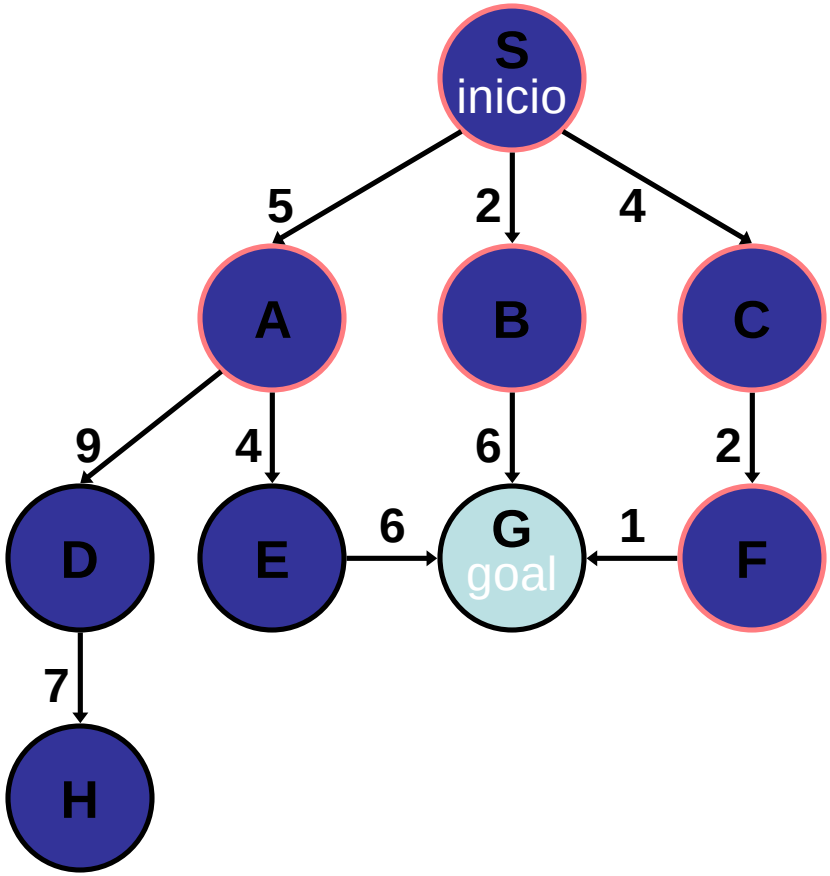
actual	Lista de nodos
	{S}
S	**{B:2,C:4,A:5}**
B	**{C:4,A:5,G:8}**
C	**{A:5,F:6,G:8}**
A	**{F:6,G:8,E:9,D:14}**
F no objetivo	**{G:4+2+1,G:8,E:9,D:14}**



Ejemplo 7

nodos comprobados: 6, expandidos: 5

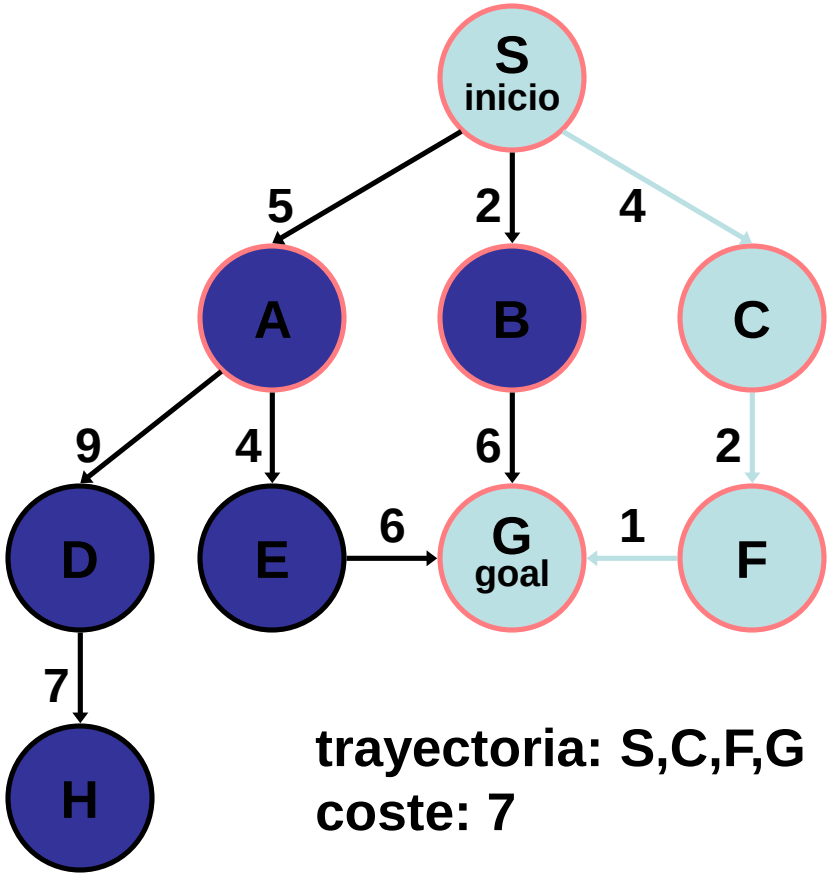
actual	Lista de nodos
	{S}
S	**{B:2,C:4,A:5}**
B	**{C:4,A:5,G:8}**
C	**{A:5,F:6,G:8}**
A	**{F:6,G:8,E:9,D:14}**
F	**{G:7,G:8,E:9,D:14}**
G objetivo	**{G:8,E:9,D:14}** no expandir



