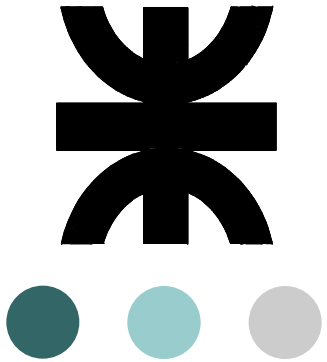


Paradigmas de Programación

Programación Lógica: Introducción

Universidad Tecnológica Nacional
Facultad Regional Resistencia

UTN
FRRe

Contenidos

- ***Introducción***
  - ***Problema***
  - ***Sistemas de Producción***
  - ***Búsqueda***
- Programación con Lógica Proposicional
  - Sintaxis
  - Consecuencia lógica
  - Resolución
  - Refutación y Deducción
  - Árbol de Resolución
  - Negación
  - Resolución SLD
- Programación con Lógica de Predicados
  - Sintaxis
  - Cláusulas
  - Unificación
  - Resolución
- ProLog
  - Ejemplos

Introducción

Cómo hacer para alcanzar un objetivo cuando no hay un algoritmo sistemático o directo?

Probar todas las alternativas posibles, mediante ensayo y error, con la esperanza de hallar en algún momento la solución.

Cada acción adoptada abre nuevas posibilidades, una especie de ramificación, denominada árbol de búsqueda.

Programación Convencional vs. programación Lógica

Programación Convencional	**Programación Lógica**
Algoritmos (pasos de solución explícitos)	Búsqueda (pasos de solución implícitos) Explosión Combinatoria!!
Información y control integrados	Estructura de control separados del conocimiento
Dificultad de Modificación	Facilidad de Modificación
Requerimiento de respuestas optimas	Aceptación de respuestas satisfactorias

Problemas

- Una persona se enfrenta con **un problema** cuando tienen que llegar a un objetivo y no conoce la acción o serie de acciones que debe seguir para conseguirlo.

P = (I, O, C)

Problemas 2

- Diremos que el estado I´ es sucesor del estado I, si es alcanzable a partir de I mediante la aplicación de alguna secuencia de operadores.
- Llamaremos acción al resultado de aplicar el operador O_j a una expresión.
- El conjunto de todos los estados que pueden ser alcanzados aplicando operadores a partir del estado inicial, se denomina **Espacio de búsqueda o espacio de estados.**

Problemas: Solución

- Una secuencia de operadores $O_1 \ldots O_n$ constituye una solución a un problema si el resultado de su aplicación al estado inicial I, satisface la condición C.

Si $C(O_n(\ldots(O_1(I))))) \Rightarrow O_1 \ldots O_n$ es la solución de (I,O,C)

- Soluciones
  - Podrá haber una solución al problema
  - Podrá haber muchas soluciones
  - La solución **optima** (no existe otra solución que la mejore).
  - Puede ser **semi-optima.**

Problemas: Ejemplos

- **El problema del viajante de Comercio.**
- **Problema de la suma de subconjuntos.** **(Dado un conjunto S de enteros, ¿existe un subconjunto no vacío de S cuyos elementos sumen cero?)**
- **Torres de Hanoi**

- **15-Puzzle**
  - **Samuel Lloyd (1841 -1911)**

- **3 Misioneros y 3 Caníbales**

Problemas: Ejemplos

Problema de las Jarras

Se tienen dos jarras con capacidades de 3 y 4 litros respectivamente. Se desea dejar dos litros de agua en la jarra de 4 litros. Las jarras no tienen mediciones.

El *espacio de estado* puede ser descrito mediante un par de entero [X, Y], tal que X varía entre 0 y 4 e Y entre 0 y 3.

X e Y representan la cantidad de litros que contienen las jarras de capacidad 4 y 3 litros respectivamente.

