Inteligencia Artificial

Tema 3. Estrategias de Búsqueda No Informada

Inteligencia Artificial

2º curso Grado en Ingeniería Informática

Elisa Guerrero Vázquez

Esther L. Silva Ramírez

Departamento de Ingeniería Informática

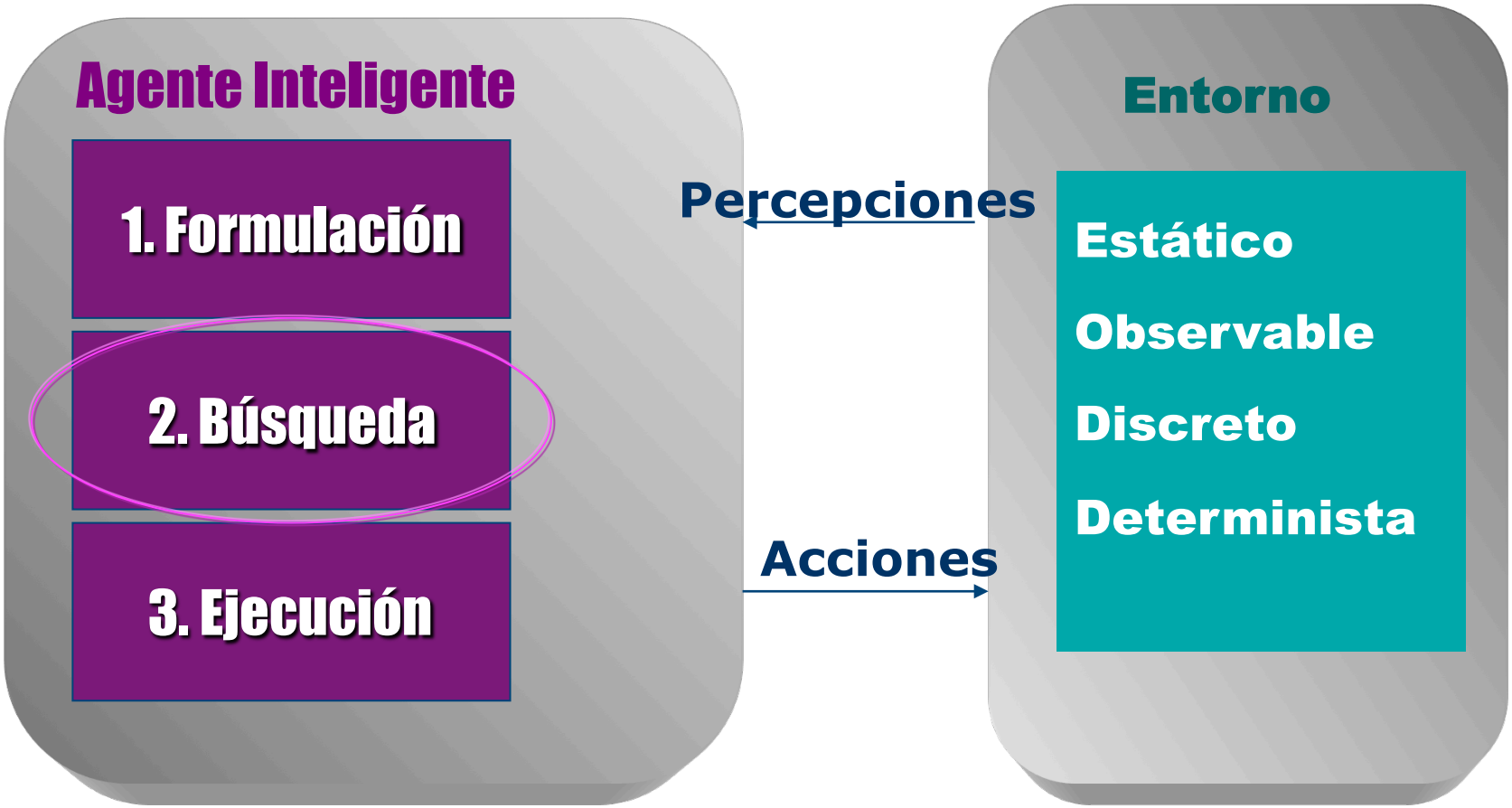
UCA

Objetivos

- Al finalizar este tema el alumno deberá ser capaz de:
  - Resolver los problemas de búsqueda utilizando las distintas técnicas de búsqueda no informada
  - Analizar el coste de las distintas técnicas para evaluar las ventajas y desventajas de cada método
  - Implementar los distintos algoritmos estudiados

Agentes que resuelven problemas

- **Búsqueda:** hallar la secuencia de acciones que conduzcan a un agente a un estado objetivo.



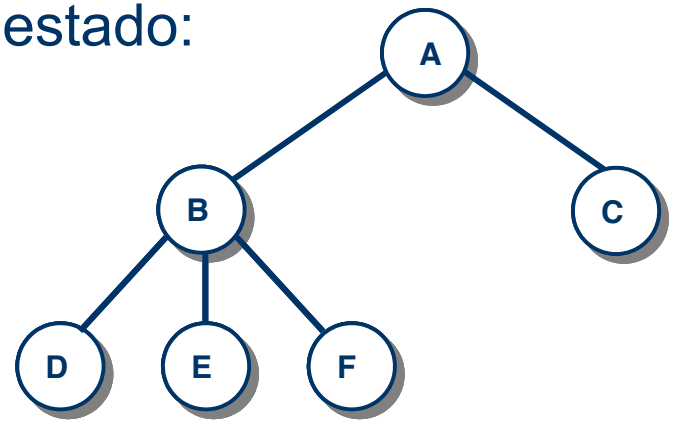
Formulación de un problema de búsqueda

Un problema se define mediante:

- **Estado Inicial**
- **Test Objetivo**
- **Operadores (espacio de estados)**
- **Camino o solución y su coste**

Diseño del Árbol o Grafo de Búsqueda

- **Estado**: configuración del mundo en un momento dado
- **Nodo**: estructura de datos para representar toda la información referente a cada estado:
  - Estado
  - Nodo Padre
  - Acción
  - Coste
  - Profundidad



- Selección: **Estrategia concreta de selección del nodo**
- ¿Es Objetivo el Nodo Actual?
- Función sucesor al nodo actual cuando no es un nodo objetivo
- Lista de Nodos ABIERTOS
- Lista de Nodos CERRADOS

Estrategias de búsqueda

Cuando hay varias posibilidades en el espacio de búsqueda, la estrategia debe determinar cuál es el siguiente estado a considerar

- **Búsqueda No Informada:** Exploración sistemática del espacio de búsqueda, pero sin información que ayude a determinar qué camino seguir

- **Búsqueda Informada o Heurística:** Se evalúa en cada momento qué estado pudiera ser mejor que otro para su expansión, utilizando cierta información en el dominio del problema

Algoritmo General de Búsqueda

```
Solucion: función Búsqueda (tNodo: Inicial, entero: estrategia)
inicio
  tNodo Actual
  tLista: Abiertos ← {Inicial}  // El nodo inicial se guarda en Abiertos
  logico Objetivo: Falso
  mientras (No Vacia(Abiertos)) Y (No Objetivo)
          Actual ← Primero(Abiertos) // selecciona primer nodo de Abiertos
          si EsObjetivo(Actual) entonces
                  Objetivo ← Verdadero
          si_no
                  Sucesores ← Expandir(Actual)
                  Abiertos ← {Abiertos+Sucesores} //de acuerdo a estrategia
          fin_si
          Cerrados ← {Cerrados+Actual}
  fin_mientras
  si Objetivo entonces
       devolver  Camino a la Solución
  si_no devolver Fallo
fin_función
```

Estrategias de búsqueda no informada

- Búsqueda en Anchura
- Búsqueda en Profundidad
  - Con retroceso
  - Profundidad Limitada
  - Profundidad iterativa
- Búsqueda Biridireccional

Búsqueda 1º en Anchura

- Estrategia de selección del nodo actual de acuerdo a una estructura FIFO: el primer nodo que se genera es el primero que se expande (el más antiguo en la lista de Abiertos)

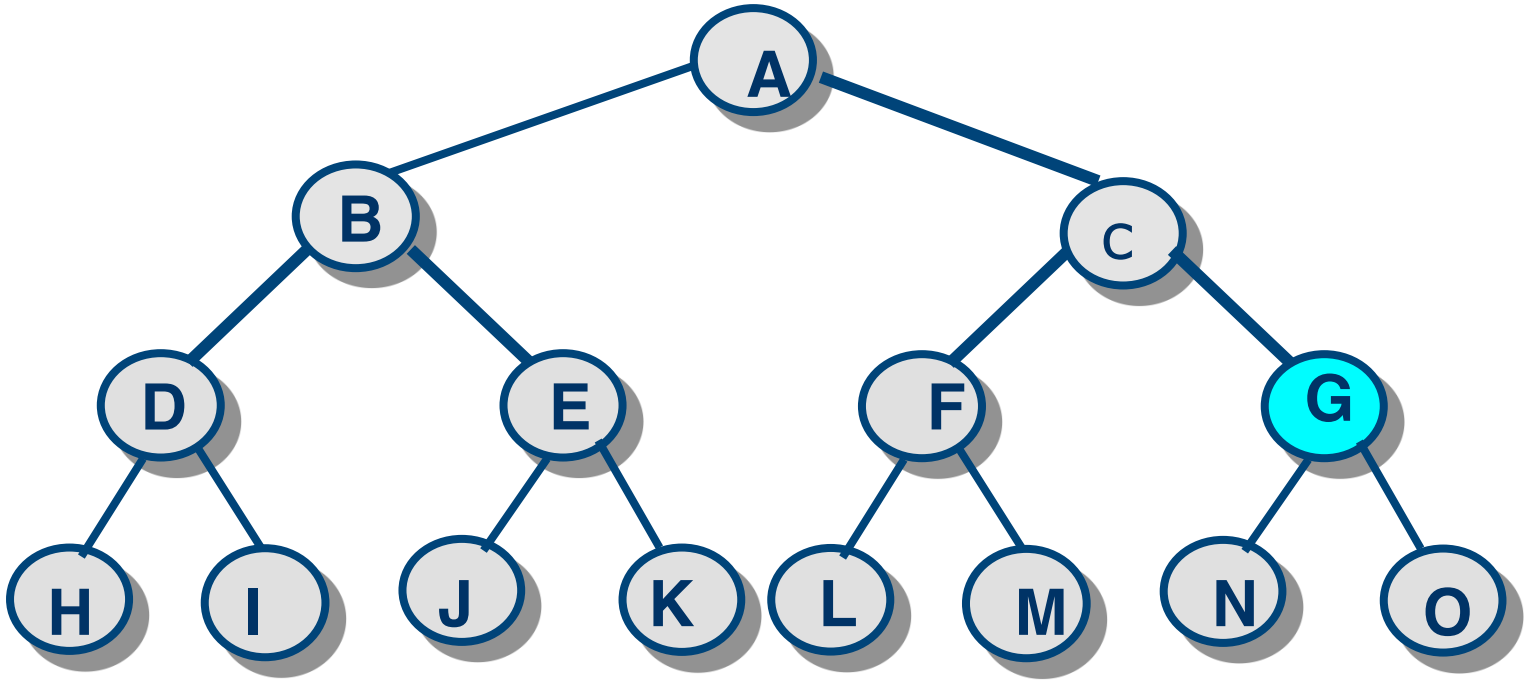
- Desde la perspectiva de la creación de un árbol de búsqueda: todos los nodos de un nivel se expanden antes que los del nivel siguiente

Búsqueda 1º en Anchura

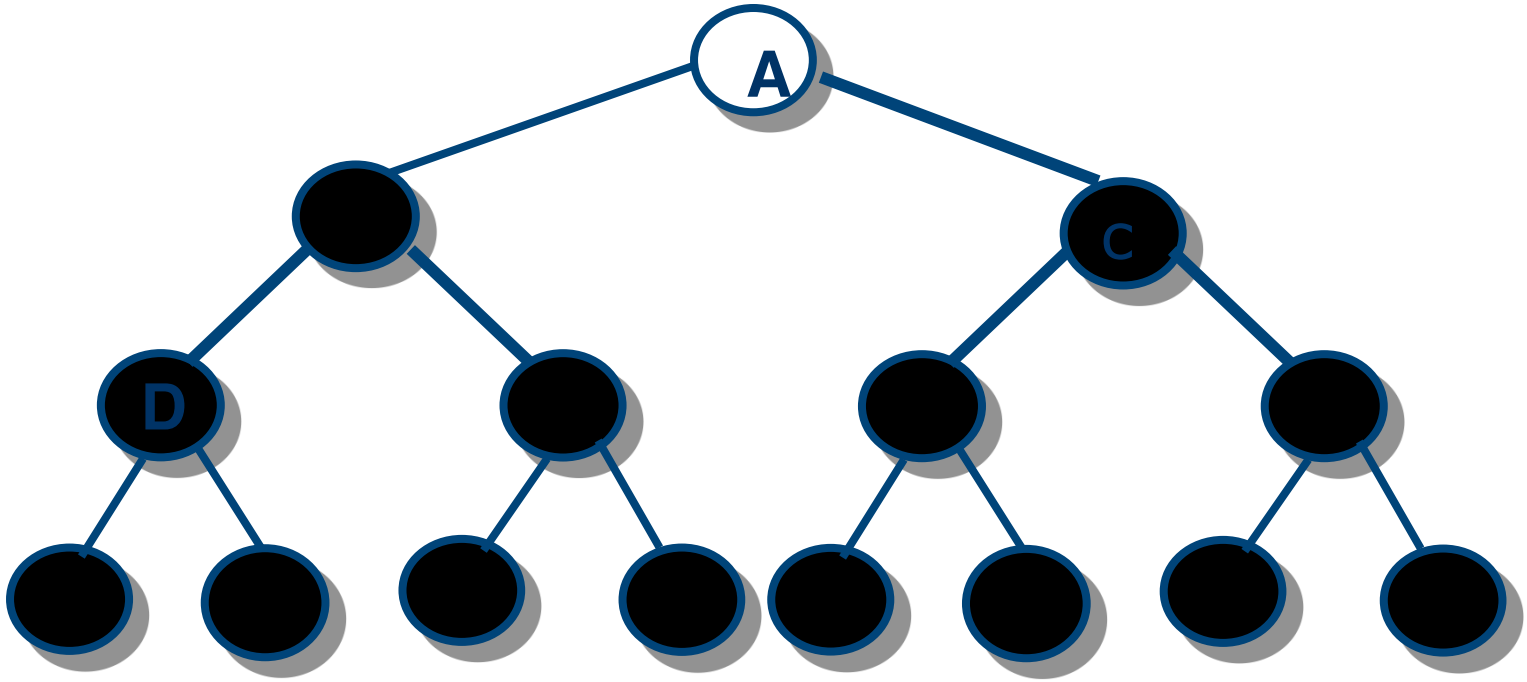
- Seleccionar el nodo raíz (nivel 0)
- Seleccionar los nodos generados a partir del nodo raíz (nivel 1)
- Después los sucesores (nivel 2) y así sucesivamente

