

01168

2  
24

“ TECNICAS DE RAMIFICACION

Y

ACOTAMIENTO.

ANALISIS METODOLOGIA

Y

APLICACIONES ”

ALUMNA:

IDALIA FLORES DE LA MOTA.

TESIS CON  
FALLA DE ORIGEN

1990



Universidad Nacional  
Autónoma de México



## **UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso**

### **DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

## INDICE

	PAGINA
INTRODUCCION	
CAPITULO I.	
LA TECNICA DE RAMIFICACION Y ACOTAMIENTO ENFOQUE CLASICO	
1.1 DESCRIPCION CONCEPTUAL DE RAMIFICACION Y ACOTAMIENTO.	1
1.2 EL ARBOL DE BUSQUEDA	3
1.3 ESTRATEGIAS OPERATIVAS EN EL ALGORITMO BASICO	4
1.4 EJEMPLOS ILUSTRATIVOS	10
1.5 REFERENCIAS AL CPITULO I	16
CAPITULO II.	
PROGRAMACION ENTERA MIXTA:ENFOQUE CLASICO	
2.1 EL ALGORITMO DE LAND Y DOIG	17
2.2 EJEMPLO ILUSTRATIVO	19
2.3 EL ALGORITMO DE DAKIN	23
2.4 EJEMPLO NUMERICO	25
2.5 COMPARACION CON EL METODO DE LAND Y DOIG	27
2.6 ALGORITMO DE DRIEBEEK	30
2.7 REFERENCIAS AL CAPITULO II.	38
CAPITULO III.	
ANALISIS DE SENSIBILIDAD	
3.1 CONSIDERACIONES BASICAS DE PROGRAMACION LINEAL Y ANALISIS DE SENSIBILIDAD	39

	<b>PAGINA</b>
<b>3.2 RESULTADOS PARA RAMIFICACION Y ACOTAMIENTO</b>	<b>42</b>
<b>3.3 EJEMPLOS ILUSTRATIVOS</b>	<b>44</b>
<b>3.4 ANALISIS DE SENSIBILIDAD PARA EL ALGORITMO DE DRIEBEEK.</b>	<b>53</b>
<b>3.5 REFERENCIAS AL CAPITULO III.</b>	<b>55</b>
 <b>CAPITULO IV.</b>	
<b>PROGRAMACION NO LINEAL</b>	
<b>4.1 EL PROBLEMA NO LINEAL Y SUS CARACTERISITICAS</b>	<b>57</b>
<b>4.2 CONDICIONES DE KUHN-TUCKER</b>	<b>62</b>
<b>4.3 ESTRATEGIAS DE RAMIFICACION Y ACOTAMIENTO PARA PROBLEMAS DE PROGRAMACION ENTERA NO LINEAL</b>	<b>65</b>
<b>4.4 ALGORITMO DE GRADIENTE REDUCIDO GENERALIZADO</b>	<b>68</b>
<b>4.5 EJEMPLO USANDO EL PAQUETE GINO</b>	<b>74</b>
<b>4.6 CONCLUSIONES Y RECOMENDACIONES</b>	<b>75</b>
<b>4.7 REFERENCIAS AL CAPITULO IV</b>	<b>76</b>
 <b>CAPITULO V.</b>	
<b>EL PROBLEMA DEL AGENTE VIAJERO Y EL PROBLEMA DE LA MOCHILA</b>	
<b>5.1 EL PROBLEMA DEL AGENTE VIAJERO</b>	<b>78</b>
<b>5.2 ALGORITMO DE RAMIFICACION Y ACOTAMIENTO</b>	<b>81</b>
<b>5.3 EL PROBLEMA DE LA MOCHILA</b>	<b>86</b>
<b>5.4 METODO DE RAMIFICACION Y ACOTAMIENTO</b>	<b>90</b>
<b>5.5 REFERENCIAS AL CAPITULO V</b>	<b>99</b>
 <b>CONCLUSIONES</b>	<b>100</b>
<b>APENDICES</b>	<b>101</b>

## INTRODUCCION

Es muy difícil construir una única teoría capaz de describir todo el universo. En vez de ello, nos vemos forzados, de momento, a dividir el problema en varias partes, inventando un cierto número de teorías parciales.

Stephen W. Hawking. "Historia del Tiempo"

La toma de decisiones ha estado presente a lo largo de toda la historia de la humanidad. Sin embargo la racionalización y sistematización de esta tarea en problemas industriales y públicos de envergadura, solo fué posible hasta fines de la Segunda Guerra Mundial al desarrollarse lo que hoy conocemos como I. de O. Dos factores importantes confluieron e impusieron un rápido crecimiento a la nueva disciplina, como base fundamental para la toma de decisiones en todas las esferas de la sociedad:

El primero fué el progreso sustancial en la modelación matemática y la consolidación del método simplex para resolver problemas de programación lineal, desarrollada por el matemático G. Dantzig, así como la programación dinámica, la teoría de inventarios y la teoría de colas que estaban relativamente bien desarrolladas antes de finalizar los años 1950.

El segundo factor es, la irrupción del computador digital que proporcionó al tomador de decisiones una tremenda capacidad en velocidad de cómputo, almacenamiento y retiro de información. Permitiendo la aplicación y popularización de las herramientas convencionales de la I. de O.

Estos factores han dejado profunda huella durante estos primeros 40 años de desarrollo de la I. de O. El primero de ellos, sobre todo la programación lineal, tuvo un papel predominante en las técnicas de solución de problemas, mientras que el segundo juega un papel de apoyo al primero, al permitir efectuar largos procedimientos de cálculo. Esto ocasionó una de las primeras desvirtuaciones que tenemos de la I. de O., la de reducirla a una simple herramienta de solución de problemas, más aún, de confundir la investigación de operaciones con la programación lineal, este hecho se manifiesta en planes de estudio, de maestrías en administración y diversas licenciaturas donde existe la materia de Investigación de Operaciones, con programas correspondientes relativos a la programación lineal.

La Investigación de Operaciones, tuvo sus orígenes a lo largo de la segunda guerra mundial. La ciencia ha contribuido, desde los tiempos de Arquímedes, a aportar ideas destructoras, y en las dos grandes guerras de este siglo dió su ayuda técnica para hacer posible el desarrollo de sus principales armas, desde la ametralladora hasta la bomba atómica. En la última guerra, los analistas de operaciones militares se encontraron trabajando en lugares extraños y en diversas circunstancias. En Burma hubo matemáticos que discutieron problemas artilleros con soldados británicos, en Princes Risborough, un cuartel seguro fuera de Londres; unos químicos, en combinación con colegas economistas valoraron la capacidad destructora de una bomba; hubo generales que consultaron la estrategia de los carros de combate en la campaña de Italia con bioquímicos y abogados; un famoso zoólogo británico fue el hombre clave en el trazado de un plan de bombardeo sobre Pantellaria; oficiales de marina pusieron bajo secreto a estadísticos y entomólogos, en relación con las pérdidas de submarinos en el Pacífico. Fue un intenso, irregular y paradójico intercambio de ideas entre aficionados y profesionales de la guerra, que produjo algunos éxitos brillantes, sin embargo hubo más fracasos que éxitos, pero el balance final es impresionante.

Lo que aportaron los científicos a los problemas operativos -aparte de sus conocimientos especiales- fue su aspecto científico. De hecho ésta fué su principal contribución, aportaron ideas nuevas, desecharon conceptos preconcebidos y obraron sólo ante la evidencia. El cálculo de probabilidades fué su herramienta indispensable, e hicieron uso de su teoría más sutil y de su técnica más eficaz, no estuvieron limitados por axiomas de laboratorio, sino que su guía inseparable fue el sistema experimental. Sin embargo a pesar de haber empezado en la guerra, la investigación de operaciones, se ha ido extendiendo a la ingeniería, las comunicaciones, la minería, los negocios, la manufactura y otras ramas de la industria.

En nuestro país la Investigación de Operaciones es realmente muy joven ya que hasta hace aproximadamente veinte años se comenzó a trabajar en forma sistemática. El primer paso en este sentido fue la familiarización con las técnicas básicas de la I. de O. Como es de esperarse, al principio - y actualmente todavía sucede pero en menor grado- lo que se hizo fué copiar las aplicaciones desarrolladas sin ninguna creatividad o esfuerzo por usar las herramientas de la I. de O., por lo que son, instrumentos de trabajo para la solución de problemas.

Conviene señalar que las ideas que mueven a la I. de O. han sido señaladas desde hace mucho tiempo, no en el campo científico, pero sí en el ámbito artístico. Un ejemplo típico es la cita del célebre escritor inglés William Shakespeare, quien no sabía de la Investigación de Operaciones, pero sí de sus aspectos más importantes. El aconseja en Enrique IV parte dos que la planeación de modelos que no correspondan certeramente a la realidad es perjudicial. En el Acto I, escena 3 el Arzobispo de York y sus allegados deliberan si obtendrán éxito al dejar 25 000 hombres en batalla en contra del rey Enrique IV. En este contexto Lord Bardolph dice:

Quando tenemos la intención de construir, examinamos primero el emplazamiento, después dibujamos el plano; y cuando vemos la forma de la casa, entonces calculamos el costo de la construcción. Si encontramos que excede de nuestros recursos ¿que hacemos entonces? Reanudamos nuestro proyecto sobre un plano menos amplio o renunciamos por completo a construir. Mucho más es necesario todavía en ésta gran empresa, que consiste casi en desarraigat un reino para elevar otro, estudiar nuestro terreno y nuestro plano. Nos es preciso partir de una base segura, consultar a los expertos, hacer la cuenta de nuestros propios recursos para saber si la acción en que nos empeñamos podrá mantenerse contra la acción opuesta; de otro modo, es formar las tropas sobre el papel y en cifras y emplear la palabra hombre en vez de hombres reales. Nos asemejaríamos al que dibuja el plano de una casa sin recursos para construirla, y que, abandonándola a medio hacer, deja la parte que ha levantado con grandes gastos como una esclava desnuda destinada a recibir los llantos de las nubes, como una presa condenada a soportar la tiranía del riguroso invierno.

Shakespeare. Enrique IV Parte 2.

El creciente uso de la I. de O. ha hecho que sus técnicas se vayan desarrollando y diferenciando más, teniendo un desarrollo propio cada una de ellas. Entre otras podemos identificar la Teoría de Juegos, Teoría de Colas, Simulación Digital, Procesos Estocásticos y Optimización; la tabla 1 nos muestra esquemáticamente ésta idea, y desagregamos el modulo de optimización debido a la importancia para el presente trabajo acuerdo a los diferentes problemas que resuelve. En forma semejante podríamos desagregar cada modulo de la tabla 1, pero no es objeto de éste trabajo.

En relación a las técnicas de optimización podemos decir que la Programación Lineal se ha usado con éxito en la solución de problemas referentes a la asignación de personal, la mezcla de materiales, la distribución y el transporte y las carteras de inversión. La programación dinámica se ha aplicado con buenos resultados en áreas tales como la planeación de los gastos de comercialización, la estrategia de ventas y la planeación de la producción. La teoría de colas ha tenido aplicaciones en la solución de problemas referentes al congestionamiento del tráfico, al servicio de máquinas sujetas a descomposturas, a la determinación del nivel de la mano de obra, a la programación del tráfico aéreo, al diseño de presas, a la programación de la producción y a la administración de hospitales. Otras técnicas de Investigación de Operaciones, como la teoría de inventarios, la teoría de juegos y la simulación, han tenido exitosas aplicaciones en una gran variedad de contextos.

En cuanto al módulo de optimización es conveniente puntualizar que algunos problemas reales, exigen que sus soluciones sean enteras, esto es, las variables de decisión tienen sentido sólo si adquieren valores enteros. Por ejemplo, es necesario con frecuencia asignar hombres, máquinas y vehículos a las actividades, en cantidades que deben ser enteras. Esta restricción es difícil de manejarse en forma matemática, sin embargo, se han hecho algunos progresos en el desarrollo de procedimientos de solución para el caso de problemas de programación lineal, sujetos a la restricción adicional de que las variables de decisión (todas o algunas) deben tener valores enteros.

El origen de los métodos de solución del problema de la programación entera se remonta al año de 1958, cuando Gomory propuso una técnica para generar soluciones enteras usando programación lineal. Como consecuencia de esta idea se desarrollaron diferentes métodos y se han escrito diversos códigos de computación a nivel comercial. Entre los métodos de solución que se dispone se pueden mencionar:

- a) Método de planos de corte
- b) Método de Ramificación y Acotamiento
- c) Método de la teoría de grupos
- d) Métodos de la programación dinámica
- e) Métodos heurísticos.

El método de planos de corte es poco eficiente computacionalmente para el problema general de programación entera y requiere demasiadas iteraciones. Debido a estas desventajas este método no se utiliza para resolver problemas grandes de programación entera.

Los métodos de teoría de grupos resultan teóricamente interesantes pero son poco eficientes. Los métodos de programación dinámica y los heurísticos tienen una eficiencia estrechamente relacionada con el problema a que se aplican.

Un enfoque clásico a los problemas de programación lineal entera también conocidos como problemas de optimización discreta ha sido usar el método simplex (ignorando en consecuencia la restricción entera) y después, si es necesario, redondear los valores no enteros, de la solución óptima resultante. Aunque a menudo esto es adecuado, existen riesgos en este enfoque. Uno de ellos es que la solución óptima de programación lineal no necesariamente es factible después de redondearla. Asimismo es difícil saber, en que sentido debe redondearse la solución para mantener la factibilidad. Incluso puede ser necesario cambiar el valor de algunas variables en una o más unidades después del redondeo.

Aún cuando la solución óptima de programación lineal se redondee de manera satisfactoria, existe todavía otro riesgo. No hay garantía de que esta solución redondeada sea la solución óptima entera. Por estas razones, resulta útil contar con un procedimiento eficiente de solución para obtener una solución óptima entera, las técnicas de Ramificación y Acotamiento han resultado ser muy eficientes en este sentido.

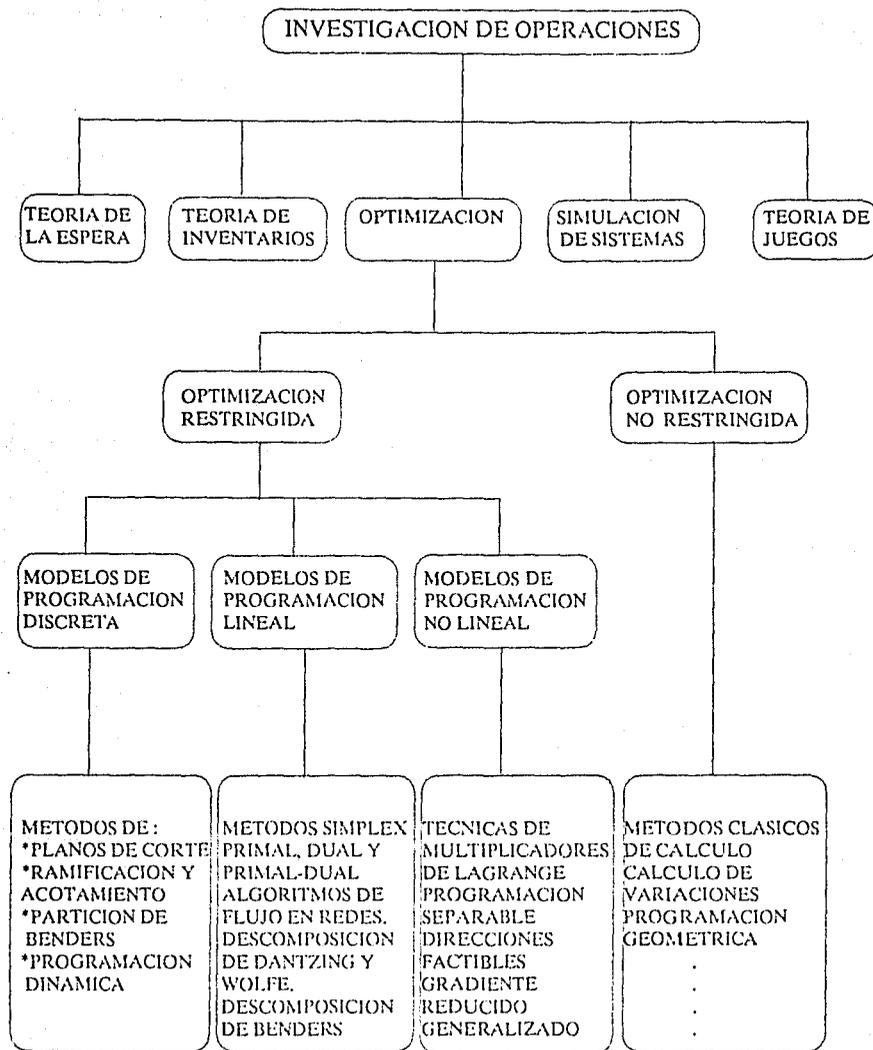
En este trabajo se revisa el método de Ramificación y Acotamiento (R-A), en la solución de problemas enteros, lineales y no lineales, y algunas de las técnicas que implantan este método y se aplican a distintas problemáticas enteras. El método de R-A, consiste esencialmente en dividir el conjunto de soluciones factibles de un problema en subconjuntos donde se buscará la solución óptima, desechando aquellos que bajo un criterio o una cota no son susceptibles de tomarse en consideración, ahorrando con esto una considerable cantidad de tiempo y esfuerzo. Actualmente las técnicas de ramificación y acotamiento se están empleando en varios campos de aplicación tales como: problemas de distribución, secuenciación de instalaciones, agente viajero, rutas de vehículos, el problema de la mochila, problemas de programación no lineal; el progreso que se ha tenido se debe en gran parte al desarrollo que han tenido las computadoras, ya que por su gran velocidad se tiene una solución más exacta y rápida sin necesidad de recurrir a métodos aproximados.

Conviene mencionar que se dispone también de paquetes computacionales que usan ramificación y acotamiento (R-A) en sus procedimientos de solución, tales como el LINDO para resolver problemas de programación lineal y entera y el GINO para problemas no lineales. Además se están desarrollando algoritmos de R-A para computación paralela.

El objetivo de este trabajo es el análisis de las bases teóricas, desarrollo, clasificación, implantación y comparación de las distintas técnicas de ramificación y acotamiento, así como la presentación de algunas de las aplicaciones clásicas y otras recientes más novedosa. Con un enfoque de síntesis y aplicaciones, presentar el estado del arte del método de ramificación y acotamiento.

El trabajo está organizado de la siguiente manera. En el primer capítulo, se define el problema y se establece el método de ramificación y acotamiento. El capítulo dos, revisa el enfoque clásico del método para programación entera mixta. En el capítulo tres, se efectúa el análisis de sensibilidad. En el cuarto capítulo, se revisa la programación no lineal. El quinto capítulo se aplica una técnica de ramificación y acotamiento a problemas clásicos. Finalmente se presentan las conclusiones del trabajo desarrollado y se agregan tres apéndices que apoyan algunas aseveraciones que se hacen a lo largo del trabajo.

TABLA N°1



# CAPITULO I

## LA TECNICA DE RAMIFICACION Y ACOTAMIENTO: ENFOQUE CLASICO

*"El segundo (principio), era la división de cada una de las dificultades con que tropieza la inteligencia al investigar la verdad, en tantas partes como fuera necesario para resolverlas. Y el último (principio), consistía en hacer enumeraciones tan completas y generales, que me dieran la seguridad de no haber incurrido en ninguna omisión"*

Descartes, Discurso del Método, Vol II.

El procedimiento de ramificación y acotamiento (R-A) proporciona una metodología de búsqueda de la solución óptima en un problema de optimización discreta. En el método de R-A, el conjunto de soluciones factibles se particiona en subconjuntos más simples (Esto es lo que se debería hacer en la práctica, si uno está buscando por ejemplo una aguja en un pajar. El pajar es grande y es imposible buscar en todo simultáneamente, así que se puede dividir visualmente en lado derecho e izquierdo y seleccionar uno de ellos para buscar la aguja, manteniendo el otro lado en espera para buscar después si es necesario). A continuación un subconjunto prometedor se selecciona y se hace un esfuerzo para encontrar la mejor solución factible y se almacena esta información, en algunos casos podría darse por terminado el método. Se particiona nuevamente el subconjunto en dos o más subconjuntos más simples (bajo la operación denominada ramificación) y se repite el mismo proceso

Este capítulo se desarrolla como sigue: En la primera sección se describe el problema de optimización discreta y los elementos básicos del método de R-A. La segunda sección se presenta el algoritmo básico de ramificación y acotamiento y el árbol de búsqueda que registra el desarrollo del algoritmo. La tercera sección se describe las distintas opciones que hay para ramificar y desarrollar la búsqueda de la solución óptima. Finalmente en la cuarta sección se exponen algunos ejemplos ilustrativos muy generales.

### 1.1 DESCRIPCION CONCEPTUAL DE RAMIFICACION Y ACOTAMIENTO

El método de Ramificación y Acotamiento (R-A), surgido hace dos décadas, ha resultado ser una de las mejores herramientas prácticas para la solución de problemas de optimización discreta. Su atractivo radica en la habilidad de eliminar implícitamente grupos grandes de soluciones potenciales sin evaluarlos explícitamente. A semejanza de la programación dinámica, la técnica de ramificación y acotamiento es una estrategia, y como tal se debe combinar con la estructura del problema específico que se desea resolver, para así formar un algoritmo de solución adecuado.

Hay muchos problemas de optimización discreta para los cuales los métodos "directos" o no existen o son ineficaces. Los problemas pueden ser tales que las restricciones o función(es) objetivo(s) son no convexas, o que todos o algunos

de los valores están restringidos a valores discretos. La técnica de ramificación y acotamiento nos posibilita resolver problemas "difíciles" usando los métodos existentes para la solución de problemas "fáciles".

Suponga que se desea resolver un problema "difícil" de optimización discreta, como el siguiente:

$$\begin{array}{l} \text{Minimizar } C^0(x) \\ \text{Sujeto a} \\ g_1^0(x) \geq 0, \\ g_2^0(x) \geq 0, \\ \dots \\ g_m^0(x) \geq 0 \\ x \in X^0 \end{array}$$

donde el conjunto  $X^0$  denota el dominio permisible de optimización por ejemplo el octante positivo del espacio  $n$ -euclidiano,  $x$  denota un vector  $(x_1, x_2, \dots, x_n)$  y  $g_i^0(x)$  son las funciones que restringen el problema. Un vector  $x$  que satisfaga las restricciones y se encuentre en el dominio de optimización se dice que es una solución factible, y uno que además minimiza la función objetivo es una solución óptima.

Existen cuatro razones importantes para aceptar ampliamente los algoritmos de ramificación y acotamiento en la optimización discreta: a) El método es conceptualmente simple y fácil de entender, b) Es fácilmente adaptable a un amplio rango de situaciones problemáticas, c) es ajustable para su implantación computacional y d) Los métodos alternativos usualmente no están disponibles.

La estrategia de aplicación de método de R-A se basa en la premisa de que el problema que se va a resolver tenga los siguientes atributos:

a) **Naturaleza Combinatoria.** Un problema combinatorio tiene como mínimo las siguientes propiedades:

a.1) Se da un conjunto finito de objetos

a.2) Cada objeto puede tomar un cierto rango de atributos.

a.3) Se desarrolla una solución al problema fijando los valores de atributos para todos los objetos

a.4) Solo se permiten ciertas combinaciones de los valores de atributos.

b) **Ramificabilidad.** Es una propiedad del problema que implica:

b.1) Construir un conjunto finito y contable que contenga todas las soluciones del problema (ésto se sigue de 1);

b.2) Particionar recursivamente un conjunto no vacío de soluciones en subconjuntos disjuntos.

c) **Racionalidad.** Un problema racional es tal que:

c.1) Cada solución tiene un único valor calculado de los valores de sus atributos.

c.2) La "mejor" solución es aquella con mayor (o menor) valor.

d) **Acotabilidad.** Una estimación del valor de la mejor solución contenida en cualquier conjunto de soluciones se puede obtener tal que:

d.1) El valor actual de la mejor solución en el conjunto es inferior o igual a la estimación (así la estimación es una cota superior);

d.2) Se hace un mínimo esfuerzo para obtener dicha estimación;

d.3) La estimación es razonablemente cercana al valor actual.

El concepto de R-A explota estas propiedades para poder implícita y explícitamente construir un árbol que describa todas las operaciones realizadas en el problema, y efectuar una búsqueda para encontrar la mejor solución.

## 1.2 EL ARBOL DE BUSQUEDA.

El proceso de ramificación del problema de optimización discreta puede representarse por medio de un árbol de búsqueda. Cada nodo del árbol está asociado a un subconjunto de  $T$ , conjunto de todas las soluciones originales del problema. Sea  $T_i$  un subconjunto de  $T$ , esto es, una colección de soluciones del problema. La Ramificación es el proceso de particionar un subconjunto  $T_i$  en  $m$  subconjuntos disjuntos  $v_1, v_2, \dots, v_m$  donde

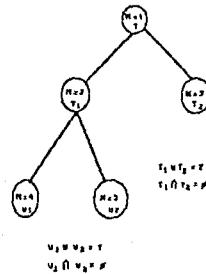
$$v_1 \cup v_2 \cup \dots \cup v_m = T_i,$$

$$v_i \cap v_j = \emptyset, \quad i \text{ distinta de } j.$$

O bien la unión de ellos es  $T_i$  y la intersección entre ellos es vacía.

El proceso de ramificación se puede visualizar como la creación o desarrollo ordenado de un árbol donde el nodo inicial representa a  $T$ , y los nodos representan subconjuntos  $T_i$  de  $T$ . A cada nodo  $N$  se le asocia un subconjunto  $T_i$ .

FIGURA 1.1



En la figura 1.1 se ilustra un árbol que exhibe estos conceptos para el caso  $m=2$ . El nodo  $N=1$  corresponde a  $T$ , el conjunto de todas las soluciones. Los nodos 2 y 3 están asociados con los conjuntos ajenos  $T_1$  y  $T_2$  de  $T$ . Note que no hay ramificaciones desde los nodos 3, 4 y 5. La relación, entre subconjuntos de  $T$  y la ramificación para la creación del árbol, se formaliza como sigue:

$T_i(N)$  es el subconjunto  $T_i$  de  $T$  representado por el nodo  $N$ .

Un Nodo Intermedio es un nodo  $N$  en el cual no ha habido ramificación.

Un Nodo Final es un nodo intermedio  $N$  para el cual  $T_i(N)$  consiste de una solución única.

$L(N)$  es una cota inferior de la función objetivo  $Z(x)$  en el conjunto de las soluciones asociadas con el nodo  $N$ , esto es,  $L(N) \leq Z(x)$  para toda  $x \in T_i(N)$ .

Una forma conceptual y al mismo tiempo general de describir el método de Ramificación y Acotamiento es como sigue.

## EL ALGORITMO BASICO DE RAMIFICACION Y ACOTAMIENTO

**PROPOSITO:** Encontrar la solución óptima de un problema de optimización discreta

### DESCRIPCION

**Paso 0.** Se inicia con el conjunto de todas las soluciones factibles del problema en cuestión. Se forma el primer nodo del árbol

**Paso 1.** Se procede a ramificar los nodos hacia nodos nuevos, utilizando alguna regla de ramificación

**Paso 2.** Se determinan cotas inferiores para los nuevos nodos. Para cada nuevo nodo N se obtiene una cota inferior  $L(N)$  sobre el valor de la función objetivo para las soluciones factibles del nodo.

**Paso 3.** Se selecciona un nodo intermedio desde el cual se ramifica. Cada nuevo nodo se excluye o se sondea si: Se encuentra que el nodo no contiene soluciones factibles, o se ha identificado la mejor solución factible en el nodo (de tal forma que  $L(N)$  corresponde al valor de su función objetivo);

**Paso 4.** Reconocer cuando un nodo final contiene una solución óptima. El proceso termina cuando no existen nodos restantes (no sondeados) y la solución de apoyo actual es la solución óptima. (si no existe tal solución de apoyo entonces el problema no tiene soluciones factibles). De otra manera se regresa al paso 1.

Si el objetivo es maximizar solo se invierten los papeles de las cotas y se invierten las direcciones de las desigualdades.

### CONVERGENCIA DEL ALGORITMO.

La operación de ramificación del algoritmo garantiza una solución óptima en un número finito de iteraciones. Como T es finito el proceso de ramificación conducirá eventualmente a una solución  $T_i$  como nodo final, a menos que se detenga previamente. El proceso de ramificación produce una partición de T para la cual cada subconjunto  $T_i$  obedece, a la definición de partición. Así todos los nodos finales posibles se pueden generar y obtenerse así la solución óptima. Las características de la ramificación prometen una enumeración completa para el algoritmo de R-A a menos que se pueda encontrar una solución óptima antes de hacerlo. Las características de acotamiento del algoritmo de R-A proporcionan la posibilidad de reconocer una solución óptima antes de completar la enumeración.

### 1.3 ESTRATEGIAS OPERATIVAS EN EL ALGORITMO BASICO.

Empezaremos primero con las distintas opciones para efectuar la operación de ramificación. Identificamos tres formas distintas de ramificar: ordenada, inmediata del sucesor y ramificación del sucesor. En aras de hacer más sencilla la exposición planteamos un problema de permutación y a través de él se exponen las distintas técnicas de ramificación.

Se desea ordenar un conjunto de n artículos  $A = \{a_1, \dots, a_n\}$ , esto es, buscamos un ordenamiento que sea óptimo de los n artículos bajo condiciones dadas. El conjunto de soluciones factibles consiste de vectores n-dimensionales.

$$F = \{x | x_i \in A \text{ y } x_i \text{ distinto de } x_j \text{ si } i \text{ diferente de } j\}$$

donde el vector x, representa un arreglo de los n artículos. La exigencia de que  $x_i$  sea diferente de  $x_j$  si i es distinto de j, se debe a que un mismo artículo  $a_i$ , no puede ocupar dos lugares en un mismo arreglo.

A continuación se discuten las posibles reglas de ramificación para el caso en que  $n=4$  y  $A = \{a, b, c, d\}$

### RAMIFICACION ORDENADA

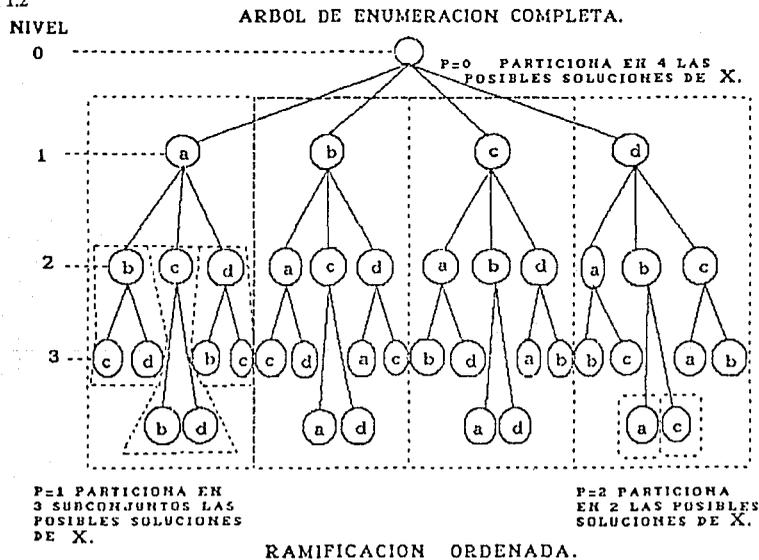
Vamos a llamar  $X$  al conjunto de soluciones factibles, que contiene los artículos ya seleccionados, llamaremos  $R(X)$  al rango de  $X$ , es el artículo que vamos a seleccionar para introducir a la solución factible que se construye, llamaremos nivel de  $X$ , al número de artículos seleccionados, así sea  $R = \{r_1, \dots, r_p\}$  un subconjunto de los índices de los artículos  $\{1, 2, \dots, n\}$  que contiene  $p$  elementos distintos, que se han introducido al arreglo. Un nodo  $X$  de un árbol de búsqueda se determina al imponer las siguientes restricciones.

$$x_{r_1} = a_{j_1}, \quad x_{r_2} = a_{j_2}, \dots, x_{r_p} = a_{j_p}$$

para algún conjunto  $\{a_{j_1}, \dots, a_{j_p}\}$  de  $p$  artículos distintos (los seleccionados), de donde, la siguiente selección de  $X$  la haremos sobre los restantes  $n-p$  artículos a ordenar, lo que nos genera una partición de  $X$  en  $n-p$  subconjuntos ajenos, al especificar el artículo que tenga orden  $r(X)$  para alguna  $r(X)$  que no está en  $R$ , seleccionamos también el nuevo conjunto de soluciones factibles a particionar.

La regla de ramificación ordenada, nos especifica las reglas explícitas de selección, para asignar un rango a  $X$ :  $r(X)$ , de esta forma  $r(X)$  se considera sólo en función del nivel de  $X$  (donde  $p = \text{nivel de } X$ ). Por ejemplo:  $r(X) = p + 1$  consiste en seleccionar primero, el artículo que irá en primer lugar del arreglo, después el del segundo lugar y así sucesivamente, de ésta forma, si se tiene un conjunto  $X = \{x_1, x_2, x_3, x_4\}$  donde la regla de selección es "mayor que" se tiene  $x_1 > x_2 > x_3 > x_4$ . Si la regla fuera  $r(X) = n - p$  entonces empezaríamos por seleccionar el último del arreglo, y terminar en el primero, en el ejemplo anterior la solución es la misma pero la selección se hace considerando primero a  $x_4$ , después a  $x_3$  y así hasta  $x_1$ .

FIGURA 1.2



La figura 1.2 muestra un árbol de enumeración completa, para  $n=4$  y  $A = \{a, b, c, d\}$  generado por ramificación ordenada.

Otra regla de selección consiste en ir insertando en forma ordenada cada uno de los artículos, de la manera siguiente, al primer artículo le asignamos su lugar en el arreglo solución. Por ejemplo considere los nodos a, b, c y d en los niveles 0, 1 2 y 3 respectivamente. Entonces Z corresponde a la solución

(b,d,a,c) si  $r(a)=3, r(b)=1, r(c)=4, r(d)=2$

(b,a,d,c) si  $r(a)=2, r(b)=1, r(c)=4, r(d)=3$

(a,b,c,d) si  $r(X) = (\text{Nivel de } X) + 1$  para toda X

(d,c,b,a) si  $r(X) = n - (\text{Nivel de } X)$  para toda X

### RAMIFICACION INMEDIATA DEL SUCESOR

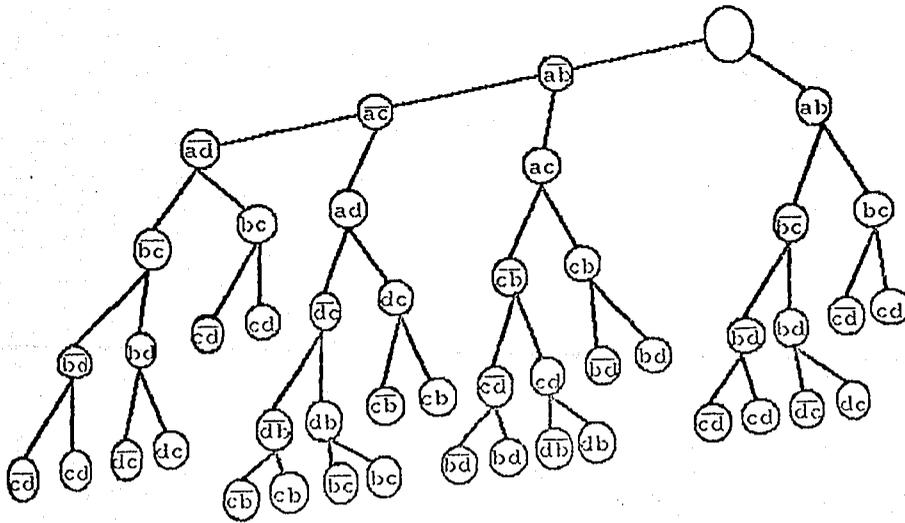
Sea X un nodo del árbol especificado por p (= nivel de X) relaciones de la forma  $r_j = r_i + 1$  que puede interpretarse como el renglón  $a_j$  ordenado inmediatamente después del renglón  $a_i$ ). Entonces X se puede particionar en dos conjuntos:

$$X(kl) = \{ x \mid x \in X \text{ y } r_l = r_k + 1 \}$$

$$X(kl^*) = X - X(kl) = \{ x \mid x \in X \text{ y } r_l \text{ distinto de } r_k + 1 \}$$

Se requiere una regla de selección para decidir el par de índices k y l. La figura 1.3 da un ejemplo de un árbol de enumeración completa para  $n=4$  y  $A = \{a, b, c, d\}$

FIGURA 1.3



### RAMIFICACION DEL SUCESOR

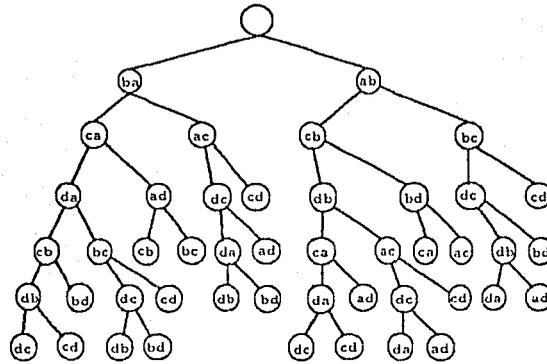
Sea  $X$  un nodo del árbol especificado por  $p$  relaciones  $r_j > r_i$  (que se interpretan como un renglón  $a_j$  ordenado posteriormente a un renglón  $a_i$ ). Donde  $X$  puede particionarse en dos conjuntos

$$X(kl) = \{x | x \in X \text{ y } x_l > x_k\}$$

$$X(lk) = X - X(kl) = \{x | x \in X \text{ y } x_k > x_l\}$$

Nuevamente se requiere de una regla de selección para escoger la pareja  $(k, l)$ ,  $k \neq l$ . La figura 1.4 da un ejemplo de un árbol de enumeración completa.

FIGURA 1.4



### ACOTAMIENTO

Un árbol de búsqueda puede formarse con una pequeña parte de un árbol de enumeración completa (en el cual no se descarta ningún nodo, se consideran todos los posibles), debido al tamaño tan grande de éste. Para poder hacer esto escogemos una regla de acotamiento adecuada. Una función de acotamiento es una función  $g: 2^I \rightarrow \mathbb{R}$  que asigna un número real  $g(X)$  a cada subconjunto  $T$  tal que si  $X$  está contenido en  $T$  entonces la cota  $g(X)$  satisface

$$g(X) \leq f(x) \quad \forall x \in X \cap F$$

(Note que las cotas son todas cotas inferiores ya que estamos suponiendo un problema de minimización). Podemos suponer también que

$$g(\{x\}) = f(x) \quad \forall x \in F \quad (1.4.1)$$

Cotas para un problema dado  $P$  pueden obtenerse relajando alguna de las restricciones. Esto da el problema relajado  $Q$  cuyo conjunto factible  $F_Q$  contiene al conjunto factible  $F_P$  del problema original  $P$ . Sea  $X$  un subconjunto de  $T$  entonces

$$g(X) = \min_{y \in X \cap F_Q} f(y) \leq \min_{y \in X \cap F_P} f(y) \leq f(x) \quad \forall x \in X \cap F$$

claramente  $g$  es una función de acotamiento inferior.

Una vez que hemos establecido las reglas de ramificación y acotamiento, describiremos a continuación las principales estrategias de desarrollo del algoritmo de R-A, las más importantes son:

#### BUSQUEDA A PRIMER PROFUNDIDAD O REGLA DE LA COTA MAS RECIENTE O TAMBIEN LIFO

Es la estrategia de ramificación del nodo activo más reciente (un nodo  $Y$  es activo dependiendo de que  $f(S) > g(Y)$  donde  $S$  es la solución actual, y  $Y$  es algún nodo del conjunto total de soluciones), esto es, se escoge el nodo de máxima profundidad entre aquellos nodos que aún no se ramifican. Si hay más de uno, entonces se selecciona aquél que corresponda al valor de cota inferior.

Esta regla logra que el árbol llegue a un nodo terminal rápidamente, aunque se podrían requerir más cálculos computacionales, se requiere un mínimo de memoria, ya que es suficiente para acumular problemas correspondientes a cada nodo en una ruta de la raíz del árbol al nodo terminal.

En la terminología de Ramificación y Acotamiento esta estrategia de desarrollo se conoce como búsqueda en la lista de nodos activos, bajo el criterio que el último en entrar es el primero en salir.

#### PRIMER ALCANCE O REGLA DE LA MENOR COTA O TAMBIEN FIFO

Esta regla también se conoce como método o regla de separación progresiva y evaluación. Aquí se escoge sistemáticamente el nodo con el valor de la menor cota observando que es en éste en el que se tiene intuitivamente mayor oportunidad de que contenga una solución óptima entre sus sucesores. Sin embargo, con este método corremos el riesgo de tener que explorar una gran porción del árbol antes de encontrar una solución. Nuevamente, la primer solución que se encuentra es en general mejor (esto es, de menor costo) que la que se encuentra por la estrategia de búsqueda de "primer profundidad".

Esta regla se usa también en la programación dinámica y es la opuesta a la de primera profundidad en la estrategia de ramificar un "mínimo nodo activo recientemente creado". También se conoce como FIFO porque si se tiene una lista de nodos activos es el primero en entrar y el primero en salir.

#### COSTO UNIFORME

Esta regla consiste en escoger un nodo  $X$  del árbol parcialmente expandido, para el cual:

$$g(X) \leq g(Y) \quad \forall \text{ nodo } Y$$

los empates se resuelven por alguna regla subsidiaria. Esta es la ramificación de la menor cota o estrategia de costo uniforme (UC) y es computacionalmente óptima en el sentido de que efectivamente requiere un mínimo número de nodos a ser desarrollados. Aunque produce árboles pequeños, puede caer en dejar un mínimo tiempo de cómputo y también sufre el defecto de producir primero una solución factible en o cerca del fin de los cálculos.

