La Estrategia de Árboles de Búsqueda

Cristian López Del Alamo [1], Jheisson Rojas [2]
Carlos Arias [1]
Israel Medina Velazco [1]
Eddy René Cáceres Huacarpuma [3]
José Miguel Huamán Cruz [2]

Universidad Nacional de San Agustín
Escuela Profesional de Ciencia de la Computación

Julio 2012

[1] Universidad La Salle
[2] Universidad Católica San Pablo

Introducción

- Se considera el problema de satisfactibilidad.
- Para determinar si este conjunto de clausulas es factible se analiza todas las posibles combinaciones
- Si hay *n* variables x_1,x_2, $\ldots x_n$. Se analizan todas las 2^n combinaciones posibles.

x_1	x_2	x_3
F	F	F
F	F	V
F	V	F
F	V	V
V	F	F
V	V	F
V	V	V

Representación de árbol de ocho combinaciones

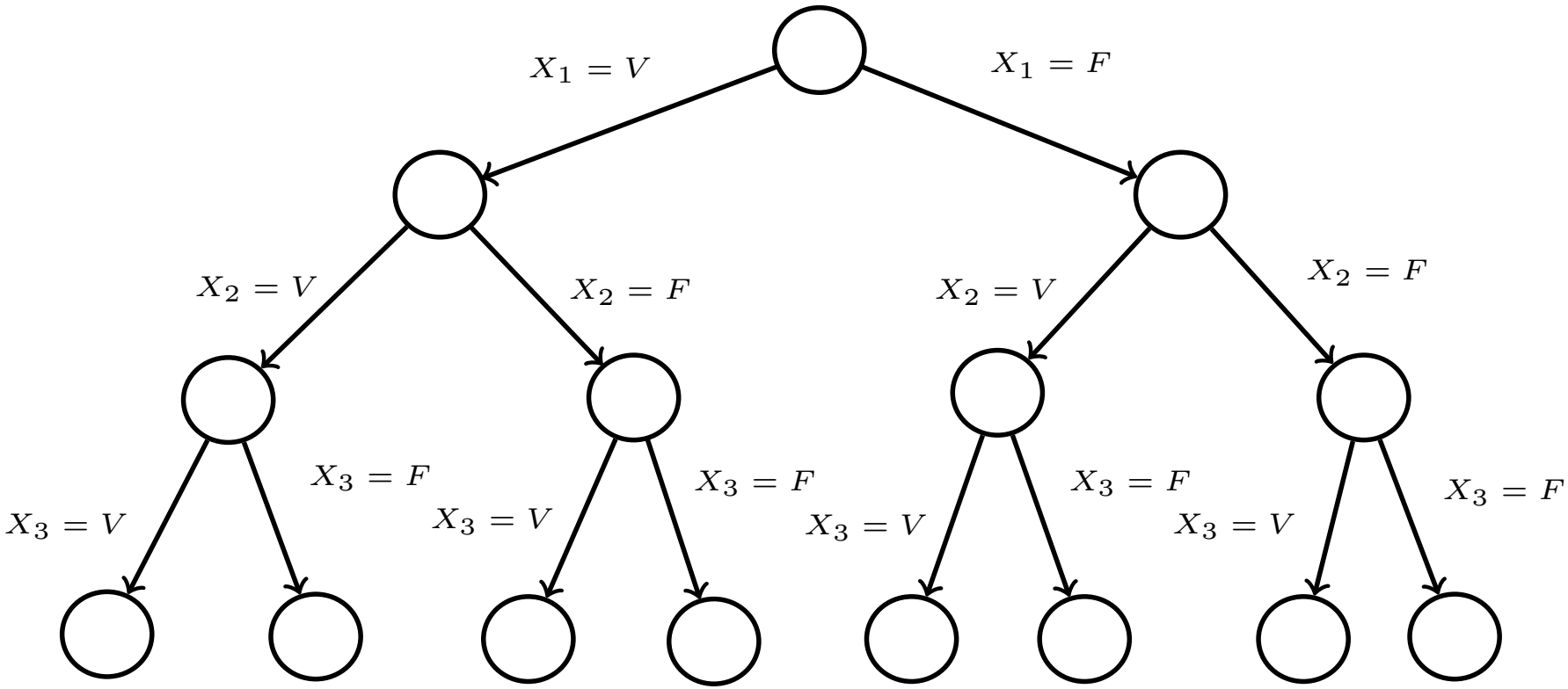


Figura: Representación de árbol de ocho representaciones

Representación de árbol de ocho combinaciones

Es posible determinar la satisfactibilidad sin necesidad de analizar todas las combinaciones: sólo todas las clases de combinaciones

$-x_1$ (1)

x_1 (2)

$x_2 v x_5$ (3)

$-x_2$ (4)

Problema del rompecabezas

- Posición inicial del problema.

2	3	
5	1	4
6	8	7

Problema del rompecabezas

- Posición final del problema.

1	2	3
8		4
7	6	5

Problema del ciclo Hamiltoniano

- Existe un ciclo Hamiltoniano

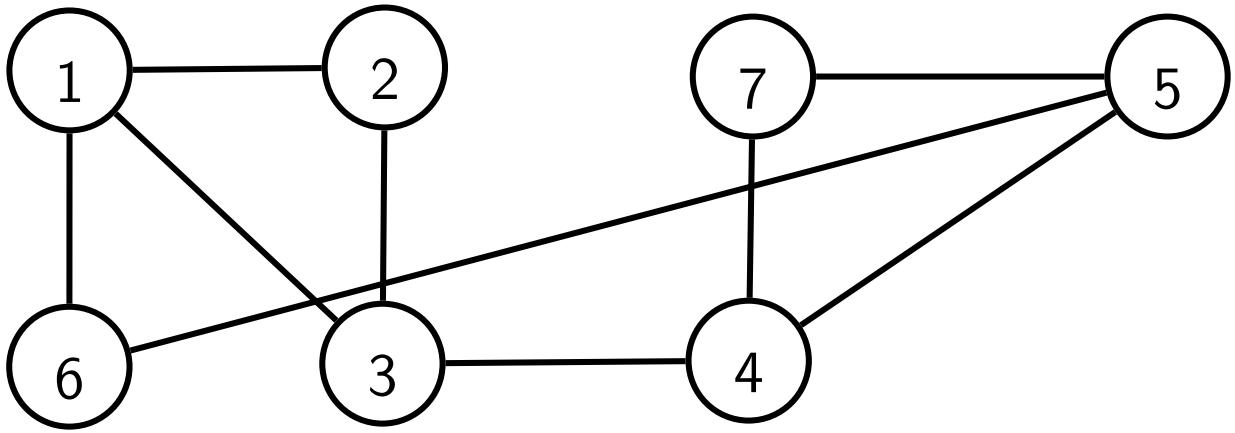


Figura: Gráfica que contiene un ciclo Hamiltoniano

Problema del ciclo Hamiltoniano

- No existe un ciclo Hamiltoniano

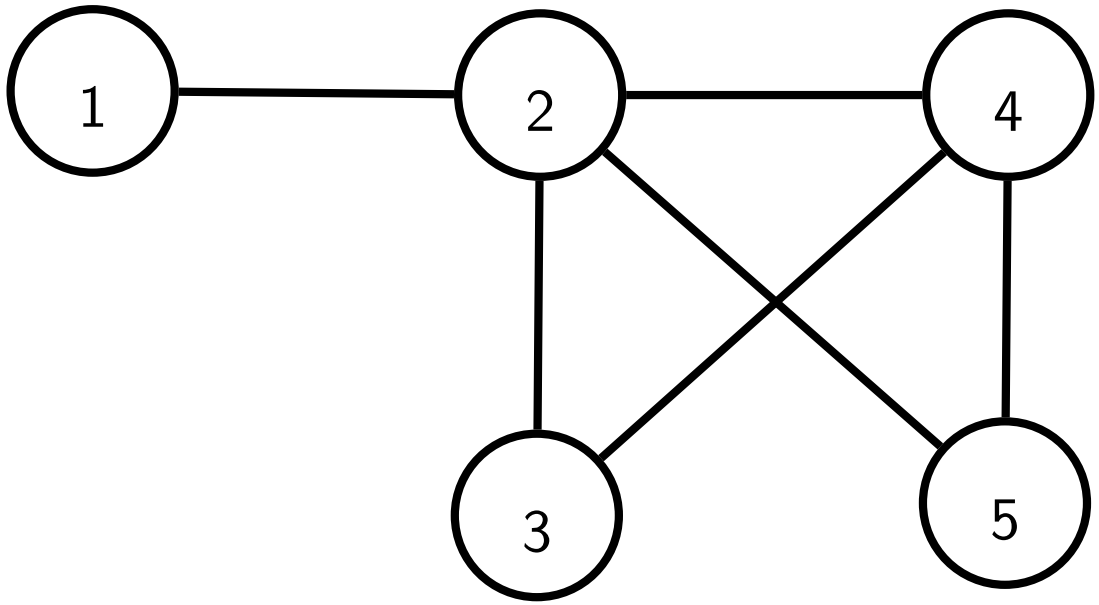


Figura: Gráfica que no contiene un ciclo Hamiltoniano

Problema del ciclo Hamiltoniano

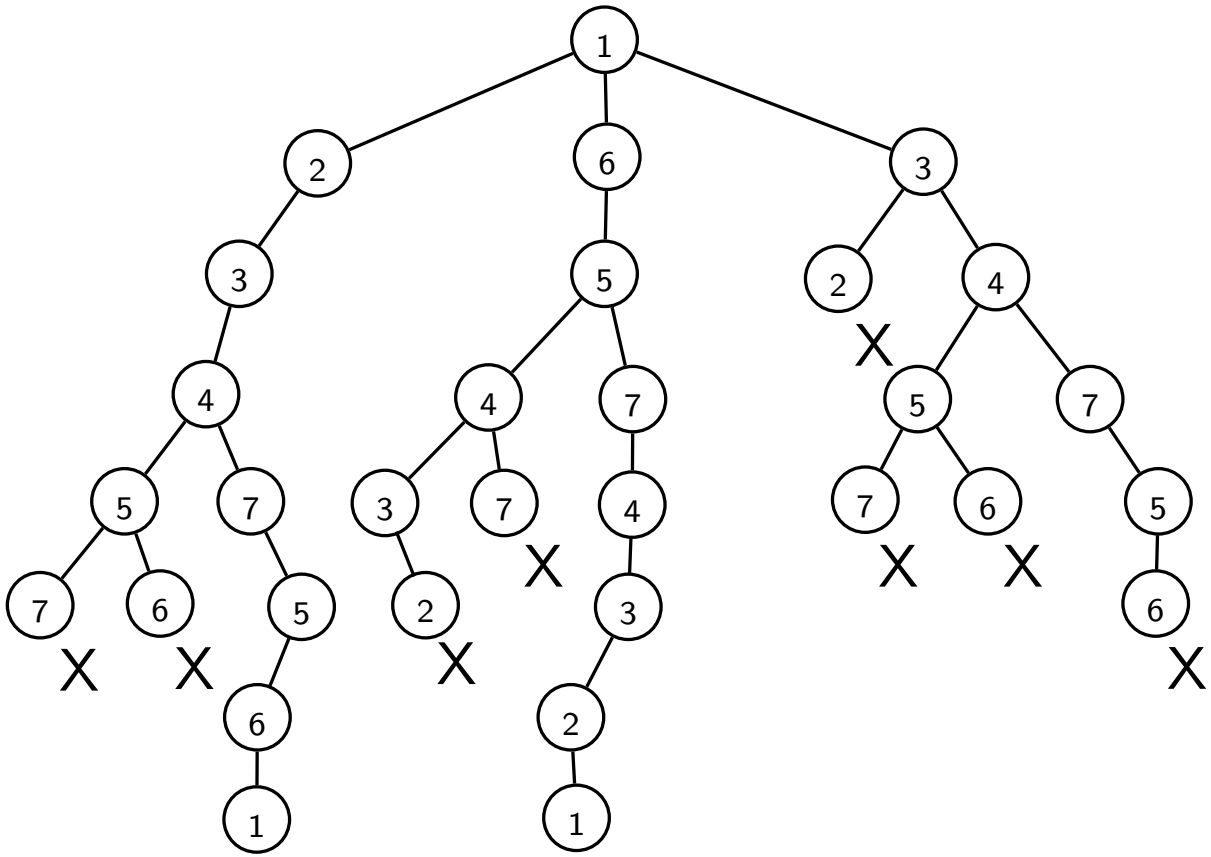


Figura: Representación de árbol sobre la existencia o no de un ciclo Hamiltoniano

Búsqueda de primero en amplitud (Breadth - First Search)

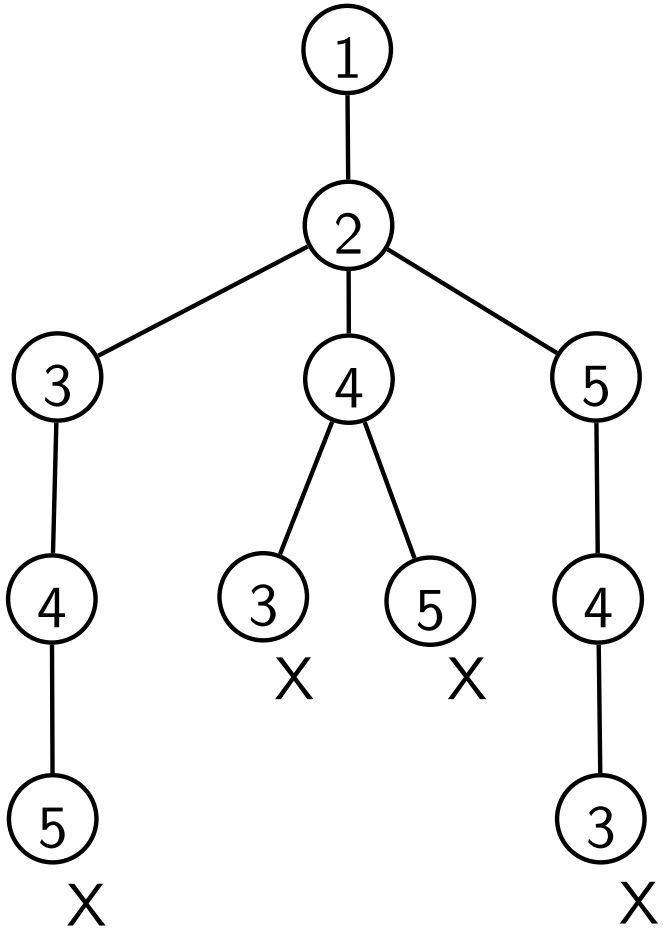


Figura: Árbol que muestra la inexistencia de Ciclo Hamiltoniano

Búsqueda de primero en amplitud (Breadth - First Search)

