UMA TÉCNICA DE PÓS-OTIMIZAÇÃO EM PROGRAMAÇÃO 0-1 UTILIZANDO ENUMERAÇÃO IMPLÍCITA.

Manuel Bernardino Lino Salvador

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.)

Aprovada por:

Félix Eduardo Vaca Obando
(Presidente)

Nelson Maculan Filho

Claudio Thomas Bornstein

Paulo O. Boaventura Netto

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 1978

RESUMEN

El presente trabajo tiene por objetivo desarrollar un algoritmo para análisis de sensibilidad y post-otimización em programación cero-uno, basado en el algoritmo de enumeración implicita de Balas.

Son presentadas las técnicas para resolver problemas de análisis de sensibilidad y de post-otimización de la función objetivo, del segundo miembro y de los coeficientes de las restricciones.

Tambien fué elaborado un programa en lènguaje Fortran para resolver esta classe de problemas.

RESUMO

O presente trabalho tem por objetivo desenvolver um algoritmo para análise de sensibilidade e pós-otimização em programação zero-um, baseado no algoritmo de enumeração implícita de Balas.

São apresentadas as técnicas para resolver problemas de análise de sensibilidade e de pós-otimização da função objetivo, do segundo membro e dos coeficientes das restrições

Também foi elaborado um programa em linguagem Fortran para resolver esta classe de problemas.

ABSTRACT

This work intends to develop an algorithm for sensibility analysis and post-optimality zero-one programming, based upon implicit enumeration Balas' algorithm.

It describes the ways (techniques) to solve problems of sensibility analysis and post-optimality from objective function, of second member, and of the constrained coefficients.

It was also prepared a FORTRAN program in order to solve this kind of problems.

AGRADECIMIENTOS

Al profesor Félix Eduardo Vaca Obando, orientador y amigo que en todo momento con absoluta seguridad me mostro el camino a seguir.

A la COPPE; en las personas: Coordinador del programa de Ingeneiria de Sistemas y Computación Prof.João Lizardo R. H. de Araújo y al Prof. Nelson Maculan Filho, por las grandes facilidades proporcionadas haciendo que este trabajo mejorase.

A los funcionarios de la Biblioteca Central, en especial a la seccion periódicos y revistas por el desprendimiento y atención.

Al Núcleo de Computación Electrónica por las facilidades prestadas.

A la organización de Estados Americanos (OEA) por el apoyo financiero.

A todos los que directa o indirectamente, colaboraron para que marcase un hito mas en mi vida.

INDICE

PÁGINA

C A P Í T U L O I

POST-OPTIMIZATION EN PROGRAMACIÓN BIVALENTE

I. INTRODUCIÓN

Un gran número de problemas de decisión pueden formularse como problemas de programación entera bivalente, esto es, problemas de programación entera cuyas variables solamente pueden asumir valores zero o uno. Modelos matemáticos de esta forma sirven para la solución de: presupuestos de capitales, selección de proyectos, flujos y redes de comunicación, recuperación de información, designación de interruptores de circuitos, detección de defectos, designación de experimentos, facilidad de localización, problemas de designación, dimensionamiento de flotas de veículos y muchos otros problemas. Vários de estos problemas, matematicamente, pueden ser asociados con grafos y considerados como problemas de optimización combinatória.

La enumeración implícita pertenece a la misma clase general de métodos de enumeración del tipo Branch-and-Bound propuesto para programación entera-mixta por LAND Y DOIG |[13]|. En el presente trabajo se hará referencia al algoritmo aditivo desarrollado por BALAS |[1]|. Estos dos algoritmos presentan diferencias a pesar de generar ambos una red tipo árbol para la búsqueda de una solución. Entre esas diferencias tenemos:

a. El algoritmo de Lang y Doig se basa en la solución de un problema lineal para cada vértice del árbol, en cuanto el al goritmo de Balas usa tests lógicos.

b. La estratégia de búsqueda del algoritmo de Lang y Doig es del tipo: "BREADTH FIRST" que da como resultado un gran re querimiento de almacenamiento, mientras que el procedimento de Balas es "DEPTH FIRST" da un módico requerimiento de al macenamiento.

ROODMAN |10| y posteriormente PIPER Y ZOL TNERS |11| han adaptado el algoritmo de Balas para hacer un análisis post-optimal, Roodman crea entonces, un método para analisis de Rango y cambios de parâmetros. El método es basa- do primero en coleccionar información durante la fase soluci- ón, informàción que consiste en un lista de soluciones parcia les "sondeadas" por el algoritmo. En la fase post-optimal se hará un aňálisis individual de cada una de estas soluciones parciales sondeadas para hallar un nuevo óptimo si él existe.

La simplicidad de los tests en el algoritmo básico de Balas hacen posible que en cada sondeo se pueda al macenar información de la solución parcial. Roodman, añade al algoritmo de Geoffrion un test de inviabilidad que causa un sondeo violento y reduce el número de iteraciones del algo ritmo.

C A P Í T U L O II

II. EL PROBLEMA DE PROGRAMACION ZERO-UNO

2.1. La forma general del problema de programación lineal con variables zero-uno puede ser escrito como sigue:

Hallar $\underline{x}$ que:

$$\text{Minimice (maximice) } z = \underline{c}\ \underline{x} \qquad (1')$$

(P1) sujeto a:

$$A\ \underline{x} \underset{>}{\leq} \underline{b} \qquad (2')$$

$$x_j \in \{0,1\}^n \ (j \in N) \qquad (3')$$

donde

$\underline{x} = (x_j)$ vector-columna con n componentes

$\underline{c} = (c_j)$ vector-fila con n componentes

$A = (a_{ij})$ matriz m x n

$\underline{b} = (b_i)$ vector-columna com m componentes

$\underline{c}$, A, $\underline{b}$ son constantes conocidas.

$M = \{1, 2, \ldots, m\}$; $N = \{1, 2, \ldots, n\}$.

2.2. Forma Standard del Modelo Matemático

La forma standard del problema de programación zero-uno es:

$$
(P2)\quad \begin{cases} \min z = \underline{c}\ \underline{x} & (1) \\ \text{sujeto a:} & \\ Ax \leq b & (2) \\ x_j \in \{0,1\}^n \ (j \in N) & (3) \\ \underline{c} \geq \underline{0} & \end{cases}
$$

donde todos los coeficientes de la función objetivo a minimizar deben ser NO NEGATIVOS. La función objetivo (1') facilmente se transforma en (1) con el siguiente cambio de variables:

$$
x_j = \begin{cases} x'_j \ ; \text{ para } c'_j \geq 0 \text{ cuando se minimiza y para } c'_j \leq 0 \text{ si se maximiza.} \\ 1-x'_j \ ; \text{ para } c'_j < 0 \text{ cuando se minimiza y para } c'_j > 0 \text{ cuando se maximiza.} \end{cases}
$$

Las restricciones (2') se transforman en (2) reemplazando todas las ecuaciones (igualdad) por dos inecuaciones y multiplicando por - 1 todas las desigualdades de la forma $\geq$.

Entonces la forma general (P1) es trans

formada en la forma standar (P2) donde:

$\underline{c} \geq 0$ y $\underline{x}$, $\underline{c}$, A y $\underline{b}$ son obtenidos desde $\underline{x}'$, $\underline{c}'$, A' y $\underline{b}'$según las transformaciones anteriores.

2.3. Otras Formas

(P2) también se puede expresar como:

$$\text{minimizar } z = \sum_{j=1}^{n} c_j \; x_j$$

sujeto a:

$$\sum_{j=1}^{n} a_{ij} \; x_j \leq b_i \qquad i = 1,2,\ldots,m$$

$$x_j \; \varepsilon \; \{0,1\}^n \quad j = 1,2,\ldots,n$$

donde $c_j \geq 0 \quad \forall_j = 1,2,\ldots,n$

2.4. Definiciones

D_1 - Cualquier n-vector binário $\underline{x}$ es llamado una SOLUCIÓN.

D_2 - Una solución que satisface A $\underline{x} \leq b$ es llamada <u>SOLUCIÓN POSIBLE</u>.

D_3 - Una solución posible que minimiza z es llamada de <u>SOLUCIÓN OPTIMAL</u>.

D_4 - Una <u>SOLUCIÓN PARCIAL</u> es una asignación de valores binários a un subconjunto de las variables x_j.

D_5 - Los elementos del subconjunto que constituyen una solución parcial se denominan de <u>VARIABLES FIJAS</u> que pueden tomar valores 0 ó 1.
El resto de variables son llamadas <u>LIBRES</u>.

D_6 - Si a una solución parcial se le agrega una variable (una libre que va a ser fijada) se dirá que es una <u>SOLUCIÓN AUMENTADA</u>.

D_7 - Una solución parcial se dice que es <u>DESCENDENTE</u> de otra, si puede ser obtenida por aumento de esta última.

D_8 - Una solución parcial es <u>COMPLETA</u> si desde la última solución parcial se fijan todas las variables libres.

2.5. Notación

Denominaremos por S_k una solución parcial caracterizada por el conjunto J_k de indices de las variables x_j iguales a 1. J_k es un conjunto ORDENADO tal que muestra la historia de la generación de las variables. A cada solución parcial está asociado un Z_k valor de la función objetivo de la respectiva solución parcial.

Sea S_t una solución parcial tal que $J_k \subset J_t$, se dirá entonces que S_t es una solución descendente de S_k.

Ejemplo:

$$S_k = (0,1,0,1,0,1,1)$$

$$J_k = \{2,7,4,6\}$$

$$S_t = (0,1,1,1,0,1,1)$$

$$J_t = \{2,7,4,6,3\}$$

$$J_k \subset J_t$$

C A P Í T U L O III

III. ENUMERACION IMPLICITA

Enumeración Implicita es un método de programación especialmente dedicado para el caso de variables zero-uno (x_j = 0 ó1). El principio de este método es generar una red tipo árbol tal que garantice la enumeración de todas las 2^n soluciones, teniendo como idea básica enumerar sistematicamente sólo un subconjunto de todas las posibles soluciones usando tests lógicos y la propriedad binária, de tal manera que se asegure que, la totalidad del conjunto de soluciones haya sido "examinado" implicitamente, de aqui el nombre del método.

