PARADIGMAS 3 (1) 45-52 (2011)
Revista del Departamento Académico de Ingeniería.
Universidad Ricardo Palma. ISSN: 2220-2056

Solución de problemas de asignación de personal mediante la estrategia de ramificar y acotar

Troubleshooting staffing by branch and refine strategy

Augusto Cortez Vasquez[1]

Resumen

La solución de muchos problemas puede representarse mediante arboles, por lo que resolver tales problemas se convierte en problema de arboles. Existen muchas estrategias basadas en técnicas de arboles de búsqueda. El presente articulo presenta la estrategia de ramificar-y-acotar que es considerada una de las mas eficientes. Esta técnica sugiere que ante un problema pueden existir soluciones factibles, sin embargo muchas de estas soluciones suelen ser no optimas, por tanto es necesario acotar el espacio de solución. Se presenta como caso práctico el problema de asignación de personal resuelto con esta estrategia.

Palabras claves

Arboles binarios, ramificar–y-acotar, asignación de personal

Abstract

The solution of many problems can be represented by trees, so solving these problems becomes an issue of trees. There are many strategies based tree search techniques. This article presents a strategy to branch-and-narrow is considered one of the most efficient. This technique suggests that there may be a problem with feasible solutions, however many of these solutions are often not optimal, it is therefore necessary to limit the solution space. Practical case is presented as the staffing problem solved with this strategy.

Key words

Binary tree, branch-and-narrow, staffing

1 Licenciado en Computación. Magíster en Computación e Informática. Docente U.R.P.

Introducción

La técnica de **Ramificación y poda** consiste en construir un árbol de soluciones, en donde cada rama nos lleva a una posible solución posterior a la actual. La característica principal de esta técnica con respecto a otras anteriores es que el algoritmo se encarga de detectar en qué ramificación las soluciones dadas ya no están siendo óptimas, para «podar» esa rama del árbol y no continuar malgastando recursos y procesos.
El problema de asignación de personal es un problema clásico que puede resolverse de muchas formas, una de ellas es basada en arboles binarios, especialmente con la estrategia de ramificar-y-acotar con el fin de acotar el espacio de soluciones. El problema tratado se clasifica como NP-completo[1], sin embargo no se analizara su complejidad. Se utilizara el concepto de ordenamiento topológico de grafos.[5,6].

Contenido

Conjunto parcialmente ordenado

Sea $A=\{x_1, x_2, \ldots x_n\}$ y la relación R sobre A definida xRy
Si R es un orden parcial, entonces R define un conjunto parcialmente ordenado denotado (A,R)
Como ejemplo consideremos A como el conjunto de cursos de la Escuela de Ingeniería de Sistemas. Definimos R sobre A xRy si x,y son el mismo curso o si x es un prerrequisito de y. Entonces R hace de A un conjunto parcialmente ordenado.[2,3]

Ordenamiento topológico[2]

Dado un grafo G(V,A)

donde: V: Número de vértices o nodos
A: Número de aristas o arcos

Consideremos que para ensamblar un nuevo producto hay que realizar n tareas de acuerdo al siguiente diagrama:

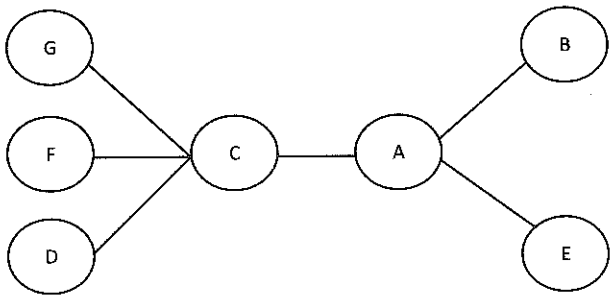

Figura 1. Ordenamiento parcial

las tareas deben ejecutarse en el orden parcial dado. Puesto que el conjunto de instrucciones debe ser una lista de tareas numeradas, esta lista debe asegurar conservar el orden parcial del diagrama, y para ello utilizaremos la técnica de ordenamiento topológico.
La lista sería : (1-E), (2-B), (3-A), (4-C), (5-G) , (6-F) , (7-D)

Los algoritmos usuales para ordenamiento topológico tienen un tiempo de ejecución en función de la cantidad de nodos mas la cantidad de aristas $O(|V|+|A|)$ [3,4]

Árbol de decisión

Un árbol de decisión es una abstracción con que se demuestran cotas inferiores. En nuestro contexto un árbol de decisión es un árbol binario. Cada nodo representa un conjunto de ordenamientos posibles, consistente en las operaciones realizadas, entre los elementos. Los resultados de las comparaciones son las aristas del árbol [7,8].
Un árbol de decisión es una estructura que permite modelar funciones discretas, con el objetivo de determinar el valor combinado de un conjunto de variables, a partir del valor de cada una de ellas, y poder decidir que acción tomar.