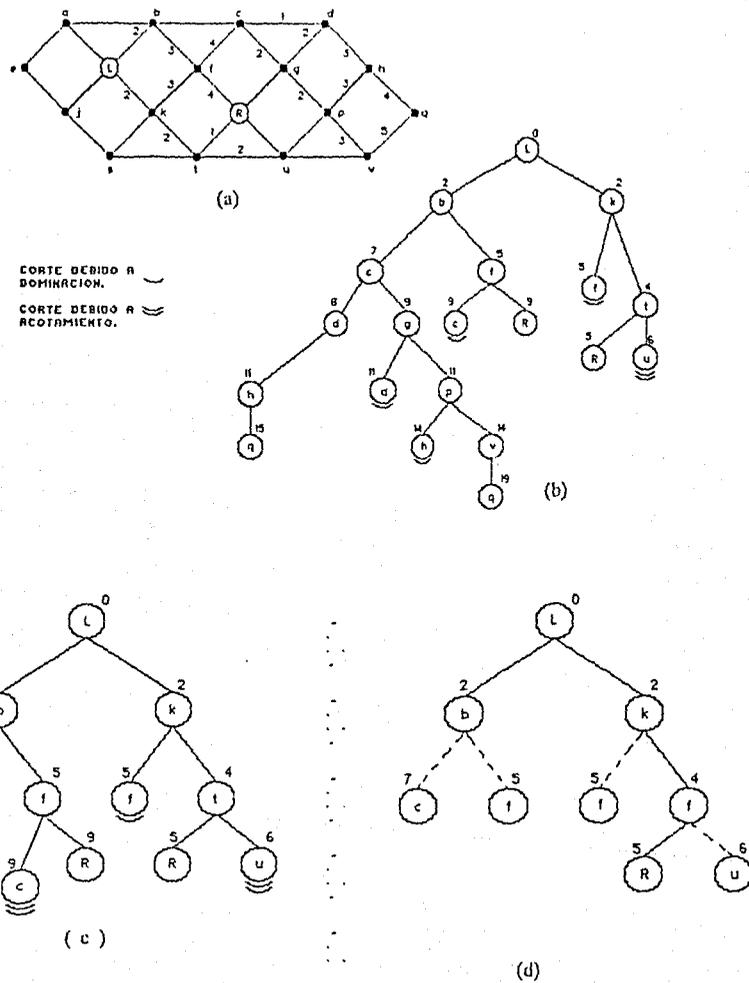
Tres estrategias se describen en la figura 1.5 para el problema de búsqueda para encontrar la trayectoria mas corta de  $L$  a  $R$ , de la gráfica a). En b) se usa primer profundidad, en c) primer amplitud y en d) costo uniforme.

#### DOMINACION

En algunas aplicaciones de ramificación y acotamiento puede suceder que hay caminos fuera de la estructura de R-A que permiten establecer que para algún par de conjuntos  $X, Y$  contenidos en  $T$  la mejor solución en  $X$  es al menos tan buena como la mejor solución en  $Y$ . Entonces se dice que  $X$  domina a  $Y$ , y el nodo de  $Y$  en el árbol no necesita desarrollarse (Si  $X$  domina a  $Y$  y  $Y$  domina a  $X$  entonces uno de ellos puede estar inactivo).

Finalmente, puede notarse que las técnicas de ramificación y acotamiento son exactas en el sentido de que garantizan soluciones óptimas. Sin embargo, involucran una cantidad considerable de cálculos y si una buena solución subóptima es aceptable entonces a menudo es mejor usar ramificación y acotamiento "heurísticamente" en donde la mejor solución factible generada, en un tiempo dado, es la que se considera como la mejor.

FIGURA 1.5



### 1.5 EJEMPLOS ILUSTRATIVOS.

En los ejemplos siguientes se exhibe la forma en que se pueden emplear las reglas de ramificación y acotación. El último ejemplo es el famoso caso de las cuatro reinas en donde ilustramos las reglas de desarrollo a primer profundidad y de primer alcance.

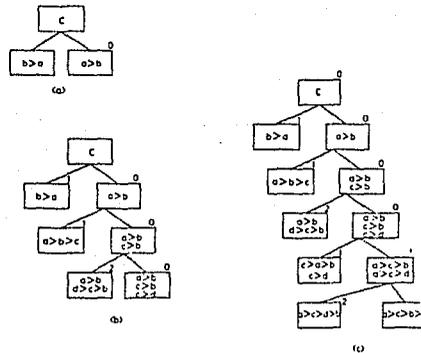
#### Ejemplo 1.1 (Ramificación)

En un experimento de comparación apareada sobre los elementos de un conjunto  $\{a,b,c,d\}$  las preferencias se resumen en la siguiente matriz, donde 1 indica que existe discrepancia y 0 que no la hay. Si aceptamos que las preferencias queden como (abcd) el costo en que se incurre es de 2 calculado como sigue: por preferir a, a sobre las opciones b, c y d se paga una unidad, por preferir b sobre c y d se paga otra unidad y finalmente por preferir c a d no se paga nada. Se requiere encontrar un ordenamiento que minimice el número de discrepancias.

	a	b	c	d	puntuación total
a	-	1	1	0	2
b	0	-	0	1	1
c	0	1	-	1	2
d	1	0	0	-	1

Empezaremos con el conjunto  $\{a, b, c, d\}$  donde la primera decisión que se presentará es: ¿Podría a ordenarse por encima de b, o no? y aplicamos la regla de ramificación del sucesor.

FIGURA 1.6



En la figura 1.6 se muestran tres etapas del desarrollo del árbol de búsqueda en esta solución. Las cotas inferiores se muestran a un lado de cada nodo. De los dos nodos aumentados al árbol de búsqueda, el primero con  $a > b$  (esto es a ordenada por encima de b) es el más prometedor, donde  $a > b$  no implica -inmediatamente- discrepancias. De la matriz de preferencias se observa que c, también es notablemente mejor que b, y la segunda ramificación corresponde de un modo u otro a  $c > b$ .

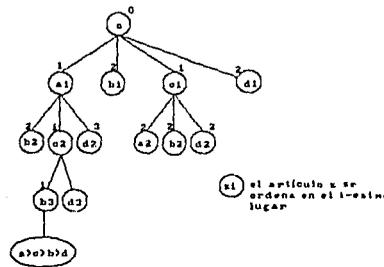
Después de tres ramificaciones el árbol de búsqueda desarrollado se muestra en la figura 1.6c). Ya que el ordenamiento  $a > c > b > d$  involucra solo una discrepancia y cualquier solución correspondiente al subconjunto de

soluciones en cualquier otro nodo implica al menos una discrepancia,  $a > c > b > d$  debe de ser un ordenamiento óptimo. En esta etapa todavía no se demuestra que esta solución óptima sea única. Para establecer esto, o encontrar otras soluciones óptimas, es necesario desarrollar todos los nodos que consisten de sólo una discrepancia.

Este método proporciona fácilmente una solución para este ejemplo, en cierto modo, debido a la manera juiciosa de escoger la ramificación. Otra forma de resolver este problema consiste en usar la regla de ramificación ordenada, con  $r(X) = p + 1$ .

Sea entonces  $x_1 > x_2 > x_3 > x_4$  un ordenamiento, donde  $x_i$  con  $i = 1, \dots, 4$  puede ser un elemento cualquiera del conjunto  $\{a, b, c, d\}$ . Hay  $4^4 = 256$  cuartetos distintos  $(x_1, x_2, x_3, x_4)$  de las que solo  $4! = 24$  corresponden a ordenamientos válidos. Esto es, de los 256 candidatos a solución, 232 no son factibles y deben de ser eliminados.

La regla de ramificación será adoptada de la siguiente manera: La primer ramificación corresponde a la selección de  $x_1$ , el elemento del ordenamiento mas alto, la segunda y tercera ramificaciones corresponden a la selección de  $x_2$  y  $x_3$  respectivamente (si se especifican  $x_1, x_2$ , y  $x_3$  entonces  $x_4$  queda automáticamente determinada). El árbol de búsqueda



se desarrolla en forma similar que en el primer método, como se muestra en la figura 1.7. Como vimos anteriormente  $a > c > b > d$  se encuentra que es óptima.

En este caso, ambos métodos de solución nos conducen a un ordenamiento óptimo. En general no es éste el caso y cierto retroceso será necesario. La solución que se obtiene con ambos métodos es  $a > c > b > d$  con una discrepancia.

### Ejemplo 1.2 Problema de Asignación Lineal (Acotamiento)

Suponga que se desean usar cuatro barcos cargueros para transportar bienes de un puerto base a otros cuatro puertos (numerados 1, 2, 3 y 4). Se puede usar cualquier barco para hacer cualquiera de los cuatro viajes. Sin embargo, dadas las diferencias entre barcos y las cargas a llevar, el costo total de carga, transporte y descarga de bienes para distintas combinaciones de barcos y puertos varía. Los costos se muestran en la siguiente tabla donde el componente (1,1) de dicha matriz es 15, que es el costo de transportar carga con el barco 1 al puerto 1 (desde el puerto base).

PUERTO		1	2	3	4	mínimo por renglón
B						
A	1	15	5	9	5	5
R	2	17	14	4	7	4
C	3	9	13	13	7	7
O	4	13	10	6	15	6

Se desea asignar los barcos a los puertos, en una correspondencia uno a uno de manera que se minimice el costo total. Si definimos  $T = \{x = (i,j,k,l) \in \{1,2,3,4\}\}$  como el conjunto de soluciones posibles., el problema de asignación (lineal) P consiste en minimizar

$$z = c_{1i} + c_{2j} + c_{3k} + c_{4l}$$

donde  $c_{ij}$  son los costos de transportar carga con el barco  $i$  al puerto  $j$  y donde  $(i,j,k,l)$  son las permutaciones de  $(1,2,3,4)$

Ejemplos de algunas cotas de costo son :

$$g(T) = \min c_1 + \min c_{2\beta} + \min c_3 + \min c_4 = 5 + 4 + 7 + 6 = 22$$

$$1 \leq \alpha \leq 4 \quad 1 \leq \beta \leq 4 \quad 1 \leq \gamma \leq 4 \quad 1 \leq \delta \leq 4$$

$$g(T\{i=3\}) = c_{13} + \min c_{2\beta} + \min c_3 + \min c_4 = 9 + 4 + 7 + 6 = 26$$

$$1 \leq \beta \leq 4 \quad 1 \leq \gamma \leq 4 \quad 1 \leq \delta \leq 4$$

$$g(T\{i=3, j=2\}) = c_{13} + c_{22} + \min c_3 + \min c_4 = 9 + 14 + 7 + 6 = 36$$

$$1 \leq \gamma \leq 4 \quad 1 \leq \delta \leq 4$$

donde  $T\{i=3\}$  denota el subconjunto de  $T$  con el índice  $i$  fijo en 3. Se puede ver que si  $i$  es el conjunto evaluado en 3 podemos insistir que los índices usados en la minimización sean  $\beta, \gamma$  y  $\delta = 3$ . Esto permite perfeccionar la función de acotamiento  $g'$

$$g'(T) = g(T) = 22$$

$$g'(T\{i=3\}) = c_{13} + \min_{\beta=1,2,4} c_{2\beta} + \min_{\gamma=1,2,4} c_3 + \min_{\delta=1,2,4} c_4 =$$

$$= 9 + \min\{17, 14, 7\} + \min\{9, 13, 7\} + \min\{13, 10, 15\}$$

$$= 9 + 7 + 7 + 10 = 33$$

$$g'(T\{i=3, j=2\}) = c_{13} + c_{22} + \min_{\gamma=1,2,4} c_3 + \min_{\delta=1,2,4} c_4 =$$

$$= 9 + 14 + \min\{9, 7\} + \min\{13, 15\} = 43$$

Observe que  $g'$  satisface la ecuación (1.4.1) sin modificación y nótese también que  $g'(X) \geq g(X)$  para toda  $X$  en  $T$ ; se dirá que  $g'$  es más fuerte que  $g$ .

El siguiente problema es un ejemplo clásico del uso de los métodos de tanteo y de los algoritmos de rastreo inverso. Se dice que fue investigado por Gauss en 1850, pero no lo resolvió completamente.

Ejemplo 1.3 (El Problema de las Cuatro Reinas).

-¿Sabes sumar?-le preguntó la reina blanca- ¿Cuánto es uno y uno?

-No sé -dijo Alicia- he perdido la cuenta...

-¡No tiene ni idea de matemáticas! -sentenciaron enfáticamente ambas reinas a la vez.

Lewis Carroll. Alicia a través del Espejo.

Cuatro reinas se deben colocar en un tablero de cuatro por cuatro, de forma tal que ninguna reina pueda atacar a otra. En otras palabras, la colocación debe ser tal que en cada renglón, columna o diagonal del tablero, a lo más contenga a una reina. El problema puede considerarse como una sucesión de problemas, en donde por ejemplo se coloca una reina en el primer cuadro del renglón de arriba, entonces se coloca otra reina en el segundo renglón de forma tal que no pueda ser atacada por la primera, y de igual forma se procede con la tercera y la cuarta reina.

Podemos asociar posiciones con nodos de un árbol, donde el nodo raíz  $T$  corresponde a la posición inicial sin reinas, los siguientes nodos corresponden a las posiciones posibles de una reina en el primer renglón (Es suficiente considerar como posiciones iniciales los primeros dos cuadros, ya que los otros dos dejan posiciones simétricas), conectados con el nodo  $T$ , por arcos de longitud cero, las siguientes ramas del árbol se construyen a partir de éstos nodos, considerando

las posibles posiciones de una reina en el segundo renglón, sin que ésta ataque a la reina que se ha colocado con anterioridad, procediendo a unir éstos nodos con arcos de longitud cero, así se continúa hasta colocar a las cuatro reinas en sus posiciones respectivas.

Resolveremos primero éste problema usando la estrategia de búsqueda de primer profundidad. Numeramos los renglones y las columnas del tablero del 1 al 4, las reinas también pueden numerarse del 1 al 4. Ya que cada reina debe estar en diferente renglón, podemos suponer sin pérdida de generalidad que la reina  $i$  debe encontrarse en el renglón  $i$ . Todas las soluciones al problema deben presentarse entonces como cuartetos  $(x_1, x_2, x_3, x_4)$  donde  $x_i$  es la columna donde se pone la reina  $i$ . Las restricciones explícitas usando ésta formulación son  $S_i = \{1, 2, 3, 4\}$ ,  $1 \leq i \leq n$ . Entonces el espacio de soluciones consiste de  $4^4$  cuartetos. Las restricciones implícitas para éste problema es que no puede haber dos  $x_i$ 's que sean iguales (esto es, todas las reinas deben estar en columnas distintas) y dos reinas no pueden estar en la misma diagonal. La primera de éstas dos restricciones implica que todas las soluciones son permutaciones de los cuartetos  $(1, 2, 3, 4)$ . Esta realización reduce el tamaño del espacio de soluciones de  $4^4$  a  $4!$ . Mas tarde veremos como formular la segunda restricción en términos de las  $x_i$ . Expresada como una solución la cuarteta de la figura 1.8 se tiene  $(2, 4, 1, 3)$ .

FIGURA 1.8

	R		
			R
R			
		R	

Como una función de acotamiento usamos el criterio obvio de que si  $(x_1, x_2, \dots, x_i)$  es la trayectoria del T-nodo actual entonces todos los nodos hijos con etiquetas padre-hijo  $x_{i+1}$  tales que  $(x_1, \dots, x_{i+1})$  representa una configuración en donde no se pueden atacar dos reinas. Comenzamos con el nodo raíz como el único nodo vivo, éste es el T-nodo y la trayectoria es (). Se genera un hijo. Suponemos que generamos hijos en orden ascendente, así el nodo numerado 2 de la figura 1.11 se genera y la trayectoria es ahora (1). Esto corresponde a poner la reina 1 en la columna 1. El nodo 2 se convierte en el T-nodo. El nodo 3 se genera e inmediatamente se muere. El siguiente nodo generado es el nodo 8 y la trayectoria se convierte en (1, 3). El nodo 8 se convierte en el T-nodo, sin embargo se muere ya que sus hijos representan configuraciones que no producen una solución. Regresamos al nodo 2 y generamos otro hijo, el nodo 13. La trayectoria es ahora (1, 4). La figura 1.9 muestra las configuraciones en el tablero como procedimientos recursivos, y muestra gráficamente los pasos que sigue el algoritmo al tratar de encontrar la solución, los puntos indican los lugares de una reina que se probaron y rechazaron porque otra reina estaba atacando. En b) la segunda reina se pone en las columnas 1, 2 y finalmente se fija en la columna 3. En c) el algoritmo trata las cuatro columnas y no es posible colocar en ninguna a la próxima reina. Regresando se tiene en d) que la segunda reina se mueve a la próxima columna posible, la columna 4 y se pone la tercer reina en la columna 2. Estos tableros muestran los pasos que el algoritmo sigue hasta encontrar la solución. La figura 1.10 muestra la parte del árbol que se ha generado usando la estrategia de primera profundidad. Los nodos que se han muerto como resultado de las funciones de acotamiento tienen una M debajo de ellos.

FIGURA 1.9

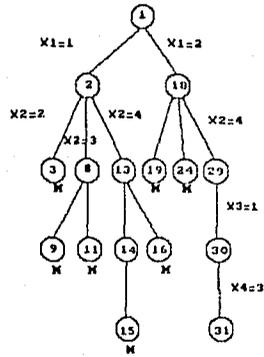
1			1		1		1		
			2			2			2

(A)      (B)      (C)      (D)

1			1		1		1		
		2				2			2
	3							3	
									4

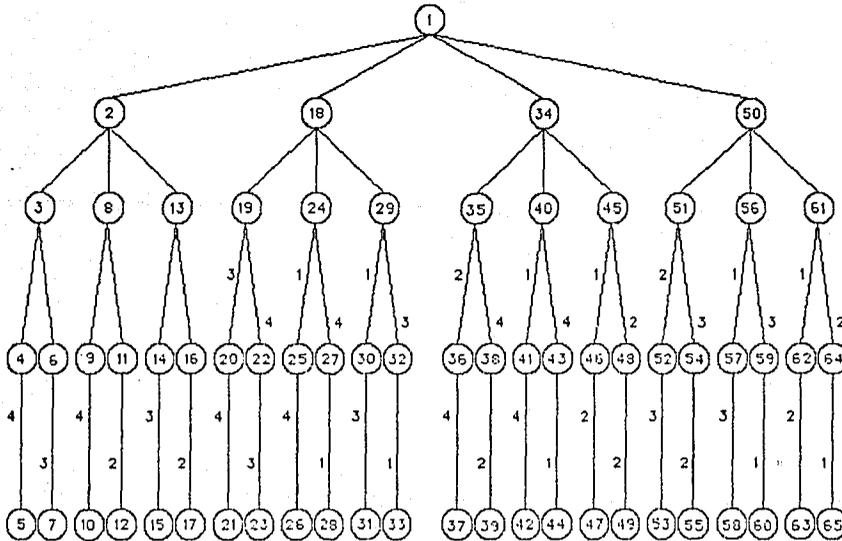
(E)      (F)      (G)      (H)

FIGURA 1.10



El árbol completo, usando búsqueda de primer profundidad y sin considerar las funciones de acotamiento se muestra en la figura 1.11

FIGURA 1.11

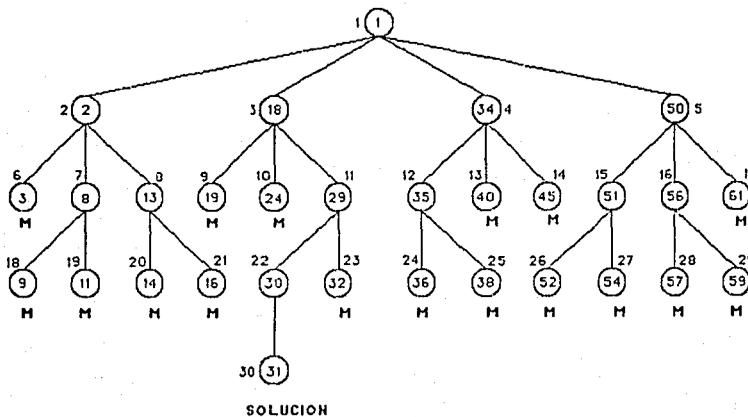


Otra forma de resolverlo es usando la estrategia de primer amplitud o FIFO. Como ya se decía inicialmente hay un solo nodo vivo, el T-nodo y ahora le asignamos el número 1, (este es el caso en que no se tienen reinas en el tablero). Este nodo se particiona y sus hijos los nodos 2, 18, 34 y 50 se generan. éstos nodos representan el tablero con una reina en el renglón 1 y columnas 1,2,3 y 4 respectivamente. Ahora los nodos vivos son los nodos 2,18,34 y 50. Si los nodos se generaron en éste orden el próximo T-nodo sería el nodo 2, éste se ramifica y los nodos 3, 8 y 13 se generan. El nodo 3 se mata inmediatamente usando la función de acotamiento.

Los nodos 8 y 13 se aumentan a la cola de nodos vivos. El nodo 18 se convierte en el nuevo T-nodo. Los nodos 19, 24 y 29 se generan. Los nodos 19 y 24 se mueren como resultado de la función de acotamiento, que es la misma que se usó en la solución por primer profundidad. El nodo 29 se aumenta a la cola de nodos vivos. El próximo T-nodo es el nodo 34. La figura 1.12 muestra el árbol generado por la estrategia FIFO (primero en entrar primero en salir).

Los números dentro del nodo son los mismos se la estrategia de primer profundidad y los números fuera del nodo corresponden al orden en que fueron generados por FIFO, en el momento en que se alcanza el nodo solución 31, los únicos nodos que permanecen vivos son los nodos 38 y 54. Una comparación de las figuras 1.10 y 1.12 indican que la estrategia de primer profundidad es un mejor método para éste problema.

FIGURA 1.12



#### REFERENCIAS PARA EL CAPITULO I.

1. AGIN, N. "Optimum Seeking with Branch and Bound", Management Science 13, p.B-176-185, 1966
2. BALAS, E. "An Additive Algorithm for Solving Linear Programs with Zero-One Variables". Operations Research 13, p.517-546, 1965.
3. BALAS, E. "A Note on the Branch and Bound Principle". Opns. Res. 16, p.442-445, 1968.
4. BOFFEY, T.B. "Graph Theory in Operations Research". Capítulo 3, Mac Millan 1982.
5. FUENTES MAYA, S. "Programación Entera". DEPMI, 1985.
6. HOROWITZ, E. Y SAHNI, S. "Fundamentals of Computer Algorithms" Capítulo 8, Computer Science Press, 1984.
7. LARNDER, H. "The Origin of Operational Research". Opns. Res. 32, p.465-475, 1984.
8. LAWLER, E.L. Y WOOD, D.E. "Branch-and-Bound Methods: A Survey". Opns. Res. 14, p.669-719, 1966.
9. MITTEN, L. G. "Branch-and-Bound Methods: General Formulation and Properties". Opns. Res. 24-35, 1969
10. MORSE, P.M. "The Beginnings of Operations Research in the United States". Opns. Res. 34, 10-17, 1986.
11. OCHOA-ROSSO, F. "Applications of Discrete Optimization Techniques to Capital Investment and Network Synthesis Problems" Research Report R68-42 Massachusetts Institute of Technology, Cambridge 1968.
12. THESEN, A. "Computer Methods in O.R." Capítulo 8, Academic Press, 1978.

## CAPITULO II

### PROGRAMACION ENTERA MIXTA: ENFOQUE CLASICO

"Primero, deben abrazarse de una ojeada todas las ideas particulares desparramadas acá y allá, y reunir las bajo una sola idea general, para hacer comprender, por una definición exacta, el objeto que se quiere tratar."

"Después se debe saber dividir de nuevo la idea general en sus elementos, como otras tantas articulaciones naturales, guardándose, sin embargo, de mutilar ninguno de éstos elementos primitivos, como acostumbra un mal cocinero cuando trincha".

Fedro, Platón.

El origen de los métodos de solución de la programación entera se remonta al año 1958, cuando Gomory propuso una técnica para generar soluciones enteras usando programación lineal. Como consecuencia de esto se desarrollaron diferentes métodos y se han escrito diversos códigos de computación a nivel comercial. Entre los métodos de solución actuales, están los de ramificación y acotamiento.

El objetivo de éste capítulo es describir el desarrollo de los métodos clásicos de solución de programación entera mixta y la forma en que se han ido depurando. El primer método que se presenta es el que desarrollaron A.H.Land y A.G. Doig, en 1960. Posteriormente R.J.Dakin propone un nuevo algoritmo, basado en el anterior pero con modificaciones que lo hacen más rápido y más tarde Norman J. Driebeck y Stanley Zionts presentan un nuevo método que plantea una alternativa distinta a las dos anteriores. Cabe señalar que aunque todos estos algoritmos se refieren a problemas de programación lineal mixta se pueden aplicar a problemas de programación entera pura.

#### 2.1 EL ALGORITMO DE LAND Y DOIG.

Considere el problema:

$$(2.1) \text{ maximizar } z = cx + dy \\ \text{ sujeto a}$$

$$(2.2) Ax + Dy \geq b$$

$$(2.3) x \text{ vector columna con componentes enteros no negativos}$$

$$(2.3') x \geq 0$$

$$(2.4) y \geq 0$$

donde  $z$  es un escalar;  $b$  es un vector columna de  $m$  componentes  $e^i$ ,  $x$  son vectores columna de  $n_1$  renglones;  $d^i$  y  $s$  son vectores columna de  $(n-n_1)$  componentes;  $A$  es una matriz de orden  $m \times n_1$ ; y  $D$  es una matriz de orden  $m \times (n-n_1)$ . Una solución factible al problema es aquella que satisface (2.2), (2.3) y (2.4).

Este método usa sistemáticamente hiperplanos paralelos al definido por la función objetivo. Una primer cota superior a la función objetivo se obtiene al resolver el problema sin las restricciones de variable discreta. Y sucesivamente se van encontrando cotas superiores más restrictivas  $z^1, z^2, \dots, z^k$  al valor de la función objetivo conforme se van agregando restricciones de variable entera, al agregar éstas restricciones se definen subproblemas subsidiarios de la siguiente manera:

$P(j)$ : Maximizar  $z$  sujeto a las restricciones (2.2), (2.3) y (2.4) y las restricciones adicionales en el sentido de que  $j$  de las variables son enteros no negativos ( $j = 1, 2, \dots, n_1$ ).

Sea  $S_j$  el conjunto de todas las soluciones factibles a problemas del tipo  $P(j)$  y sea  $S^*$  el conjunto de soluciones no factibles a cualquier problema del tipo  $P(j)$ , con esta notación, la solución requerida es un elemento de  $S_{n_1}$  para el cual se maximiza  $Z$ .

#### ALGORITMO 2.1 Land-Doig

**PROPOSITO:** Resolver el problema de programación entera mixta.

#### DESCRIPCION

La solución se construirá en forma de gráfica de árbol cuyos nodos son elementos de uno de los conjuntos  $S_j$   $j = 0, 1, \dots, n_1$  o del conjunto  $S^*$ . Los pasos de esta construcción son

**Paso 0.** El primer nodo del árbol es la solución óptima de  $P(0)$ , que está dada por la etiqueta  $z^0$ . Si esta solución satisface también  $P(n_1)$ , terminamos se tiene la solución óptima.

**Paso 1** Si  $z^0$  no es el valor óptimo se traza un arco a cada uno de los dos puntos en  $S_1$  (o posiblemente en  $S^*$ ). Se evalúa  $z$  en esos puntos, si están en  $S_1$ . Si están en  $S^*$  es inmediato que no necesitan calcularse.

**Paso 2** Si los nodos  $z^0, z^1, \dots, z^{k-1}$  ya se han etiquetado, el nodo más alto no etiquetado (de acuerdo al valor de  $z$  que no está en  $S^*$ ) se etiqueta  $z^k$ .

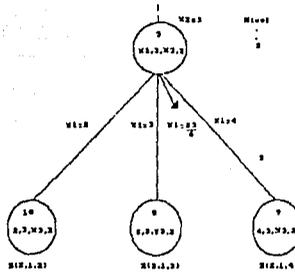
**Paso 3** La solución final se ha alcanzado cuando por primera vez un nodo etiquetado es un elemento de  $S_{n_1}$ , esto ocurre tan pronto como todas las variables de una solución son enteros no negativos. Si  $z^k$  no es tal solución se supone que es un elemento de  $S_j$ . Se dibuja un nuevo arco originando así el nodo que está inmediatamente abajo de  $z^k$  (etiquetado, que no necesariamente  $z^{k+1}$ ) y terminando en un punto en el mismo subconjunto de soluciones ( $S_j$ ) como  $z^k$  o en  $S^*$ . Si este nuevo nodo está en  $S_j$ , su valor  $z$  es necesariamente menor o igual que  $z^k$ .

**Paso 4** Se dibujan dos arcos de  $z^k$  a los puntos en  $S_{j+1}$  o en  $S^*$ . Nuevamente, si esos puntos están en  $S_{j+1}$  sus valores  $z$  son menores o iguales a  $z^k$ . Los últimos dos pasos aumentan tres nuevos arcos y nodos al árbol y regresamos al Paso 2.

Los nodos etiquetados forman una sucesión de cotas superiores no crecientes en el valor  $z$  de la solución final. Además el valor  $z$  de un nodo no etiquetado del conjunto  $S_j$  no puede exceder al del nodo etiquetado y entonces a la vez que  $z^k$  está etiquetado, es la mejor cota superior actual en el valor óptimo de  $z$ . Así, el método usado para aumentar arcos y nodos a la gráfica cubre todas las posibilidades, el algoritmo debe alcanzar la solución óptima de  $P(n_1)$ , siempre que tal solución exista.

Como se podrá observar los algoritmos enumerativos son usualmente más sencillos de entender si se ilustran geoméricamente a través de un árbol compuesto de nodos y arcos. Un nodo también se puede hacer corresponder a un punto  $x^k$  y un arco une a ese nodo  $x^k$  con otro nodo  $x^{k+1}$ , donde definimos el último punto del anterior haciendo una de sus  $n-k$  variables libres igual a un entero no negativo. Como una variable libre  $x_r$  puede tener usualmente varios valores, es posible tener muchos arcos que emanen desde el nodo  $x^k$ . Como se puede ver en la figura 2.1 los nodos numerados 8, 9 y 10 se crearon haciendo  $x_1$ , una variable libre en el nodo 5 igual a 3, 4 y 2 respectivamente. Los nodos que no se han usado para crear otros nodos, o que no tienen arcos se llaman activos. Por ejemplo, el nodo 5 no es un nodo activo. También el número de nivel  $l$  se refiere al número de variables fijas.

FIGURA 2.1



## 2.2 EJEMPLO ILUSTRATIVO

Considere el problema

$$\begin{aligned} \text{MAX } Z &= 77.9 X_1 + 76.8 X_2 + 89.6 X_3 + 97.1 Y_1 + 31.3 Y_2 \\ \text{SUJETO A:} \\ 10.9 X_1 + 3.6 X_2 - 40.8 X_3 + 43.9 Y_1 + 7.1 Y_2 + Y_3 &= 82.3 \\ -86.8 X_1 + 32.7 X_2 + 24.3 X_3 + 13.8 Y_1 - 12.6 Y_2 + Y_4 &= 77.3 \\ 60.9 X_1 + 68.9 X_2 + 69 X_3 - 56.9 Y_1 + 22.5 Y_2 &= 86.5 \\ \text{con } X \geq 0, Y \geq 0, X \text{ entero} \end{aligned}$$

Resolver usando el método de Land-Doig.

### ITERACION 1

Paso 0 Resolvemos el problema original sin incluir las restricciones de variables enteras, aplicando el método simplex y obtenemos:

$$\text{VALOR DE LA FUNCION OBJETIVO} = 1165.50600 = z^0$$

Con variables de decisión:

$$\begin{aligned} X_1 &= 1.496009 \\ X_2 &= 0 \\ X_3 &= 5.021035 \\ Y_1 &= 6.169743 \\ Y_2 &= Y_3 = Y_4 = 0 \end{aligned}$$

Paso 1. Las variables no enteras, son  $X_1$  y  $X_3$ . Elegimos como variable de ramificación la que tenga el valor más apartado al entero más cercano;  $X_1$  es la más lejana a un valor entero. Calculamos el primer valor entero en las funciones  $\text{Min } X_1$  y  $\text{Max } X_1$ , es inmediato que  $1 = \text{Min } X_1$  y  $2 = \text{Max } X_1$

1.1  $\text{Min } X_1$ . Sea  $X_1 = 1$ , el valor óptimo de la función objetivo es

$$Z = 1051.89500$$

con los siguientes valores en las variables

$$X_1 = 1$$

$$\begin{aligned} X_2 &= 0 \\ X_3 &= 4.408884 \\ Y_1 &= 5.483788 \\ Y_2 &= 1.485092 \\ Y_3 &= Y_4 = 0 \end{aligned}$$

1.2 Sea  $X_1 = 2$ . El valor de la función objetivo es

$$Z = 770.698300$$

con los siguientes valores en las variables

$$\begin{aligned} X_1 &= 2 \\ X_2 &= 0 \\ X_3 &= 2.675005 \\ Y_1 &= 3.864242 \\ Y_2 &= Y_3 = 0 \\ Y_4 &= 132.570900 \end{aligned}$$

Paso 2 El mayor de los dos valores objetivo es  $Z = 1051.8952$  con  $X_1 = 1$ , y se le asigna la etiqueta  $z^1$ . Para completar ésta etapa es necesario saber los valores de  $z$  para los enteros "cercaños".

Paso 3 Se calcula el mínimo de  $X_1$  hasta su segundo valor entero:  $X_1 = 0$  y se tiene que el valor de la función objetivo es:

$$Z = 822.845600$$

Con valores en las variables

$$\begin{aligned} X_1 &= 0 \\ X_2 &= 0 \\ X_3 &= 3.174731 \\ Y_1 &= 4.100841 \\ Y_2 &= 4.479172 \\ Y_3 &= Y_4 = 0 \end{aligned}$$

Estos tres valores de  $z$  se guardan en una lista en orden decreciente, para cada paso, así al inicio de cada Paso 2, se tomará un valor de la lista y al final de los pasos 3 y 4 se aumentarán 3 valores (inferiores). Los cálculos se terminan cuando todos los valores restantes de la lista son menores que un valor  $z$  asociado a una solución entera. En ésta etapa la lista está dada por:

Z	obtenida en el paso
1051.8952	$z^1$ 1
770.6983	1
822.8457	3

el valor más alto es 1051.8952 que se ha etiquetado como  $z^1$  (Paso 2). La solución y restricciones en este punto son  $Z^1 = 1051.8952$ ,  $X_1 = 1$ ,  $X_2 = 0$ ,  $X_3 = 4.4089$ ,  $Y_1 = 5.4838$ ,  $Y_2 = 1.4851$ ,  $Y_3 = 0$

Paso 4

4.1 Se calcula Min  $X_3$ . Se hace  $X_3 = 4$  entonces

$$Z = 990.638200$$

con valores en las variables:

$$\begin{aligned}
 X_1 &= 1 \\
 X_2 &= .276776 \\
 X_3 &= 4.000000 \\
 Y_1 &= 5.151334 \\
 Y_2 &= 1.050713 \\
 Y_3 &= Y_4 = 0
 \end{aligned}$$

4.2 Se calcula Máx  $X_3$ . Entonces  $X_3 = 5$ , y  $X_1 = 1$ . Sin embargo se obtiene una solución no factible.

Nota: Este resultado aparentemente anormal se produce porque max  $X_3$  es una función fallida de Z. Lo que implica que el punto para el cual  $X_1 = 1$ ,  $X_3 = 5$  cae en  $S$ , ya que el problema con tales restricciones no es factible.

La lista de valores de Z es:

Z	obtenida en el paso
1051.8952	Z <sup>1</sup> 1
770.6983	1
822.8457	3
990.6382	4

## ITERACION 2

Paso 2 El mejor valor no etiquetado es 990.6382, que será Z<sup>2</sup>

Paso 3 LLevamos a min  $X_3$  un paso mas adelante. Sea  $X_3 = 3$  y se obtiene el siguiente valor de la función objetivo:

$$Z = 840.659200$$

Con valores en las variables

$$\begin{aligned}
 X_1 &= 1 \\
 X_2 &= .948665 \\
 X_3 &= 3 \\
 Y_1 &= 4.336784 \\
 Y_2 &= Y_3 = 0 \\
 Y_4 &= .331045
 \end{aligned}$$

Como en  $X_3 = 3$  Z=840.6592, en éste punto la mejor solución no etiquetada es 990.6382 por lo cual la solución y restricciones son ahora: Z<sup>2</sup>=990.6382,  $X_1 = 1$ ,  $X_2 = 0.276$ ,  $X_3 = 4$ ,  $Y_1 = 5.1513$ ,  $Y_2 = 1.0507$

## Paso 4

4.1 Se calcula Mín  $X_2$  y se hace  $X_2 = 0$ , con valor en la función objetivo:

$$Z = 981.602300$$

Y valores de las variables:

$$\begin{aligned}
 X_1 &= 1 \\
 X_2 &= 0 \\
 X_3 &= 4 \\
 Y_1 &= 5.070157 \\
 Y_2 &= 1.692975 \\
 Y_3 &= 0 \\
 Y_4 &= 18.263330
 \end{aligned}$$

4.2 Se calcula Máx  $X_2$  y se hace  $X_2 = 1$  Aquí se agrega otro punto a  $S^*$ , puesto que se tiene una solución no factible.

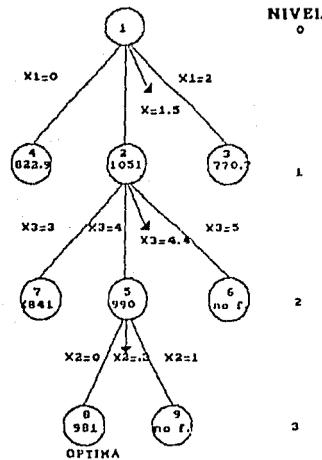
Entonces se tiene una solución en

$$\begin{aligned} Z &= 981.6023 \\ X_1 &= 1 \\ X_2 &= 0 \\ X_3 &= 4 \\ Y_1 &= 5.0702 \\ Y_2 &= 1.0930 \end{aligned}$$

Como éste valor de Z es el mayor de los restantes en la lista, ésta es la solución óptima

El árbol de expansión que ilustra los cálculos anteriores se presenta a continuación

FIGURA 2.2



Algunas características importantes de éste algoritmo son las siguientes

Cuando un subproblema en el nivel  $n$  es factible, se produce una solución al Problema Entero Mixto. Por lo tanto el algoritmo converge, ya que en el peor de los casos todos los nodos del árbol se enumeran explícitamente, y los nodos en el último renglón representan todas las combinaciones posibles de enteros.

Para reducir el esfuerzo computacional es probable que valga la pena introducir explícitamente la desigualdad  $cx + dy \geq z^0 + \epsilon$  para cada problema lineal PL, donde  $\epsilon$  es un número pequeño positivo. Haciendo ésto un problema lineal factible siempre indica un nodo activo y se tienen entonces, mejores soluciones del Problema Entero Mixto.

En ésto algoritmo se selecciona para ramificar el nodo con la solución lineal mayor. Con ésta estrategia se pretende encontrar rápidamente una buena solución entera mixta, sin embargo ésto procedimiento de selección puede requerir mucho almacenamiento computacional.

La variable que se selecciona para ramificar es aquella con componente más fraccionario

En este algoritmo, la mayoría (si no todos) los problemas lineales se resuelven completamente. Esto resulta costoso desde el punto de vista computacional, especialmente cuando la única información necesaria en un nodo puede ser el valor de la función objetivo.

El proceso continuo de ramificar y acotar, sugiere la designación de Ramificación y Acotamiento al algoritmo.

Algunos inconvenientes de este algoritmo son:

Es posible que un número grande de arcos se puedan originar del mismo nodo. El problema es que esto no se puede predecir por anticipado, lo que complica el árbol de búsqueda y puede proporcionar un aumento severo en la memoria de la computadora.

Si el algoritmo se toma literalmente, será necesario resolver un problema lineal para cada ramificación en aras de determinar la cota superior propia. Esto lo hace aparecer como un método costoso, especialmente porque algunos de los nodos resultantes probablemente nunca se ramifiquen.

La determinación de una cota inferior sucede sólo como resultado de resolver el programa lineal en los diferentes nodos. Lo que significa que no obstante la importancia de cálculos truncados, no hay un método sistemático que asegure una cota inferior "estrecha" en una etapa temprana del procedimiento.

### 2.3 EL ALGORITMO DE DAKIN

"A partir del momento en que escribí esa página vi claramente que mi búsqueda de la exactitud se bifurcaba en dos direcciones. Por una parte la reducción de los acontecimientos contingentes a esquemas abstractos con los que se pueden efectuar operaciones y demostrar teoremas; y por otra, el esfuerzo de las palabras por expresar con la mayor precisión posible el aspecto sensible de las cosas"

Italo Calvino. Seis Propuestas para el Próximo Milenio

Un algoritmo similar pero más sencillo de instrumentar que el algoritmo de Land y Doig, fue propuesto por Dakin. El algoritmo es aplicable a problemas tanto lineales como no lineales, sin embargo sólo se tiene experiencia computacional para el caso de programación lineal. Para explicar las bases de tal algoritmo considere nuevamente el problema:

- (2.1) Maximizar  $z = cx + dy$
- sujeto a
- (2.2)  $Ax + Dy = b$
- (2.3)  $x$  vector entero
- (2.4)  $x \geq 0, y \geq 0$

El problema (2.1), (2.2), y (2.4) (sin la condición de que  $x$  sea entero) se denomina el problema continuo. Una solución al problema continuo que también satisfaga (2.3) se llama una solución entera; en caso contrario se dice que es una solución no entera.