- Solución del problema de rompecabezas

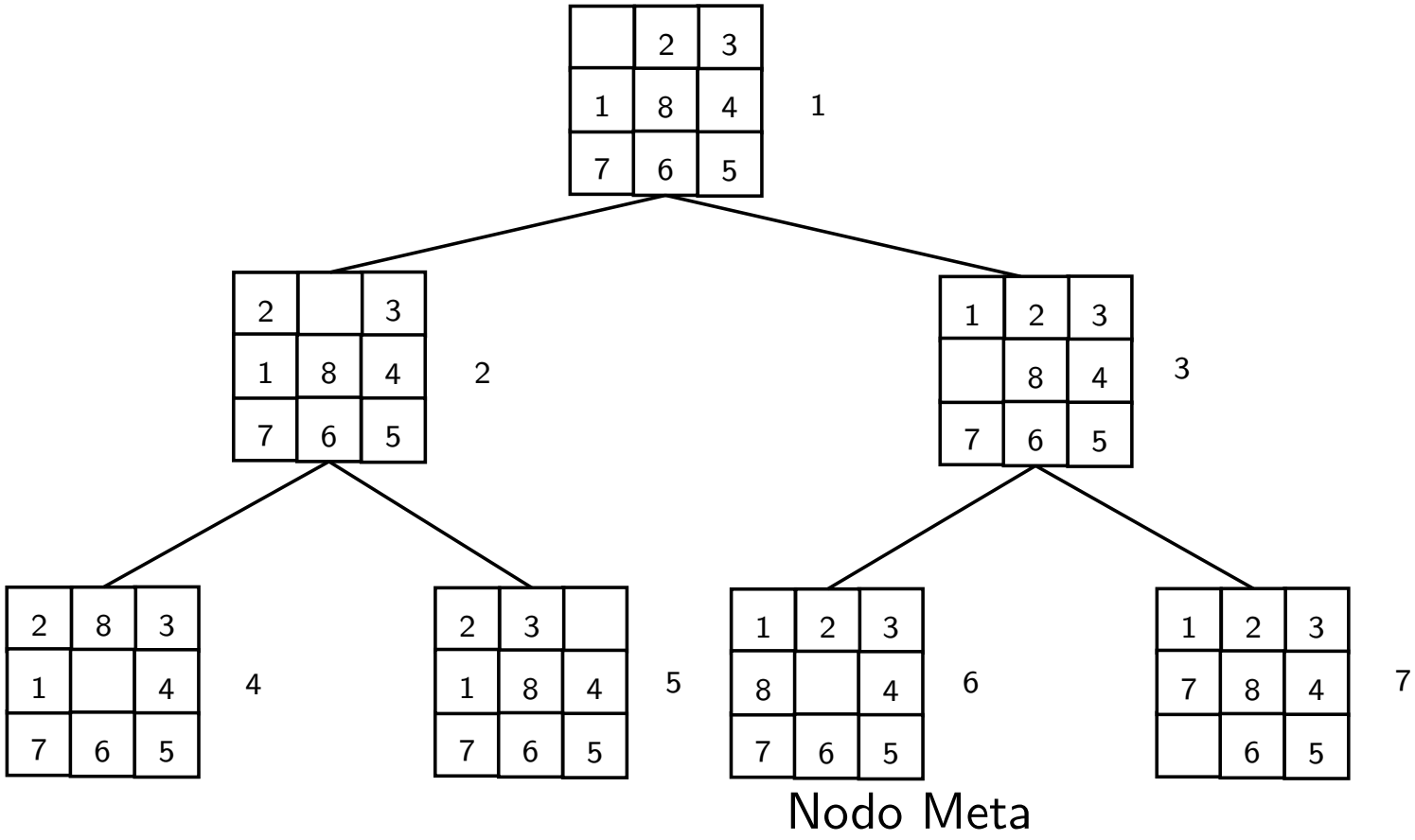


Figura: Árbol de búsqueda producido por una búsqueda de primero en amplitud

Búsqueda de primero en amplitud (Breadth - First Search)

Algoritmo:

- **Paso 1.** Formar una cola de un elemento que consta del nodo raiz.
- **Paso 2.** Probar si el primer elemento en la cola es un nodo meta. En caso afirmativo detenerse, en caso contrario, ir al paso 3.
- **Paso 3.** Quitar el primer elemento de la cola. Agregar los descendientes (nodos hijos) del primer elemento, en caso de haber alguno, al final de la cola.
- **Paso 4.** Si la cola está vacia, regresar falla. En caso contrario, ir a paso 2.

Búsqueda de primero en profundidad (Depth - First Search)

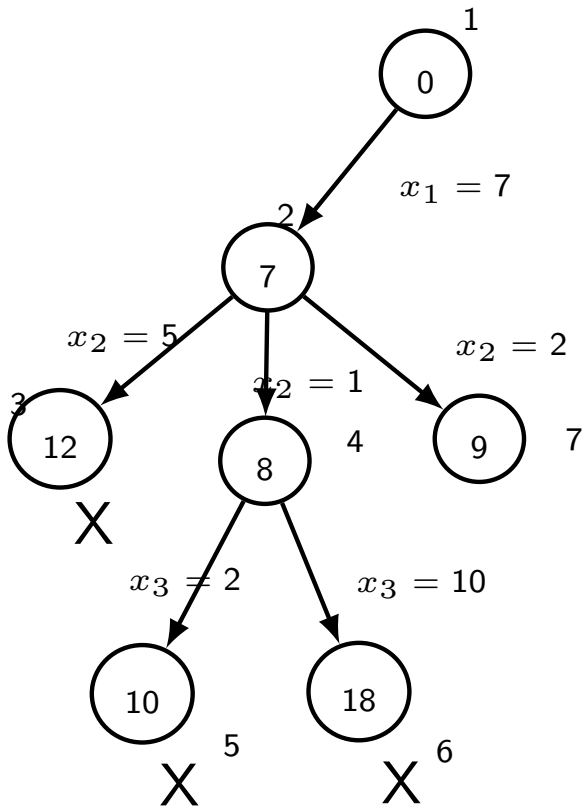


Figura: Gráfica que contiene un ciclo Hamiltoniano

Búsqueda de primero en profundidad (Depth - First Search)

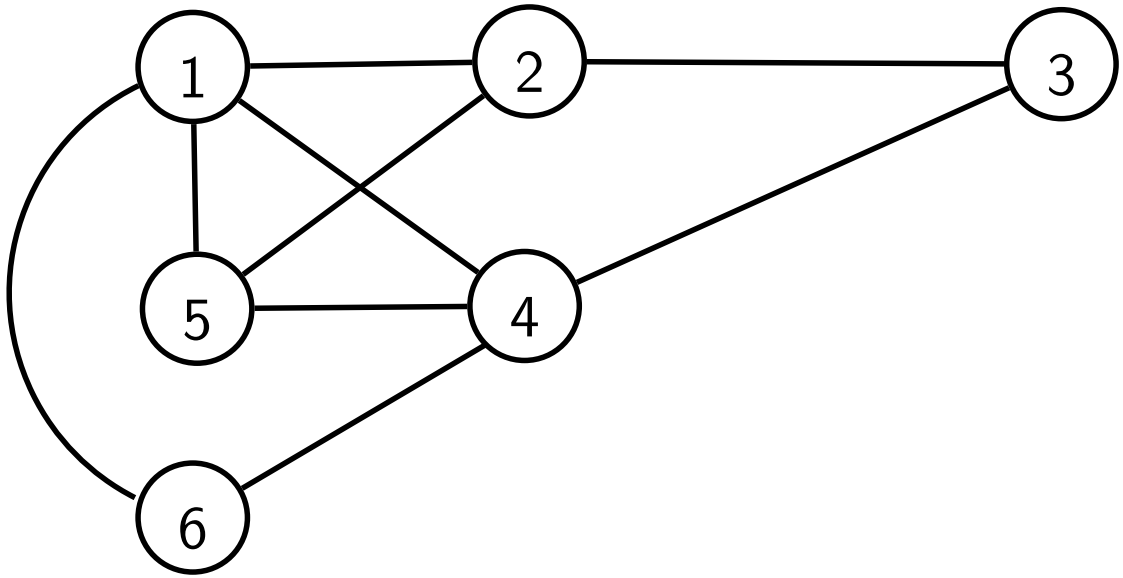


Figura: Una suma del problema del subconjunto resuelta por búsqueda en profundidad

Búsqueda de primero en profundidad

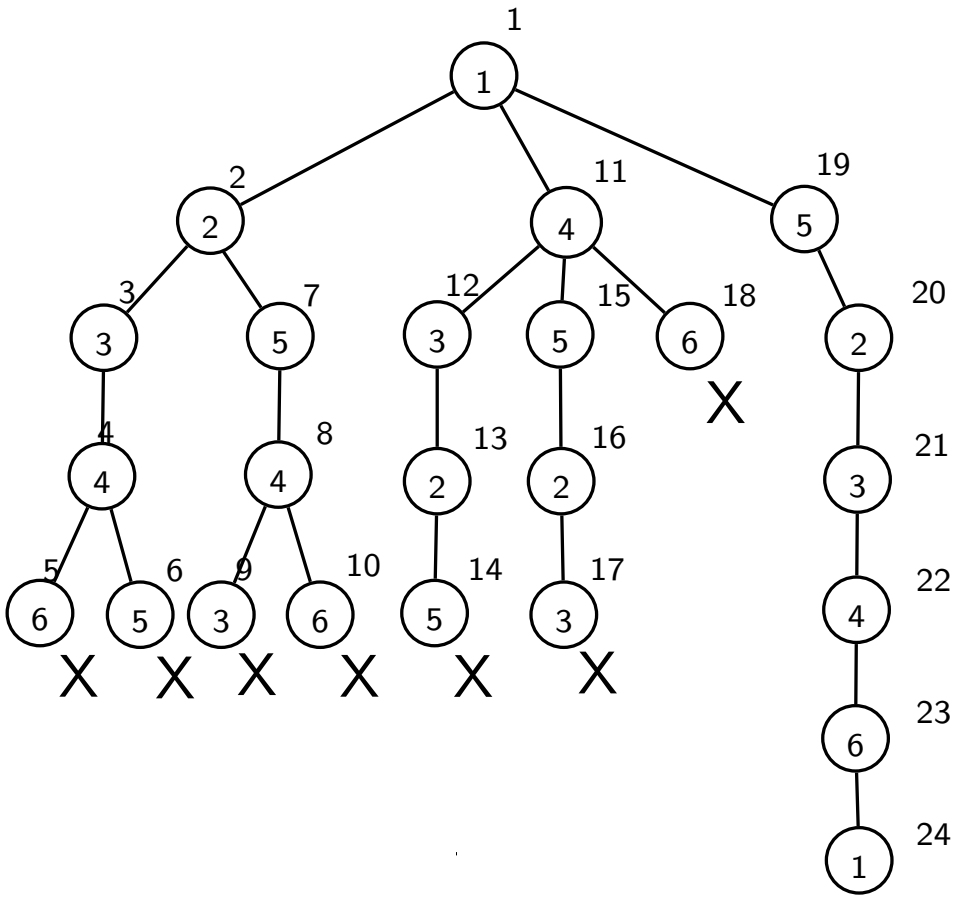


Figura: Ciclo Hamiltoniano producido por búsqueda de primero en profundidad

Búsqueda de primero en profundidad (Depth-First Search)

Algoritmo:

- **Paso 1.** Formar una pila de un elemento que conste del nodo raiz.
- **Paso 2.** Probar si el primer elemento superior en la pila es un nodo meta. En caso afirmativo detenerse, en caso contrario, ir al paso 3.
- **Paso 3.** Quitar este elemento superior de la pila y agregar sus descendientes, en caso de haber alguno a la parte superior de la pila.
- **Paso 4.** Si la pila está vacia, regresar falla. En caso contrario, ir a paso 2.

Método de Ascenso de colina (Hill Climbing)

- Es una variable de la búsqueda primero en profundidad.
- Se aplica un método codicioso para decidir la dirección en que hay que moverse.
- En otras palabras utiliza alguna medida heurística para ordenar las opciones.

$$f(n) = \omega(n)$$

donde ω(n) es el número de piezas móviles mal colocadas.

Ascenso de Colina (Solución del rompecabezas)

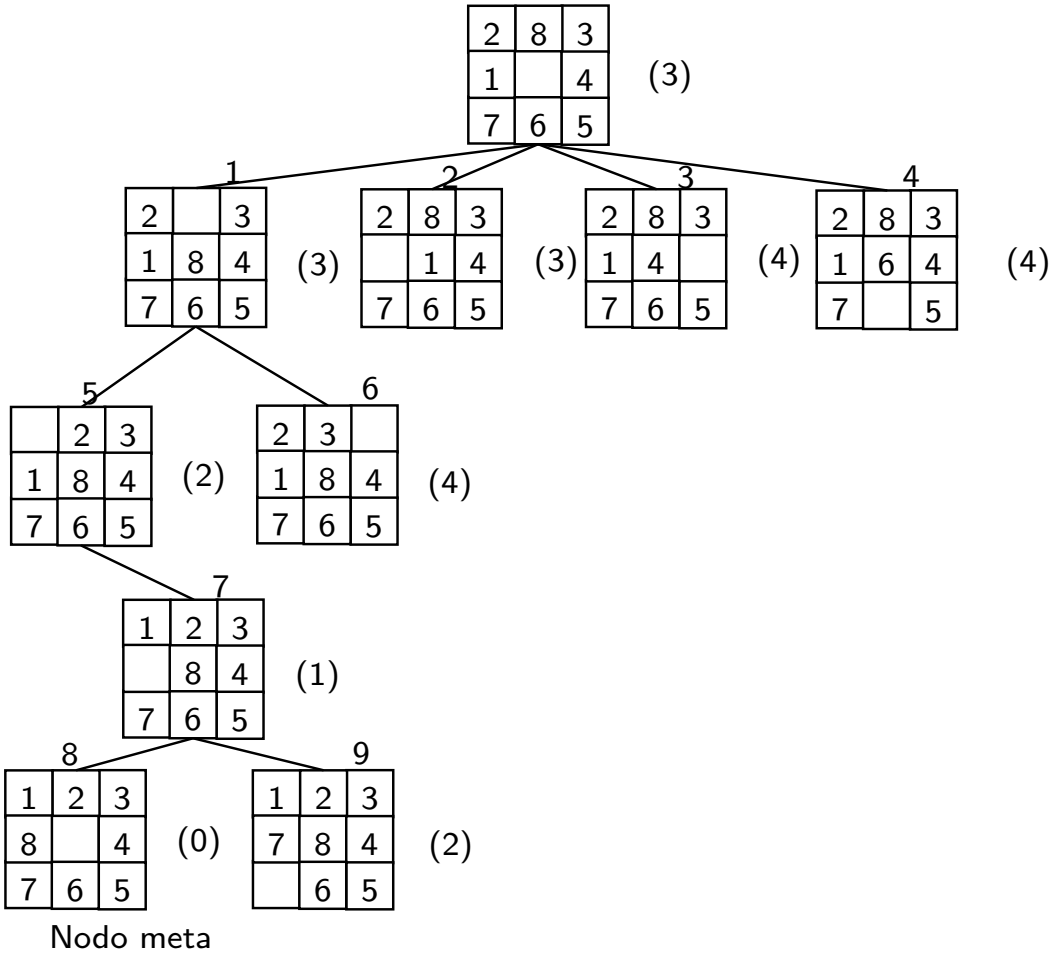


Figura: Gráfica que contiene un ciclo Hamiltoniano

Método Ascenso de Colina

Algoritmo:

- **Paso 1.** Formar una pila de un elemento que conste del nodo raiz.
- **Paso 2.** Probar si el primer elemento de la pila es un nodo meta. En caso afirmativo detenerse, en caso contrario, ir al paso 3.
- **Paso 3.** Quitar el primer elemento de la pila y expandir el elemento. Agregar los descendientes del elemento eliminado a la pila ordenanda por la función de evaluación.
- **Paso 4.** Si la pila está vacia, regresar falla. En caso contrario, ir a paso 2.