Problema de las Jarras
Partes del Problema

P = (I, O, C)

Estado condición
[2, Z]

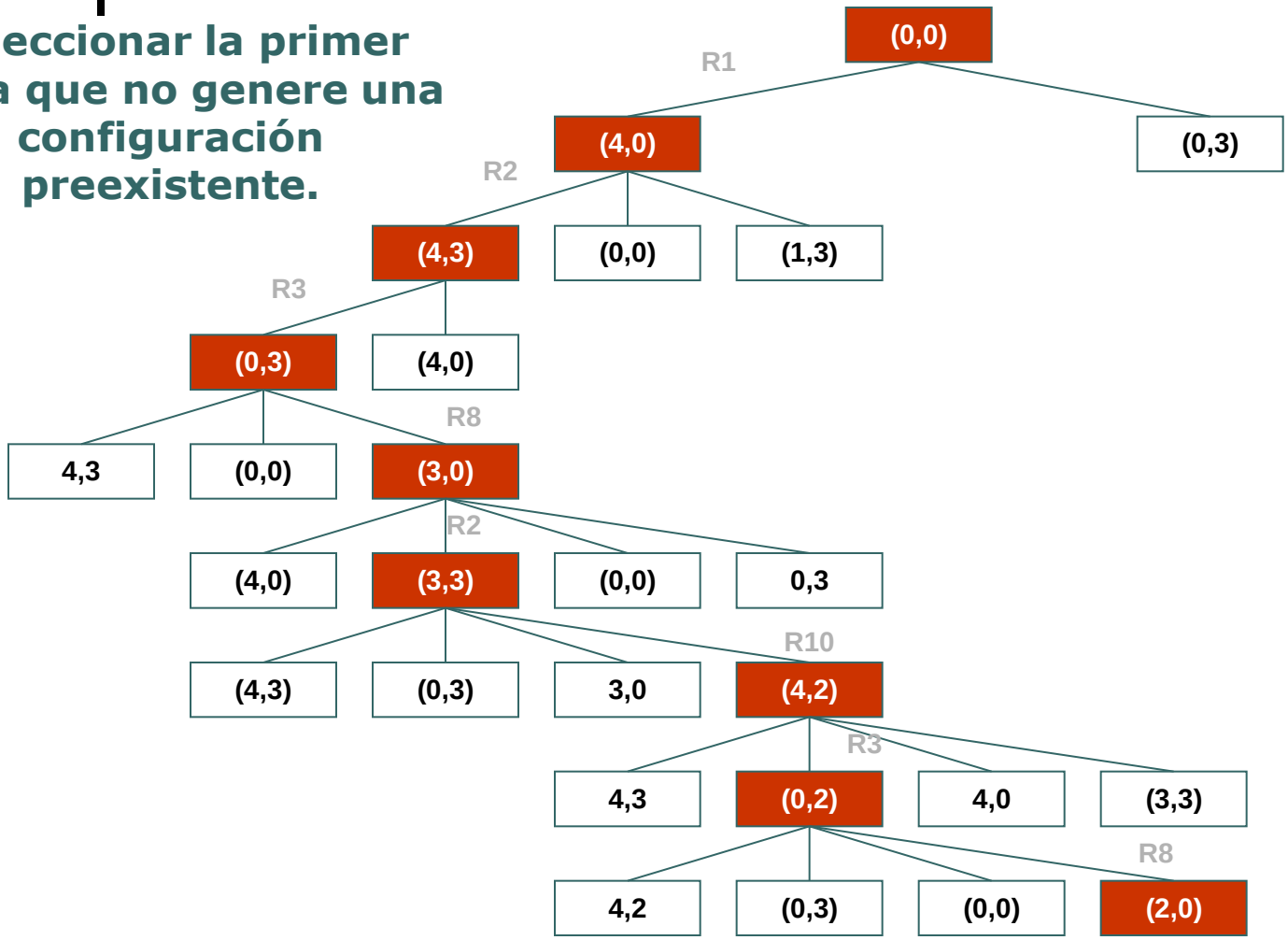
No importa el contenido de la segunda jarra.