Suponga que existe un algoritmo (que llamaremos subalgoritmo) para encontrar soluciones a los problemas resultantes del problema continuo con la adición de cotas superiores o inferiores en alguna de las variables enteras. La existencia de tal algoritmo limita nuestro problema donde  $z$  es cóncava y (2.2) y (2.4) se especifican como una región convexa.

Se demostrará que si el subalgoritmo es finito, entendiendo por esto que resuelve cada subproblema en un tiempo finito, y las variables enteras acotadas, entonces se alcanzará una solución óptima entera después de un número finito de cálculos.

Suponga que  $x_j$  es una variable entera y  $k$  es un entero. Dado que el rango  $k < x_j < k + 1$  es inaceptable, podemos dividir todas las soluciones a las restricciones (2.2), (2.3) y (2.4) en dos grupos separados

- i) Soluciones en que  $x_j \leq k$  (2.6)

ii) Soluciones en que  $x_j \geq k + 1$  (2.7)

Al igual que el algoritmo de Land y Doig, en este algoritmo se comienza con una solución al problema continuo. Si esta solución es entera entonces es la solución del problema, de otra manera al menos una variable entera - digamos  $x_j$  - es no entera y toma un valor  $b_j$  en esta solución. Se divide  $b_j$  en dos partes, entera y fraccional  $[b_j] + f_j$  respectivamente, definida por:

$$b_j = [b_j] + f_j \quad (2.8)$$

donde  $[b_j]$  es un entero y  $0 < f_j < 1$ . Sustituimos

$$k = [b_j] \quad (2.9)$$

en (2.6) y (2.7). Aparentemente no se satisface ninguna de esas relaciones con la actual solución no entera, por lo que aumentamos esta restricción al problema continuo y resolvemos los problemas resultantes. Se repite el procedimiento para cada una de las dos soluciones obtenidas. La estructura lógica de este proceso es la de un árbol, como el que se muestra en la figura 2.3

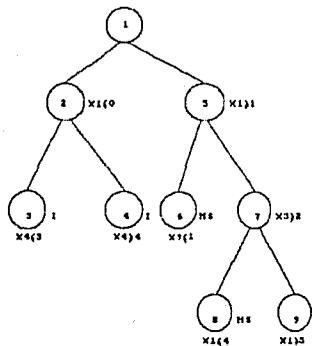


FIGURA 2.3

El árbol siempre terminará en uno de dos caminos, puede alcanzar una solución entera (denotada por "1" en la figura) o se puede encontrar que el conjunto actual de restricciones no tiene solución (denotado por "NS"). La solución al problema completo será la mejor solución entera alcanzada de esta manera.

El árbol -en general- no es único para un problema dado, ya que en cualquier etapa se tiene libertad de formar la siguiente restricción usando cualquier  $x_j$  que no sea entera; la selección de diferentes  $x_j$  producirá diferentes árboles. Como se podrá ver la selección puede tener un gran efecto en la cantidad de cálculos requeridos.

Nótese que sólo hay dos ramas y por lo tanto dos subárboles que salen de cada nodo -en contraste con el algoritmo de Land-Doig, donde puede salir cualquier número de ramas.

Se puede operar más de una restricción en una variable al mismo tiempo. Por ejemplo, en el nodo 9 de la figura 2.4 se fijan las restricciones  $x_1 \geq 1$  y  $x_1 \geq 5$  a la vez. El número de restricciones que pueden operar al mismo tiempo en una variable está limitado por el rango de valores que puede tomar la variable. Así, si la restricción original (2) limita a  $x_j$  a un rango de  $0 \leq x_j \leq v$  entonces el entero  $v$  acota a  $x_j$  en este rango y es consistente con  $x_j = k$ ; por ejemplo

$$x_j \geq 1, x_j \geq 2, \dots, x_j \geq k, x_j \leq k, x_j \leq k + 1, \dots, x_j \leq v - 1.$$

### ALGORITMO 2.2: Dakin

**PROPOSITO:** Resolver el problema de programación entera mixta

#### DESCRIPCION

**Paso 1** Se resuelve el problema entero por medio del método simplex de programación lineal. Si la solución es entera, pare, se ha encontrado la solución óptima.

**Paso 2** Se escoge arbitrariamente una variable entera  $x_j$  cuyo resultado en el Paso 1 sea fraccionario e igual a  $f_j$ .

**Paso 3.** Se resuelven dos nuevos problemas similares al problema original, uno con la restricción adicional  $x_j \leq k$ , y el otro con la restricción adicional  $x_j \geq k+1$ .

**Paso 4** De los programas lineales resueltos en el paso 3 se incluyen en el análisis a seguir sólo aquellos programas cuya solución (entera o fraccional) sea mejor a cualquiera de las soluciones enteras conocidas.

**Paso 5** Seleccione ese programa lineal que tenga el mínimo valor de la función objetivo. Si las variables enteras tienen valor entero, se ha encontrado la solución óptima. Si no, regrese al paso 2 con la estructura del problema lineal resuelto en este paso.

En los programas de cómputo que se han escrito para programación lineal, usando el método simplex como subalgoritmo y guardando el tablero completo, el requerimiento total de almacenamiento de datos es

$$\sum v_j + (m+2)(n+2) \text{ palabras}$$

donde  $m$  es el número de restricciones en el problema continuo (excluyendo restricciones de cota superior si se usa el procedimiento de variables acotadas),  $n$  es el número de variables no básicas,  $v_j$  es el rango de valores que puede tomar la variable  $x_j$ , de donde la suma de estos posibles valores acotan los requerimientos de almacenamiento.

### 2.4 Ejemplo Numérico

Suponga que se desea resolver el problema:

$$\begin{aligned} \text{MAX } Z &= 5 X_1 + 2 X_2 \\ \text{SUJETA A} \\ 2 X_1 + 2 X_2 + X_3 &= 9 \\ 3 X_1 + X_2 + X_4 &= 11 \end{aligned}$$

Usando el algoritmo de Dakin

#### ITERACION 1

**Paso 1** La solución óptima del problema continuo correspondiente tiene como valor de la función objetivo:

$$Z = 18.75$$

Y los valores de las variables son:

$$\begin{aligned} X_1 &= 3.25 \\ X_2 &= 1.25 \\ X_3 &= X_4 = 0 \end{aligned}$$

**Pasos 2-3** Se escoge arbitrariamente  $X_2 = 1.25$  y se resuelven dos problemas lineales distintos, uno con la restricción adicional  $X_2 \leq [1.25] = 1$  y el otro con la restricción adicional  $X_2 \geq [1.25] + 1 = 2$  Es decir:

Problema (1).- Consiste en resolver

$$\text{MAX } Z = 5 X_1 + 2 X_2$$

SUJETO A

$$\begin{aligned} 2 X_1 + 2 X_2 + X_3 &= 9 \\ 3 X_1 + X_2 + X_4 &= 11 \\ X_2 + X_5 &= 1 \end{aligned}$$

cuyo valor óptimo de la función objetivo es

$$Z = 18.6666700$$

Y los valores de las variables son:

$$\begin{aligned} X_1 &= 3.333333 \\ X_2 &= 1 \\ X_3 &= .333333 \\ X_4 &= X_5 = 0 \end{aligned}$$

Problema (2).- consistente en resolver

$$\text{MAX } Z = 5 X_1 + 2 X_2$$

SUJETO A

$$\begin{aligned} 2 X_1 + 2 X_2 + X_3 &= 9 \\ 3 X_1 + X_2 + X_4 &= 11 \\ X_2 - X_5 &= 2 \end{aligned}$$

cuyo valor óptimo de la función objetivo es igual a:

$$Z = 16.5$$

Y los valores de las variables son:

$$\begin{aligned} X_1 &= 2.5 \\ X_2 &= 2 \\ X_3 &= X_5 = 0 \\ X_4 &= 1.5 \end{aligned}$$

Paso 4. Como no ha habido ninguna solución entera en todo el proceso, se incluyen ambos problemas en el análisis

Paso 5 Como la mejor función objetivo hasta el momento corresponde a una solución no entera ( $Z = 18.67$ ), se regresa al paso 2 con el problema (1) como candidato

## ITERACION 2

Paso 2 En el Problema (1) se escoge arbitrariamente la variable  $X_1 = 3.33$  y se resuelven dos nuevos problemas. Uno que es igual al problema (1) más la restricción  $X_1[3.33] = 3$ . El otro que es igual al problema (1) más la restricción  $X_1[3.33] + 1 = 4$ . Específicamente:

Problema (3).- consistente en resolver

$$\text{MAX } Z = 5X_1 + 2X_2$$

SUJETO A

$$\begin{aligned} 2X_1 + 2X_2 + X_3 &= 9 \\ 3X_1 + X_2 + X_4 &= 11 \\ X_2 + X_5 &= 1 \\ X_1 + X_6 &= 3 \end{aligned}$$

Paso 3 Se obtienen las soluciones óptimas al problema lineal (3). El valor óptimo de la función objetivo es:

$$Z = 17$$

Y los valores de las variables son :

$$\begin{aligned} X_1 &= 3 \\ X_2 = X_3 = X_4 &= 1 \\ X_5 = X_6 &= 0 \end{aligned}$$

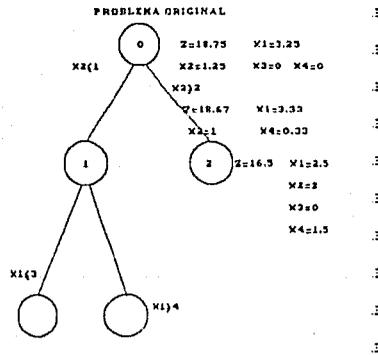
El problema (4) no tiene solución factible y no se incluye en el listado.

Paso 4 Por tener una solución entera se incluye en el análisis.

Paso 5 Por ser el mejor valor de la función objetivo y además, ser entero, es la solución óptima.

El árbol que se muestra el desarrollo del método se describe en la figura 2.4.

FIGURA 2.4



## 2.5 Comparación con el Método de Land-Doig

La cercana conexión entre el algoritmo descrito y el propuesto por Land-Doig (1960) es evidente. La diferencia básica es que el algoritmo de L-D fuerza a las variables enteras a tomar valores enteros en lugar de aplicar cotas, de tal forma que es necesario buscar sobre el rango de valores enteros para los cuales  $z$  es mayor que el valor que toma en la mejor solución entera conocida. Si todas las variables enteras se restringen a los valores 0 y 1 los dos métodos son iguales y la única contribución del método presentado consiste en proporcionar un procedimiento computacional conveniente para llevarlo a cabo.

La convergencia de ambos métodos depende de la selección de la variable entera que se maneja en cada etapa. Sin embargo hay indicios que hacen suponer una convergencia más rápida para este algoritmo en muchos casos, ya que hará una búsqueda sobre un rango más pequeño de valores para una variable entera que el algoritmo de Land-Doig, pues podemos detenernos tan pronto como se alcance una solución entera. Más aún, este método en algunos casos se saltará valores que alguna variable entera pueda tomar con el algoritmo de Land-Doig, y como se ha visto, se hace la ramificación en dos ramas y no en más como el método de L-D.

A continuación se desarrolla un ejemplo usando tanto el algoritmo de Land-Doig, como el de Dakin para observar claramente la diferencia entre ambos

#### EJEMPLO 4

Considere el problema entero

- (1)  $\text{MAX } Z = -4 X_1 - 5 X_2$
- SUJETO A
- (2)  $-X_1 - 4 X_2 \leq -5$
- (3)  $-3 X_1 - 2 X_2 \leq -7$
- (3')  $X_1, X_2 \text{ ENTEROS NO NEGATIVOS}$
- (4)  $X_1 \geq 0, X_2 \geq 0$

Aplique el algoritmo de Land-Doig:

Paso 0. La solución al problema definido por (1), (2), (3) y (4) tiene como valor óptimo para su función objetivo:

$$Z = -11.20$$

Cuyos valores en las variables son:  $X_1 = 1.80, X_2 = .80$

Paso 1. La cota superior de  $z$  y la solución actual al iniciar este paso es

$$\begin{aligned} Z^0 &= -11.2 \\ X_1 &= 1.8 \\ X_2 &= 0.8 \end{aligned}$$

Se escoge cualquiera de las dos variables, pues ambas están igual de lejanas al entero más próximo, sea  $X_1 = 1.8$  y se hace  $\max X_1$  y  $\min X_1$

1.1  $\min X_1$  implica  $X_1 = 1$ , cuyo valor óptimo de la función objetivo es igual a:  $Z = -14$ . Y los valores en las variables:  $X_1 = 1, X_2 = 2$

1.2  $\text{Máx } X_1$  implica  $X_1 = 2$ , con valor óptimo de la función objetivo igual a:  $Z = -11.75$ , y los valores de las variables son:  $X_1 = 2, X_2 = .75$

Paso 2 El mayor de los dos es  $Z = -11.75$  con  $X_1 = 2$ , y se le asigna la etiqueta  $Z^1$ . Para completar esta etapa es necesario saber los valores de  $z$  para los enteros que se pueden alcanzar "más cercanos"

Paso 3. Se continúa con  $\max X_1$  hasta su segundo valor entero:  $X_1 = 3$  y se tiene que el valor óptimo de la función objetivo es:  $Z = -14.5000000$ , y los valores de las variables:  $X_1 = 3, X_2 = .5$

Estos tres valores de  $Z$  se guardan en una lista y se tiene

Z	Z <sup>1</sup>	obtenida en el paso
-11.75	1	1
-14		1
-14.5		3

El valor más alto es -11.75 que se ha etiquetado como Z<sup>1</sup>. La solución y restricciones en éste punto son:

restricciones  $X_1 = 2$

solución	Z <sup>1</sup> = -11.75
	X <sub>1</sub> = 2
	X <sub>2</sub> = .75

Paso 4

4.1 Min X<sub>2</sub> implica X<sub>2</sub> = 0, no se tiene solución factible

4.2 Max X<sub>2</sub> implica X<sub>2</sub> = 1, cuyo valor óptimo de la función objetivo es: Z = -13, y los valores de las variables son : X<sub>1</sub> = 2, X<sub>2</sub> = 1

La figura 2.5 ilustra el árbol de búsqueda asociado a éste problema

FIGURA 2.5

Ahora desarrollamos el mismo ejemplo usando el algoritmo de Dakin

$$\begin{aligned} \text{MAX } Z &= -4 X_1 - 5 X_2 \\ \text{SUJETO A} \\ -X_1 - 4X_2 &\leq -5 \\ -3X_1 - 2 X_2 &\leq -7 \end{aligned}$$

#### ITERACION 1

Paso 1 La solución óptima del problema lineal correspondiente tiene como valor óptimo de la función objetivo: Z = -11.2 y con valores en las variables: X<sub>1</sub> = 1.8, X<sub>2</sub> = .80

Paso 2 Se escoge arbitrariamente X<sub>1</sub> = 1.8, y se resuelven los dos problemas lineales distintos uno con la restricción adicional X<sub>1</sub> ≤ [1.8] = 1 y el otro con la restricción adicional X<sub>1</sub> ≥ [1.8] + 1 = 2.

Paso 3 Se tienen los tableaus óptimos

Problema (1) Con X<sub>1</sub> ≤ 1, se tiene que el óptimo en la función objetivo es Z = -14. Y con valores en las variables: X<sub>1</sub> = 1, X<sub>2</sub> = 2

Problema (2) Con X<sub>1</sub> ≥ 2, se tiene que el óptimo en la función objetivo es Z = -11.75. Y los valores de las variables: X<sub>1</sub> = 2, X<sub>2</sub> = .75

Paso 4 Hay una solución entera sin embargo la mejor función objetivo corresponde a una solución no entera, y nos regresamos al Paso 2

#### ITERACION 2

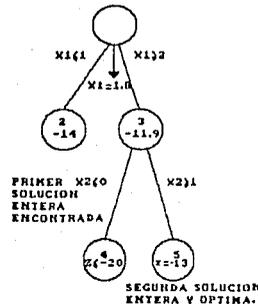
Paso 2 Escogemos X<sub>2</sub> = .75 y se resuelven los dos nuevos problemas. Uno igual al problema 1 más la restricción X<sub>2</sub> [0] = 0 y el otro que es igual al problema 1 más la restricción X<sub>2</sub> ≥ [0.75] + 1 = 1 es decir

Problema (3)  $X_2 \leq 0$ , se tiene que no hay solución factible

Problema (4) Con  $X_2 \geq 1$ , se tiene que el valor óptimo de la función objetivo es igual a:  $Z = -13$ . Cuyos valores en las variables son:  $X_1 = 2$ ,  $X_2 = 1$

La figura 2.6 ilustra el árbol de búsqueda correspondiente a éste problema.

FIGURA 2.6



## 2.6 ALGORITMO PARA LA SOLUCION DE PROBLEMAS DE PROGRAMACION ENTERA MIXTA DE DRIEBEEK

Este algoritmo, basado en el método de Land-Doig convierte a todas las variables enteras en binarias (cero-uno). Una vez realizado esto se sustituyen en el problema original y se encuentra la solución óptima utilizando programación lineal con ciertos ajustes, que se denominan costos penalizados. Dichos costos sirven para seleccionar la mejor variable entera que genera una bifurcación, y por lo tanto aceleran al método de R-A, es decir, lo hacen converger más rápidamente a la solución óptima. El algoritmo fue propuesto por Norman J. Driebeek en marzo de 1966 y posteriormente en marzo de 1968 Stanley Zionts propuso algunas modificaciones al algoritmo que permiten considerar una restricción y  $n + 1$  variables para cada variable entera en lugar de  $n + 1$  restricciones y  $2n + 1$  variables como se hacía originalmente. A continuación se presenta el algoritmo modificado por Zionts:

### ALGORITMO 2.3: Driebeek

PROPOSITO: Resolver el problema entero mixto de programación lineal.

#### DESCRIPCION

1. Se convierte a todas las variables enteras en binarias de la siguiente manera: Sea  $X_i$  una variable entera, entonces

$$X_i = \sum_{k=1}^n a_{ik}$$

donde  $a_{ik} = 0$  o  $1$ , para toda  $i, k$

$$\sum a_{ik} = 1.$$

Si  $a_{ik} = 1$ , se implica que la variable entera es  $X_i = k$  y se resuelve el problema no entero. Si se encuentra una solución óptima entera, parar.

2. Se calculan las penalizaciones de la siguiente manera:

a) Por cada variable  $a_{ik}$  no básica, el costo penalizado por alcanzar un valor  $a_{ik} = 1$ , es decir  $X_i = k$ , es precisamente el "precio sombra" de  $a_{ik}$  es decir el valor  $z_{ik} - c_{ik}$  del tableau correspondiente.

b) Por cada variable  $a_{ik}$  básica, el costo penalizado por incrementar  $a_{ik}$  al valor 1, es decir, hacer  $X_i = k$ , es el mínimo incremento en la función objetivo por hacer  $a_{ik} = 1$  (o sea, hay que usar análisis de sensibilidad al hacer el término  $X_{ik}$  correspondiente al vector básico  $a_{ik}$  igual a uno). El cambio mínimo en el valor de la función objetivo es la cantidad en la que cambia la función objetivo en la primer iteración del método dual simplex, al hacer el cambio correspondiente en el lado derecho del tableau. Este cambio es igual a

$$(X_{B_r} - 1) \text{ M} \max \{(z_j - c_j) / Y_{rj} \mid Y_{rj} \leq 0\}$$

donde  $X_{B_r}$  es el valor del vector básico  $a_{ik}$  y  $Y_{rj}$  son los valores del tableau correspondiente

3. De los costos penalizados calculados en a) se selecciona el mínimo. Si hay un empate, se escogen todos. Se combinan entonces con el costo mínimo penalizado calculado en b) y se selecciona aquella solución factible cuya suma de costos penalizados (los calculados en a) y en b)) sea mínima.

## CONVERGENCIA

La experiencia computacional con el algoritmo ha sido buena, pues el algoritmo busca esencialmente una reducción mínima en el valor de la función objetivo cuando se activan las restricciones enteras. Porque las soluciones prueba se inician en la base de las penalizaciones, y porque aquellos puntos que tienen menores penalizaciones se prueban primero, se espera que la mejor o al menos una muy buena solución entera, se encuentre en una de las primeras pruebas; de los problemas que se han resuelto se ha encontrado la solución óptima en las primeras pruebas. En un problema que contiene 21 variables cero-uno o  $(2)^{21}$  puntos, la solución óptima se encontró en la segunda prueba. Después de que una prueba se inicia, ésta sólo termina si:

- En una iteración dual, el valor de la función objetivo disminuye por debajo de la solución encontrada.
- Se encuentra una condición dual no acotado (primal no factible).
- Una solución entera válida, mejor que la que se encontró previamente.

El tiempo total de corrida del algoritmo depende principalmente del número de puntos de la red en el problema y menos del significado de las variables enteras. Cuando las variables se usan para representar costos iniciales, la solución se encuentra con mayor facilidad que cuando las variables enteras se usen para decidir entre parejas de variables muy similares. La mayoría de los problemas enteros se han resuelto de  $k$  a  $3k$  iteraciones, donde  $k$  es el número de iteraciones requeridas para alcanzar una solución continua.

**EJEMPLO 4**

Considere el problema entero-mixto

$$\text{Máx } Z = 5X_1 + 2X_2$$

sujeto a

$$\begin{aligned} 2X_1 + 2X_2 + X_3 &= 9 \\ 3X_1 + X_2 + X_4 &= 11 \end{aligned}$$

$$\begin{aligned} X_1 \geq 0, X_2 \geq 0, X_3 \geq 0, X_4 \geq 0 \\ X_1, X_2 \text{ enteros} \end{aligned}$$

Resolverlo usando el método de Dricbeck.

Empezamos por hacer:

$$\begin{aligned} X_1 &= 0\alpha_{10} + 1\alpha_{11} + 2\alpha_{12} + 3\alpha_{13} + \dots \\ X_2 &= 0\alpha_{20} + 1\alpha_{21} + 2\alpha_{22} + 3\alpha_{23} + \dots \end{aligned}$$

con  $\alpha_{ij} = 0$  o  $1$ ,

y  $\begin{aligned} \alpha_{10} + \alpha_{11} + \alpha_{12} + \dots &= 1, \\ \alpha_{20} + \alpha_{22} + \alpha_{23} + \dots &= 1 \end{aligned}$

Se tiene que como  $X_1$  puede alcanzar un valor máximo factible de 3, y  $X_2$  de 4, se toman los primeros cuatro términos de  $\alpha_{ij}$  y los primeros cinco términos de  $\alpha_{2j}$ , y el problema original se convierte en:

$$\text{MAX } 5\alpha_{11} + 10\alpha_{12} + 15\alpha_{13} + 2\alpha_{21} + 4\alpha_{22} + 6\alpha_{23} + 8\alpha_{24}$$

SUJETO A

$$\begin{aligned} 2\alpha_{11} + 4\alpha_{12} + 6\alpha_{13} + 2\alpha_{21} + 4\alpha_{22} + 6\alpha_{23} + 8\alpha_{24} + X_3 &= 9 \\ 3\alpha_{11} + 6\alpha_{12} + 9\alpha_{13} + \alpha_{21} + 2\alpha_{22} + 3\alpha_{23} + 4\alpha_{24} + X_4 &= 11 \\ \alpha_{11} + \alpha_{12} + \alpha_{13} + \alpha_{10} &= 1 \\ \alpha_{21} + \alpha_{22} + \alpha_{23} + \alpha_{24} + \alpha_{20} &= 1 \end{aligned}$$

EL TABLEAU

REN (BASIS)	$\alpha_{11}$	$\alpha_{12}$	$\alpha_{13}$	$\alpha_{21}$	$\alpha_{22}$
1 ART	-5.000	-10.000	-15.000	-2.000	-4.000
2 ART	2.000	4.000	6.000	2.000	4.000
3 ART	3.000	6.000	9.000	1.000	2.000
4 ART	1.000	1.000	1.000	.000	.000
5 ART	.000	.000	.000	1.000	1.000

I	REN	$\alpha_{23}$	$\alpha_{24}$	$X_3$	$X_4$	$\alpha_{10}$
1		-6.000	-8.000	.000	.000	.000
2		6.000	8.000	1.000	.000	.000
3		3.000	4.000	.000	1.000	.000
4		.000	.000	.000	.000	1.000
5		1.000	1.000	.000	.000	.000

I	REN	$\alpha_{20}$	$X_B$
1		.000	.000
2		.000	9.000
3		.000	11.000
4		.000	1.000
5		1.000	1.000

Después de dos iteraciones del método simplex, se obtiene la siguiente solución óptima no entera:

#### EL TABLEAU

REN (BASE)	$\alpha_{11}$	$\alpha_{12}$	$\alpha_{13}$	$\alpha_{21}$	$\alpha_{22}$
1 ART	6.000	3.000	.000	.000	.000
2 $\alpha_{24}$	-5.000	-2.250	.000	.250	.500
3 ART	-4.000	-2.000	.000	.000	.000
4 $\alpha_{13}$	1.000	1.000	1.000	.000	.000
5 ART	.500	.250	.000	.750	.500

I	REN	$\alpha_{23}$	$\alpha_{24}$	$X_3$	$X_4$	$\alpha_{10}$
1		.000	.000	1.000	.000	9.000
2		.750	1.000	.125	.000	-7.50
3		.000	.000	-5.000	1.000	-6.000
4		.000	.000	.000	.000	1.000
5		.250	.000	-.125	.000	.750

I	REN	$\alpha_{20}$	$X_B$
1		.000	18.000
2		.000	.375
3		.000	.500
4		.000	1.000
5		1.000	.625

El cálculo de costos penalizados al pasar de  $\alpha_{ik}$  distinto de 1 a  $\alpha_{ik} = 1$  es el siguiente:

$\alpha_{ij}$  no básicas

$\alpha_{ij}$	$X_i = j$	costo penalizado = $(z_j - c_j)$
$\alpha_{10}$	$X_1 = 0$	9
$\alpha_{11}$	$X_1 = 1$	6
$\alpha_{12}$	$X_1 = 2$	3
$\alpha_{21}$	$X_2 = 1$	0
$\alpha_{22}$	$X_2 = 2$	0
$\alpha_{23}$	$X_3 = 3$	

$\alpha_{ij}$  básicos

$$\alpha_{ij} \quad X_i = j \quad \text{costo penalizado} = (X_{B_i} - 1) \text{ Máx } \{z_j - c_j / Y_{ij} \mid Y_{ij} < 0\}$$

$$\alpha_{13} \quad X_1 = 3 \quad (1-1) \text{ Máx } \{\text{no hay candidatos}\} = 0$$

$$\alpha_{20} \quad X_2 = 0 \quad (5/8 - 1) \text{ Máx } \{1/(-1/8)\} = (-3/8)/(-8) = 3$$

$$\alpha_{24} \quad X_2 = 4 \quad (3/3 - 1) \text{ Máx } \{9/(-3/4), 6/(-1.2), 3/(-1/4)\} = 15/2$$

Los costos penalizados mínimos se obtienen cuando:

$X_1 = 3$  y  $X_2 = 1$  (solución factible) con costo penalizado total  $0 + 0 = 0$

$X_1 = 3$  y  $X_2 = 2$  (solución no factible) con costo penalizado total  $0 + 0 = 0$

$X_1 = 3$  y  $X_2 = 3$  (solución no factible) con costo penalizado total  $0 + 0 = 0$

Como las dos últimas soluciones no son factibles, pues violan alguna de las restricciones y la primera solución es factible, es también óptima por lo que la solución queda:

$$Z = 17, X_1 = 3, X_2 = 1$$

Como se podrá observar el algoritmo determina una cota superior (inferior) de la función objetivo en el caso de maximización (minimización), y por medio de los costos penalizados se ajusta con el costo penalizado mínimo, la solución óptima del problema entero. Este algoritmo puede dejar de iterar cuando la solución factible que se tiene se aproxima en un 80%-90% a la solución óptima.

Como veremos en el siguiente ejemplo, los costos penalizados sirven para seleccionar la mejor variable entera. Dada una variable básica  $X_r = X_{Br}$  cuyo resultado final debe ser entero, y por el momento todavía es fraccionario, se tiene que el costo penalizado de hacer  $X_r = \{X_{Br}\}$  (donde  $\{X\}$  es el número entero  $z$  más grande, menor o igual a  $X$  y  $\langle X \rangle$  el número entero  $z$  más pequeño, mayor o igual a  $X$ ) es:

$$(X_{Br} - \{X_{Br}\}) \text{ Mfn } \{(z_j - c_j) / Y_{rj} \mid Y_{rj} \geq 0\}$$

mientras que el costo penalizado de hacer  $X_r = \langle X_{Br} \rangle$  es

$$(1 - X_{Br} + \{X_{Br}\}) \text{ Mfn } \{(z_j - c_j) / -Y_{rj} \mid Y_{rj} \geq 0\}$$

Entonces, asociada a cada variable básica  $X_r$ , que aún es fraccionaria, pero que en la solución óptima debe ser entera, se tienen dos costos penalizados, uno asociado con el cambio  $X_r = \{X_{Br}\}$  y el otro con  $X_r = \langle X_{Br} \rangle$ . Para identificación, denótese al primer costo penalizado por  $[CP]_r$  y al segundo por  $\langle CP \rangle_r$ . La variable que se selecciona para ramificarse es aquella cuyo  $[CP]_r$  ó  $\langle CP \rangle_r$  es el máximo entre todas las variables básicas  $X_r$ , descartando por supuesto a las de holgura, que siendo aún fraccionarias, deben ser enteras en la solución óptima. Este proceso acelera al método, pues al elegir una variable básica con costo penalizado alto, se evita aumentar el número de iteraciones al eliminar implícitamente soluciones peores a las actuales.

## EJEMPLO 5

$$\text{MAX } Z = 5X_1 + 2X_2$$

SUJETO A

$$2X_1 + 2X_2 + X_3 = 9$$

$$3X_1 + X_2 + X_4 = 11$$

$$20X_1 - 10X_2 - X_5 = 51$$

$$X_i \geq 0 \text{ enteros, } i = 1, \dots, 5$$

La solución óptima del problema lineal asociado tiene como valor óptimo de la función objetivo

$$Z = 18.7500000$$

Y los valores de las variables son:

$$\begin{aligned} X_1 &= 3.25 \\ X_2 &= 1.25 \\ X_3 &= X_4 = 0 \\ X_5 &= 1.5 \end{aligned}$$

El cálculo de los costos penalizados  $[CP]_i$ ,  $\langle CP \rangle_i$ ,  $i = 1, 2$  para las variables básicas  $(x_1, x_2)$  es:

$$\begin{aligned} [CP]_1 &= (3.25-3) \text{ Mín } \{0/1, 1.5/.5\} = 0 \\ \langle CP \rangle_1 &= (1-3.25 + 3) \text{ Mín } \{-.25/.25\} = .75 \\ [CP]_2 &= (1.25-1) \text{ Mín } \{0/1, .25/.75\} = 0 \\ \langle CP \rangle_2 &= (1-1.25 + 1) \text{ Mín } \{1.5/(-.5)\} = 0.75(3) = 2.25 \end{aligned}$$

La variable que se selecciona es la  $X_2$  y las dos nuevas ramificaciones estarán dadas por las nuevas restricciones  $X_2 \leq 1$  y  $X_2 \geq 2$ .

1. Sea  $x_2 \leq 1$

La solución óptima del problema lineal asociado es

(BASIS)	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_B$
ART	.000	.000	.000	1.667	.000	.333	18.667
$X_3$	.000	.000	1.000	-.667	.000	-1.333	.333
$X_5$	.000	.000	.000	6.667	1.000	-16.667	5.667
$X_1$	1.000	.000	.000	.333	.000	-.333	3.333
$X_2$	.000	1.000	.000	.000	.000	1.000	1.000

Y el valor óptimo de la función objetivo es:

$$Z = 18.666670$$

Con valores en las variables

$$\begin{aligned} X_1 &= 3.333333 \\ X_2 &= 1 \\ X_3 &= .333333 \\ X_4 &= X_6 = 0 \\ X_5 &= 5.666667 \end{aligned}$$

Se calculan nuevamente los costos penalizados  $[CP]_i$ ,  $\langle CP \rangle_i$  para las variables básicas  $X_1$  y  $X_2$

$$\begin{aligned} [CP]_1 &= (3.333-3) \text{ Mín } \{0, 1.666/.333\} = 0 \\ \langle XP \rangle_1 &= (1-3.333 + 3) \text{ Mín } \{-.333/-.333\} = .66666 \\ [CP]_2 &= (1-1) \text{ Mín } \{0, .333/1\} = 0 \\ \langle CP \rangle_2 &= (1-1 + 1) \text{ Mín } \{ \text{No hay disponibles} \} = 0 \end{aligned}$$

Ahora vemos si hay solución para  $X_2 \geq 2$

2. Con  $X_2 \geq 2$  no se tiene solución factible, entonces escogemos del problema con  $X_2 \leq 1$  la variable, con el mayor costo de penalización que es  $X_1$ , entonces planteamos los dos nuevos problemas: con  $X_1 \leq 3$  y  $X_1 \geq 4$  y se tiene

3. Sea  $X_1 \leq 3$  cuyo valor óptimo de la función objetivo es :

$$Z = 16.8$$

Y los valores de las variables:

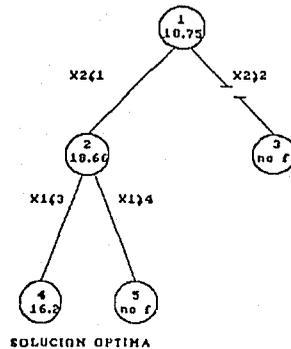
$$\begin{aligned} X_1 &= 3 \\ X_2 &= .90 \\ X_3 &= 1.20 \\ X_4 &= 1.10 \\ X_5 = X_7 &= 0 \\ X_6 &= .10 \end{aligned}$$

4. Con  $X_1 \geq 4$ , no hay solución factible

Como en éste problema no hay solución factible entonces se tendría que ramificar sobre el problema 3, sin embargo los costos penalizados asociados son iguales a cero por lo que la solución óptima está dada por  $Z = 17$ ,  $X_1 = 3$ ,  $X_2 = 1$

En la figura 2.7 se muestra el árbol de búsqueda asociado a éste problema

FIGURA 2.7



Estos costos penalizados tienen la grandísima ventaja que permiten en cualquier instante del algoritmo de bifurcación y acotación (aunque no se haya obtenido aún la solución óptima), calcular una cota superior en el caso de maximización y una cota inferior en el caso de minimización entre la diferencia del valor de la función objetivo de la mejor solución entera obtenida hasta ese momento y el valor óptimo de la función objetivo. Esta cota se calcula de

$$\text{Máx}_{r \in I} \left\{ \text{Mín}_{j \in N} \begin{cases} \frac{z_j - c_j (X_{Br} - [X_{Br}])}{Y_{rj}}, \text{ para } Y_{rj} > 0 \quad \forall r, j \\ \frac{z_j - c_j (1 - X_{Br} + [X_{Br}])}{-Y_{rj}}, \text{ para } Y_{rj} < 0 \end{cases} \right.$$

donde  $I$  es el conjunto de todos los vectores básicos en la iteración en cuestión (cuando se para el algoritmo) y  $N$  es el conjunto de todos los vectores no-básicos.

**EJEMPLO 6**

Considere el siguiente problema entero

$$\text{Mín } Z = 7x_1 + 3x_2 + 4x_3$$

**SUJETO A**

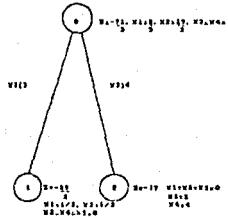
$$x_1 + 2x_2 + 3x_3 - x_4 = 8$$

$$3x_1 + x_2 + x_3 - x_5 = 5$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0, \text{ enteros}$$

Se supone que después de ramificarse como lo indica la figura 2.8 se ha obtenido una solución entera y se para el algoritmo, se desea conocer cuán alejada se encuentra esa solución de la solución óptima. El tableau asociado al nodo 2 es:

**FIGURA 2.8**



vectores en la base

Z	X <sub>1</sub>	X <sub>2</sub>	X <sub>3</sub>	X <sub>4</sub>	X <sub>5</sub>	X <sub>B</sub>
Z	1	5	1	0	0	4
X <sub>3</sub>	0	3	1	1	0	-1
X <sub>4</sub>	0	8	1	0	1	-3

Calculando los costos penalizados  $[CP]_r$  y  $\langle CP \rangle_r$ , para toda variable básica (sin incluir a las de holgura) ( $X_3$ ) se tiene

$$\begin{aligned} & \text{Máx}\{\text{Mín}\{(2-2)(5/3), (2-2)(1/1)\}, \text{Mín}\{(1-2+2)(4/(-1))\}\} \\ & = \text{Máx}\{\text{Mín}\{(0,0), \text{Mín}\{4\}\}\} \\ & = \text{Máx}\{0,4\} = 4 \end{aligned}$$

Lo que quiere decir que la función objetivo óptima no puede tener un valor mayor a  $-17 + 4 = -13$  (puesto que se está minimizando se obtiene una cota inferior, en el caso de maximización se obtiene una cota superior, dada por el valor de la función objetivo al momento de parar, menos el valor del costo penalizado). Es decir 13 es una cota inferior del valor óptimo de la función objetivo del problema entero. De hecho el valor óptimo es  $Z=15$  con  $X_2=5, X_4=2, X_1=X_3=X_5=0$ .

#### REFERENCIAS AL CAPITULO II.

1. BALINSKI, M.L. "Integer Programming: Methods, Uses, Computation", Man.Sci. 12, No.3 p. 253-313, 1965.
2. DAKIN, R.J. "A Tree-Search Algorithm for Mixed Integer Programming Problems", The Computer Journal 8, p.250-255, 1965.
3. DRIEBBEK, N.J. "An Algorithm for the Solution of Mixed Integer Programming Problems", Man.Sci.12, No. 7, 576-587, 1966.
4. GEOFFRION, A.M. y MARSTEN, R. E. "Integer Programming Algorithms: A Framework and State-of-the Art Survey", Management Science 18, no 9 p.465-491, 1972.
5. LAND, A.H. Y DOIG, A.G. "An Automatic Method of Solving Discrete Programming Problems", Econometría 28, 497-520, 1960.
6. PRAWDA, J. "Metodos y Modelos de Investigación de Operaciones" Limusa, 1979.
7. SALKIN, H.M. "Integer Programming", Addison-Wesley, 1974.
8. TAHA, H.A. "Integer Programming Theory, Applications and Computations", Academic Press, 1975.
9. ZIANTS, S. "On An Algorithm for the Solution of Mixed Integer Programming Problems", Man.Sci.15, 113 -116.

## CAPÍTULO III

### ANÁLISIS DE SENSIBILIDAD

"la razón, confundida en sí misma, veía enlace en la divisibilidad, la absorción del uno en el otro, lo simple y lo compuesto a la vez

Y exclamaba: ¡¡¡ qué duo verdadero parece éste solo canto!!!  
La razón del amor es que carece de razón, si así puede unirse lo separado"

William Shakespeare. El Fénix y la Tortola

En la mayoría de las aplicaciones prácticas, algunos de los datos del problema no se conocen con exactitud y se tienen que estimar lo mejor posible. En consecuencia es importante poder determinar la solución óptima del problema conforme se dispone de nuevas estimaciones, sin la costosa tarea de resolver el problema desde el principio. Asimismo, en las primeras etapas de la formulación del problema se pueden haber pasado por alto algunos factores. En tal caso es importante actualizar la solución presente de tal forma que se tomen en cuenta éstos factores. Por otra parte, en muchas situaciones las restricciones no son muy rígidas. Entonces es deseable que, sin tener que resolver el nuevo problema, se pueda examinar el efecto de relajar algunas de las restricciones sobre el valor del objetivo óptimo; éstos y otros temas relacionados constituyen el análisis de sensibilidad.

Este capítulo se desarrollará de la siguiente manera, en la sección 3.1 se dan algunas consideraciones básicas del problema de programación lineal y de análisis de sensibilidad, en la sección 3.2 se plantea el caso del problema lineal mixto y como se puede aplicar análisis de sensibilidad considerando que se resuelve el problema usando ramificación y acotamiento, en la sección 3.3 se presentan unos ejemplos que ilustran los conceptos que se manejan en la sección 3.2 y finalmente en la sección 3.4 se plantea un análisis de sensibilidad para el algoritmo de Driebeck.

#### 3.1 CONSIDERACIONES BASICAS DE PROGRAMACION LINEAL Y ANALISIS DE SENSIBILIDAD.

Considere el problema entero mixto

$$\begin{aligned} & \text{Maximizar } Z = \sum_{j=1}^n c_j x_j \\ & \text{Sujeto a} \\ (P) \quad & \sum_{j=1}^n a_{ij} x_j \leq b \\ & x_j \geq 0 \text{ para } j \in N = \{1, 2, \dots, n\} \\ & x_j \in \{0, 1\} \text{ para } j \in I \subset N \end{aligned}$$

donde  $a_j$  y  $b$  son vectores columna en  $\mathbb{R}^m$ . Por simplicidad, suponemos que  $c_j$  y los elementos de  $a_j$  y  $b$  son números racionales.

Una vez que se ha resuelto el problema (P), puede darse el caso de que uno o varios parámetros de la formulación original cambien, dando origen a un nuevo problema. Afortunadamente no es necesario volver a resolver el problema desde el principio, ya que existen métodos llamados análisis de sensibilidad, que permiten ahorrar muchas iteraciones, al resolver el nuevo problema partiendo de la solución óptima del problema original. Ahorrar iteraciones implica un ahorro considerable en los costos de utilización de una computadora y también un ahorro en tiempo para resolver el problema. El nuevo problema puede diferir del original en uno o varios de los siguientes cambios que pueden ocurrir simultáneamente:

- a. Cambio en el vector  $b$ , lo que significa cambio en la disponibilidad de los recursos.
- b. Cambios en el vector  $c$ , lo que significa cambios en los precios o costos unitarios.
- c. Cambios en la matriz  $A$ , lo que significa cambios en los coeficientes tecnológicos  $a_{ij}$ .
- d. Cambio en el vector  $X$ , lo que significa cambios en el número de actividades, cuyo nivel debe decidirse.
- e. Cambios en el número de restricciones del sistema lineal a optimizarse.