En general, el espacio solución de un problema entero puede ser engañoso por poseer un número finito de posibles puntos viables. Un método directo para resolver problemas enteros es la enumeración exahustiva o explicita de todos los puntos. En este caso, la solución optimal es determinada por el punto (S) que da el mejor valor a la función objetivo.

La obvia desvantaja de la técnica anterior es que el número de puntos solución impracticablemente grande, de como resultado que la solución no sea encontrada en razonable tiempo médio. La idea de enumeración implicita, es llamada así, por considerar sólo una porción de todos los posibles puntos solución, al mismo tiempo que automaticamente descarta algunas que no son viables.

Consideremos, a manera de ilustración, determinar todas las soluciones posibles para la siguiente inecuación:

$$3x_1 - 8x_2 + 5x_3 \leq -6$$

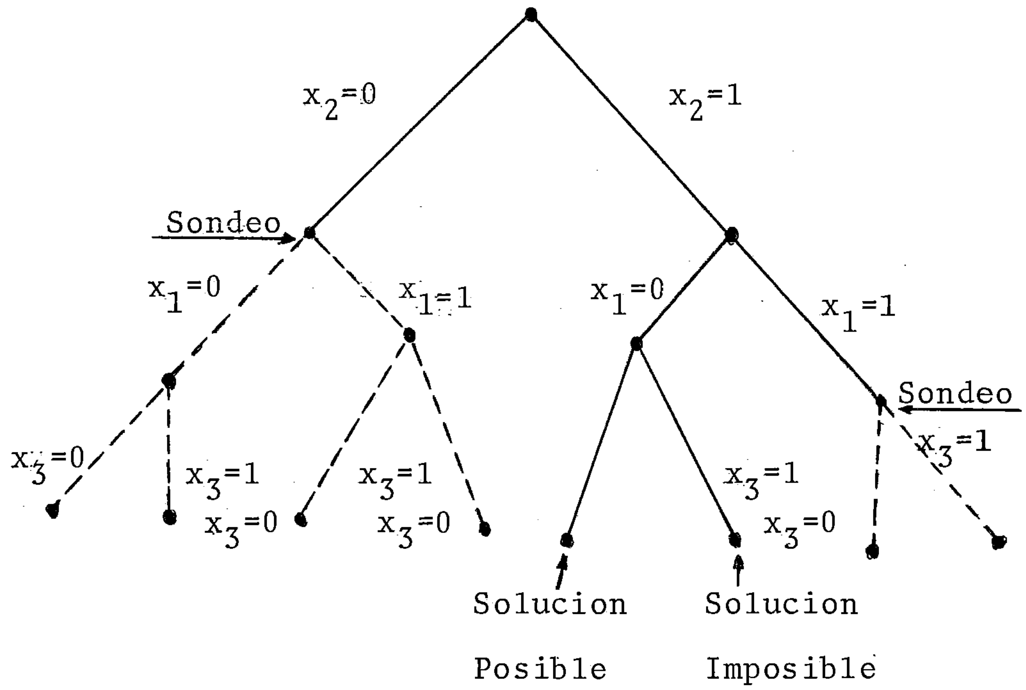
$$x_j = (0,1) \quad j = 1,2,3$$

Por simple inspección vemos que cualquier solución posible tendrá x_2 fijada igual a 1. Esto es, cualquier combinación binária (x_1, 0 , x_3) no puede dar una solución viable. Quiere decir que las combinaciones (0,0,0), (1, 0,0), (0,0,1) y (1,0,1) son automaticamente descartadas. Dado que x_2 = 1 la inecuación queda como:

$$3x_1 + 5x_3 \leq 2, \qquad x_2 = 1$$

x_1 y x_3 deben tomar valores tales que al elemento de la derecha no exceda de 2. Como el coeficiente de x_1 es 3, x_1 debe ser fijado en cero quiere decir que (1,1,0) (1,1,1) são enumeraciones implicitas no viables. De igual manera el coeficiente de x_3 es 5 y (0,1,1) será descartado, la única combinación que resta es (0,1,0) que es una solución posible para la inecuación.

Para apreciar el impacto del uso de enumeración implícita, representemos graficamente esta idea para el ejemplo anterior



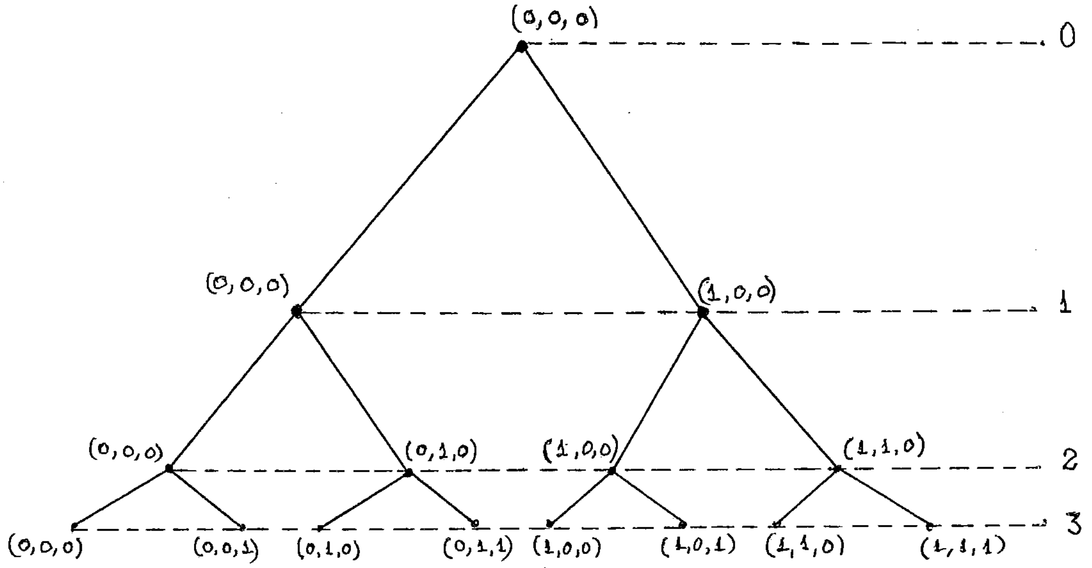
La primera conclusión es que x_2 debe ser fijada en uno, esto es, todos los "brazos" del árbol que nacen de $x_2=0$ son descartados. En este caso $x_2=0$ es llamado un "SONDEO". Para $x_2=1$, la inecuación demuestra que $x_1=1$ no puede dar una solución viable es entonces que ($x_2 = 1$, $x_1 = 1$) es un "SONDEO". Notemos que $x_2 = 1$, $x_1 = 0$, $x_3 = 1$ da una solución imposible y $x_2 = 1$, $x_1= 0$, $x_3=0$ es una solución posible. Estoquiere decir que todas las 2^3 combinaciones han sido consideradas.

C A P Í T U L O IV

IV. EL ALGORITMO DE BALAS

El conjunto de todas las soluciones del problema (P2) es igual a 2^n, siendo n el número de variables.

Por ejemplo si n = 3 las soluciones representadas en un árbol serian:



El algoritmo de Balas se propone "Podar" el árbol mediante tests lógicos en cada vértice, utilizando también las propriedades inherentes del modelo.

El modelo (P2) puede ser modificado con la introducción de variables de holgura y_i así:

$$
(P3)\quad \left\{\begin{array}{ll}
\text{minimizar } z = \sum_{j=1}^{n} c_j\, x_j & \\
\text{sujeto a:} & \\
\sum_{j=1}^{n} a_{ij}\, x_j + y_i = b_i & i=1,2,\ldots,m \\
x_j \in' \{0,1\}^n & j=1,2,\ldots,n \\
y_i \geq 0 & i=1,2,\ldots,m
\end{array}\right.
$$

Se verifica que la función objetivo del problema P3 será siempre mayor o igual a cero puesto que x_j y c_j son no-negativos para todo $j = 1,2,\ldots,n$.

En el modelo (P3) si $y_i = b_i \quad \forall i=1,2,\ldots,n$ y si $x_j = 0 \quad \forall j = 1,2,\ldots,m$ entonces $S_o = (0,0,0,\ldots,0)$ es una solución viable al problema (P3).

El algoritmo de Balas se basa en la idea de partir de una solución S_o viable tal que $J_o = \phi$ e ir buscando una solución parcial en cada nivel del árbol. La búsqueda de la solución es hecha en profundidad "podando" el árbol en cada solución inviable encontrada, retornando a la última solución parcial viable encontrada y reiniciar la búsqueda en profundidad.

Intuitivamente el algoritmo de Balas funciona como sigue:

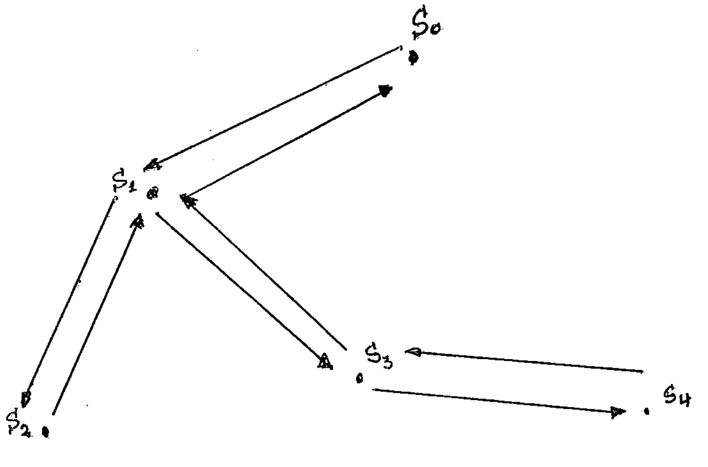
Paso 1: Partir de una solución S_o; $\bar{z} = \infty$

Paso 2: Generar una solución S_i, si es inviable, desistir de la búsqueda de soluciones descendentes de S_i porque serán peores ya que se está minimizando la función objetivo ; $\overline{z} = z_i$.