Estado inicial
[0, 0]

1. [llenar la jarra 1]
2. [llenar la jarra 2]
3. [vaciar la jarra 1]
4. [vaciar la jarra 2]
5. [tirar cierta cantidad de agua de la jarra 1]
6. [tirar cierta cantidad de agua de la jarra 2]
7. [volcar el contenido de 1 en 2]
8. [volcar el contenido de 2 en 1]
9. [volcar el contenido de 1 en 2 hasta que se llene 2]
10. [volcar el contenido de 2 en 1 hasta que se llene 1]

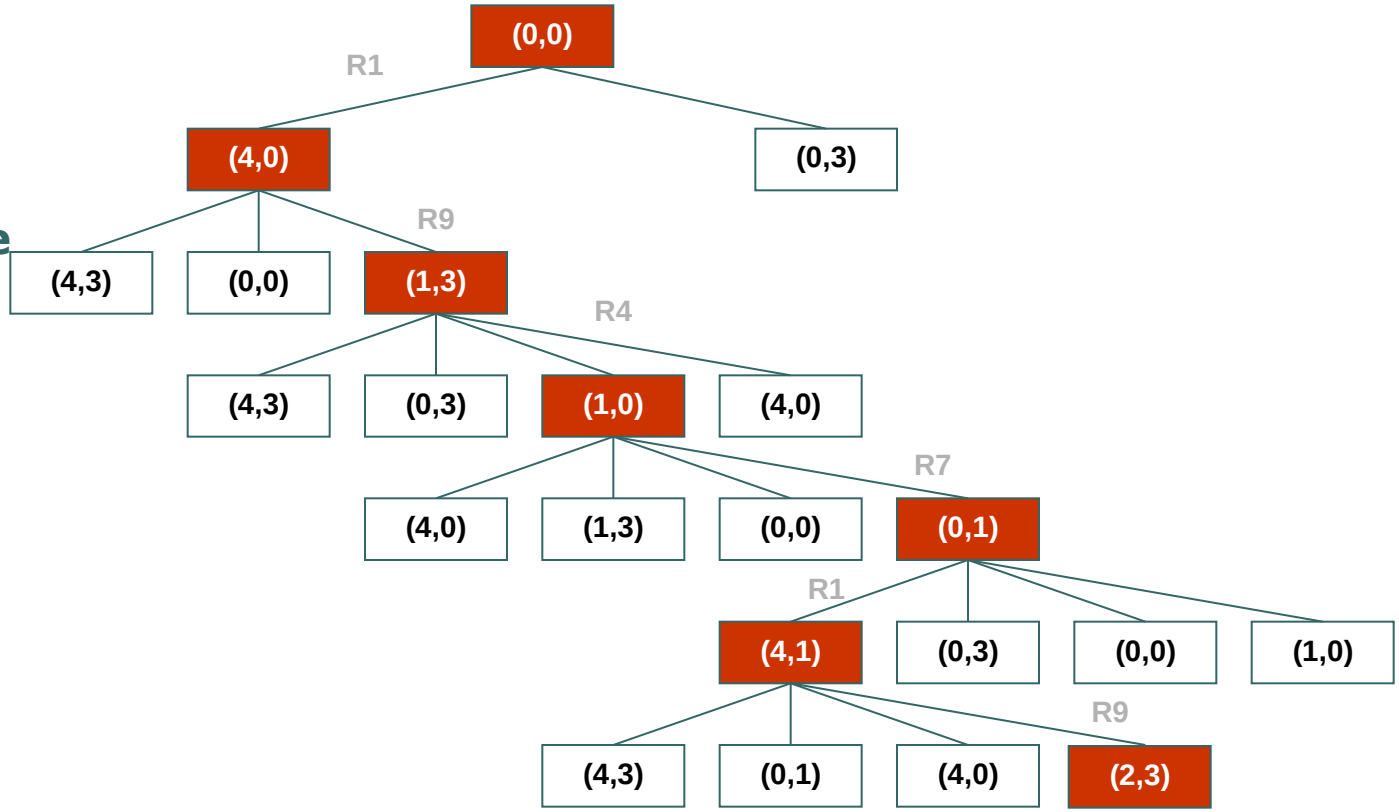
Árbol de Búsqueda

Seleccionar la primer regla que no genere una configuración preexistente.