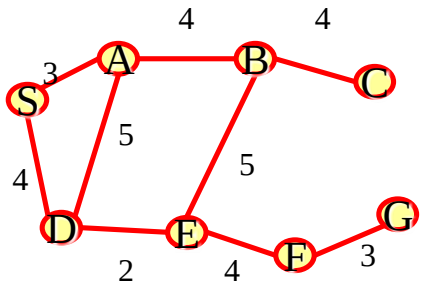
Ejemplo 8

nodos comprobados: 6, expandidos: 5

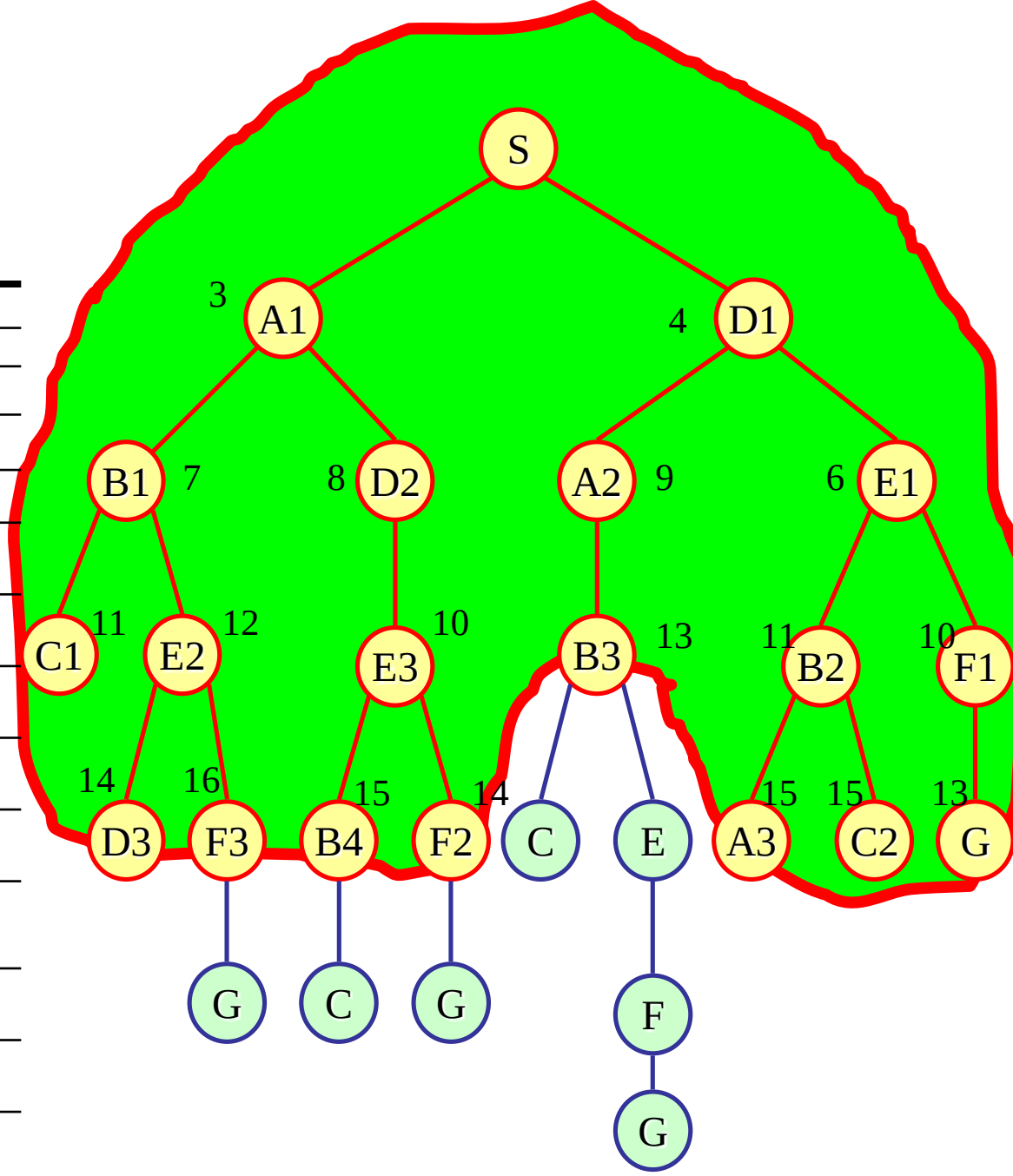
actual	Lista de nodos
	{S}
S	**{B:2,C:4,A:5}**
B	**{C:4,A:5,G:8}**
C	**{A:5,F:6,G:8}**
A	**{F:6,G:8,E:9,D:14}**
F	**{G:7,G:8,E:9,D:14}**
G	**{G:8,E:9,D:14}**