- Estrategia **FIFO** = seleccionar el nodo más antiguo en la lista

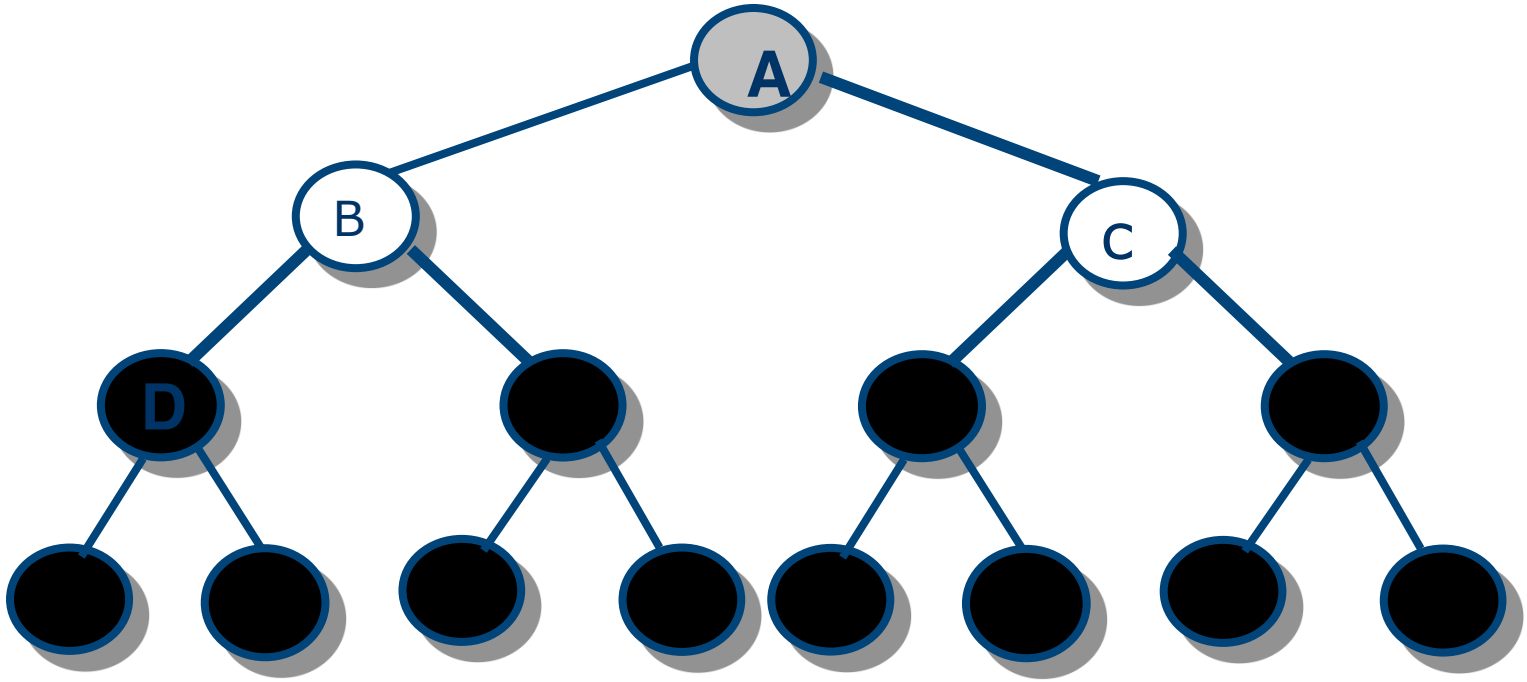
1º en Anchura. Ejemplo Genérico



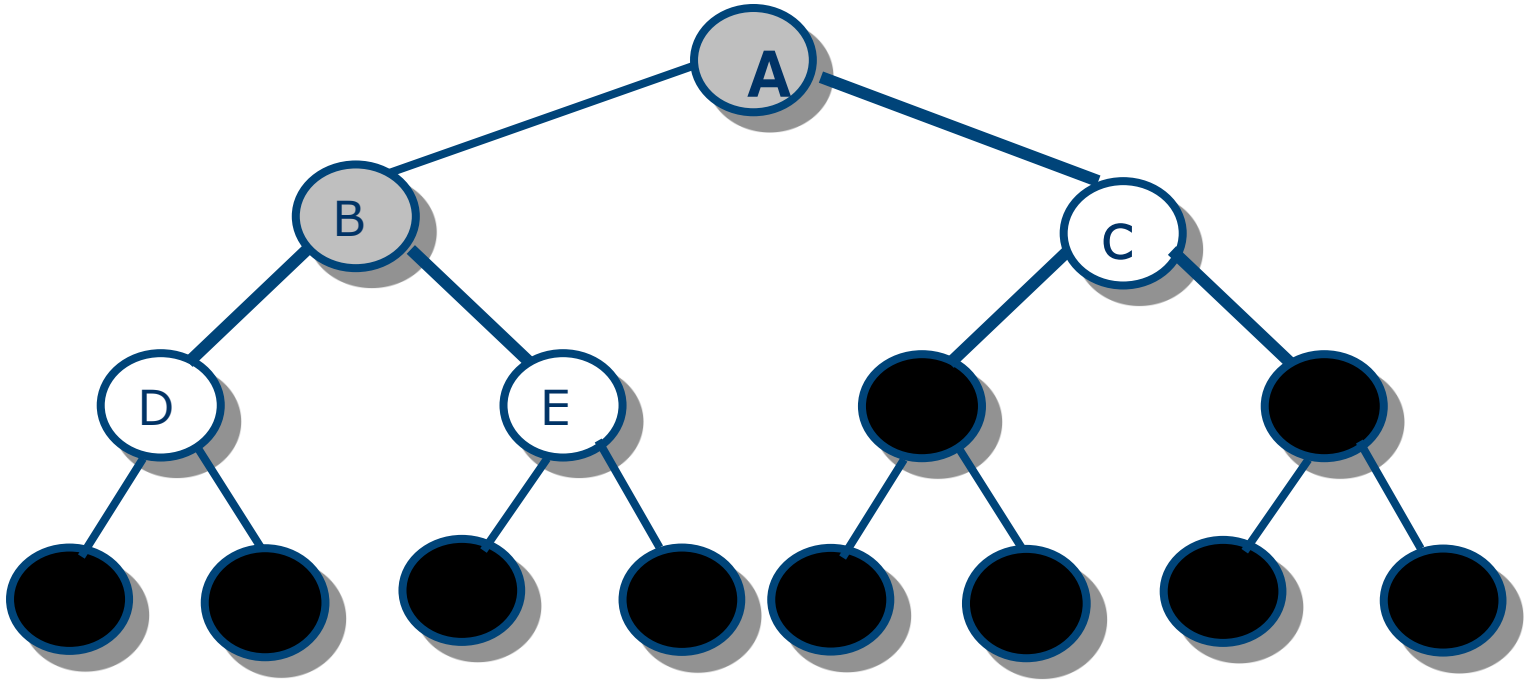
1º en Anchura. Ejemplo Genérico



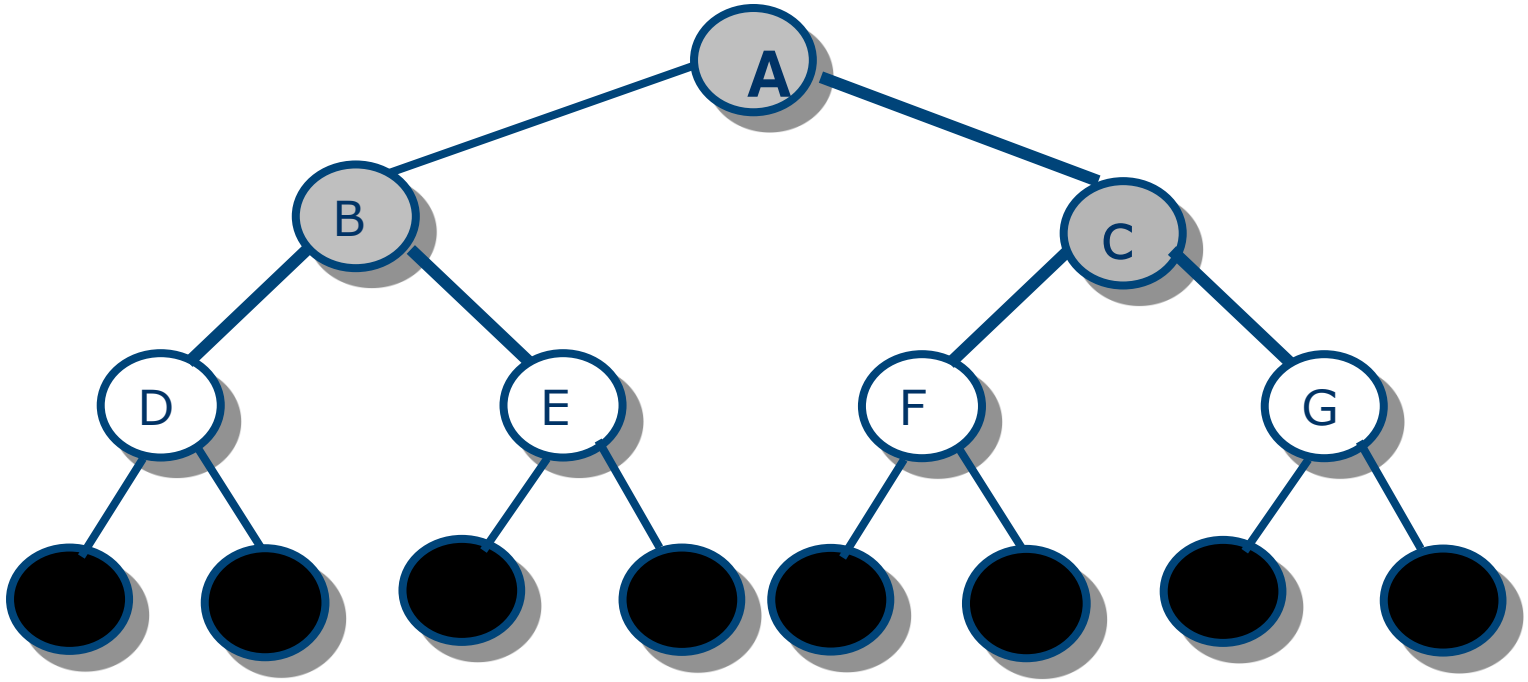
1º en Anchura. Ejemplo Genérico



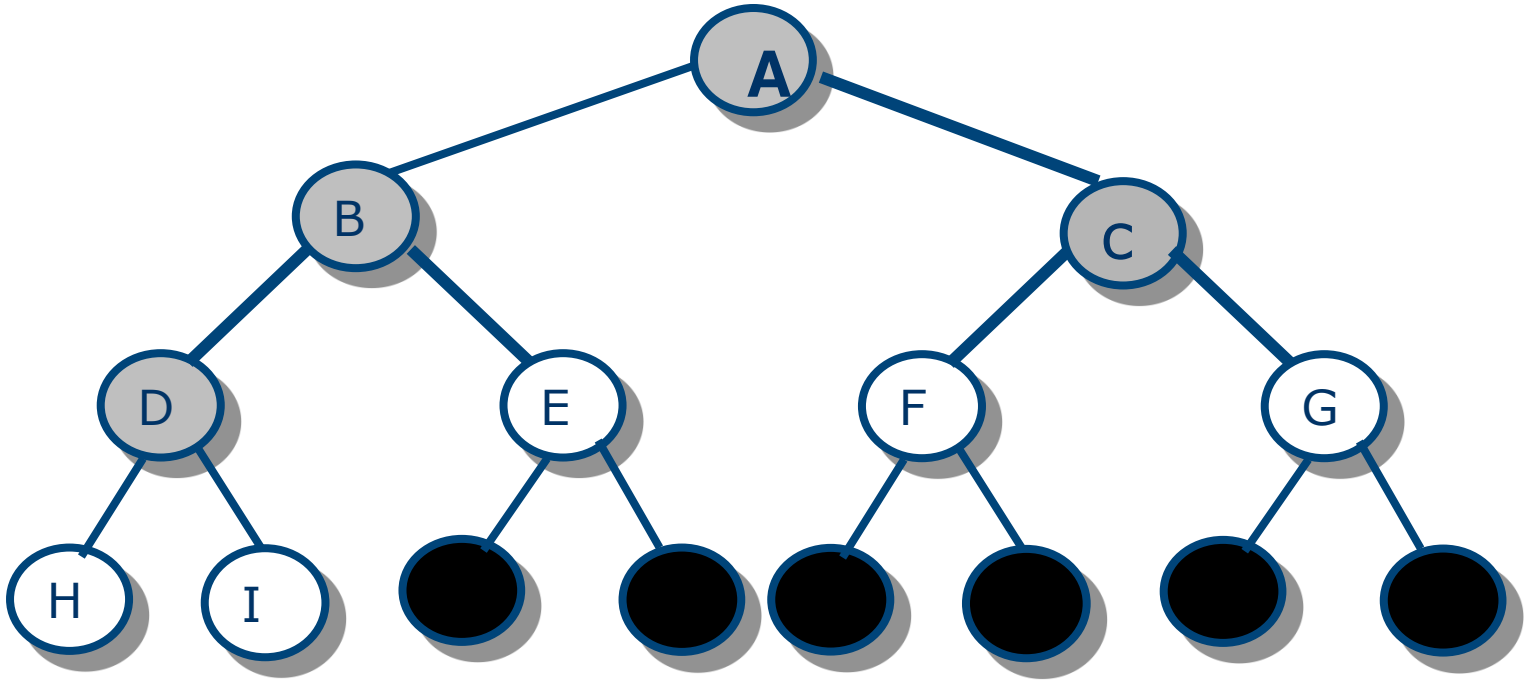
1º en Anchura. Ejemplo Genérico



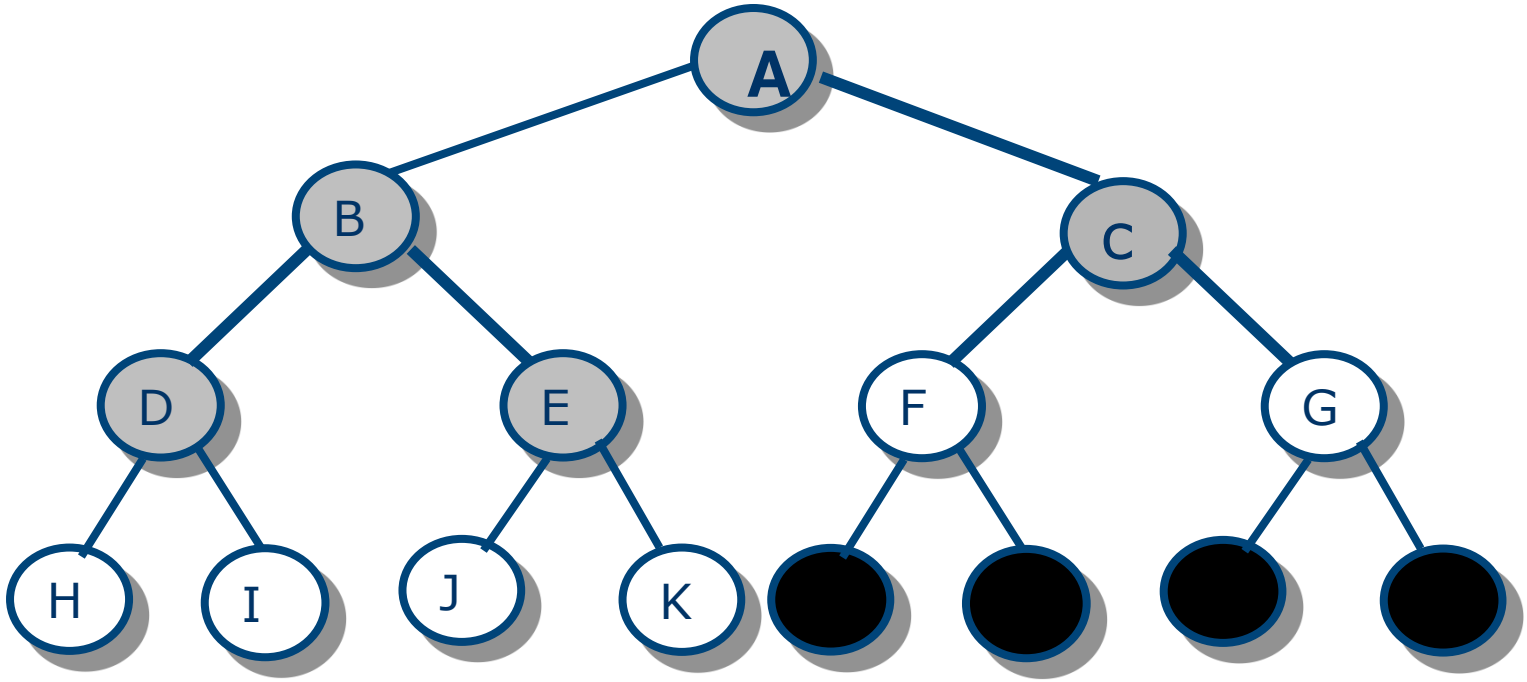
1º en Anchura. Ejemplo Genérico



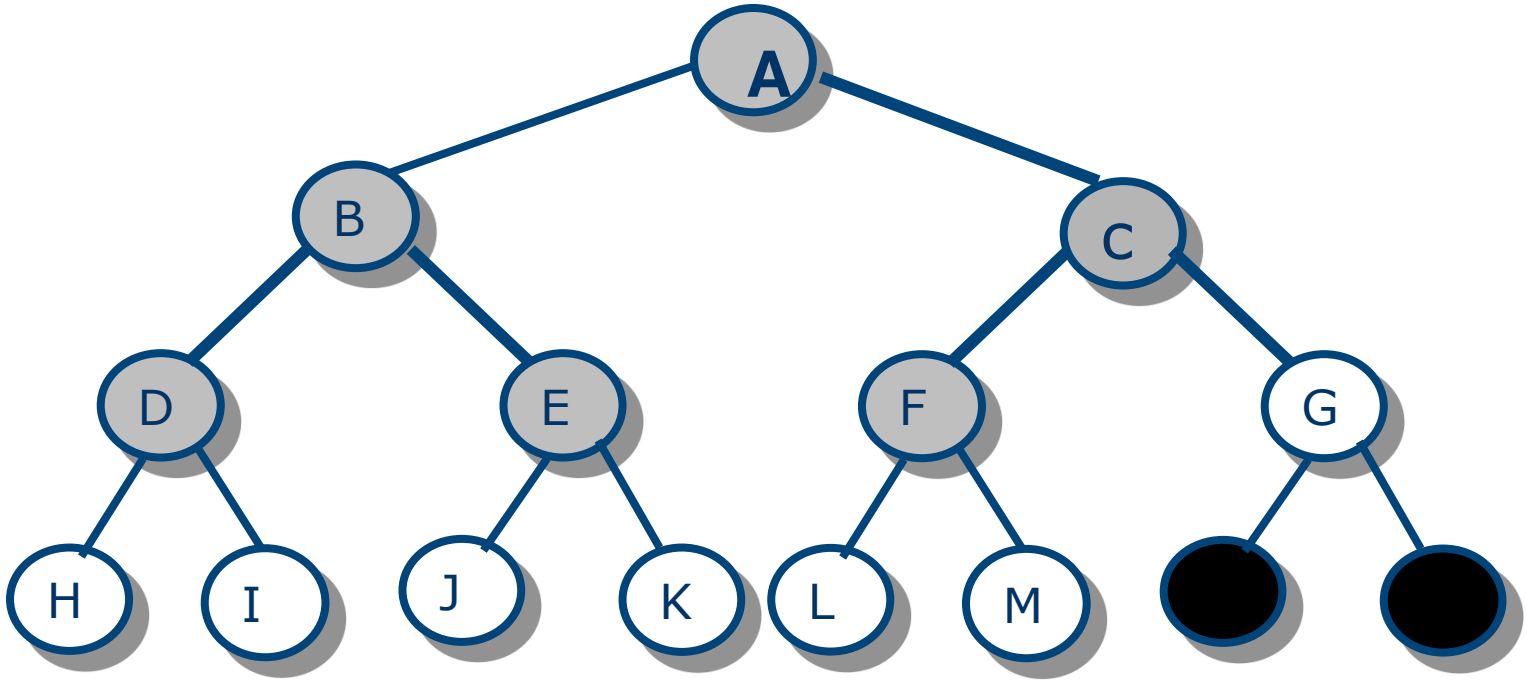
1º en Anchura. Ejemplo Genérico