Método de Búsqueda de primero el mejor

- Posee una visión general
- A diferencia de Ascenso de colina inspecciona la que parece la mejor trayectoria hasta el final; la búsqueda primero el mejor analiza varias trayectorias a la vez, siempre siguiendo la mejor trayectoria parcial conocida al momento.
- Generalmente la búsqueda primero el mejor encuentra trayectorias más cortas a los estados meta.

Representación : $f'(n) = g(n) + h'(n)$
Donde : g es el Costo de llegar desde nodo inicial a nodo actual
h' es Costo de llegar desde nodo actual a nodo objetivo
Complejidad : dependiendo de la funcion heurística, sera exponencial o lineal
$h'(x) < g(y) - g(x) + h'(y)$

Método de Búsqueda de primero el mejor

Algoritmo:

- **Paso 1.** Construir un heap usando la función de evaluación, un elemento (nodo raiz) .
- **Paso 2.** Probar si el elemento raiz en el heap es un nodo meta. Si es asi detenerse, en caso contrario, ir al paso 3.
- **Paso 3.** Quitar el primer elemento raiz del heap y expandir el elemento. Agregar los descendientes del elemento eliminado al heap.
- **Paso 4.** Si el heap está vacio, regresar falla. En caso contrario, ir a paso 2.

Solución rompecabezas con Primero el mejor

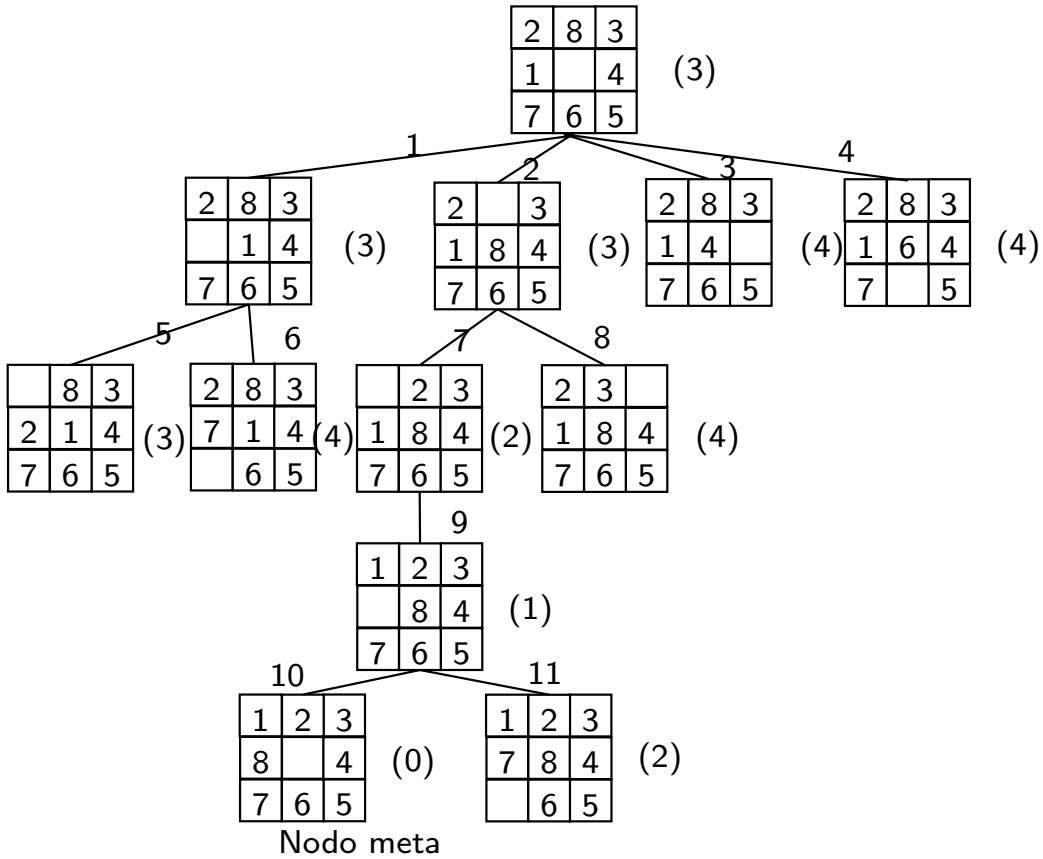
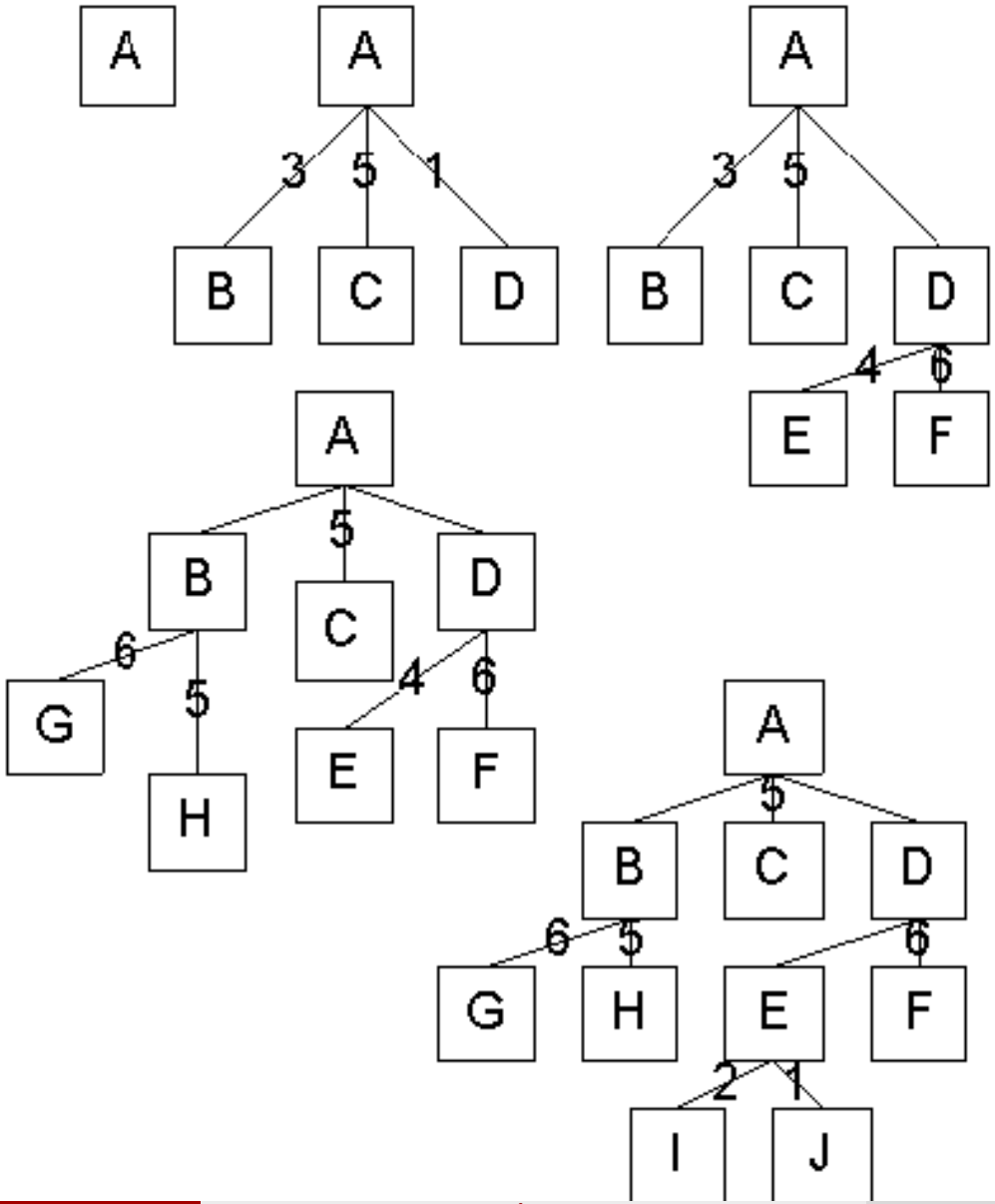


Figura: Problema del rompecabezas de 8 piezas resuelto por el método de búsqueda de primero el mejor

ejemplo



Estrategia de Ramificar y Acotar

Posee una visión general

Algoritmo:

- Utilizado para resolver problemas de optimización
- Acota el espacio solución
- Por ejemplo encontrar la ruta más corta del siguiente grafo