trayectoria: S,C,F,G
coste: 7



actual	ABIERTA
	{S}
S	{A1/3, D1/4}
A1	{D1/4, B1/3+4, D2/3+5}
D1	{E1/4+2, B1/7, D2/8, A2/4+5}
E1	{B1/7, D2/8, A2/9, F1/4+2+4, B2/4+2+5}
B1	{D2/8, A2/9, F1/10, C1/3+4+4, B2/11, E2/3+4+5}
D2	{A2/9, E3/3+5+2 ,F1/10, C1/11, B2/11, E2/12}
A2	{E3/10, F1/10, C1/11, B2/11, E2/12, B3/4+5+4}
E3	{F1/10, C1/11, B2/11, E2/12, B3/13, F2/3+5+2+4, B4/3+5+2+5}
F1	{C1/11, B2/11, E2/12, G/4+2+4+3, B3/13, F2/14, B4/15}
C1	{B2/11, E2/12, G/13, B3/13, F2/14, B4/15}
B2	{E2/12, G/13, B3/13, F2/14, B4/15, A3/4+2+5+4, C2/4+2+5+4}
E2	{G/13, D3/3+4+5+2, F2/14, B4/15, A3/15, C2/15, F3/3+4+5+4,}
G	Objetivo



Características

- Complejidad
  - **Espacial y Temporal** : $O(b^d)$ **Exponencial**
    - *d* es la profundidad de la solución
    - *b*: el factor de ramificación en cada nodo no terminal
      y siempre que sea asumible que todas las trayectorias parciales tienen el mismo coste (pe 8-puzzle)
- Completitud: **completo sobre árboles finitos**
- Optimalidad (si *b* es finito, *coste >= exponencial*)

Estrategia de Ramificación y Acotación con Subestimación

- Idea: mejorar Ramificación y Acotación añadiendo una estima del coste hasta el nodo objetivo, al coste parcial acumulado de la trayectoria actual.
- Es decir:
  - Considerar no sólo el camino hasta alcanzar un estado sino también
  - Las expectativas acerca de lo próximo que está ese estado del estado objetivo

Estrategia de Ramificación y Acotación con Subestimación (II)

Es decir se combina:

1. Conocimiento Determinista procedente del coste producido para alcanzar el estado actual desde el inicial
2. Conocimiento Heurístico proveniente de la estimación de costes de la trayectoria restante por determinar entre el estado actual y el objetivo

Estrategia de Ramificación y Acotación con Subestimación (III)

- *A efectos del procedimiento los costes se consideran sólo valores no negativos*
- Consideremos que las heurísticas tienen carácter de diferencia o distancia y, por tanto, los procedimientos deberán tender a obtener trayectorias que minimicen distancias
- Para evitar el riesgo de no expandir un nodo que lleve a la trayectoria óptima y, por tanto, no alcanzarla, es interesante que el coste estimado sea una subestimación del coste restante.
- Se puede observar intuitivamente que una sobrestimación para alguna trayectoria, puede llegar a provocar que se desestime cuando podría dar lugar a una solución útil

Función de Coste Total Estimado

Se hace uso de una función de coste total estimado acumulado para cada nodo intermedio *n* en el proceso de exploración:

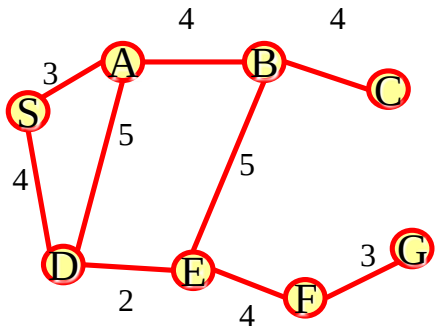
$$f^*(n) = g(n) + h^*(n)$$

$g(n)$ → coste real acumulado (determinista) desde el estado inicial al estado actual *n*