Con base en éstas consideraciones cabe hacernos las siguientes preguntas:

1. El lado derecho  $b$  de (P) es reemplazado por un nuevo lado derecho  $d$  ¿Cuál es una cota superior del nuevo valor de  $z$ ?
2. Se dispone de una nueva actividad  $a_j$  0-1 ¿Para cual cota superior en el valor de  $c_j$  la actividad  $a_j$  permanece en el nivel cero en la solución óptima de programación entera?

Suponemos que los valores de  $d$  y  $a_j$  no se conocen a priori, y empezamos por sentar las bases conceptuales para contestar estas preguntas.

Recientemente se ha desarrollado una teoría de dualidad para programación entera. Examinamos aquí algunas implicaciones económicas de ésta teoría, en particular la necesidad de usar funciones de precios en lugar de precios, y la posibilidad de efectuar análisis de sensibilidad de soluciones óptimas.

Los resultados que se desarrollan a continuación no sólo se cumplen para programación entera en general, también se cumplen para programación 0-1 y problemas con cotas superiores explícitas en las variables, si se incorporan las restricciones de acotamiento explícitamente en el conjunto de restricciones  $Ax \leq b$ .

Sea  $\Gamma(\mathbb{R}^m)$  el conjunto de funciones  $F: \mathbb{R}^m \rightarrow \mathbb{R}$ . Una función  $F \in \Gamma(\mathbb{R}^m)$  es superaditiva en  $\Delta \subseteq \mathbb{R}^m$  si

$$F(d_1) + F(d_2) \leq F(d_1 + d_2) \quad \forall d_1, d_2 \in \Delta$$

Sea  $\Gamma^m = \{F \in \Gamma(\mathbb{R}^m) \mid F \text{ es superaditiva y no decreciente en } \mathbb{R}^m \text{ y } F(0) = 0\}$ .

También sea  $L^m = \{F \in \Gamma(\mathbb{R}^m) \mid F(d) = \sum_{i=1}^m u_i d_i, u_i \geq 0 \ i = 1, \dots, m\}$  el conjunto de precios.

Considérese el problema dual del problema (P) sin la restricción (2), esto es, un problema entero puro.

$$(D) \quad \begin{array}{l} \min W = F(b) \\ \text{F} \\ \text{sujeto a} \\ F(a_j) \geq c_j \quad j = 1, \dots, n \\ F \in \Gamma^m \end{array}$$

Podemos adoptar la convención estándar de que  $Z = -\infty$ , si (P) no es factible, y  $Z = +\infty$  si (P) tiene soluciones factibles de valores arbitrariamente grandes, y una convención similar para (D).

**Teorema 1 (Dualidad Débil)**  $cx \leq F(b)$  para todos los vectores de producción factibles primales  $x$ , y todas las funciones de precios factibles  $F$ . Donde  $F(b)$  se asocia a la función objetivo primal en cada ramificación

**Corolario**

- i)  $Z \leq W$
- ii) (P) no es factible si  $W \rightarrow -\infty$
- iii) (D) no es factible si  $Z \rightarrow +\infty$

**Teorema 2 (Dualidad Fuerte)**

a) Si (P) o (D) tienen un valor óptimo finito entonces existe un vector de producción óptimo factible  $x^*$ , y una función de precios  $F^*$  óptima factible dual para la cual  $Z = cx^* = F^*(b) = W$

b) Si (P) no es factible entonces (D) no es factible o  $W \rightarrow -\infty$

c) Si (D) no es factible, entonces (P) no es factible o  $Z \rightarrow +\infty$

**Teorema 3 (Holgura Complementaria)** Sean  $x^*$ ,  $F^*$  una pareja de soluciones óptimas para (P) y (D) respectivamente. Sean  $s^* = b - \sum_{j=1}^n a_j x_j^*$  y  $v_j^* = c_j - F^*(a_j)$   $j = 1, \dots, n$  entonces

i)  $v_j^* \leq 0$ ,  $j = 1, \dots, n$  o  $c_j - F^*(a_j) \leq 0$  o  $c_j \leq F^*(a_j)$  (ningún proceso en los coeficientes tecnológicos produce un beneficio positivo).

ii) Si  $x_j^* > 0$ , entonces  $v_j^* = 0$  o  $c_j = F^*(a_j)$  (todo proceso en uso produce beneficio cero).

iii)  $F^*$  es no decreciente y superaditiva (recursos mancomunados nunca pueden ser perjudiciales).

iv)  $F^*(b) = F^*(\sum_{j=1}^n a_j x_j^*)$  y  $F^*(s^*) = 0$  (el precio del actual vector de consumo no cambia por añadir recursos nuevos).

La condición de superaditividad en la función de precios conlleva a una relación obvia para el concepto de utilidad creciente en escala como

$$\sum_{j=1}^n F(a_j) x_j \leq F(\sum_{j=1}^n a_j x_j) \text{ para } x \geq 0 \text{ entero}$$

Por otro lado vemos que  $F$  no siempre es homogénea. Sin embargo debido a la superaditividad de  $F^*$ ,

$$F^*(\sum_{j=1}^n a_j y_j) = \sum_{j=1}^n c_j y_j^* \quad \forall \quad 0 \leq y_j^* \leq x_j^* \quad y_j^* \text{ entero, } j = 1, \dots, n$$

entonces la función de precios óptima es homogénea al menos en los niveles de actividad óptima. De cualquier modo, es posible que  $F^*(a_j(x_j + 1)) > c_j(x_j + 1)$ , y se obtiene información útil en el nivel de una variable activa primal de la función de precios óptima  $F^*$  (lo que es imposible en el caso lineal).

Consideramos aquí los posibles cambios en el problema primal (P) una vez que se han encontrado una pareja de soluciones óptimas  $(x^*, F^*)$  con  $F^* \in \Gamma^m$ . Sean  $Z'$ ,  $x'$  el valor/solución óptimos respectivamente del nuevo problema

Consideramos brevemente cuatro casos:

1. Cambios en los recursos  $b \rightarrow b'$ :

(a)  $F^*$  permanece dual factible, entonces  $Z' \leq F^*(b)$

(b) Sea  $Y^* = \{y \mid \bar{F}^*(\sum_{j=1}^n a_j y_j) = \sum_{j=1}^n c_j y_j\}$ . Si  $F^*$  permanece óptima, entonces  $x^*$  está en  $Y^*$ .

2. Cambios en la función objetivo  $c \rightarrow c'$

(a)  $x^*$  permanece primal factible, entonces  $Z' \geq c' x^*$

(b) Si  $c'_j \leq F^*(a_j)$  para toda  $j$ , entonces  $F^*$  permanece dual factible, y  $Z' \leq F^*(b)$

(c) Si  $c'_j \leq F^*(a_j)$  cuando  $x^*_j = 0$  y  $c'_j = c_j$  cuando  $x^*_j > 0$  entonces  $x^*$  permanece óptima.

3. Introducción de una nueva actividad  $(c, a)$ :

(a)  $x^*$  permanece primal factible, entonces  $Z' \geq Z(b)$   $x^*$  permanece óptima si  $F^*(a) \geq c$

4. Introducción de una nueva restricción  $\sum_{j=1}^n h_j x_j \leq h_0$ :

(a) Si  $x^*$  permanece factible,  $x^*$  es óptima

(b) La función de precios  $F: \mathbb{R} \rightarrow \mathbb{R}$  definida por  $F(d, d_{m+1}) = F^*(d)$  es dual factible para el nuevo problema, entonces  $Z' \leq Z = F(b, h_0)$ .

Las observaciones anteriores permiten un análisis postóptimo si se conoce la función de precios duales óptima. Se obtiene una interpretación alternativa si suponemos que cada decisión en el valor de un nivel de actividad  $x_j$  se toma independiente con respecto al centro de decisión. Entonces una función de precios  $F^*$  proporciona una pauta para el analista, similar al papel jugado por los precios en el modelo lineal.

### 3.2 RESULTADOS PARA RAMIFICACION Y ACOTAMIENTO.

La solución para el problema (P) usando ramificación y acotamiento construye un árbol de problemas lineales. Asociados con un nodo  $t$  del árbol están el problema lineal y los precios duales factibles  $\Pi^t$  y  $\mu^t$ :

$Q^t(b)$ :  $\max Z^t(b) = cx$   
 sujeto a

$$\sum_{j=1}^n a_{ij}x_j \leq b \quad (\pi^t)$$

$$\begin{aligned} x_j &\geq 0 \text{ para } j \in N & (\theta_j^t) \\ L_j^t \leq x_j \leq U_j^t &\text{ para } j \in I & (\mu_j^t) \end{aligned}$$

donde

$$\begin{aligned} L_j^t = U_j^t = 0 &\text{ para } j \in F_0^t \\ L_j^t = U_j^t = 1 &\text{ para } j \in F_1^t \\ L_j^t = 0, U_j^t = 1 &\text{ para } j \in I - F_0^t - F_1^t \end{aligned}$$

y  $F_0^t, F_1^t$  son los conjuntos de variables en  $I$  fijados en  $0$  y  $1$ , respectivamente. Usamos el mismo símbolo,  $\mu_j^t$  para denotar los precios duales de las restricciones  $x_j \geq L_j^t$  y  $x_j \leq U_j^t$  sin ambigüedad, porque a lo más una de estas restricciones puede cumplirse con cada  $j$ . Si  $x_j \geq L_j^t$  se cumple, entonces  $\mu_j^t \leq 0$ , de otra manera  $\mu_j^t \geq 0$ .

Si  $Q^t(b)$  no tiene solución factible, sea  $\Pi^t, \mu^t$  el vector dual asociado con el objetivo de la Fase I de maximizar el negativo de la suma de infactibilidades.

Sea  $P^t(b)$  el programa mixto entero obtenido del programa lineal  $Q^t(b)$  reemplazando las restricciones  $0 \leq x_j \leq 1$  por  $x_j \in \{0, 1\}$  para  $j \in I - F_0^t - F_1^t$ . Sea  $Z^t(b)$  el valor de la función objetivo óptima para  $P^t(b)$ . Más aún sean  $C^t(d)$  y  $B^t(d)$  las funciones de acotamiento superior en los valores de las funciones objetivo de los problemas  $Q^t(d)$  y  $P^t(d)$  respectivamente (éstas funciones no son otras que las funciones de precios duales  $F(d)$  asociadas a los problemas  $Q^t(d)$  y  $P^t(d)$  respectivamente). Suponiendo que el nodo  $t$  es la raíz del árbol, nuestro principal propósito es obtener  $B^t(d)$ . Proponemos la siguiente formulación para  $C^t(d)$  y  $B^t(d)$ :

Suponemos que  $Q^t(b)$  se ha resuelto óptimamente en la fase I o la fase II, como sea apropiado (De hecho, una solución dual factible es satisfactoria).

Para cada nodo  $t$  definimos

$$C^t(d) = \Pi^t d + \sum_{j \in I - F_0 - F_1} \max\{0, \mu_j^t\} + \sum_{j \in F_1} \mu_j^t$$

El operador  $\max$  toma en cuenta el hecho de que si la restricción  $x_j \geq 0$  se cumple, entonces  $\mu_j^t \leq 0$  y el producto  $\mu_j^t$  y el lado derecho de  $Q^t(d)$  no contribuye a  $C^t(d)$ .

Para cada nodo terminal con una solución factible para  $Q^t(b)$ , se define  $B^t(d) = C^t(d)$ . Para un nodo terminal con solución no factible se define

$$B^t(d) = \begin{cases} -\infty & \text{si } C^t(d) < 0 \\ +\infty & \text{si } C^t(d) \geq 0 \end{cases}$$

Supongamos que cualquier nodo no terminal  $t$  tiene dos hijos  $L(t)$  y  $R(t)$ . Para éste nodo se define

$$B^t(d) = \min\{C^t(d), \max\{B^{L(t)}(d), B^{R(t)}(d)\}\}$$

#### Teorema 4

$Z \leq B^1(d)$  cuando el problema (P) se resuelve con el lado derecho de  $d$ .

Prueba. Demostraremos que  $B^1(d)$  proporciona una cota superior al valor óptimo de la solución de  $P^1(d)$  para cualquier nodo  $t$ , y cualquier  $d$ .

Si  $t$  es un nodo terminal, entonces la veracidad de esta afirmación se sigue inmediatamente de la dualidad del programa lineal y la definición de  $B^1(d)$  para los nodos terminales.

Ahora suponga que  $t$  no es un nodo terminal, pero que  $B^{L(t)}(d)$  y  $B^{R(t)}(d)$  son funciones que acotan superiormente los problemas  $P^{L(t)}$  y  $P^{R(t)}$  respectivamente. Nuevamente de la programación lineal,  $Z^1(d) \leq Z^{L(t)}(d) \leq C^L(d)$ . De la estructura del árbol de ramificación y acotamiento

$$Z^1(d) = \max\{Z^{L(t)}(d), Z^{R(t)}(d)\} \leq \max\{B^{L(t)}(d), B^{R(t)}(d)\}$$

Concluimos entonces que

$$Z^1(d) \leq \min\{C^1(d), \max\{B^{L(t)}(d), B^{R(t)}(d)\}\},$$

y por lo tanto  $Z^1(d) \leq B^1(d)$ . Por inducción desde la raíz, el nodo 1 del árbol,  $Z \leq B^1(d)$  y el teorema se cumple

### Teorema 5

Si  $(P)$  se ha resuelto óptimamente y una nueva columna 0-1  $a_j$  se aumenta al problema  $(P)$  para dar el problema  $P^\wedge$ , entonces existe una solución óptima para  $P^\wedge$  con  $x_j = 0$  si  $c_j \leq B^1(b) - B^1(b - a_j)$

**Prueba** Suponga que la nueva columna  $a_j$  es una solución óptima en el nodo 1. Podemos encontrar la solución óptima completa fijando  $x_j$  en 1 y resolviendo  $(P)$  con un lado derecho de  $b - a_j$ . Sabemos que el valor de la solución óptima a este problema es menor o igual a  $B^1(b - a_j)$ .

Se tiene que  $B^1(b)$  es compacto, esto es, la solución óptima para  $(P)$  tiene una solución igual a  $B^1(b)$ . Así una solución con  $a_j$  en el nodo 1 puede tener una solución no mayor que  $c_j + B^1(b - a_j)$ . Necesitamos considerar soluciones que tengan  $x_j = 1$  solo si  $c_j + B^1(b - a_j) > B^1(b)$ . Así, concluimos que la solución al problema original  $(P)$  permanece óptima si  $c_j \leq B^1(b) - B^1(b - a_j)$ .

### 3.3 EJEMPLO

Considere el problema:

Maximizar

$$77x_1 + 6x_2 + 3x_3 + 6x_4 + 33x_5 + 13x_6 + 110x_7 + 21x_8 + 47x_9$$

Sujeto a

$$774x_1 + 76x_2 + 22x_3 + 42x_4 + 21x_5 + 760x_6 + 818x_7 + 62x_8 + 785x_9 \leq 1500 \quad (1)$$

$$67x_1 + 27x_2 + 794x_3 + 53x_4 + 234x_5 + 32x_6 + 792x_7 + 97x_8 + 435x_9 \leq 1500 \quad (2)$$

$$x_j = 0 \text{ o } 1 \text{ para } j = 1, 2, \dots, 9 \quad (3)$$

Se desea resolver este problema y establecer cotas que establezcan soluciones considerando posibles cambios futuros en el problema, y evitar con esto resolver nuevamente todo el problema. Como es un problema entero, vamos a usar la técnica de ramificación y acotamiento para resolverlo.

Si (3) se relaja a  $0 \leq x_j \leq 1$  para  $j = 1, \dots, 9$  entonces el problema lineal resultante  $Q^1(b)$  tiene como valor óptimo de la función objetivo:

$$Z^{*1} = 225.694600$$

Y los siguientes valores en las variables con los respectivos costos reducidos:

VARIABLE	VALOR	COSTO REDUCIDO
X <sub>1</sub>	.709742	$\mu_1 = 0$
X <sub>2</sub>	0	$\mu_2 = 1.576519$
X <sub>3</sub>	.348170	$\mu_3 = 0$
X <sub>4</sub>	1	$\mu_4 = -1.771140$
X <sub>5</sub>	1	$\mu_5 = -30.673020$
X <sub>6</sub>	0	$\mu_6 = 62.572620$
X <sub>7</sub>	1	$\mu_7 = -27.884000$
X <sub>8</sub>	1	$\mu_8 = -14.738180$
X <sub>9</sub>	0	$\mu_9 = 31.470290$

REN	HOLGURA	PRECIO DUAL
2)	0	$\Pi_1 = .099395$
3)	0	$\Pi_2 = .001024$

Donde para el nodo 1 se tiene la siguiente función de cota superior para la función objetivo del problema Q<sup>1</sup>.

$$C^1(d) = \sum_{j \in F_1} \mu_j^1 + \Pi_1 d_1 + \Pi_2 d_2 = 75.06634 + 0.099395 d_1 + 0.001024 d_2$$

Ramificamos el problema fijando la variable X<sub>1</sub> = 1 y se obtiene el siguiente resultado:

#### VALOR DE LA FUNCION OBJETIVO

$$Z = 217.8442$$

Y los siguientes valores en las variables así como los costos reducidos y los precios duales:

VARIABLE	VALOR	COSTO REDUCIDO
X <sub>1</sub>	1	$\mu_1 = 0$
X <sub>2</sub>	0	$\mu_2 = 4.217274$
X <sub>3</sub>	.604508	$\mu_3 = 0$
X <sub>4</sub>	1	$\mu_4 = -.351415$
X <sub>5</sub>	1	$\mu_5 = -30.164550$
X <sub>6</sub>	0	$\mu_6 = 89.162630$
X <sub>7</sub>	.718461	$\mu_7 = 0$
X <sub>8</sub>	1	$\mu_8 = -12.660600$
X <sub>9</sub>	0	$\mu_9 = 58.544860$

REN.	HOLGURA	PRECIO DUAL
2)	0	$\Pi_1 = .134422$
3)	0	$\Pi_2 = .000054$
4)	0	-27.046420

Para éste problema P<sup>2</sup>(d) se tiene la siguiente cota superior para la función objetivo Z<sup>2</sup>:

$$C^2(d) = 16.13018 + 0.134422d_1 + 0.000054d_2$$

Si se fija la variable X<sub>1</sub> = 0 se tiene la siguiente solución al problema P<sup>17</sup>(b):

VALOR DE LA FUNCION OBJETIVO

$$Z = 204.8750$$

Y los siguientes valores en las variables así como los costos reducidos y los precios duales:

VARIABLE	VALOR	COSTO REDUCIDO
X <sub>1</sub>	0	$\mu_1 = 0$
X <sub>2</sub>	1	$\mu_2 = -1.492977$
X <sub>3</sub>	.045355	$\mu_3 = 0$
X <sub>4</sub>	1	$\mu_4 = -3.421366$
X <sub>5</sub>	1	$\mu_5 = -31.264050$
X <sub>6</sub>	0	$\mu_6 = 31.665570$
X <sub>7</sub>	1	$\mu_7 = -60.295140$
X <sub>8</sub>	1	$\mu_8 = -17.153060$
X <sub>9</sub>	.611468	$\mu_9 = 0$

REN.	HOLGURA	PRECIO DUAL
2)	0	$\Pi_1 = .058680$
3)	0	$\Pi_2 = .002152$
4)	0	31.437580

Para éste problema se tiene la siguiente cota superior:

$$C^{17}(d) = 113.61628 + 0.05868d_1 + 0.002151 d_2$$

La solución entera del problema se dá a continuación:

VALOR DE LA FUNCION OBJETIVO

$$Z = 176.00$$

VARIABLE	VALOR	COSTO REDUCIDO
X <sub>1</sub>	0	-77
X <sub>2</sub>	1	-6
X <sub>3</sub>	0	-3
X <sub>4</sub>	1	-6
X <sub>5</sub>	1	-33
X <sub>6</sub>	0	-13
X <sub>7</sub>	1	-110
X <sub>8</sub>	1	-21
X <sub>9</sub>	0	-47

REN	HOLGURA	PRECIO DUAL
2)	481	0
3)	302	0

RANGOS EN QUE LA BASE NO CAMBIA:

VARIABLE	COEF ACTUAL	RANGOS DE COEF. OBJ.	
		INCREMENTO PERMITIDO	DECREMENTO PERMITIDO
X <sub>1</sub>	77.00	-77.00	INFINITO
X <sub>2</sub>	6.00	INFINITO	6.00
X <sub>3</sub>	3.00	-3.00	INFINITO
X <sub>4</sub>	6.00	INFINITO	6.00
X <sub>5</sub>	33.00	INFINITO	33.00
X <sub>6</sub>	13.00	-13.00	INFINITO
X <sub>7</sub>	110.00	INFINITO	110.00
X <sub>8</sub>	21.00	INFINITO	21.00
X <sub>9</sub>	47.00	-47.00	INFINITO

RANGOS DEL LADO DERECHO

REN	L.D. ACTUAL	INCREMENTO PERMITIDO	DECREMENTO PERMITIDO
2	1500.00	INFINITO	481.00
3	1500.00	INFINITO	302.00

La solución entera es

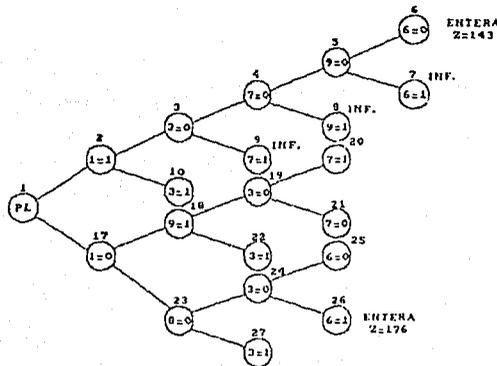
$$Z = 176$$

$$x_1 = x_3 = x_6 = x_9 = 0$$

$$x_2 = x_4 = x_5 = x_7 = x_8 = 1$$

Un árbol de enumeración de ramificación y acotamiento para éste problema aparece en la figura 3.1. El índice de identificación de un nodo aparece arriba del nodo. Dentro del nodo se indica que variable se ha fijado en que valor, por ejemplo 7=0 en el nodo 4 indica que x<sub>7</sub> es igual a cero en el nodo 4.

FIGURA 3.1



Las funciones de cota superior  $C^l(d)$  asociadas con el problema lineal en cada nodo en los dos primeros niveles del árbol se listan a continuación. Esta información indica la información adicional que se debe tener en cada nodo de la figura 3.1

Función de Nodos

- 1  $C^1(d) = 75.066 + 0.099395 d_1 + 0.001024 d_2$
- 2  $C^2(d) = 16.13018 + 0.134422 d_1 + 0.000054 d_2$
- 17  $C^{17}(d) = 113.61628 + 0.05868 d_1 + 0.002152 d_2$

Una muestra de análisis de sensibilidad aparece en la Tabla 3.1 para las combinaciones de los nueve diferentes valores de  $d_1$  y  $d_2$ . Aparecen tres números en cada celda. El número mas grande es la cota que usa la función de precios lineales duales,  $75.066 + 0.099395 d_1 + 0.001024 d_2$  asociada con el problema lineal relajado del problema original. El número de en medio es la cota usando la función de precios lineal a trazos  $B^l(d)$  del árbol estructurado y que se describió anteriormente. El número menor es la solución  $Z$  al problema entero cuando se resuelve con  $d_1$  y  $d_2$  asociadas con la celda. Para este ejemplo se puede concluir que cualitativamente la función de precios del problema entero da un análisis de sensibilidad similar al que se acostumbra en problemas lineales. Esto es, para cambios pequeños para el lado derecho, la cota es exacta.

#### PERFECCIONANDO $C^l(d)$

El siguiente mejoramiento para  $C^l(d)$  usa una sugerencia de Jeroslow (1983), en cuanto a que la información en un nodo puede proporcionar cotas en otro nodo. Sea  $A$  el conjunto de todos los nodos en el árbol. Cada problema lineal relajado  $Q^s(b)$  para  $s \in A$  es idéntico a  $Q^l(d)$  excepto que el lado derecho puede diferir en  $d$  y/o en las restricciones de acotamiento de las  $x_j$ 's. Así, los precios óptimos duales de  $Q^s(b)$  se pueden aplicar al lado derecho de  $Q^l(d)$  para dar una cota superior en  $Z^l(d)$ . En el nodo  $s$ , las restricciones de acotamiento para  $x_j$  son de la forma  $L_j^s \leq x_j \leq U_j^s$ . Si  $\mu_j^s < 0$ , entonces se cumple  $L_j^s \leq x_j$ . Así, la cota en el nodo  $s$  basada en el nodo del precio dual  $s$  es:

$$\Pi^s d + \sum_{j \in I} (\min\{0, \mu_j^s\} L_j^s + \max\{0, \mu_j^s\} U_j^s)$$

Tomando el mínimo sobre todos los nodos y reescribiendo tenemos la  $C^l(d)$  perfeccionada:

$$C^l(d) = \min_{s \in A} \{ \Pi^s d + \sum_{j \in I^1} \mu_j^s + \sum_{j \in I^0, I^1} \max\{0, \mu_j^s\} \}$$

la expresión en llaves debe reemplazarse por  $+\infty$  para nodos terminales no factibles, dependiendo de que la expresión sea o no mayor o igual a cero, por ejemplo

En segundo lugar, la función de precios del problema entero es similar a los precios duales en el problema lineal en donde los datos necesarios para construir la función se generan naturalmente como parte del proceso de solución.

**TABLA 3.1**  
**EJEMPLO DE ANALISIS DE SENSIBILIDAD**

d <sub>1</sub>	d <sub>2</sub>								
	1000	1300	1400	1450	1500	1550	1600	1700	2000
1000	175	175	175	175	176	176	176	176	176
	174	175	175	175	175	175	175	175	175
	143	170	170	170	170	170	170	170	173
1300	205	205	205	205	205	205	205	206	206
	176	176	176	176	176	176	176	190	190
	143	176	176	176	176	176	176	176	179
1400	215	215	215	215	215	215	215	215	216
	176	176	176	176	176	176	176	176	204
	143	176	176	176	176	176	176	176	179
1450	220	220	220	220	220	220	220	220	221
	176	176	176	176	176	176	176	206	211
	143	176	176	176	176	176	176	176	179
1500	225	225	225	225	225	225	225	225	226
	176	176	176	176	176	176	176	207	217
	143	176	176	176	176	176	176	176	179
1550	230	230	230	230	230	230	230	230	231
	176	176	176	176	176	176	176	207	224
	143	176	176	176	176	176	176	176	179
1600	235	235	235	235	235	235	235	235	236
	231	231	231	231	231	231	231	231	231
	187	187	187	187	187	187	187	187	187
1700	245	245	245	245	245	245	245	245	246
	244	244	244	244	244	244	244	244	244
	208	241	241	241	241	241	241	241	244
2000	274	275	275	275	275	275	275	275	275
	274	275	275	275	275	275	275	275	275
	214	253	253	253	253	253	253	253	253

**EJEMPLO 3.2**

Maximizar  $Z = 7X_1 + 2X_2$

Sujeto a

$-X_1 + 2X_2 \leq 4$

$5X_1 + X_2 \leq 20$

$-2X_1 - 2X_2 \leq -7$

$X_1, X_2 \geq 0$  enteras

Se obtiene el siguiente valor óptimo para la función objetivo:

$$Z = 30.1818200$$

Y los siguientes valores para las variables así como los costos reducidos:

VARIABLE	VALOR	COSTO REDUCIDO
$X_1$	3.272727	0
$X_2$	3.636364	0
REGLON	HOLGURA	PRECIO DUAL
2)	0	$\Pi_1 = .272727$
3)	0	$\Pi_2 = 1.454545$
4)	6.818182	$\Pi_3 = 0$

Agregamos la restricción  $X_1 \geq 4$  y se tiene que el valor óptimo de la función objetivo está dado por:

$$Z = 28.00$$

Y valores en las variables así como los costos reducidos:

VARIABLE	VALOR	COSTO REDUCIDO
$X_1$	4	$\mu = 3$
$X_2$	0	0
REGLON	HOLGURA	PRECIO DUAL
2)	8	$\Pi_1 = 0$
3)	0	$\Pi_2 = 2$
4)	1.00	$\Pi_3 = 0$

Ahora agregamos la restricción  $X_1 \leq 3$  y se tiene que el valor óptimo de la función objetivo está dado por:

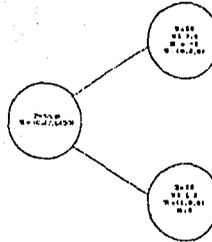
$$Z = 28.00$$

Y valores en las variables así como los costos reducidos:

VARIABLE	VALOR	COSTO REDUCIDO
$X_1$	3	0
$X_2$	3.5	0
REGLON	HOLGURA	PRECIO DUAL
2)	0	$\Pi_1 = 1$
3)	1.5	$\Pi_2 = 0$
4)	6	$\Pi_3 = 0$
5)	0	$\mu = 8$

El árbol de enumeración asociado es:

FIGURA 3.2



Para los dos nodos terminales tenemos

$$B^2(d) = -12 + 2d_2$$

$$B^3(d) = 24 + d_1$$

$$y B^1(d) = \min \{0.2727 d_1 + 1.4545d_2, \max\{B^2(d), B^3(d)\}\}$$

Ahora suponga que queremos examinar el efecto de cambiar el lado derecho de la segunda restricción de 20 a 26. Evaluando tenemos

$$\text{MAX } Z = 7 X_1 + 2 X_2$$

SUJETO A

$$- X_1 + 2X_2 \leq 4$$

$$-2X_1 - 2X_2 \leq -7$$

$$5X_1 + X_2 \leq 26$$

VALOR DE LA FUNCION OBJETIVO

$$Z = 38.9090900$$

Y valores en las variables:

VARIABLE	VALOR	COSTO REDUCIDO
X <sub>1</sub>	4.363636	0
X <sub>2</sub>	4.181818	0
REGLON	HOLGURA	PRECIO DUAL
2)	0	.272727
3)	10.090910	0
4)	0	1.454545

Aumentamos la restricción  $X_1 \geq 5$  y se obtiene el siguiente valor óptimo para la función objetivo:

$$Z = 37$$

Y valores en las variables:

VARIABLE	VALOR	COSTO REDUCIDO
$X_1$	5	3
$X_2$	1	0
REGLON	HOLGURA	PRECIO DUAL
2)	7	0
3)	5	0
4)	0	2

Si hacemos  $X_1 \leq 4$  se obtiene para la función objetivo el siguiente valor

$$Z = 36$$

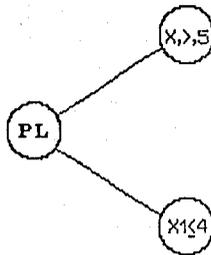
Y valores en las variables:

VARIABLE	VALOR	COSTO REDUCIDO
$X_1$	4	0
$X_2$	4	0
REGLON	HOLGURA	PRECIO DUAL
2)	0	1
3)	9	0
4)	2	0
5)	0	8

$$Z \leq B^1(4,26,-7) = 38 + 0.36363$$

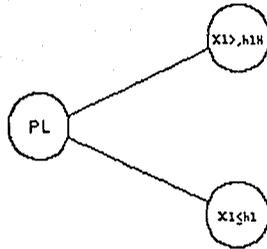
El cambio de 20 a 26 sugiere que el valor óptimo de  $X_1$  estará tan cercano a 4.5 como a 3.5, entonces, se puede tener un árbol mejor, ver figura 3.3

FIGURA 3.3



En términos más generales se puede considerar el árbol

FIGURA 3.4



Los precios duales en las restricciones  $x_1 \geq 4$  y  $x_1 \leq 3$  en los nodos 2 y 3 del árbol original fueron -3 y 8 respectivamente. De la misma forma que los precios duales en el nodo 5 se usaron para mejorar  $C^1(d)$  previamente, los precios duales en los nodos correspondientes a  $x_1 \geq 4$  y  $x_1 \leq 3$  se pueden usar para construir funciones que acoten los nodos correspondientes a  $x_1 \geq h_1 + 1$  y  $x_1 \leq h_1$ . Entonces,  $B^2(d)$ ,  $B^3(d)$  y  $B^1(d)$  se pueden generalizar como

$$B^2(d) = -3(h_1 + 1) + 2d_2$$

$$B^3(d) = 8h_1 + d_1$$

y

$$B^1(d) = \min\{0.2727 d_1 + 1.4545d_2, \max\{B^2(d), B^3(d)\}\}$$

usando  $h_1 = 4$  encontramos la cota más compacta

$$Z \quad B^1(4, 26, -7) = 37$$

De hecho la nueva solución óptima es  $x_1 = 5, x_2 = 1, Z = 37$

No es claro como se puede escoger el mejor valor de ramificación paramétrica en cada nodo. Una regla plausible es comenzar en la raíz y proceder hacia los nodos terminales, poniendo la ramificación paramétrica en un nodo igual al valor entero que minimice localmente la cota. Entonces se procede de los nodos terminales a la raíz, calculando las funciones  $B^i(d)$  como se hizo anteriormente.

### 3.4 ANALISIS DE SENSIBILIDAD PARA EL ALGORITMO DE DRIEBEEK

Considérese como se planteó en el algoritmo, que el valor de la función objetivo se va a maximizar. Cuando la matriz se construye como en el ejemplo, la introducción de un vector  $(\Delta b)$  del lado derecho reduce el valor de la función del óptimo continuo  $(b_0, cont.)$  por  $\Pi \Delta b$ , donde  $\Pi$  es el precio sombra o vector dual asociado con la matriz inversa  $B^{-1}$  correspondiente a los vectores básicos en la solución óptima continua, y  $\Delta b$  es el vector paramétrico que se usa para limitar el espacio de soluciones factibles a un único punto.

Intuitivamente esto es fácil de ver porque representa un decremento en el valor de la función cuando se aumentan restricciones adicionales. Además de esta reducción inmediata en el valor de la función objetivo, la introducción del vector  $b$  puede producir no factibilidad en el vector compuesto del lado derecho. Esta infactibilidad se puede eliminar por el algoritmo dual simplex, que reducirá aun más el valor de la función objetivo. No estamos seguros de que haya

infactibilidad, pero podemos asegurar al menos que el cambio  $\Pi \Delta b$  se efectúa. A esto se le conoce como "penalización mínima" o "penalización verdadera" en la que se incurre cuando el espacio de soluciones se reduce a un único punto.

En éste punto podemos notar también que en la multiplicación  $\Pi \Delta b$ , la penalización mínima para una solución entera especificada, es igual a la suma de costos reducidos de aquellas  $\alpha_{ij}$ 's que están asociadas con las restricciones enteras activadas, por ejemplo

El vector de precios sombra

$$\begin{array}{cccccc} \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \\ 1.0 & 1.5 & 2.0 & 1.6 & 2.4 & 3.2 \end{array}$$

Cuando se multiplica por:

$$\begin{array}{c} \Delta b \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{array}$$

da como resultado una penalización verdadera de  $2.0 + 2.4 = 4.4$

La cantidad 4.4 representa la penalización mínima de hacer la primer variable entera igual a 2 y la segunda variable entera igual a 1.

Es bien sabido que si una desigualdad se satisface con holgura en el óptimo continuo, esto es, la variable de holgura en la desigualdad está en la base óptima, entonces el precio sombra asociado con la desigualdad es igual a cero. En otras palabras, si una variable de holgura entera  $\alpha_{ij}$  está en la base, entonces el precio sombra de la restricción a  $ij$  original es cero y tenemos que buscar otra forma de estimar la penalización para activar ésta restricción. Para éstas variables de holgura enteras  $\alpha_{ij}$  que están en la base de la solución óptima continua, calculamos una "pseudo-penalización". Cuando una variable  $\alpha_{ij}$  permanece en la base óptima, entonces su nivel  $\alpha_{ij}$  debe satisfacer  $0 \leq \alpha_{ij} \leq 1$ . Si se fuera a sustraer una  $\Delta b$  con  $\Delta b_{ij} = 1.0$  y examinar la solución de éste problema modificado asociado con la base óptima de nuestro problema original, podemos encontrar que la variable  $\alpha_{ij}$  ahora tiene el valor

$$\alpha'_{ij} = \alpha_{ij} - 1.0$$

$$\text{entonces} \quad -1 \leq \alpha'_{ij} \leq 0$$

Excepto para el caso en que  $\alpha'_{ij} = 0$  se produce infactibilidad, la misma que se puede eliminar usando el dual simplex. No obstante ésta eliminación reduce el valor de la función en

$$(\alpha_{ij} - 1.0) \max\{(d_k/a_{rk}) \mid k = 1, \dots, n; a_{rk} < 0\} \quad (*)$$

donde  $A = [a_{jk}]$  es la matriz actualizada en el óptimo continuo. (max se refiere aquí a lo más cercano a cero, porque solo se consideran las razones  $(d_k/a_{rk}) \leq 0$ );  $d_k$  es el costo reducido de la  $k$ -ésima variable en la solución óptima continua y  $r$  es el índice del renglón ocupado actualmente por  $ij$  en la base. A esto llamamos una pseudo-penalización asociada

con la activación de la restricción original  $\alpha_{ij}$  o hablando en términos más simples, la pseudo-penalización asociada con  $\alpha_{ij}$ . Las pseudo-penalizaciones difieren de las penalizaciones verdaderas en que las pseudo-penalizaciones no son aditivas.

Con cada variable entera de holgura  $\alpha_{ij}$  tenemos asociada una penalización verdadera o una pseudo-penalización, donde estas penalizaciones representan la reducción en el valor de la función cuando el espacio de soluciones se reduce al nivel de la variable entera correspondiente.

Quando queremos saber si un punto específico puede ser la solución óptima a nuestro problema entero mixto, sabemos que podemos incurrir al menos en:

1. la suma de penalizaciones verdaderas asociadas con ésta solución o
2. la mayor de las pseudo-penalizaciones

Probamos esto de la siguiente manera: Cuando se elimina una infactibilidad debida a la activación de una restricción entera, ésta se remueve a través de una transformación de la base óptima continua por el método dual, la reducción inmediata en el valor de la función objetivo, así como la correspondiente variable entera de holgura se remueve de la base, es precisamente igual a la pseudo-penalización asociada. En el algoritmo dual convencional, la variable que deja la base se selecciona usualmente por el criterio  $\min\{b_i \mid b_i \leq 0\}$ .

Un criterio diferente es la selección de pivote mostrada en (\*) que es equivalente a medir el decremento total en el valor de la función que ocurrirá cuando una variable específica se remueva de la base. Ya que en el algoritmo dual cualquier selección de la variable que sale produce un decremento en el valor de la función objetivo, es aceptable seleccionar aquella variable que produzca el mayor decremento en la función objetivo.

Quando en un problema están contenidas varias variables enteras se puede considerar imponer restricciones enteras en sólo aquellas variables cuyas holguras están fuera de la base. Este paso reduce inmediatamente el valor de la función objetivo por la suma de las penalizaciones verdaderas. Inversamente, se pueden imponer restricciones enteras en sólo aquellas variables enteras cuyas holguras están en la base. Con una selección adecuada del primer vector que va a salir, podemos entonces reducir la función objetivo por la mayor "pseudo-penalización" en el primer cambio de base.

Quando simultáneamente se imponen restricciones enteras en todas las variables enteras, se incurre necesariamente en una penalización al menos tan grande como

- a. La suma de penalizaciones verdaderas
- b. La mayor de las pseudopenalizaciones.

#### REFERENCIAS AL CAPITULO III.

1. DRIEBEEK, N.J. "An Algorithm for the Solution of Mixed Integer Programming Problems", Man. Sci.12, No.7 576-587.
2. HOLM, S. y KLEIN, D. " Three Methods for Postoptimal Analysis in Integer Linear Programming", Math. Program.21,97-109, 1984.
3. SHRAGE, L. y WOLSEY, L. "Sensitivity Analysis for Branch and Bound Integer Programming", Opns. Res.33, No.5 1008-1023, 1985.
4. WOLSEY, L.A. "Integer Programming Duality: Price Functions and Sensitivity Analysis. Math. Program.20, 173-195, 1981.

## CAPITULO IV

### APLICACIONES EN AREAS ESPECIFICAS: PROGRAMACION NO LINEAL

"Las figuras que llenan el espacio de la geometría pitagórica le hicieron pensar sobre las maravillas de la geometría de las abejas. Se podrían haber agrupado células triangulares o cuadradas, conteniendo cada una la misma cantidad de miel que la célula hexagonal, pero las células hexagonales requieren menos cera. Como los espejos de Heron, esto indica nuevamente el mínimo esfuerzo en la naturaleza."

Herbert Westren

El papel fundamental que juega la programación lineal en la investigación de operaciones se refleja en su amplia aplicabilidad. Una suposición importante de programación lineal es la propiedad de linealidad que, en esencia, se cumple para muchos problemas prácticos. Sin embargo, en los últimos diez años la programación no lineal pasó de ser un tema relativamente nuevo y principalmente analítico a ser un importante instrumento de carácter general para la resolución de problemas.

Los grandes problemas de programación no lineal se presentan en problemas de estructuras mecánicas, como determinar configuraciones óptimas para puentes, armazones, etc. Algunos diseños mecánicos y configuraciones que antes se resolvían mediante ecuaciones diferenciales, ahora, suelen solucionarse resolviendo problemas de optimización apropiados. Un ejemplo lo constituye la determinación de la forma de un cable rígido suspendido entre dos puntos y soportando una carga.