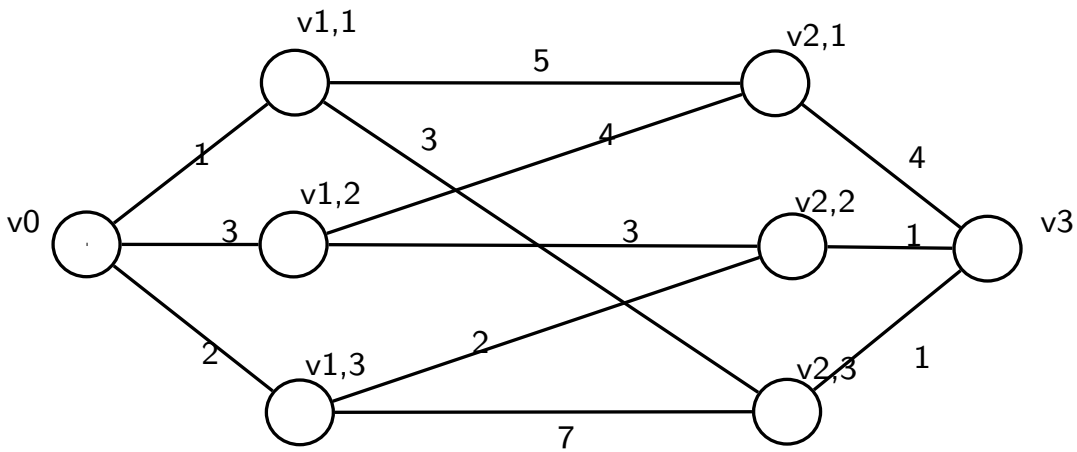


Figura: Problema multietapas de una gráfica de búsqueda

Estrategia de ramificar y acotar

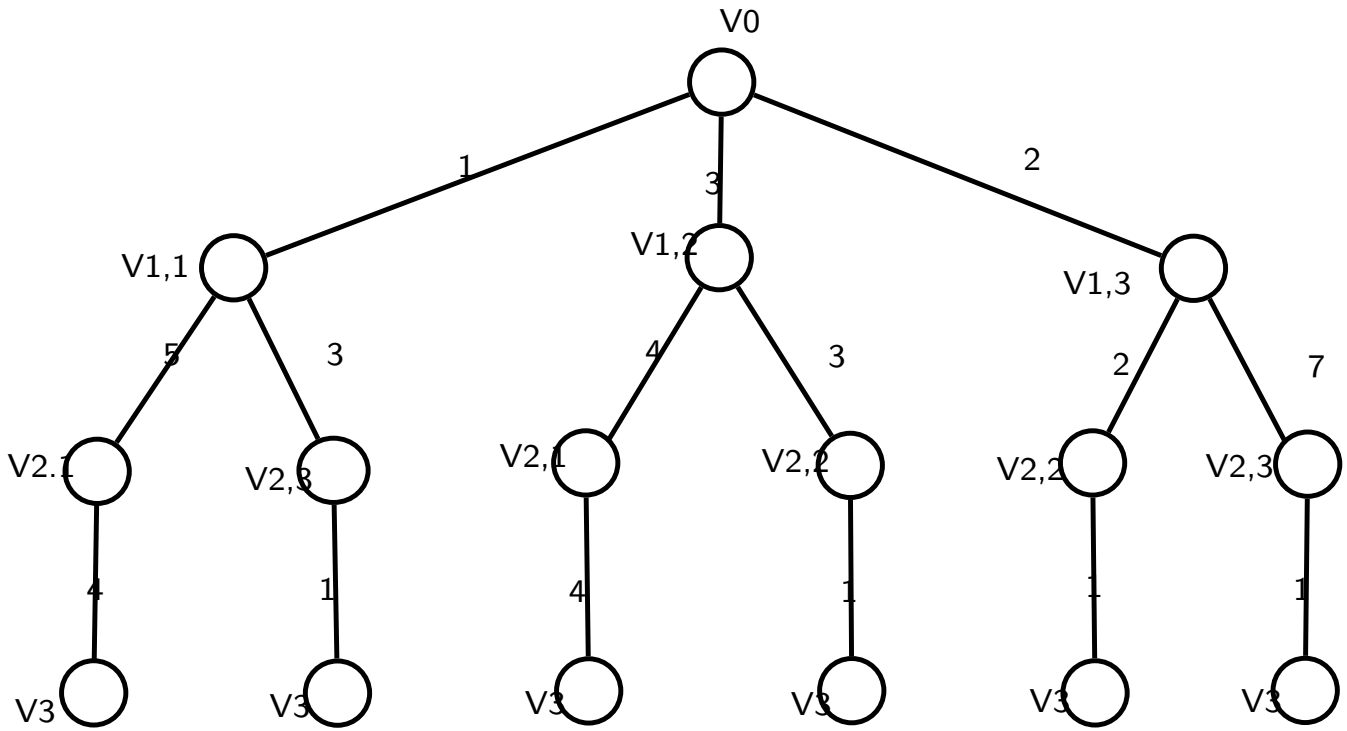


Figura: Representación de árbol de soluciones del problema de la figura anterior

Estrategia de ramificar y acotar

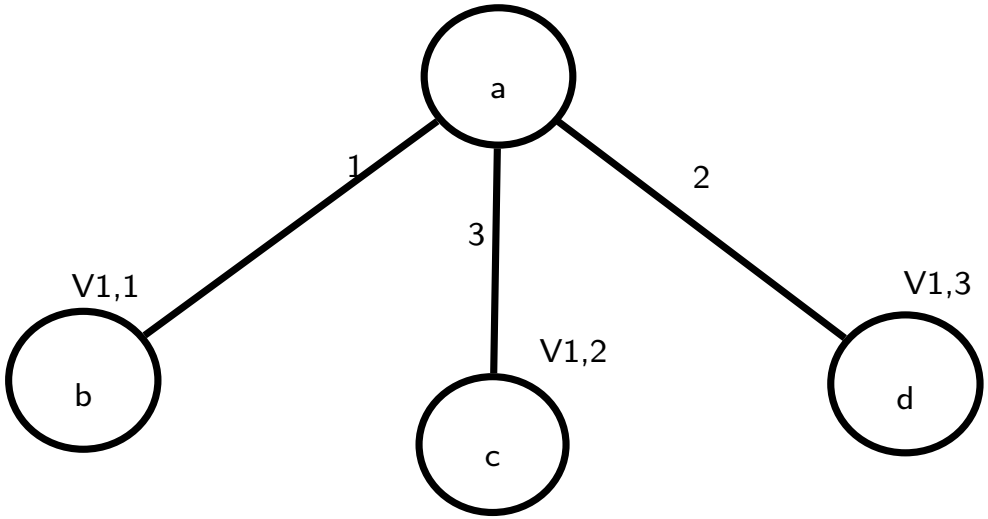


Figura: Representación de árbol de soluciones del problema de la figura anterior

Estrategia de ramificar y acotar

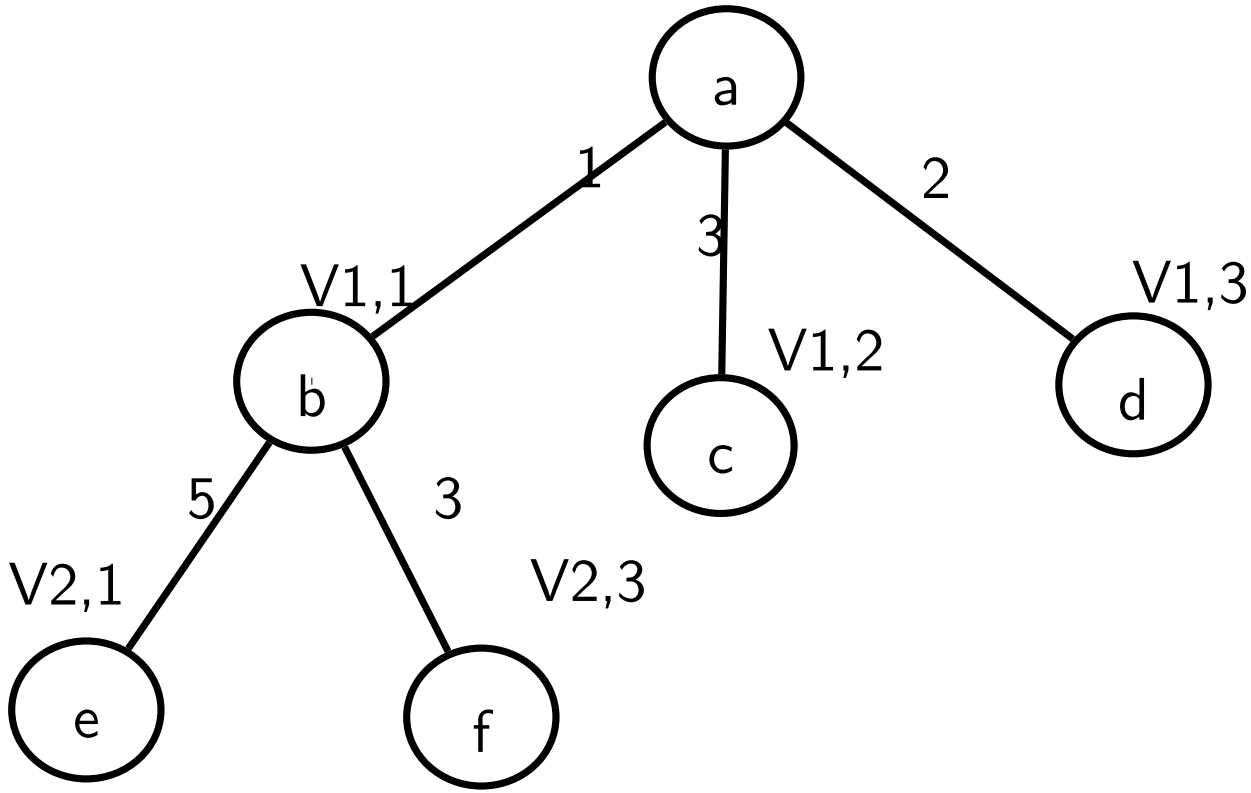


Figura: Representación de árbol de soluciones del problema de la figura anterior

Estrategia de ramificar y acotar

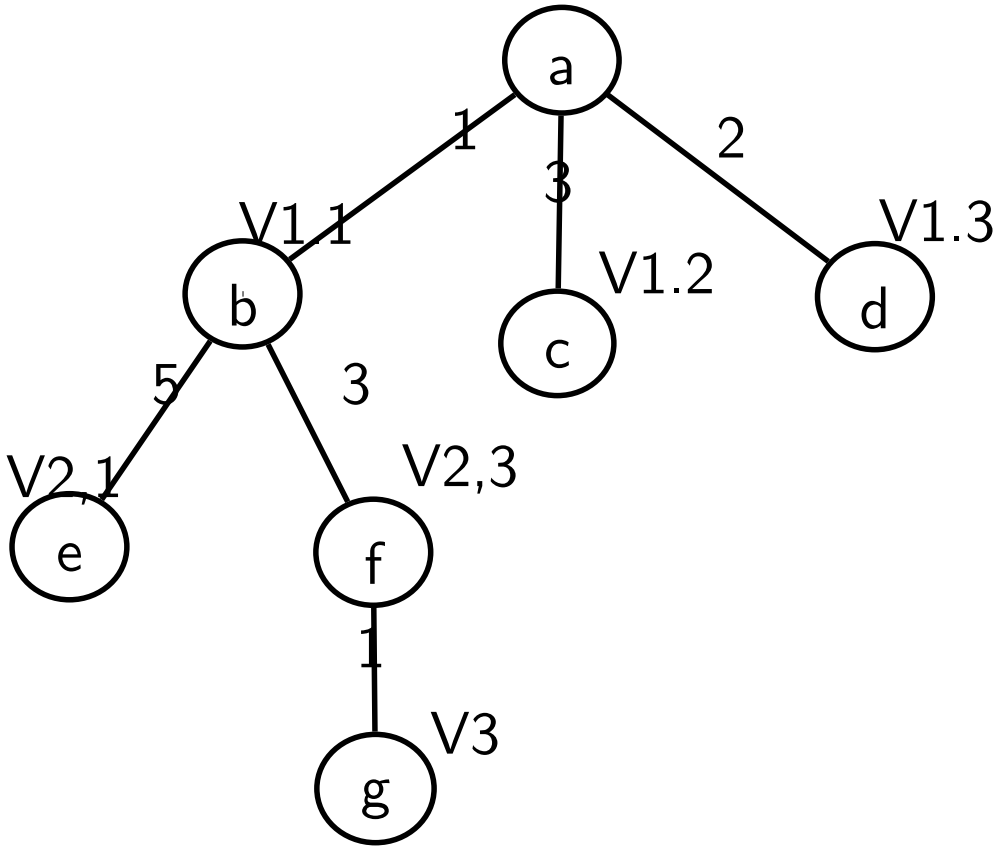


Figura: Representación de árbol de soluciones del problema de la figura anterior

Estrategia de ramificar y acotar

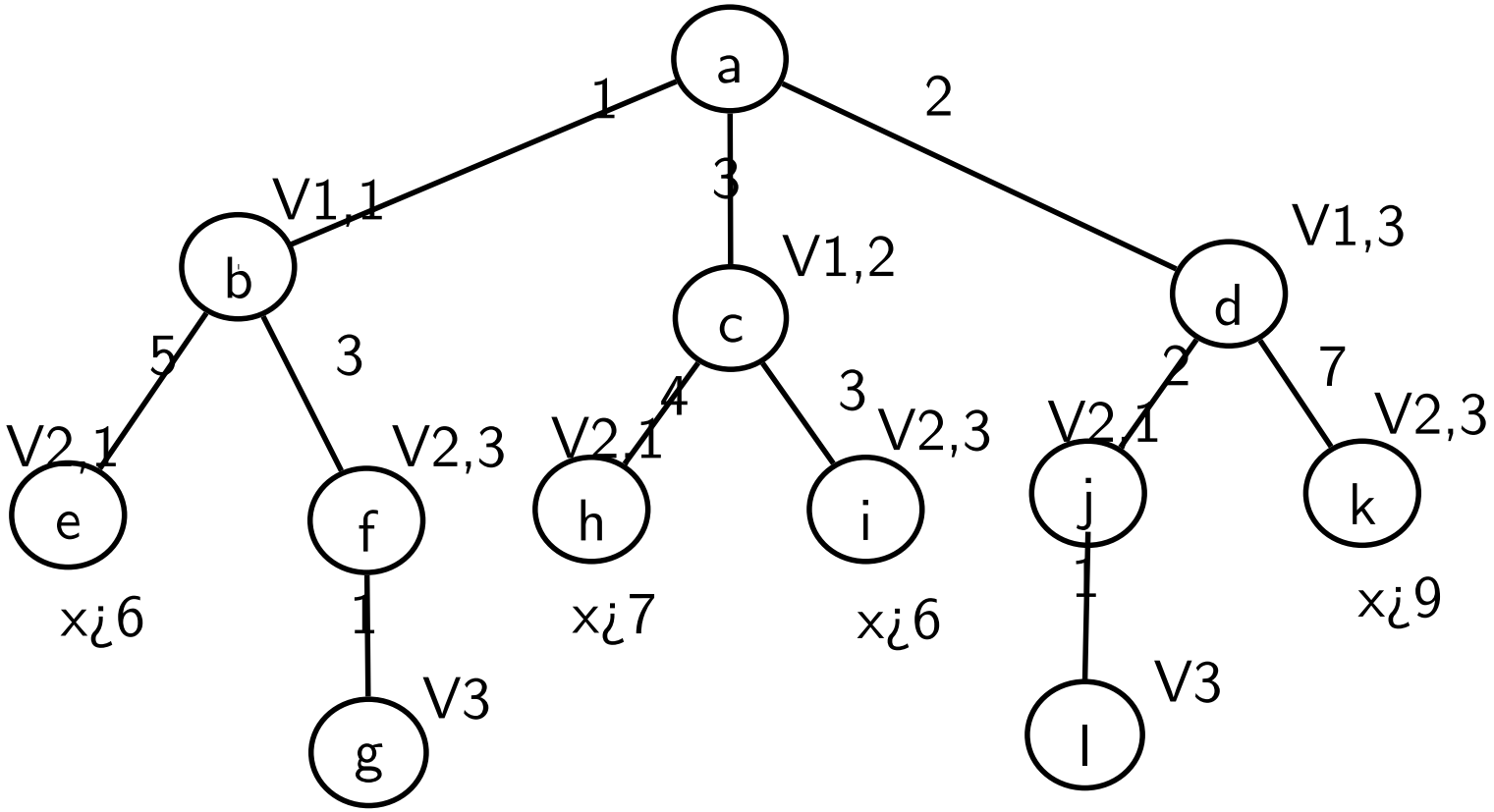


Figura: Representación de árbol de soluciones del problema de la figura anterior

Estrategia de ramificar y acotar

- Utilizando ascenso de colina
- Principio de menor costo
- El costo de la primera solución sirve como una cota superior
- Es eficiente en casos promedio

Problema de Asignación de personal (Ramificar y Acotar)

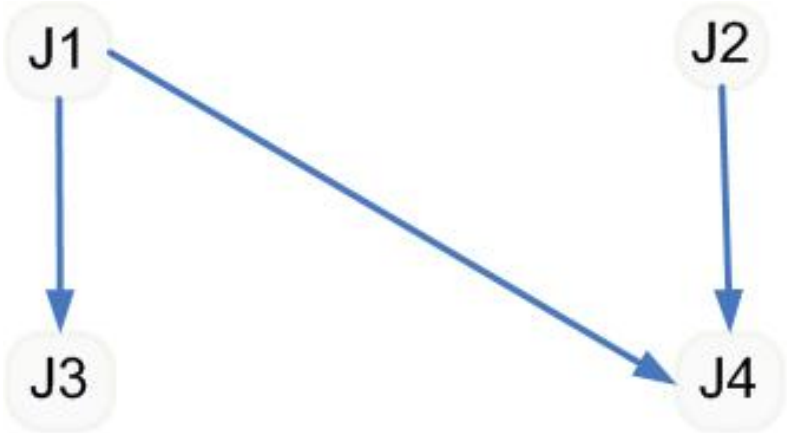
- Se tiene conjunto de personas P = $\{P_1, P_2, \ldots, P_n\}$ ordenado linealmente
- Y un conjunto de trabajos J = $\{J_1, J_2, \ldots, J_n\}$ parcialmente ordenados
- A cada persona se le asigna un trabajo f(P_i) y f(P_j)
- Si f(P_i) $\leq$ f(P_j)entonces $P_i \leq P_j$
- Donde la función f se interpreta como una signación factible que mapea personas en sus trabajos idoneos.

