

---

**capítulo****5**

## **La estrategia de árboles de búsqueda**

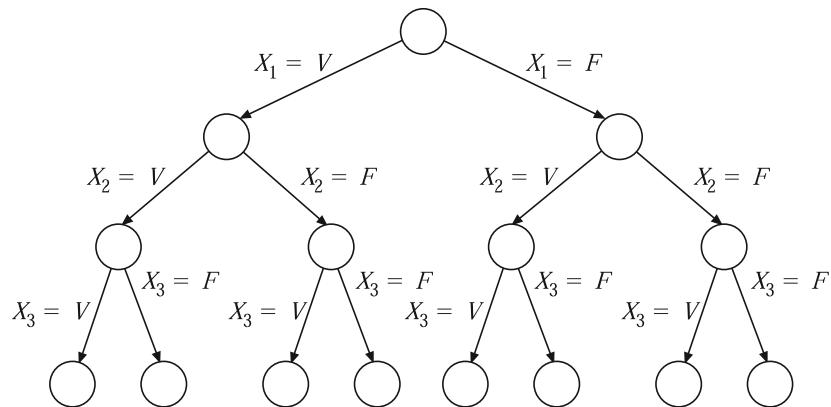
En este capítulo se demostrará que las soluciones de muchos problemas pueden representarse mediante árboles, por lo que resolver tales problemas se convierte en un problema de árboles de búsqueda. Se considerará el problema de satisfacibilidad que se abordará en el capítulo 8. Dado un conjunto de cláusulas lógicas, un método para determinar si este conjunto de cláusulas es satisfacible consiste en examinar todas las combinaciones posibles. Es decir, si hay  $n$  variables  $x_1, x_2, \dots, x_n$ , entonces simplemente se analizan todas las  $2^n$  combinaciones posibles. En cada combinación, a  $x_i$  se le asigna un valor ya sea  $V$  (Verdadera) o  $F$  (Falsa). Suponga que  $n = 3$ . Entonces es necesario analizar las siguientes combinaciones:

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

Las ocho combinaciones anteriores, por otra parte, pueden representarse mediante un árbol, como se muestra en la figura 5-1. ¿Por qué es informativo el árbol? Porque una representación de árbol de las combinaciones muestra que en el nivel superior realmente hay dos clases de combinaciones:

**Clase 1:** Las combinaciones en que  $x_1 = V$ .

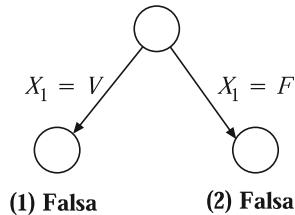
**Clase 2:** Las combinaciones en que  $x_1 = F$ .

**FIGURA 5-1** Representación de árbol de ocho combinaciones.

Así, de manera recursiva es posible clasificar cada clase de combinaciones en dos subclases. Con este punto de vista, es posible determinar la satisfacibilidad sin necesidad de analizar todas las combinaciones: sólo todas las clases de combinaciones. Por ejemplo, suponga que se tiene el siguiente conjunto de cláusulas:

$\neg X_1$	(1)
$X_1$	(2)
$X_2 \vee X_5$	(3)
$X_3$	(4)
$\neg X_2$	(5)

Ahora es posible expandir un árbol parcial, que se muestra en la figura 5-2.

**FIGURA 5-2** Árbol parcial para determinar el problema de satisfacibilidad.

Con base en el árbol de la figura 5-2, puede verse fácilmente que la combinación de  $x_1 = V$  hace falsa la cláusula (1) y que la combinación de  $x_1 = F$  hace falsa la cláusula (2). Debido a que en toda combinación a  $x_1$  se asigna  $V$  o  $F$ , no es necesario examinar cada combinación. A continuación se establece la insatisfacibilidad del conjunto de cláusulas anterior.

Muchos otros problemas semejantes también pueden resolverse aplicando técnicas con árbol de búsqueda. Considere el famoso problema del rompecabezas (pasatiempo) de 8 piezas (8-puzzle). En la figura 5-3 se muestra un cuadrado de 3 por 3 que puede contener 9 piezas (mosaicos), aunque sólo tiene ocho piezas móviles numeradas, y una casilla vacía. El problema consiste en mover las piezas numeradas hasta alcanzar el estado final, que se muestra en la figura 5-4. Las piezas numeradas sólo pueden moverse horizontal o verticalmente hacia la casilla vacía. Así, para el arreglo original que se muestra en la figura 5-3, sólo hay dos movimientos posibles (mover el 3 o el 4), como se muestra en la figura 5-5.

**FIGURA 5-3** Posición inicial del problema del rompecabezas de 8 piezas.

2	3	
5	1	4
6	8	7

**FIGURA 5-4** Meta final del problema del rompecabezas de 8 piezas.

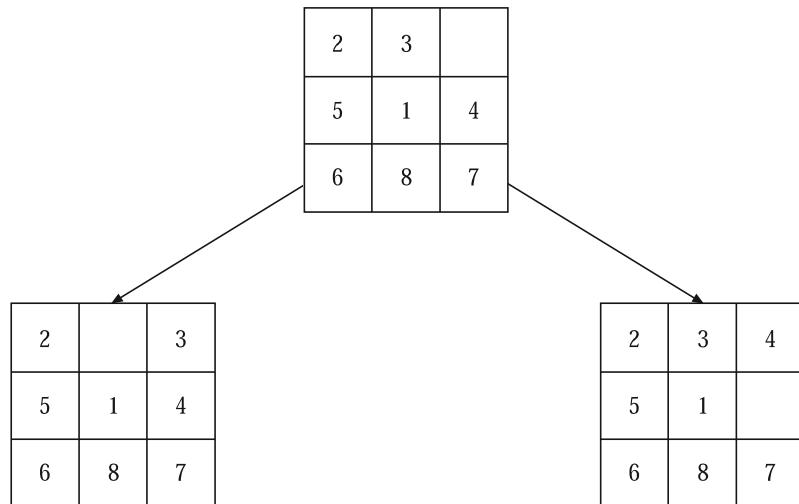
1	2	3
8		4
7	6	5

El problema del rompecabezas de 8 piezas se convierte en un problema de árbol de búsqueda porque gradualmente es posible ir expandiendo el árbol solución. El problema se resuelve cuando se alcanza el nodo que representa la meta final. Esto se presentará en apartados posteriores.

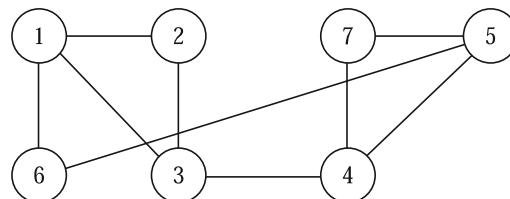
Por último, se demostrará que el espacio solución del problema del ciclo Hamiltoniano también puede representarse convenientemente por medio de un árbol. Dada una gráfica  $G = (V, E)$ , que es una gráfica conexa con  $n$  vértices, un ciclo Hamiltoniano es una ruta de viaje redondo a lo largo de  $n$  bordes de  $G$  que pasa una vez por cada vértice y después regresa a su posición inicial. Considere la figura 5-6. La ruta representada por la siguiente secuencia es un ciclo Hamiltoniano: 1, 2, 3, 4, 7, 5, 6, 1. Considere la figura 5-7. En esta gráfica no hay ningún ciclo Hamiltoniano.