Los problemas de control óptimo dan lugar a problemas de programación no lineal a gran escala. En éstos problemas, un sistema dinámico, que se describe a veces por una ecuación diferencial normal, relaciona variables de control con una trayectoria del estado del sistema. Esta ecuación diferencial, o una versión discreta de la misma, define un conjunto de restricciones. El problema consiste en seleccionar las variables de control de modo que la trayectoria resultante satisfaga varias restricciones adicionales y minimice algún criterio. Un ejemplo inicial de éste tipo de problemas, resuelto numéricamente, fue la determinación de la trayectoria de un cohete a la Luna que requería un consumo mínimo de combustible.

Hay muchos ejemplos de programación no lineal en las operaciones industriales y toma de decisiones mercantiles. La no linealidad se puede presentar en funciones de producción, curvas de costos y, de hecho, en casi todas las facetas de la formulación de problemas. De hecho muchos economistas han encontrado que un cierto grado de no linealidad es la regla y no la excepción en los problemas de planeación económica.

Los análisis de cartera, en el contexto de la inversión en el mercado de valores y en la evaluación de proyectos empresariales, es un área donde la programación no lineal es cada vez más útil. Estos problemas pueden llegar a tener fácilmente miles de variables.

Cabe esperar que esta tendencia continuará a medida que mejora la metodología para la solución de la programación no lineal.

Este capítulo se desarrolla como sigue: en la primera sección se define el problema no lineal y se dan algunos ejemplos ilustrativos. En la segunda sección se describen las condiciones de Kuhn-Tucker para soluciones de problemas no lineales. En la tercera sección se dan estrategias de análisis del problema entero no lineal mixto y se describe un experimento que realizó Gupta (1980) probando varios problemas no lineales con distintas estrategias de ramificación y acotamiento y usando un código denominado OPT basado en Gradiente Reducido Generalizado (GRG) para resolver los problemas no lineales continuos. En la cuarta sección se desarrolla un problema no lineal usando el paquete GINO para resolver los problemas continuos no lineales y se consideran las recomendaciones de Gupta para la selección de nodos y variables a ramificar con esto se hace una generalización del algoritmo de Dakin al caso no lineal y en la última sección se dan las conclusiones y algunas recomendaciones.

#### 4.1 EL PROBLEMA NO LINEAL Y SUS CARACTERISTICAS.

Sean  $f(x)$  una función continua, que denota a la función objetivo,  $h_k(x)$  funciones continuas que denotan restricciones de igualdad,  $g_i(x)$  funciones continuas que denotan restricciones de desigualdad continuas, y en general se supone que tienen segundas derivadas parciales continuas, y  $x_j$  un vector columna en el espacio euclidiano de  $n$ -dimensiones. Entonces se define un problema de optimización no lineal como sigue:

$$(4.1) \quad \min f(x) \quad x = (x_1, \dots, x_n) \in \mathbb{R}^n$$

sujeto a

$$\begin{aligned} h_k(x) &= 0 & k &= 1, 2, \dots, m \\ g_i(x) &\geq 0 & i &= m+1, \dots, p \end{aligned}$$

Quizá la primera cuestión que se presenta en el estudio del problema de minimización es saber si existe una solución. El principal resultado que se puede utilizar para encauzar esta cuestión es el teorema de Weierstrass, que establece que si  $f$  es continua y en éste caso  $\Omega = \mathbb{R}^n$  es compacta, existe una solución. Este es un resultado importante que habrá de tenerse en cuenta durante todo el desarrollo, sin embargo nos interesa caracterizar los puntos solución y diseñar los métodos efectivos para hallarlos. Donde se distinguen dos tipos de puntos solución a saber:

Un punto  $x^* \in \Omega$  es un punto mínimo relativo o un punto mínimo local de  $f \in \Omega$ , si existe un  $\epsilon > 0$  tal que  $f(x) \geq f(x^*)$  para toda  $x \in \Omega$  a una distancia de  $x^*$  menor que  $\epsilon$  (es decir,  $x \in \Omega$  y  $|x - x^*| < \epsilon$ ). Si  $f(x) > f(x^*)$  para toda  $x \in \Omega$ ,  $x$  es diferente a  $x^*$ , a una distancia de  $x^*$  menor que  $\epsilon$ , entonces se dice que  $x^*$  es un punto mínimo relativo estricto de  $f$  en  $\mathbb{R}^n$ .

Un punto  $x^* \in \Omega$  es un punto mínimo global de  $f \in \Omega$ , si  $f(x) \geq f(x^*)$  para toda  $x \in \Omega$ , con  $x$  diferente de  $x^*$ , entonces se dice que  $x^*$  es un punto mínimo global estricto de  $f \in \Omega$ .

Por definición, al formular y estudiar el problema no lineal se está buscando explícitamente un punto mínimo global de  $f$  en  $\mathbb{R}^n$ . Sin embargo, solamente se pueden hallar condiciones y soluciones globales si el problema posee ciertas propiedades de convexidad que garanticen esencialmente que cualquier mínimo relativo sea un mínimo global.

Un concepto fundamental que proporciona una mejor comprensión, además de simplificar el desarrollo teórico necesario, es el de restricción activa. Una restricción de desigualdad  $g_i(x) \leq 0$  se dice que es activa en un punto factible  $x$  si  $g_i(x) = 0$ , y es inactiva en  $x$  si  $g_i(x) < 0$ . Se adoptará la convención de considerar que cualquier restricción de igualdad  $h_i(x)$  es activa en cualquier punto factible. Las restricciones activas en un punto factible  $x$  restringen el dominio de factibilidad en las proximidades de  $x$ , mientras que las otras restricciones, las inactivas, no ejercen influencia en las proximidades de  $x$ . Por lo tanto, al analizar las propiedades de un punto mínimo local, es evidente que se puede centrar la atención en restricciones activas.

Un ejemplo de un problema de optimización no lineal es:

**EJEMPLO 4.1** Considere el problema

$$\min f(x) = x_1^2 + x_2^2 + 2x_2$$

sujeto a

$$h_1(x) = x_1 + 2x_2 - 1/2 = 0$$

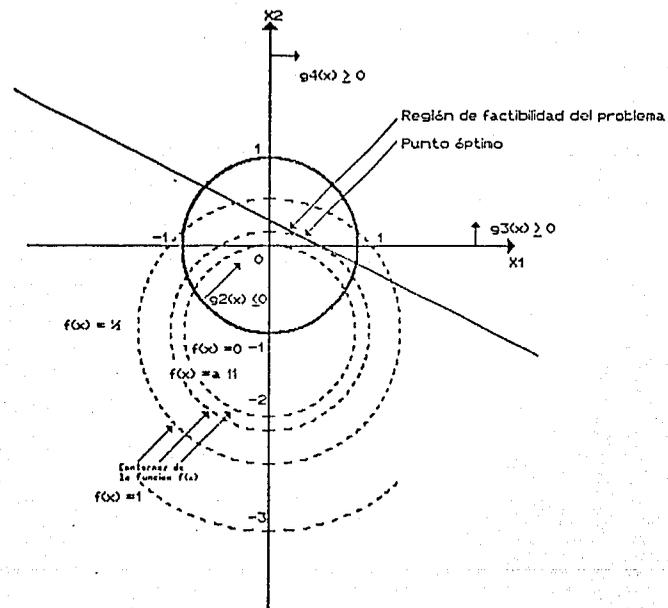
$$g_2(x) = x_1^2 + x_2^2 - 1 \leq 0$$

$$g_3(x) = x_1 \geq 0$$

$$g_4(x) = x_2 \geq 0$$

cuya representación gráfica es la siguiente:

**FIGURA 4.1**



En ésta gráfica la recta sombreada (generada por la intersección de restricciones lineales) constituye la región de factibilidad, mientras que los círculos concéntricos (con línea discontinua) son los contornos de un mismo valor de la función objetivo para diferentes valores del vector  $x = (x_1, x_2)$ .

Una de las características que hace a los problemas de optimización no lineal más difíciles de resolver que los problemas lineales, es que la solución óptima no se encuentra en un punto extremo de la región de factibilidad, como sucede en los lineales, por ejemplo:

**EJEMPLO 4.2** Considere el problema

$$\min f(x) = (x_1 - 3)^2 + (x_2 - 4)^2$$

sujeto a

$$g_1(x) = 5 - x_1 - x_2 \geq 0$$

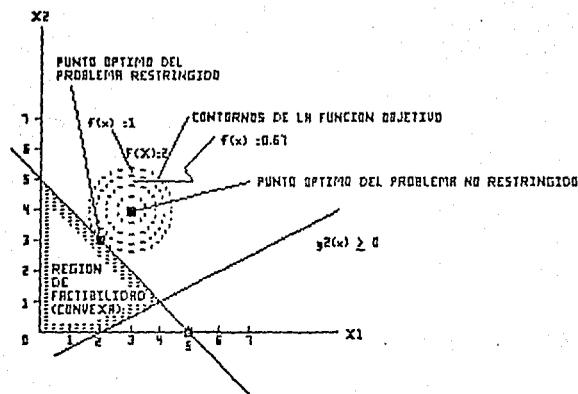
$$g_2(x) = 1/2 - x_1 + x_2 \geq 0$$

$$g_3(x) = x_1 \geq 0$$

$$g_4(x) = x_2 \geq 0$$

cuya representación gráfica es

**FIGURA 4.2**



En éste caso, el punto óptimo del problema restringido se encuentra en uno de los bordes de la frontera de la región de factibilidad, más no, en un punto extremo de la misma.

También sucede que el punto óptimo puede estar en el interior de la región de factibilidad como lo muestra el ejemplo:

**EJEMPLO 4.3** Considere el problema

$$\min f(x) = (x_1 - 2)^2 + (x_2 - 2)^2$$

sujeto a

$$g_1(x) = 5 - x_1 - x_2 \geq 0,$$

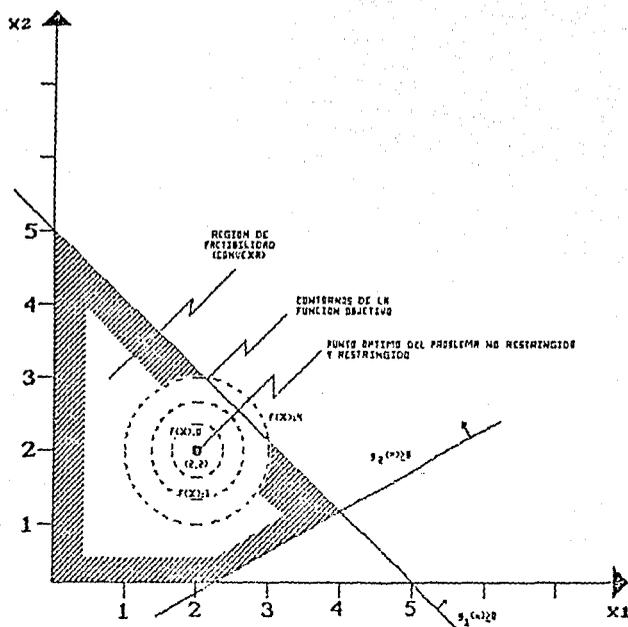
$$g_2(x) = 5/2 - x_1 + x_2 \geq 0,$$

$$g_3(x) = x_1 \geq 0,$$

$$g_4(x) = x_2 \geq 0,$$

cuya representación gráfica es:

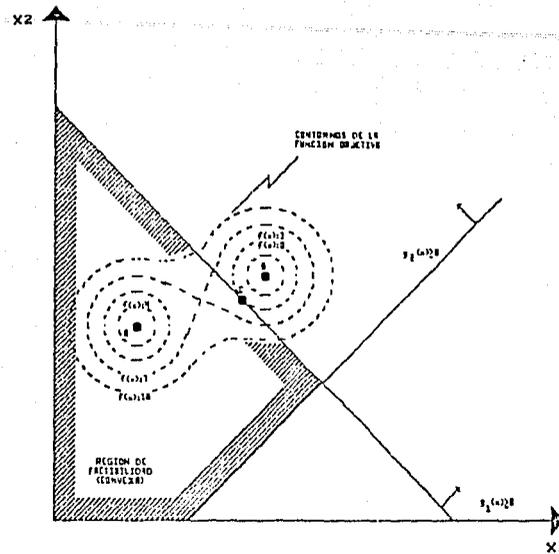
FIGURA 4.3



Una gran desventaja de los métodos de optimización no lineales, es que, generalmente encuentran un óptimo local o relativo, más no el óptimo global o absoluto. Un ejemplo sería el caso cuya representación gráfica se proporciona en la figura 4.4. En este caso los puntos A y B son mínimos locales o relativos del problema no restringido, mientras que los puntos A y C son los mínimos locales del problema restringido. Aún queda por determinar cuál es el mínimo global o absoluto. Probablemente, en este caso particular, parece ser que el punto C es un óptimo global, ya que  $f(C) = 3$  mientras que  $f(A) = 4$ . El problema serio que se presenta, es que no existe método alguno en optimización no lineal, que detecte sistemáticamente a todos los mínimos o máximos locales. Más bien éstos se encuentran por el método de error y prueba.

Otro serio problema en la optimización no lineal, es que se pueden generar regiones de factibilidad que no son necesariamente convexas. En el caso de programación lineal, todas las regiones de factibilidad son convexas. Las regiones de factibilidad no convexas las pueden generar restricciones no lineales, en nuestro caso sólo consideraremos la solución de problemas de programación convexa no lineales que requieren que todas o algunas de las variables de decisión sean enteras.

FIGURA 4.4



En general, un problema de programación entera mixta no lineal [PEMNL] puede formularse de la siguiente manera:

$$\text{minimizar } f(x), \quad x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n,$$

sujeito a:

$$\begin{aligned} g_i(x) &\geq 0 & i = 1, 2, \dots, NI \\ h_k(x) &= 0 & k = 1, 2, \dots, NE \\ x_j &\text{ es un entero} & j = 1, 2, \dots, m \leq n \end{aligned}$$

Debe notarse que de las  $n$  variables de decisión  $x_1, x_2, \dots, x_n$  las primeras  $m$  variables se identifican como variables enteras. Probablemente el procedimiento más sencillo que se puede usar para resolver tales problemas sea redondear como se hace en la programación lineal entera mixta. Se resuelve la relajación continua del PEMNL y se redondea la solución óptima continua al entero más cercano. Aunque el concepto es sencillo, es bien sabido que este método tiene muchas fallas. Puede suceder que la solución redondeada no sólo sea subóptima sino infactible. La tarea de localizar una solución factible entera puede en sí misma no ser trivial en un caso no lineal.

#### 4.2 CONDICIONES DE KUHN-TUCKER

Sea  $x^*$  un punto que satisfaga las restricciones

$$h_k(x^*) = 0 \quad g_j(x^*) \leq 0$$

Y sea  $J$  el conjunto de índices  $j$  para los cuales  $g_j(x^*) = 0$ . Entonces se dice que  $x^*$  es un punto regular de las restricciones (4.1), si los vectores gradientes  $\nabla h_i(x^*), \nabla g_j(x^*), 1 \leq i \leq m, j \in J$  son linealmente independientes.

Se observa que, según la definición de restricciones activas, un punto  $x^*$  es un punto regular si los gradientes de las restricciones activas son linealmente independientes.

### CONDICIONES DE KUHN-TUCKER.

Sea  $x^*$  un punto mínimo relativo para el problema

$$\min f(X)$$

suje to a

$$h_k = 0, k = 1, 2, \dots, p$$

$$g_i(X) \leq 0, i = 1, \dots, m$$

$$X \in E^n$$

y supóngase que  $x^*$  es un punto regular para las restricciones. Entonces existe un vector  $\lambda \in E^p$  y un vector  $\mu \in E^m$  con  $\mu \geq 0$  tal que

$$\begin{aligned} \nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) &= 0 \\ \mu^T g(x^*) &= 0 \end{aligned}$$

### REPRESENTACION GEOMETRICA DE LAS CONDICIONES DE KUHN-TUCKER.

Las condiciones de Kuhn-Tucker dicen, que en un óptimo local de un problema de optimización restringido, cualquier cambio pequeño en las variables del problema no cambian en lo absoluto el valor de la función objetivo. Por ejemplo

#### EJEMPLO 4.4

$$\text{Min } f(X_1, X_2) = (X_1 - 2)^2 + (X_2 - 1)$$

suje to a

$$g_1(X_1, X_2) = -X_2 + X_1^2 = 0$$

$$g_2(X_1, X_2) = X_1 + X_2 - 2 = 0$$

cuya representación gráfica está dada en la figura 4.5

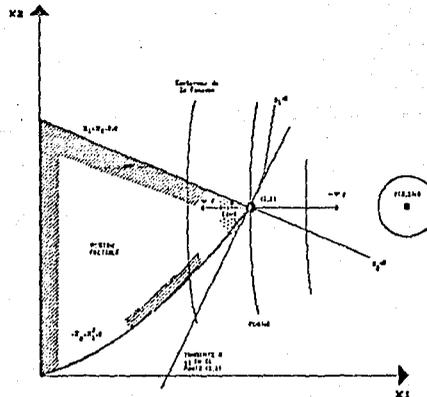


FIGURA 4.5

De la figura se puede ver que el óptimo del problema restringido es el punto (1,1). Se define como una dirección factible a un vector, cuyo movimiento a lo largo del mismo no viola ninguna de las restricciones del problema. En el entorno del punto óptimo (1,1), el conjunto de direcciones factibles se encuentra dentro del cono formado por las rectas  $g_2(X_1, X_2) = 0$  y la recta tangente a la restricción  $g_1(X_1, X_2) = 0$  en el punto (1,1). Este caso se encuentra ilustrado con puntos sombreados en la figura 4.5. El vector  $-\nabla f$  apunta en la dirección del máximo decremento de la función  $f(X)$ . Cualquier dirección que origine un ángulo menor a los 90 grados con  $-\nabla f$ , también disminuirá el valor de la función  $f(X)$ .

En la figura 4.6 se ilustra con más detalle lo que ocurre en el punto óptimo. Se nota que  $-\nabla f$  está contenido en el cono formado por  $\nabla g_1$  y  $\nabla g_2$ .

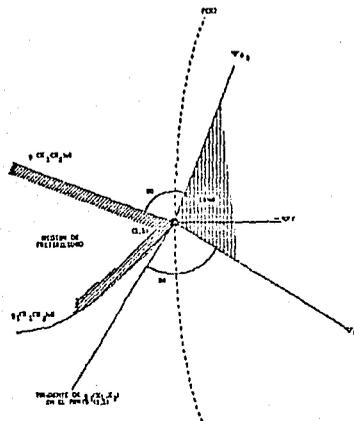


FIGURA 4.6

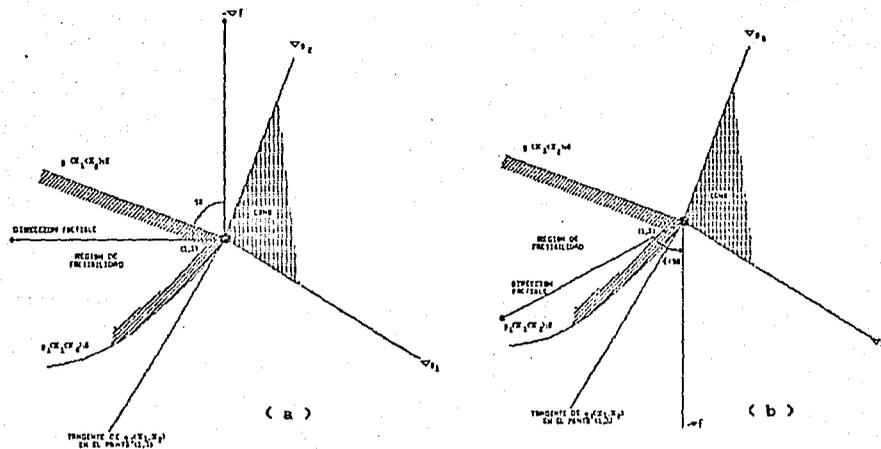


FIGURA 4.7

Si  $-\nabla f$  se encontrará arriba de  $\nabla g_2$ , como se muestra en la figura 4.7a., entonces formaría un ángulo menor de  $90^\circ$  con cualquier dirección factible que se encontrara un poco abajo de la restricción  $g_2(X_1, X_2)$ . Por otro lado, si  $-\nabla f$  se encontrara abajo de  $\nabla g_1$  tal como se muestra en la figura 4.7b, entonces formaría un ángulo menor a los  $90^\circ$  con cualquier dirección factible que se encontrara un poco arriba de la tangente de  $g_1(X_1, X_2)$  en el punto  $(1,1)$ .

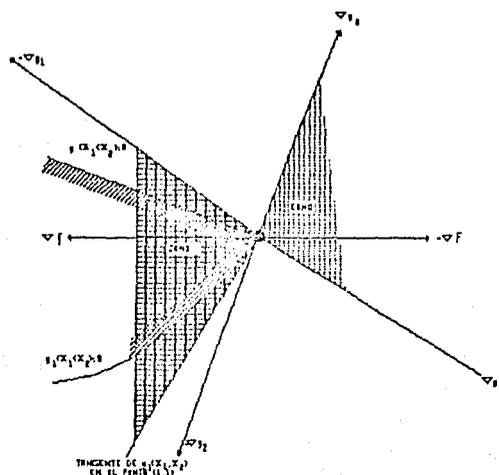


FIGURA 4.8

Se concluye por lo tanto que en un punto óptimo, ninguna dirección factible puede tener un ángulo menor a los  $90^\circ$  con  $-\nabla f$ . Los casos ilustrados en la figura 4.7 se pueden eliminar si y solo si  $-\nabla f$  está contenida en el cono formado por  $\nabla g_1$  y  $\nabla g_2$ , lo que equivale a que  $\nabla f$  está contenido en el cono formado por  $-\nabla g_1$  y  $-\nabla g_2$  como se muestra en la figura 4.8

#### REPRESENTACION MATEMATICA

De lo que se ha visto podemos concluir que las condiciones de Kuhn-Tucker, dicen que si la función objetivo  $f(X)$  y las restricciones  $g_i(X) \leq 0$ ,  $i = 1, \dots, m$  son diferenciables, una condición necesaria para que un punto  $X^0$  sea un óptimo local restringido del problema

$$\begin{aligned} & \text{Mín } f(X) \\ & \text{sujeeto a} \\ & g_i(X) \leq 0, \quad i=1, \dots, m \end{aligned}$$

es que en ese punto  $X^0$ , el gradiente de  $f(X)$ ,  $\nabla f(X^0)$ , esté contenido en el cono generado por el negativo de los gradientes de las restricciones activas, donde se entiende por restricciones activas aquellas que funcionan como igualdad en el punto considerado

Resumiendo, se tiene, que dado el problema de optimización

$$\text{Min } f(X)$$

sujeto a

$$g_i(X) \leq 0 \quad i=1, \dots, m$$

$$X \in E^n$$

donde  $f(X)$ ,  $g_i(X)$ ,  $i=1, \dots, m$ , son diferenciables, las condiciones de Kuhn-Tucker establecen que un punto óptimo local  $X^0$  debe necesariamente satisfacer

$$f(X^0) = \sum_{i=1}^m u_i (-\nabla g_i(X^0))$$

$$\begin{aligned} u_i &\geq 0 & i=1, \dots, m \\ g_i(X^0) &\leq 0 & i=1, \dots, m \\ u_i g_i(X^0) &= 0 & i=1, \dots, m \end{aligned}$$

#### 4.3 ESTRATEGIAS DE RAMIFICACION Y ACOTAMIENTO PARA PROBLEMAS DE PROGRAMACION ENTERA NO LINEAL.

Un problema de programación entera no lineal se puede escribir como

$$4.1 \text{ Min } f(X) \quad X \in R^n$$

sujeto a

$$4.2 \quad g_i(X) \geq 0 \quad i=1, 2, \dots, NI$$

$$4.3 \quad h_j(X) = 0 \quad j=1, 2, \dots, NE$$

$$4.4 \quad x_i \in I \quad 1 \leq i \leq m$$

donde  $I$  es el conjunto de enteros no negativos, las variables  $x_i$  son variables enteras, el vector  $X$  es una solución continua factible, si satisface las restricciones 4.2 y 4.3, y si satisface la restricción 4.4 es una solución entera factible.

El espacio de soluciones de un problema de programación entera no lineal se supone acotado. El número de puntos enteros que se deben investigar es finito, y la forma más simple de resolver un problema entero es enumerar todos esos puntos, descartando los no factibles, y manteniendo aquellos que sean soluciones factibles a la función objetivo. Cuando la enumeración se ha completado, la solución óptima, cuando ésta existe, está asociada con la mejor solución de la función objetivo. Esta idea de una búsqueda exhaustiva en el espacio de soluciones es simple de implantar pero computacionalmente ineficiente a menos que se tenga un problema pequeño a resolver, es por esto que recurrimos a ramificación y acotamiento.

El procedimiento básico de solución para resolver un problema de programación entera no lineal es esencialmente el mismo que el desarrollado por Dakin para el caso lineal, ya que el procedimiento de R- $\lambda$  no tiene nada que ver con la linealidad del problema. Como se recordará comenzamos relajando el problema entero a un problema continuo, si sucede que al resolver dicho problema relajado la solución es entera, ya terminamos, si no es entera entonces por lo menos una variable entera, digamos  $x_j$  es continua. Se divide entonces el valor de  $x_j$  en partes entera y fraccional,  $[x_j]$  y  $x_j'$  respectivamente, y se definen  $x_j = [x_j] + x_j'$  donde  $[x_j]$  es la parte entera y  $0 \leq x_j' \leq 1$ . Se forman dos subproblemas, uno con la restricción adicional de la cota superior  $x_j \leq [x_j]$ , y otro con la restricción de la cota inferior  $x_j \geq [x_j] + 1$ . El proceso de formación de estos subproblemas se llama "ramificación". Se resuelve cada uno de éstos problemas

nuevamente como un problema continuo. La información referente a la solución óptima de los problemas así resueltos se almacena en un nodo.

Generalmente se almacena en un nodo la solución óptima, el valor correspondiente de la función objetivo así como cotas superiores e inferiores sobre las variables. Nótese que los valores de la función objetivo proporcionan una cota inferior al valor óptimo del PEMNL.

Se continúa con este procedimiento de ramificación y la consecuente solución de una serie de problemas no lineales hasta que se encuentra una solución entera factible de uno de los problemas continuos. Cuando se obtiene una solución entera factible, el valor correspondiente de la función objetivo se vuelve una cota superior sobre el objetivo del problema entero no lineal.

A estas alturas podemos eliminar de futura consideración todas las soluciones o nodos cuyos respectivos valores de la función objetivo sean mayores que la cota superior y decimos que los nodos correspondientes están sondeados. También se eliminan o sondean los nodos cuando el problema continuo tiene una solución entera factible o es no factible. Se repite este procedimiento de R-A para cada uno de los nodos no sondeados.

Cuando se encuentra una solución factible entera y se encuentra que el valor de la función objetivo es menor que la cota superior obtenida hasta el momento, ésta se convierte en la nueva cota superior sobre el objetivo. La búsqueda para la solución óptima termina cuando todos los nodos se han sondeado. La mejor solución entera actual es la solución óptima al problema de programación entera no lineal dado.

En este momento cabe hacerse las siguientes preguntas:

- (1) ¿Que procedimiento se debe usar para resolver los problemas continuos no lineales intermedios?
- (2) ¿Qué criterio se debe usar para seleccionar la variable que se debe ramificar?
- (3) ¿Qué criterio se debe usar para seleccionar el nodo a ramificar?

Analizaremos cada una de estas preguntas por separado.

#### SELECCION DEL ALGORITMO CONTINUO NO LINEAL

Ya que es necesario resolver una gran cantidad de problemas continuos no lineales, es necesario que se emplee un algoritmo eficiente para su solución. Como es sabido, se han desarrollado algoritmos para resolver problemas continuos no lineales basados en una variedad de técnicas tales como: Métodos de penalización de funciones (Fiacco y Mc. Cormick 1968), Métodos de Gradiente Reducido Generalizado (GRG) (Abadie y Carpentier 1969, Wolfe 1967), etc. Se han desarrollado varios códigos de computadora basados en estos algoritmos. Se han hecho estudios comparativos (Colville 1968, Eason y Fenton 1974, Sandgren y Ragsdell 1980, Waren y Lasdon 1979) para evaluar el comportamiento relativo de estos códigos. Estos estudios han mostrado que los códigos basados en métodos GRG son significativamente superiores a los otros.

Esto es particularmente útil para nuestra aplicación ya que el método de R-A simplemente altera las cotas inferiores y superiores sobre las variables durante todo el proceso de solución.

El algoritmo GRG se describe brevemente a continuación:

Considere el siguiente problema de optimización

$$\text{Mín } f_0(X)$$

sujeto a

$$f(X) = 0$$

$$a \leq X \leq b$$

donde  $X$ ,  $a$ ,  $b$  son vectores columna de  $n$  dimensiones,  $f(X)$  es para cualquier  $X$  un vector columna de  $m$  dimensiones, y como  $f_0(X)$  es un número real. Ambos  $f_0$  y  $f$  se suponen continuamente diferenciables en el polítopo  $P$  definido por  $a \leq X \leq b$ . Si se permite que alguna de las  $a_j$ 's y alguna de las  $b_j$ 's sea igual a  $-\infty$  o  $+\infty$ , cualquier problema de optimización se puede escribir en la forma anterior. Sin embargo, podemos suponer que  $a$  y  $b$  están acotadas y difícilmente algunas  $a_j$ 's y  $b_j$ 's pueden ser muy grandes.

El gradiente de  $f_0(X)$  es un vector renglón denotado por  $\nabla f_0(X)$ . En forma semejante, el gradiente de  $f_i(X)$ ,  $i = 1, \dots, m$  es un vector renglón denotado por  $\nabla f_i(X)$ . La notación  $\nabla f(X)$  designa los renglones  $\nabla f_i(X)$   $i = 1, \dots, m$  de la matriz de orden  $m$  por  $N$ .

El vector  $X$  se particiona en  $x, y$ , donde  $y$  es de dimensión  $m$ , mientras que  $x$  es de dimensión  $n$ , con  $n = N - m$ . De donde se sigue que  $y$  es llamado el vector de variables básicas, mientras que  $x$  es el vector de variables independientes.

Los vectores renglón de las derivadas parciales de  $f_0$  con respecto a  $x$  y  $y$  se denotan por  $\delta f_0 / \delta x$  y  $\delta f_0 / \delta y$  respectivamente. También la matriz  $\nabla f(X)$  se particiona en la matriz  $\delta f / \delta x$  de orden  $m$  por  $n$  y la matriz  $\delta f / \delta y$  de orden  $m$  por  $m$ .

Sea el punto factible  $X^0$  tal que satisfaga la siguiente condición:

Condición de No degeneración: Existe una partición de  $X$  en  $x, y$  tal que:

$$a_j < y_j^0 < b_j, \text{ para toda } j$$

$$\delta f / \delta y^0 \text{ es no singular}$$

$$\text{donde } \delta f / \delta y^0 \text{ significa } \delta f / \delta y \text{ calculada en } x^0, y^0.$$

El algoritmo GRG entonces se puede escribir como sigue:

## ALGORITMO DE GRADIENTE REDUCIDO GENERALIZADO

PROPOSITO: Resolver el problema de programación no lineal continuo

### DESCRIPCION

Paso 1. Calcule  $-h^0$ , la dirección de movimiento de la variable  $x$ , siguiendo los siguientes pasos:

Paso 1.1 Calcule el gradiente reducido

$$g^0 = \begin{pmatrix} \delta f_0 / \delta x^0 \\ \delta f_0 / \delta y^0 \\ \delta f_0 / \delta x^0 \end{pmatrix}^{-1} \delta f$$

Paso 1.2 Calcule el gradiente reducido proyectado  $p^0$ , dado por:

para toda  $j$

$$p_j^0 = \begin{cases} 0 & \text{si } x_j^0 = a_j \text{ y } -g_j^0 < 0 \\ 0 & \text{si } x_j^0 = b_j \text{ y } -g_j^0 > 0 \\ g_j^0 & \text{en otro caso} \end{cases}$$

Paso 1.3 Calcule  $h^0$ , el gradiente reducido proyectado modificado, esto es, la dirección opuesta a la que se mueve la variable independiente. Esta dirección puede ser simplemente  $h^0 = -p^0$ , o también puede estar modificada, por una o por ambas de las ideas siguientes:

a. Haciendo todas las componentes tales que  $p_j^0(x_j^0 - a_j)$  o  $p_j^0(x_j^0 - b_j)$  es positivo o muy pequeño o igual a cero;

b. por direcciones conjugadas

Para todas estas modificaciones el producto escalar  $g^0 h^0$  es positivo.

Paso 2 Calcule un primer valor del número positivo  $\theta_1$

Paso 3 Calcule  $x^0 - \theta_1 h^0$ , y proyectelo en el politopo  $a_j \leq x_j \leq b_j$ , para toda  $j$  para obtener  $x^1$ , esto es, haga:

$$x_j^1 = \begin{cases} a_j & \text{si } x_j^0 - \theta_1 h_j^0 < a_j \\ b_j & \text{si } x_j^0 - \theta_1 h_j^0 > b_j \\ x_j^0 - \theta_1 h_j^0 & \text{en otro caso.} \end{cases}$$

Paso 4 Calcule una  $X^1$  factible correspondiente a  $y^1$ , esto es, trate de resolver con respecto a  $y$  el sistema de  $m$  ecuaciones con  $m$  incógnitas:

$$f(x^1, y) = 0 \quad (4.5)$$

esto se hace usualmente a través de algún método iterativo.

Paso 4.1 Si no se observa una convergencia rápida, entonces se decreta  $\theta_1$  y vaya al paso 3 con la misma  $h^0$

Paso 4.2 De otra manera, sea  $y^1$  la solución obtenida por (4.5), y  $X^1$  el punto correspondiente en todo el espacio de  $n$  dimensiones.

a. Si  $f_0(X^1) \geq f_0(X^0)$ , entonces decrece  $\theta_1$ , y se regresa al paso 3 con la misma  $h^0$ ;

b. En otro caso, al final del paso 4 tenemos alguna  $X^1$  factible, que satisface

$$f_0(X^1) < f_0(X^0) \quad (4.6)$$

**Paso 5** Podemos hacer ahora  $X^0 = X^1$  y comenzar una nueva iteración, o tratar de mejorar el último valor obtenido para  $f_0$ . Haciendo ésto se regresa al paso 3 con un nuevo valor experimentado para  $f_0$ , con la misma  $h_0$ , y se terminaría en el paso 5 con algún  $X^1$  que satisfaga (4.6) y se inicia una nueva iteración con  $X^0 = X^1$ .

#### SELECCION DE VARIABLES PARA RAMIFICACION

En el caso de problemas lineales se ha sabido que la regla usada para la selección de variables para ramificación puede tener un efecto significativo sobre el comportamiento total de una estrategia de R-A. Para ver ésto se consideraron tres estrategias de selección de variables mismas que se listan a continuación:

1. **El Primer Índice Menor.** Es posible tener alguna información sobre la importancia de algunas de las variables enteras en un modelo dado. Las variables enteras están arregladas en orden de importancia, procesandose primero la más importante de éstas. Esto se logra poniendo índice a las variables con las prioridades decrecientes de las variables enteras y luego se selecciona la variable con el primer índice mas bajo.

2. **Variable Entera Mas Fraccional.** Esta estrategia selecciona la variable que está mas lejos del valor entero mas cercano. Esta opción está enfocada a obtener la mayor disminución de la función objetivo cuando se efectúa la ramificación para que se puedan sondear mas nodos en una etapa preliminar.

3. **Uso de Pseudo-Costos.** El concepto de pseudocostos se ha desarrollado para resolver problemas de programación lineal entera mixta. Se usan los pseudo-costos como una medida cuantitativa de la importancia de las variables enteras y ésto permite asignar cierta prioridad a las variables.

Para cada variable entera  $x_j$  se definen dos pseudo-costos, uno inferior ( $pc_{lj}$ ) y uno superior ( $pc_{uj}$ ). Los valores de los pseudo-costos inferiores y superiores se calculan durante la búsqueda del árbol de la siguiente manera:

Supóngase que en un nodo  $k$  se selecciona la variable  $x_j$  para ramificación. Sea  $x_j^*$  la parte fraccional del valor de la variable  $x_j$ . Sea  $f_k$  el valor de la función objetivo en éste nodo  $k$ . Sea  $f_L$  el valor del objetivo cuando el problema continuo está resuelto con la restricción adicional de cota inferior  $x_j \leq [x_j]$ . El pseudo-costo inferior de  $x_j$  se define por  $pc_{lj} = (f_L - f_k) / x_j^*$ .

Sea  $f_U$  el valor de la función objetivo cuando el problema continuo se resuelve con la restricción adicional de cota superior  $x_j \geq [x_j] + 1$ . El pseudo-costo superior de  $x_j$  se define por  $pc_{uj} = (f_U - f_k) / (1 - x_j^*)$ .

Estos pseudocostos se pueden interpretar como el empeoramiento del valor de la función por unidad de cambio de la variable  $x_j$ , donde  $pc_{lj}$  corresponde al decremento de  $x_j$  y  $pc_{uj}$  corresponde al incremento de  $x_j$ . Aunque los valores de los pseudo-costos dependen del nodo donde se calculan, éstos sólo se calculan una vez y se supone quedan constantes para poder ahorrar el esfuerzo de recalcularlos en todos los nodos. Esta estrategia para seleccionar la variable de ramificación se describe de la siguiente manera:

- i) Calcular los pseudo-costos inferiores y superiores para todas las variables enteras cuando sea posible.
- ii) Calcular la cantidad  $V_j = \min\{pc_{lj} x_j^*, pc_{uj}(1-x_j^*)\}$  para cada variable entera  $x_j$
- iii) Seleccionar la variable  $x_j$  para la cual  $V_j$  es máximo.

La selección de la variable de ramificación que se ha descrito es tal que se espera que el valor de la función sea el que más aumente.

## SELECCION DE NODOS PARA RAMIFICACION.

Así como un criterio para seleccionar la variable puede afectar el comportamiento de una estrategia de R-A, se ha encontrado que el método de selección para los nodos que se van a ramificar puede de igual manera afectar significativamente el comportamiento de R-A. Implantamos las siguientes estrategias de selección de nodos para observar sus efectos sobre problemas de programación entera no lineal.

1. Ramificar desde el nodo con la menor cota. Como su nombre lo dice, ésta estrategia se selecciona para ramificar al nodo que actualmente tiene la menor cota sobre la función objetivo. Lawler y Wood (1966) argumentan que si para cualquier problema dado, el conjunto de nuevos problemas de acotamiento está únicamente determinado, entonces ésta estrategia tiene la ventaja de que el número total de cómputos se minimiza, en el sentido de que cualquier operación de ramificación que se realiza bajo ésta estrategia debe poder efectuarse bajo cualquier otra estrategia.

2. Ramificar desde el nodo mas nuevo. En ésta estrategia, siempre que se efectúa la ramificación se da preferencia a los nodos que corresponden a los nuevos problemas sobre los demás nodos no sondeados. Para la ramificación, se selecciona el nodo mas nuevo de la lista de nodos no sondeados. Esta estrategia se conoce por varios nombres como lo son: estrategia de profundidad, de ir hacia atrás, último en entrar primero en salir (LIFO), y tiene la ventaja de ahorrar espacio de almacenamiento. Como señala Ibaraki (1976), éste tipo de estrategia requiere de una cantidad de memoria que es una función lineal del tamaño del problema y es relativamente fácil de implantar.

3. Uso de estimaciones. En la estrategia de selección del nodo con la menor cota, se selecciona un nodo con la menor cota para la función objetivo, esta estrategia solo considera el valor de la función objetivo y no toma en cuenta la "calidad" de las soluciones óptimas continuas. Consideramos un problema con tres variables para ilustrar el concepto de "calidad". Suponga que necesitamos seleccionar un nodo de entre los siguientes nodos no sondeados:

$$x^{(1)} = (1.3, 2.4, 5.1), z_1 = 10.5; \text{ y } x^{(2)} = (1.0, 3.0, 5.1), z_2 = 10.6$$

De acuerdo al esquema de selección de la menor cota se daría preferencia al nodo 1 sobre el nodo 2. Sin embargo, tal selección no parece apropiada, ya que la solución  $x^{(2)}$  está mucho mas cercana a una solución entera. Si se selecciona el nodo 2, es posible que esté sondeado en la siguiente iteración mientras que el nodo 1 podría requerir de varias iteraciones antes de poderlo eliminar. Se han hecho intentos de modificar el esquema de selección de la menor cota para que también se tome en cuenta la calidad de las soluciones óptimas continuas. En éste caso lineal, Benichou (1971) lo logra usando un nuevo concepto llamado "estimación" que hace uso de los pseudo-costos.

En el nodo  $k$ , se calcula una cantidad llamada la estimación del nodo  $k$ , denotada por  $E_k$ , y se calcula:

$$E_k = f_k + \sum_{j=1}^m V_j$$

La cantidad  $E_k$  es de algún modo un estimador del valor de la función objetivo de la mejor solución entera que puede esperarse descendiente del nodo  $k$ .

Usando el concepto de estimación, ésta estrategia de selección del nodo puede enunciarse de la siguiente manera:

- i) Calcule la estimación de todos los nodos no sondeados.
- ii) Seleccione el nodo con la menor estimación

## HEURISTICAS PARA LA COTA SUPERIOR.

El menor valor de  $f_k$  en cualquier etapa, proporciona una cota inferior de la función objetivo del PEMNL. Frecuentemente es importante encontrar una cota superior sobre el objetivo tan pronto como sea posible. Esto se puede lograr encontrando inicialmente una solución entera factible al problema. las siguientes dos heurísticas se desarrollaron para encontrar soluciones iniciales enteras factibles.

Heurística A. Sea  $x = (x_1, x_2, \dots, x_n)$  la solución para un nodo no sondeado y sea  $p$  el número de variables enteras que toman valores no enteros en ésta solución. El número  $p$  se define como el Orden de Infactibilidad de éste nodo.