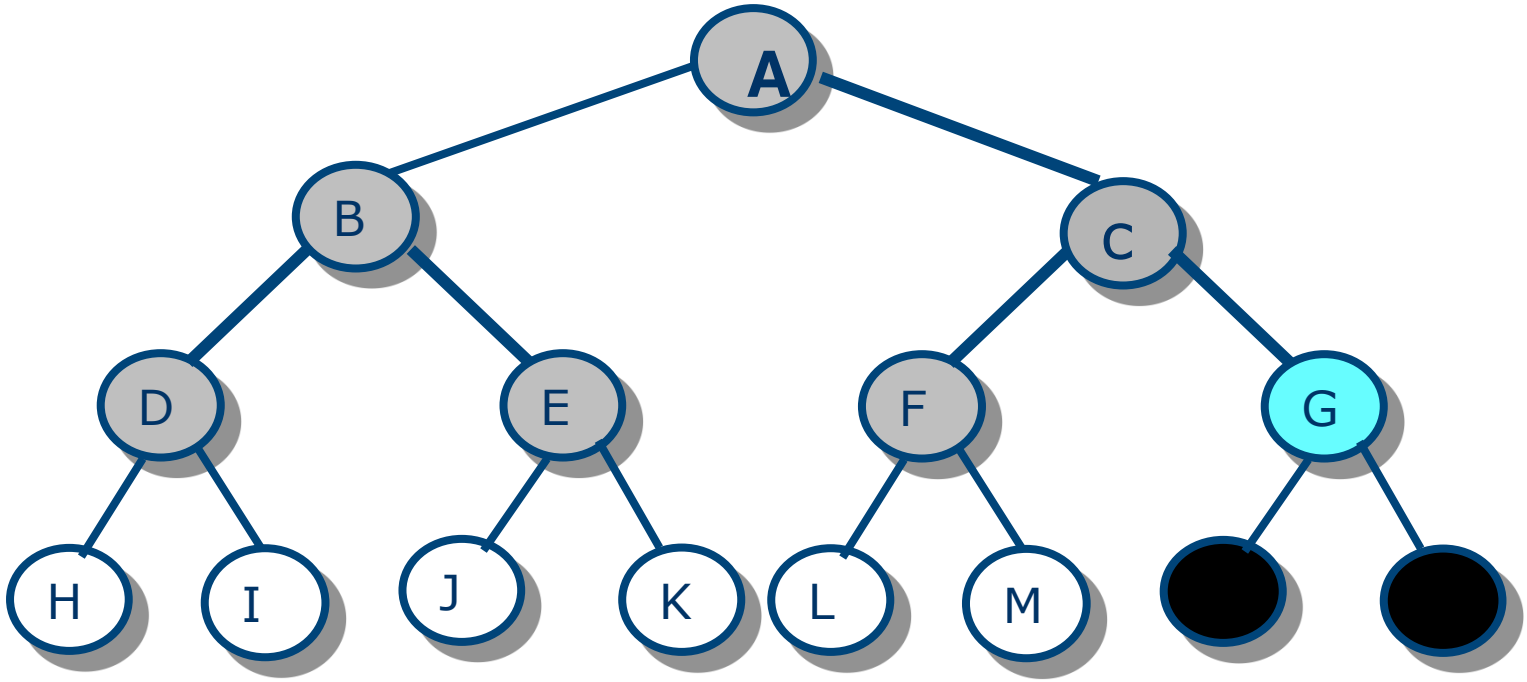
1º en Anchura. Ejemplo Genérico



1º en Anchura. Ejemplo Genérico



1º en Anchura. Ejemplo Genérico

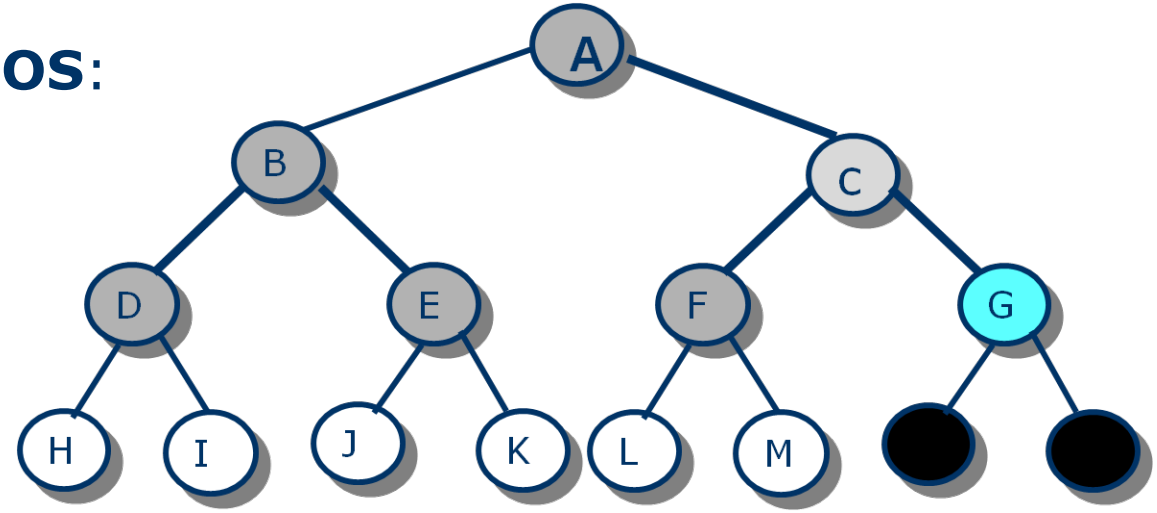


1º en Anchura. Ejemplo Genérico

Solución:
A, C, G

NODOS GENERADOS:
A,B,C,D,E,F,G,H,
I, J,K, L, M

NODOS VISITADOS:
A, B, C, D, E, F, G



Medidas del Rendimiento

Completa: la estrategia siempre que exista, encontrará una solución

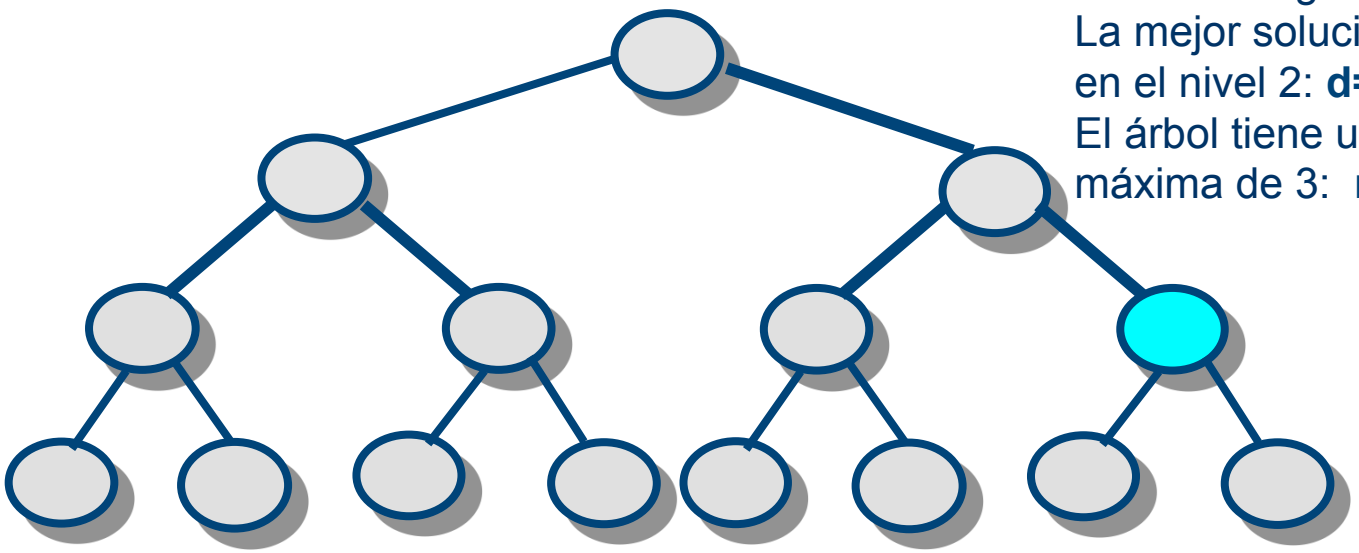
Óptima: la estrategia siempre que exista solución, encontrará primero la mejor solución