El problema del ciclo Hamiltoniano consiste en determinar si una gráfica determinada contiene o no un ciclo Hamiltoniano. Éste es un problema NP completo. No

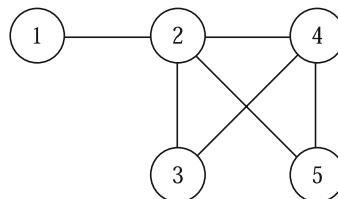
**FIGURA 5-5** Dos movimientos posibles para una posición inicial del problema del rompecabezas de 8 piezas.



**FIGURA 5-6** Gráfica que contiene un ciclo Hamiltoniano.



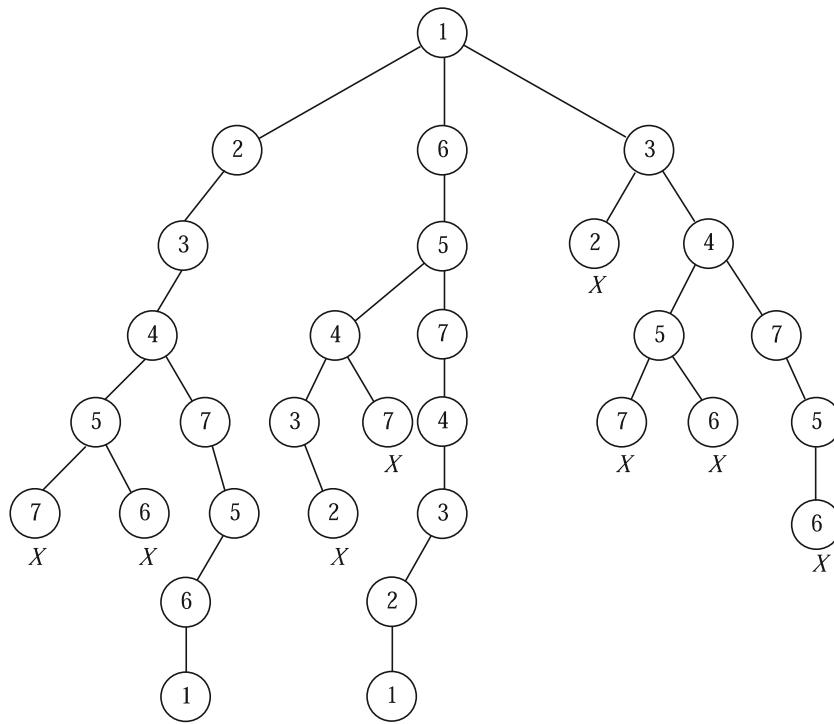
**FIGURA 5-7** Gráfica que no contiene ningún ciclo Hamiltoniano.



obstante, sigue siendo posible resolverlo mediante el análisis de todas las soluciones posibles, mismas que pueden representarse de manera conveniente por medio de un árbol. Observe que un ciclo Hamiltoniano debe pasar por cada nodo. En consecuencia, puede suponerse que el nodo 1 es el nodo inicial. Considere nuevamente la figura 5-6.

La búsqueda de un ciclo Hamiltoniano puede describirse mediante un árbol, que se muestra en la figura 5-8. Muestra que hay sólo un ciclo Hamiltoniano, ya que el ciclo 1, 2, 3, 4, 7, 5, 6, 1 es equivalente a 1, 6, 5, 7, 4, 3, 2, 1.

**FIGURA 5-8** Representación de árbol sobre la existencia o no de un ciclo Hamiltoniano en la gráfica de la figura 5-6.

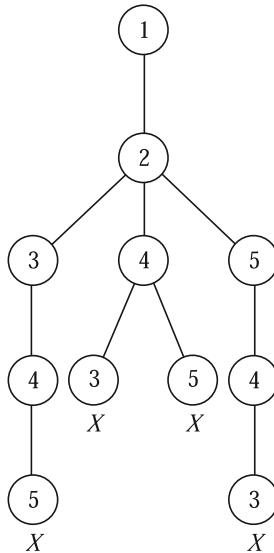


Si se analiza la figura 5-7, su representación de árbol sobre la existencia o no de un ciclo Hamiltoniano en la gráfica correspondiente se muestra en la figura 5-9. En este caso puede verse fácilmente que no existe ningún ciclo Hamiltoniano.

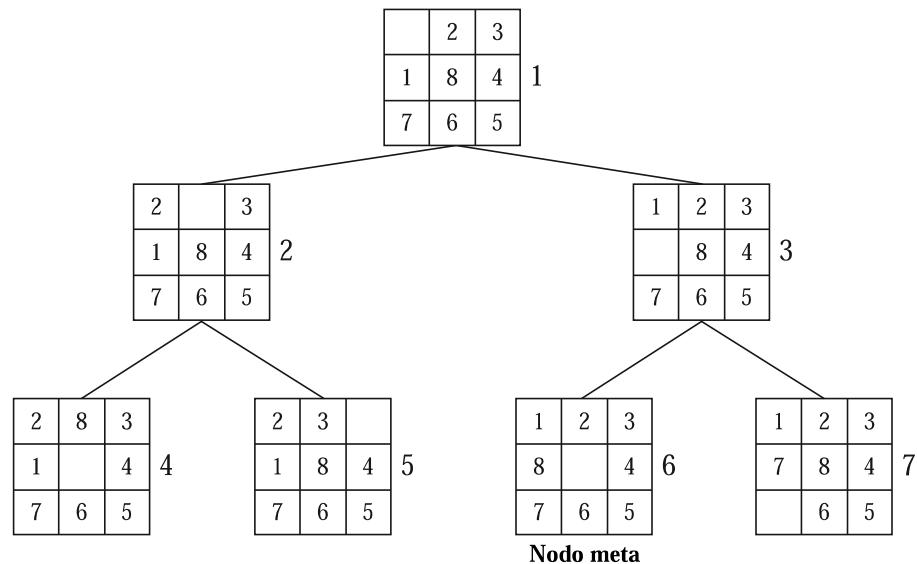
Se ha mostrado que muchos problemas pueden representarse por medio de árboles. En el resto de este capítulo se analizarán las estrategias de poda de árboles para resolver los problemas.

### 5-1 BÚSQUEDA DE PRIMERO EN AMPLITUD (BREADTH-FIRST SEARCH)

La búsqueda de primero en amplitud es quizás la forma más directa de podar un árbol. En esta búsqueda, todos los nodos de un nivel del árbol se revisan antes de analizar

**FIGURA 5-9** Árbol que muestra la inexistencia de ciclo Hamiltoniano.

los nodos del siguiente nivel. En la figura 5-10 se muestra una búsqueda de primero amplitud típica que resuelve un problema del rompecabezas de 8 piezas.

**FIGURA 5-10** Árbol de búsqueda producido por una búsqueda de primero amplitud.

Se observa que el nodo 6 representa un nodo meta. En consecuencia, es necesario detener la búsqueda. La estructura de datos básica de la búsqueda de anchura es una cola que contiene todos los nodos expandidos. El siguiente método ilustra la búsqueda de primero amplitud.