La Heurística A se efectúa de la siguiente manera:

1. Seleccione un nodo para el cual el orden de infactibilidad es mínimo. Seleccione una variable  $x_j$ , para ramificar de acuerdo con la regla de selección preestablecida. Para éste nodo, un número de variables enteras ya pueden tener valores enteros fijos, esto es, mientras se ramifica no se permiten cambios en estas variables para ninguno de los dos subproblemas. Esta idea consiste en reducir el número de variables enteras con valores no enteros, y así trabajar hacia soluciones con órdenes menores de infactibilidad.

2. Si el subproblema con la restricción de cota superior  $x_j \leq [x_j]$  es "factible continuo" se fija la variable  $x_j$  a  $[x_j]$  en todos los subsecuentes descendientes de éste nodo. Similarmente, si el subproblema con la restricción de cota inferior  $x_j \geq [x_j] + 1$  es "factible continuo" se fija la variable  $x_j$  a  $[x_j] + 1$  en todos los subsecuentes descendientes de éste nodo.

Si después de resolver los dos subproblemas se obtiene una solución entera factible, deténgase. De otro modo, regrese al paso 1.

Heurística B. Para simplificar notación suponga que todas las variables son variables enteras. La Heurística B se efectúa de la siguiente manera:

1. Resuelva el problema no lineal continuo. Sea  $x = (x_1^0, x_2^0, \dots, x_n^0)$  la solución óptima continua. Encuentre el entero  $y_i$  mas cercano a  $x_i^0$ , para  $i = 1, 2, \dots, n$  y forme un vector  $y = (y_1, y_2, \dots, y_n)$ . Inicie un contador  $i = 0$ .

2. Si  $y$  es una solución continua factible, vaya al paso 4. de otra forma, siga al paso 3.

3. Vuelva a poner el contador  $i$  en  $i + 1$ , y reemplace el valor de la variable  $x_i$  por  $x_i^0$ . También reemplace el vector  $y$  por el vector  $y = (x_1^0, x_2^0, \dots, x_i^0, y_{i+1}, \dots, y_n)$  y regrese al paso 2.

4. Si  $y$  es entero factible deténgase. De otro modo siga al paso 5.

5. Construya un problema  $i$ -dimensional sustituyendo las variables  $x_{i+1}, x_{i+2}, \dots, x_n$  por los valores enteros  $y_{i+1}, y_{i+2}, \dots, y_n$  en la función objetivo y restricciones del problema dado. Así se construye un problema de programación entera no lineal  $i$ -dimensional con  $x_1, x_2, \dots, x_i$  como sus variables enteras. Comenzando con  $(x_1^0, x_2^0, \dots, x_i^0)$  como una solución inicial, aplique R-A para resolver este problema  $i$ -dimensional. Si cualquiera de los descendientes de éste problema tiene una solución entera factible, digamos  $(z_1, z_2, \dots, z_i)$  deténgase. El vector  $x = (z_1, z_2, \dots, z_i, y_{i+1}, \dots, y_n)$  sería una solución entera a nuestro problema original. De otro modo regrese al paso 3.

Se debe notar que al inicio de la heurística los problemas están mas pequeños en tamaño, y por lo tanto se pueden resolver rápidamente. Sin embargo, conforme se desarrolla el proceso de solución los problemas se vuelven mas grandes en tamaño, y en la peor situación posible el problema original en si mismo estará resuelto. Si el problema original tiene una solución entera factible, en nuestra búsqueda, ésta sería descubierta. Por lo tanto, ésta heurística garantiza que se obtendrá una solución factible entera cuando la haya para el problema dado.

## DISEÑO EXPERIMENTAL

Las variables primarias evaluadas en el trabajo experimental fueron las estrategias para la selección de variables de ramificación y selección de nodos. Tres opciones para cada una de éstas dos categorías de selección dieron nueve combinaciones de estrategias. Para cada combinación, hemos explorado tres opciones sobre las heurísticas:

- 1) Sin heurística
- 2) Heurística A
- 3) Heurística B

Así se tuvo un total de 27 estrategias de R-A que consisten en un diseño experimental con tres niveles de tratamiento. En el estudio se usó un código de computadora BBNLMIP desarrollado por Gupta (ver apéndice C.3) que tiene la capacidad de llamar a cada una de las 27 estrategias que se usaron en el estudio. BBNLMIP emplea el código no lineal OPT para resolver los problemas continuos no lineales. Cada estrategia se identificó por los parámetros (K, L, N) donde K fue la opción para la selección de la variable de ramificación, L para la selección del nodo de ramificación y N la selección de heurística. Las opciones para K, L y N se denotaron como sigue:

- K=1 : Primer índice menor
- K=2 : Variable entera más fraccional
- K=3 : Uso de pseudo-costos
- L=1 : Ramificación desde el nodo con la menor función objetivo
- L=2 : Último en entrar primero en salir
- L=3 : Ramificación desde el nodo con la menor estimación.
- N=1 : Sin heurística
- N=2 : Selección de la Heurística A
- N=3 : Selección de la Heurística B

## PROBLEMAS DE PRUEBA

En cualquier estudio computacional es importante examinar una amplia gama de problemas de prueba. Se debe señalar que, en contraste con su contraparte lineal, en la programación entera no lineal, el tamaño del problema es tan sólo uno de los muchos factores en la determinación de su complejidad total; el grado de no linealidad de la función objetivo y de las funciones de restricción incluyendo el número de términos no lineales contribuye significativamente a la complejidad del problema.

Ya que según lo que sabemos no se ha llevado a cabo ningún estudio experimental de esta naturaleza, se consideran problemas de prueba de estudios anteriores sobre algoritmos continuos no lineales. Entre los problemas de prueba disponibles, sólo se consideraron aquellos problemas convexos donde ciertas variables se pueden tratar como variables enteras; por ejemplo aquellos problemas que tienen algunas variables con cotas razonablemente estrechas y regiones factibles con puntos enteros.

No sirve a nuestros propósitos considerar aquellos problemas que no tienen puntos enteros en su región factible o donde las variables toman valores elevados. Así, se incluyeron 22 problemas en total, los cuales abarcan desde 2 hasta 16 variables continuas y de 2 a 10 variables enteras. De los 22 problemas, los problemas 5, 8, 20 y 21 fueron del tipo de

programación entera no lineal mixta, y los demás fueron problemas de programación entera no lineal pura. El número de restricciones varió de 0 a 8. El problema más grande que se resolvió fue el problema 20 con 24 variables (8 enteras) y 8 restricciones lineales involucrando 16 términos en cada restricción. La función objetivo era polinomial de cuarto grado involucrando 45 términos no lineales.

Como lo ilustra la Tabla 1, el tamaño del problema no determina el esfuerzo computacional. El problema 19 que fué el que mas tiempo tardó para resolverse sólo tenía 16 variables (8 enteras) y 8 restricciones. Sin embargo su función objetivo involucraba 40 términos cuadráticos y cada una de las restricciones tenía entre 25 y 32 términos cuadráticos. Similarmente el problema 17 sólo tenía 14 variables (7 enteras) y 7 restricciones, pero la función objetivo tenía 3 términos cuadráticos y cada restricción tenía entre 24 y 28 términos cuadráticos. Las funciones objetivo de todos los problemas eran no lineales, sin embargo los problemas 4, 6,9 y 16 no tenían restricciones y los problemas 3,15 y 20 no tenían restricciones lineales. También se consideraron problemas adicionales de mayor dificultad pero el excesivo costo computacional los hizo imprácticos para usarlos como problemas de prueba. Se puede encontrar una lista completa de problemas de prueba en Gupta (1980)

TABLA 1

Tiempos Actuales de Solución (segundos)

Prob	T. de Sol. Prom.	Peor T. de Sol	Mejor T. de Sol.	Razón P/M
1	1.035	1.433	0.822	1.74
2	12.038	22.150	5.339	4.15
3	0.759	0.875	0.704	1.24
4	0.466	0.527	0.428	1.23
5	6.250	6.835	5.720	1.19
6	0.563	0.651	0.521	1.25
7	0.856	1.136	0.631	1.80
8	2.504	2.991	2.136	1.40
9	3.256	4.707	2.527	1.86
10	0.728	0.921	0.626	1.47
11	2.098	2.902	1.401	2.07
12	4.816	8.044	2.760	2.91
13	13.668	24.677	7.660	3.22
14	12.196	20.748	9.679	2.14
15	1.303	1.581	1.131	1.40
16	0.624	0.828	0.503	1.65
17	108.131	220.418	49.378	4.46
18	83.554	237.756	33.377	7.12
19	241.522	496.349	90.992	5.45
20	66.261	109.011	50.257	2.17
21	1.945	2.061	1.877	1.10
22	8.233	12.118	5.973	2.03
				Razón 2.41 Promedio

#### 4.5 EJEMPLO NUMERICO USANDO GINO.

Considere el siguiente problema de programación entera no lineal:

$$\text{MIN } Z = 7X_1^2 + 6X_2^2 - 35X_1 - 80.4 X_2$$

SUJETO A

$$9X_1^2 - 10X_1X_2 - 8X_2^2 > - 583$$

$$6X_1^2 - 8X_1X_2 - 6X_2^2 > - 441$$

$$X_1, X_2 \geq 0 \text{ enteras}$$

Se desea resolver éste problema usando un paquete para programación no lineal y ramificación y acotamiento para cumplir las restricciones de variable entera.

Resolvemos primero el problema no lineal sin las restricciones de variable entera, obteniéndose el siguiente valor para la función objetivo:

$$Z = -313.090010$$

Y los siguientes valores en las variables:  $X_1 = 2.5, X_2 = 6.7$

De aquí el paso siguiente es seleccionar la variable a ramificar, que siguiendo la recomendación de Gupta (1980), sería la variable más fraccional, por lo cual seleccionamos  $X_1$ , y agregamos al problema relajado la restricción  $X_1 = 2$  de donde se tiene que el valor de la función objetivo es:

$$Z = -311.340010$$

Con los siguientes valores en las variables:  $X_1 = 2, X_2 = 6.7$

Ahora agregamos la restricción  $X_1 = 3$  en lugar de la anterior y se obtienen los siguientes resultados:

$$Z = -311.340010$$

Con los siguientes valores en las variables:  $X_1 = 3, X_2 = 6.7$

Como se puede observar el valor de las variables y de la función objetivo en éstos dos nodos coinciden por lo que no importa en cual de éstos dos nodos se continúa la ramificación, seleccionamos el nodo que tiene  $X_1 = 2$  y agregamos la restricción  $X_2 = 7$ , obteniéndose el siguiente valor en la función objetivo:

$$Z = -310.800011$$

Y valores en las variables:  $X_1 = 2, X_2 = 7$

Si se fija  $X_2 = 6$ , se obtiene el siguiente valor en la función objetivo:

$$Z = -308.400009$$

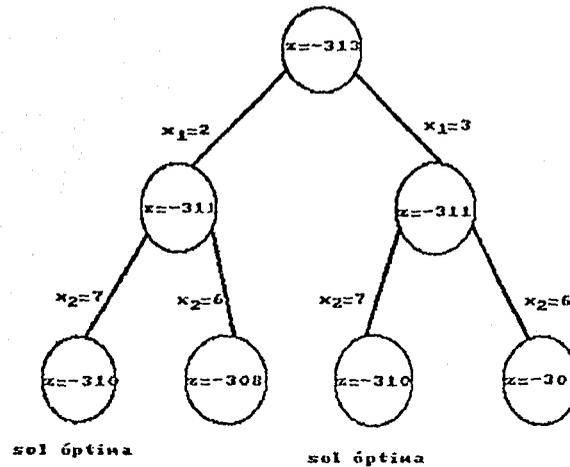
Y valores en las variables:  $X_1 = 2, X_2 = 6$

El paso siguiente consistiría en seguir la ramificación desde el nodo donde se fija la variable  $X_1 = 3$ , sin embargo los resultados obtenidos en éste caso son idénticos que los que se obtienen para la rama que corresponde al nodo con  $X_1 = 2$ , por lo cual no es necesario que se reproduzcan aquí. De éstos dos últimos resultados para la función objetivo escogemos aquel que la minimice por lo cual la solución óptima está dada por:

$$Z = -310.80$$

Y valores en las variables:  $x_1 = 2$  y  $x_2 = 7$ . En la figura 4.9 se ilustra el árbol que describe éste procedimiento.

FIGURA 4.9



#### 4.6 CONCLUSIONES Y RECOMENDACIONES.

En éste capítulo hemos investigado la factibilidad de aplicar el método de R-A a problemas de programación entera no lineal. Como hemos demostrado, se pueden implantar los métodos de ramificación y acotamiento como herramientas útiles para resolver tales problemas. Hemos probado los conceptos de pseudo-costos y estimaciones para seleccionar variables y nodos para ramificación. Sin embargo los resultados muestran que entre las tres opciones para la selección de variables para ramificar que hemos probado, la mejor fue la estrategia de seleccionar la variable con el valor mas fraccional. Entre las estrategias de selección de nodos, el criterio de seleccionar el nodo mas reciente fue la peor. Se debe notar que se hace un esfuerzo considerable para calcular los pseudo-costos y las estimaciones al principio de cada corrida y por eso probablemente no se explota su fuerza en problemas de prueba menos complejos.

La importancia de encontrar rápidamente cotas superiores para la función objetivo se discutió y en efecto, se desarrollaron tres heurísticas. La Heurística 3 es aplicable solo a una clase especial de problemas; por lo cual la ventaja computacional de las otras dos heurísticas es la que se ha podido probar. Comparando los tiempos finales de solución la Heurística 2 (opción 3) es casi tan rápida como aquellos casos en que no se usan heurísticas. La heurística 1 (opción 2) resultó peor que la 2. Sin embargo, los resultados fueron drásticamente diferentes cuando los tiempos de solución para obtener las cotas se compararon. La Heurística 2 pudo encontrar cotas superiores en menos del 60% del tiempo promedio de los tiempos de solución sin heurísticas, la heurística 1 no fue tan rápida, sin embargo se pudieron encontrar cotas en menos del 90% del tiempo promedio de las soluciones sin heurísticas. De éstas observaciones se concluye que la aplicación de la Heurística 2 mejora significativamente la aptitud de resolver problemas con los procedimientos de ramificación y acotamiento.

Experimentos adicionales demostraron que los parámetros del problema tales como el número de variables enteras y restricciones tienen un efecto significativo en los tiempos de solución. Un análisis de regresión estadística demostró que el número de variables enteras tienen un efecto cuadrático, y el número de restricciones tienen un efecto en los tiempos de solución.

## RECOMENDACIONES

En los problemas que se probaron se usó el código OPT basado en el método GRG para resolver los problemas no lineales, sin embargo sería interesante estudiar la factibilidad de usar ramificación y acotamiento con otros códigos GRG.

El algoritmo de ramificación y acotamiento también se puede compara con alguna de las otras técnicas usadas para resolver problemas enteros. Y por otro parte se puede extender a problemas de optimización discreta no lineal. Las reglas de ramificación se pueden modificar para que solo valores discretos se puedan considerar para las variables discretas correspondientes. Las heurísticas, los conceptos de pseudo-costos y las estimaciones pueden no ser aplicables en el caso discreto.

Finalmente habría que señalar que el método de ramificación y acotamiento requiere condiciones propias de convexidad-concavidad para garantizar optimalidad.

## REFERENCIAS AL CAPITULO IV.

1. ABADIE, J. "Integer and Nonlinear Programming", North Holland Publishing Company, 1970.
2. BAZARAA, M. S. "Nonlinear programming" Wiley, 1979.
3. GUPTA, O.K. y RAVINDRAN, A. "Nonlinear Integer Programming Algorithms: A Survey", Opsearch 20, No. 4, p. 189-206 1983.
4. GUPTA, O.K. y RAVINDRAN, A. "Branch and Bound Experiments in Convex Nonlinear Integer Programming", Man. Sci. 31, No. 12 p. 1533-1546, 1985.
5. GUPTA, O.K. "Branch and Bound Experiments in Nonlinear Programming" Ph D. Thesis, School of Industrial Engineering, Purdue University, West Lafayette, Ind., 1980.
6. KENNEDY, D. "Some Branch and Bound Techniques for Nonlinear Optimization", Math. Program. 42, p. 147-157, 1988.
7. LUENBERGER, D.E. "Programación Lineal y No Lineal", Addison-Wesley Iberoamericana, 1989.
8. WARREN, A.D. y LASDON, L. S. "The Status of Nonlinear Programming Software, Ops. Res. 27, No.3, p. 431-457, 1979.

## CAPITULO V

### APLICACIONES : EL PROBLEMA DEL AGENTE VIAJERO Y EL PROBLEMA DE LA MOCHILA.

*"Tiresias me ordenó que recorriera muchísimas ciudades, llevando en la mano un manejable remo, hasta llegar a aquellos hombres que nunca vieron el mar, ni comen manjares sazonados con sal, ni conocen las naves de púrpuros flancos, ni tienen noticia de los manejables rentos que son como las alas de los bajetes."*

Homero. La Odisea.

En el pasado reciente, los algoritmos de ramificación y acotamiento se han formulado para resolver una amplia variedad de problemas de optimización combinatoria. Entendiendo por problema combinatorio a aquél que asigna valores numéricos discretos a algún conjunto finito de variables  $X$ , de tal forma que satisfaga un conjunto de restricciones y mínimice alguna función objetivo. Dos de tales problemas, tales como el del agente viajero y el de la mochila los proporcionamos como representativos no del rango de todas las posibles problemas combinatorios, pero sí del tipo de problemas a donde se puede aplicar ramificación y acotamiento. La función objetivo que puede ser no lineal, discontinua o no necesariamente definida matemáticamente, se puede resolver con R-A.

Problemas como el del agente viajero o el de la mochila se conocen como no polinomiales completos o NP-completos. Un problema se dice polinomial si existe un algoritmo para el cual el tiempo requerido para su solución, está acotado por una función polinomial del tamaño del problema (donde entendemos el tamaño del problema como la longitud de un código, por ejemplo binario de los datos del problema). Se tiene así por ejemplo que en una gráfica  $G = \{X, A\}$  con  $N = X$  nodos y  $M = A$  arcos, una ruta más corta se encuentra a lo mas en un tiempo  $O(MN)$ , un flujo máximo en un tiempo  $O(N^3)$ , un árbol de peso mínimo en un tiempo  $O(N^2)$ .

Un algoritmo se dice no determinístico si contiene afirmaciones que permiten una selección además de las afirmaciones usuales (determinísticas). La clase de problemas que se pueden resolver en tiempo polinomial por algoritmos no determinísticos se llaman de clase NP, en otras palabras, si para cada instrucción seleccionada, la selección es correcta, entonces el tiempo de computadora es polinomial. Por otro lado si se enumeran las posibles selecciones el algoritmo se vuelve determinístico pero con tiempo de computadora exponencial. Un problema NP-completo es aquel para el cual las únicas soluciones conocidas consisten, en esencia, en intentar todas las posibilidades. Tanto en el caso del problema del agente viajero como en el de la mochila los algoritmos que se basan en el método de R-A tienen un tiempo de computadora exponencial.

Este capítulo se desarrolla como sigue: en la primera sección se describe el problema del agente viajero, en la segunda sección se describe el algoritmo de ramificación y acotamiento que se utiliza en la solución del problema del viajero, en la tercera sección se desarrolla un ejemplo que se programó en computadora, en la cuarta sección se describe el problema de la mochila, en la quinta sección se exponen el algoritmo de R-A desarrollado por Kolesar y el de Martello y Toth para el problema de la mochila y se resuelve un ejemplo, para cada uno.

## 5.1 EL PROBLEMA DEL AGENTE VIAJERO

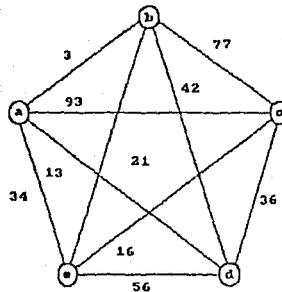
El problema del agente viajero es un problema clásico en optimización combinatoria, y uno de los problemas más viejos en la optimización de redes que ha atraído mucho la atención en los últimos cuarenta años. Cuenta con numerosas aplicaciones, por ejemplo, en problemas de distribución donde se tiene que el almacén central de una compañía desea distribuir materia prima a cada una de sus sucursales a un costo mínimo o problemas de carteros, ¿cómo debe el cartero atravesar la ciudad para repartir la correspondencia minimizando el tiempo de viaje?. O considere otra situación: una fábrica produce  $n$  artículos y se debe cambiar el montaje siempre que cambia la producción de artículos, conociendo el costo de montaje entre los artículos se desea encontrar una secuencia de producción óptima.

Plantear el problema es muy sencillo y lo haremos a continuación:

Un agente viajero debe visitar cada ciudad de su territorio exactamente una vez y regresar a su punto de partida. Dado el costo del viaje entre cada par de ciudades, ¿cómo debe plantear su itinerario de tal forma que visite cada ciudad una sola vez y el costo total del recorrido sea mínimo?

En términos de la teoría de redes, el problema consiste en encontrar en una gráfica completa de  $n$  nodos, un circuito de peso mínimo de longitud  $n$ . Recuérdese que una gráfica completa es aquella en la cual entre cualquier par de nodos existe un arco que los une. Considere por ejemplo, un problema con cinco ciudades, como se muestra en la figura 5.1

FIGURAS.1



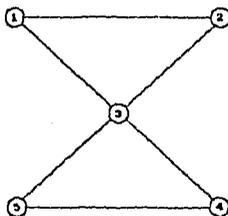
El peso del circuito o costo de la ruta (a, b, c, d, e, a) es 206, mientras que el circuito de peso mínimo es (a, b, e, c, d, a) con peso 89.

Hay varias variantes a éste problema, pero primero comenzaremos por precisar conceptos:

Sea  $W = [w_{ij}]$  el peso de la matriz de la gráfica, entonces se puede tener que los pesos en los arcos pueden no ser simétricos  $w_{ij} = w_{ji}$ , en cuyo caso estamos tratando con una gráfica dirigida y resolviendo un problema del agente viajero asimétrico. O algunas veces la gráfica dada no es completa, lo que significa que existen parejas de nodos que no están directamente conectados por arcos. No obstante, que toda gráfica completa siempre tiene circuitos de longitud  $n$ , una que no lo es puede no tener circuitos de longitud  $n$ . Por ejemplo la gráfica en la figura 5.2 no tiene circuitos de longitud 5, en tal caso el problema del agente viajero no tiene solución.

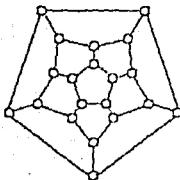
ESTA TERCERA  
SALIR DE LA BIBLIOTECA

FIGURAS.2



El problema de determinar si una gráfica dada con  $n$  nodos tiene un circuito de longitud  $n$  se conoce como un problema de circuito hamiltoniano, originado en un juego inventado por el matemático irlandés Sir William Rowan Hamilton en 1859. Trataba sobre un viaje alrededor de mundo, el cual se representaba en forma simplificada por un dodecaedro (poliedro de 12 caras pentagonales con 20 vértices), y se requiere que se pase una sola vez por cada vértice o ciudad, usando solamente las caras del dodecaedro y se regrese al punto inicial. Este juego es equivalente a buscar un circuito hamiltoniano en la figura 5.3

FIGURAS.3

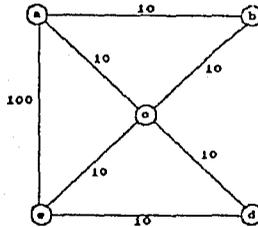


Un circuito que pasa por cada nodo en la gráfica es llamado un circuito hamiltoniano.

Si el agente viajero no quiere regresar al nodo inicial, después de visitar todos los nodos en la gráfica exactamente una vez, el problema se convierte entonces en encontrar la trayectoria de peso mínimo de longitud  $(n-1)$  y no un circuito de longitud  $n$ .

Se podrá notar un cambio muy drástico si al resolver éste problema si se permite que el agente viajero visite un nodo mas de una vez (si ésto es más barato). Por ejemplo en la figura 5.4 se tiene que si se permite que visite un nodo mas de una vez la ruta óptima (a, b, c, d, e, c, a) que además no es un circuito tiene un peso de 60, pero si no se permite que visite un nodo mas de una vez, la ruta (a, b, c, d, e, a) es la única solución posible, con un peso de 140.

FIGURA 5.4



La condición bajo la cual al menos uno de los circuitos hamiltonianos tenga un circuito de peso mínimo es que los pesos satisfagan la desigualdad del triángulo  $w_{ij} \leq w_{ik} + w_{kj}$ .

Una forma de resolver el problema del agente viajero es haciendo una enumeración exhaustiva de todas las soluciones posibles y entre ellas escoger la mejor. Hay  $n!$  permutaciones de los  $n$  nodos, pero solo  $(n-1)!$  de ellas son circuitos hamiltonianos distintos (en una gráfica dirigida completa), porque podemos fijar alguno de los  $n$  nodos, y posteriormente hay  $(n-1)!$  formas de acomodar los  $(n-1)$  nodos restantes en el circuito. En caso de que la red dada no sea una red dirigida entonces hay  $(n-1)!/2$  circuitos distintos.

Teóricamente, el problema se puede resolver generando los  $(n-1)!$  circuitos y comparar sus pesos, sin embargo como método éste es muy ineficiente. Por ejemplo, dada una gráfica de 20 nodos se tienen  $19! > 10^{17}$  circuitos, por lo que se requieren años de cálculos continuos para resolver el problema.

Como se podrá ver el problema del agente viajero es uno de los problemas de optimización combinatoria para los cuales no se conoce un algoritmo eficiente de tiempo polinomial. Todos los algoritmos conocidos requieren de un tiempo de computadora exponencial en el número de ciudades  $n$ .

Para salvar ésta dificultad computacional hay dos formas de resolverla:

1. Usando técnicas refinadas tales como ramificación y acotamiento o programación dinámica, que reducen drásticamente el efecto que se da en enumeración exhaustiva. Tales técnicas refinadas enumerativas son buenas para encontrar una solución óptima, pero en el peor de los casos si se tiene un problema muy grande pueden requerir un número exponencial de cálculos que se hacen prohibitivamente grandes.

2. Empleando métodos de solución aproximados pero rápidos (en tiempo polinomial), que no producen una solución óptima, pero sí soluciones subóptimas que estén aceptablemente cercanas a la óptima.

Ambos métodos son útiles, sin embargo para nuestros propósitos sólo discutiremos el caso de ramificación y acotamiento.

## 5.2 ALGORITMO DE RAMIFICACION Y ACOTAMIENTO

"Entre éstos dos caminos vacilo continuamente, y cuando siento que he explorado al máximo las posibilidades de uno, me lanzo al otro y viceversa"

Italo Calvino. Seis Propuestas para el Próximo Milenio.

Un método de ramificación y acotamiento para un problema en particular queda definido al especificar sus operaciones de R-A. En el caso del problema del viajero, existen tres métodos de ramificación y acotamiento :

1. Construcción del circuito según la matriz reducida.
2. Eliminación de subcircuitos a partir de la solución de problemas de asignación.
3. Construir la trayectoria basandose en el árbol de expansión mínima.

En nuestro caso solo desarrollaremos el primero, pues no es nuestra intención hacer un análisis exhaustivo del problema del agente viajero, quienes deseen ver mas sobre el mismo pueden consultar [15]

### METODO DE RAMIFICACION Y ACOTAMIENTO BASADO EN LA MATRIZ REDUCIDA

El funcionamiento de éste método se basa en la construcción del circuito hamiltoniano según la matriz reducida. Para ilustrar la idea de éste método, veamos el siguiente ejemplo:

#### EJEMPLO 5.1

Sea la matriz de tiempos D que se presenta en la Tabla 5.1, entonces teniendo el tiempo de duración de cada uno de los viajes entre las distintas paradas, se desea encontrar el recorrido hamiltoniano de menor duración.

TABLA 5.1

	A	B	C	D	E	F
A	$\infty$	27	43	16	30	26
B	7	$\infty$	16	1	30	25
C	20	13	$\infty$	35	5	0
D	21	16	25	$\infty$	18	18
E	12	46	27	28	$\infty$	5
F	23	5	5	9	5	$\infty$

Si T es la duración de un recorrido hamiltoniano asociado a la matriz de tiempos de la tabla 5.1 entonces la duración de ese mismo recorrido con la matriz de tiempos obtenida de restar un escalar  $h_i$  al renglón i está dado por  $T-h_i$ , porque cada circuito contiene uno y solo un elemento de éste renglón. Lo mismo sucede si restamos un escalar  $h^j$  de una columna j ( $j=A,B,C,D,E,F$ ). Una cota inferior del recorrido hamiltoniano óptimo es sencilla de obtener si restamos de cada renglón de la matriz de tiempos D, el mínimo elemento correspondiente. O bien

$$h_i = \min_j d_{ij} \geq 0$$

y entonces, en la matriz D' así reducida, restamos de cada columna j la constante

$$h^j = \min_i d_{ij} \geq 0$$

La nueva matriz  $D^1$ , tiene elementos no-negativos, y contiene al menos un elemento cero en cada renglón y cada columna. Como se muestra en la tabla 5.2

TABLA 5.2

	A	B	C	D	E	F
A	$\infty$	11	27	0	14	10
B	1	$\infty$	15	0	29	24
C	15	13	$\infty$	35	5	0
D	0	0	9	$\infty$	2	2
E	2	41	22	23	$\infty$	0
F	13	0	0	4	0	$\infty$

$$\text{Tomamos } h = \sum_i h_i + \sum_j h^j$$

La duración  $z(t)$  de un circuito hamiltoniano  $t$  de la matriz  $D$  es entonces igual a  $z(t) = h + z'(t)$ , donde  $z'(t)$  es la duración del circuito  $t$  de la matriz  $D^1$ . Ya que  $z'(t)$  (porque  $d_{ij}^1 \geq 0$ ), tenemos  $z(t) \geq h$  donde  $h$  es una evaluación por cota inferior del conjunto  $\Omega$  de circuitos hamiltonianos y se denota  $cv(\Omega)$ .

De la matriz dada  $D$  obtenemos

$$h_1 = 16, h_2 = 1, h_3 = 0, h_4 = 16, h_5 = 5, h_6 = 5$$

$$\text{y } h^1 = 5, h^2 = h^3 = h^4 = h^5 = h^6 = 0$$

Un aspecto importante es que cada recorrido hamiltoniano asociado a la Tabla 5.2 tiene una duración no negativa y que difiere en tiempo del recorrido original en 48 unidades de tiempo. De donde una cota inferior de la duración del recorrido mínimo es 48 o bien

$$cv(\Omega) = \sum_i h_i + \sum_j h^j = 16 + 1 + 16 + 5 + 5 + 5 = 48$$

Una manera de proceder a la determinación del recorrido hamiltoniano de mínima duración es particionar el conjunto de recorridos hamiltonianos como sigue:

- Recorridos hamiltonianos que usan el arco AD
- Recorridos hamiltonianos que no usan el arco AD

En el primer caso la matriz de tiempos entre localidades se reduce a una nueva matriz en donde se elimina al renglón A y la columna D. Asimismo, el tiempo entre la localidad D a la A se hace igual a infinito o a un número muy grande para evitar usar el arco DA; pues sabemos que no forma parte del recorrido hamiltoniano mínimo. Si no hacemos éste tiempo infinito, existe la posibilidad de la aparición de subcircuitos. La tabla 5.3 muestra la matriz reducida donde se han eliminado el renglón A y la columna D.

TABLA 5.3

	A	B	C	E	F
B	1	$\infty$	15	29	24
C	15	13	$\infty$	5	0
D	0	0	9	2	2
E	2	41	22	$\infty$	0
F	13	0	0	0	$\infty$

Una cota inferior de la duración de los recorridos hamiltonianos asociados con esta tabla es sencilla de obtener si restamos una unidad a cada elemento del renglón uno como se muestra en la tabla 5.4

TABLA 5.4

	A	B	C	E	F
B	0	$\infty$	14	28	23
C	15	13	$\infty$	5	0
D	0	0	9	2	2
E	2	41	22	$\infty$	0
F	13	0	0	0	$\infty$

Cabe hacer notar que como resultado de las manipulaciones anteriores podemos decir que los recorridos hamiltonianos asociados a la tabla 5.1, que usen el arco AD tiene una duración no menor de 49 unidades de tiempo, 48 acumuladas hasta la obtención de la tabla 5.2 y una unidad de tiempo al pasar de la tabla 5.3 a la tabla 5.4, por lo que 49 unidades es una cota inferior a la duración de los recorridos hamiltonianos que usan el arco AD.

Una cota inferior a la duración de los recorridos hamiltonianos que no usan el arco AD es sencilla de obtener, suponga que en general un arco  $(i, j)$  con  $d_{ij} = 0$  no se escoge. Ya que debemos dejar el punto  $i$ , debemos usar algún arco comenzando en  $i$  y penalizarlo por  $\alpha_i$ . También debemos usar algún arco en  $j$ , y podríamos penalizarlo por  $\beta_j$ , con lo cual  $\theta_{ij} = \alpha_i + \beta_j$  es la penalización de no escoger  $(i, j)$ . Dicha penalización  $\theta_{ij}$  se puede interpretar en éste caso como un retardo.

Si  $ev(S)$  es la evaluación obtenida reduciendo la matriz por el nodo  $S$ , entonces  $ev(S) + \theta_{ij}$  es una primer evaluación por cota inferior del nodo  $ij$ , obtenido de  $S$  por no escoger el arco  $(i, j)$

Calculamos el retardo  $\theta_{ij}$  correspondiente a  $d_{ij} = 0$

$$\begin{aligned} \theta_{AD} &= \alpha_A + \beta_D = 10 + 0 = 10 \\ \theta_{BD} &= \alpha_B + \beta_D = 1 + 0 = 1 \\ \theta_{CF} &= \alpha_C + \beta_F = 5 + 0 = 5 \\ \theta_{DA} &= \alpha_D + \beta_A = 0 + 1 = 1 \\ \theta_{DB} &= \alpha_D + \beta_B = 0 + 0 = 0 \\ \theta_{EF} &= \alpha_E + \beta_F = 2 + 0 = 2 \\ \theta_{FB} &= \alpha_F + \beta_B = 0 + 0 = 0 \\ \theta_{FC} &= \alpha_F + \beta_C = 0 + 9 = 9 \\ \theta_{FE} &= \alpha_F + \beta_E = 0 + 2 = 2 \end{aligned}$$

Separamos la pareja  $(i, j)$  que maximiza el retardo  $\theta_{ij}$  para garantizar una mayor cota inferior de la duración de los recorridos hamiltonianos:  $\theta_{AD} = 10$

Entonces el nodo AD tiene una evaluación por cota inferior igual a  $48 + 10 = 58$

Una cuestión que conviene analizar en éste momento es: ¿por qué se efectuó la ramificación sobre el arco AD y no sobre otro que tuviese el tiempo entre localidades igual a cero en la tabla 5.2? La razón es la siguiente: estrictamente se debió analizar para cada arco (i, j) con tiempo entre localidades igual a cero en la tabla 5.2 el retardo correspondiente asociado al eliminar ese arco y elegir el que tenga máximo retardo (que es el arco AD) para garantizar una mayor cota inferior de la duración de los recorridos hamiltonianos.

El siguiente paso consiste en calcular los retardos correspondientes a la tabla 5.4

$$\begin{aligned} \theta_{BA} &= 14+2 = 16 \\ \theta_{CF} &= 5+0 = 5 \\ \theta_{DB} &= 2+0 = 2 \\ \theta_{EF} &= 2+0 = 0 \\ \theta_{FB} &= 0+0 = 0 \\ \theta_{FC} &= 0+9 = 9 \\ \theta_{FE} &= 0+2 = 2 \end{aligned}$$

Se escoge el arco BA para efectuar la ramificación como sigue:

- a. Recorridos hamiltonianos que usan el arco BA
- b. Recorridos hamiltonianos que no usan el arco BA

En el primer caso partimos de la tabla 5.4 eliminando el renglón B y la columna A y haciendo el tiempo de A a B igual a infinito para evitar circuitos innecesarios. En éste momento, como AD y BA se han seleccionado para formar el recorrido hamiltoniano hacemos que el tiempo de DA sea infinito, la tabla 5.5 exhibe ésta situación

TABLA 5.5

	B	C	E	F
C	13	$\infty$	5	0
D	$\infty$	9	2	2
E	41	22	$\infty$	0
F	0	0	0	$\infty$

Con el propósito de tener un elemento cero en cada renglón y columna así como mejorar la cota inferior de los recorridos hamiltonianos procedemos a restar dos unidades del renglón dos en la tabla 5.5 para obtener:

TABLA 5.6

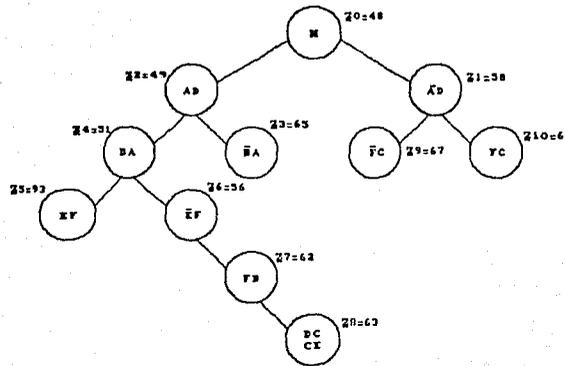
	B	C	E	F
C	13	$\infty$	5	0
D	$\infty$	9	0	0
E	41	22	$\infty$	0
F	0	0	0	$\infty$

Y se puede observar que una cota inferior de los recorridos hamiltonianos que usan BA es 51 unidades de tiempo.

Una cota inferior de la duración de los recorridos que no usan el arco BA es 51 unidades de tiempo. Una forma esquemática de las ramificaciones y anotaciones desarrolladas sobre los recorridos hamiltonianos se tiene en la figura 5.5 en donde se muestran el resto de las operaciones de manera resumida para llegar a demostrar que el recorrido hamiltoniano óptimo es A-D-C-E-F-B-A con una duración de 63 unidades de tiempo.

Para verificar que ésta solución es óptima, debemos examinar el vértice AD cuya evaluación por cota inferior es 58 (<63). La separación del vértice AD produce: usando FC una cota de 63 y sin usar FC una cota de 67, por lo cual el circuito propuesto es óptimo.

FIGURAS.5



La formalización del algoritmo es la siguiente:

#### ALGORITMO 5.1: LITTLE-MURTY

PROPOSITO: Determinar el circuito hamiltoniano de costo mínimo dada una matriz de costos. El método utiliza la propiedad de la matriz de costos reducida para probar la inclusión o exclusión de un arco en el circuito.

#### DESCRIPCION

Paso 1. Dada la matriz de costos  $D$ , se efectúan sustracciones en los renglones y las columnas de la matriz  $D$ , sin permitir que aparezcan valores negativos. Con esto obtenemos una cota inferior del problema del viajero al sumar los elementos que se restaron a los renglones y a las columnas. La matriz  $D'$  es la matriz reducida de  $D$ .

Paso 2. Sea  $S$  un nodo del árbol y  $cv(S)$  la cota inferior de éste nodo  $S$ , con cada arco  $(i, j)$  con  $d_{ij}' = 0$ , debemos usar algún arco comenzando en  $i$  y penalizarlo por  $\alpha_i$ . También debemos usar algún arco en  $j$ , y podríamos penalizarlo por  $\beta_j$ , con lo cual  $\theta_{ij} = \alpha_i + \beta_j$  es la penalización de no escoger  $(i, j)$ . Se selecciona el arco que tiene el máximo de los  $\theta_{ij}$ .

Paso 3. Si el arco  $(i, j)$  no se selecciona,  $cv(S) + \theta_{ij}$  es una cota inferior. Si el arco  $(i, j)$  se seleccionó, entonces la matriz se reduce omitiendo el renglón  $i$  y la columna  $j$ . Buscar la condición adicional sobre  $D'$  para excluir subcircuitos posibles.

Paso 4. Seleccione el vértice que tiene menor costo y vaya al paso 1.

### 5.3 EL PROBLEMA DE LA MOCHILA

Muchos problemas industriales se pueden formular como problemas de tipo mochila, por ejemplo problemas de cargo fijo, selección de proyectos, corte en inventarios, control de presupuestos, etc. La versión más popular del problema contiene sólo una restricción lineal, pero casi cualquier problema lineal entero y muchos otros problemas combinatorios se pueden reducir a él. El problema de la mochila se presenta también como un subproblema en varios algoritmos de programación lineal pura y entera.

Hay muchas versiones distintas del problema de la mochila, en nuestro caso consideraremos el problema de la mochila 0-1 que se expresa de la siguiente manera:

$$\text{Maximizar } \sum_{i=1}^n p_i x_i \quad (5.1)$$

Sujeto a

$$\sum_{i=1}^n w_i x_i \leq W \quad (5.2)$$

$$x_i = 0 \text{ o } 1 \quad i = 1, 2, \dots, n \quad (5.3)$$

donde  $p_i$ ,  $w_i$  ( $i = 1, 2, \dots, n$ ) y  $W$  son números enteros. En otros términos, suponga que se tiene que llenar una mochila con diferentes objetos con un beneficio  $p_i$  y peso  $w_i$  sin exceder un peso total dado  $W$ . El problema consiste en encontrar una asignación factible de objetos para que el valor total de los objetos en la mochila sea el máximo.