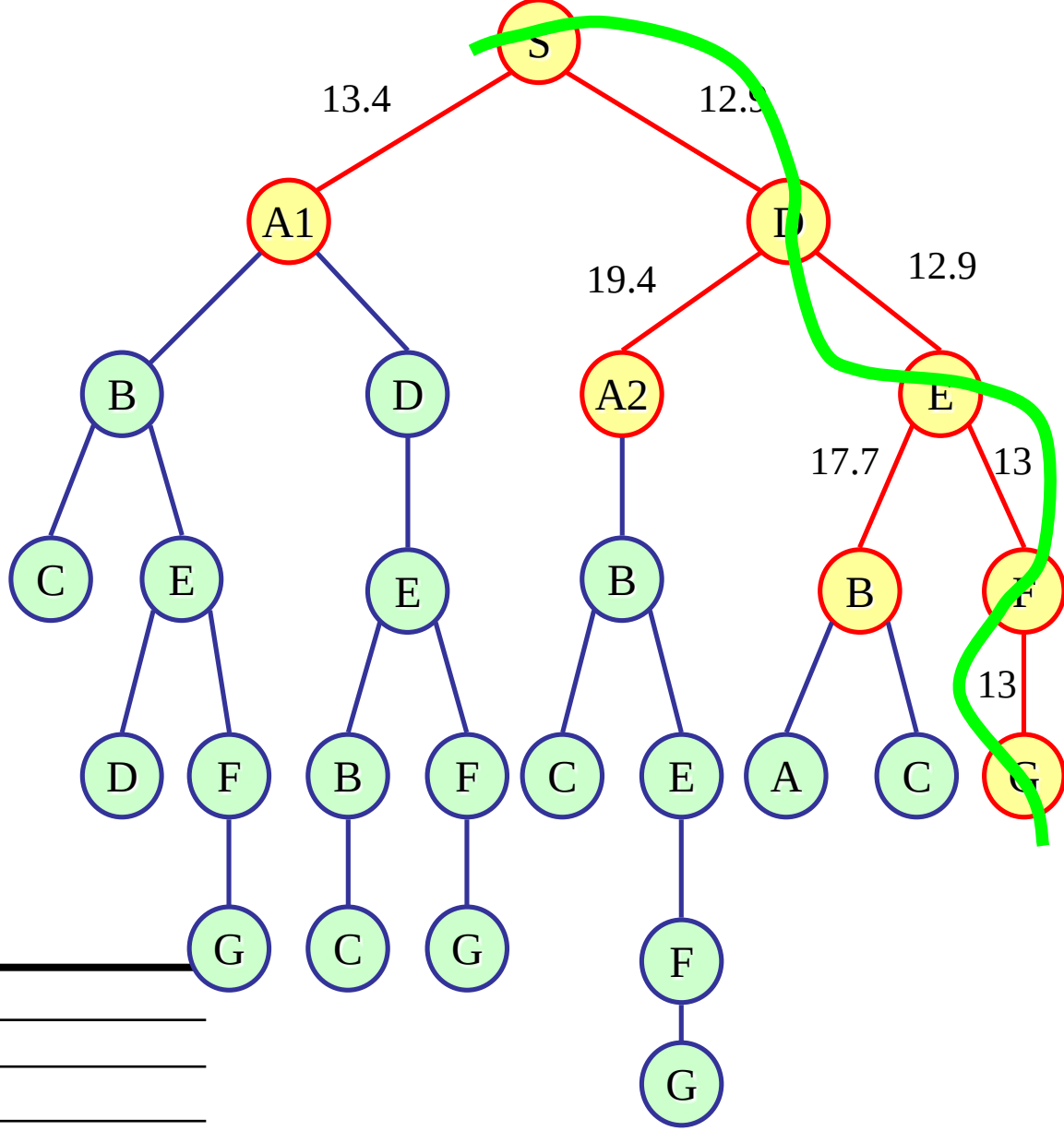
$h^*(n)$ → estimación (heurística) del coste del estado actual al estado objetivo

Procedimiento por Ramificación y Acotación con Subestimación

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   - Si el primer nodo es el Objetivo, entonces salir del bucle
   - Si en primer nodo no el Objetivo, eliminar esta trayectoria de ABIERTA y añadir sus trayectorias descendientes, si existen, *a la lista* ordenando la lista ABIERTA entera en base al coste total estimado, colocando en primer lugar la trayectoria de menor coste
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*



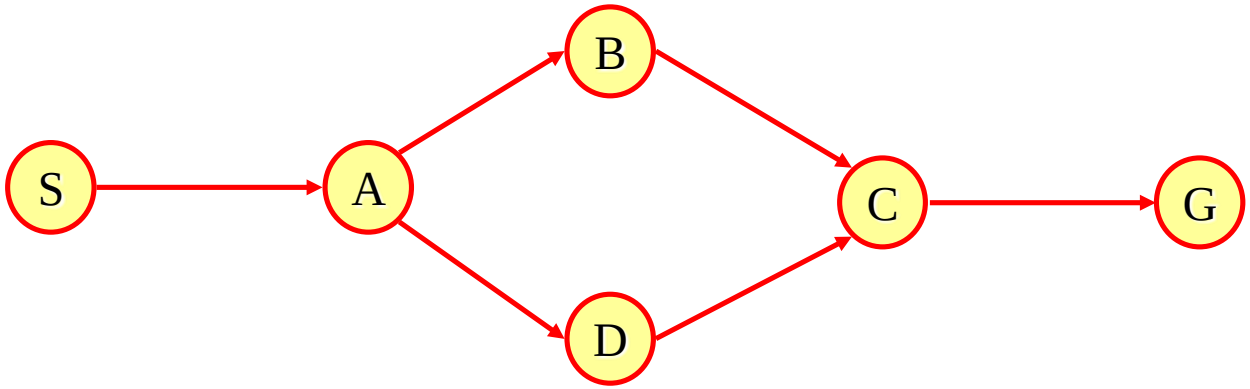
	h
h(A,G)	10,4
h(D,G)	8,9
h(E,G)	6,9
h(B,G)	6,7
h(C,G)	4,0
h(F,G)	3,0