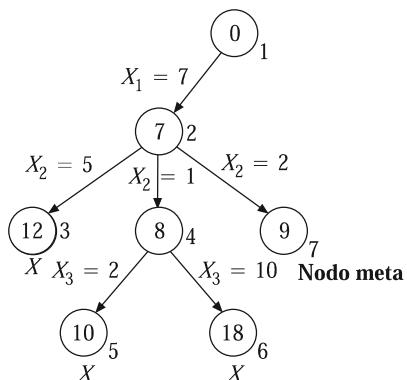
### Búsqueda de primero amplitud

- Paso 1.** Formar una cola de un elemento que consta del nodo raíz.
- Paso 2.** Probar si el primer elemento en la cola es un nodo meta. En caso afirmativo, detenerse; en caso contrario, ir a paso 3.
- Paso 3.** Quitar este 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 vacía, regresar Falla. En caso contrario, ir a paso 2.

## 5-2 / BÚSQUEDA DE PRIMERO EN PROFUNDIDAD (DEPTH-FIRST SEARCH)

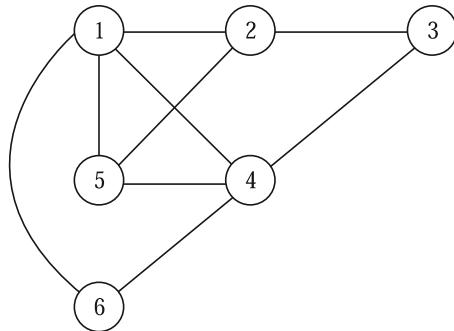
La búsqueda de primero en profundidad siempre escoge el nodo más profundo para expandirlo. Se considerará la siguiente suma del problema del subconjunto. Se tiene  $S = \{7, 5, 1, 2, 10\}$  y se quiere determinar si existe un subconjunto  $S'$  tal que la suma de elementos en  $S'$  sea igual a 9. Este problema puede resolverse fácilmente mediante el árbol de búsqueda de primero en profundidad que se muestra en la figura 5-11. En esta figura, muchos nodos están terminados porque resulta evidente que no llevan a ninguna solución. Los números encerrados en un círculo en la figura 5-11 representan la suma de un subconjunto. Observe que siempre se selecciona el nodo más profundo para expandirlo en el proceso.

**FIGURA 5-11** Una suma del problema del subconjunto resuelta por búsqueda en profundidad.

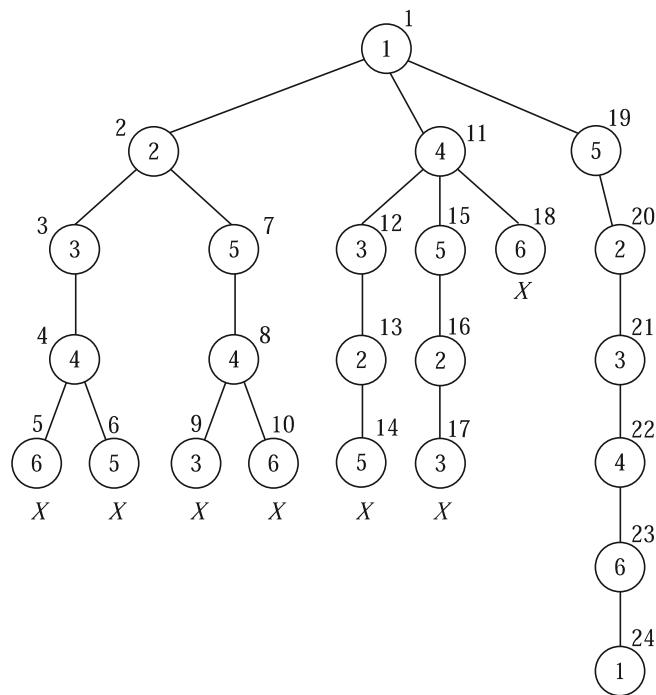


A continuación se considerará el problema del ciclo Hamiltoniano. Para la gráfica que se muestra en la figura 5-12, es posible encontrar un ciclo Hamiltoniano mediante búsqueda de primero en profundidad, que se muestra en la figura 5-13.

**FIGURA 5-12** Gráfica que contiene un ciclo Hamiltoniano.



**FIGURA 5-13** Ciclo Hamiltoniano producido por búsqueda de primero en profundidad.



Ahora, la búsqueda de primero en profundidad se resume como sigue:

### Búsqueda de primero en profundidad

- Paso 1.** Formar una pila de un elemento que conste del nodo raíz.
- Paso 2.** Probar si el elemento superior en la pila es un nodo meta. Si es así, detener el proceso; en caso contrario, ir a 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 vacía, regresar Falla. En caso contrario, ir a paso 2.

### 5-3 MÉTODO DE ASCENSO DE COLINA (HILL CLIMBING)

Después de leer el apartado sobre búsqueda de primero en profundidad, quizás el lector se pregunte sobre un problema: de entre todos los descendientes, ¿qué nodo debe seleccionarse para expandirlo? En esta sección se presentará un método denominado ascenso de colina, que es una variante de la búsqueda de primero en profundidad en la cual se aplica algún método codicioso como ayuda para decidir la dirección en que hay que moverse en el espacio de búsqueda. En términos generales, el método codicioso usa alguna medida heurística para ordenar las opciones. Y, mientras mejor sea la heurística, mejor es el ascenso de colina.

Nuevamente se considerará el problema del rompecabezas de 8 piezas. Suponga que el método codicioso usa la siguiente función de evaluación simple  $f(n)$  para ordenar las opciones:

$$f(n) = w(n)$$

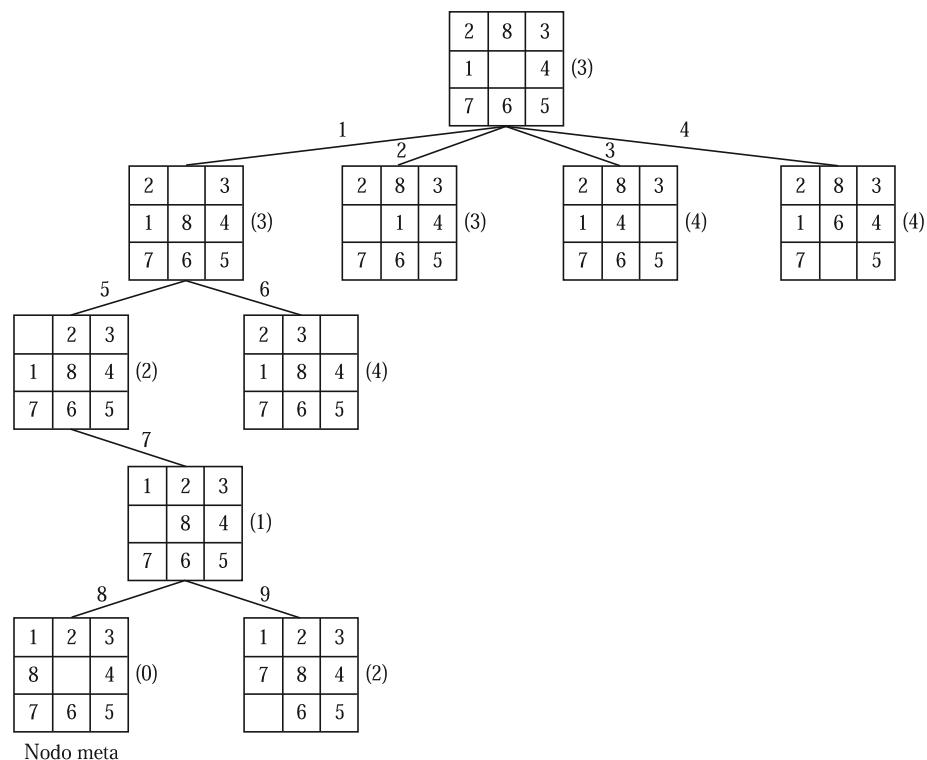
donde  $w(n)$  es el número de piezas móviles mal colocadas en el nodo  $n$ . Entonces, si el nodo inicial está colocado como se muestra en la figura 5-14, entonces  $f(n)$  es igual a 3 porque 1, 2 y 8 están mal colocados.