Árbol de Búsqueda 2

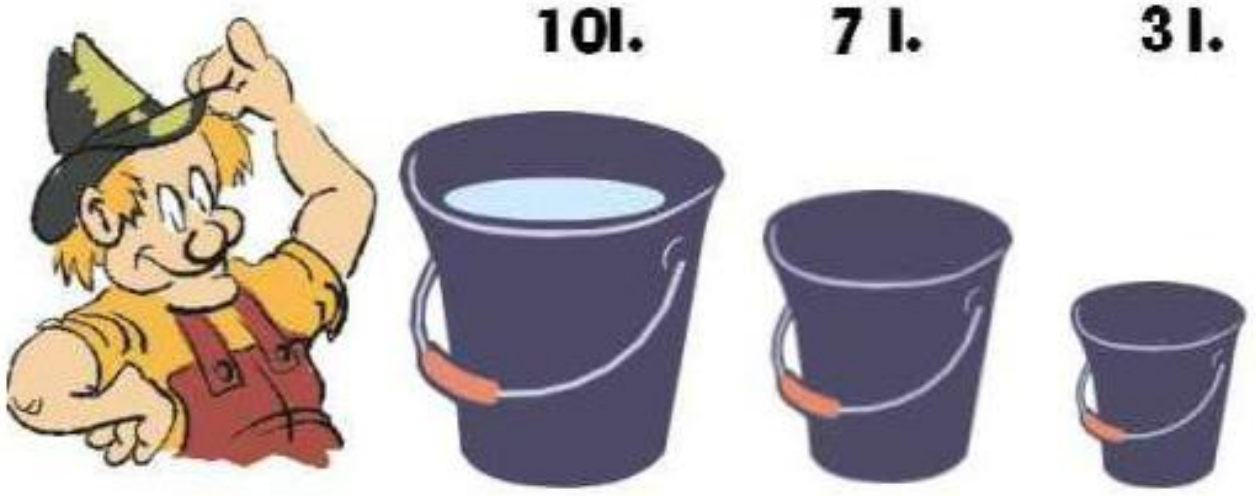
Seleccionando la regla en que la diferencia del contenido de la primera jarra con respecto a 2 fuese mínima.



Búsqueda

Encontrar la solución al siguiente caso:

Este granjero está en un serio problema. Tiene tres vasijas, una llena con 10 l. de leche , y dos vacías de 7 y 3 l. cada una. Debe dejar sólo 5 l. de leche en la primer vasija Puedes explicarle como debe hacer, pero sólo midiendo con la ayuda de las otras dos?



Búsqueda en Grafos

- El proceso de búsqueda establece un isomorfismo entre encontrar la secuencia de operadores que solucione el problema y encontrar un camino a través de un grafo dirigido.

- Cada nodo del grafo representa un estado I_k del problema. Existirá un arco entre el nodo i y j si y solo si existe un operador O que transforme a I_i en I_j.

Algoritmo vs. Búsqueda

- Un algoritmo halla la solución del problema en forma directa sin examinar distintas alternativas.
- Describe una descripción exacta sobre la realización, en una secuencia determinada, de acciones conducentes en un número finito de pasos a la solución de una clase específica de problemas.
- Un proceso de búsqueda consiste en ensayar exhaustiva y sistemáticamente todas las operaciones permitidas. Aplicando este procedimiento, es seguro que se hallará la solución, si al menos existe alguna.
- Si las combinaciones son infinitas y el problema es insoluble, el computador no se detendrá nunca.

Algoritmo de Control

Es un proceso que:

1. Extrae las reglas aplicables (aquellas que satisfacen la precondición)
2. Selecciona la regla a aplicar del conjunto disponible
3. Aplica la regla seleccionada.

Requisitos:

- Causar movimiento (evitar que se llegue al mismo estado para no obtener ciclos)
- Ser sistemática (no al azar – seguir un comportamiento predeterminado – primero en profundidad o primero en amplitud)