Complejidad en tiempo: número de nodos generados durante la búsqueda

Complejidad en espacio: máximo número de nodos en memoria

Medidas del Rendimiento

- **b**: Factor de ramificación, número de nodos generados a partir de cada nodo (máximo número de sucesores de cualquier nodo)
- **d**: profundidad de la solución óptima
- **m**: máxima profundidad del árbol



Cada nodo genera 2 nodos: **b=2**
La mejor solución se encuentra en el nivel 2: **d=2**
El árbol tiene una profundidad máxima de 3: **m=3**

1º en Anchura.
Medidas del Rendimiento

Completa: ¿encuentra una solución?

- Sí, es Completa. Siempre que exista solución, la encontrará en un nº finito de pasos

Óptima: ¿encuentra la solución óptima?

- Sí cuando el coste del camino a la solución es una función no decreciente de la profundidad del nodo (p.e. cuando todas las acciones tienen el mismo coste)

1º en Anchura.
Medidas del Rendimiento

Complejidad en tiempo: número de nodos generados

- La raíz genera b nodos, cada uno genera b nodos, … :

$$1 + b + b^2 + \ldots$$

- Si la solución está a profundidad **d**, en el peor de los casos se han de generar todos excepto el último nodo en el nivel d (el objetivo no se expande), por tanto:

$$1 + b + b^2 + \ldots + b^d + (b^{d+1} - b) \qquad \mathbf{O(b^{d+1})}$$

Complejidad en espacio: máximo nº de nodos en memoria

- Si la solución está a profundidad **d**, cada nodo generado debe permanecer en memoria, por tanto en memoria habrá:

$$(1 + b^{d+1} - b) \qquad \mathbf{O(b^{d+1})}$$

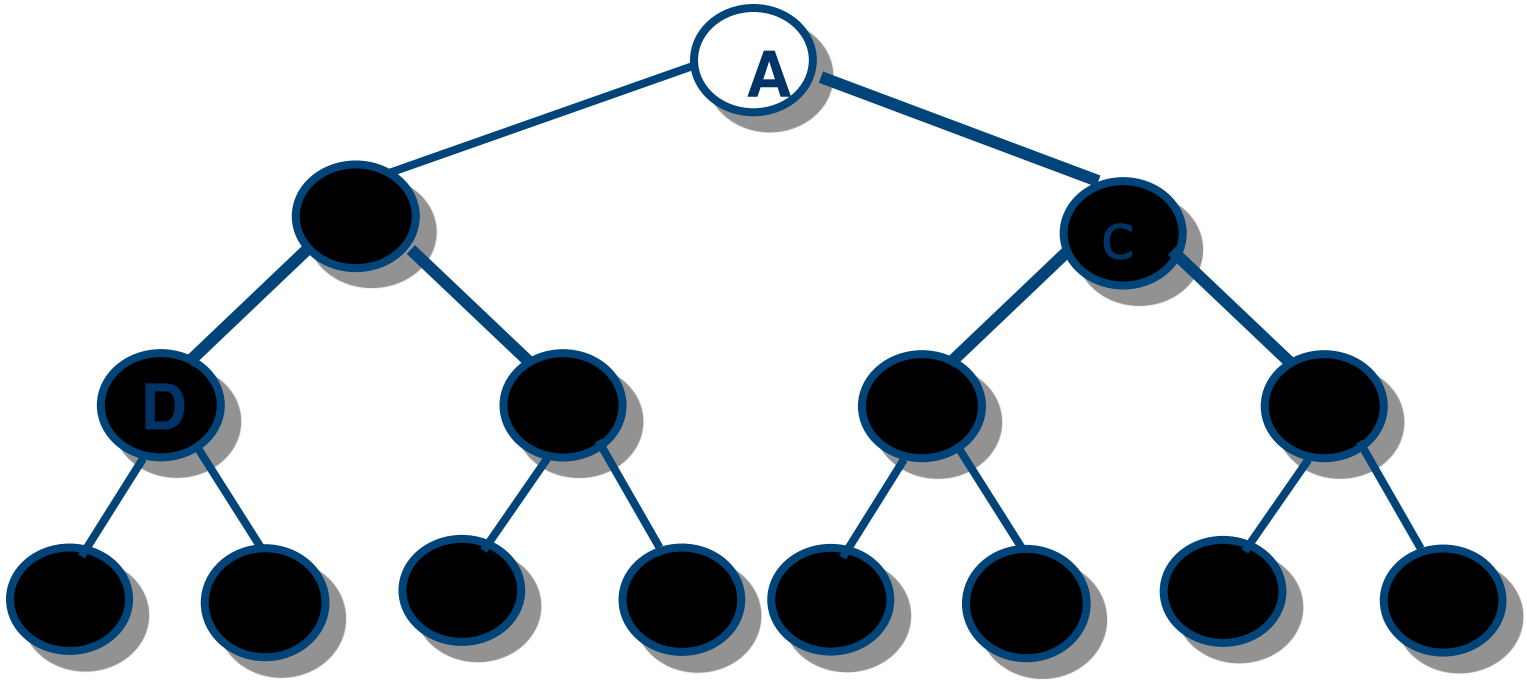
1º en Anchura. Análisis

- El requerimiento de memoria es más importante que el tiempo que tarda en encontrar la solución
- Si existe solución no entrará en bucles infinitos, encontrará la solución
- Ineficaz cuando explora la misma zona del espacio de estados en numerosas ocasiones

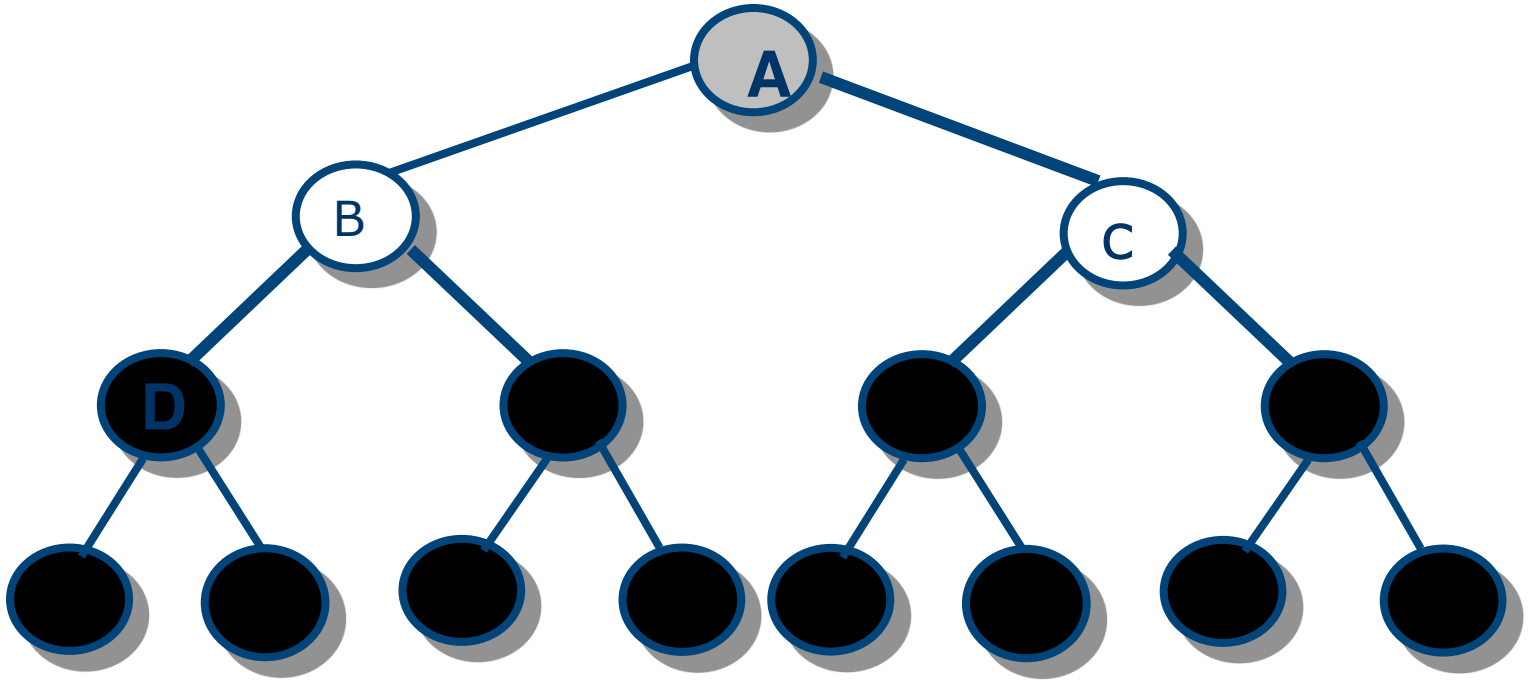
Búsqueda 1º en Profundidad

- Siempre expande el nodo más profundo
- Sólo cuando la búsqueda encuentra un nodo que no se puede expandir se retrocede para expandir el siguiente nodo más profundo
- Estrategia = seleccionar el nodo más reciente para la expansión (LIFO)