**FIGURA 5-14** Nodo inicial de un problema del rompecabezas de 8 piezas.

2	8	3
1		4
7	6	5

En la figura 5-15 se muestra el resultado de aplicar el método de ascenso de colina usando  $f$  como heurística para ordenar las opciones de entre todos los descendientes de un nodo. El valor de  $f$  para cada nodo se muestra en la figura 5-15. Sobre cada borde también se indica el orden en que se desarrollan los nodos. Observe que el método de ascenso de colina sigue usando la búsqueda de primero en profundidad, excepto que de entre todos los descendientes de un nodo, este método selecciona el nodo localmente óptimo para expandirlo.

**FIGURA 5-15** Problema del rompecabezas de 8 piezas resuelto con el método de ascenso de colina.



El lector no debe tener una impresión equivocada de que el método de ascenso de colina es extraordinariamente eficiente debido al ejemplo mostrado en la figura 5-15. En esta figura, de entre los primeros nodos expandidos, hay dos nodos que tienen el mismo valor que la función de evaluación. Si se hubieran expandido los otros nodos, para llegar a la solución se requeriría mucho más tiempo.

**Método del ascenso de colina**

- Paso 1.** Formar una pila consistente de un único elemento nodo raíz.
- Paso 2.** Probar si el primer elemento de la pila es un nodo meta. Si es así, detenerse; en caso contrario, ir a paso 3.
- Paso 3.** Quitar el primer elemento de la pila y expandir el elemento. Agregar los descendientes del elemento eliminado a la pila ordenada por la función de evaluación.
- Paso 4.** Si la lista está vacía, regresar falla. En caso contrario, ir a paso 2.

**5-4 ESTRATEGIA DE BÚSQUEDA DE PRIMERO EL MEJOR  
(BEST-FIRST SEARCH)**

Esta estrategia constituye una forma para combinar en un solo método las ventajas de la búsqueda del primero en profundidad y la búsqueda del primero en amplitud. En búsqueda de primero el mejor hay una función de evaluación y se selecciona el nodo con el menor costo de entre todos los nodos que se han expandido hasta el momento. Puede verse que el método de búsqueda de primero el mejor, a diferencia del método de ascenso de colina, posee una visión general.

**Método de búsqueda de primero el mejor**

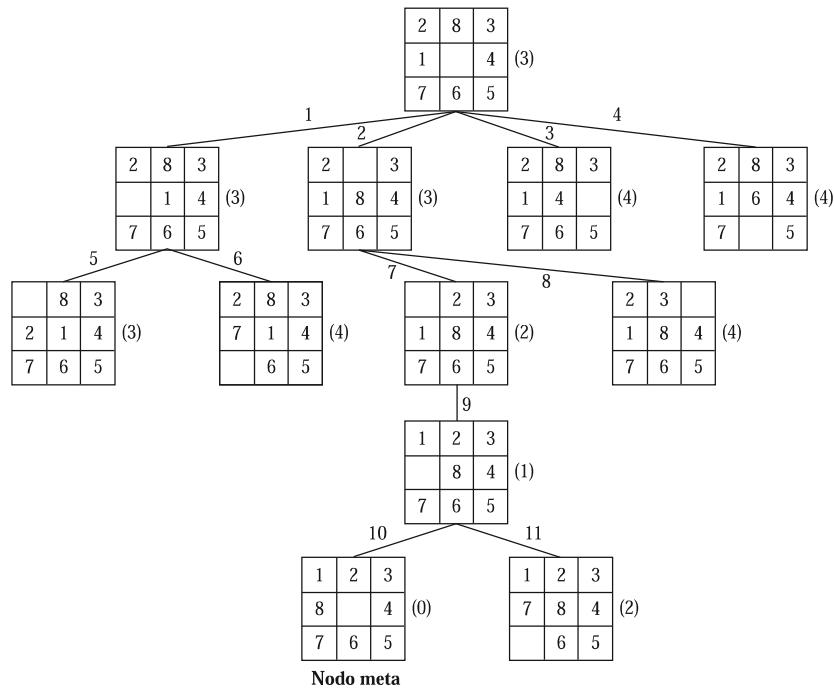
- Paso 1.** Construir un heap usando la función de evaluación. Primero, formar un heap de 1 elemento (el nodo raíz).
- Paso 2.** Probar si el elemento raíz en el heap es un nodo meta. Si es así, detenerse; en caso contrario, ir a paso 3.
- Paso 3.** Quitar el elemento raíz del heap y expandir el elemento. Agregar los descendientes (nodos hijos) del elemento eliminado al heap.
- Paso 4.** Si el heap está vacío, hay una falla. De otra forma, ir a paso 2.

Si se aplica la misma heurística que con el método de ascenso de colina para esta primera mejor búsqueda, entonces el problema del rompecabezas de 8 piezas se resuelve como se muestra en la figura 5-16.

**5-5 ESTRATEGIA DE RAMIFICAR-Y-ACOTAR  
(BRANCH-AND-BOUND)**

En las secciones previas se mostró que muchos problemas pueden resolverse aplicando las técnicas del árbol de búsqueda. Observe que ninguno de esos problemas es de optimización. Debe resultar interesante para el lector observar que ninguno de los métodos anteriores puede usarse para resolver un problema de optimización. En esta sección

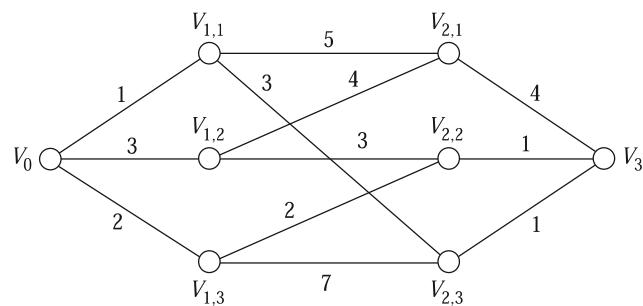
**FIGURA 5-16** Problema del rompecabezas de 8 piezas resuelto por el método de búsqueda de primero el mejor.



se presenta la estrategia de ramificar-y-acotar, que quizás es una de las más eficientes para resolver un problema combinatorio grande. Básicamente, sugiere que un problema puede tener soluciones factibles. No obstante, una vez que se descubre que muchas soluciones pueden no ser óptimas, es necesario tratar de acotar el espacio solución.

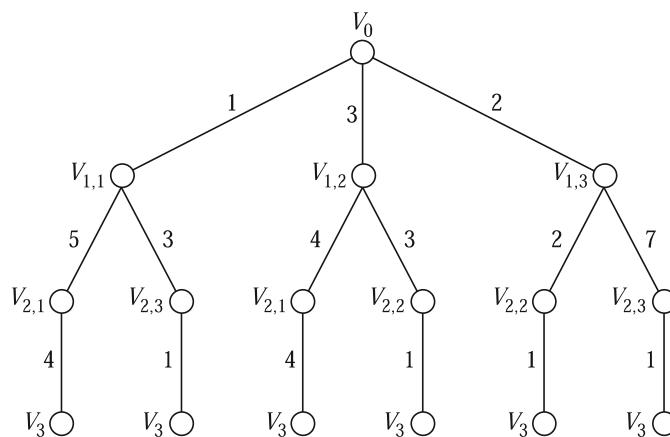
A continuación se explicarán los principios básicos de la estrategia de ramificar-y-acotar utilizando la figura 5-17.