actual	ABIERTA
	{S}
S	{D/4+8.9, A1/3+10.4}
D	{E/6+6.9, A1/13.4, A2/4+5+10.4}
E	{F/4+2+4+3.0 ,A1/13.4, B/4+2+5+6.7, A2/19.4}
F	{G/4+2+4+3, A1/13.4, B/17.7, A2/19,4}
G	Objetivo

Estrategia de Ramificación y Acotación con Programación Dinámica

- Nota: no puede considerarse un procedimiento informado, pero se describe aquí a efectos didácticos en vez de en el apartado de los no informados
- Observación: en los procedimientos anteriores hay subtrayectorias desechadas una vez por los procedimientos que son vueltas a recorrer de nuevo
- Principio: en una trayectoria óptima, todas las subtrayectorias también lo son



- Si SABCG es óptima entre S y G, entonces ABC es óptima entre A y C

Estrategia de Ramificación y Acotación con Programación Dinámica (II)

- Reflexión: si en la lista de trayectorias recorridas existen varias que tienen el último nodo común, sólo la de menor coste podrá formar parte de la trayectoria óptima
- Solución Posible: eliminar de la lista a las trayectorias que tengan los mismos nodos inicial y final y que no sean de menor coste
- Comentario: esta solución, por tanto, no compromete la optimalidad

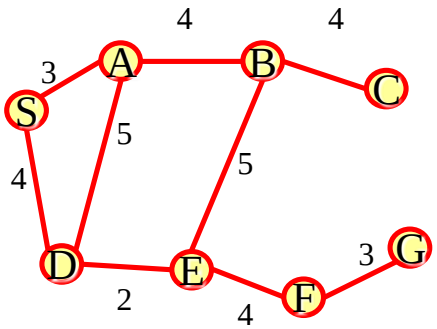
Estrategia de Ramificación y Acotación con Programación Dinámica (III)

Ímplementación - se precisan dos listas

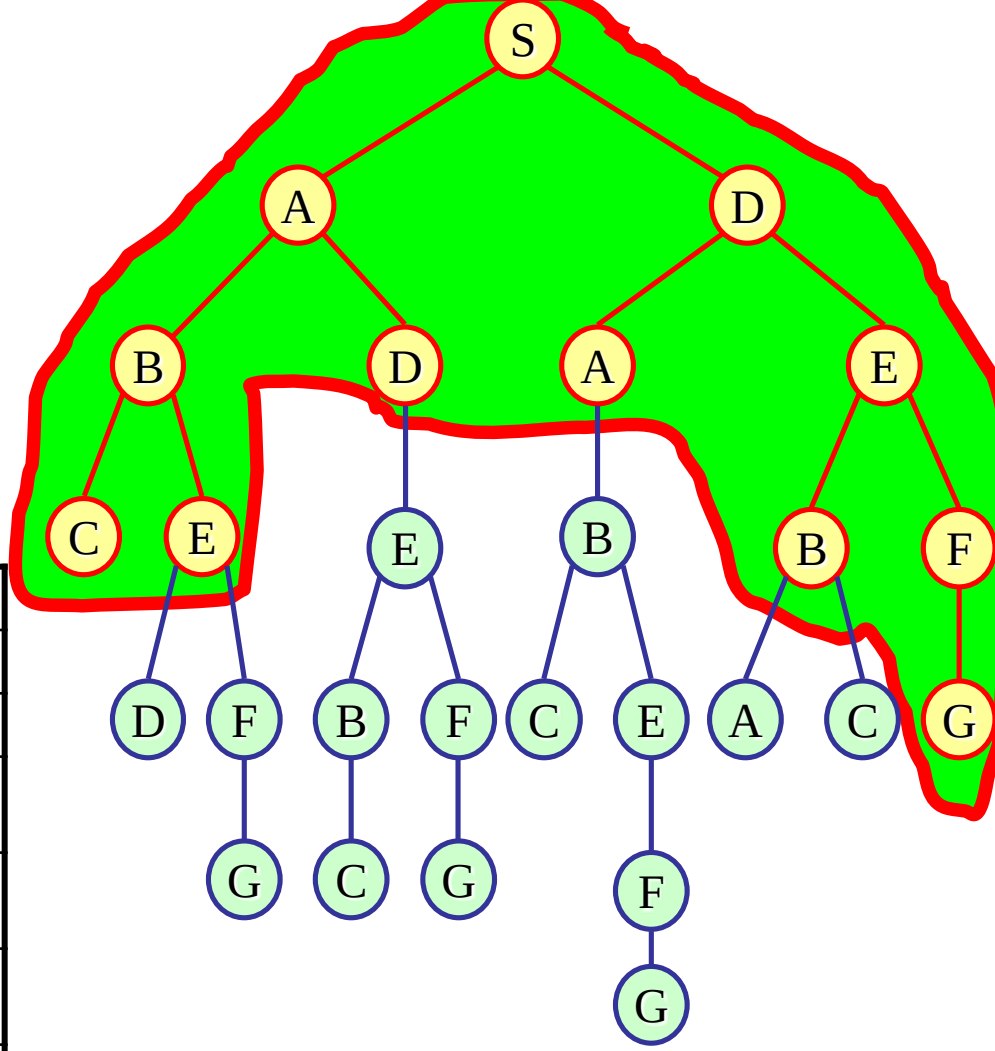
- ABIERTA: como en los casos anteriores, contiene la lista de trayectorias parciales susceptibles de ser estudiadas
- CERRADA: contiene las trayectorias desechadas que son de coste mínimo de entre las que tienen nodos extremos comunes y que han sido eliminadas de la lista abierta