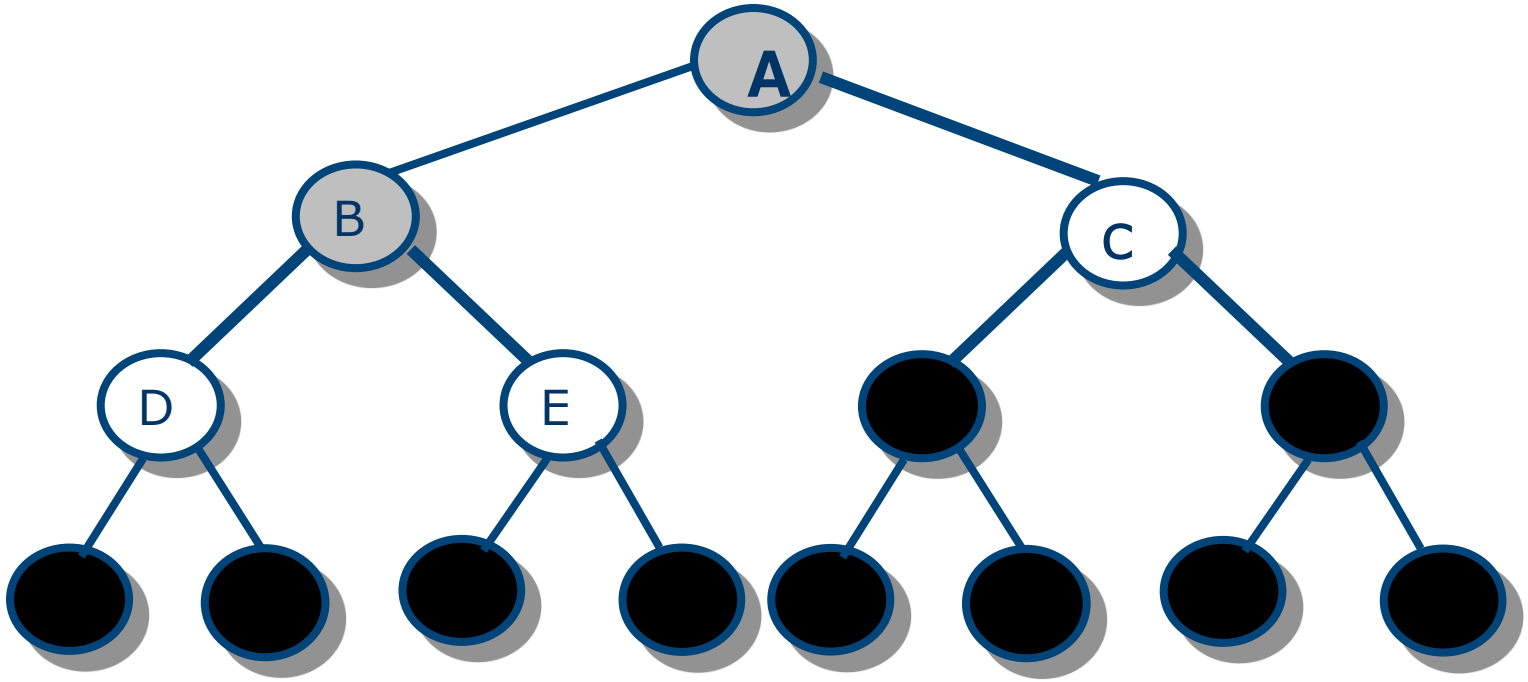
1º en Profundidad. Ejemplo Genérico



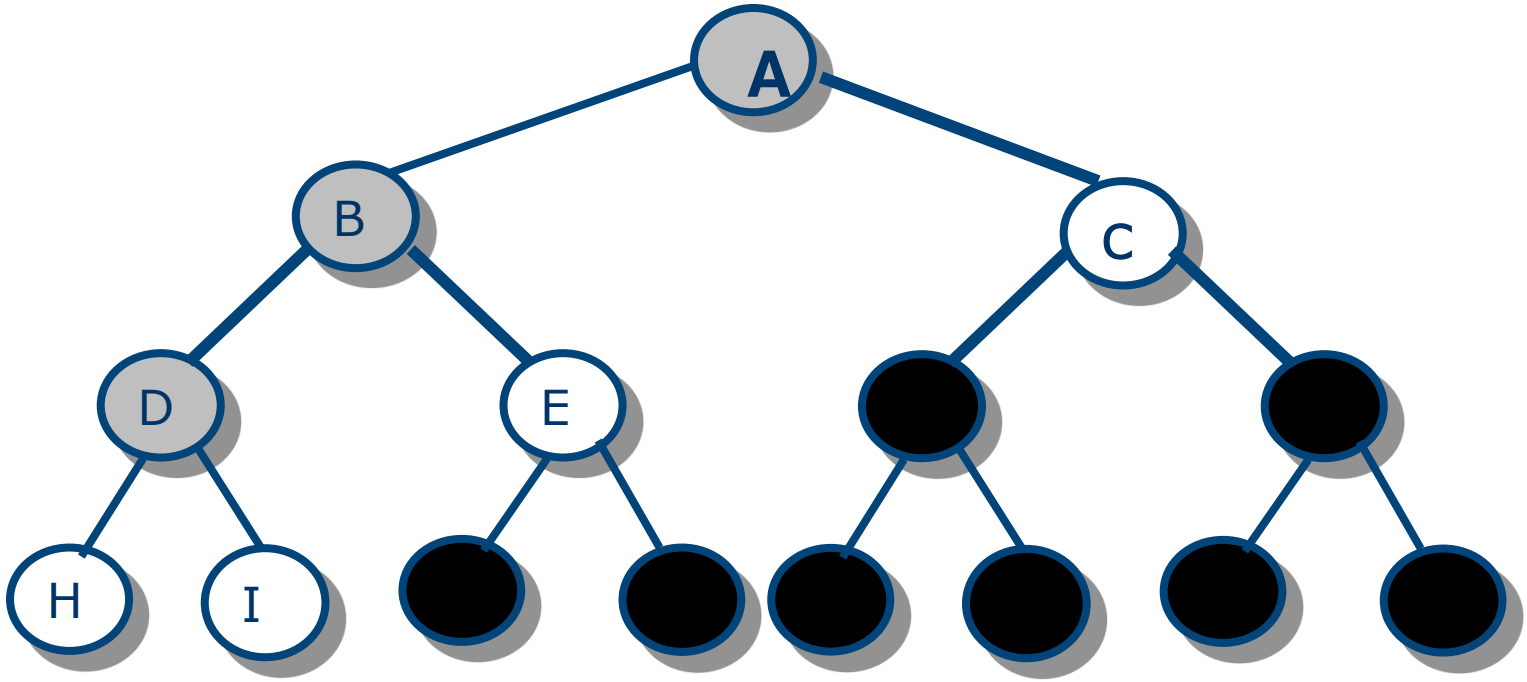
1º en Profundidad. Ejemplo Genérico



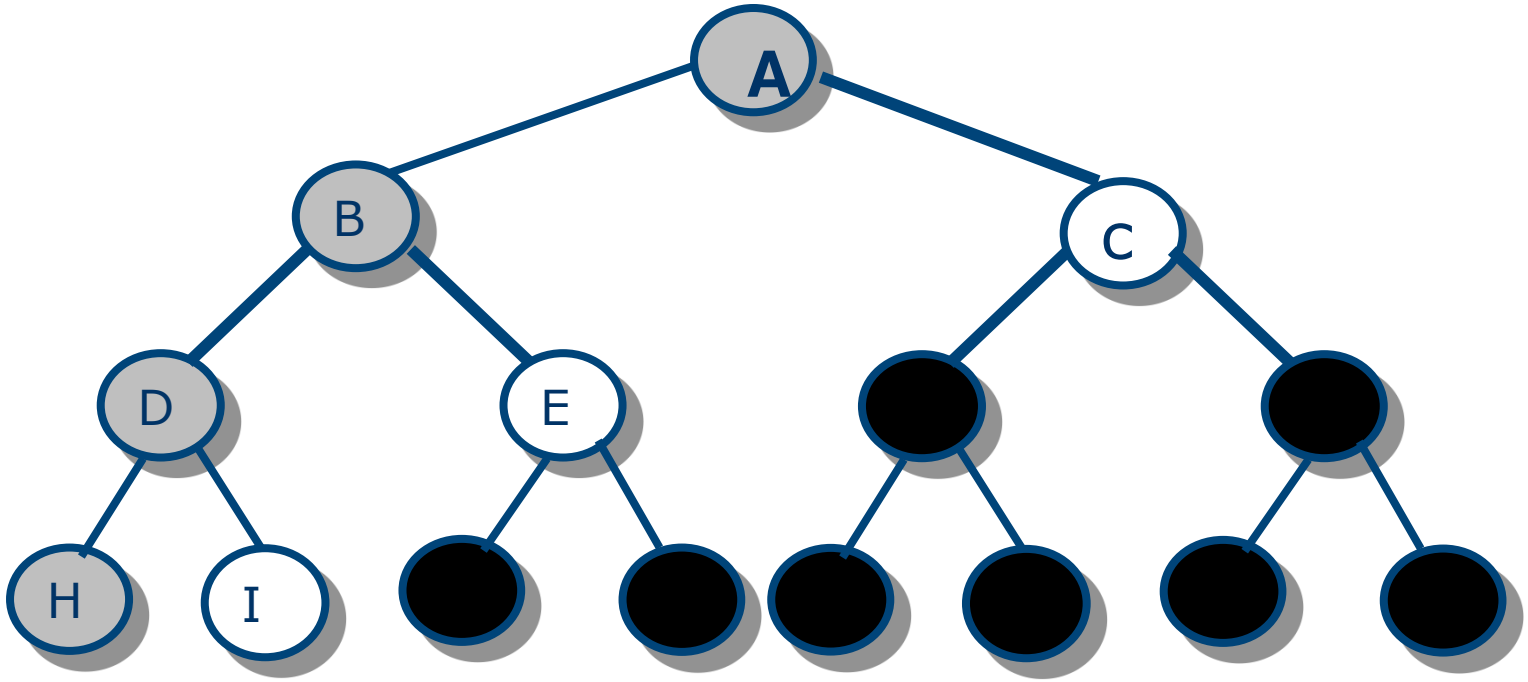
1º en Profundidad. Ejemplo Genérico



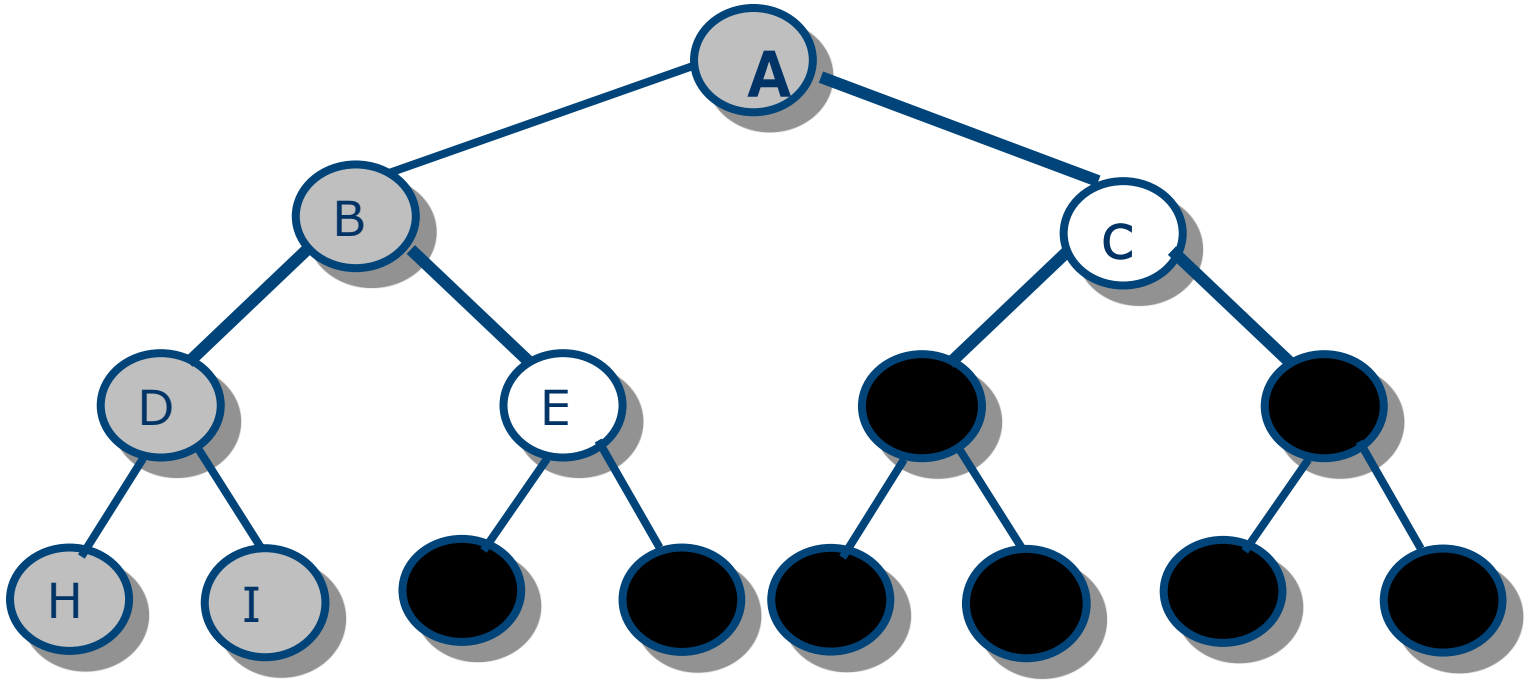
1º en Profundidad. Ejemplo Genérico



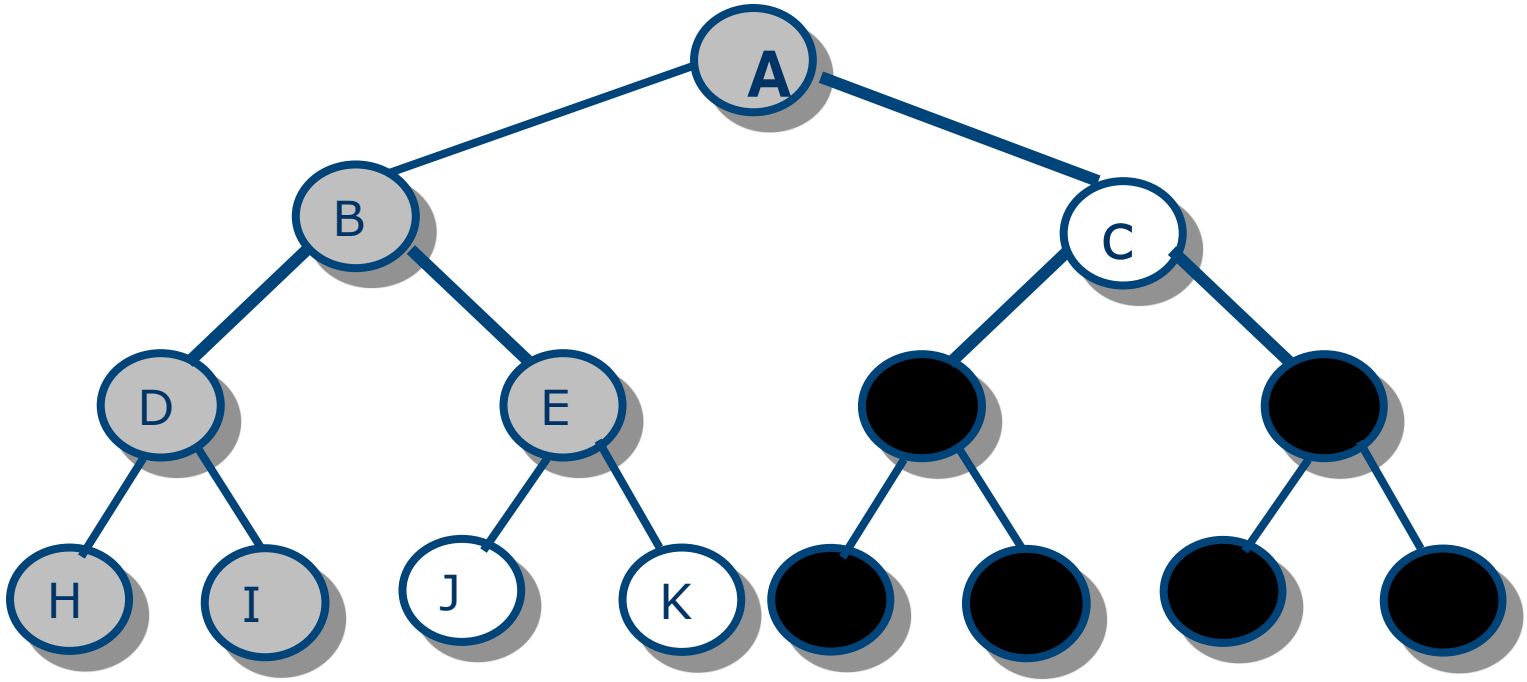
1º en Profundidad. Ejemplo Genérico



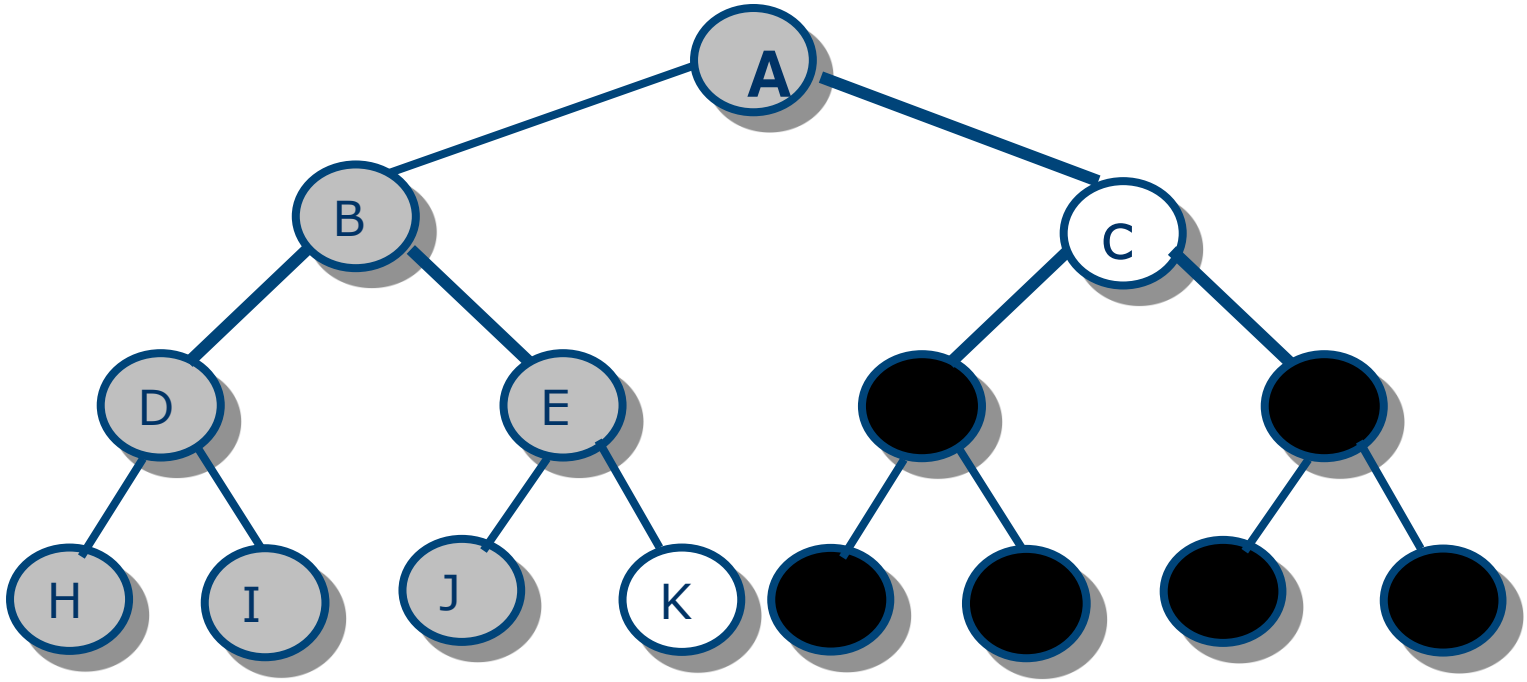
1º en Profundidad. Ejemplo Genérico



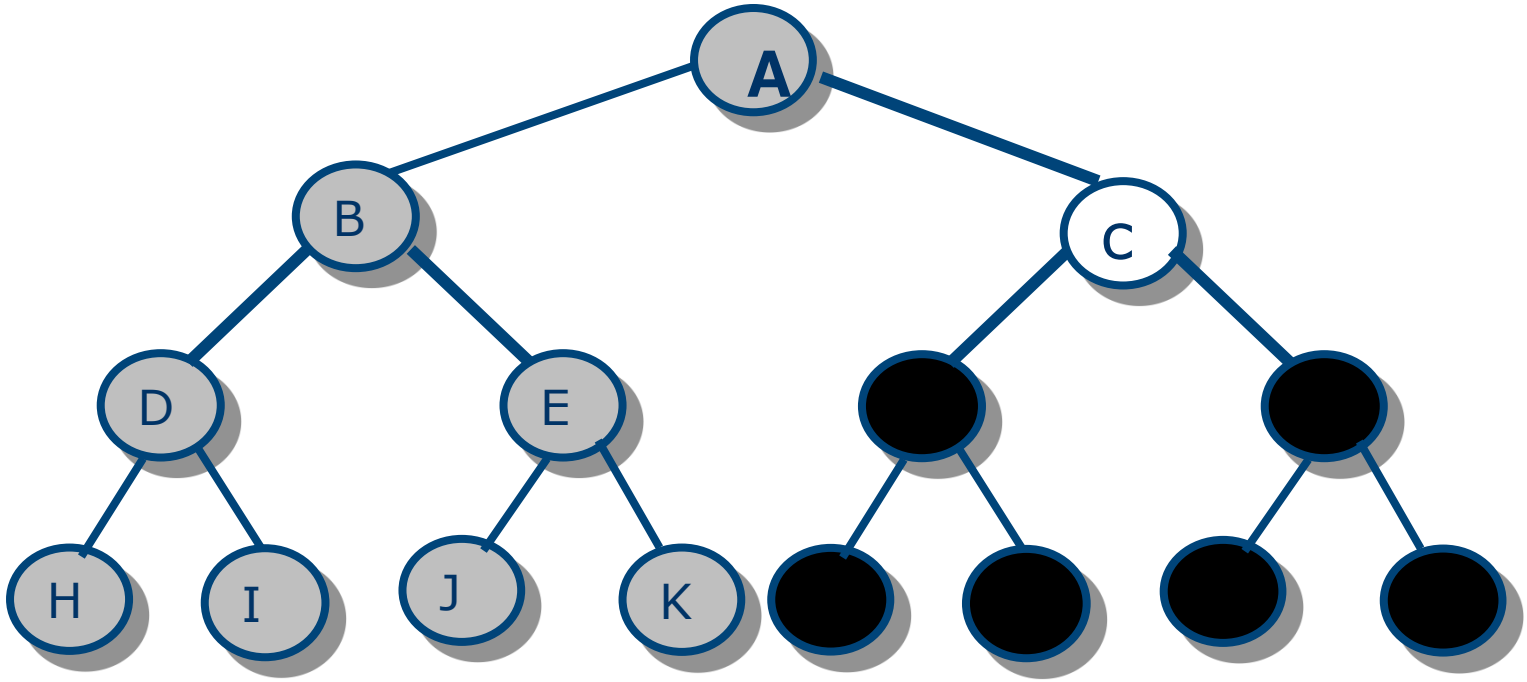