**FIGURA 5-17** Problema multietapas de una gráfica de búsqueda.



En la figura 5-17, el problema consiste en encontrar una ruta más corta de  $V_0$  a  $V_3$ . Este problema puede resolverse de manera eficiente transformando primero el problema en un problema de árbol de búsqueda como se muestra en la figura 5-18.

**FIGURA 5-18** Representación de árbol de soluciones del problema de la figura 5-17.

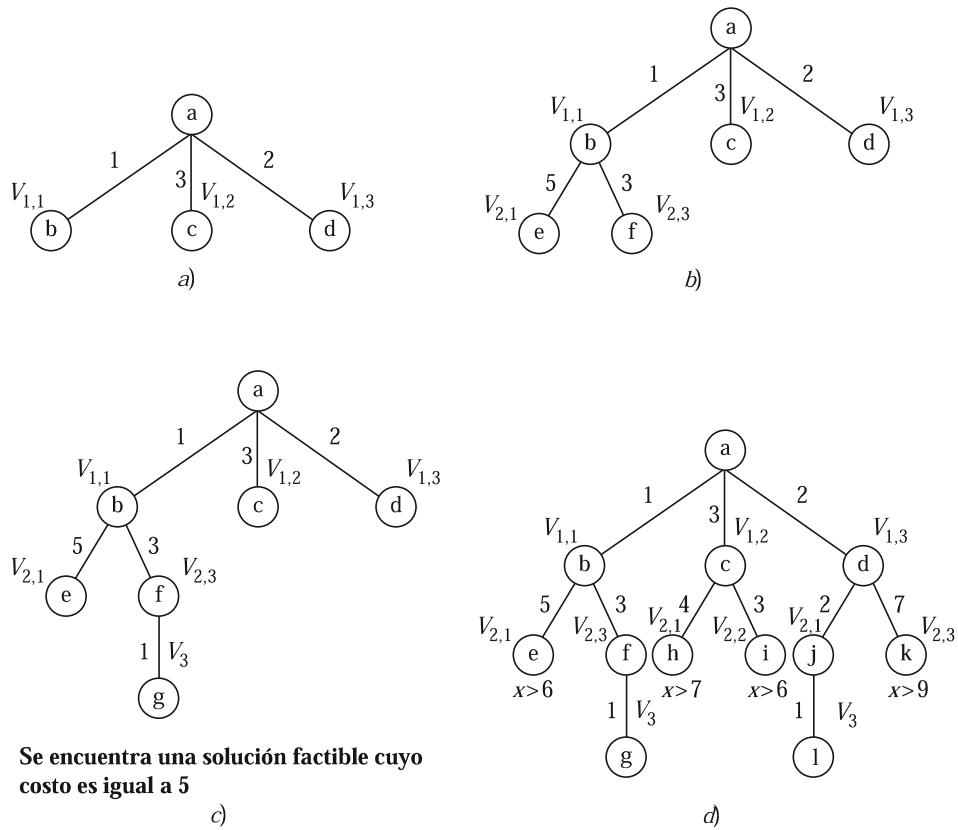


En la figura 5-18 se observan las seis soluciones factibles. ¿Cómo puede la estrategia de ramificar-y-acotar ser de utilidad para encontrar la ruta más corta sin necesidad de realizar una búsqueda exhaustiva? Considere la figura 5-19, que ilustra el proceso de usar algún tipo de método de ascenso de colina. En la figura 5-19a), desde la raíz del árbol de búsqueda se han desarrollado tres nodos. De entre estos tres nodos, se ha escogido uno para expandirlo. Puede haber muchas formas para seleccionar el siguiente nodo a expandir.

En nuestro caso, se considerará que se ha usado el método de ascenso de colina. Es decir, de entre los nodos que se han desarrollado más recientemente, el que se escoge para expandirlo a continuación siempre es el nodo asociado con el menor costo.

Usando este principio, se expandirá el nodo  $b$ . Sus dos descendientes se muestran en la figura 5-19b). Debido a que el nodo  $f$  corresponde a  $V_{2,3}$  y su costo asociado es el más pequeño, de entre los nodos  $e$  y  $f$ , el que se expandirá es el nodo  $f$ . Debido a que el nodo  $g$  es un nodo meta, debe encontrarse una solución factible cuyo costo sea igual a 5, como se muestra en la figura 5-19c).

El costo de esta solución factible, que es igual a 5, sirve como una cota superior de la solución óptima. Cualquier solución con costo superior a 5 no puede ser una solución óptima. En consecuencia, esta cota puede usarse para eliminar prematuramente

**FIGURA 5-19** Ilustración de la estrategia de ramificar-y-acotar.

muchas ramas. Por ejemplo, el nodo *e* jamás conducirá a una solución óptima porque cualquier solución con el nodo *e* tendrá un costo superior a 6.

Como se muestra en la figura 5-19, es posible evitar una búsqueda exhaustiva de todo el espacio solución. Por supuesto, es necesario indicar que hay otra solución que también es óptima.

El ejemplo anterior ilustra el principio básico de la estrategia de ramificar-y-acotar. Esta estrategia consta de dos mecanismos importantes: un mecanismo para generar ramificaciones y un mecanismo para generar una cota de modo que sea posible eliminar muchas ramificaciones. Aunque esta estrategia suele ser muy eficiente, en los peores casos, aún puede generarse un árbol muy grande. Por lo tanto, es necesario darse cuenta de que la estrategia de ramificar-y-acotar es eficiente en casos promedio.

### 5-6 UN PROBLEMA DE ASIGNACIÓN DE PERSONAL RESUELTO CON LA ESTRATEGIA DE RAMIFICAR-Y-ACOTAR

A continuación se mostrará cómo un problema de asignación de personal, que es NP completo, puede resolverse de manera eficiente aplicando la estrategia de ramificar-y-acotar. Considere un conjunto de personas  $P = \{P_1, P_2, \dots, P_n\}$  ordenado linealmente, donde  $P_1 < P_2 < \dots < P_n$ . Puede imaginarse que el ordenamiento de las personas es determinado por algún criterio como estatura, edad, antigüedad, etc. También se considera un conjunto de trabajos  $J = \{J_1, J_2, \dots, J_n\}$ , de los cuales se supone que están parcialmente ordenados. A cada persona puede asignarse un trabajo. Sean las personas  $P_i$  y  $P_j$ , a quienes se asignan los trabajos  $f(P_i)$  y  $f(P_j)$ , respectivamente. Se requiere que si  $f(P_i) \leq f(P_j)$ , entonces  $P_i \leq P_j$ . La función  $f$  puede interpretarse como una asignación factible que mapea personas en sus trabajos idóneos. También requiere que si  $i \neq j$ , entonces  $f(P_i) \neq f(P_j)$ .

Considere el siguiente ejemplo.  $P = \{P_1, P_2, P_3\}$ ,  $J = \{J_1, J_2, J_3\}$  y el ordenamiento parcial de  $J$  es  $J_1 \leq J_3$  y  $J_2 \leq J_3$ . En este caso,  $P_1 \rightarrow J_1$ ,  $P_2 \rightarrow J_2$  y  $P_3 \rightarrow J_3$  son combinaciones factibles, mientras  $P_1 \rightarrow J_1$ ,  $P_2 \rightarrow J_3$  y  $P_3 \rightarrow J_2$  no lo son.