Paso 3: Retornar a una solución parcial por el camino recorrido hasta S_i y generar nuevamente una solución S_k.

Si no existen mas variables a generar, parada final, sino comparar z_k con z_i. Si $z_k < z_i \Longrightarrow \overline{z} = z_k$, $i = k$ ir a paso 2.

Graficamente ilustramos el proceso:



4.1. Algoritmo (*)

Paso 0 - Poner $\bar{z} = +\infty$ y $k = 0$, $S_k = \phi$,
$\vec{y}_{s_k} \leftarrow \vec{b}$; $\vec{x}_k = \vec{0}$.

Paso 1 - Si todos los $\vec{y}_{s_k} \geq 0$ ir a Paso 5.
Caso contrário continue.

Paso 2 - Calcular el conjunto T_{s_k}.

$T_{s_k} = \{j$ libre/$\underline{c}\ \underline{x}_{s_k} + c_j < \bar{z}$ y

$a_{ij} \geq 0$ para algún i tal que

$y_i^{s_k} < 0\}$.

Paso 3 - Si $y_i^{s_k} + \sum_{j \in T_{s_k}} \text{Max}\ \{0, a_{ij}\}$ es menor que zero para algún i tal que $y_i^{s_k} < 0$ ir a Paso 6, en otro caso continue.

(*) - Este algoritmo es una versión simplificada del algoritmo aditivo de Balas, dado por ARTHUR M. GEOFFRION |2|.

Paso 4 - Aumentar un $j \in T^{s_k}$ a S_k tal que sea el máximo de

$$\sum_{i=1}^{m} \min \{y_i^{s_k} + a_{ij}, 0\} \quad \forall j \in T^{s_k}$$

$k = k + 1$, $S_k \leftarrow S_{k-1}$

Ir a <u>Paso 1</u>.

Paso 5 - Si $\underline{c}\ \underline{x}^{s_k}$ es menor que $\bar{z}$ entonces $\bar{z} \leftarrow \underline{c}\ \underline{x}^{s_k}$ y $\hat{x} \leftarrow x^{s_k}$. Sino Paso 6.

Paso 6 - Retorno: localizar el elemento que está mas a la derecha aún no subrayado, liberar los de la derecha al último no subrayado, sino existe fin. Caso contrário $k = k + 1$, $S_k \leftarrow S_{k+1}$ ir a Paso 1.

4.2. <u>Generación del Conjunto</u> T_{s_k}

Vale la pena detallar la generación del conjunto T^{s_k} definido como:

$$T^{s_k} = \{j \text{ libre}/\underline{c}\ \underline{x}^{s_k} + c_j < \bar{z} \text{ y } a_{ij} \geq 0 \text{ para algún } z' \text{ tal que } y_i^{s_k} < 0\}.$$

Sea por ejemplo la matrix $A\overline{x} + \overline{y} = \overline{b}$ siendo:

$$A = \begin{bmatrix} -1 & 4 & -3 & 5 & -7 \\ 8 & -6 & -5 & 1 & 1 \\ 0 & 2 & -1 & 9 & -1 \end{bmatrix} \quad b = \begin{bmatrix} -2 \\ 0 \\ -4 \end{bmatrix}$$

$$-x_1 + 4x_2 - 3x_3 + 5x_4 - 7x_5 + y_1 = -2$$

$$8x_1 - 6x_2 - 5x_3 + x_4 + x_5 + y_2 = 0$$

$$2x_2 - x_3 + 9x_4 - x_5 + y_3 = -4$$

$$y_1 = -2 + x_1 - 4x_2 + 3x_3 - 5x_4 + 7x_5$$

$$y_2 = 0 - 8x_1 + 6x_3 + 5x_3 - x_4 - x_5$$

$$y_3 = -4 \quad - 2x_2 + 1x_3 - 9x_4 + x_5$$

supongamos ahora que $\overline{z} = \infty$ y todos los j libres. j represen - tan las columnas e i las filas. Los $y_i < 0$ son $i = 1 \wedge i = 3$. Son vālidas entonces solo las filas 1 y 3.De Los coeficientes correspondientes a estas filas cumplen la condiciōn sōlo las columnas $j = 1$, $j = 3 \wedge j = 5$. Entonces el conjunto

$$T = \{1, 3, 5\}.$$

4.3. Ejemplo Numérico (Algoritmo de Balas)

$$\text{Minimizar} \quad 5x_1 + 7x_2 + 10x_3 + 3x_4 + x_5$$

$$\begin{aligned} \text{sujeto a:} \quad -x_1 + 3x_2 - 5x_3 - x_4 + 4x_5 &\leq -2 \\ 2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 &\leq 0 \\ x_2 - 2x_3 + x_4 + x_5 &\leq -1 \end{aligned}$$

Aumentando las variables de holgura.

$$\begin{aligned} -x_1 + 3x_2 - 5x_3 - x_4 + 4x_5 + y_1 &= -2 \\ 2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 + y_2 &= 0 \\ x_2 - 2x_3 + x_4 + x_5 + y_3 &= -1 \end{aligned}$$

$$\begin{aligned} y_1^o &= -2 + x_1 - 3x_2 + 5x_3 + x_4 - 4x_5 \\ y_2^o &= 0 - 2x_1 + 6x_2 - 3x_3 - 2x_4 + 2x_5 \\ y_3^o &= -1 + 0x_1 - x_2 + 2x_3 - x_4 - x_5 \end{aligned}$$

Paso 0 - $K = 0$

$\bar{z} = + \infty$

$x_o = (0,0,0,0,0)$, $S = \{\phi\}$

Paso 1 - $y_1^o = -2$

$y_2^o = 0$

$y_3^o = -1$

(C) el conjunto de variables libres $(j) = \{1,2,3,4,5\}$. (Columnas)

(F) conjunto de filas tal que y_i es menor que zero $(i) = \{1,3\}$.

Paso 2 - Cálculo del conjunto T:

Vemos si existe por lo menos un a_{ij} mayor que cero en las filas del conjunto (F) para las columnas libres, es decir el conjunto (C).

Esto se verifica en la columna correspondiente a la variable $j = 1$ y filas 1 y 3 de (F), los coeficientes a_{ij} son:

$$y_1 = \ldots + 1\ x_1 \ldots\ldots$$

$$\ldots\ldots\ldots\ldots\ldots\ldots \qquad T = \{1, \ldots\ldots$$

$$y_3 = \ldots + 0\ x_1 \ldots\ldots$$

Analogamente para las columnas donde $\exists$ por lo menos un $a_{ij} > 0$.

$$y_1 = \ldots + 1\ x_1 \ldots + 5\ x_3 + 1\ x_4 \ldots$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$y_3 = \ldots + 0\ x_1 \ldots + 2\ x_3 - \ x_4 \ldots$$

$T = \{1,3,4\}$.

<u>Paso 3</u> - Verificación si el conjunto $(y_i + \sum_{j \varepsilon T} \text{Max} \{0, a_{ij}\}$ es menor que zero. $\forall i \ \varepsilon \ F$.

$F = \{1,3\}$, $T = \{1,3,4\}$

$$i = 1 \Longrightarrow -2 + \max\{0,1\} + \max\{0,5\} + \max\{0,1\}$$

$$= -2 + 1 + 5 + 1 = 5 \geq 0$$

$$i = 3 \Longrightarrow -1 + \max\{0,0\} + \max\{0,2\} + \max\{0,-1\}$$

$$= -1 + 2 \geq 0$$

<u>Paso 4</u> - Aumento de una variable $j \in T$ tal que sea máximo de $\sum_{i=1}^{m} \text{mín}\ \{y_i + a_{ij}, 0\}$ $\forall j \in T$.

$j = 1 \Rightarrow$ mín{-2+1,0}+mín{0-2,0}+mín{-1,0}=-1-2-1=-4

$j = 3 \Rightarrow$ mín{-2+5,0}+mín{0-3,0}+mín{0+2,0}=0-3+0=-3

$j = 4 \Rightarrow$ mín{-2+1,0}+mín{0-2,0}+mín{-1-1,0}=1-2-1=-5

Max {-4,-3,-5} = -3, $\boxed{j = 3}$

$k = 1, x_1=(0,0,1,0,0)$, $S_1 = \{3\}$

<u>Paso 1</u> - fijando la variable 3 en 1 tenemos:

$$y_1^1 = -2 + x_1 - 3x_2 + 5x1 + x_4 - 4x_5$$

$$y_2^1 = 0 - 2x_1 + 6x_2 - 3x1 - 2x_4 + 2x_5$$

$$y_3^1 = -1 + 0x_1 - x_2 - 2x1 - x_4 - x_5$$

$$y_1^1 = 3 + x_1 - 3x_2 + x_4 - 4x_5$$

$$y_2^1 = -3 - 2x_1 + 6x_2 - 2x_4 + 2x_5$$

$$y_3^1 = 1 + 0x_1 - x_2 - x_4 - x_5$$

$y_1^1 = 3$

$y_2^1 = -3$

$y_3^1 = 1$

C = {1,2,4,5} variables libres (3 prē fijado)

F = {2} ($y_i^k < 0$)

<u>Paso 2</u> - $y_2^1 = -3 - 2x_1 + 6x_2 - 2x_4 + 2x_5$

T = {2,5}.