El problema de la mochila 0-1 es un caso especial del problema de la mochila acotado, que se define igual que el anterior y solo difiere en la restricción (5.3) ya que en éste caso se tiene

$$0 \leq x_i \leq b_i, \text{ donde } x_i \text{ es entero, } i = 1, 2, \dots, n$$

En el problema de la mochila acotado, la mochila se puede llenar con a lo más  $b_i$  objetos del tipo  $i$ . En el problema general de la mochila, que a veces se denomina no acotado, la restricción (5.3) se relaja a

$$x_i \geq 0, x_i \text{ entero, } i = 1, 2, \dots, n$$

Sin pérdida de generalidad podemos suponer que los parámetros  $p_i$ ,  $w_i$  y  $W$  en los problemas anteriores satisfacen las condiciones:

$$p_i \text{ y } w_i \text{ son enteros positivos} \quad i = 1, 2, \dots, n$$

$$w_i \leq W \quad i = 1, 2, \dots, n$$

$$\sum_{i=1}^n w_i > W$$

Los problemas de la mochila 0-1, acotado y generalizado a veces se conocen como problemas unidimensionales, donde el uno se refiere al número de restricciones lineales en el problema. Los más populares son los de valor independiente (cuando  $w_i = p_i$ ) y el problema de hacer cambios, éste problema consiste en encontrar el menor número

de monedas de tipos o valores especificados  $w_i$  que constituyen exactamente un cambio dado  $V$ . Suponemos que se dispone de cada tipo de moneda en una cantidad ilimitada, formalmente el problema es

$$\text{Minimizar } \sum_{i=1}^n x_i$$

Sujeto a

$$\sum_{i=1}^n x_i w_i = W$$

$$x_i \geq 0, x_i \text{ entero } i=1, \dots, n$$

Observese que como la restricción en éste problema es de igualdad no siempre existe solución, a menos que alguna de las monedas disponibles valga 1.

Los problemas tipo mochila unidimensionales se pueden generalizar de muchas maneras, la generalización más natural es aquella en que los objetos que tenemos que guardar pueden ponerse en  $m$  mochilas, cada una con capacidad  $W_j$  ( $j=1, 2, \dots, m-1$ ). Sea  $x_{ij}$  una variable 0-1 tal que  $x_{ij}=1$  si el  $i$ -ésimo objeto se asigna a la  $j$ -ésima mochila. El problema 0-1 multimochila, se expresa como

$$\text{Maximizar } \sum_{i=1}^n \sum_{j=1}^m p_i x_{ij}$$

Sujeto a

$$\sum_{i=1}^n w_j x_{ij} \leq W_j \quad j=1, 2, \dots, m$$

$$\sum_{j=1}^m x_{ij} \leq 1 \quad i=1, 2, \dots, n$$

$$x_{ij} = 0 \text{ o } 1 \quad i=1, \dots, n; j=1, \dots, m$$

La primera restricción significa que en una restricción factible de objetos no se sobrecarga ninguna mochila y la segunda, que cada objeto puede asignarse a lo más a una mochila, pueden formularse de aquí las versiones acotada y no acotada de éste problema.

Si antes de poner los objetos en una mochila se tienen que comprar, a un costo  $c_i$  para el  $i$ -ésimo objeto, y se tiene una cantidad limitada de dinero  $C$ , se tiene entonces el problema de asignar objetos a la mochila que no pesen más de  $W$  y no cuesten más de  $C$ , entonces el problema se convierte en

$$\text{Maximizar } \sum_{i=1}^n p_i x_i$$

Sujeto a

$$\sum_{i=1}^n w_i x_i \leq W$$

$$\sum_{i=1}^n c_i x_i \leq C$$

$$x_i = 0 \text{ o } 1 \quad i = 1, 2, \dots, n$$

En general, se pueden introducir muchas restricciones al asignar objetos a una mochila, el problema se convierte entonces en un problema de la mochila multidimensional, donde evidentemente se pueden considerar los casos acotado y no acotado.

Los problemas de tipo mochila a menudo se refieren como problemas de carga, pero de hecho, el problema de carga estándar consiste en asignar objetos dados con volúmenes conocidos a cajas, que tienen restricciones de capacidad, con el objeto de minimizar el número de cajas usadas. Sea  $k_j$  la capacidad de la  $j$ -ésima caja y  $w_i$  el volumen del  $i$ -ésimo objeto. El problema de carga se define como sigue:

$$\text{Minimizar } \sum_{j=1}^m y_j$$

Sujeto a

$$\sum_{j=1}^m x_{ij} = 1 \quad i = 1, 2, \dots, n$$

$$\sum_{i=1}^n w_i x_{ij} \leq k_j \quad j = 1, 2, \dots, m$$

$$y_j, x_{ij} = 0 \text{ o } 1 \quad i = 1, \dots, n; \quad j = 1, \dots, m$$

En un cargo factible, tenemos  $x_{ij} = 1$  si el  $i$ -ésimo objeto se pone en la  $j$ -ésima caja, y  $y_j = 1$  si se usa la  $j$ -ésima caja.

El problema de la mochila acotado o no acotado puede también representar el problema de cortar objetos unidimensionales (por ejemplo la longitud de un papel, vidrio y acero) en piezas pequeñas de valores y tamaños dados para maximizar el valor total de las piezas o minimizar el material que sobra.

Este pequeño panorama de las posibles generalizaciones y modificaciones del problema de la mochila 0-1 indica la variedad de problemas del mundo real que se pueden modelar por problemas que provienen de la familia de problemas de tipo mochila.

#### REDUCCION DEL PROBLEMA LINEAL ENTERO AL PROBLEMA DE LA MOCHILA.

Cada sistema de ecuaciones lineales con coeficientes enteros se pueden transformar en una sola ecuación lineal que tiene el mismo conjunto de soluciones enteras no negativas que el correspondiente sistema lineal entero. Entonces, las restricciones de un problema lineal entero se pueden primero transformar en una única restricción y así resolverlo como un problema de tipo mochila.

El paso básico del proceso de transformación agrega dos ecuaciones de tal forma que el conjunto de soluciones enteras no negativas no se altera. Describiremos un método que supone todas las variables acotadas.

Sean las dos ecuaciones enteras en variables acotadas

$$g_j(x) = b_j - \sum_{i=1}^n a_{ji} x_i = 0 \quad j=1,2 \quad (5.4)$$

$$0 \leq x_i \leq u_i, x_i \text{ es entero} \quad i=1,2,\dots,n$$

Determinaremos ahora los multiplicadores válidos  $\alpha_1$  y  $\alpha_2$  tales que son distintos de cero y la ecuación

$$\alpha_1 g_1(x) + \alpha_2 g_2(x) = 0 \quad (5.5)$$

implica que  $g_1(x) = g_2(x) = 0$

ya que (5.5) es equivalente a  $g_1(x) + (\alpha_2/\alpha_1)g_2(x) = 0$ , podemos preguntar por sólo un multiplicador  $\alpha$  tal que

$$g_1(x) + \alpha g_2(x) = 0 \quad (5.6)$$

implica que  $g_1(x) = g_2(x) = 0$

Usando cotas superiores en las variables,  $g_j(x)$  ( $j=1,2$ ) se pueden acotar de la siguiente manera:

$$b_j - \sum_{i=1}^n a_{ji}^+ u_i \leq g_j(x) \leq b_j - \sum_{i=1}^n a_{ji}^- u_i$$

donde  $a_{ji}^+ = \max(a_{ji}, 0)$  y  $a_{ji}^- = \min(a_{ji}, 0)$ .

#### Teorema 5.1

El vector entero  $x$  que satisface  $0 \leq x \leq u$  es una solución de (5.4) si y sólo si es una solución de (5.6) con  $\alpha$  que satisface,

$$\alpha > \max \{b_1 - a_{11} u_1, -b_1 + a_{11}^+ u_1\}$$

#### Ejemplo 5.1

Aplicaremos el Teorema 5.1 al siguiente sistema de ecuaciones:

$$\begin{aligned} 4x_1 - x_2 + 2x_3 + x_4 + x_5 &= 0 \\ x_1 - x_3 - x_4 - x_6 &= -1 \\ x_i &= 0 \text{ o } 1 \quad i = 1, 2, \dots, 6 \end{aligned}$$

Por la enumeración de todos los posibles vectores 0-1  $(x_1, x_2, x_3, x_4, x_5, x_6)$  encontramos las únicas soluciones de éste sistema  $(0, 1, 0, 1, 0, 0)$ ,  $(0, 1, 0, 0, 1, 1)$  y  $(0, 0, 0, 0, 0, 1)$ . Primero tratamos con dos multiplicadores arbitrarios  $\alpha_1 = 1$  y  $\alpha_2 = 1$ . La ecuación agregada es de la forma

$$5x_1 - x_2 + x_3 + x_5 - x_6 = -1$$

y se tienen las tres soluciones anteriores mas  $(0, 1, 0, 0, 0, 0)$ ,  $(0, 0, 0, 1, 0, 1)$ ,  $(0, 1, 1, 0, 0, 1)$  y  $(0, 1, 0, 1, 1, 1)$ . Sin embargo aplicando el Teorema 5.1 se encuentra que  $\alpha$  debe ser mayor que 8. Para  $\alpha = 9$ , la ecuación válida agregada es de la forma

$$13x_1 - x_2 - 7x_3 - 8x_4 + x_5 - 9x_6 = -9$$

Una vez que sabemos como agregar dos ecuaciones en una, podemos aplicar éste proceso iterativamente a un sistema de ecuaciones arbitrario. Este método es de uso limitado para programas lineales enteros debido al rápido crecimiento en los valores de los coeficientes conforme se van agrgando restricciones.

## METODOS COMPUTACIONALES

Desde el punto de vista computacional, el problema de la mochila es un problema difícil. No se ha encontrado un algoritmo de tiempo polinomial para él, y es muy poco probable que tal algoritmo exista. Se han propuesto varias técnicas como métodos de solución para éstos problemas como las siguientes:

1. Métodos de Reducción y Aproximación (del tipo gradiente y Lagrangeano).
2. Métodos exactos
  - a. Redes
  - b. Programación dinámica
  - c. Métodos de Enumeración (Ramificación y Acotamiento)

Los métodos del tipo gradiente generalizado son eficientes computacionalmente sólo cuando se requieren soluciones aproximadas.

Los algoritmos de programación dinámica son eficientes cuando el valor de  $W$  es pequeño, cuando éste crece tiende a ser ineficiente porque se requiere un almacenamiento excesivo, mientras que las técnicas de enumeración resultan menos afectadas por ésta desventaja. El caso de la solución a través de redes formulan el problema de la mochila como un problema de ruta más corta, resultan también ineficientes porque se producen redes de tamaño muy grande

En éste trabajo sólo analizaremos el problema a través de el método de ramificación y acotamiento como veremos a continuación.

### 5.4 METODO DE RAMIFICACION Y ACOTAMIENTO

El primer método de ramificación y acotamiento que se usó para resolver el problema de la mochila es debido a Peter J. Kolesar (1967), éste método usa la estrategia de búsqueda por primer amplitud; el gran uso de memoria de computadora y los requerimientos de tiempo de éste algoritmo se redujeron mucho con el método de Greenberg y Hegerich (1970) que usa la estrategia de primer profundidad. Posteriormente Horowitz y Sahni (1974) propusieron un procedimiento de ramificación y acotamiento altamente eficiente, basado en el esquema de Greenberg-Hegerich, este mismo algoritmo se obtuvo independientemente por Ahrens y Finke (1975), y se han presentado más perfeccionamientos por Barr y Ross (1975), Fayard y Plateau (1975), Zoltners (1978), Nauss (1976), Suhl (1977), Martello y Toth (1977).

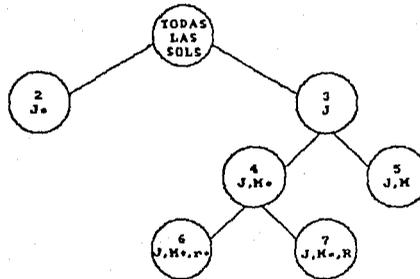
A continuación, se expone el algoritmo de Kolesar y posteriormente el algoritmo de Martello y Toth (1977).

#### ALGORITMO DE KOLESAR

El algoritmo que se propone es un algoritmo de ramificación y acotamiento en el sentido de Little, et al.

Este método considera que un nodo lleva un índice  $i$  si el artículo se incluye y un índice  $i^*$  si no se incluye. Un nodo con índice  $(i, j)$  significa que se incluye el artículo  $i$  primero y después el artículo  $j$ , mientras que el índice  $(i^*, j)$  significa que el artículo  $i$  no se incluye pero el  $j$  sí como se muestra en la figura 5.7

FIGURAS.7



Se denota por  $B(n)$  la cota superior asociada con el nodo  $n$ . Dado el nodo  $n$  tenemos tres categorías de artículos:

Los artículos incluidos  $I_n$  es el conjunto de artículos que están incluidos explícitamente en las soluciones contenidas en el nodo  $n$ ; los artículos excluidos  $E_n$  como su nombre lo dice están explícitamente excluidos de las soluciones contenidas en el nodo  $n$  y finalmente los artículos no asignados son aquellos que no están incluidos en ninguna de las dos categorías anteriores, cuando el conjunto de artículos no asignados en un nodo es vacío, el nodo contiene una solución y ya no es posible ramificarlo más. En el mismo árbol de la figura 5.6, el nodo 1 representa la clase de todas las soluciones factibles, el nodo 2 representa todas las soluciones factibles que no incluyen al artículo  $j$ , mientras que el nodo 3 representa a las que si incluyen al artículo  $j$ , el nodo 7 representa a todas las que incluyen a  $j$  y  $r$  pero no a  $m$ . Los conjuntos  $I_n, E_n$  para todos los nodos de la figura 5.6 son:

$I_1 = \emptyset$	$E_1 = \emptyset$
$I_2 = \emptyset$	$E_2 = (j)$
$I_3 = (j)$	$E_3 = \emptyset$
$I_4 = (j)$	$E_4 = (m)$
$I_5 = (j,m)$	$E_5 = (m)$
$I_6 = (j)$	$E_6 = (m,r)$
$I_7 = (j,r)$	$E_7 = (m)$

El cálculo de cotas superiores se basa en dos observaciones. La primera es que la relajación de la restricción de indivisibilidad en uno o más de los artículos conlleva a un nuevo problema de optimización cuyo valor óptimo de la función objetivo no puede ser menor que el valor óptimo de la función objetivo para el problema original. La segunda observación es que para un problema de cargo en que todos los artículos son perfectamente divisibles, esto es:

$$\text{Maximizar } \sum_{i=1}^m p_i y_i \quad (5.7)$$

Sujeto a

$$\sum_{i=1}^m y_i w_i \leq W \quad (5.8)$$

$$0 \leq y_i \leq 1 \quad (5.9)$$

Se obtiene una solución óptima arreglando los artículos en orden decreciente de magnitud de la razón  $p_i/w_i$ , y entonces, se comienza con el primer artículo de la lista, secuencialmente se va cargando cada artículo, hasta que se alcanza un peso límite  $W$ .

Aplicamos ahora éstos dos hechos al cálculo de cotas superiores. Supongamos que estamos en el nodo  $n$ , con los conjuntos  $I_n$  y  $E_n$  como se definieron anteriormente, antes de calcular una cota superior en el nodo  $n$ , probamos la factibilidad de la clase de soluciones contenidas en el nodo verificando si

$$\sum_{i \in I_n} w_i \leq W \quad (5.10)$$

Los nodos que no satisfagan (5.10) no se incluyen en los cálculos. Para calcular la cota superior en el nodo  $n$ , decimos que es suficiente con resolver el siguiente problema:

$$\text{Maximizar } \sum_{i=1}^n p_i x_i$$

Sujeto a

$$\sum_{i=1}^n x_i w_i \leq W$$

$$\begin{aligned} x_i &= 1 & \text{si } i \in I_n \\ x_i &= 0 & \text{si } i \in E_n \\ 0 &\leq x_i \leq 1 & \text{si } i \in (I_n \cup E_n) \end{aligned} \quad (5.11)$$

La solución de éste sistema es una cota superior en el nodo  $n$  aplicando la primera observación, ya que en (5.11) hemos relajado la restricción para los artículos no asignados en el nodo  $n$ , la segunda observación da una solución directa a éste sistema ya que por una simple transformación se puede poner en la forma (5.7), (5.8), (5.9). La solución está dada por:

a. Todos los artículos en  $(I_n \cup E_n)'$  se ordenan decrecientemente  $p_i/w_i$  y se comienza con el primer artículo en la lista y se continúa hasta que el peso total de los artículos sea igual a  $W - \sum_{i \in I_n} w_i$ .

b. La cota superior en el nodo  $n$ ,  $B(n)$  es igual al valor total cargado en a. más  $\sum_{i \in I_n} p_i$ .

Para desarrollar la operación de ramificación, se deben tomar dos decisiones. Primero se debe seleccionar el nodo terminal en el que se hace la siguiente ramificación, y segunda, se debe seleccionar el artículo que se deberá sumar a la lista de artículos incluidos y excluidos en los dos nodos resultantes, llamamos a éste al artículo "pivote". La selección del nodo a ramificar está dada por el método de ramificación y acotamiento, de asignar el nodo terminal al que tenga la mayor cota  $B(n)$ . Por otro lado la selección del elemento pivote es arbitraria, se desea que la selección sea de tal forma que el algoritmo alcance una solución óptima más rápidamente, resolviendo (5.7), (5.8), (5.9) se tiene una buena selección heurística. Suponiendo que los artículos originales están arreglados en orden decreciente de la razón  $p_i/w_i$ , selecciona-

mos el elemento pivote como la variable no asignada con el mayor  $p_i/w_i$ . En caso de haber empate se selecciona el primer artículo que aparece en la secuencia.

#### ALGORITMO 5.2: Kolesar

PROPOSITO: Resolver el problema tipo mochila 0-1 usando ramificación y acotamiento.

#### DESCRIPCION

##### Paso 1

1.1 Pruebe la factibilidad no trivial del problema verificando al menos uno de los índices  $i = 1, 2, \dots, n$

$$w_i \leq W$$

si el problema es factible no trivial, siga a 1.2 si no, pare.

1.2 Si  $\sum w_i \leq W$ ; entonces se pueden vaciar todos los artículos y el problema es trivial, si no vaya a 1.3

1.3 Ordene los artículos en orden decreciente de  $p_i/w_i$ , en lo que sigue se supondrá que los artículos están ordenados, vaya a 1.4

1.4 Para el nodo 1 haga  $B(1) = 0$ ,  $I_1 = \emptyset$ ,  $E_1 = \emptyset$  y vaya al paso 2

##### Paso 2.

Selección del nodo para ramificar

2.1 Encuentre el nodo terminal con el mayor valor de  $B(n)$ . Este es el nodo desde el cual se va a ramificar.

2.2 Pruebe si el nodo actual  $k$ ,  $(I_k \cup E_k)' = \emptyset$ , si es cierto, hay una solución óptima dada por los índices contenidos en  $I_k$ . Si no seleccione el nuevo artículo pivote  $i^*$  por  $i^* = i$  tal que  $p_i/w_i$  es máximo para  $i \in (I_k \cup E_k)'$ , y vaya al paso 3.

##### Paso 3.

Cálculo de las cotas superiores.

3.1 Si  $n = n + 1$ ,  $I_n = I_k$ ,  $E_n = E_k \cup \{i^*\}$  vaya a 3.3

3.2 Si  $n = n + 1$ ,  $I_n = I_k \cup \{i^*\}$ ,  $E_n = E_k$  vaya a 3.3

3.3 Pruebe la factibilidad de las soluciones contenidas en el nodo  $n$  verificando si  $\sum_{i \in I_n} w_i \leq W$ , si la prueba falla haga  $B(n) = -999$ , de otra forma proceda a calcular la cota superior metiendo todos los artículos en el conjunto  $I_n$  y procediendo entonces en forma ordenada  $i = 1, 2, \dots, n$ , cargando tantos artículos como sea posible del conjunto  $(I_n \cup E_n)'$  hasta que el peso total cargado sea igual a  $W$ . El valor total es  $B(n)$ . Pruebe si el nodo con índice  $n$  es par. Si lo es proceda con 3.2 si no vaya al Paso 2.

### EJEMPLO

Considere el siguiente problema tipo mochila con siete artículos cuyos pesos y valores se dan a continuación:

ARTICULO No.	PESO	VALOR
1	40	40
2	50	60
3	30	10
4	10	10
5	10	3
6	40	20
7	30	60

El peso total permitido es  $W=100$ . Se desea maximizar el valor de los artículos que se van a llevar en la mochila.

Una prueba preliminar revela que el problema posee una solución factible y no es trivial,  $\sum w_i > 100$ . Calculamos las razones  $p_i/w_i$  y reordenamos los artículos, como se muestra a continuación con un nuevo índice:

NUEVO INDICE	ARTICULO No.	PESO	VALOR	RAZON
1	7	30	60	2
2	2	50	60	6/5
3	1	40	40	1
4	4	10	10	1
5	6	40	20	1/2
6	3	30	10	1/3
7	5	10	3	3/10

El primer nodo es el que incluye todas las soluciones posibles, y la primer ramificación usa el índice 1 como primer pivote, y en el nodo 2 se excluye éste índice de la solución, y la cota superior es igual a :

$$B(2) = p_2 + p_3 + p_4 = 110$$

Mientras en el nodo 3 donde éste índice se incluye tenemos:

$$B(3) = p_1 + p_2 + p_3 = 140$$

Como  $B(3) > B(2)$  se ramifica desde aquí, entonces el índice 2 es el pivote y se tienen los nodos 4 y 5 donde se excluye y se incluye el artículo con índice 2 respectivamente. Calculamos las cotas para éstos nodos, para el nodo 4 se tiene

$$B(4) = p_1 + p_3 + p_4 + p_5 = 120$$

Para el nodo 5

$$B(5) = p_1 + p_2 + p_3 = 140$$

La solución óptima se obtiene aplicando en forma repetida el algoritmo, como se ilustra en la figura 5.8. El óptimo se alcanza en el nodo 15. El valor total es 133 llevando los artículos 7, 2, 4 y 5.

### ALGORITMO 5.3: MARTELLO Y TOTH

**PROPOSITO:** Resolver el problema tipo mochila 0-1, usando ramificación y acotamiento.

#### DESCRIPCION

**Paso 1.** (Inicia) Ordene los artículos en orden decreciente de  $p_i/w_i$ . Calcule  $p^* = \sum_{j=1}^s p_j$  con  $s =$  índice mas grande que cumple

$$w^* = \sum_{j=1}^s w_j \leq W$$

Si  $w^* = W$ , la solución óptima está dada por  $P = p^*$ ,  $(X_j = 1, j = 1, \dots, s)$   $(X_j = 0, j = s+1, \dots, n)$  Pare.

En otro caso, calcule  $(M_i = \min \{w_k | k \leq n\}, j = 1, \dots, n-1), M_n = \infty$ . Sea  $U = UB_2, p = P = 0, (x_j = 0, j = 1, \dots, n), i = 1, s = n$ . Vaya al Paso 4.

**Paso 2** (Trata de incluir el  $i$ -ésimo artículo en la solución actual). Si  $w_i \leq W$ , vaya al Paso 3. En otro caso, si  $P \geq p + [W/p_i + 1/w_i + 1]$ , vaya al Paso 5; si no haga  $i = i + 1$  y repita el paso 2.

**Paso 3** (Construye una nueva solución actual). Calcule  $p^* = p_i + \sum_{j=z_i}^s p_j$ , con  $s =$  índice mas grande para el cual  $w^* = w_i + \sum_{j=z_i}^s w_j \leq W$  y  $s \leq n$  (si  $w_i + w_{z_i} > W$ , sea  $s = z_i - 1$ ) Existen dos posibilidades:

3.1  $w^* = W$  y  $s < n$ : si  $P \geq p + p^* + [(W - w^*)/p_{s+1} + 1/w_{s+1}]$ , vaya al Paso 6, en otro caso, vaya al Paso 4.

3.2  $w^* = W$  o  $s = n$ : Si  $P \geq p + p^*$  vaya al Paso 6, en otro caso, haga  $P = p + p^*$ ,  $(X_j = x_j, j = 1, \dots, i-1), (X_j = 1, j = i, \dots, s), (X_j = 0, j = s+1, \dots, n)$ ; si  $P = U$ , pare; si no vaya al Paso 6.

**Paso 4** (Salvar la solución actual). Sea  $W = W - w^*, p = p + p^*, (x_j = 1, j = 1, \dots, s)$ . Calcule  $w_i = w^*, p_i = p^*, z_i = s + 1$  ( $w_j = w_{j-1} - w_{j-1} p_j = p_{j-1}, z_j = s + 1$  para  $j = i + 1, \dots, s$ ), ( $w_j = p_j = 0, z_j = j$  para  $j = s + 1, \dots, n$ ); sea  $s = s$ . Existen tres posibilidades:

4.1  $s < n-2$ : sea  $i = s + 2$ . Si  $W < M_{i-1}$ , vaya al paso 5, en otro caso vaya al Paso 2.

4.2  $s = n-2$ : Si  $W \geq w_n$ , sea  $W = W - w_n, p = p + p_n, x_n = 1$ . En cualquier caso haga  $i = n-1$  y vaya al Paso 5.

4.3  $s = n-1$ : Sea  $i = n$  y vaya al Paso 5.

**Paso 5** (Salvar la solución actual). Si  $P < p$ , haga  $P = p, (X_j = x_j, j = 1, \dots, n)$ ; si  $P = u$ , pare. En otro caso ( $P \geq p$  o  $P = U$ ), si  $x_n = 1$ , haga  $W = W + w_n, p = p - p_n, x_n = 0$ ; en cualquier caso vaya al Paso 6.

**Paso 6.** Encuentre el mayor  $k < i$  para el cual  $x_k = 0$ . Si  $R \geq M_k$ , haga  $i = k + 1$  y vaya a 2. En otro caso, haga  $i = k, h = k + 1$  y vaya al paso 7

**Paso 7** (Tratar de sustituir el  $h$ -ésimo artículo por el  $k$ -ésimo). Si  $h > n$  o  $P \geq p + [W/p_h/w_h]$ , vaya al Paso 6. En otro caso haga  $D = w_h - w_k$ ; existen tres posibilidades:

7.1  $D = 0$ : Haga  $h = h + 1$  y repita el Paso 7.

7.2  $D > 0$ : Si  $D > R$  o  $P \geq p + p_h, (X_j = x_j, j = 1, \dots, k), (X_j = 0, j = k + 1, \dots, n, j = h), X_h = 1$ . Si  $P = U$ ; si no, haga  $R = R - D, k = h, h = h + 1$  y repita el Paso 7.

7.3  $D < 0$ : Si  $R - D < M_h$ , haga  $h = h + 1$  y repita el Paso 7. En otro caso, si  $P \geq p + p_h + [(R - D)p_h/w_h]$ , vaya al Paso 6; si no, haga  $W = W - w_h, p = p + p_h, x_h = 1, i = h + 1, w_h = w_h, p_h = p_h, z_h = h + 1, (w_j = p_j = 0, z_j = j$  para  $j = h + 1, \dots, s), s = h$  y vaya al Paso 2.

En el paso 1 se manejó la cota superior  $UB_2$  que es igual al máximo valor entre  $B_1$  y  $B_2$  donde :

$$B_1 = \sum_{j=1}^s p_j + [(W - \sum_{j=1}^s w_j) p_s + 2/w_s + 2]$$

$$B_2 = \sum_{j=1}^s p_j + [p_{s+1} - (w_{s+1} - (W - \sum_{j=1}^s w_j)) p_s / w_s]$$

El vector  $(M_i)$ , que también se definió en el paso 1, nos permite saber cuándo, dado el valor actual de  $W$ , se puede introducir al menos un artículo en la solución, esto es cuando los pasos 2 y 3 se deben desarrollar. El propósito de los vectores  $(p_i), (w_i), (z_i)$  es salvar partes de la solución actual que se pueden reutilizar; supongamos que se construye una solución actual introduciendo artículos del  $i$ -ésimo al  $s$ -ésimo, entonces cuando se trata de introducir artículos del  $i$ -ésimo ( $i < i \leq s$ ), y si nada ha cambiado en la solución actual antes del  $i$ -ésimo artículo, es posible introducir en la nueva solución actual una sucesión de elementos del  $i$ -ésimo al  $s$ -ésimo.

Un movimiento hacia adelante se realiza aquí en los pasos 2, 3, y 4. El paso 2 es sólo preliminar al movimiento efectivo hacia adelante del Paso 3, donde se construyen las nuevas soluciones actuales. En el caso 3.2,  $p_i$  no puede crecer más con el valor presente de  $i$ ; y si vale la pena se salva una nueva solución óptima, pero el vector  $(x_i)$  no se abandona, así que es inútil regresar a los valores  $(x_1, \dots, x_s)$  que se han evitado. En el caso 3.1 si la solución actual encontrada - a través de movimientos subsecuentes hacia adelante - mejora la solución actual, se continua al Paso 4, de otra forma se sigue al paso 6.

En el Paso 6 el movimiento hacia atrás en el  $k$ -ésimo artículo es seguido de un movimiento hacia adelante solo si  $R$  (el valor de  $W$  precedente al movimiento hacia atrás) es grande y permite introducir en la solución al menos uno de los artículos siguientes al  $k$ -ésimo. En otro caso, un procedimiento particular hacia adelante (Paso 7) se utiliza, basado en la siguiente consideración: la solución actual se podría mejorar sólo si el  $k$ -ésimo artículo se reemplaza por un artículo (digamos el  $h$ -ésimo) que tenga una  $p_h$  mayor y una  $w_h$  suficientemente menor que permita introducir al menos otro dos artículos, esto es, el  $h$ -ésimo y el  $(h+r)$ -ésimo con  $w_h$  y  $w_{h+r}$  menor que  $w_k$ .

Así, el Paso 7 considera los artículos siguientes desde el  $k$ -ésimo hasta el último que pueda ser útil. En algunos casos (7.1; 7.2 con  $D > R$  o  $P \geq p + p_h$ ; 7.3 con  $R-D < M_h$ ) el artículo considerado se rechaza, ya que no se puede mejorar la solución o ya que es muy larga. En el caso 7.3 con  $D \leq R$  y  $P < p + p_h$ , si los reemplazos mejoran la solución óptima actual,  $P$  y el vector  $(X_j)$  se abandonan y la búsqueda comienza otra vez desde la nueva situación. En el caso 7.3 con  $R-D \geq M_h$ , la solución actual se deja y se sigue una búsqueda hacia adelante normal.

La cota superior del Paso 3.1 se ha calculado en la forma clásica de Dantzig, pero se puede calcular como la cota superior del problema, esto es, la prueba sería:  $P \geq p + p + \max \{[(W - w) p_s + 2/w_s + 2], [p_{s+1} - (w_{s+1} - (W - w)) p_s / w_s]\}$ . En éste caso se puede probar que la nueva cota es mejor que la de Dantzig, pero la anterior requiere menos operaciones, que se puede compensar por el gran número de movimientos hacia adelante y hacia atrás involucrados, por lo cual no se puede definir una estrategia absoluta, y solo estudios empíricos en los datos particulares pueden sugerir una selección entre ambas posibilidades.

## EJEMPLO

El siguiente ejemplo ilustra el algoritmo anterior, primero se resuelve aplicando directamente el algoritmo y posteriormente usando un programa de computadora basado en éste algoritmo, se usó una IBM 55 SX resolviéndose en un tiempo de

Se desea resolver el siguiente problema tipo mochila:

$$\text{MAX } Z = 70X_1 + 20X_2 + 39X_3 + 37X_4 + 7X_5 + 5X_6 + 10X_7$$

SUJETO A

$$31X_1 + 10X_2 + 20X_3 + 19X_4 + 4X_5 + 3X_6 + 6X_7 \leq 50$$

$$X_j \geq 0 \text{ enteras } j=1,\dots,7$$

Aplicando el algoritmo de Martello y Toth

Sean  $n=7$ ,  $W=50$ , donde

$$(p_i) = (70, 20, 39, 37, 7, 5, 10) \text{ y } (w_i) = (31, 10, 20, 19, 4, 3, 6)$$

Paso 1

INDICE	ARTICULO Nº	PRECIOS	PESO	RAZON
1	1	70	31	2.25
2	2	20	10	2
3	3	39	20	1.95
4	4	37	19	1.94
5	5	7	4	1.75
6	6	5	3	1.67
7	7	10	6	1.67

Calculamos  $p^* = \sum_{j=1}^s p_j$  donde  $s$  es el mayor índice para el cual

$$w^* = \sum_{j=1}^s w_j \leq W$$

$$s=2; p^* = 70 + 20 = 90, w^* = 41 \leq 50 = W$$

$$(M_j) = \min\{w_k, j - n_j = 1, \dots, n-1\} = (3, 3, 3, 3, 3, 6, \infty);$$

$$U = UB_2 = \max\{B_1, B_2\} = 90 + \max\{[9 \times 37/19], [39 - 11 \times 20/10]\} = 107.$$

$$P = p = 0, (x_j) = (0, 0, 0, 0, 0, 0, 0), i = 1, s = 7$$

Paso 4.

$$W = W - w^* = 50 - 41 = 9, p = p + p^* = 0 + 90 = 90, x_1 = x_2 = 1;$$

$$w_i = 9, p_i = 90, z_i = 3 \quad (w_j) = (41, 10, 0, 0, 0, 0, 0)$$

$(p_j) = (90, 20, 0, 0, 0, 0, 0)$ ,  $(z_j) = (3, 3, 3, 4, 5, 6, 7)$ ,  $s=2$   $n-2=7-2=5$  de 4.1  $i=s+2=4$ . Si  $9 > M_3$  ir al paso 5 ¿ $50 < 90$ ? no ir a 2.

$$\text{Paso 2 } \hat{w}_4 \leq W? \quad 19 > 9 \text{ entonces } w_4 > 9, \text{ y } P < 90 + [9(7/4)] = 105: i = 4 + 1 = 5$$

Paso 2 ¿ $w_5 < 9$ ?  $4 < 9$  vaya al paso 3

Paso 3. Calcular  $p^* = p_i + \sum p_j$  con  $s=6$ ,  $p^* = 0 + 12 = 12$ ,  $w^* = 0 + 7 = 7 < W$  implica 3.1  $P < 90 + 12 + [2(10/2)] = 105$ .

Paso 4.  $W = 9 - 7 = 2$ ,  $p = 90 + 12 = 102$ ,  $x_5 = x_6 = 1$ ;  $w_5 = w^* = 7$ ,  $p_5 = 12$ ,  $z_5 = 7$ ,  $w_6 = 3$ ,  $p_6 = 5$ ,  $z_6 = 7$ ,  $s = 6$ ,  $s = n - 1$  de 4.3  $i = 7$

Paso 5. Si  $P < p$  entonces  $P = p$ ,  $90 < 102$  entonces  $P = 102$ ,  $(X_j) = (1, 1, 0, 0, 1, 1, 0)$ ;  $P < U = 107$

Paso 6.  $k < i$   $x_k = 1$   $k = 6$ ,  $R = W = 2$ ,  $W = 2 + w_6 = 2 + 3 = 5$ ,  $p = 102 - p_6 = 102 - 5 = 97$ ,  $x_6 = 0$ .  $R < M_6$  entonces  $i = k$   $h = k + 1$  por lo cual,  $i = 6$ ,  $h = 7$ .

Paso 7.  $ih > n$ ? no,  $ih \geq p + [W ph/wh]$ ?  $P = 97 + [5(10/6)] = 105$  por lo tanto  $D = w_h - w_k = 9 - 6 = 3 > 0$  ir a paso 6.2

Paso 7.2  $iD > R$ ?  $3 > 2$  si,  $P \geq p + p_h$ ,  $102 < 107$  pero  $D > R$  entonces  $h = h + 1$  por lo tanto  $h = 8$ .

Paso 7.  $h > n$

Paso 6  $k = 5$ ,  $R = 5$ ,  $W = 5 + 4 = 9$ ,  $p = 97 - 7 = 90$ ,  $x_5 = 0$ .  $R = M_5$ :  $i = 6$

Paso 2.  $w_6 < 9$

Paso 3  $s = 7$ ,  $p^* = 5 + 10 = 15$ ,  $w^* = 3 + 6 = 9 = W$  de 3.2  $P < 90 + 15$ :  $P = 105$ ,  $(X_j) = (1, 1, 0, 0, 0, 1, 1)$ ;  $P < U$ .

Paso 6.  $k = 2$ ,  $R = 9$ ,  $W = 9 + 10 = 19$ ,  $p = 90 - 20 = 70$ ,  $x_2 = 0$ .  $R > M_2$ :  $i = 3$

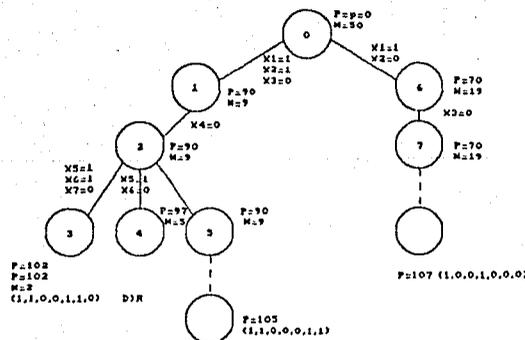
Paso 2.  $w_3 > 19$ ;  $P < 70 + [19(37/19)] = 107$ :  $i = 4$

Paso 2.  $w_4 = 19$

Paso 3.  $s = 4$ ,  $p^* = 0 + 37 = 37$ ,  $w^* = 0 + 19 = 19 = W$  de 3.2  $P < 70 + 37$ :  $P = 107$ .

La figura 5.8 muestra el árbol de decisión para éste ejemplo:

FIGURA 5.8



## CONCLUSIONES

Como ya hemos mencionado antes, tanto el problema del agente viajero, como el problema de la mochila son problemas de optimización combinatoria para los cuales no existen algoritmos eficientes, esto es, con tiempo de computadora acotado por un polinomio en  $n$ , donde  $n$  es el número de nodos. Los métodos de ramificación y acotamiento presentados requieren una búsqueda exhaustiva en el espacio de soluciones donde el tamaño es exponencial en el número de nodos, entonces en el peor de los casos el tiempo de computadora es exponencial en el número de nodos.

Para el caso del problema de la mochila se presentó un algoritmo reciente elaborado por Martello y Toth que ha mostrado ser eficiente sin embargo se han seguido elaborando algoritmos que han resultado ser más rápidos, tal es el caso del algoritmo de D. Fayard y G. Plateau, como se muestra en la tabla siguiente, donde se comparan los tiempos en milisegundos del algoritmo de Martello y Toth MT 78 y el de Fayard y Plateau FPK 79 para problemas con 50, 100, hasta 7500 variables, se dan los tiempos promedio y entre paréntesis los tiempos máximos.

n	FPK 79	MT 78
50	6.6 (13.2)	10.3 (20.2)
100	13.2 (31.3)	22.0 (37.2)
500	50.0 (105.6)	110.7 (151.2)
1000	83.9 (142.8)	203.8 (268.2)
2000	165.2 (266.0)	416.1 (469.9)
5000	390.9 (532.0)	1109.0 (1370)
7500	533.6 (1042.0)	-----

En el caso del agente viajero se presentó un programa de computadora basado en el algoritmo de Little, aunque a últimas fechas se siguen perfeccionando algoritmos para el caso del agente viajero, éstos se refieren a casos de múltiples agentes viajeros y otras variaciones del problema. Aquí sólo consideramos el caso del agente viajero clásico.

## REFERENCIAS AL CAPITULO V.

1. CHRISTOFIDES, N. "Combinatorial Optimization", Ed. John Wiley, 1979.
2. GONDRAN, M. Y MINOUX, M. "Graphs and Algorithms", Ed. John Wiley, 1979.
3. JIANYU, Y. "El Problema del Viajero y sus Extensiones", Tesis de Maestría DEPEI-UNAM, 1985.
4. KOLESAR, P.J. "A Branch and Bound Algorithm for the Knapsack Problem", Management Science 13, p.723-735, 1967.
5. LITTLE, J., MURTY, K., SWEENEY, D. Y KAREL, C. "An Algorithm for the Traveling Salesman Problem", Opns. Res. 34, p.698-717, 1963.
6. MARTELLO, . Y TOTH, P. "Algorithm for the Solution of 0-1 Single Knapsack Problem", Computing 21, p.81-86, 1978.
7. SYSLO, M. "Discrete Optimization Algorithms" Ed.

## CONCLUSIONES

Este trabajo ha estudiado el método de ramificación y acotamiento, tanto en su propia estructura así como los métodos clásicos que se han desarrollado, las aplicaciones que se presentan, pretenden mostrar la amplia aplicabilidad del método. En el caso del agente viajero se proporciona un programa que resuelve el problema basándose en el algoritmo de Little et al.

El método de ramificación y acotamiento como se ha podido ver tiene vigencia, y la tendrá mientras se requieran resolver problemas de investigación de operaciones que conlleven una gran cantidad de cálculos y de almacenamiento de información. A pesar del gran avance que existe en cuanto a velocidad y capacidad de almacenamiento de las modernas computadoras, se siguen presentando este tipo de problemas, por lo que se puede decir que los métodos de ramificación y acotamiento todavía tienen un largo camino que recorrer, no sólo mejorando en sus propios aspectos, también el desarrollo de la computación trae consigo desarrollo para la R-A, el caso más reciente es el de la computación paralela que en nuestro país todavía no se aplica; sin embargo dada la magnitud de los cambios y avances tecnológicos así como la velocidad con que éstos se están desarrollando, no dudemos que en menos de cinco años ya estemos trabajando con este tipo de sistemas.

La aplicabilidad del método de ramificación y acotamiento es tan amplia que es imposible tratar aquí toda la gama de problemas que resuelve, tal es el caso de problemas de secuenciación, de rutas de vehículos, de asignación, etc. Con esta inquietud en mente se tratan de presentar los casos más representativos aquí y se lista en un anexo una cantidad considerable de artículos que se reunieron y que se ponen a disposición de los interesados en el tema.

## APENDICE A

### PRINCIPIO DE RAMIFICACION Y ACOTAMIENTO : MITTEN

Aunque el procedimiento de ramificación y acotamiento se ha aplicado a varias situaciones particulares, el desarrollo de un principio general se produjo como un sucesor de tales aplicaciones. Bertier y Roy (1965) fueron los primeros en presentar una teoría general para ramificación y acotamiento. Balas (1968) reescribió esta teoría en una forma más simple. Más tarde Mitten (1970) generalizó y extendió un poco más el trabajo de Balas. Esta presentación se basa en las ideas de Mitten y Balas.