Además, se supone que hay un costo  $C_{ij}$  en el que incurre una persona  $P_i$  a la que se asigna el trabajo  $J_j$ . Sea  $X_{ij}$  igual a 1 si a  $P_i$  se asigna  $J_j$  e igual a 0 en caso contrario. Entonces, el costo total correspondiente a una asignación factible  $f$  es

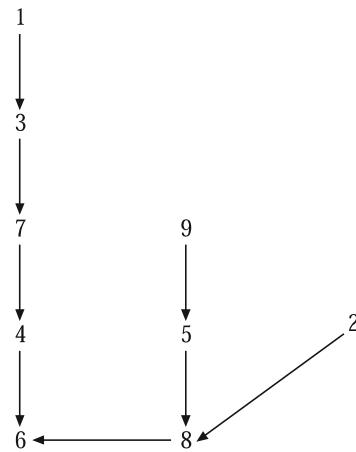
$$\sum_{i,j} C_{ij} X_{ij}$$

El problema de asignación de personal se define con precisión como sigue: se tiene un conjunto de personas  $P = \{P_1, P_2, \dots, P_n\}$  ordenado linealmente, donde  $P_1 < P_2 < \dots < P_n$  y un conjunto de trabajos  $J = \{J_1, J_2, \dots, J_n\}$  ordenado parcialmente. El costo  $C_{ij}$  es igual al costo de asignar  $P_i$  a  $J_j$ . Cada persona se asigna a un trabajo a ningún par de personas se asigna el mismo trabajo. El problema consiste en encontrar una asignación factible óptima que minimice la siguiente cantidad

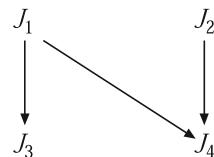
$$\sum_{i,j} C_{ij} X_{ij}.$$

Así, este problema es de optimización, del cual puede mostrarse que es NP-difícil. Aquí no se abordará la dificultad NP.

Para resolver el problema se usará el concepto de “ordenamiento topológico”. Para un ordenamiento parcial dado de un conjunto  $S$ , una secuencia lineal  $S_1, S_2, \dots, S_n$  está ordenada topológicamente con respecto a  $S$  si  $S_i \leq S_j$  en el ordenamiento parcial implica que  $S_i$  está ubicado antes que  $S_j$  en la secuencia. Por ejemplo, para el ordenamiento parcial que se muestra en la figura 5-20, una secuencia ordenada topológicamente correspondiente es 1, 3, 7, 4, 9, 2, 5, 8, 6.

**FIGURA 5-20** Ordenamiento parcial.

Sea  $P_1 \rightarrow J_{k_1}$ ,  $P_2 \rightarrow J_{k_2}$ , ...,  $P_n \rightarrow J_{k_n}$  una asignación factible. Según la definición de nuestro problema, los trabajos están ordenados parcialmente y las personas están ordenadas linealmente. En consecuencia,  $J_{k_1}, J_{k_2}, \dots, J_{k_n}$  debe ser una secuencia ordenada topológicamente con respecto al ordenamiento parcial de los trabajos. Esta idea se ilustrará con un ejemplo. Considere  $J = \{J_1, J_2, J_3, J_4\}$  y  $P = \{P_1, P_2, P_3, P_4\}$ . El ordenamiento parcial de  $J$  se muestra en la figura 5-21.

**FIGURA 5-21** Un ordenamiento parcial de los trabajos.

Las siguientes secuencias están ordenadas topológicamente:

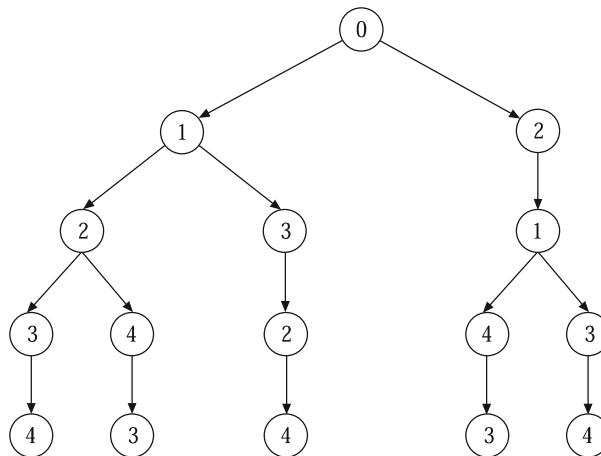
- $J_1, J_2, J_3, J_4$
- $J_1, J_2, J_4, J_3$
- $J_1, J_3, J_2, J_4$
- $J_2, J_1, J_3, J_4$
- $J_2, J_1, J_4, J_3$ .

Cada secuencia representa una asignación factible. Por ejemplo, la primera secuencia corresponde a la asignación factible

$$P_1 \rightarrow J_1, P_2 \rightarrow J_2, P_3 \rightarrow J_3, P_4 \rightarrow J_4.$$

Para encontrar todas las secuencias ordenadas topológicamente pueden usarse fácilmente tres técnicas de búsqueda. Por ejemplo, para el ordenamiento parcial que se muestra en la figura 5-21, en la figura 5-22 se observa un árbol que muestra todas las secuencias ordenadas topológicamente.

**FIGURA 5-22** Representación de árbol de todas las secuencias ordenadas topológicamente correspondientes a la figura 5-21.



El árbol en la figura 5-22 se genera usando tres pasos básicos.

1. Tomar un elemento que no esté precedido por ningún otro elemento en el ordenamiento parcial.
2. Seleccionar este elemento como un elemento en una secuencia ordenada topológicamente.
3. Eliminar del conjunto de ordenamiento parcial este elemento que acaba de seleccionarse. El conjunto resultante sigue estando parcialmente ordenado.

Por ejemplo, para el ordenamiento parcial que se muestra en la figura 5-21, en el principio,  $J_1$  y  $J_2$  son los elementos sin predecesores. Así, están en el mismo nivel del árbol. Considere el nodo correspondiente a 1. Si se elimina 1, el conjunto parcialmente ordenado ahora contiene a 2, 3 y 4. Sólo 2 y 3 carecen de predecesores en este nuevo conjunto. En consecuencia, se generan 2 y 3.

Una vez explicado cómo el espacio solución de nuestro problema se describe por medio de un árbol, procederemos a la demostración de cómo aplicar la estrategia de ramificar-y-acotar para encontrar una solución óptima.

Dada una matriz de costos, de inmediato es posible calcular una cota inferior de nuestras soluciones. Esta cota inferior se obtiene al reducir la matriz de costos de forma que no se afecten las soluciones y de que en cada renglón y en cada columna haya por lo menos un cero y que todos los elementos restantes de la matriz de costos sean no negativos.

Observe que si de cualquier renglón o cualquier columna de la matriz de costos se resta una constante, una solución óptima permanece sin cambio. Considere la tabla 5-1, donde se muestra una matriz de costos. Para este conjunto de datos, es posible restar 12, 26, 3 y 10 de los renglones 1, 2, 3 y 4, respectivamente. Después de lo anterior es posible restar 3 de la columna 2. La matriz resultante es una matriz de costos reducida donde todo renglón y toda columna contienen por lo menos un cero y todos los elementos restantes de la matriz son no negativos, como se muestra en la tabla 5-2. El costo total restado es  $12 + 26 + 3 + 10 + 3 = 54$ . Ésta es la cota inferior de nuestras soluciones.