<u>Paso 3</u> - $i = 2 \Rightarrow -3 + \max\{6,0\} + \max\{2,0\} = 5>0$

<u>Paso 4</u> - $j = 2 \Rightarrow \min\{(3-3),0\} + \min\{(3+6),0\} +$

$+\min\{(1-1),0\} = 0+0+0=0$

$j = 5 \Rightarrow \min\{(3-4),0\} + \min\{(-3+2),0\} +$

$+\min\{(1-1),0\} = 1-1+0=-2$

$\text{Max}\ \{0,-2\} = 0 \quad \Rightarrow \quad \boxed{j = 2}$

$k = 2 \quad x_2 = (0,1,1,0,0),\ S_2 = \{3,2\}$

Paso 1 : $y_1^2 = 3 + x_1 - 3x1 + x_4 - 4x_5$

$y_2^2 = -3 - 2x_1 + 6x1 - 2x_4 + 2x_5$

$y_3^2 = 1 + 0x_1 - 1x1 - x_4 - x_5$

$y_1^2 = 0 + x_1 + x_4 - 4x_5$

$y_2^2 = 3 - 2x_1 - 2x_4 + 2x_5$

$y_3^2 = 0 \quad - x_4 - x_5$

$y_1^2 = 0$

$y_2^2 = 3$

$y_3^2 = 0$

Paso 5 : $\vec{c}\,\vec{x}^s = [5,7,10,3,1]\begin{bmatrix}0\\1\\1\\0\\0\end{bmatrix} = 17$

Como $\vec{c}\,\vec{x}^s = 17$ es menor que $\bar{z} = \infty$, entonces $\bar{z} = 17 \quad \hat{x} = (0,1,1,0,0)^T$.

$k = 3$

Paso 6 : $S_3 = \{3,-2\}$ lo que implica que retornamos a $x_3 = (0,0,1,0,0)$, $c = \{1,4,5\}$.

<u>Paso 1</u> : $y_1^3 = 3 + x_1 - 3x_2 + x_4 - 4x_5$

$y_2^3 = -3 - 2x_1 + 6x_2 - 2x_4 + 2x_5$

$y_3^3 = 1 + 0x_1 - x_2 - x_4 - x_5$

Notemos aqui que j = 2 (columna 2) no entra en el test porque C ={1, 4,5}.

<u>Paso 2</u> : T = {5} , F = {2}.

<u>Paso 3</u> : $i = 2 \Rightarrow -3 + \max\{2,0\} = -3+2 = -1 < 0$

<u>Paso 6</u> : k = 4

$S_4 = \{-\underline{3},-\underline{2}\}$ $C = \{1,4,5\}$.

<u>Paso 1</u> : $y_1^4 = -2 + x_1 - 3x_2 + 5x_3 + x_4 - 4x_5$

$y_2^4 = 0 - 2x_1 + 6x_2 - 3x_3 - 2x_4 + 2x_5$

$y_3^4 = -1 + 0x_1 - x_2 + 2x_3 - x_4 - x_5$

$y_1^4 = -2$

$\Longrightarrow y_2^4 = 0$

$y_3^4 = -1$

<u>Paso 2</u> : T = {1,4} F = {1,3}

$i = 1 \Rightarrow -2 + \max\{1,0\} + \max\{1,0\} = -2+1+1 = 0$

$i = 3 \Rightarrow -1 + 0 + 0 = -1 < 0$

Paso 6 : como todos los elementos de S están subrayados entonces el algoritmo termina siendo la solución óptima:

$\hat{x} = (0,1,1,0,0)$ con $\bar{z} = 17$.

4.4. Algunas Aplicaciones

4.4.1. Aplicación 1

Recuperación de Información |14|

Introdución

El problema de seleción óptima de Archivos puede ser tratado como un problema de Programación Entera Bivalente.

Se trata del problema de extración óptima de una información que está contenida en varios archivos.

El almacenamiento típico de un Sistema de Información contiene numerosos items clasificados en categorias de várias formas. Cada categoría por ejemplo puede contener: ciudad, sector industrial, tipo de produto y así sucesivamente.

Un registro de información contiene diversos items que pueden pertenecer a una o más y diferentes categorias. Un registro puede sumarizar información concisa extraída desde un documento publicado o no, una fuente verbal, u otra información o fuente inteligente. Un archivo de información contiene un número grande de registros.

El Problema de Recuperación

Un sistema de almacenamiento general puede ser clasificado con respecto a los registros en super-categorias y que constituirá un archivo distinto. Cada archivo entonces contiene registros con items correspondientes a las mas detalladas categorías, sistema llamado también como, sistema multiplo de almacenamiento de Archivo de Datos, e cuya caracteristica típica es la simultaniedad de caracteres, esto es, un registro dado puede contener información relativa a más de una supercategoria y por esta razón puede ser que está duplicado en diversas partes del sistema. Un pedido de información para un cierto item, específico y perteneciente a una categoria puede ser solucionada con la búsqueda en los diferentes archivos, en consecuencia un gran consumo de tiempo , por la gran cantidad de composiciones. Si se tiene un gran número de pedidos el tiempo consumido en la búsqueda del archivo se hará considerable. El problema en este caso será minimizar el tiempo de selección del archivo y búsqueda del registro con el item correspondiente . El problema general de minimizar tiempo de computador en procesos grandes, con muchos grupos de pedidos es complicado, por factores técnicos próprios del computador, número de cintas o archivos que pueden ser interrogados simultaneamente, así como también la estructura y longitud de los diversos archivos y las prioridades de los diversos pedidos.

Modelo para el Problema

El modelo de selección de archivos es resuelto por programación entera bivalente (modelo simplificado) facilmente puede ser extendido para el problema mas general.

Supongamos que son m archivos A_1, A_2, ..., A_m desde donde la información puede ser extraída. Sea x_i, $i = 1,2,\ldots,m$, una variable de desición con la siguiente condición:

$x_i = 1$ el archivo A_i es seleccionado

$x_i = 0$ el archivo A_i no es seleccionado.

También supongamos que cada archivo tiene una longitud definida por el número de registros que contiene. Sea l_i la longitud del archivo A_i, $i = 1,2,\ldots,m$. Supongamos que se tiene K pedidos P_1, P_2, P_3, ..., P_K y sea a_{ij} un "coeficiente de sa -

tisfacción" que describe la satisfacción o no del pedido P_i por el archivo A_j. Si P_i puede ser satisfecho por selección de A_j entonces $a_{ij} = 1$.

$$a_{ij} = 0 \text{ en otro caso:}$$

$$\text{El vector columna } a_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{kj} \end{bmatrix}$$

describe los pedidos que pueden o no ser satisfechos por el archivo A_j.

$$\text{El vector fila } a_i = |a_{i1}, a_{i2}, \ldots, a_{im}|$$

describe los archivos en el cual puede o no ser usado y satisface al pedido P_i.

$$\text{La matriz } A = \begin{bmatrix} a_{11}, & \ldots, & a_{1m} \\ a_{21}, & \ldots, & a_{2m} \\ \vdots & \ldots, & \ldots \\ a_{k1}, & \ldots, & a_{km} \end{bmatrix}$$

describe la estructura de "satisfacción"que ocurre en el grupo de pedidos. Asumimos que un archivo (y sólo uno) puede ser seleccionado en cada vez. Se desea que cada pedido sea satisfecho, y que todos los items de información contenidos en el sistema total de archivos y que correspondan a cada pedido sea extraído. El requerimiento de cada pedido puede ser expresado por la siguiente inecuación lineal.

$$a_{i1}\, x_1 + a_{i2}\, x_2 + \ldots + a_{im}\, x_m \geq 1$$

$$i = 1,2,\ldots,k$$

$$x_i \; \varepsilon \; (0,1)$$

esto es por lo menos un x_i debe ser igual a 1 y que corresponde a un a_{ij} también igual a 1. Esto es, para todo pedido (indice i) por lo menos un archivo (indice j) debe ser satisfecho. La función objetivo es expresada:

$$\text{Mín } x_o = l_1\, x_1 + \ldots + l_m\, x_m$$

que es, minimizar el tiempo total de búsqueda donde l_i e x_i son variables enteras, x_i es bivalente.

Modelo:

$$\text{Mín } x_o = l_1\, x_1 + \ldots + l_m\, x_m$$

$$\text{sujeto a: } Ax \geq 1$$

$$x_i \; \varepsilon \; (0,1)$$

Ilustración

Supongamos que se tiene 4 archivos y siete pedidos, la matriz describe las "satisfacciones".

	A_1	A_2	A_3	A_4
P_1	1	1	1	0
P_2	1	1	0	1
P_3	1	0	1	1
P_4	1	0	0	1
P_5	0	1	1	1
P_6	0	1	0	1
P_7	0	0	1	1
	l_1	l_2	l_3	l_4
	16	8	4	49

los cuatro archivos tienen longitud 16, 8, 4 y 49 respectiva - mente. El coeficiente de la fila P_1 y columna A_1 es uno lo que indica que el primer archivo puede satisfazer el 1º pedido, asî tambiēn el archivo 3 no satisface el pedido 2, etc. En este ejemplo 2 candidatos existen para la soluciōn:

	x_1	x_2	x_3	x_4	x_o
soluciōn 1	1	1	1	0	28
soluciōn 2	0	0	0	1	49

la soluciōn optima, evidentemente es: soluciōn 1. Aparentemente esta soluciōn no es la mejor porque tenemos que testar 3 archivos, mientras que la soluciōn 2 solamente 1 archivo, pero en tērminos de tiempo de computador y propositos mas generales la solución 1 es la mejor. Si la longitud l_4 fuera 28 entonces la solución será la mas eficiente, el archivo A_4 tiene información duplicada lo que puede aumentar considerablemente el tiempo de computo por la necesidad de clasificar y descartar registros duplicados. En realidad no siempre la "longitud" tomada como número de registros puede ser la mas conveniente. Otros factores complicados pueden ser considerados como "pesos" l_i $(i = 1,2,\ldots,m)$ o modificaciones en la misma matriz A.