Problema de Asignación de personal

- Por ejemplo:
  - Dados los conjuntos P $= \{P_1, P_2, P_3\}$ y J $= \{J_1, J_2, J_3\}$
  - Ordenamiento parcial de J $J_1 \leq J_3$ y $J_2 \leq J_3$
  - Convinaciones factibles $P_1 \rightarrow J_1$, $P_2 \rightarrow J_2$, $P_3 \rightarrow J_3$
- Exiate un costo C_{ij} de asigna un trabajo a cada persona, a ningun para de personas se le asigna el mismo trabajo
- y Sea X_{ij} igual 1 si se asigna un trabajo y $= 0$ en caso contrario
- Entonces CT $= \Sigma\ C_{ij} X_{ij}$

Problema de Asignación de personal

Ordenamiento parcial para trabajos J $= \{J_1, J_2, J_3, J_4\}$

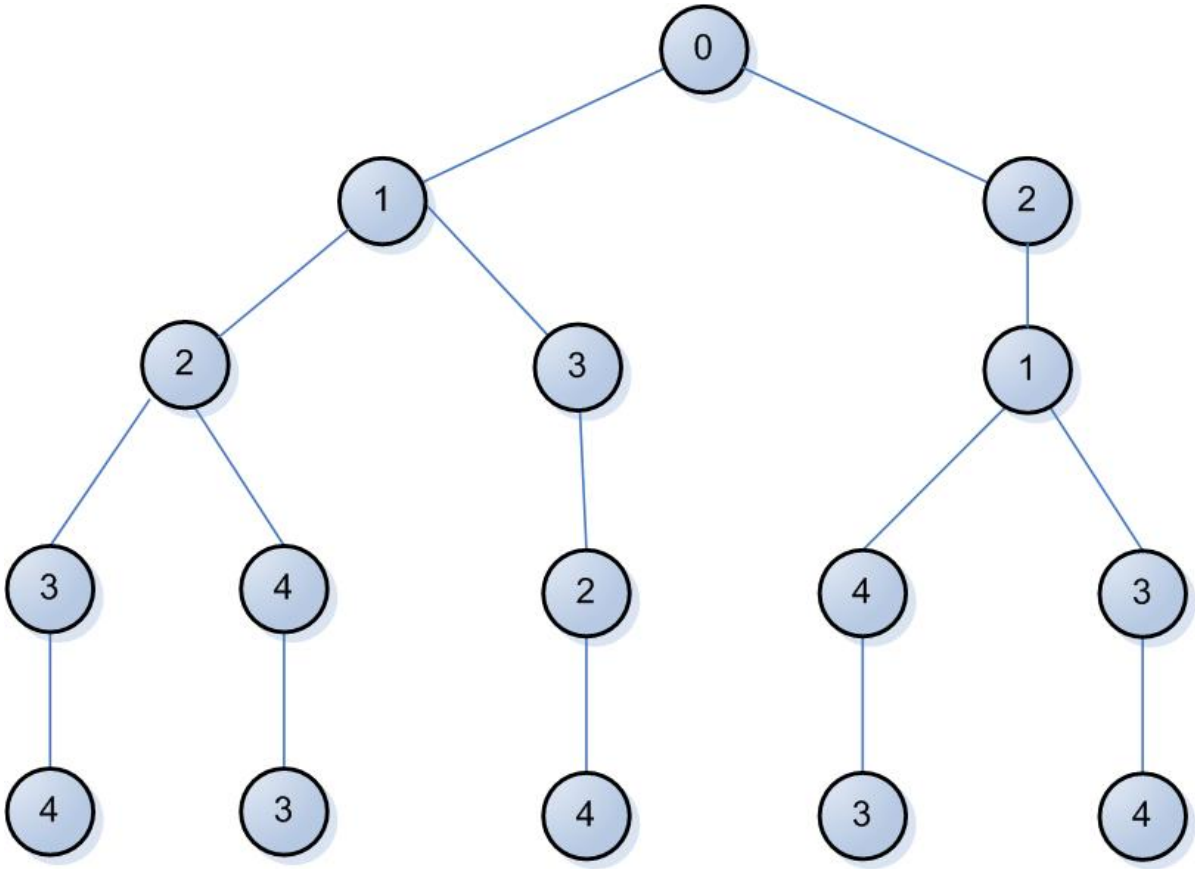


Las siguientes secuancias estan ordenadas topológicamente

- J_1, J_2, J_3, J_4n
- J_1, J_2, J_4, J_3
- J_1, J_3, J_2, J_4
- J_2, J_1, J_3, J_4
- J_2, J_1, J_4, J_3

Problema de Asignación de personal

Representación del árbol con todas las secuencias ordenadas topológicamente



Representa el espacio solución de nuestro problema

Estrategia Ramificar y Acotar (Asignación de personal)

Matriz de costos para un problema de asignación de personal

	Trabajos			
personas	**1**	**2**	**3**	**4**
1)	29	19	17	12
2)	32	30	26	28
3)	3	21	7	9
4)	18	13	10	15

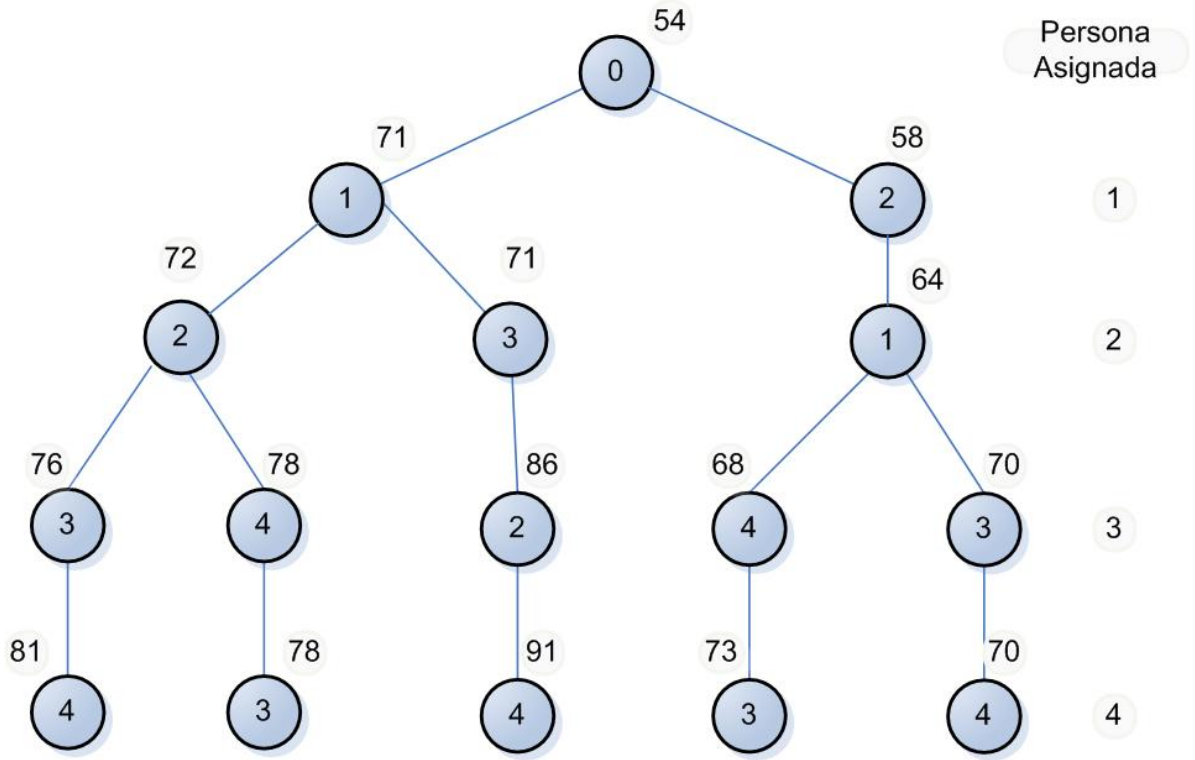
Matriz reducida (No puede tener valores negativos)

	Trabajos			
personas	**1**	**2**	**3**	**4**
1)	17	4	5	0
2)	6	1	0	2
3)	0	15	4	6
4)	8	0	0	5

Total $= 12+26+3+10+3 = 54 \rightarrow$ Cota inferior

Problema de Asignación de personal

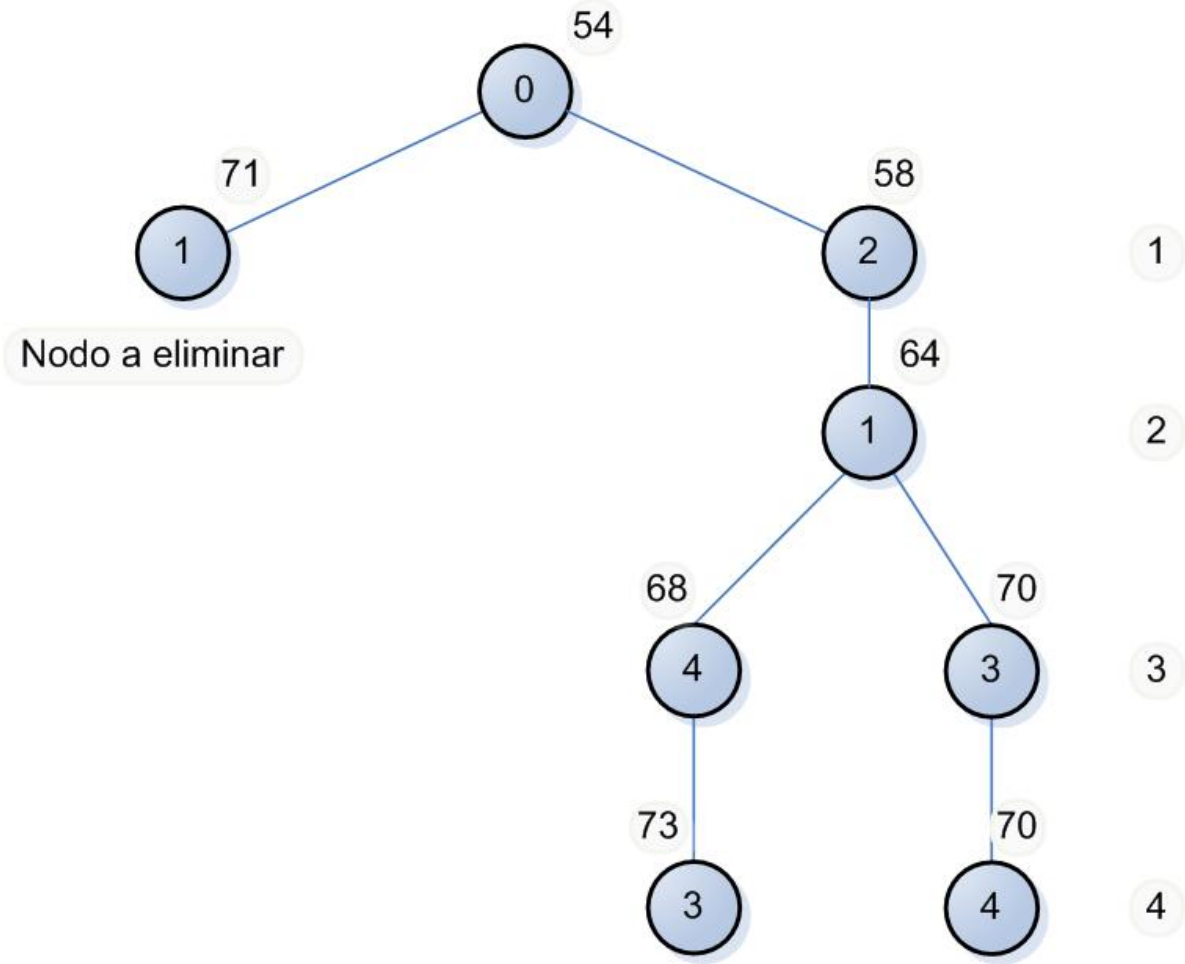
Árbol de enumeración asociado con la matriz de costos reducida



Las soluciones que no son óptimas se podan en una etapa mas temprana

Problema de Asignación de personal

Acotamiento de las subsoluciones



Es posible acotar el trabajo 1 porque toma un valor grande al aplicar la cota inferior

Problema de Optimización del agente viajero

- Hay una forma de dividir el espacio solución
- Hay una forma de pronosticar una cota inferior
- Hay una forma de encontrar una cota superior que se compara con la cota inferior
- Si la cota inferior es mayor a la cota superior la solución no es óptima
- Entonces se elimina la ramificación asociada a esa soluión
-
- El problema se define sobre una gráfica de n puntos planos
- Entre cada par de vértices hay un arco asociado con un costo no negativo
- El problema consiste en encontrar un recorrido que pase por todos los vértices y regrese al vértice inicial con costo mínimo.

Problema de Optimización del agente viajero

Matriz de Costos para el problema de Agente Viajero

		j					
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	∞	3	93	13	33	9	57
2	4	∞	77	42	21	16	34
3	45	17	∞	36	16	28	25
4	39	90	80	∞	56	7	91
5	28	46	88	33	∞	25	57
6	3	88	18	46	92	∞	7
7	44	26	33	27	84	39	∞

La estrategia de ramificar y acotar divide l solución en dos grupos: un grupo que incluye un arco particular y otro grupo que excluye este arco. Cada división incurre en una cota inferior y el árbol de búsqueda se atravesará con la cota inferior "más baja "

Problema de Optimización del agente viajero

Matriz de costos reducida

			j				
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	∞	0	90	10	30	6	54
2	0	∞	73	38	17	12	30
3	29	1	∞	20	0	12	9
4	32	83	73	∞	49	0	84
5	3	21	63	8	∞	0	32
6	0	85	15	43	89	∞	4
7	18	0	7	1	58	13	∞

El costo restado es $3 + 4 + 16 + 7 + 25 + 3 + 26 = 84$ Se puede reducir aun mas la matriz restando 7, 1 y 4 a las columnas 3, 4 y 7 respectivamente

Problema de Optimización del Agente Viajero

Matriz de costos reducida

		j					
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	∞	0	83	9	30	6	50
2	0	∞	66	37	17	12	26
3	29	1	∞	19	0	12	5
4	32	83	66	∞	49	0	80
5	3	21	56	7	∞	0	28
6	0	85	8	42	89	∞	0
7	18	0	0	0	58	13	∞

El costo restado es $7 + 1 + 4 = 12$ El costo Total es : $84 + 12 = 96 \rightarrow$ Cota inferior

Problema de Optimización del agente viajero

Considerando el siguiente problema : suponga que se sabe que un recorrido incluye el arco 4-6 cuyo costo es cero,¿Cuál es la cota inferior? la cota inferior sigue siendo 96
Suponga que se sabe que el recorrido excluye el arco 4-6 ¿Cuál es la cota inferior?. Si un recorrido no incluye el arco 4-6 entonces debe incluir algun otro arco que salga de 4. El arco con el menosr costo que sale de 4 es el arco 4-1 cuyo costo es 32, el arco con el menor costo que sale de 6 es el 5-6. Asi la nueva cota inferior es $96 + (32 + 0) = 128$
Se escogio el arco 4-6 para la división porque provoca el mayor incremento de la cota inferior