**TABLA 5-1** Una matriz de costos para un problema de asignación de personal.

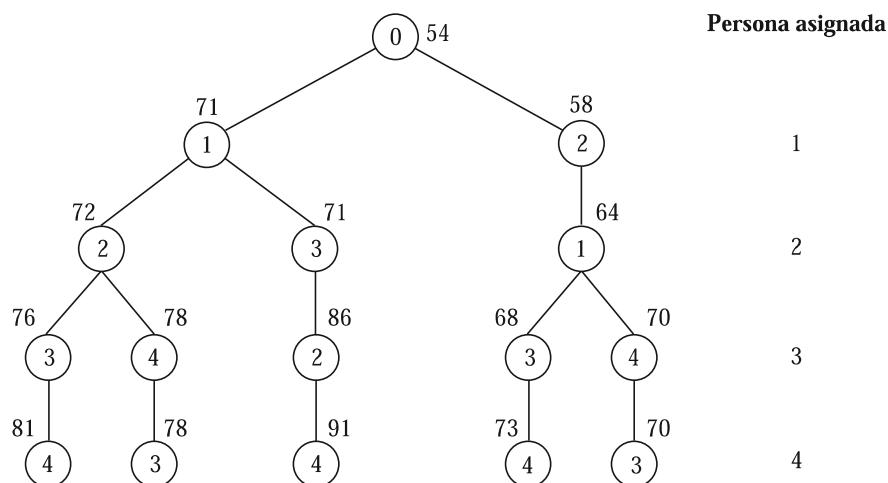
Personas \ Trabajos	1	2	3	4
1	29	19	17	12
2	32	30	26	28
3	3	21	7	9
4	18	13	10	15

**TABLA 5-2** Una matriz de costos reducida.

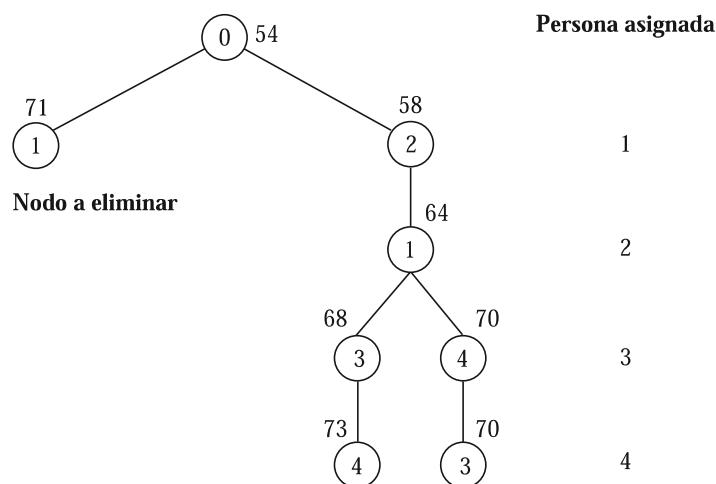
Personas \ Trabajos	1	2	3	4	
1	17	4	5	0	
2	6	1	0	2	Total = 54
3	0	15	4	6	
4	8	0	0	5	

En la figura 5-23 se muestra un árbol de enumeración asociado con esta matriz de costos reducida. Si se usa la mínima cota inferior, las subsoluciones que no pueden conducir a soluciones óptimas se podan en una etapa mucho más temprana, lo cual se muestra en la figura 5-24.

**FIGURA 5-23** Árbol de enumeración asociado con la matriz de costos reducida en la tabla 5-2.



**FIGURA 5-24** Acotamiento de las subsoluciones.



En la figura 5-24 puede verse que después de que se encuentra una solución con costo 70, de inmediato es posible acotar todas las soluciones empezando con la asignación de  $P_1$  a  $J_1$  porque su costo es 71, que es mayor que 70.

¿Por qué se restaron costos de la matriz de costos? Suponga que no se hiciera esto. Luego, considere el nodo correspondiente a asignar  $P_1 \rightarrow J_1$ . El costo asociado con este nodo es sólo 29. Imagine que se ha encontrado una solución factible con costo 70; a saber, al asignar  $P_1 \rightarrow J_2$ ,  $P_2 \rightarrow J_1$ ,  $P_3 \rightarrow J_4$  y  $P_4 \rightarrow J_3$ . Aunque se ha encontrado una cota superior, ésta no puede usarse para acotar al nodo correspondiente a  $P_1 \rightarrow J_1$  porque su costo es sólo 29, menor que 70.

Observe nuevamente la figura 5-24. Ahora puede verse que el costo asociado con asignar  $P_1$  a  $J_1$  es 71, en vez de 29. Así, aparece una cota. ¿Por qué es posible tener un costo tan elevado? Esto se debe a que se han restado costos de la matriz de costos original de modo que cada renglón y cada columna contienen un cero. Así, después de restar, se tiene una mejor cota inferior para todas las soluciones; a saber, 54. En otras palabras, ninguna solución puede tener un costo menor que 54. Con esta información, se sabe que la cota inferior de asignar  $P_1$  a  $J_1$  es  $54 + 17 = 71$ , en vez de sólo 29. Por supuesto, una cota inferior más alta conduce a una eliminación más temprana.

### 5-7 / EL PROBLEMA DE OPTIMIZACIÓN DEL AGENTE VIAJERO

#### RESUELTO CON LA ESTRATEGIA DE RAMIFICAR-Y-ACOTAR

El problema de decisión del agente viajero es un problema NP-completo. Así, este problema es difícil de resolver en los peores casos. Sin embargo, como se verá en esta sección, el problema del agente viajero puede resolverse aplicando la estrategia de ramificar-y-acotar. Esto es, si se tiene suerte, es posible evitar una búsqueda exhaustiva a través del espacio solución.

El principio fundamental al aplicar la estrategia de ramificar-y-acotar para resolver el problema de optimización del agente viajero consta de dos partes.

1. Hay una forma de dividir el espacio solución.
2. Hay una forma de pronosticar una cota inferior para una clase de soluciones. También hay una forma de encontrar una cota superior de una solución óptima. Si la cota inferior de una solución excede a esta cota superior, entonces esta solución no puede ser óptima. Así, es necesario eliminar la ramificación asociada con esta solución.

El problema del agente viajero se define sobre una gráfica, o en puntos planos. Si el problema del agente viajero se define sobre un conjunto de puntos planos, es posible usar muchos trucos para que el algoritmo sea todavía más eficiente. En esta sección se

supondrá que el problema se define sobre una gráfica. Para simplificar el análisis, en nuestro ejemplo se supone que entre un vértice y él mismo no hay ningún arco y que entre cada par de vértices hay un arco asociado con un costo no negativo. El problema del agente viajero consiste en encontrar un recorrido, empezando desde cualquier vértice, que pase por todos los demás vértices y regrese al vértice inicial, con costo mínimo.

Considere la matriz de costos en la tabla 5-3.

**TABLA 5-3** Matriz de costos para un problema del agente viajero.

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

La estrategia de ramificar-y-acotar divide la 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”.