Procedimiento de Ramificación y Acotación con Programación Dinámica

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz. Inicializar a vacía la lista CERRADA
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   a. Si el primer nodo es el Objetivo, entonces salir del bucle
   b. Si en primer nodo no el Objetivo, entonces:
      - Eliminar la primera trayectoria de la lista ABIERTA, incluyéndola en la lista cerrada en el caso de que sea de menor coste que una similar ya contenida, eliminando en su caso la antigua trayectoria contenida en CERRADA
      - Formar nuevas trayectorias a partir de la trayectoria eliminada de ABIERTA, ramificando el último nodo de la misma
      - Añadir las nuevas trayectorias a la lista ABIERTA, si existen
      - Ordenar la lista ABIERTA en base al costo acumulado de cada una, colocando la de mínimo coste al inicio de la lista
      - Si dos o más trayectorias de ABIERTA acaban en un nodo común, borrar las mismas excepto la que posee mínimo coste de entre ellas. Eliminar esta última también si existe una similar con menor coste en la lista CERRADA. Al eliminar trayectorias de ABIERTA deben insertarse en CERRADA salvo que ya exista allí una similar de menor coste
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*



Actual	ABIERTA	CERRADA
	{S}	{NIL}
S	{SA/3, SD/4}	{S}
A	{SD/4, SAP/3+4, ~~SAD/3+5~~}	{SA/3, SAD/8}
D	{SDE/4+2, SAP/7, ~~SDA/4+5~~}	{SD/4, SA/3, ~~SAD/8~~}
E	{SAP/7, SDEF/4+2+4, ~~SDEB/4+2+5~~}	{SD/4, SA/3, SDE/6, SDEB/11}
B	{SDEF/10, SABC/3+4+4, SABE/3+4+5}	{SD/4, SA/3, SDE/6, ~~SDEB/11~~, SAB/7}
F	{SABC/11, ~~SABE/12~~, SDEFG/4+2+4+3}	{SD/4, SA/3, SDE/6, SAB/7, SDEF/10}
C	{SDEFG/13}	{SD/4, SA/3, SDE/6, SAB/7, SDEF/10}
G	Objetivo	



Estrategia A*

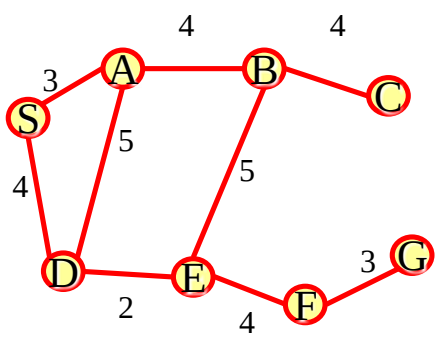
- Familia de Procedimientos (Hart, Nilsson, Raphael)
- Objetivo: Mejorar Primero el Mejor
  - Introduciendo evaluación del camino recorrido junto a predicción del camino por recorrer, como en Ramificación y Salto con Subestimación

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

  - Eliminando repeticiones como en Ramificación y Salto con Programación Dinámica, recordando “enlaces” entre estados ya visitados
- Solución: combinación de ambos procedimientos anteriores