Problema de Optimización del agente viajero

Matriz de costos reducida si se incluye 4-6

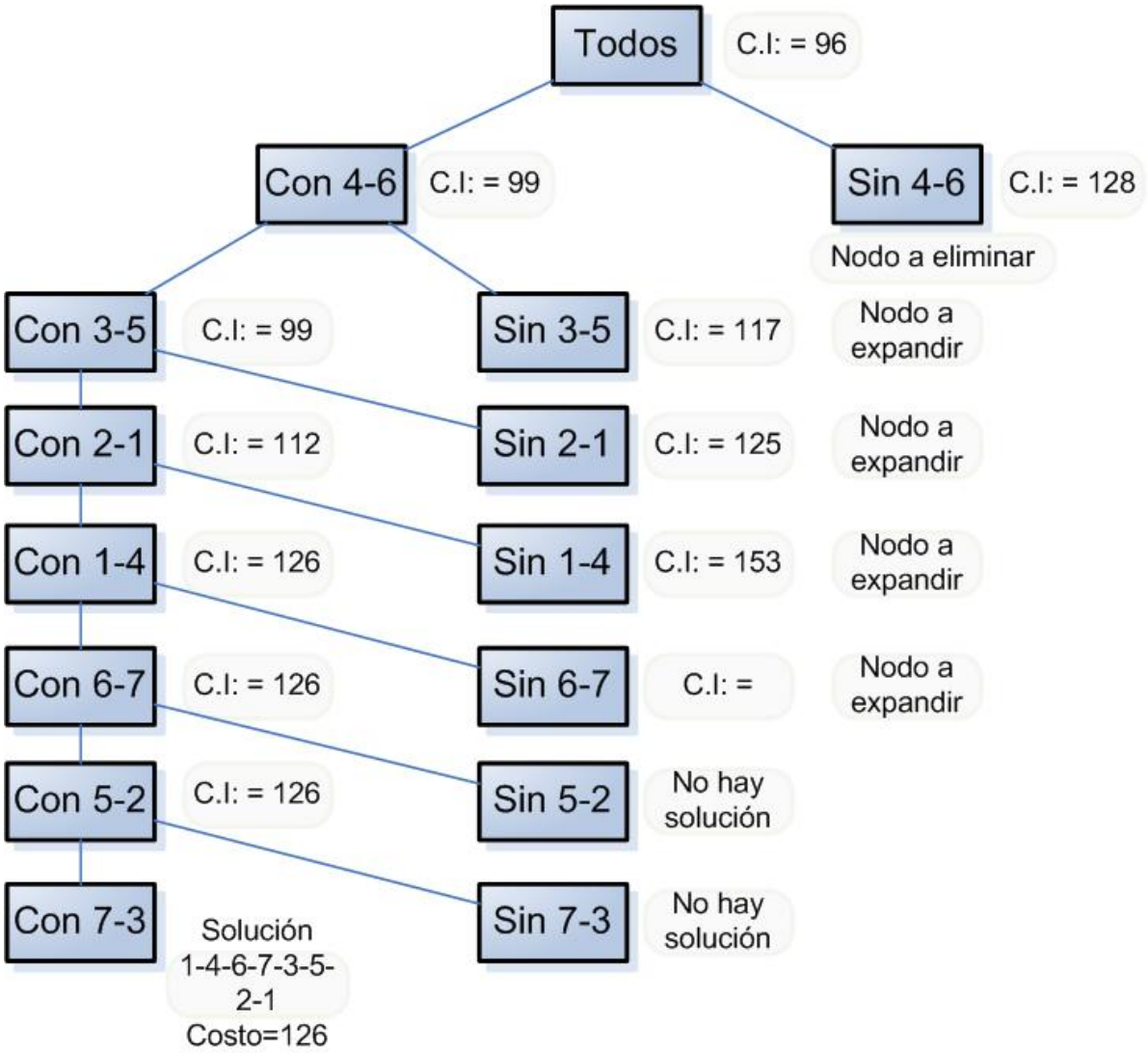
	j					
i	**1**	**2**	**3**	**4**	**5**	**7**
1	∞	0	83	9	30	50
2	0	∞	66	37	17	26
3	29	1	∞	19	0	5
5	3	21	56	7	∞	28
6	0	85	8	∞	89	0
7	18	0	0	0	58	∞

Problema de Optimización del agente viajero

			j			
i	**1**	**2**	**3**	**4**	**5**	**7**
1	∞	0	83	9	30	50
2	0	∞	66	37	17	26
3	29	1	∞	19	0	5
5	0	18	53	4	∞	25
6	0	85	8	∞	89	0
7	18	0	0	0	58	∞

Cota inferior $96 + 3 = 99$

Problema de Optimización del agente viajero



Problema de Optimización del agente viajero

La solución factible de costo 126 sirve como cota superior y permite eliminar muchas ramificaciones porque sus cotas inferiores exedena esta cota

Matriz de costos reducida de todas las soluciones con los arcos 4-6,3-5 y 2-1 incluidos

	j			
i	**2**	**3**	**4**	**7**
1	∞	74	0	41
5	14	∞	0	21
6	85	8	∞	0
7	0	0	0	∞

Problema de Optimización del agente viajero

El arco 1-4 puede usarse para dividir y el árbol resultantes es

	j		
i	**2**	**3**	**7**
5	14	∞	21
6	85	8	0
7	0	0	∞

Si se usa el arco 6-2 se forma un ciclo, es necesario igualarlo a ∞, en consecuancia se tendra la matriz de costos

	j		
i	**2**	**3**	**7**
5	14	∞	21
6	∞	8	0
7	0	0	∞

Calendarizacion del trabajo (B&B)

Introducción

Que tenemos que hacer?

Dada un ordenamiento parcial y un perfil temporal; encontar una solucion con los pasos temporales minimos para terminar todos los trabajos. Es un Problema NP-Dificil.

Como vamos a hacerlo?

Usaremos el algoritmo de (BRANCH-and-BOUND) Ramificar y Acotar con algunas reglas.

Calendarizacion del trabajo (B&B)

El problema de basa en las siguientes hipotesis:

1. Todos los procesadores son identicos y cualquier trabajo puede realizarse en cualquier procesador.
2. Hay un ordenamiento parcial de trabajos. Un trabajo no puede ejecutarse si uno de sus trabajos precedentes, si existe, aun no se haya ejecutado.
3. Siempre que un procesador esté ocioso y haya un trabajo por efectuar, este procesador debe empezar a realizar el trabajo.
4. Cada trabajo requiere el mismo tiempo de ejecución.
5. Se proporciona un perfil temporal que especifica el numero de procesadores que es posible usar simultaneamente en en cada periodo.

Calendarizacion del trabajo (B&B)

Objetivo:

El objetivo de la candelarización es minimizar el tiempo de terminacion máximo, que es el periodo en que se termina el último trabajo.

Calendarizacion del trabajo (B&B)

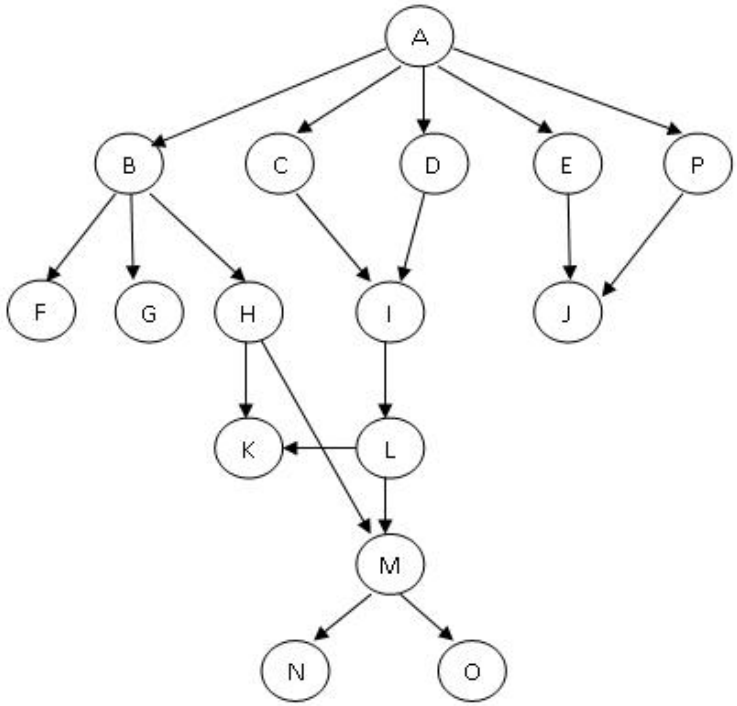


Figura: 1. Ordenamiento Parcial

Calendarizacion del trabajo (B&B)

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
3	2	2	2	4	5	3	2	3

Cuadro: 1. Perfil Temporal

En este perfil temporal se indica que en el tiempo $t = 1$, sólo pueden usarse 3 procesadores y que en $t = 5$, pueden usarse 4 procesadores.

Calendarizacion del trabajo (B&B)

T_1	T_2	T_3	T_4	T_5	T_6
A	B	C	H	M	J
$*$	D	I	L	E	K
$*$				F	N
				P	O
					G

Tiempo $= 6$

Cuadro: 2. Solución 1

Calendarizacion del trabajo (B&B)

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
A	B	D	F	H	L	M	N
$*$	C	E	G	I	J	K	O
$*$				P	$*$	$*$	
				$*$	$*$		
					$*$		

Tiempo $= 8$

Cuadro: 3. Solución 2

Calendarizacion del trabajo (B&B)

Resulta evidente que la Solución 1 es mejor que la Solución 2.
Se tiene una grafica con ordenamiento parcial (Figura 1) y un perfil temporal(Cuadro 1); se pide encontrar una solucion con los pasos temporales minimos para terminar todos los trabajos.

Calendarizacion del trabajo (B&B)

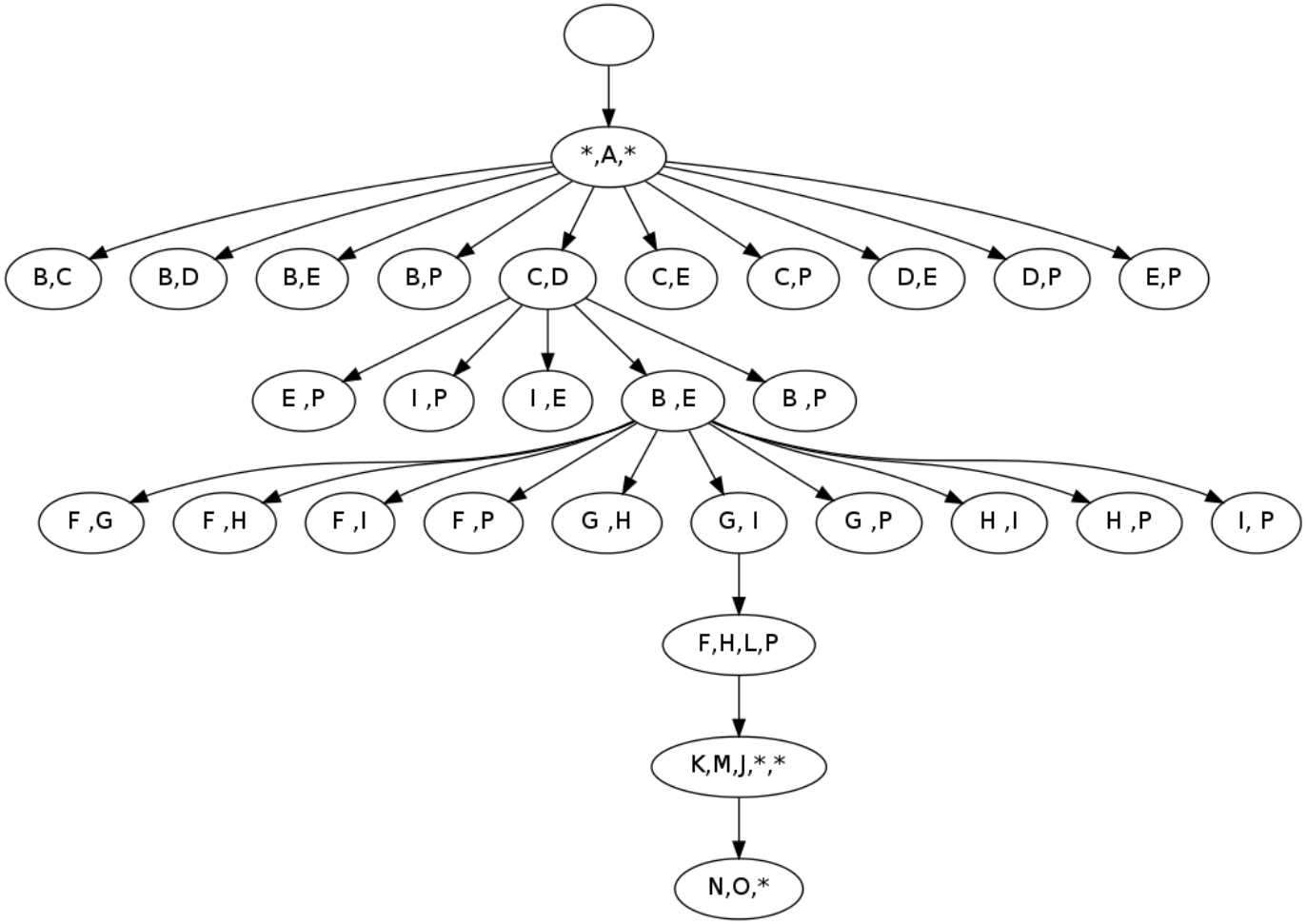


Figura: 2. Parte de un arbol de Solución

Calendarizacion del trabajo (B&B)

Para la solucion de este problema usaremos reglas adicionales para resolverlo con la estrategia ramificar y acotar.

Regla 1:

Efecto de sucesores comunes.

Consideremos la figura 1, los trabajos C y D comparten el mismo descendiente, trabajo I; de igual manera los trabajos E y P comparten a J. Entonces esta regla nos dice que debemos desarrollar solo C,E o D,P, ya que cualquier solucion optima encabezada por cualquier par, tiene el mismo costo;

Calendarizacion del trabajo (B&B)

Regla 2:

Estrategia del nodo interno.

El nodo interno de la grafica de procedencia de trabajos debe procesarse antes que el nodo hoja. Para mostrar esto consideraremos el siguente perfil y la grafica 1.

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
1	3	2	3	2	2	3	2	3

Cuadro: 4. Perfil Temporal

Calendarizacion del trabajo (B&B)

Estrategia del nodo interno

1. Dividimos los procesos en nodos internos activos y nodo hoja activos.
2. La prioridad la tienen los nodos internos activos asi que tratamos de procesarlos todos.
3. Cuando el numero de procesadores activos es mayor que el conjunto de nodos internos activos, escojemos los del otro grupo de manera arbitraria.

Calendarizacion del trabajo (B&B)

Regla 3:

Maximizacion del número de trabajos procesados.

- Sea $P(S', i)$ un conjunto de calendarios ya procesados en la candelarización S desarrollada parcial o completamente. desde el periodo 1 hasta i:
- Un calendario S no es una solución óptima si $P(S, i)$ esta contenido en $P(S', i)$ para algun otro calendario S'.

Calendarizacion del trabajo (B&B)

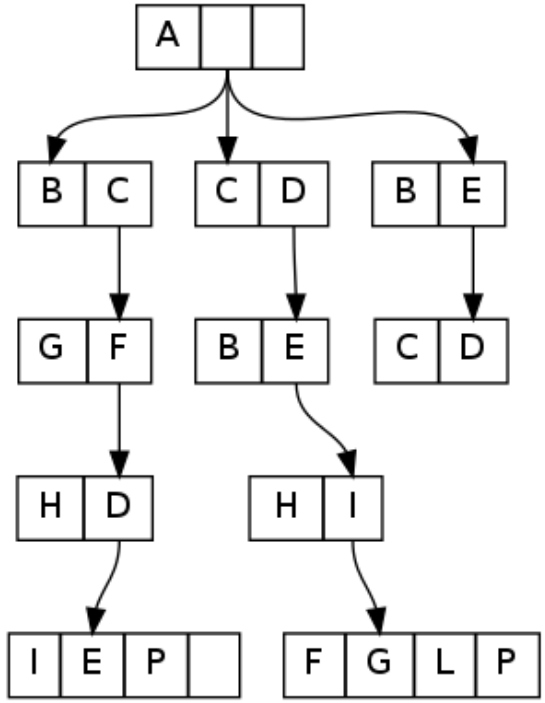


Figura: 3. Efecto de trabajos procesados

Calendarizacion del trabajo (B&B)

Explicacion

$$\begin{cases} P(S',3) &= (A,B,C,D,E) \\ P(S,3) &= (A,B,C,D,E) \end{cases}$$

$$P(S',3) = P(S,3)$$

$$\begin{cases} P(S',5) &= (A,B,C,D,E,F,G,H,I,P) \\ P(S,5) &= (A,B,C,D,E,F,G,H,I,P,L) \end{cases}$$

$P(S',5)$ está contenido en $P(S,5)$

Calendarizacion del trabajo (B&B)

Regla 4:

Estrategia de procesadores ociosos acumulados.