1º en Profundidad. Ejemplo Genérico



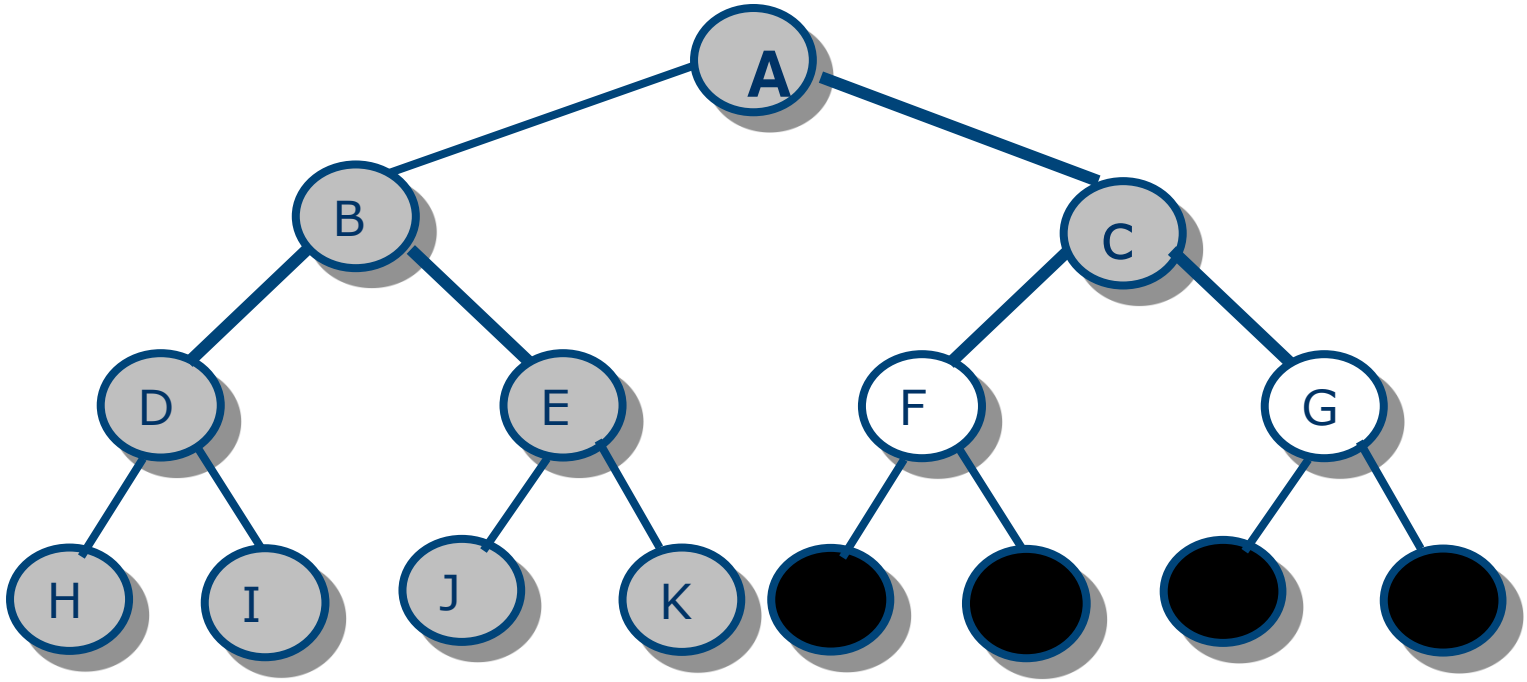
1º en Profundidad. Ejemplo Genérico



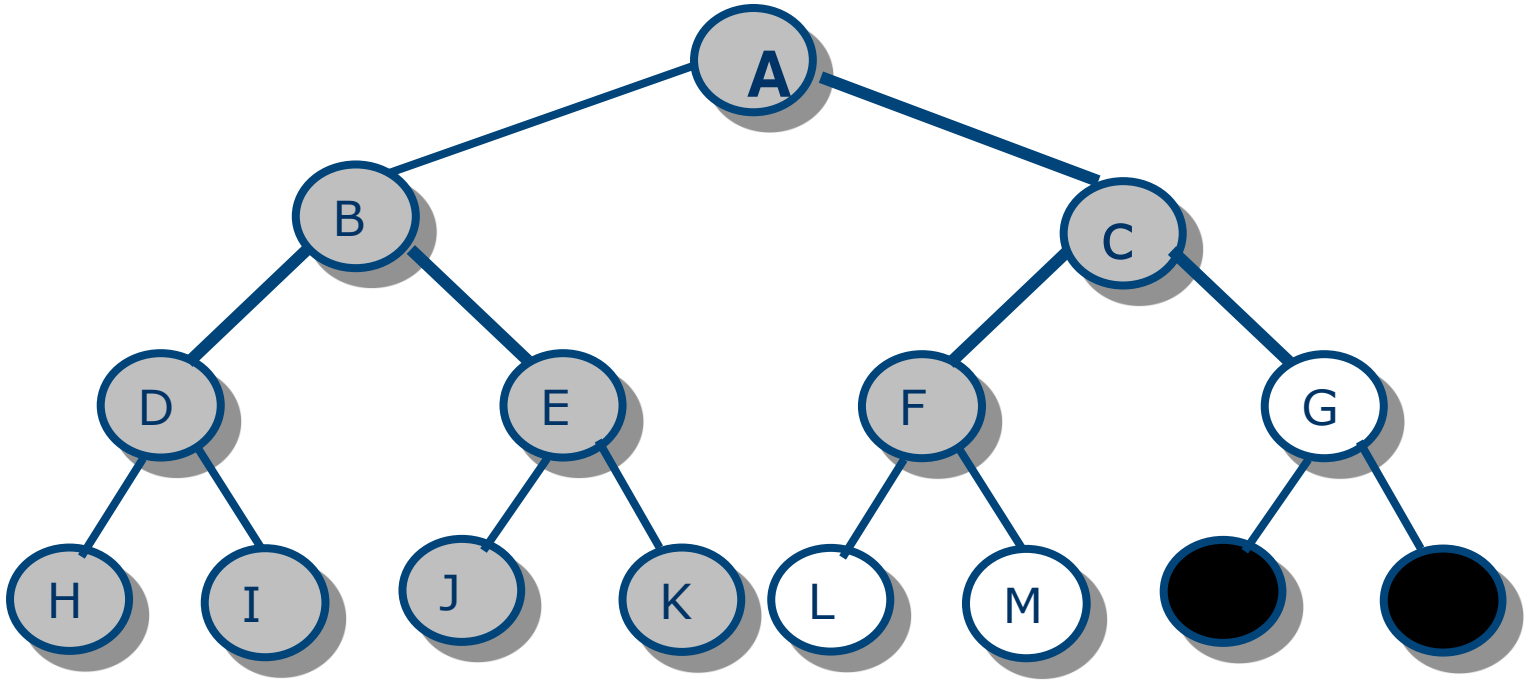
1º en Profundidad. Ejemplo Genérico



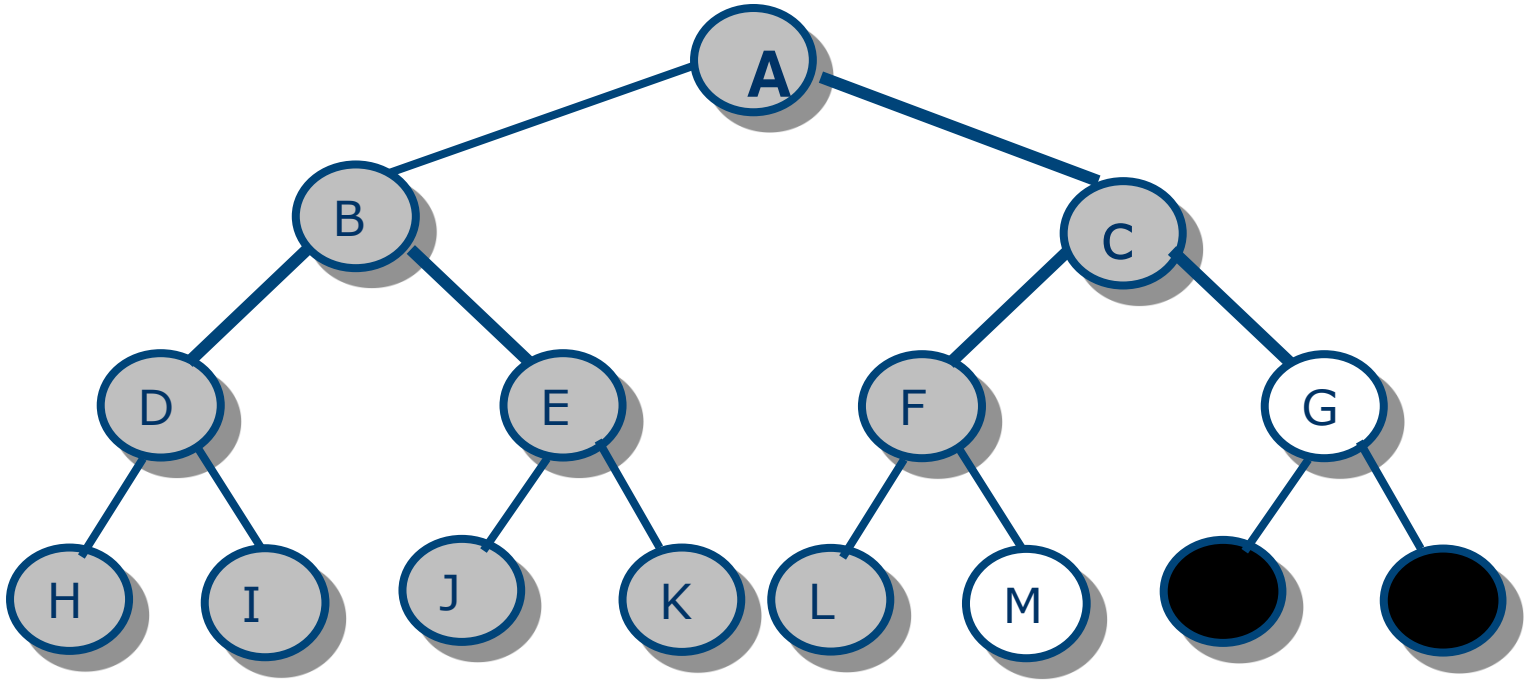
1º en Profundidad. Ejemplo Genérico



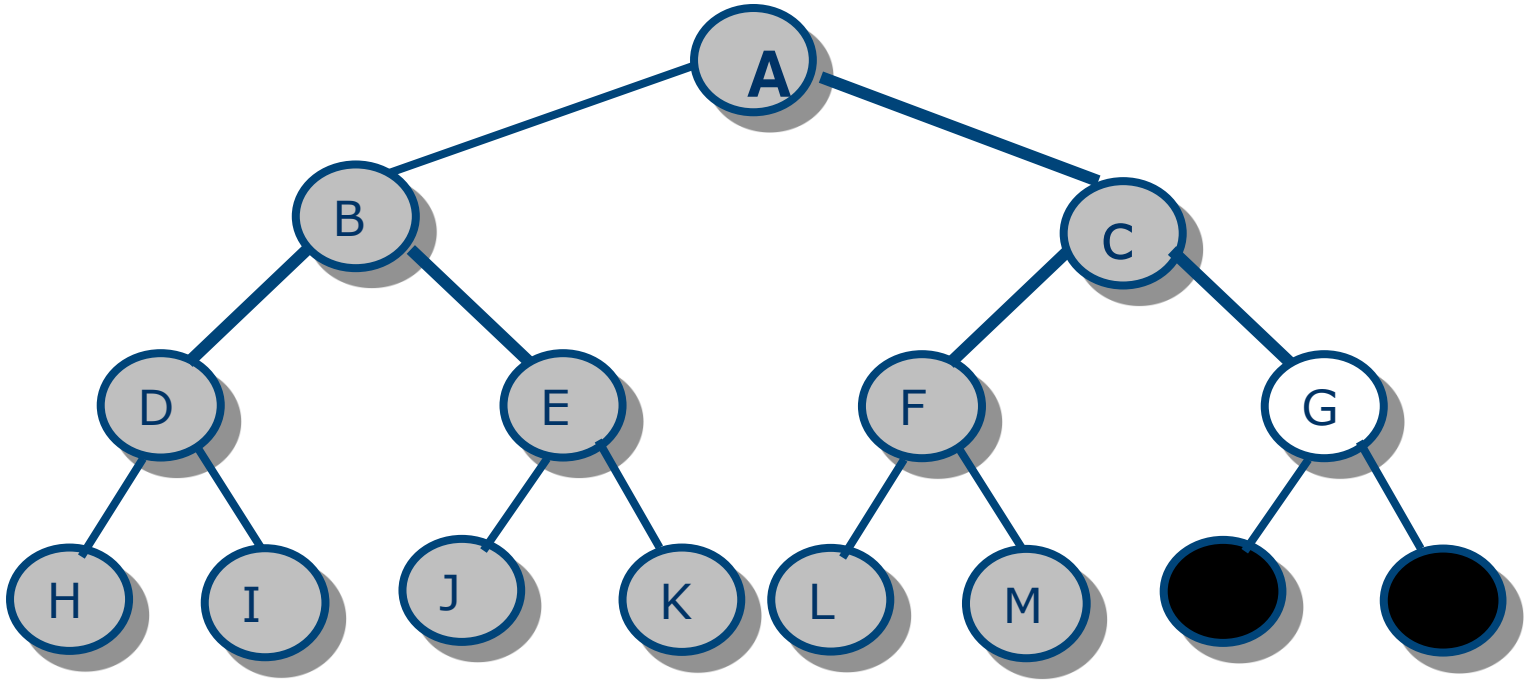
1º en Profundidad. Ejemplo Genérico



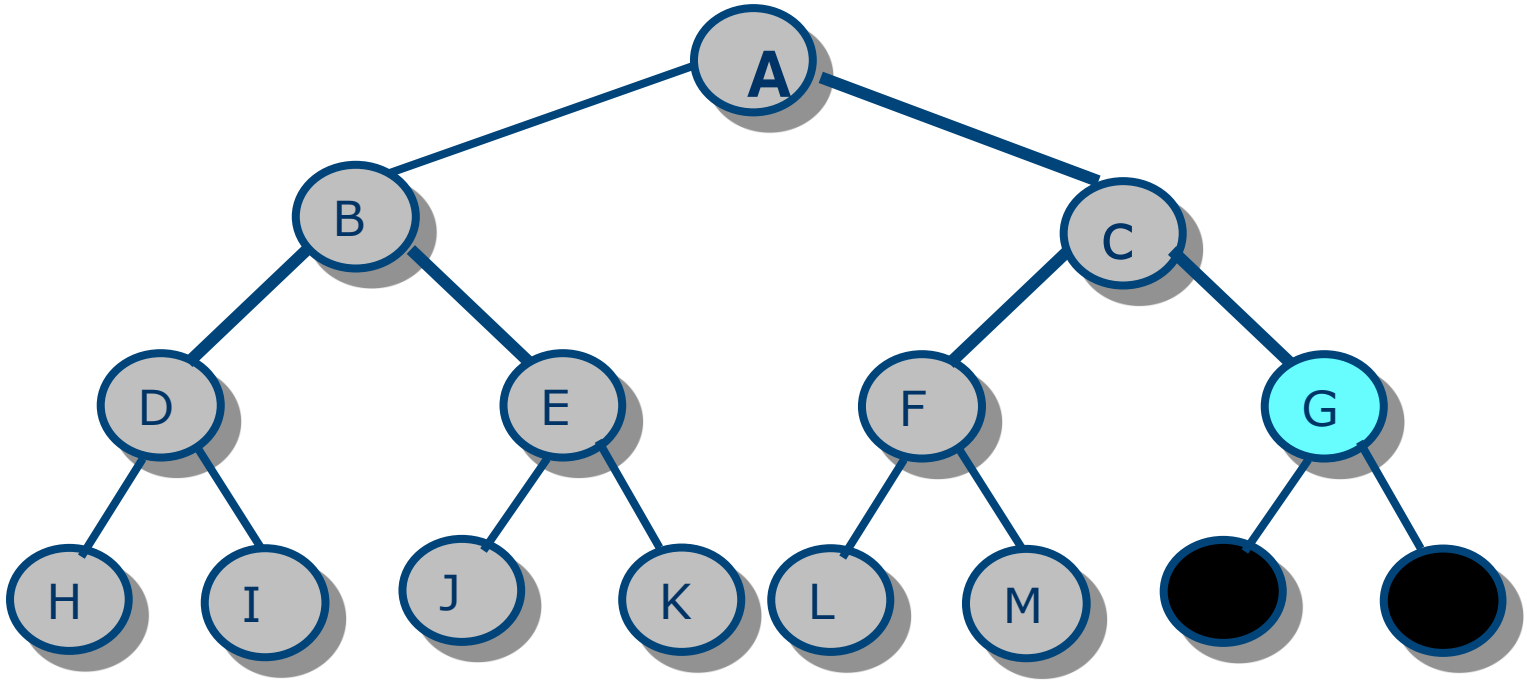
1º en Profundidad. Ejemplo Genérico



1º en Profundidad. Ejemplo Genérico



1º en Profundidad. Ejemplo Genérico