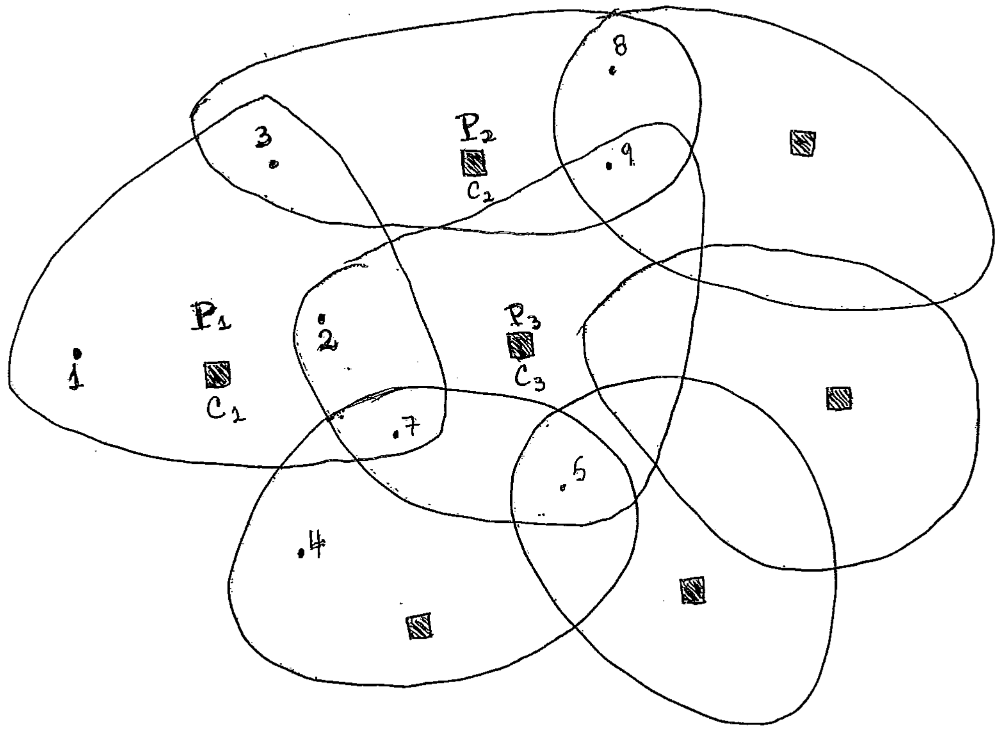
4.4.2. Aplicación 2:

Problema de Asignación

Supongamos que en un cierto pais se quiera instalar un red de repetidoras de un canal de televisión de tal manera que la red cubra todo el território del País. Haciendo un estudio geográfico e infraestructura se ha recolectado un número determinado de lugares donde se podería instalar una repetidora y también el número de puntos vivos que cada repetidora daria una señal de televisión, eficientemente:

$i = 1,2,\ldots,M$ (puntos vivos; Ej: ciudades)

P_j repetidora en el lugar j. $j=1,2,\ldots,n$.



$i = 1, \ldots , M$ ciudades

$c_1 \ldots, c_n$ costos en cada lugar puesta una repetidora

$P_1, \ldots P_n$ punto donde se encontrá la repetidora.

Modelo:

	P_1	P_2	P_3		
	x_1	x_2	x_3		$j = 1,2,\ldots,n$
$i = 1$	1	0	0		
2	1	0	1		
3	1	1	0		
4	0	0	0		
5	0	0	1	A	
6	0	0	0		
7	1	0	1		
8	0	1	0		
9	0	1	1		
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
m	0	0	0		

$$\text{Mín} \sum_{j=1}^{m} c_j \, x_j$$

sujeto a:

$$\sum_{j=1}^{n} a_{ij} \, x_j \geq 1 \qquad \forall i = 1, \ldots, m$$

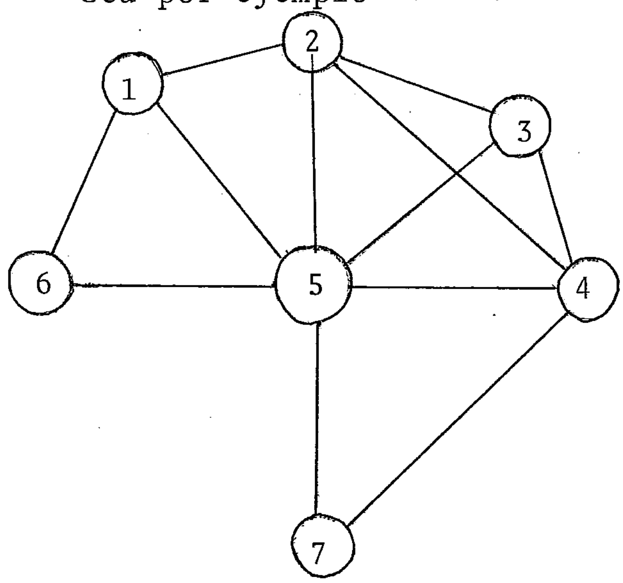
$$x_j \; \varepsilon \; (0,1)$$

4.4.3. Aplicación 3

Problema de los Colores

Se tiene una red no orientada tal que se quiere dar un color a cada vértice de tal forma que ningún vértice adyacente tenga el mismo color. Este problema puede formularse como un modelo de programación bivalente de la forma siguiente:

Sea por ejemplo la red:



Primero se debe determinar los "subconjuntos estables" máximos entendiendose por subconjunto estable aquél conjunto de vértices que no son adyacentes, y será máximo aquél que tenga mayor número de vértices. Por ejemplo:

El vértice 6 tiene como conjunto estables:

Subconjunto estable

$$\begin{array}{c|ccc|cc|}
1 & 0 & 0 & 0 & 1 & 1 \\
2 & 1 & 0 & 0 & 0 & 0 \\
3 & 0 & 1 & 0 & 1 & 0 \\
4 & 0 & 0 & 1 & 0 & 1 \\
5 & 0 & 0 & 0 & 0 & 0 \\
6 & 1 & 1 & 1 & 0 & 0 \\
7 & 1 & 1 & 1 & 1 & 1
\end{array}$$

Subconjunto estables para el vértice 6 — Para 1

El subconjunto estable máximo para cada vértice está dado por la variable x_i. El problema de los colores está resuelto, con el modelo entero bivalente:

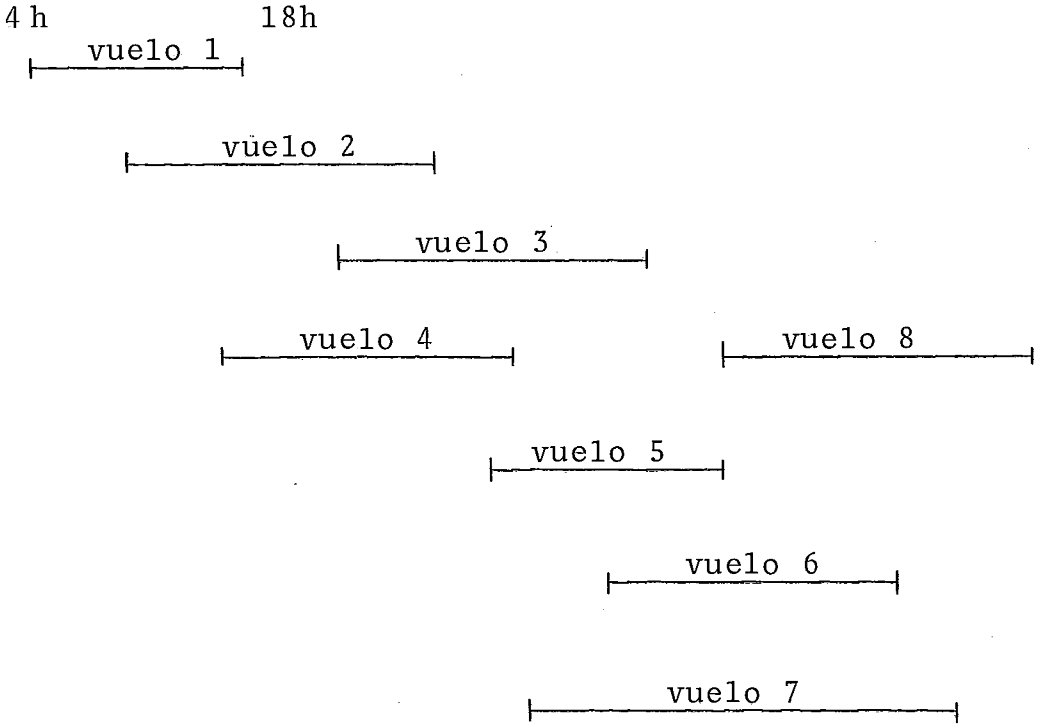
$$\text{Mín} \quad \sum x_i$$

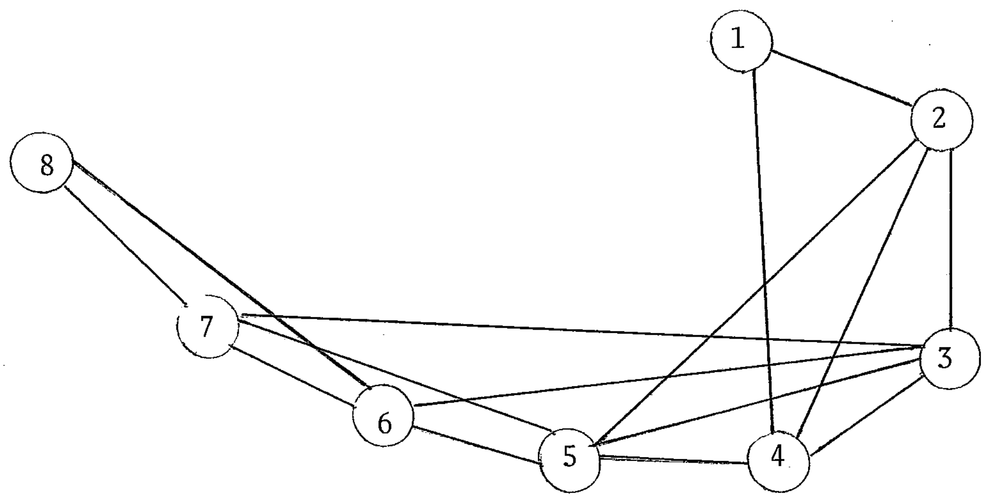
$$Ax \geq 1$$

$$x_i \in (0,1)$$

Este modelo es aplicado en la planificación de vuelos aereos con el diseño siguiente:

Supongamos que una compañia aerea tenga 10 vuelos a la semana y se quiere usar el menor número de aviones que cubran esos vuelos. Se construye una tabla de cada vuelo con el respectivo horario de vuelo , y se diseña un grafo de tal manera que cada vértice represente un vuelo y cada arista une dos vértices cuando el horario es incompatible, es decir , hay cruce de horario, así se construirá un grafo del tipo anterior. El problema es entonces; minimizar el número de aviones:




1
2
8
7
3
6
5
4

CAPÍTULO V

V. EXTENSIONES PARA EL ALGORITMO DE BALAS

Consideramos extensiones al algoritmo de Balas, modificaciones en el modelo standard y el própio algoritmo respetando la base y filosofia del algoritmo.