Un problema típico de ramificación y acotamiento se describe a continuación. Sea  $S$  un conjunto arbitrario cuyos elementos se denominan soluciones factibles, y sea  $f: S \rightarrow \mathbb{R}$  una función real tal que

$$\sup_{x \in S} f(x) = f^*$$

El objetivo en el problema de optimización discreta es encontrar el conjunto de soluciones factibles óptimas denotado por  $S^* = \{x \in S \mid f(x) = f^*\}$ . En situaciones anómalas puede suceder que  $S^* = \emptyset$ .

El problema original frecuentemente se anida en un problema mayor que involucre suposiciones menos restrictivas (por ejemplo, una restricción entera en un problema de programación puede relajarse para facilitar el cálculo de las cotas). Para éstos propósitos se introduce un superconjunto no vacío  $T$  (donde  $S \subset T$ ) junto con una función de extensión  $g: T \rightarrow \mathbb{R}$  de la función  $f$  con los requerimientos siguientes

$$(a) \quad g(x) = f(x) \text{ cuando } x \in S$$

$$(b) \quad \text{Existe una } x \in T \text{ tal que } g(x) = f^*$$

Una buena parte del arte de aplicar los métodos de R-A consiste en escoger  $T$  y  $g$  que produzcan procedimientos computacionalmente eficientes.

Los métodos de R-A tratan con subconjuntos y colecciones de subconjuntos de  $T$ . Para introducir estos elementos, sea  $\tau$  el conjunto de todos los subconjuntos de  $T$  o el conjunto potencia de  $T$ , y sea  $V$  el conjunto de todas las colecciones de subconjuntos de  $T$ , esto es,  $\tau = 2^T$  y  $V = 2^\tau$ . Entonces  $T_i \in \tau$  es un subconjunto de  $T$  y cualquier  $v \in V$  es una colección de subconjuntos de  $T$ . Para abreviar a veces se referirá a una colección de subconjuntos  $v$  simplemente como una "colección".

En éste trabajo  $\{x\}$  es el conjunto que contiene un único elemento  $x$  y la unión de todos los subconjuntos en cualquier colección  $v$  se denotará como  $U(v)$ , es decir  $U(v) = \cup T_i$ , donde la unión es sobre todos los subconjuntos  $T_i \in v$ .

Un elemento importante en los desarrollos subsecuentes es el conjunto  $V^* = \{v \in V \mid U(v) = S^*\}$  que consiste de todas las colecciones de subconjuntos cuyos elementos incluyen todas las soluciones factibles óptimas. La meta del procedimiento de Ramificación y Acotamiento es identificar explícitamente uno de los elementos de  $V^*$ . Si  $S^* = \emptyset$ , claramente  $V^* = \emptyset$ .

Para eliminar soluciones no factibles, es necesario que se identifiquen. Este básicamente un requerimiento impuesto en el procedimiento computacional empleado y se introduce explícitamente a través de las siguientes suposiciones.

(I) El procedimiento computacional especifica una colección  $v_0$  con las siguientes propiedades:

- (a) Los elementos de  $v_0$  contienen solo soluciones no factibles es decir  $U(v_0) \subset T-S$ .
- (b) Todos los subconjuntos no factibles con un sólo elemento están incluidos en  $v_0$ , es decir, si  $x \in T-S$ , entonces  $\{x\} \in v_0$ .

Se supone que  $v_0$  está bien definido, esto es que para cualquier subconjunto  $T_i \subset T$  se pueda determinar que  $T_i \in v_0$  o no. Nótese que (b) es una regla mínima ya que dada una solución  $x \in T$ , es posible determinar cuando es factible o no. Si existen métodos disponibles para determinar si los subconjuntos de mas de un elemento contienen todas las soluciones no factibles, entonces  $v_0$  contendrá mas subconjuntos que el mínimo prescrito en (b) y en consecuencia se podrá mejorar la eficacia del procedimiento de R-A.

### RAMIFICACION

La operación de ramificación divide los elementos de una colección dada en subconjuntos. Las propiedades necesarias de ésta operación se describen a continuación, y el analista es responsable de crear reglas específicas para una situación particular.

(II) La regla de ramificación es una función  $B: V \rightarrow V$  con las siguientes propiedades:

- (a)  $U[B(v)] = U(v)$ ;
- (b)  $T_i' \in B(v)$  sólo si  $T_i' \subset T_i \in v$ ;
- (c)  $B(v) = v$  si y solo si  $v$  contiene un subconjunto  $T_i$  consistente de mas de un elemento.

Las condiciones (a) y (b) establecen que la regla de ramificación  $B$  divide los elementos de la colección  $v$  en subconjuntos que colectivamente incluyen los mismos puntos que la colección original  $v$ . La condición (c) asegura que si existe al menos, un subconjunto divisible en  $v$  éste se encuentra dividido en subconjuntos propios.

### ACOTAMIENTO

Los métodos de R-A emplean dos tipos de cotas -una cota inferior sobre  $f^*$  y cotas superiores en el valor de  $g(x)$  sobre subconjuntos de  $T$ .

(III) La regla de la cota superior es una función  $B: \tau \rightarrow R$  con las siguientes propiedades:

- (a)  $g(x) \leq B(T_i)$  para toda  $x \in T_i \subset T$ ;
- (b)  $B(T_i') \leq B(T_i)$  si  $T_i' \subset T_i \subset T$ ;
- (c)  $B(\{x\}) = g(x)$ .

La condición (a) establece que para cualquier punto en un subconjunto  $T_i$  existe una cota superior, la condición (b) asegura que eliminar puntos a los subconjuntos  $T_i$  no produce cotas superiores mas altas, mientras que la condición (c) asegura que la cota inferior de subconjuntos con un sólo elemento no se pierden innecesariamente. Por conveniencia notacional, se supone también que  $B(\emptyset) = -$  (o un valor finito apropiadamente pequeño).

(IV) La regla de la cota inferior es una función  $b: V \rightarrow R$  con las siguientes propiedades para cualquier colección  $v$ :

- (a)  $b(v) \leq f^*$ ;
- (b)  $b[B(v)] \geq b(v)$ ;
- (c)  $b(v) \geq f(x)$  si  $x \in S$  y  $\{x\} \in v$ ;
- (d)  $b(v') = b(v)$  si  $v' \subset v$ , y, para todo  $T_i \in v-v'$ , entonces  $T_i \in v_0$  ó  $B(T_i) < b(v)$ .

La condición (a) asegura que la cota inferior siempre es menor que el valor óptimo, la condición (b) establece que dividir los conjuntos en subconjuntos no reduce el valor de la cota inferior  $b(v)$ . La condición (c) asegura que la cota inferior sobre conjuntos no se pierde innecesariamente. La condición (d) establece que un subconjunto  $T_i$ , que es sabido que no es factible (es decir  $T_i \notin v$ ) o dominado [i.e.,  $B(T_i) < b(v)$ ], no puede afectar el valor de la cota inferior  $b(v)$ .

Para cualquier colección  $v$ , el símbolo  $v^-$  será usado para designar la subcolección que contiene aquellos elementos de  $v$  que están dominados o se sabe que no son factibles -esto es,  $v^- = (v \cap v_0) \cup \{T_i | T_i \in v \text{ y } B(T_i) < (v)\}$ . Así, ningún elemento de  $v^-$  puede contener una solución factible óptima.

#### RAMIFICACION Y ACOTAMIENTO

La operación recursiva de Ramificación y Acotamiento consiste en formar nuevas colecciones de subconjuntos de los cuales se han excluido aquellos elementos que se sabe no contienen una solución factible óptima. Específicamente

(V) La operación recursiva de R-A es una función  $B:V \rightarrow V$  con la propiedad de que, si  $B(v^-) = v^-$ , entonces  $B(v) = v - v^-$ . Entonces, la función recursiva de ramificación y acotamiento usa la regla de ramificación para dividir los elementos de la colección  $v$  en subconjuntos. Aplicando las funciones de acotamiento  $B$  y  $b$  a la nueva colección  $v$  se define la colección  $v^-$  de subconjuntos de  $v$  que son dominados o no factibles. Quitando los elementos de  $v^-$  de  $v$  se obtiene la nueva colección  $B(v)$ .

## APENDICE B

En éste apéndice se desarrolla un ejemplo sencillo del Algoritmo de Land y Doig: Se aplica el algoritmo a un problema de programación lineal en dos variables sin restricción de variable discreta.

Sea el problema de optimización discreta:

$$(B.1) \text{ MAXIMIZAR } Z = CX + DY$$

SUJETO A

$$(B.2) AX + DY \leq B$$

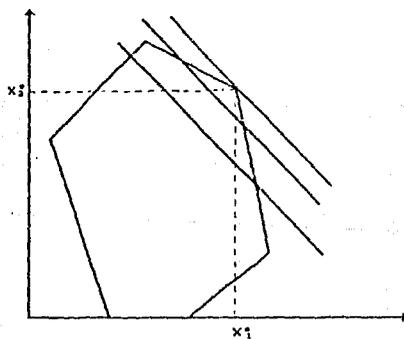
$$(B.3) X \text{ vector columna con componentes enteros no negativos}$$

$$(B.4) Y \geq 0$$

Donde  $Z$  es un escalar,  $B$  es un vector columna de  $m$  renglones,  $C^1$ ,  $X$  vectores columna de  $n_1$  renglones,  $D^1$ ,  $Y$  vectores columna de  $(n-n_1)$  renglones,  $A$  matriz de orden  $m$  por  $n_1$ , y  $D$  matriz de orden  $m$  por  $(n-n_1)$ . Una solución factible al problema es aquella que satisface

En la figura siguiente se puede ver que la función (representada por la familia de rectas paralelas) alcanza el máximo en  $(x_1^0, x_2^0)$

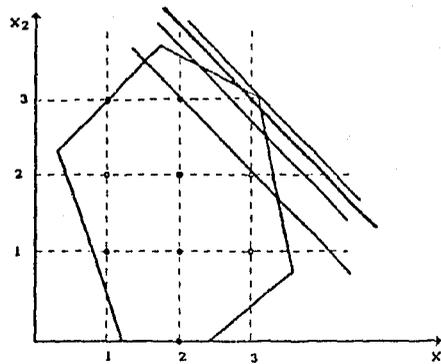
FIGURA B.1



las restricciones de variable discreta limitan el conjunto de soluciones factibles a los puntos que están en el conjunto original para los cuales ambas coordenadas son enteras; en la figura 2.2 se muestra el conjunto completo de soluciones factibles para el problema de programación discreta

el cual consta de 10 puntos, es fácil ver que en éste caso la solución máxima es  $x_1 = 3$ ,  $x_2 = 3$  (indicada por el punto E). El procedimiento para llegar a ella se puede describir como "mover hacia abajo la función lineal hasta encontrar un punto entero".

FIGURA B.2



#### DESCRIPCION DEL METODO

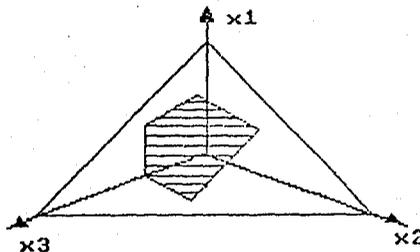
Este método usa sistemáticamente paralelas en el hiperplano en términos de reducir la función objetivo hasta encontrar un punto en el conjunto de soluciones factibles con coordenadas enteras, en las dimensiones que se especifican. Esto es en principio, ya que en la práctica no se tiene la facultad de "ver" un hiperplano en un espacio de  $n$  dimensiones, en términos de determinar si contiene un punto cuyas  $n$  coordenadas son enteras. Los métodos numéricos sólo pueden examinar un punto a la vez, pero al pasar el hiperplano de una región a otra no se debe omitir ningún punto entero.

Si la cota superior de la función en cualquier etapa es  $Z^K$  entonces se ha demostrado que hay una solución no discreta con un valor más alto del maximando que  $Z^K$ .

Considere el conjunto convexo de soluciones de un problema de programación lineal y  $Z^K$  cualquier valor factible de la función objetivo, que está únicamente asociado con una posición definida del hiperplano, que en general corta al conjunto convexo de  $n$  dimensiones y en el caso especial del valor óptimo toca al conjunto convexo de  $n-1$  dimensiones.

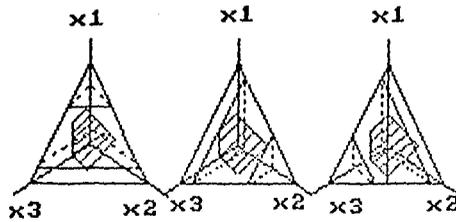
Por ejemplo la figura B.3 representa una intersección con un conjunto tridimensional.

FIGURA B.3



Los puntos  $x_1$ ,  $x_2$  y  $x_3$  son las intersecciones del hiperplano con los tres ejes respectivamente y el área sombreada representa el conjunto convexo que está dentro de las restricciones del problema de programación lineal ordinario. En tal caso el conjunto de  $n-1$  dimensiones tendría un mínimo y un máximo para cada variable como se muestra en la figura B.4

FIGURA B.4



Si  $Z^k = Z^0$  esto es, el hiperplano es el asociado con la solución óptima del problema de programación lineal ordinario, este conjunto convexo consistiría normalmente de un único punto (a menos que tuviera soluciones óptimas múltiples). Es también cierto que, para cualquier valor del maximando (posición del hiperplano) hay un valor mínimo y uno máximo (que puede coincidir) para cada variable, consistente con las restricciones del problema de programación lineal ordinario.

Conociendo un valor particular de  $Z$ , los valores mínimo y máximo para cada variable se conocen así como un posible valor entero de cada variable para la posición del hiperplano. Si no hay al menos un posible valor entero para cada variable entonces se puede decir inmediatamente que no hay solución al problema en el valor del maximando. Desafortunadamente lo inverso no es cierto, el hecho de que cada variable tome un valor entero en algún punto o puntos en el conjunto no es condición suficiente para que haya una intersección de esas coordenadas enteras en el conjunto y por lo tanto una solución factible.

Ya que hay un valor mínimo y un valor máximo para cualquier variable  $X_k$  en un valor particular de  $Z$ , de aquí se sigue que, se pueden definir dos funciones  $\min X_k$  y  $\max X_k$  entre  $X_k$  y el hiperplano (valor de  $Z$ ).

La condición entre estas dos funciones y el problema fundamental puede demostrarse a través del siguiente sistema de desigualdades.

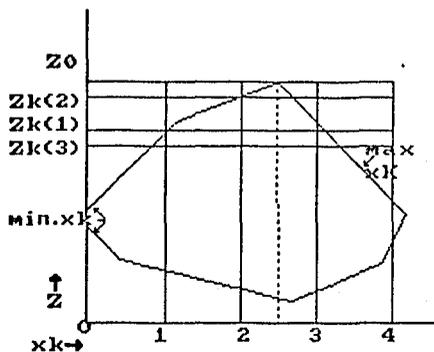
$$\begin{aligned} (B.1') \quad & Cx + dy - z = 0 \\ (B.2') \quad & Ax + Dy \leq b \\ (B.3') \quad & x \geq 0 \\ (B.4') \quad & y \geq 0 \\ (B.5') \quad & z \geq 0 \end{aligned}$$

Este sistema define un conjunto poliédrico convexo en un espacio de  $n+1$  dimensiones y como el plano proyectado de un conjunto convexo es convexo, la proyección de este conjunto en el plano  $(x_k, z)$  producirá un polígono convexo cuya cota superior (inferior) es una función cóncava (convexa) de la abscisa. Tal polígono se muestra en la figura B.5.

El valor  $Z^0$  que toma  $z$  en el punto más alto del polígono es el máximo valor que puede tomar sujeto al sistema de desigualdades dado. Esto es equivalente a decir que  $Z^0$  es el valor óptimo de la función  $z$  en el problema de programación lineal con restricciones (B.2), (B.3') y (B.4). Más aún  $X_k$  toma el valor  $X_k^0$  en esta solución. (si  $X_k$  no estuviera en la

base óptima, el pico del polígono podría estar en el eje de  $z$ ). Las cotas del polígono son las funciones  $\min X_k$  y  $\max X_k$ . Estas funciones se pueden calcular resolviendo las dos familias de problemas de programación lineal definidas por (B.1'), (B.2), (B.3'), (B.4) y (B.5) con funciones "minimizar  $X_k$ " y "maximizar  $X_k$ " para todos los posibles valores de  $Z$  usando programación lineal paramétrica. Un polígono de dos dimensiones de este tipo podría determinarse para cada variable  $x$  del problema original discreto.

FIGURA B.5



En la figura B.5,  $X_k^0$  cae entre los valores 2 y 3. Si el  $\min X_k$  y el  $\max X_k$  se trazan reduciendo  $z$  de  $Z^0$  hasta que se encuentra un primer valor entero de  $X_k$  en cada uno. Los dos valores  $Z_k(2)$  y  $Z_k(3)$  se obtienen de esta forma en la figura 2.1. Esto es equivalente a examinar la trayectoria que se traza por dos puntos específicos de la intersección del hiperplano funcional con el conjunto convexo de soluciones factibles, cuando el hiperplano sistemáticamente se mueve hacia atrás de su posición máxima  $Z^0$ . Se puede decir de  $Z_k(2)$  que es una cota superior del maximando ya que al no haber un valor más alto de  $z$  puede  $x_k$  tomar un valor entero.

La solución del problema de programación discreta se desarrollará usando sistemáticamente este argumento. En aras de facilitar la exposición definimos un conjunto de problemas subsidiarios  $P(j)$  como sigue:

$P(j)$ : Maximizar  $z$  sujeto a las restricciones (B.2), (B.3') y (B.4) y las restricciones adicionales en el sentido de que  $j$  de las variables  $x$  son enteros no negativos ( $j = 0, 1, 2, \dots, n_1$ ).

Sea  $S_j$  el conjunto de todas las soluciones factibles a problemas del tipo  $P(j)$  y sea  $S$  el conjunto de soluciones no factibles a cualquier problema del tipo  $P(j)$ . El problema particular en que por ejemplo,  $x_2$  y  $x_4$  se restringen a ser enteros no negativos, se escribirá  $P(2; 2, 4)$  y el conjunto de sus soluciones factibles se escribe  $S_2(2, 4)$ . En esta notación, la solución requerida es el elemento de  $S_{n_1}$  para el cual  $z$  se maximiza. El valor de  $z$  de esta solución está acotado por el máximo de  $z$  sobre el conjunto  $S_{n_1-1}$  el cual está acotado por el máximo de  $z$  sobre  $S_{n_1-2}$  etc. La última cota superior alcanzada de esta manera es  $Z^0$ , el máximo valor de  $z$  sobre  $S_0$ .

#### REGLAS PARA AUMENTAR ARCOS Y NODOS

En el Paso 0, el árbol consiste de un sólo vértice etiquetado  $Z^0$  que es un elemento de  $S_0$ .

La decisión que ahora se toma es: dado que  $x_r$  está en la base de  $Z^0$  se restringe a tomar valores enteros no negativos, esto es, la atención se centra en el problema  $P(1;r)$ . Sea  $x_r = x_r^0$  en  $Z^0$ . Las funciones  $\min x_r$  y  $\max x_r$  se trazan hasta los puntos  $(\{x_r^0\}, z_{im})$  y  $(\{x_r^0\} + 1, z_{st})$  respectivamente, donde:

$\{x_r^0\}$  es el mayor entero menor o igual a  $x_r^0$

$z_{rm}$  es el máximo valor de  $z$  sujeto a  $P(0)$  y la condición  $x_r = \{x_r^0\}$ , y

$z_{rM}$  es el máximo valor de  $z$  sujeto a  $P(0)$  y la condición  $x_r = \{x_r^0\} + 1$ .

Los nodos  $z_{rm}$  y  $z_{rM}$ , si existen, son elementos de  $S_1$ . Se ha visto, así mismo, que ambos pueden evaluarse resolviendo el problema de programación estándar en  $(n-1)$  variables. Si uno de esos problemas dice "maximizar  $z$  sujeto a  $P(0)$  y a  $x_r = \{x_r^0\}$ " y no es factible, entonces no existe  $z_{rm}$  y se aumenta un nodo a  $S$ . Esto implica que  $S_1$  no contiene un elemento para el cual  $x_r = \{x_r^0\}$ , por lo cual sólo los valores de  $x_r$  que satisfacen  $x_r \{x_r^0\} + 1$  necesitan considerarse en el desarrollo del árbol. Si no existen  $z_{rm}$  y  $z_{rM}$ , entonces  $x_r$  no se puede restringir a un valor entero y por lo tanto el problema no posee solución factible. Se adicionan entonces los dos arcos y nodos al Paso 1

Del Paso 2,  $z^1 = \max(z_{rm}, z_{rM})$  al calcular  $z_{rm}$  y  $z_{rM}$ , se ha examinado cada valor de  $z$  entre  $z^0$  y  $z^1$ , y se ha demostrado que  $z^1$  es el máximo valor de  $z$  que es compatible con un valor entero de  $x_r$ . Supóngase que  $x_r = v$  en  $z^1$ , ésta ecuación se aumenta al problema como una nueva restricción que se aplica a cualquier rama del árbol que se traza después de  $z^1$ . La variable  $x_r$  se elimina de los cálculos, reduciéndose de ésta manera, las dimensiones del conjunto de soluciones exploradas a  $(n-1)$ , y cada cota  $b_i$  se reduce por una cantidad  $a_{ir}v$ . Esta nueva restricción es válida con respecto a las soluciones de  $P(1;r)$  sobre el rango de valores de  $z$  para el que el único valor entero permisible de  $x_r$  es  $v$ . La extremidad superior de éste rango es  $z^1$  y, ya que la cota superior del polígono de la figura B.5 es una función cóncava de la abscisa, el extremo inferior debe ser  $z_r(v-1)$  o  $z_r(v+1)$ . El mayor de éstos dos es la segunda mejor solución de  $P(1;r)$ . Uno de los valores vecinos ya se conoce (es el  $\min(z_{rm}, z_{rM})$ ) y una cota superior para el otro se puede encontrar por extrapolación de una función apropiada para  $x_r$ . Si ésta cota superior excede el otro valor vecino, el valor  $z$  correspondiente se podría determinar en forma exacta. Los arcos y nodos del Paso 3 se aumentan al árbol.

Para ilustrar éste argumento, considere el problema de dos dimensiones mostrado en la figura B.2.

Paso 0. La solución en A proporciona el primer vértice del árbol,  $z^0$

Paso 1 Se selecciona la variable  $x_2$  y se determinan los puntos B ( $x_2 = 2, z = z_{2m} = z_2(2)$ ) y C ( $x_2 = 3, z = z_{2M} = z_2(3)$ ).

Paso 2  $z^1 = z_{2M}$  para  $v = 3$

Paso 3  $z^1$  no es la solución final, ya que  $x_2 = 4$  está completamente afuera del conjunto convexo  $S_0$ ,  $z_2(4)$  está en  $S$ . El único posible valor entero de  $x_2$  para todos los valores de  $z$  que satisfacen  $z_2(2) < z \leq z_2(3)$  es  $x_2 = 3$ ; ésta restricción es válida para la recta CD que es un subconjunto unidimensional del conjunto de soluciones factibles de  $P(0)$ .

Paso 4  $x_1$  está ahora restringido a valores enteros. Uno de los nuevos vértices ( $z_{1m}$ ) está en  $S_2$  y el otro en  $S$

Paso 2  $z^2 = z_{1m} > z_{2M}$

Paso 3  $z^2$  es un elemento de  $S_2$ ; entonces el punto E, correspondiente a  $z^2$  es la solución requerida. En éste punto,  $x_1 = 2, x_2 = 3$

Claramente en cualquier etapa de la solución, las únicas variables  $x$  que pueden violar las restricciones discretas son las que están en la base actual. Entonces, restringiendo sucesivamente cada variable en la forma que se describió anteriormente, se encontrará una solución factible de  $P(n_1)$  o se demostrará que tal solución no existe.

En el caso general, la primera cota superior en la función  $z^0$  es la solución óptima a  $P(0)$ . La segunda cota superior es  $z^1 \leq z^0$ ;  $z^1$  es la solución óptima de  $P(1;r)$ . Esta segunda cota superior se puede afinar encontrando la solución óptima a todos los problemas de tipo  $P(1)$  y seleccionando al menos uno de ellos como  $z^1$ . En éste análisis es más económico en términos de esfuerzo computacional trazar las funciones  $\min$  y  $\max$  de una sola variable en cada etapa.

Si  $z^1$  no es una solución para  $P(n_1)$ , se escoge una nueva variable  $x$  entre aquellas que están en la base de  $z^1$ . El Paso 4 consiste en trazar las funciones  $\max$  y  $\min$  de ésta variable a los respectivos valores enteros más cercanos por arriba y

por debajo de su valor en  $z^1$ . Esto aumenta dos nuevos nodos al árbol, y del paso 2,  $z^2$  es el máximo valor de  $z$  tomado sobre ellos y los valores vecinos  $z^1(v-1)$  y  $z^1(v+1)$  de  $z^1$ . El argumento completo se puede repetir en  $z^2$  reemplazando  $z^1$  como la cota superior actual en el valor óptimo de  $z$ . Continuando este proceso, se forma un árbol donde cada uno de sus nodos representa un conjunto conocido de restricciones enteras ( $z^1$ , por ejemplo, representa  $x_r = v$ ). Una ramificación termina si alcanza un nodo de  $S$ . Podemos concluir de aquí que: si todas las ramas terminan en  $S$  el problema no tiene solución factible o si se alcanza un nodo  $z^1$ , todas las variables  $x$  son enteros no negativos. En principio, restricciones apropiadas se aplican a cualquier variable  $x$  que no se ha restringido por un valor entero, y un nodo  $z^1$  en  $S_{nt}$  se alcanza con el mismo valor  $z$  que el nodo etiquetado  $z^1$ . Ya que los nodos etiquetados son cotas superiores en la función  $z^1$  debe ser la solución óptima requerida.

## APENDICE C

### RESULTADOS Y ANALISIS DEL EXPERIMENTO DE GUPTA.

Cada una de las 27 estrategias se probaron con todos los problemas, en un sistema de computadoras CDC 6500, con la excepción del problema 19, todas las estrategias pudieron resolver cada problema en menos de 240 segundos de tiempo de ejecución. El problema 19 resultó ser el más difícil de resolver ya que 2 de las estrategias  $(K, L, N) = (1,2,1)$  y  $(1,2,3)$  no pudieron encontrar soluciones óptimas dentro de un tiempo límite de 500 segundos. Dos estrategias más,  $(3,2,3)$  y  $(3,3,3)$  generaron subproblemas para el problema 19 para los cuales el código OPT no pudo siquiera encontrar soluciones óptimas continuas. Aunque las 23 estrategias restantes sí encontraron soluciones óptimas a éste problema, algunas estrategias tardaron casi 500 segundos en llegar a la solución final.

El hecho de que 4 estrategias fueran incapaces de llegar a la solución final de uno de los 22 problemas de prueba resta poco al comportamiento de estas estrategias o del código OPT ya que fueron exitosas para todos los demás problemas.

#### Criterios de Jerarquización de Estrategias.

El costo significativo para ejecutar un programa entero no lineal es el tiempo de solución de la computadora. Por eso decidimos usar el tiempo de solución como el indicador inicial del comportamiento de las estrategias. Se encontró que nuestros problemas de prueba tenían una amplia variación en tiempo de solución, que iba desde menos de 0.5 segundos a casi 500 segundos. La Tabla 1 presenta los tiempos promedios de solución, los peores y los mejores tiempos de solución. También da la proporción de los peores a los mejores.

Ya que nuestro principal interés es comparar varias estrategias de selección de heurísticas, nodos y variables para ramificación, primero calculamos los tiempos promedios de solución para cada problema de prueba tomados sobre las 9 combinaciones mientras se mantiene el valor de los parámetros fijado en uno a la vez. Los resultados se muestran en las tablas C.1, C.2, C.3. Como se muestra en la Tabla C.1, el promedio total de los tiempos de solución para la opción 2 de la selección de variable ramificar (es decir, seleccionando la variable entera más fraccional) es de 19.766 segundos, lo que es significativamente menor que los promedios totales obtenidos por las otras dos opciones de selección de variable.

**TABLA C1.**

**Tiempos Promedio de Solución para las Opciones de Variable a Ramificar**

Problema	k=1	k=2	k=3
1	1.323	0.891	0.891
2	13.918	10.224	11.972
3	0.760	0.760	0.759
4	0.466	0.467	0.466
5	6.257	6.241	6.251
6	0.564	0.563	0.561
7	1.053	0.755	0.759
8	2.702	2.410	2.400
9	3.265	3.240	3.262
10	0.726	0.730	7.282
11	2.134	2.078	2.081
12	4.867	4.794	4.788
13	13.099	13.880	14.034
14	11.507	11.766	13.316
15	1.168	1.372	1.367
16	0.754	0.557	0.561
17	124.705	92.828	107.971
18	78.826	44.470	124.033
19	271.261	157.641	262.163
20	53.290	70.459	74.403
21	1.945	1.949	1.940
22	6.720	6.768	11.210
Prom.Tot.	27.361	19.766	29.658

Similarmente la tabla C.2 muestra que la opción 2 de selección del nodo es significativamente peor que las otras dos ya que su tiempo promedio total de solución es relativamente muy grande

**TABLA C.2**

**Tiempo Promedio de Solución para las Opciones de Selección de Nodos**

Problema	L=1	L=2	L=3
1	1.036	1.038	1.032
2	14.396	11.157	10.562
3	0.760	0.758	0.761
4	0.469	0.466	0.464
5	6.256	6.242	6.251
6	0.562	0.565	0.561
7	0.852	0.851	0.864
8	2.420	2.688	2.405
9	3.254	3.254	3.260
10	0.676	0.840	0.668
11	1.785	2.721	1.788
12	3.920	6.579	3.950
13	10.171	20.910	9.922
14	10.277	16.110	10.253
15	1.297	1.346	1.277
16	0.611	0.651	0.610
17	83.139	157.788	83.467
18	76.187	105.564	68.912
19	162.101	398.140	178.578
20	62.455	73.898	62.429
21	1.950	1.936	1.949
22	7.968	8.671	8.059
Prom. Tot.	20.563	36.462	20.819

La Tabla C.3 indica que las heurísticas no proporcionan ninguna ventaja importante para reducir el tiempo de cálculo.

TABLA C.3

Tiempos Promedio de Solución para las Opciones de Selección de Heurísticas.

Problema	N = 1	N = 2	N = 3
1	0.981	1.145	0.980
2	10.093	16.414	9.607
3	8.711	0.854	0.713
4	0.441	0.518	0.440
5	5.771	6.176	6.802
6	0.534	0.622	0.530
7	0.737	0.916	0.915
8	2.383	2.731	2.399
9	2.554	4.658	2.555
10	0.690	0.798	0.696
11	1.785	2.266	2.243
12	4.355	5.985	4.110
13	12.394	16.190	12.418
14	12.000	12.550	12.039
15	1.304	1.353	1.252
16	0.624	0.665	0.583
17	103.813	123.919	96.661
18	79.389	85.895	85.379
19	189.179	221.563	242.506
20	57.682	83.749	57.351
21	1.906	2.021	1.908
22	7.959	8.816	7.922
Prom. Tot.	22.604	27.264	25.000

Hemos hecho éste análisis considerando tan sólo los tiempos de solución reales. Una mirada mas minuciosa a los tiempos de solución, indica una buena cantidad de tiempo gastada en los problemas 17, 18, 19 y 20. Por lo cual un simple promedio total de los tiempos de solución podría ser engañoso debido al peso diferencial. Así decidimos hacer otra comparación usando "tiempos de solución normalizados" en lugar de los tiempos reales.

Tiempo de Solución Normalizado (NST) Para dar igual importancia a cada problema de prueba se usa el siguiente procedimiento para normalizar los tiempos de solución:

1. Sea  $t_{ij}$  el tiempo de solución para la estrategia  $i$ -ésima para resolver el problema  $j$ -ésimo, y  $n_j$  que denota el número de estrategias que pudieron alcanzar la solución óptima final para el problema  $j$ -ésimo dentro de un límite de tiempo especificado de 500 segundos ( $i = 1, 2, \dots, 27; j = 1, 2, \dots, 22$ ).

2. Se forma la suma  $S_j = \sum_i t_{ij}$  donde la suma se toma sobre todas las estrategias  $i$  que podrían resolver el problema  $j$ ; así, el tiempo promedio de solución para el problema  $j$ -ésimo es  $A_j = S_j/n_j$  ( $j = 1, 2, \dots, 22$ )

3. Se forman los tiempos de solución normalizados ( $NST_{ij}$ ) para la estrategia  $i$ -ésima que resuelve el problema  $j$ -ésimo como  $NST_{ij} = t_{ij}/A_j$ .

Ya que el tiempo de solución promedio normalizado para cada problema es una unidad, los resultados de los problemas de prueba ahora son comparables, y se pueden hacer comparaciones directas. Una estrategia con tiempos de solución normalizados mas bajos (menos que uno) tendría una puntuación mejor que una estrategia con tiempos de

solución normalizados mas altos (mayor que uno). Para evaluar el comportamiento de las estrategias se usa el siguiente procedimiento

1. Primero se calculan los tiempos normalizados de solución ( $NST_{ij}$ ) para cada estrategia y problema de prueba.
2. Se forma la suma  $ST_i = \sum_j NST_{ij}$ , y el promedio  $AV_i = ST_i/N_i$ , donde  $N_i$  es el número de problemas resueltos por la estrategia  $i$ .
3. Se jerarquizan los tiempos normalizados de solución promedios para cada estrategia, siendo la mejor estrategia aquella con el menor valor de  $AV_i$ .

La jerarquización y tiempos promedio normalizados de solución se muestran en la tabla C.4 Se pueden hacer las siguientes observaciones:

1. Ninguna de las 12 estrategias mas altas utiliza ni ramificación del nodo mas nuevo ni la implantación de la heurística A. De hecho, éstas 12 estrategias consisten de la combinación exclusiva y exhaustiva de las otras estrategias de selección (un diseño (3)(2)(2)).

2. Las mejores cuatro estrategias utilizan la variable entera mas fraccional como la estrategia de selección de variable para ramificación, y las siguientes 8 estrategias usan las otras dos estrategias de selección de variable para ramificar. El patrón de la estrategia de selección de variable entera mas fraccional siendo mejor que las otras dos estrategias continúa cuando una combinación de selección de nodo y selección de heurística está fija, y así se puede concluir que la estrategia de la variable entera mas fraccional es una estrategia relativamente mejor que las otras dos estrategias para la selección de variable a ramificar en estos problemas.

TABLA C.4

JERARQUIA	Jerarquias de estrategias			Promedio NST
	K	L	N	
1	2	3	1	0.779
2	2	1	1	0.790
3	2	3	3	0.816
4	2	1	3	0.839
5	1	3	1	0.867
6	3	3	1	0.869
7	1	1	1	0.877
8	3	1	1	0.897
9	1	3	3	0.909
10	3	3	3	0.917
11	1	1	3	0.932
12	3	1	3	0.989
13	2	3	2	0.994
14	2	2	3	1.009
15	2	1	2	1.026
16	3	3	2	1.031
17	2	2	1	1.037
18	3	2	3	1.093
19	1	1	2	1.099
20	1	2	3	1.108
21	3	2	1	1.109
22	3	1	2	1.113
23	1	3	2	1.133
24	1	2	1	1.141
25	2	2	2	1.193
26	1	2	2	1.257
27	3	2	2	1.373

**Análisis de Variancia.** Para poder hacer comparaciones estadísticas se usó el paquete SPSS para analizar los datos proporcionados por los tiempos normalizados de solución para el propósito del análisis de la variación. Antes de considerar la tabla de análisis de variancia, es interesante examinar los valores promedios completos de los tres parámetros de selección. Las tablas C.5, C.6 y C.7 presentan los promedios completos para la selección de variable, para ramificación, selección de nodo y las estrategias de selección de heurística. Los resultados son consistentes con las observaciones hechas previamente. La estrategia de selección de variable entera mas fraccional funciona mejor que las otras dos estrategias de selección de variable, mismas que se comportan mas o menos igual. La selección del nodo mas reciente resulta ser la peor de las estrategias de selección de nodos. La estrategia que usa estimaciones parece comportarse mejor que la que selecciona el nodo con la menor cota. Ninguna de las heurísticas ofrece una ventaja computacional.

**TABLA C.5**

**Valores Medios de NST para las Estrategias de Selección de Variable a Ramificar**

Primer índice menor	1.03
Variable más fraccional	0.94
Seudo-costos	1.04

**TABLA C.6**

**Valores Medios de NST para las Estrategias de Selección de Nodos.**

Nodo de Menor Cota	0.95
Nodo más reciente	1.15
Estimación	0.92

**TABLA C.7**

**Valores Medios de NST para las Estrategias de Selección Heurísticas**

No Heurísticas	0.93
Heurística A	1.14
Heurística B	0.96

Con éstas observaciones en mente, se puede examinar el análisis de variancia en la Tabla C.9. Los resultados allí mostrados indican que existen diferencias significativas entre los promedios de tiempos de solución para los efectos de las tres principales estrategias de selección. Esto no debe causar sorpresa dada la discusión anterior. La tabla también indica que no hay diferencias significativas debidas a interacciones de dos vías. Esto da evidencia de que los tres parámetros de selección están independientes el uno del otro.

**TABLA No C.8**  
Análisis de Varianza (Tiempo de solución normalizado)

Fuente de Variación	Suma de Cuadrados	D.F.	Cuadrado Medio	F-razón Significancia	
Efectos med.	12.000	6	2.000	30.621	0.001
S.de Var	1.243	2	0.621	9.513	0.001
S. de Nodo	5.783	2	2.892	44.272	0.001
S. de Heurística	4.937	2	2.469	37.796	0.001
Iteración Doble	0.229	12	0.019	0.293	0.991
Variable-Nodo	0.076	4	0.019	0.292	0.853
Var-Heurística	0.010	4	0.002	0.037	0.997
Nodo-Heurística	0.142	4	0.036	0.545	0.703
Iteraciones triples	0.251	8	0.031	0.480	0.870
Var-Nod-Heuris	0.251	8	0.031	0.480	0.870
Explicados	12.480	26	0.480	7.349	0.001
Restantes	36.773	563	0.065		
Total	49.253	589	0.084		

**TABLA C.9**  
Jerarquías de estrategias

JERARQUIA K		L	N	Promedio NLP
1	2	3	1	0.781
2	2	1	1	0.802
3	3	3	1	0.827
4	1	3	1	0.836
5	1	1	1	0.839
6	2	3	3	0.840
7	3	3	3	0.851
8	3	1	1	0.859
9	2	1	3	0.862
10	1	1	3	0.899
11	1	3	3	0.902
12	3	1	3	0.906
13	2	3	2	1.013
14	2	1	2	1.040
15	3	3	2	1.044
16	1	1	2	1.075
17	1	3	2	1.082
18	3	1	2	1.096
19	2	2	3	1.101
20	2	2	1	1.108
21	3	2	3	1.131
22	1	2	3	1.143
23	3	2	1	1.190
24	1	2	1	1.200
25	1	2	2	1.218
26	2	2	2	1.249
27	3	2	2	1.302

Otro criterio de Jerarquización de Estrategias.

En el procedimiento de jerarquización anterior hemos comparado los tiempos normalizados de solución que se han calculado a partir de sus correspondientes tiempos de solución reales. Frecuentemente el tiempo de solución depende de lo siguiente:

1. El sistema de computadoras usado para el estudio
2. El programa de computadora usado para resolver el problema. En éste caso, el programa BBNLMIP involucra lo siguiente:
  - a. El código de R-A para computadora donde todas las operaciones lógicas de R-A se llevan a cabo excepto resolver los problemas continuos no lineales y
  - b. El código no lineal OPT que resuelve todos los problemas continuos intermedios. Ya que el tiempo de computadora depende del sistema de computación del código de R-A y del código no lineal (OPT en nuestro caso), la jerarquización de las estrategias también podría depender de éstos factores. Es por ésto que se usa también otro criterio de jerarquización para hacer comparaciones donde éstos factores no tienen ningún efecto.

Este criterio compara el número de problemas no lineales continuos resueltos en lugar de los tiempos de solución. Los problemas continuos que se resuelven bajo una estrategia particular de R-A de hecho definen el árbol correspondiente de R-A y es por esto que estos problemas quedarían iguales sin importar el sistema de computadora, el código de R-A y el algoritmo no lineal usado para resolver el problema.

Se tabuló el número total de problemas continuos no lineales resueltos por cada estrategia y un procedimiento similar a aquel usado para normalizar los tiempos de solución se usó para normalizar el número de problemas continuos no lineales resueltos para cada problema. Las estrategias se jerarquizaron según su valor promedio del número normalizado de problemas no lineales (NNLP). Los resultados se ven en la tabla 10.

Nótese que las doce estrategias mejores son las mismas. Ninguna de ellas hace uso ni de la ramificación del nodo más reciente ni de la implantación de la heurística A. Las mejores dos estrategias siguen siendo las mismas. Las jerarquizaciones de las otras estrategias no cambian significativamente. Los promedios totales para los parámetros de selección se encuentran en las tablas C.10, C.11 y C.12. Nótese que los resultados son similares a los de las tablas C.5, C.6 y C.7

TABLA C.10

Valores Medios de Estrategias de Selección	NNLP para las Variable a Ramificar
Primer índice menor	1.02
Variable más fraccional	0.98
Seudo-costos	1.02

TABLA C.11

Valores Medios de Estrategias de Selección	NNLP para las Nodos.
Nodo de Menor Cota	0.93
Nodo más reciente	1.18
Estimación	0.91

TABLA C.12	
Valores Estrategias	Medios de NLP para las Selección Heurísticas
No Heurísticas	0.94
Heurística A	1.12
Heurística B	0.96