- Cualquier calendario parcialmente desarrollado con el numero de procesadores ociosos acumulados mayor que el de un calendario factible no puede ser mejor que esta solucion factible, y en cosecuencia es posible eliminarlo.

Calendarizacion del trabajo (B&B)

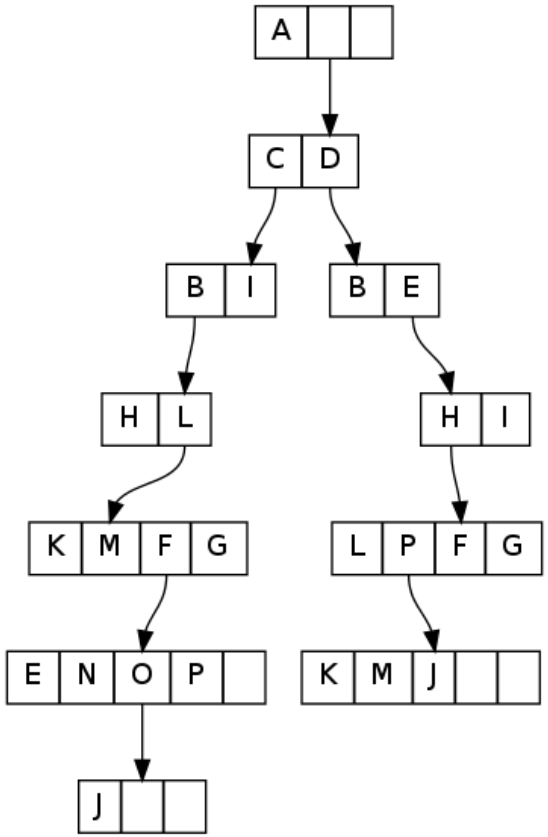


Figura: 4. Efecto de trabajos procesados

El Algoritmo A*

- Este algoritmo constituye una buena estrategia de arbol de busqueda bastante favorecida por los investigadores de dedicados a la inteligencia artificial.
- Una solucion factible que se haya obtenido debe ser optima.
- Se aplica a problemas de optimizaciòn.
- Utiliza la estrategia de primero el mejor (primero el menor costo).
- Una de los elementos criticos es la funcion de costos.
- Si un nodo seleccionado tambien es un nodo meta, entonces este nodo representa una solucion optima y el proceso termina.

El Algoritmo A*

Definimos:

$g(n) =$ *La ruta desde la raiz hasta un nodo n*

$h^*(n) =$ *La ruta minima desde n hasta un nodo meta*,el algoritmo señala que $h*(n)$ se calcula haciendo una estimacion de $h(n) \leq h^*(n)$.

$F(n) = g(n) + h^*(n)$, *La funcion de costo*

El Algoritmo A*

Reglas fundamentales

1. El algoritmo A^* usa la regla de primero el mejor.
2. En el algoritmo A^*, la funcion de costo $f(n)$ de define como sigue:
   $f(n) = g(n) + h(n)$
   donde $g(n)$ es la longitud de la ruta que va de la raiz del arbol a n y $h(n)$ es la estimacion de h^*(n) que es la longitud de la ruta optima desde n hasta algun nodo meta.
3. $h(n) \leq h^*(n)$ para toda n.
4. El algoritmo A^* se detiene si y solo si el nodo seleccionado es nodo meta y regresa una solucion óptima.

El Algoritmo A*

Ejemplo:

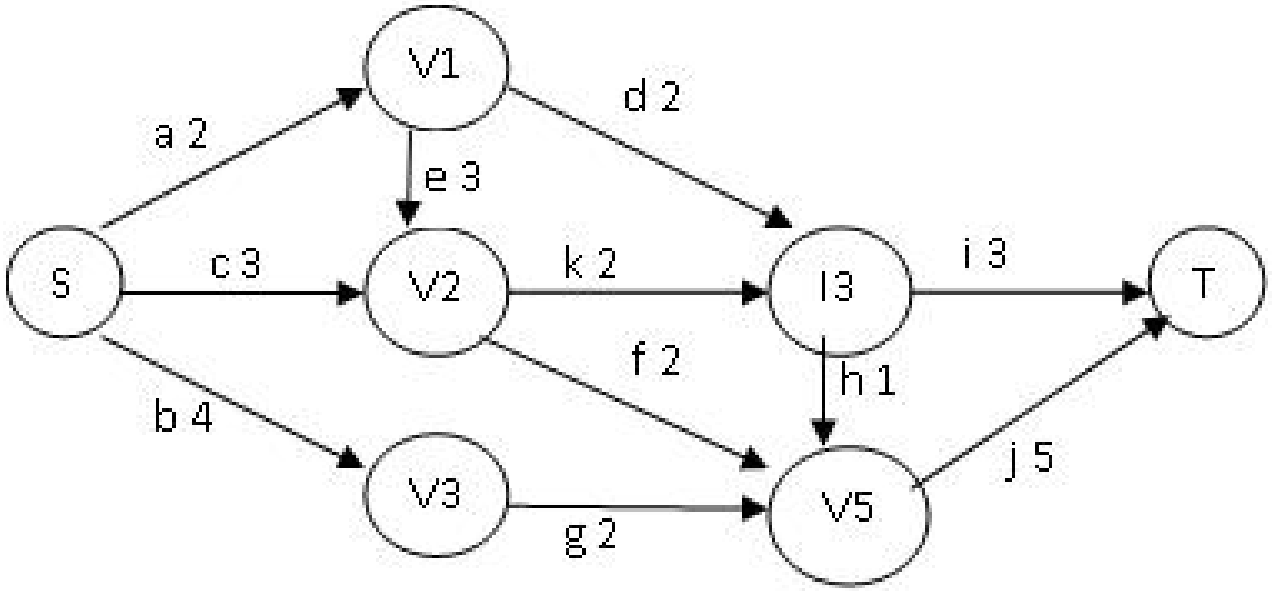


Figura: Grafica

El Algoritmo A*

Expandir R

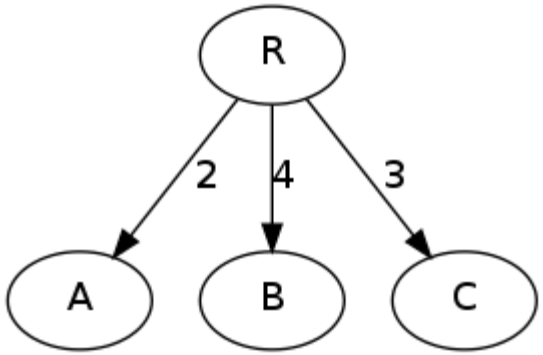


Figura: Primer Paso

$$g(A) = 2 \quad h(A) = min\{2, 3\} = 2 \quad f(A) = 2 + 2 = 4$$
$$g(B) = 2 \quad h(B) = min\{2\} = 2 \quad f(B) = 4 + 2 = 6$$
$$g(C) = 2 \quad h(C) = min\{2, 2\} = 2 \quad f(C) = 3 + 2 = 5$$

El Algoritmo A*

Expandir A

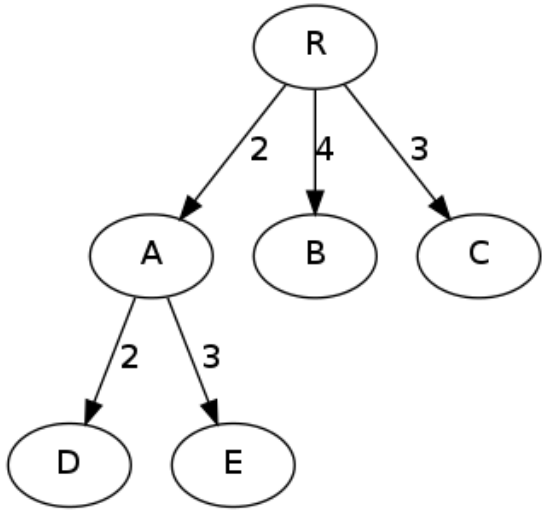


Figura: Segundo Paso

$g(D) = 2 + 2 = 4$ $h(D) = min\{3, 1\} = 2$ $f(D) = 4 + 1 = 5$

$g(E) = 2 + 3 = 5$ $h(E) = min\{2, 2\} = 2$ $f(E) = 5 + 2 = 7$

El Algoritmo A*

Expandir C

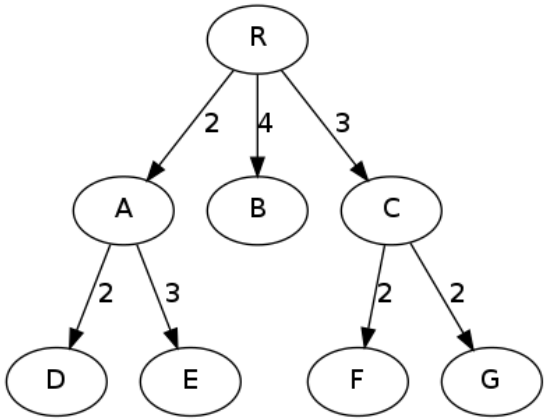


Figura: Tercer Paso

$$g(F) = 3 + 2 = 5 \qquad h(F) = min\{3, 1\} = 2 \qquad f(F) = 5 + 1 = 6$$
$$g(G) = 3 + 2 = 5 \qquad h(G) = min\{5\} = 5 \qquad f(G) = 5 + 5 = 10$$

El Algoritmo A*

Expandir D

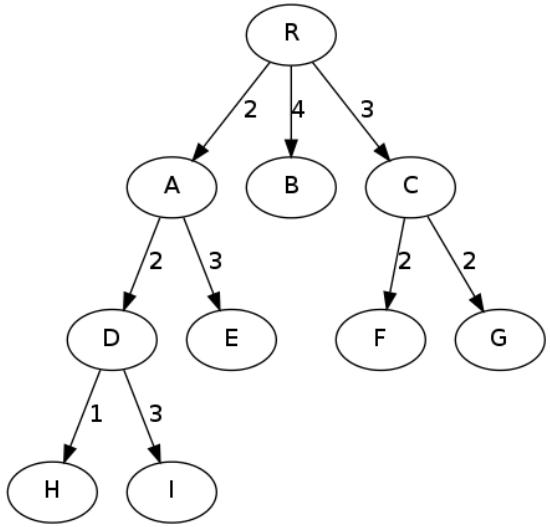


Figura: Cuarto Paso

$g(H) = 2 + 2 + 1 = 5 \quad h(H) = min\{5\} = 5 \quad f(H) = 5 + 5 = 10$

$g(I) = 2 + 2 + 3 = 7 \quad h(I) = 0 \quad f(I) = 7 + 0 = 7$

El Algoritmo A*

Expandir B

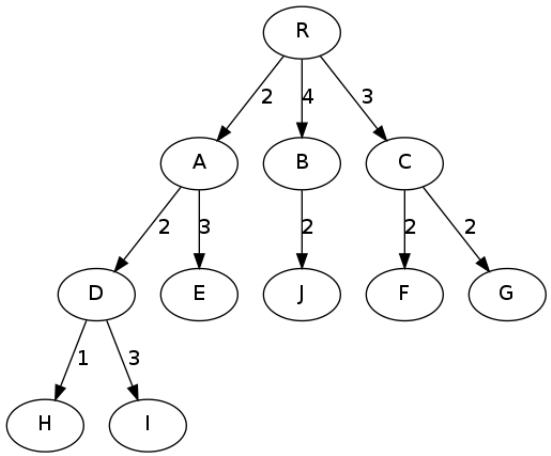


Figura: Quinto Paso

$$g(J) = 4 + 2 = 6 \qquad h(J) = min\{5\} = 5 \qquad f(J) = 6 + 5 = 11$$

El Algoritmo A*

Expandir F

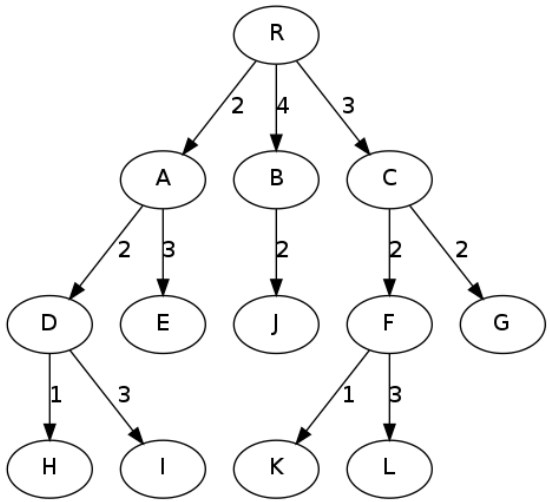


Figura: Sexto Paso

$g(K) = 3 + 2 + 1 = 6 \quad h(K) = min\{5\} = 5 \quad f(K) = 6 + 5 = 11$

$g(L) = 3 + 2 + 3 = 8 \quad h(L) = 0 \quad f(L) = 8 + 0 = 8$

El Algoritmo A*

Expandir I

Debido a que I es un nodo meta el procedimiento llega a un alto y como solución óptima se regresa:
$S \rightarrow V_1 \rightarrow V_4 \rightarrow T$
Septimo Pasox

El Algoritmo A*

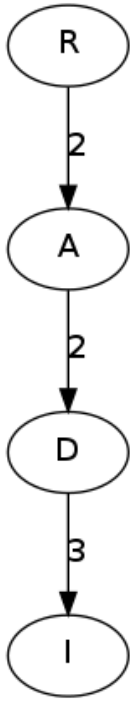


Figura: Solución

Problema de direccion de canales con A*

Consideremos la grafica ...