El algoritmo de modificaciones podemos agruparlos en 2:

a. Modificaciones al modelo:

   i. función objetivo no-lineal y restricciones lineales.

   ii. Forma polinomial de la función objetivo y restricciones.

b. Extensiones al algoritmo. Analisis post-optimal.

5.1. Función Objetivo No-Lineal y Restricciones Lineales

El esquema enumerativo fundamental de Balas depende muy poco de la natureza lineal de la función objetivo, podemos modificar los pasos 2 para el cálculo de T y el paso 5 para reemplazar el valor de z para poder manipular funciones - objetivos no-lineales f(x), esto es:

$$\text{minimizar } f(x)$$

sujeto a:

$$Ax \leq b$$

$$x \in \{0,1\}^n$$

El Problema No-Lineal Zero-Uno

El problema no-lineal zero-uno, puede ser resuelto por enumeración implícita, provistos de funciones que satisfagan ciertas condiciones de monotonía.

El problema no-lineal debe aparecer en la forma:

$$\text{minimizar} \quad z(x) = g_{o1}(x) - g_{o2}(x)$$

sujeto a:

$$g_{i1}(x) - g_{i2}(x) \leq b_i \qquad i = 1,2,\ldots,m$$

donde $\quad x = (x_1,x_2,\ldots,x_n), \; x_j = (0,1), \; j=1,2,\ldots,n$

Una restricción importante es que cada una de las funciones

g_{o1}, g_{o2}, g_{11}, $g_{12}, \ldots, g_{m1}$, g_{m2} es monotona no-decreciente en cada una de las variables x_1, $x_2, \ldots, x_n$.

El algoritmo de Balas versión Geoffrion, sigue siendo válido para las siguientes modificaciones

i. El conjunto T definido como:

$$T = \{j \text{ libre}/f(x^{s/j}) < \bar{z} \text{ y } \tilde{a}_{ij} > 0 \text{ para algún } i \text{ tal que } y_i^s < 0\}.$$

donde $x^{s/j}$ es justamente el vector x^s con la j-ésima variable complementada. $\tilde{a}_{ij}$ definida así:

$$\tilde{a}_{ij} = \begin{cases} a_{ij} & \text{si} \quad x_j^s = 0 \\ -a_{ij} & \text{si} \quad x_j^s = 1 \end{cases}$$

ii. $\tilde{a}_{ij}$ reemplaza a_{ij} en los pasos 3 y 4

iii. $f(x^s)$ reemplaza $\vec{c}\,\vec{x}^s$ en paso 5.

Modelos del tipo función objetivo y restricciones no lineales aparecen en problemas de alocación y asignación de objetivos, normalmente del tipo probabilístico, por ejemplo:

$$f(\vec{x}) = \sum_k v_k \Pi_j (p_{jk})^{x_j}$$

donde $v_k \geq 0, \; 0 \leq p_{jk} \leq 1$

$$g(\vec{x}) = - \Pi_j (1 - p_j^{x_j})$$

donde $0 \leq p_j \leq 1$

Tambiēn se encuentra aplicabilidad en problemas de reemplazo y analisis marginal en programaciōn dinãmica.

5.2. Forma Polinomial en la Funciōn Objetivo y Restricciones

Consideremos el problema de programaciōn no-lineal

Optimizar $f(x_1, x_2, \ldots, x_n)$

sujeto a:

$$g_i (x_1, x_2, \ldots, x_n) \leq b_i \quad (i = 1, 2, \ldots, m)$$

donde $x_1, x_2, \ldots, x_n$ asumen valores enteros.

$f(x_1,x_2,...,x_n)$ y $g_i(x_1,x_2,...,x_n)$ son polinomios.

Consideremos una función simple, de la forma cuadrática $g(x_1,x_2,...,x_n)$ en la cual las variables deben ser 0-1. Obviamente que x_j^2 puede ser reemplazada por x_j y en nada será afectado el valor de la función.

El producto de dos variables x_j x_k es reemplazado por una nueva variable x_{jk} según el valor correspondiente a los valores de x_j y x_k así:

x_j	x_k	x_{jk}
0	0	0
0	1	0
1	0	0
1	1	1

resultado que podemos expresarlo como:

$$x_j + x_k - x_{jk} \leq 1$$

$$-x_j - x_k + 2x_{jk} \leq 0$$

$$x_{jk} = 0 \text{ ó } 1$$

resultado que se puede generalizar facilmente:

En general, dado un conjunto Q, compuesto de q variables bivalentes. El producto $\prod_{j \in Q} x_j^p$ (para cualquier va lor positivo de p) y puede ser reemplazado por la nueva variable x_Q e imponer las siguientes restricciones adicionales:

$$\sum_{j \varepsilon Q} x_j - x_Q \leq q - 1$$

$$-\sum_{j \varepsilon Q} x_j + q\, x_Q \leq 0$$

$$x_Q = 0 \text{ ó } 1.$$

C A P Í T U L O VI

VI. ANALISIS POST-OPTIMAL EN PROGRAMACIÓN ZERO - UNO POR ENUMERACIÓN IMPLÍCITA

6.1. Generalidades

Un proceso competente para analisis post-optimal de programación zero-uno es desarrollado por Roodman G. |10|, y es obtenido a partir del algoritmo de enumeración implícita de Balas, empleando información coleccionada durante la optimización del problema inicial para su análisis consecuente.

PIPER Y ZOLTERN |11| a partir de las ideas de Roodman hacen unas modificaciones mas finas a los test de colección y clasificación de las soluciones parciales generadas por el algoritmo aditivo. En este trabajo el algoritmo a ser utilizado es el de GEOFFRION (enumeración implícita) |2|. El análisis post-optimal comprende:

1. Analisis de Rango para los costos y flujo exógeno.
2. Cambio de parámetros para los costos, flujo exógeno y elementos de la Matriz, así como también eliminación de restricciones y adición de nuevas variables.

6.2. Algoritmo de Colección y Clasificación de Soluciones Parciales

Este algoritmo acumula la información requerida para la post-optimization del problema entero zero-uno.

$$\text{Mín} \sum_{j=1}^{n} c_j \, x_j$$

sujeito a:

$$\sum_{j=1}^{n} a_{ij} \, x_j \leq b_i \qquad i \in M = \{0,1,\ldots,m)$$

$$x_j \in \{0 \text{ ó } 1\} \qquad j \in N = \{1,2,\ldots,n\}$$

con $\qquad c_j \geq 0 \qquad \forall j \in N.$

Definimos $a_{oj} = c_j$ $(j \in N)$ y $b_o = \bar{z}$, donde $\bar{z}$ es un limite superior válido sobre la solución optimal, llamada fila zero ó "restricción de la función objetivo".

El proceso de búsqueda de la enumeración implícita consiste en generar soluciones parciales en un intento de hallar un zero-completo posible. Una solución parcial es llamada "sondeada":

a) Si es un zero-completo posible

b) Si se puede demostrar que ninguma de estas completitudes puede dar una solución posible mejor que la encontrada por la solución posible hallada hasta el momento.

Después de hallar cada zero-completo posible, la "restricción función objetivo" es actualizada tal que sólo soluciones menores o iguales al valor de la función objetivo son consideradas. Por ejemplo:

Supongamos el mejor cero-completo posible encontrado que para la función objetivo de un valor $\bar{z}$ entonces la restricción de la función objetivo será:

$$\sum_{j=1}^{n} a_{oj}\ x_j \leq \bar{z} \quad \text{esto es } b_o \text{ es } \bar{z}$$

La clasificación de cada solución parcial es dada por la identidad de la restricción y por el valor de la función objetivo.

Definimos el siguiente conjunto de soluciones parciales "sondeadas".

Λ_f = {solución tal que el cero-completo es posible}

$$\Lambda_k = \{\text{solución tal que el cero-completo } S \text{ es imposible y es sondeada sólo por la restricción } k\} \quad k = 0,1,\ldots,m.$$

$$\Lambda = \Lambda_f \cup \Lambda_k$$

Los conjuntos Λ_γ $(\gamma = f, 1,\ldots,m)$ forman una partición de todas las soluciones parciales sondeadas y generadas durante la solución del problema zero-uno usando e numeración implícita.

Asociamos el siguiente conjunto de indices para cada solución parcial obtenida durante la búsqueda:

$$J_S^1 = \{j \in N \, / \, x_j \, (S) = 1\}$$

$$J_S^o = \{j \in N \, / \, x_j \, (S) = 0\}$$

$$N_S = N - J_S^1 - J_S^o$$

N_S es el conjunto de indices de las variables libres con respecto a la solución S.

J_S^1 y J_S^o son conjuntos de indices para las variables fijas.