CASO PRACTICO: Problema de asignación de personal

Consideremos el problema de asignación de recursos.

Sea $P=\{p_1,p_2 \ldots p_n\}$ donde $p_i < p_{i+1}$ para todo i de 1 a n-1. Conjunto de personas

$T=\{T_1,T_2 \ldots T_n\}$ donde $t_i < t_{i+1}$ para todo i de 1 a n-1. Conjunto de trabajos

Sea $\acute{O}$: $P \longrightarrow T$ $\quad$ $\acute{O}(p_i)$ corresponde al trabajo asignado a la persona p_i

Precondiciones

a) A cada persona se le puede asignar un trabajo
b) Si $p_i \neq p_j$ entonces $\acute{O}(p_i) \neq \acute{O}(p_j)$
c) Si $\acute{O}(p_i) \leq \acute{O}(p_j)$ entonces $p_i \leq p_j$
d) El conjunto T está parcialmente ordenado

Definimos

C_{ij} el costo en el que incurre una persona p_i a la que se asigna el trabajo T_j.

$$\text{Sea } X_{ij} = \begin{cases} 1 & P_i \text{ se asigna } J_j \\ 0 & \text{otro caso} \end{cases}$$

Por tanto el costo total correspondiente a una asignación factible $\acute{O}$ es $\sum_{i\,j} C_{ij}X_{ij}$

Lo que se pretende es asignar tareas a los personas de tal forma que se minimice el costo total de ejecutar las n tareas. Utilizaremos el concepto de ordenamiento topológico.

Consideremos el siguiente cuadro de personas y trabajos

Personas-trabajo	1	2	3	4
T_1	25	17	16	12
T_2	14	6	8	15
T_3	4	12	7	5
T_4	9	5	16	6

Los trabajos deben estar ordenados topológicamente respecto del ordenamiento parcial de los trabajos

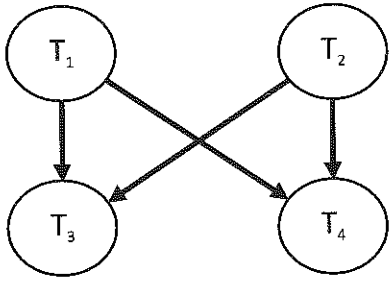

Figura 2. Ordenamiento parcial

La matriz de costos puede reducirse de forma que no afecte las soluciones y de que en cada renglón y en cada columna haya por lo menos un cero y de que todos los elementos restantes de la matriz sean no negativos.

Restamos 12, 6, 4,5 de las filas 1, 2 ,3 y 4 respectivamente

Personas-trabajo	1	2	3	4
1	13	5	4	0
2	8	0	2	9
3	0	8	3	1
4	4	0	11	1

Restamos 2 a la columna 3

Personas-trabajo	1	2	3	4
1	13	5	2	0
2	8	0	0	9
3	0	8	1	1
4	4	0	9	1

Se observa que toda columna y toda fila tiene por lo me nos un cero. El costo total restado es 12+6+4+5+2=29 que constituye la cota inferior de nuestras soluciones.

El espacio solución puede describirse por medio de un árbol

En el nivel 0 el nodo raíz se etiqueta con la cota 29. Los nodos del nivel 1 acumulan el valor del trabajo asociado al nodo.

Para construir el árbol se parte de lo siguiente:

A partir del ordenamiento parcial del grafo de la fig. 2 se nota que T_1 y T_2 no tienen predecesores, por tanto están en el mismo nivel del árbol. Consideramos el nodo T_1 si eliminamos T_1, el conjunto ordenado ahora contiene lo siguiente

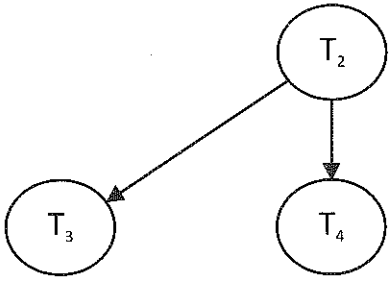

Figura 3.