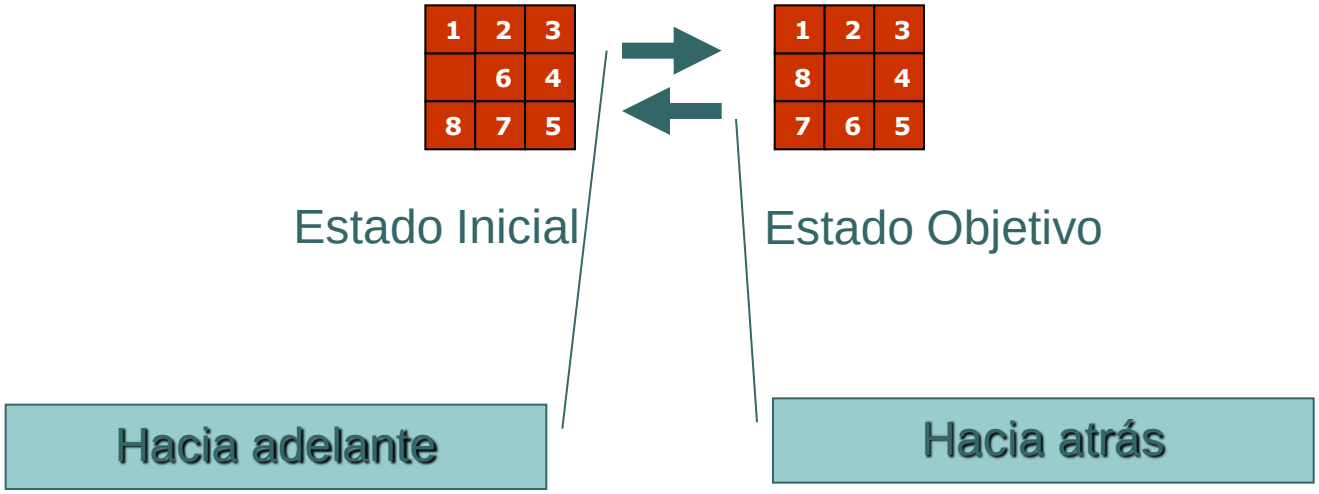
Dirección de la Búsqueda

- El objetivo del procedimiento de búsqueda es encontrar un camino entre la configuración inicial y la final.

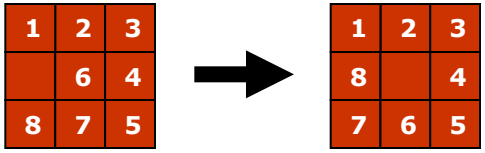
  Hacia delante (Forward): consisten en aplicar operadores al estado inicial, luego a sus sucesores y así sucesivamente hasta alcanzar el estado final.

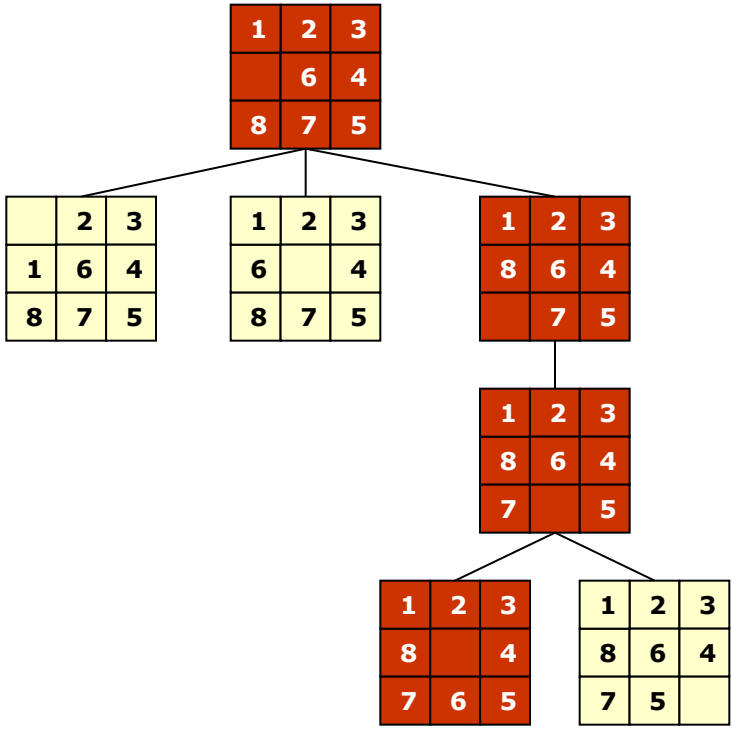
  Hacia atrás (Backward): consiste en aplicar los operadores al estado objetivo, que es convertido en uno o mas subobjetivos tal que sus soluciones son suficientes para resolver el objetivo original. Estos subobjetivos son reducidos a su vez a subobjetivos hasta que cada uno de ellos tenga una solución trivial.