Usando estos conjuntos de índices definimos las siguientes variables para cada fila $i \varepsilon M$:

$$t_i\ (S) = b_i - \sum_{j \varepsilon J_S^1} a_{ij}$$

$$\rho_i\ (S) = \sum_{j \varepsilon N_S} \min\ \{a_{ij}, 0\}$$

$$\gamma_i\ (S) = t_i\ (S) - \rho_i\ (S)$$

esta última cantidad, $\gamma_i(S)$, puede ser interpretada como el límite superior sobre la holgura en la restricción i que puede acontecer desde cualquier completitud de S. Consecuentemente, valores negativos de γ_i (S) no son permitidos que ocurran durante el cálculo, lo que implica que son testadas las imposibilidades de todas las completitudes de S.

El algoritmo testa las siguientes dos implicaciones:

POSIBLE: Si $t_i\ (S) \geq 0 \quad \forall i \varepsilon M$, la solución parcial S es posible.

IMPOSIBLE: Para cualquier solución parcial S y $j \varepsilon N_S$ definimos el siguiente conjunto:

$$G = \{i \in M \ / \ \gamma_i \ (S) < | \ a_{ij}|\}.$$

Si $G \neq \phi$ es una condición para que todas las completitudes de S sean imposibles.

Algoritmo de Colección y Clasificación de Soluciones Parciales

Paso 1 : Obtener valores de p, J_S^1, J_S^o, N_S y $\Lambda_\gamma (\gamma = f,0,1,m)$ Cuando el algoritmo es usado para resolver el problema cero-uno inicializar $p = 0$, $J_S^1 = \phi$, $J_S^o = \phi$, $N_S = N$, $\Lambda_\gamma = \phi$ para $\gamma = f$, $0,1,2,\ldots,m)$

Paso 2 : Calcular t_i (S), ρ_i (S) Λ γ_i (S) . Si t_i (S) < 0 para algún $i \in M$ ir a paso 3 ,en otro caso, S es posible con valor de la función objetivo z(S).
Aumentar S a Λ_f.
Hacer $b_o = \bar{z} = z$ (S) y $p = p + 1$.
RETORNO, e ir a paso 2.

Paso 3 : Teste de Roodman, cálculo de G; GE

NERAR y si γ_i (S) = t_i (S) - ρ_i(S) < | a_{ij}| para algún i y algún j ε N_S (digamos i = k y j = γ) entonces termina el examen de todas las completitudes de S en el cual:

$$x_\gamma = 0 \quad \text{si} \quad a_{k\gamma} < 0$$

$$x_\gamma = 1 \quad \text{si} \quad a_{k\gamma} \geq 0$$

Esto quiere decir que ocurre sondeo para la solución parcial S' formada por S con $x_\gamma^{S'} = 0$ si $a_{k\gamma} < 0$ ó $x_\gamma^{S'} = 1$ si $a_{k\gamma} \geq 0$.

El aumento de la solución parcial al conjunto Λ_k es atribuible a la restricción k. RETORNO e ir a paso 2. En otro caso"SKIP"la solución parcial e ir a paso 3.

6.3. Análisis Post-Optimal

La simplicidad de los tests en el algoritmo básico aditivo hacen posible, que en cada solución parcial examinada se atribuya la restricción que causó el sondeo.

Esta forma de particionamiento de las soluciones parciales facilitan encontrar limites de optimalidad, es to es, análisis tipo Rango y también selección de parametros para su posible cambio.

El procedimiento en líneas generales será usar el conjunto Λ, ya que una nueva completitud será encontrada a partir de un miembro de Λ porque cualquier partición de Λ induce una partición de 2^n soluciones. Este hecho es usado en 2 fases:

La primera, una partición de las 2^n soluciones es llevado a cabo por particionamiento de Λ. Esta partición es usada en la segunda fase al hacer el analisis post-optimal.

6.3.1. Analisis de Rango

6.3.1.1. Elementos de la Derecha (flujo exógeno) (b_j)

El rango no es otra cosa que el limite superior e inferior en el intervalo en el que el elemento b_k puede ser cambiado sin modificar la solución optima actual, ni alguna solución parcial deje de ser "sondeada".

Sean b_k y b_k^+ límites inferior y superior, entonces para cualquier cambio b_k' definido en:

$$\boxed{b_k - b_k \leq b_k' \leq b_k + b_k^+}$$

mantendrá la optimalidad actual.

Definimos b_k^- y b_k^+ de la siguiente manera:

Para cada $S \varepsilon \Lambda_k$, $k \neq 0$

$$b_k^+ (S) = -1 - \gamma_k (S)$$

$$b_k^+ (S) = \min_{S \varepsilon \Lambda_k} \{b_k^+ (S) \; / \; z (S) \leq z^p\}$$

z (S) es el valor de la función objetivo en la solución parcial. Así mismo

$$b_k^- (S) = t_k (S)$$

$$b_k^- = \max_{S \varepsilon \Lambda_f} \{b_k^- (S)\}$$

El máximo de Λ_f ó mínimo de Λ_k es para todo S en que x_γ es libre ó $x_\gamma^S = 1$.

6.3.1.2. Elementos de la Función Objetivo (Costos) (c_j)

El problema inicial mantendrá la solución optimal si los costos permanecen en el rango:

$$c^*_\gamma > c_\gamma - w_\gamma$$

donde $w_\gamma = \text{mín} \{\phi\ (S)\}$

$$\Phi\ (S) = \begin{cases} z(S) - z^p + c_\gamma & \text{si } x_\gamma \text{ libre en solución parcial} \\ z(S) - \bar{z} & \text{en otro caso} \end{cases}$$

mín. de $\phi(S)$ es para todo $S \in \Lambda_f$ en el cual x_γ es libre ó $x^S_\gamma = 1$.

6.3.1.3. Elementos de la Matriz A (I,J)

El proceso de análisis Rango de los coeficientes de la matriz A es similar al de los elementos de la de recha b_k.

Definimos $v_{k\gamma} = \min \{\psi\ (S')\}$

donde

$$\psi\ (S') = \begin{cases} \gamma_k\ (S) + a_{k\gamma} & \text{si } x_\gamma \text{ es libre y} \\ a_{k\gamma} \geq 0 & \\ \gamma_k\ (S') & \text{en otro caso.} \end{cases}$$

el mínimo $\{\psi\ (S)\}$ es para todo $S \varepsilon \Lambda_k$ en el cual x_γ es libre ó $x_\gamma^S = 1$.

6.3.2. Cambio de Parametros

Con cualquier solución parcial S, se puede asociar un problema parcial de la forma.

$$(IP^S) \begin{cases} \min z = \sum_{j=1}^{n} c_j\ x_j + z^S \\ \text{sujeto a:} \\ \sum_{j=1}^{n} a_{ij}\ x_j \leq b_i^S \qquad i \varepsilon M = \{1,\ldots,m\} \end{cases}$$

$$x_j \in \{0,1\} \ , \ j \in N_S \text{ (variables libres)}$$

donde

$$z^S = \sum_{j \in J_S^1} c_j \, x_j^S \quad y \quad b_i^S = b_i - \sum_{j \in J_S^1} a_{ij} \, x_j^S$$

Cada solución para (IP^S) juntándola con S forma una completitud para (IP). La mejor solución de (IP^S) será la mejor completitud de S para (IP). También cualquier solución posible (imposible) para (IP^S) será también completitud posible (imposible) de S para (IP).

6.3.2.1. Cambio de los Elementos del Segundo Miembro (b_k)

Para cada solución parcial $S \in \Lambda_k$, formular un problema parcial (IP^S) usando el valor revisado b_k y un valor inicial $\bar{z}$, determinado por la mejor solución posible, hasta el momento encontrada (en la fase solución o en cualquier problema parcial previamente resuelto). Resolver este problema completamente, usando el algoritmo aditivo. Cuando todos los problemas parciales (IP^S) $S \in \Lambda_k$, han sido examinados, una nueva solución optimal, si existe será encontrada.

6.3.2.2. Cambio de los Elementos de la Función Objetivo (c_γ)

Para cada S ε Λ_f, en el cual x_γ no es fijado a cero, formular el problema parcial (IP^S) usando el valor revisado de c_γ y un valor inicial de $\bar{z}$ determinado por la mejor solución posible hasta el momento encontrada. Cuando cada una de las soluciones parciales (IP^S) ha sido examinada, la nueva solución, si existe será encontrada.

6.3.2.3. Cambio en los Elementos $a_{k\gamma}$ ε A

En este caso se procede sobre el conjunto de soluciones parciales pertenecientes a la restrición k, en la misma forma que para cambio en b_k.

6.4. Ejemplo Ilustrativo de Análisis Post-Optimal

$$\text{Mín} \quad z = 2x_1 + 5x_2 + 5x_3 + 2x_4 + 4x_5 + y_6 + 8x_7 + x_8$$

sujeto a:

$$-2x_1 + 0x_2 + 0x_3 + 2x_4 - 6x_5 + x_6 - x_7 + 2x_8 \leq -5$$

$$-4x_1 + 11x_2 - 11x_3 - 7x_4 + 4x_5 + 3x_6 - 5x_7 + x_8 \leq -6$$

$$0x_1 + x_2 + x_3 - x_4 - x_5 - 2x_6 + 0x_7 + x_8 \leq 0$$

$$y_1^1 = -5 + 2x_1 + 0x_2 + 0x_3 - 2x_4 + 6x_5 - x_6 + x_7 - 2x_8$$

$$y_2^1 = -6 + 4x_1 - 11x_2 + 11x_3 + 7x_4 - 4x_5 - 3x_6 + 5x_7 - x_8$$

$$y_3^1 = 0 + 0x_1 - x_2 - x_3 - x_4 + x_5 + 2x_6 + 0x_7 - x_8$$

$t' = (9999,-5,-6,0)$ $\qquad$ $F = \{1,2\}$

$c = \{1,2,3,4,5,6,7,8\}$

$\rho' = (0,-9,-27,-3)$ $\qquad$ $T = \{1,3,4,5,7\}$

$i = 1$ $\qquad -5 + \max\{-6 + 4,0\} + \max\{0,0\} +$

$\qquad + \max\{0,-2\} + \max\{0,6\} + \max\{0,1\} = 4$

$i = 2 \quad \Rightarrow \quad -6 + 4 + 11 + 7 + 5 = 21$

$j = 1 \quad \Rightarrow \quad \text{Min}\ \{-5 + 2,0\} + \text{min}\ \{-6 + 4,0\} +$