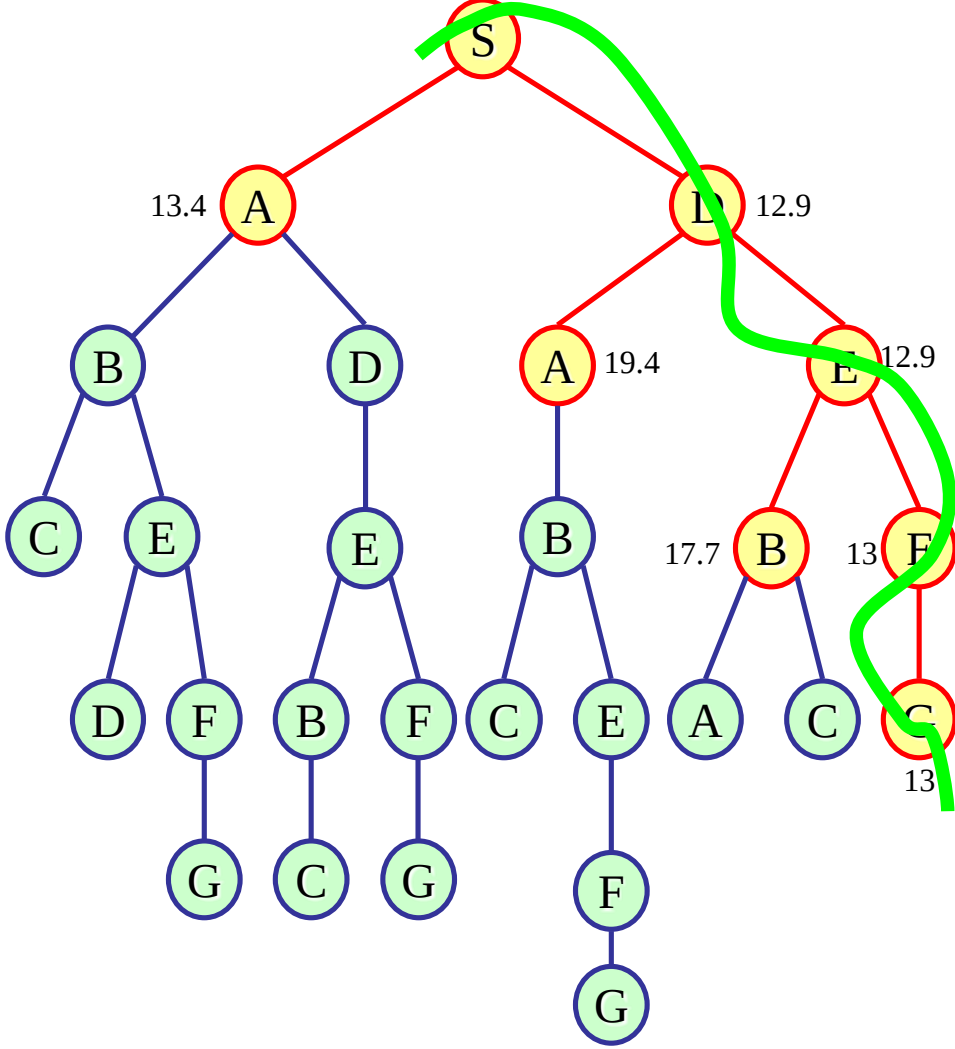
Procedimiento A*

1. Introducir en ABIERTA una trayectoria inicial que contenga solamente el nodo raíz. Inicializar a vacía la lista CERRADA
2. Hasta que lista esté vacía o se encuentre el objetivo, examinar la primera trayectoria de la lista
   a. Si el primer nodo es el Objetivo, entonces salir del bucle
   b. Si en primer nodo no el Objetivo, entonces:
      - Eliminar la primera trayectoria de la lista ABIERTA, incluyéndola en la lista cerrada en el caso de que sea de coste más óptimo que una similar ya contenida, eliminando en su caso la antigua trayectoria contenida en CERRADA
      - Formar nuevas trayectorias a partir de la trayectoria eliminada de ABIERTA, ramificando el último nodo de la misma
      - Añadir las nuevas trayectorias a la lista ABIERTA, si existen
      - Ordenar la lista ABIERTA en base al costo acumulado de cada una, colocando la de mejor valor de función evaluación al inicio de la lista
      - Si dos o más trayectorias de ABIERTA acaban en un nodo común, borrar las mismas excepto la que posee mínimo valor de función de evaluación de entre ellas. Eliminar esta última también si existe una similar con menor valor de función de evaluación en la lista CERRADA. Al eliminar trayectorias de ABIERTA deben insertarse en CERRADA salvo que ya exista allí una similar de menor coste
3. Si se ha encontrado el objetivo, finaliza con ÉXITO, y *la solución es la primera trayectoria en la lista*
4. Si no, *Problema sin Solución*



	h
h(A,G)	10,4
h(D,G)	8,9
h(E,G)	6,9
h(B,G)	6,7
h(C,G)	4,0
h(F,G)	3,0

Actual	ABIERTA	CERRADA
	{S}	{NIL}
S	{SD/8.9+4, SA/10.4+3}	{S}
D	{SDA/4+5+10.4, SDE/4+2+6.9, SA/13.4}	{S, SD/12,9 SDA/19.4}
A	{SDEF/4+2+4+3, SA/13.4, SDEB/4+2+5+6.7}	{S, SD/12.9, SDA/19.4, SDE/12.9}
F	{SDEFG/4+2+4+3, SA/13.4, SDEB/17.7}	{S, SD/12.9, SDA/19.4, SDE/12.9, SDEF/13}
G	Objetivo	