Dirección de la Búsqueda 2

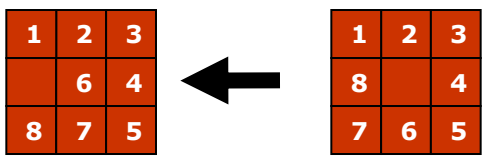


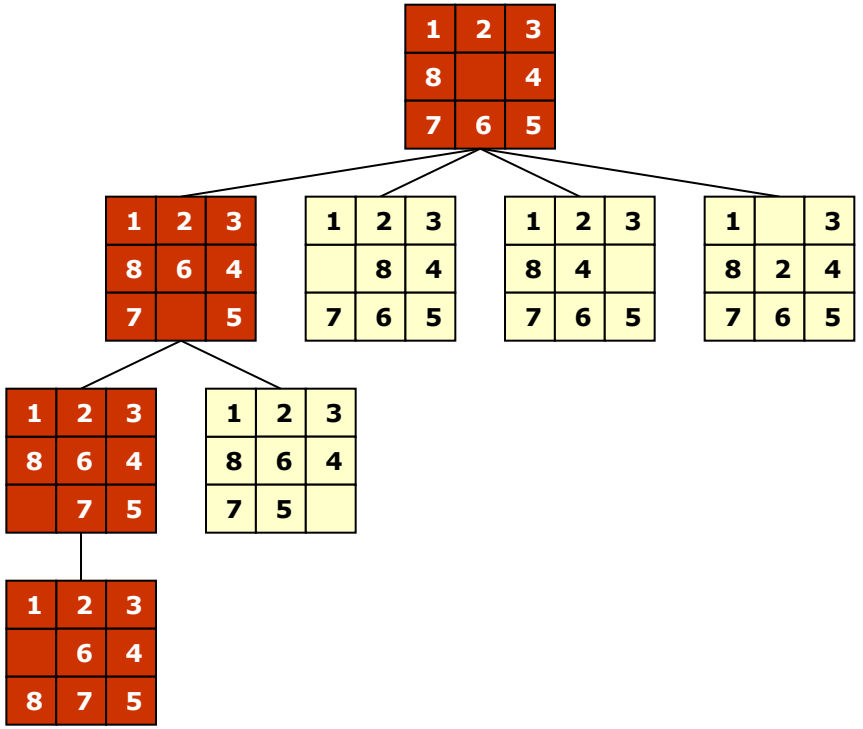
Encadenamiento hacia adelante





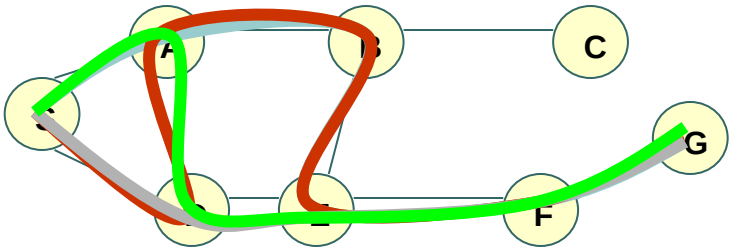
Encadenamiento hacia atrás





Métodos de Búsqueda

Consideremos el caso donde haya que encontrar un camino desde la ciudad S a la G.



Caminos posibles (que surgen de explorar todo el árbol de búsqueda)