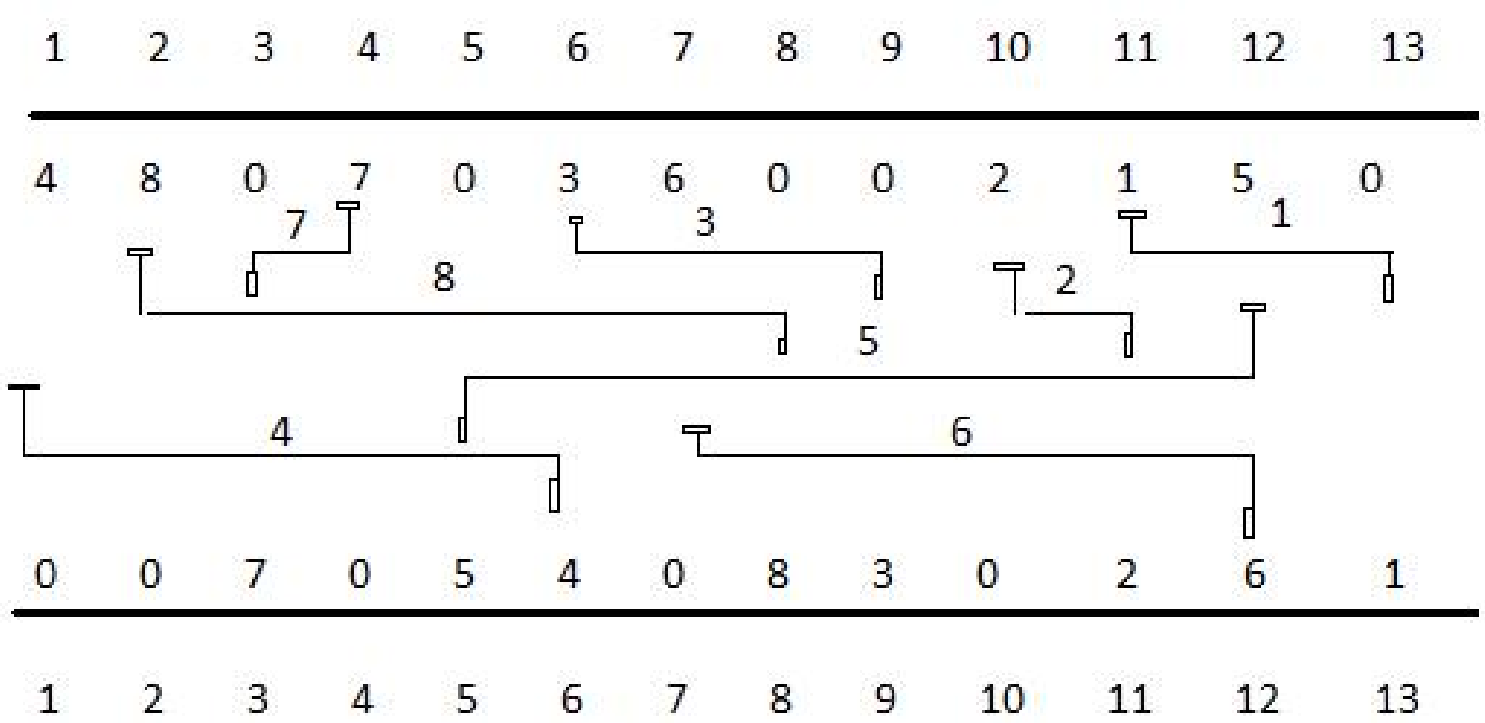


Figura: 6. Grafica de ejemplo

Problema de direccion de canales con A*

Que tenemos?

- Un problema NP-dificil. - Un canal con dos reglones de terminales. - Un conjunto de redes etiquetadas del 1-7. (hay q notar que la red 7 conecta la terminal 3 con la terminal 4). Que nos Piden?

Minimizar el número de pistas.

Problema de direccion de canales con A*

Restricciones Horizontales

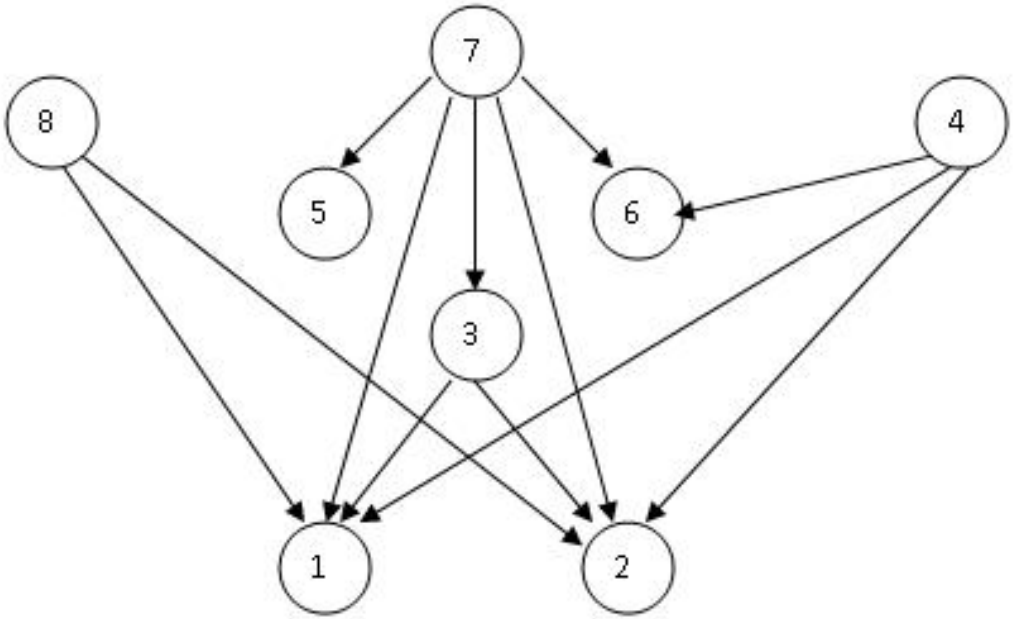


Figura: 6. Grafica de ejemplo

Problema de direccion de canales con A*

Restricciones Verticales

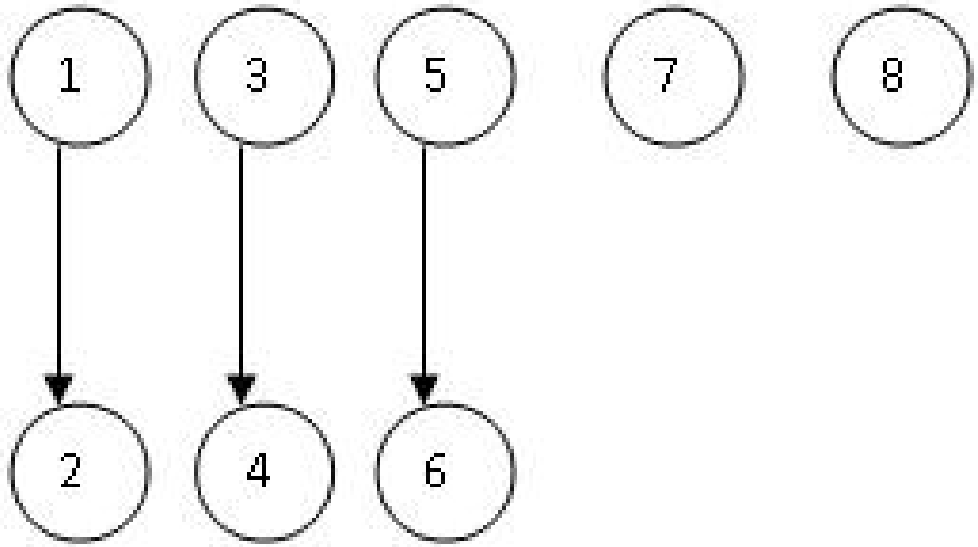


Figura: 6. Grafica de ejemplo

Problema de direccion de canales con A*

Para resolver el problema necesitamos dos funciones $g(n)$ y $h(n)$.

- $g(n)$ puede definirse como el nivel del arbol.
- $h(n)$ se puede calcular usando el numero minimo de pistas que deben usarse despues de que se ha asignado cierto numero de piezas.
- $h(t) = h^{'}(t) = 1 = g(meta) - g(t)$

Problema de direccion de canales con A*

Primer nivel del arbol

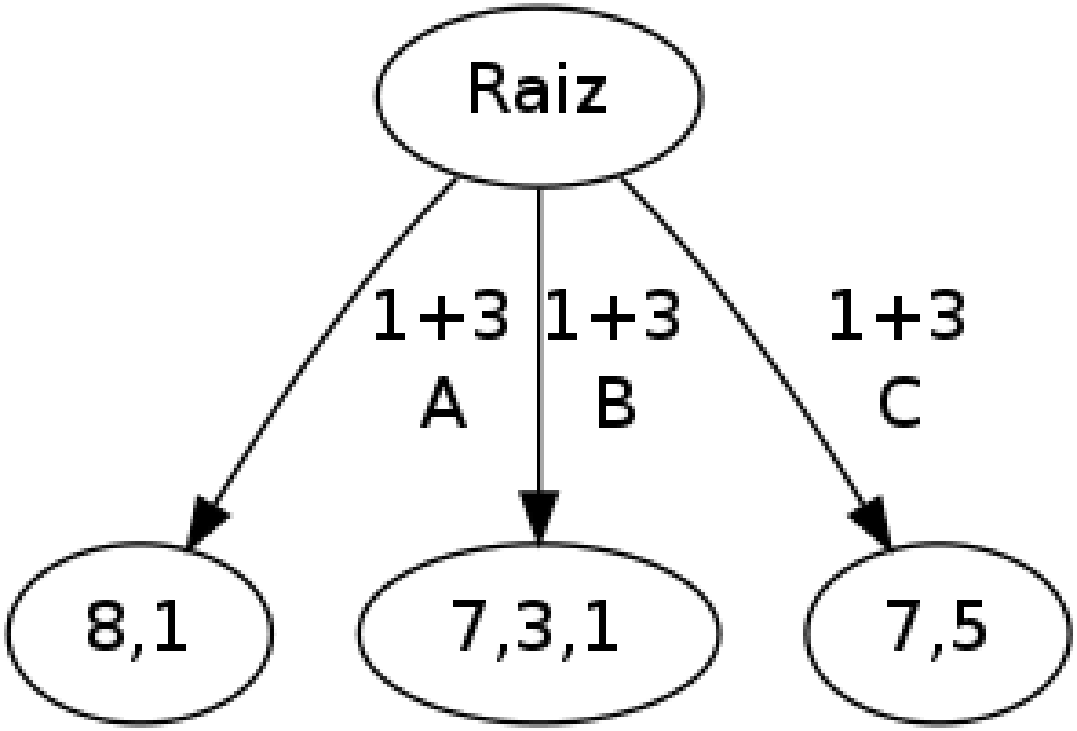


Figura: 7. Primer Arbol

Problema de direccion de canales con A*

Como lo resolvemos?

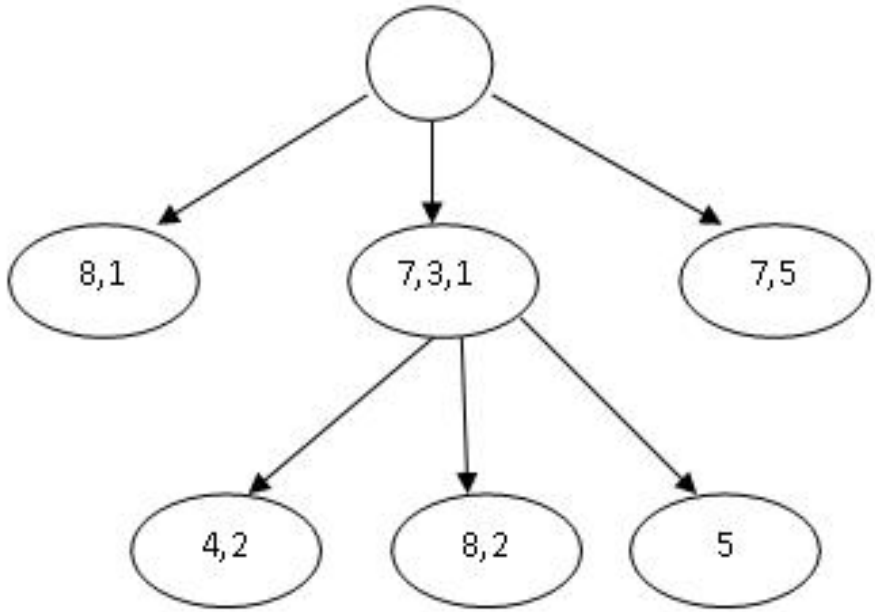


Figura: 8. Grafica de ejemplo

Problema de direccion de canales con A*

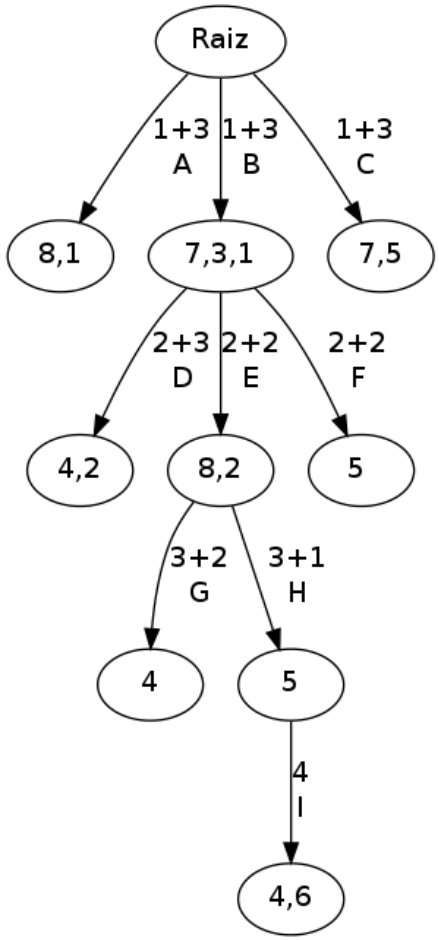


Figura: 9. Arbol de solución parcial

Problema de decodificacion de bloque lineal con A*

Que tenemos que hacer??

Decodifcar mensajes binarios, por ejemplo:

supongamos que tenemos que mandar los numero del 0-7 pero en binario usando 3 bits($0 = 000$) entonces nosotros no enviariamos 0 sino 000 asi asi sucesivamente, lo que causa problema es que el mensaje enviado puede cambiar por el ruido del canal de transmicion, si mandamos 100 y recibimos 000 es un error demasiado peligroso.
Entonces lo que hacemos es codificar la palabra y en vez de enviar 100 enviamos 100110 entonces calculando la distacia de hamming y buscando la diferencia minima entre las palabras codigo y las recibidas podemos decodificar el mensaje y evitar errores.

Problema de decodificacion de bloque lineal con A*

Como resolvemos este problema??
Lo que hacemos es usar un arbol de codigo para representar todas las palabras codigo.
La decodificacion de un bloque recibido es encontrar una palabra codigo lo mas cercana a la recibida, lo cual es un problema de arbol de busqueda.

Para ello el bloque enviado debe de ser tranformado a -1(1) y 1(0) antes de enviar.

$$G(n) = \sum_{i=0}^{t} (r_i - (-1)_i^c)^2$$

$$H(n) = \sum_{i=0}^{t} (|r_i| - 1)^2$$

Problema de decodificacion de bloque lineal con A*

FIN