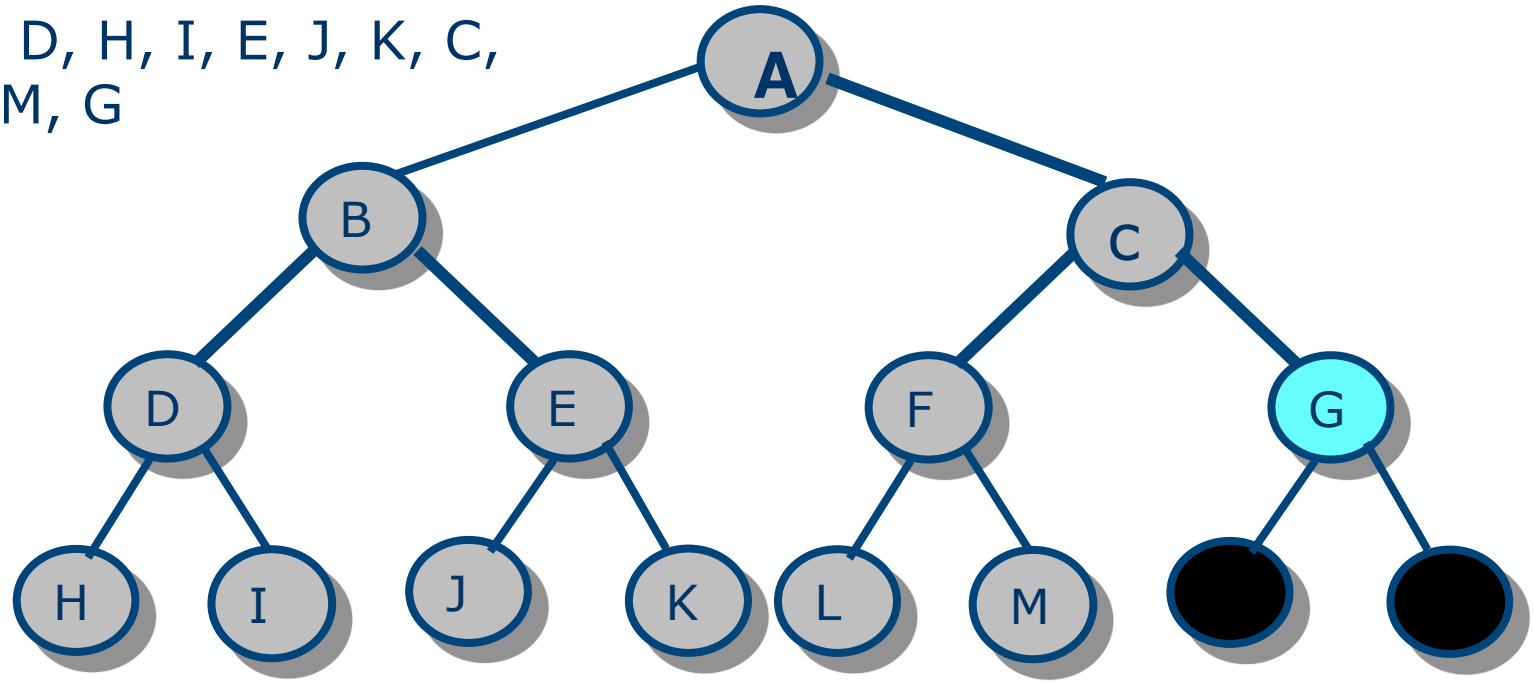


1º en Profundidad. Ejemplo Genérico

Solución:
A, C, G

NODOS GENERADOS:
A, B, C, D, E, H, I, J, K,
F, L, M, G

NODOS VISITADOS:
A, B, D, H, I, E, J, K, C,
F, L, M, G



1º en Profundidad. Medidas del Rendimiento

Completa: ¿encuentra una solución?

- No, no es Completa. Puede perderse en un bucle infinito por una rama por la que no haya solución.

Óptima: ¿encuentra la solución óptima?

- No, no es Óptima. No garantiza que la solución encontrada sea la óptima.