Grado de Información Relativa entre Heurística

La función de estima heurística $h^*(n)$ debe ser una subestimación del valor real $h(n)$:

$$0 \leq h^*(n) \leq h(n)$$

De dos heurísticas, una de ellas h_1 se dice que está más informada que la otra h_2 si:

$$0 \leq h_2^*(n) \leq h_1^*(n) \leq h(n)$$

Una heurística estará tanto más informada cuanto más próximo sea su valor estimado al valor real

Admisibilidad

- Una heurística *h*(n)* se dice admisible si asegura encontrar una solución óptima si esta existe

- Esto se cumple si nunca sobreestima el coste de alcanzar la mejor solución desde *n*:

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

- *Si en un procedimiento, todos los costes son positivos y h*(n) es una subestimación de h(n), entonces se garantiza encontrar la trayectoria de mínimo coste si existe y el procedimiento se dice admisible*

Conclusión

- Si $h^*(n)$ es admisible entonces $f^*(n)$ tampoco sobreestima el coste real de la mejor solución que pase por en estado n

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

Optimalidad

- Sea un proceso de determinación de trayectoria entre un estado inicial $S = n_0$ y un estado objetivo G
- Sea que hemos alcanzado un estado intermedio n_k:

$$f^*(n_k) = g(n_k) + h^*(n_k) = \sum_{i=1}^{k} c_i + h^*(n_k)$$

- c_i es el coste de la trayectoria parcial desde n_{i-1} hasta n_i

- Para una trayectoria de $L+1$ nodos con $n_L = G$

$$f^*(G) = f(G) = \sum_{i=1}^{L} c_i$$

Optimalidad (II)

- Para cualquier nodo intermedio $n_k < L$ entre n_0 y G, si lo que se pretende con la estrategia es minimizar distancias o coste de caminos se debe cumplir:

$$f^*(n_k) = \sum_{i=1}^{k} c_i + h^*(n_k) \leq f(G) = \sum_{i=1}^{L} c_i$$

$$h^*(n_k) \leq \sum_{i=1}^{L} c_i - \sum_{i=1}^{k} c_i = \sum_{i=k+1}^{L} c_i$$

Optimalidad (III)

- Los procesos menos informados, es decir, los que emplean peores heurísticas expanden más nodos

- El proceso más informado, es decir aquel tal que $h^*(n) = h(n)$ es el que menos nodos expande

Conclusión

- La heurística no debe sobreestimar el valor real, porque se pueden desechar caminos más óptimos
- A más informada es una heurística, menos nodos tiende a expandir
- Interesa una heurística que no sobreestime el valor real, (por tanto ADMISIBLE), pero LO MÁS INFORMADA posible

Características de A*

- Es óptimo y completo si:
  - Todo nodo tiene un número finito de sucesores
  - El coste de aplicación de cada operador (transición) es no negativo (>0)
  - La función *h(e)* es una heurística admisible

Complejidad

- Espacial: $O(b^p)$
- Temporal: $O(b^p)$

Donde: b es el factor de ramificación y p la profundidad de la solución

- En el peor de los casos, cuando $h^*(n) = 0$, sigue siendo necesario recorrer todo el árbol
- En el caso promedio:
  - El consumo de memoria es alto, por la necesidad de almacenar estados pendientes y visitados
  - Tiempo promedio aceptable (mejora la búsqueda en profundidad)

En Resumen

- El algoritmo A* resuelve problemas en los que es necesario encontrar la mejor solución
- Su coste en espacio y tiempo en el caso medio es mejor que los algoritmos de búsqueda ciega si el heurístico es adecuado

Variantes

- RTA*
- A*PI
- A*SRM

RTA*

- Real Time A*
- Para tareas en tiempo real que, por tanto no pueden esperar a encontrar la solución óptima
- Obliga a tomar una decisión cada periodo de tiempo k*t
- El periodo de tiempo determina la profundidad alcanzable en búsqueda
  - Busca hasta donde le da tiempo
  - Indica la operación sobre el estado actual que inicia el camino que lleva el mejor estado encontrado

A*PI

- A* con profundización iterativa
- Búsqueda con profundización iterativa controlada por límite en la función de evaluación A*

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

  Nota: en principio no comprueba estados repetidos
- Objetivo: reducir necesidades de memoria
- Límite de coste *k*, no de profundidad
- Idea: expandir sólo estados n con coste dentro de la cota $f^*(n) \leq k$
- El resultado de cada iteración se utiliza para establecer la cota de la siguiente

FIN DE BÚSQUEDA INFORMADA