Antes que todo se observa, como se analizó en el apartado previo, que si una constante se resta de cualquier renglón o cualquier columna de la matriz de costos, una solución óptima no cambia. Para el ejemplo de la tabla 5-3, si el costo mínimo se resta de cada renglón de la matriz de costos, la cantidad total que se resta es una cota inferior para la solución del problema del agente viajero. Así, es posible restar 3, 4, 16, 7, 25, 3 y 26 de los renglones 1 a 7, respectivamente. El costo total restado es  $3 + 4 + 16 + 7 + 25 + 3 + 26 = 84$ . De esta forma es posible obtener una matriz reducida, que se muestra en la tabla 5-4.

En la matriz que se muestra en la tabla 5-4, cada renglón contiene un cero. Sin embargo, algunas columnas, a saber, las columnas 3, 4 y 7, aún no contienen ningún cero. Entonces, además se resta 7, 1 y 4 de las columnas 3, 4 y 7, respectivamente. (El costo restado es  $7 + 4 + 1 = 12$ .) La matriz reducida resultante se muestra en la tabla 5-5.

**TABLA 5-4** Una matriz de costos reducida.

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

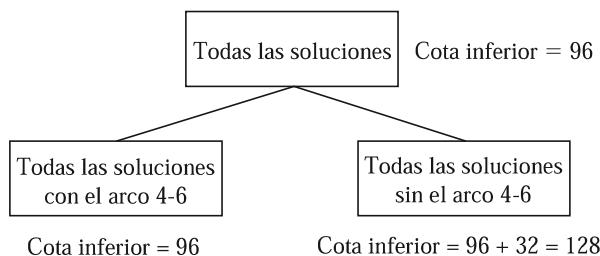
**TABLA 5-5** Otra matriz de costos reducida.

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

Debido a que se restó un costo total de  $84 + 12 = 96$ , se sabe que una cota inferior de este problema del agente viajero es 96.

A continuación se considerará el siguiente problema: suponga que se sabe que un recorrido incluye al arco 4-6, cuyo costo es cero. ¿Cuál es la cota inferior del costo de este recorrido? Es muy fácil encontrar la respuesta: la cota inferior sigue siendo 96.

Suponga que se sabe que el recorrido excluye al arco 4-6. ¿Cuál es la nueva cota inferior? Observe la tabla 5-5. Si un recorrido no incluye al arco 4-6, entonces debe incluir algún otro arco que salga de 4. El arco con el menor 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 5-6, cuyo costo es 0. Así, la nueva cota inferior es  $96 + (32 + 0) = 128$ . En consecuencia, se tiene el árbol binario que se muestra en la figura 5-25.

**FIGURA 5-25** Nivel más elevado de un árbol decisión.

¿Por qué se escogió el arco 4-6 para dividir la solución? Esto se debe al hecho de que el arco 4-6 provoca el mayor incremento de la cota inferior. Suponga que para esta división se usa el arco 3-5. En este caso, la cota inferior sólo puede incrementarse por  $1 + 17 = 18$ .

A continuación se considerará el subárbol izquierdo. En éste se incluye el arco 4-6. Así, es necesario eliminar el cuarto renglón y la sexta columna de la matriz de costos. Además, como se usa el arco 4-6, no es posible usar el arco 6-4. Es necesario igualar  $c_{6-4}$  a infinito. La matriz resultante se muestra en la tabla 5-6.

**TABLA 5-6** Una matriz de costos reducida si se incluye el arco 4-6.

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

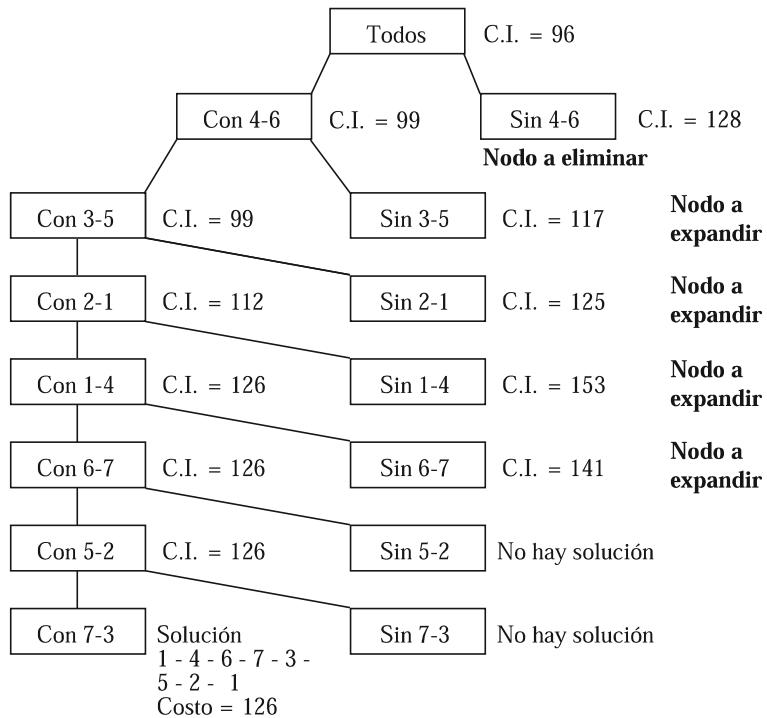
De nuevo, se observa que el renglón 5 aún no contiene ningún cero. Así, es posible restar 3 del renglón 5. La matriz de costos reducida para el subárbol izquierdo se muestra en la tabla 5-7. También es necesario sumar 3 a la cota inferior del subárbol izquierdo (soluciones con el arco 4-6).

En cuanto a la matriz de costos del subárbol derecho, soluciones sin el arco 4-6, basta igualar  $c_{4-6}$  a  $\infty$ . El proceso de separación continuaría y produciría el árbol que se muestra en la figura 5-26. En este proceso, si se sigue la ruta de menor costo, se obtiene

**TABLA 5-7** Una matriz de costos reducida para la tabla 5-6.

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

**FIGURA 5-26** Una solución de ramificar-y-acotar de un problema del agente viajero.



una solución factible con costo 126. Este costo 126 sirve como cota superior y permite eliminar muchas ramificaciones porque sus cotas inferiores exceden a esta cota.

Aquí es necesario mencionar otro punto. Éste puede explicarse al considerar la matriz de costos reducida de todas las soluciones con los arcos 4-6, 3-5 y 2-1 incluidos, como se muestra en la tabla 5-8.

**TABLA 5-8** Una matriz de costos reducida.

<i>i</i>	<i>j</i>	2	3	4	7
1		$\infty$	74	0	41
5		14	$\infty$	0	21
6		85	8	$\infty$	0
7		0	0	0	$\infty$

El arco 1-4 puede usarse para dividir y el subárbol resultante es como se muestra en la tabla 5-9.

**TABLA 5-9** Una matriz de costos reducida.

<i>i</i>	<i>j</i>	2	3	7
5		14	$\infty$	21
6		85	8	0
7		0	0	$\infty$

Observe que ya se ha decidido que los arcos 4-6 y 2-1 están incluidos en la solución. Ahora es necesario agregar el arco 1-4. Resulta evidente que es necesario impedir el uso del arco 6-2. Si se usa este arco, se forma un ciclo que está prohibido. Así, es necesario igualar  $c_{6-2}$  a  $\infty$ . En consecuencia, se tendrá la matriz de costos para el subárbol izquierdo como se muestra en la tabla 5-10.

En general, si las rutas  $i_1-i_2-\dots-i_m$  y  $j_1-j_2-\dots-j_n$  ya se han incluido y debe sumarse una ruta de  $i_m$  a  $j_1$ , entonces es necesario evitar la ruta de  $j_n$  a  $i_1$ .