1º en Profundidad. Medidas del Rendimiento

Complejidad en tiempo: número de nodos generados

- La raíz genera b nodos, cada uno genera b nodos, … :

$$1 + b + b^2 + \ldots$$

- Si el árbol tiene una profundidad **m**, en el peor de los casos se ha de llegar hasta la profundidad máxima del árbol, por tanto: **$O(b^m)$**

Complejidad en espacio: máximo nº de nodos en memoria

- Sólo se necesitan almacenar (b-1)*m + 1 nodos por tanto: **O(b·m)**

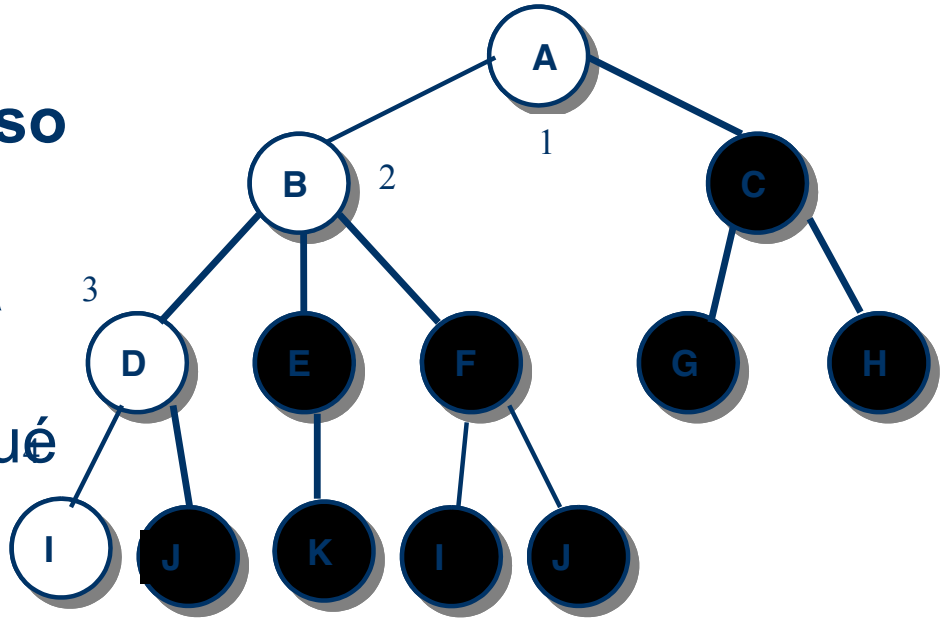
1º en Profundidad
Análisis del Algoritmo

- No es completo (cuando el árbol de búsqueda es infinito)
- Puede caer fácilmente en bucles
- No es óptimo
- Almacena camino desde el nodo raíz a un nodo hoja, junto con los nodos generados pero no expandidos
- Puede encontrar una solución rápidamente

1º en Profundidad. Variantes

Búsqueda con Retroceso

- Sólo se genera un sucesor a la vez, cada nodo parcialmente expandido recuerda qué sucesor se expande a continuación.
- Complejidad en espacio:
  O(m)
- No apto para grandes descripciones de estados

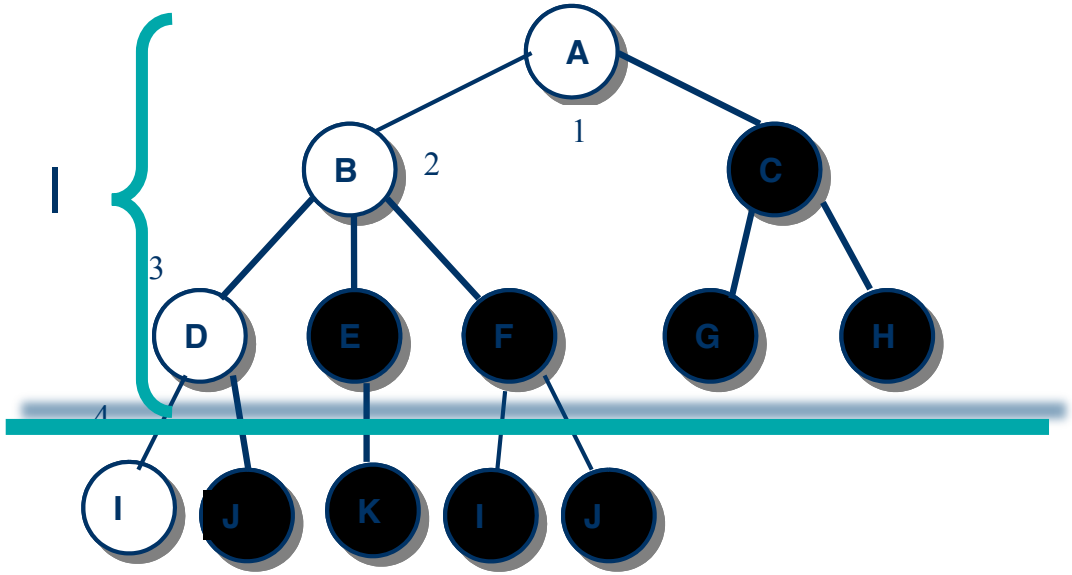


Menos Memoria

1º en Profundidad. Variantes

Búsqueda de Profundidad Limitada

- Poner un límite a la profundidad
  - Incompleta cuando l<d
  - No óptima si l>d
  - Resuelve el problema de caminos infinitos



Evita Caminos Infinitos

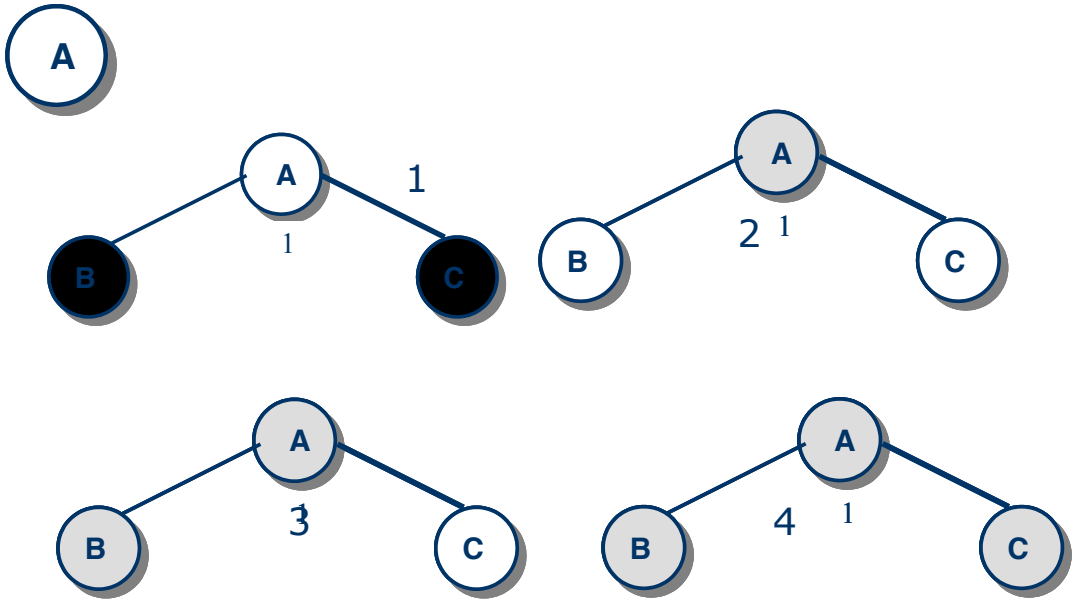
1º en Profundidad. Variantes

Profundidad Iterativa

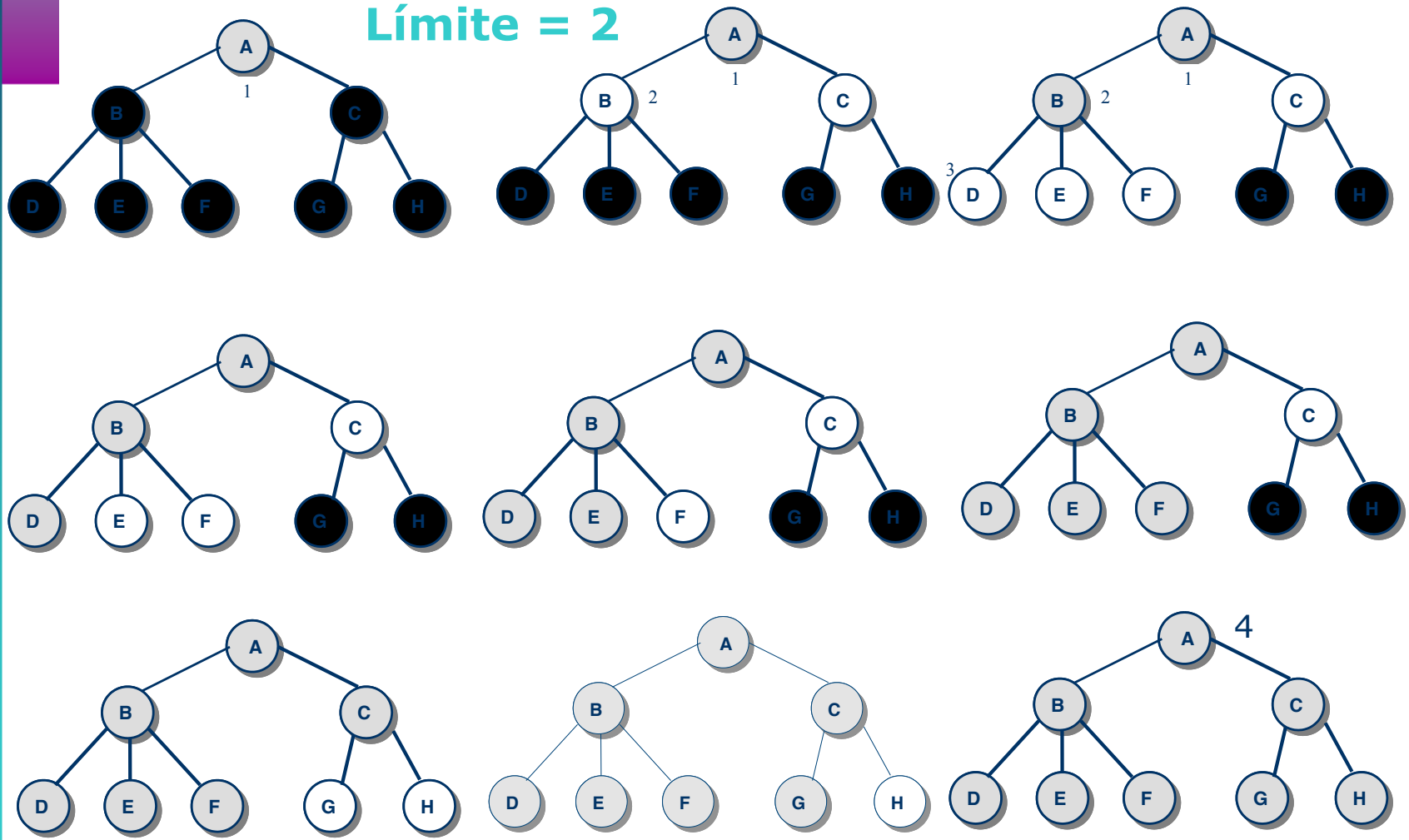
- Aumenta gradualmente el límite de profundidad hasta encontrar un objetivo
  - Ventajas de la Búsqueda en Anchura y en Profundidad

Límite = 0

Límite = 1



1º en Profundidad, de profundidad iterativa



Búsqueda Bidireccional

- Ejecutar dos búsquedas simultáneas: una hacia delante desde el estado inicial y la otra hacia atrás desde el objetivo, parando cuando las dos búsquedas se encuentren en un estado.
- **Función Predecesor**
- **Búsqueda en Anchura** de al menos una de las búsquedas, para garantizar la convergencia del algoritmo.
- **Complejidad Temporal:** $O(b^{d/2})$
- **Complejidad Espacial:** $O(b^{d/2})$

Comparación de Complejidad

Criterio	1º Anchura	1º Profundidad	Profundidad Limitada	Profundidad Iterativa	Bidireccional
Óptima	Sí	No	No	Sí	Sí
Completa	Sí	No	Sí, si l≥d	Sí	Sí
Espacio	b^{d+1}	$b \bullet m$	b^{l}	b^{d}	$b^{d/2}$
Tiempo	b^{d+1}	b^{m}	b^{l}	b^{d}	$b^{d/2}$

b = factor de ramificación
d = profundidad de la solución
m = máxima profundidad
l = límite de profundidad

Evitar estados repetidos

- No volver a un estado del que justo se viene
- No crear caminos cíclicos
- No crear un estado que ya ha sido creado antes
  - Aumenta la efectividad del método
  - Reduce el tamaño del espacio de estados
  - Aumenta el coste computacional

CERRADOS: nodos expandidos

ABIERTOS: nodos generados no expandidos

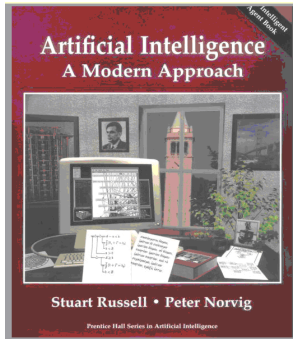
Si el nodo actual se encuentra en Cerrados se elimina en vez de expandirse

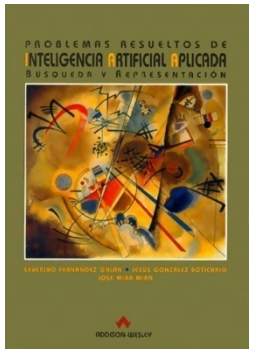
Referencias

- Russell & Norvig : Capítulo 3

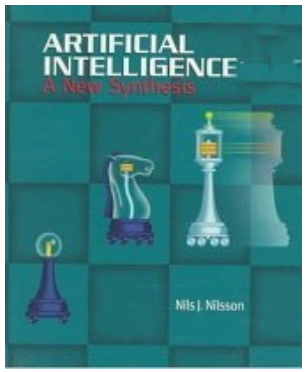
Realiza un repaso a las técnicas de búsqueda no informada, analizando el rendimiento de cada solución y planteando alternativas menos costosas. Además plantea el tratamiento de los estados repetidos, el cual se puede aplicar a cualquier estrategia seleccionada.

Referencias Bibliográficas





- **Inteligencia Artificial: Un Enfoque Moderno.** S. Russell y P. Norvig, 2005
- **Problemas Resueltos de IA Aplicada. Búsqueda y Representación.** Fernández et al. (2003)



- **Aspectos básicos de la Inteligencia Artificial**. Mira et al. , 2003