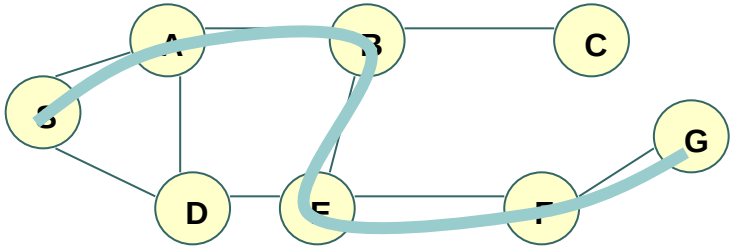
S-A-B-E-F-G

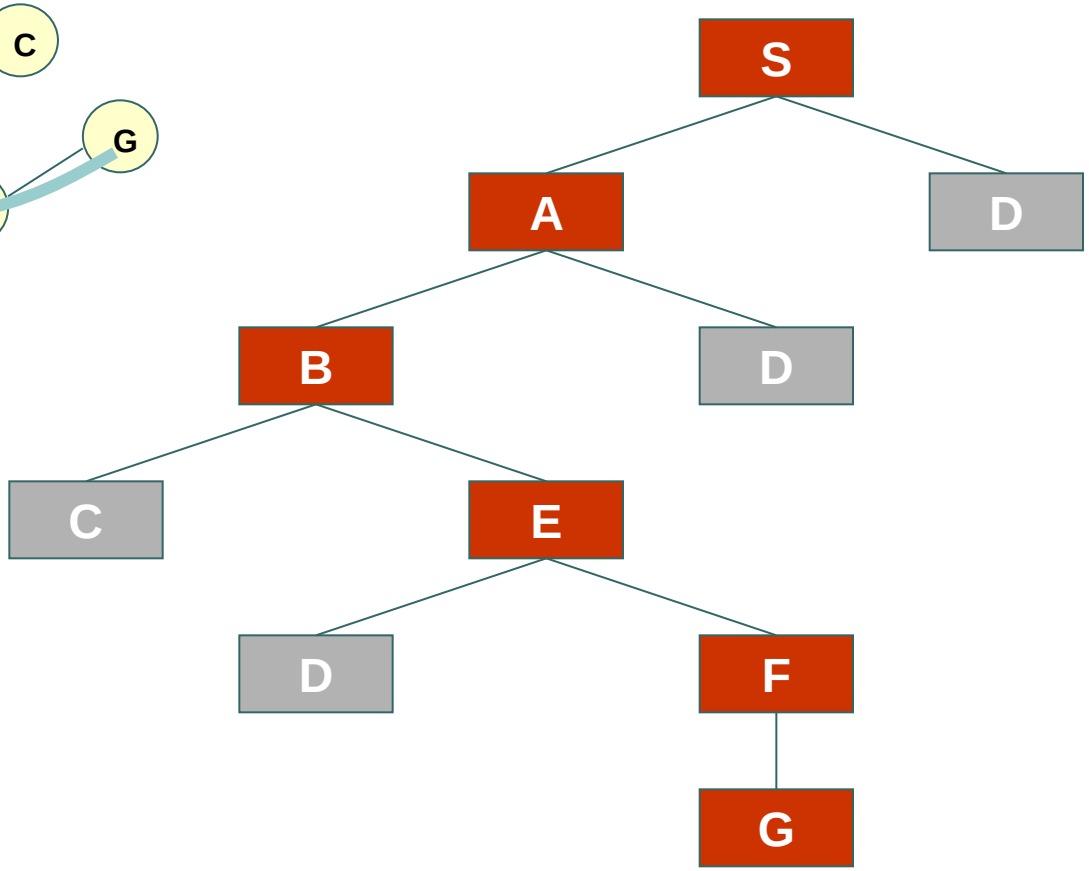
S-D-A-B-E-F-G

S-D-E-F-G

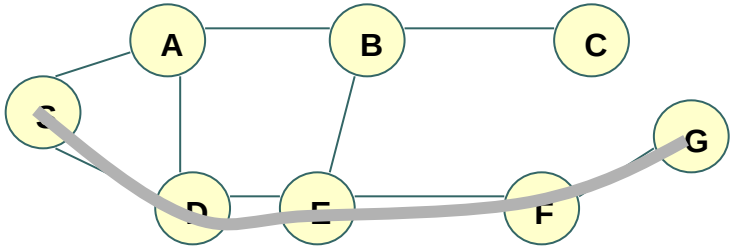
S-A-D-E-F-G

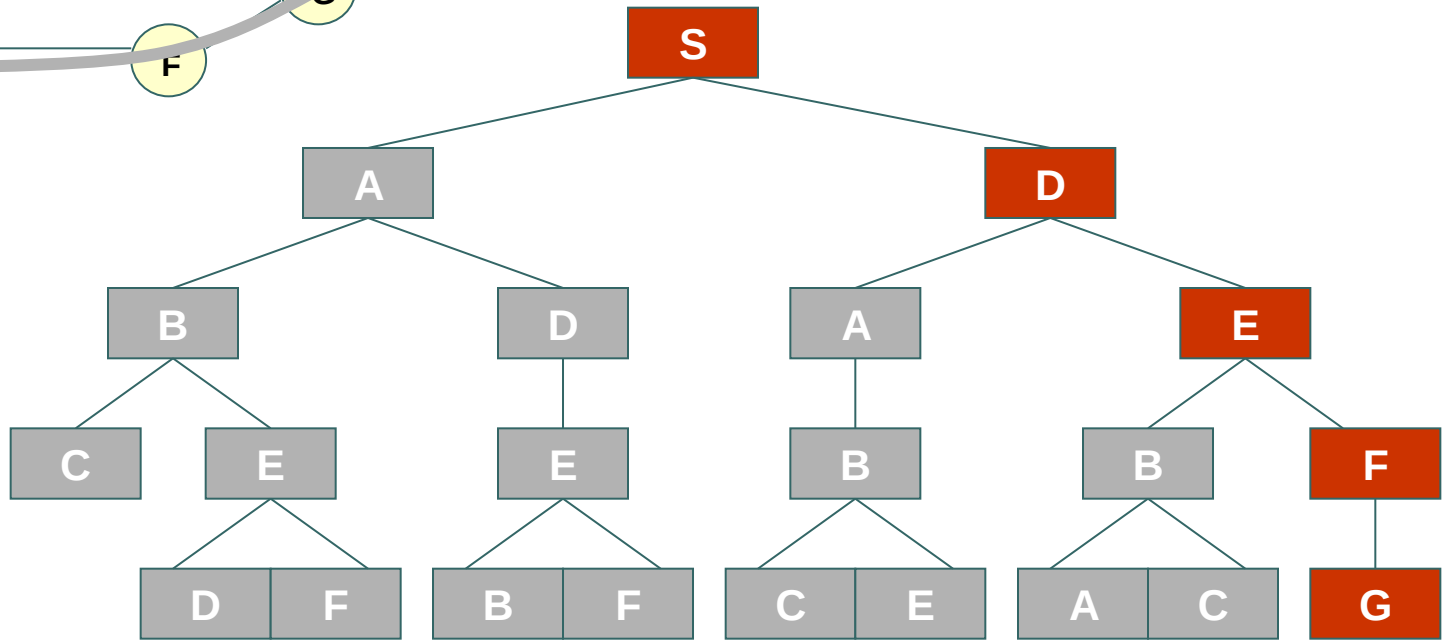
Primero en Profundidad



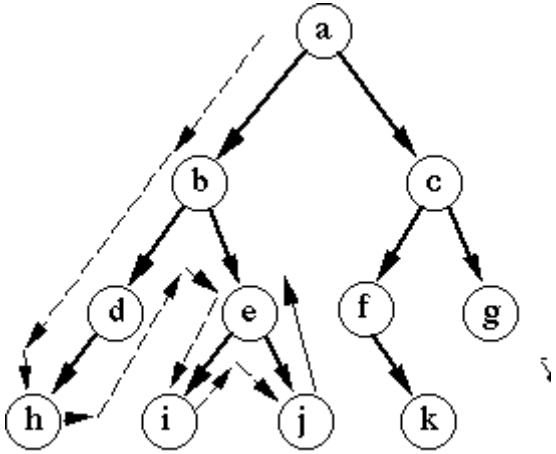


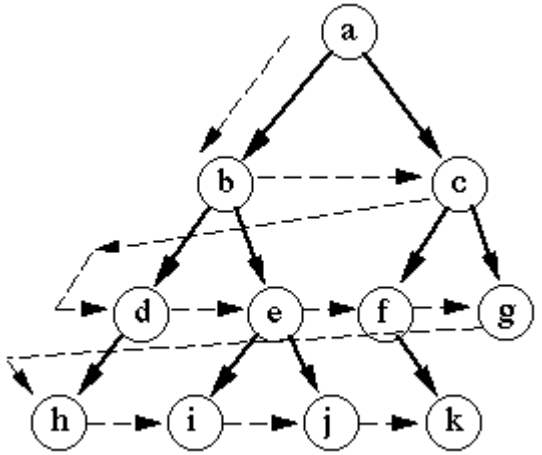
Primero en Amplitud





Primero en Profundidad vs. Primero en Amplitud





- **Necesita menos memoria**
- **Puede encontrar una solución sin tener que explorar gran parte del espacio de estados.**
- **Puede encontrar una solución no mínima.**

- **Funciona con árboles de profundidad infinita.**
- **Alcanzará la solución mínima. (mínimo número de pasos).**
- **No queda atrapada en caminos que no conducen a estados objetivos.**
- **Utiliza demasiada memoria para guardar los nodos.**