Solo T_2 no tienen predecesores, por tanto si se elimina T_2 el conjunto ordenado ahora contiene lo siguiente

Figura 4.

T_3 y T_4 no tienen predecesores, por tanto están en el mismo nivel del árbol.
El árbol obtenido queda de la siguiente forma:

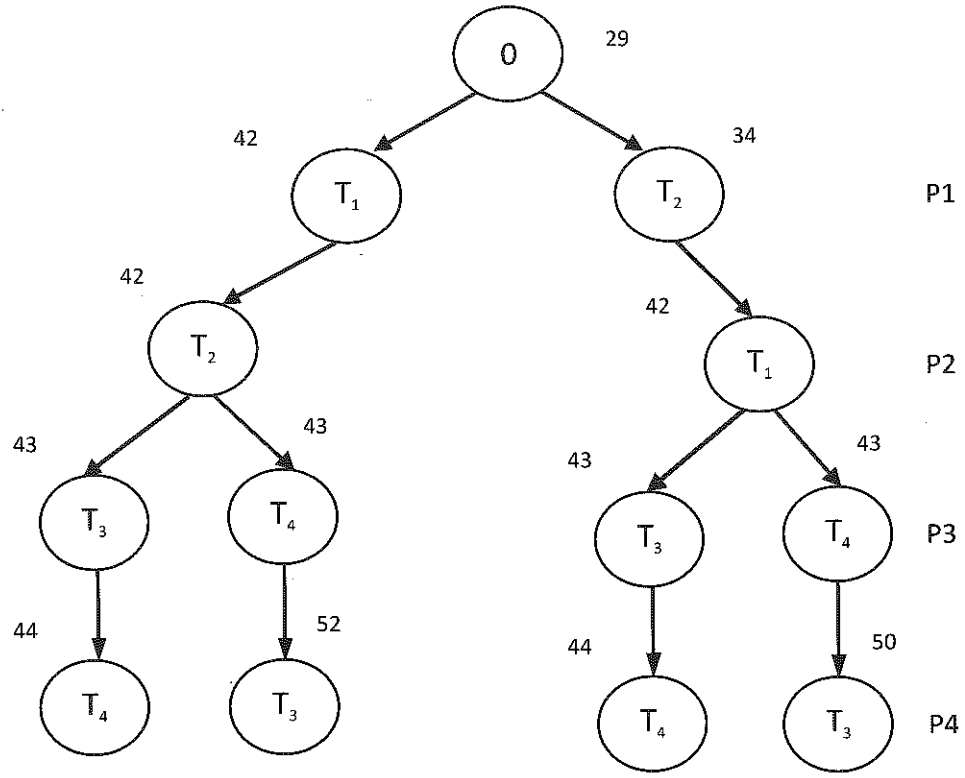

Figura 4.

Se ve en el árbol que los caminos
T_1, T_2, T_3, T_4, y T_2, T_1, T_3, T_4

Corresponde a la mejor asignación con costo 44

Durante el proceso de poda se eliminan las soluciones que no pueden conducir a soluciones óptimas respecto a la cota inferior. La exploración y poda es una técnica para explorar un grafo dirigido. Este grafo dirigido es acíclico[3] e incluso un árbol. En cada nodo se calcula una cota del posible valor de aquellas soluciones que pudieran encontrarse mas adelante. Si la cota muestra que cualquiera de estas soluciones tiene que ser necesariamente peor que la mejor solución hallada hasta el momento, entonces no necesitamos seguir explorando esta parte del grafo.[1,4,12]

Discusión

El problema de la asignación tiene muchas aplicaciones. Por ejemplo en lugar de hablar de personas y tareas, podemos formular el problema en términos de edificios y solares, en este caso Cij es el costo

de de construir el edificio i en el solar j y deseamos minimizar el costo total de los edificios. La necesidad de mantener una lista de nodos que han sido generados pero que no han sido explorados en su totalidad, situados en diferentes niveles del árbol y preferiblemente ordenados por orden de las cotas correspondientes, hace que la ramificación y poda resulten difíciles de programar. Es conveniente utilizar una representación de árbol como montículo. Aunque el programador no dispone de una formulación recursiva elegante de ramificación y poda como ocurre en un recorrido en profundidad y técnicas relacionadas, la técnica es tan potente que suele emplearse en aplicaciones prácticas.

CONCLUSIONES

La estrategia de ramificar-y-acotar consta de dos mecanismos importantes: 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, se corre el riesgo de generarse un árbol muy grande. Por tanto, es necesario darse cuenta de que la estrategia de ramificar y acotar es eficiente en el caso promedio. Resulta casi imposible precisar lo bien que se va a comportar esta técnica en un problema dado, empleando una cota dada. Siempre tenemos que llegar a un compromiso en lo concerniente a la calidad de la cota calculada. Con una cota mejor examinamos menos nodos, y si tenemos suerte obtendremos una solución óptima con rapidez. Por otra parte, lo más probable es que pasemos en cada nodo más tiempo calculando la cota correspondiente. En el caso peor, puede ocurrir incluso que una cota excelente no nos permita cortar ninguna rama del árbol, así que se desperdiciara todo el trabajo adicional efectuado en cada nodo. Se recomienda por tanto que para aplicaciones de tamaño considerable es rentable invertir el tiempo necesario para calcular la mejor cota posible.

REFERENCIAS BIBLIOGRÁFICAS

[1] AHO ALFRED. (1998) Estructuras de datos y algoritmos
Edit Adisson Wesley México ISBN 968-444-345-5

[2] BRASSARD G.. (1998) Fundamentos de Algoritmia
Edit Prentice Hall Madrid ISBN 0-13-335068-1

[3] CORTEZ AUGUSTO (2009), Algoritmica Edit ESVEGA ISBN 978-612-000257-5 Lima - Perú

[4] DROZDK, ADAM. (2007), Estructura de datos y algoritmos en JAVA.
Edit Thomspon México 2007 ISBN 0-534-49252-5

[5] GRASSMANN W.. (1997) Matemática Discreta y Lógica
Edit Prentice Hall Madrid ISBN 0-13-501206-6

[6] GRIMALDI R. (1997) Matemática Discreta y Combinatoria
Edit Adisson Wesley México ISBN 0-201-65376-1

[7] JOHNSONBAUGH R. (1999) Matemáticas Discretas
Edit Prentice Hall México ISBN 0-13-518242-5

[8] LEE R.C. (2007) Introducción al diseño y análisis de algoritmos
Edit McGraw-Hill Interamericana México ISBN 978-970-10-6124-4

[9] WEISS MARK (2000) Estructura de datos en Java
Edit Adisson Wesley España ISBN 0-201-54991-3

[10] WEISS MARK (1995), Estructura de datos y algoritmos. ISBN 0-201-62571-7 Edit Adisson Wesley 1995 Wilmington E.U.A.

[11] http://es.scribd.com/doc/41336427/OPT-08-4-Tecnicas-de-ramificacion-y-poda consultado junio 2011

[12] Thomas H. Cormen (2001), Introduction to algoritmic
Second Edition Mc Graw Hill ISBN 0-262-03293-7.

NOTAS DEL AUTOR

1. **NP-completo:** En teoría de la complejidad computacional, la clase de complejidad NP-completo es el subconjunto de los problemas de decisión en NP tal que todo problema en NP se puede reducir en cada uno de los problemas de NP-completo. Se puede decir que los problemas de NP-completo son los problemas más difíciles de NP y muy probablemente no formen parte de la clase de complejidad P. La clase P contiene aquellos problemas de decisión que pueden ser resueltos en tiempo polinómico por una MT determinista, esto es, aquellas en las que para cada par estado y símbolo exista a lo sumo una posibilidad de ejecución. Los problemas de complejidad polinómica son tratables, es decir, en la práctica se pueden resolver en un tiempo razonable. La clase NP contiene los problemas de decisión que son resueltos por una maquina de turing no determinista en tiempo polinómico y de ahí su nombre: Non-Deterministic Polynomial-time.

2. **Ordenamiento topológico:** se dice que un ordenamiento topológico de los nodos de un grafo G es una secuencia de nodos (W) que cumplen la siguiente condición: Si existe un camino desde el nodo i hacia el nodo j en G, entonces i debe aparecer antes que j en W.

3. **Grafo dirigido a cíclico:** también conocido como DAG por sus siglas en ingles, un grafo G se dice a cíclico si no existe un camino de longitud mayor a 0 desde el nodo u hacia si mismo para todo nodo u. Los DAG aparecen en modelos donde no tiene sentido que un vértice tenga un camino directo a él mismo.