$\text{min}\ \{0 + 0,0\} = -3 - 2 + 0 = -5$

$j = 3 \quad \Rightarrow \quad \text{Min}\ \{-5 + 0,0\} + \text{min}\ \{-6 + 11,0\} +$

$+ \text{min}\ \{0 - 1,0\} = -5 + 0 - 1 = -6$

$j = 4 \quad \Rightarrow \quad \text{Min}\ \{-5 - 2,0\} + \text{min}\ \{-6 + 7,0\} +$

$+ \text{min}\ \{0 -1,0\} = -7 + 0 - 1 = -8$

$j = 5 \quad \Rightarrow \quad \text{Min}\ \{-5 + 6,0\} + \text{min}\ \{-6 - 4,0\} +$

$+ \text{min}\ \{0 + 1,0\} = 0 - 10 + 0 = -10$

$j = 7 \quad \Rightarrow \quad \text{Min}\ \{-5 + 1,0\} + \text{min}\ \{-6 + 5,0\} +$

$+ \text{min}\ \{0 + 0,0\} = -4 - 1 + 0 = -5$

$S = \{7\}$

$$y_1^2 = -4 + 2x_1 + 0x_2 + 0x_3 - 2x_4 + 6x_5 - x_6 - 2x_8$$

$$y_2^2 = -1 + 4x_1 - 11x_2 + 11x_3 + 7x_4 - 4x_5 - 3x_6 - x_8$$

$$y_3^2 = 0 + 0x_1 - x_2 - x_3 - x_4 + x_5 + 2x_6 - x_8$$

$$t^2 = (9991, -4, -1, 0)$$

$$\rho^2 = (0, -8, -22, -7)$$

$$\begin{bmatrix} -8 \\ -22 \\ -3 \end{bmatrix} \quad \begin{bmatrix} 2^+ & 0^+ & 0^+ & 2^- & 6^+ & 1^- & 2^- \\ 4^+ & 11^- & 11^+ & 7^+ & 4^- & 3^- & 1^- \\ 0^+ & 1^- & 1^- & 1^- & 1^+ & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} -4 \\ -1 \\ 0 \end{bmatrix}$$

$$\rho_i^S + |a_{ij}| > t_i^S$$

$a_{15} > 0$ $\quad j = 5$ fija

$S = \{7, \underline{5}\}$ variable 5 fijada a 1 por restric-
ción 1. Neste caso guardar solu-
ción $\{7, -5\}$

$$y_1^3 = 2 + 2x_1 + 0x_2 + 0x_3 - 2x_4 - x_6 - 2x_8$$

$$y_2^3 = -5 + 4x_1 - 11x_2 + 11x_3 + 7x_4 - 3x_6 - x_8$$

$$y_3^3 = 1 + 0x_1 - x_2 - x_3 - x_4 + 2x_6 - x_8$$

$t^3 = (9987,\ 2,\ -5,\ 1) \qquad C = \{1,2,3,4,6,8\}$

$\rho^3 = (0,\ -2,\ -22,\ -2) \qquad T = \{1,3,4\}, F = \{2\}$

$i = 2 \Rightarrow -5 + \text{Max}\{0,4\} + \text{Max}\{0,10\} + \text{Max}\{0,7\} = -5+4+11+$

$+ 7 = 17 > 0$

$j = 1 \Rightarrow \text{Mín}\{2+2,0\} + \text{Mín}\{-5+4,0\} + \text{Mín}\{1+0,0\} =$

$= 0 - 1 + 0 = -1$

$j = 3 \Rightarrow \text{Mín}\{2+0,0\} + \text{Mín}\{-5+11,0\} + \text{Mín}\{1-1,0\} =$

$= 0 + 0 + 0 = 0$

$j = 4 \Rightarrow \text{Mín}\{2-2,0\} + \text{Mín}\{-5+7,0\} + \text{Mín}\{1-1,0\} =$

$= 0 + 0 + 0 = 0$

$S = \{7,\ \underline{5},\ 4\}$ Solución posible, restricción 0, guardar $\{7,\ \underline{5}, 4\}$

$$y_1^4 = 0 + 2x_1 + 0x_2 + 0x_3 - x_6 - 2x_8$$

$$y_2^4 = 2 + 4x_1 - 11x_2 + 11x_3 - 3x_6 - x_8$$

$$y_3^4 = 0 + 0x_1 - x_2 - x_3 + 2x_6 - x_8$$

$t^4 = (9981,\ 0,\ 2,0)$

Solución: $(0,0,0,1,1,0,1,0)$ $\quad \overline{z} = 14$

$\rho^4 = (0,\ -2,\ -15,\ -2)$ $\quad A = (0,0,2,0)$

$\rho = (0,\ -2,\ -15,\ -2)$

Retorno:

$S = \{7,\ \underline{5},\ -\underline{4}\}$

$C = \{1,2,3,6,8\}$ $\quad F = \{2\}$ $\quad T = \{1,3\}$

$$y_1^5 = 2 + 2x_1 + 0x_2 + 0x_3 - x_6 - 2x_8$$

$$y_2^5 = -5 + 4x_1 - 11x_2 + 11x_3 - 3x_6 - x_8$$

$$y_3^5 = 1 + 0x_1 - x_2 - x_3 + 2x_6 - x_8$$

$\rho^5 = (0,\ -2,\ -15,\ -2)$

$t^5 = (6,\ 2,\ -5,\ 1)$

$$\begin{bmatrix} 2 \\ -5 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 2^+ & 0^+ & 0^+ & 1^- & 2^- \\ 4^+ & 11^- & 11^+ & 3^- & 1^- \\ 0^+ & 1^- & 1^- & 2^+ & 1^- \end{bmatrix} \qquad \begin{bmatrix} -2 \\ -15 \\ -2 \end{bmatrix}$$

$S = \{7,\ \underline{5},\ -\underline{4},\ -\underline{2}\}$

Variable 2 fijada a cero por restricción 2 guardar {7, $\underline{5}$, -$\underline{4}$, 2}.

$$y_1^6 = 2 + 2x_1 + 0x_3 - x_6 - 2x_8$$

$$y_2^6 = -5 + 4x_1 + 11x_3 - 3x_6 - x_8$$

$$y_3^6 = 1 + 0x_1 - x_3 + 2x_6 - x_8$$

$c = \{1,3,6,8\}$

$t^6 = \{6,2,-5,1\}$

$\rho^6 = (0, -2, -15, -2)$

$$\begin{bmatrix} 2 \\ -5 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 2^+ & 0^+ & 1^- & 2^- \\ 4^+ & 11^+ & 3^- & 1^- \\ 0^+ & 1^- & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} -2 \\ -15 \\ -2 \end{bmatrix}$$

S = {7,$\underline{5}$,-$\underline{4}$,-$\underline{2}$,$\underline{3}$} variable 3 fijada en 1 por restricción 2; guardar {7, $\underline{5}$, -$\underline{4}$, -$\underline{2}$,-3}.

$y_1^7 = 2 + 2x_1 - x_6 - 2x_8$ $\quad c = \{1,6,8\}$

$y_2^7 = 6 + 4x_1 - 3x_6 - x_8$ $\quad t^7 = (1,2,6,2)$

$y_3^7 = 0 + 0x_1 + 2x_6 - x_8$ $\quad \rho^7 = (0,-2,-4,-2)$

Solución posible guardar {7,5,-4,-2,3}.

Retorno:

$S = \{-\underline{7}\}$ $\quad c = \{1,6,8,5,4,2,3\}$

$t^8 = (17,\ -5,\ -6,\ 0)$

$\rho^8 = (0,\ -8,\ -22,\ -3)$

$$y_1^8 = -5 + 2x_1 + 2x_2 - 0x_3 - 2x_4 + 6x_5 - x_6 - 2x_8$$

$$y_2^8 = -6 + 4x_1 - 11x_2 + 11x_3 - 4x_4 - 4x_5 - 3x_6 - x_8$$

$$y_3^8 = 0 + 0x_1 - x_2 - x_3 - x_4 + x_5 + 2x_6 - x_8$$

$$\begin{bmatrix} -8 \\ -22 \\ -3 \end{bmatrix} \quad \begin{bmatrix} 2^+ & 0^+ & 0^+ & 2^- & 6^+ & 1^- & 2^- \\ 4^+ & 11^- & 11^+ & 7^+ & 4^- & 3^- & 1^- \\ 0^+ & 1^- & 1^- & 1^- & 1^+ & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} -5 \\ -6 \\ 0 \end{bmatrix}$$

$S = \{-\underline{7}, \underline{5}\}$ $\qquad c = \{1,6,8,4,2,3\}$

Variable 5 fijada a 1 por restricción 1

guardar solución $\{-7,\ -\underline{5}\}$

$$y_1^9 = 1 + 2x_1 + 0x_2 + 0x_3 - 2x_4 - x_6 - 2x_8$$

$$y_2^9 = -10 + 4x_1 - 11x_2 + 11x_3 + 7x_4 - 3x_6 - x_8$$

$$y_3^9 = 1 + 0x_1 - x_2 - x_3 - x_4 + 2x_6 - x_8$$

$$t^9 = (13,1,-10,1)$$

$$\rho^9 = (0,-2,-22,-2)$$

$(-22)\ (4^+\ 11^-\ 11^+\ 7^+\ 3^-\ 1^-)(-10)$ No

$F = \{2\}$ $T = \{1,3,4\}$

$$i = 2 \Rightarrow -10 + \text{Max}\{0,4\} + \text{Max}\{0,11\} + \text{Max}\{0,7\} =$$
$$= 12 > 0$$

$$j = 1 \Rightarrow \text{Min}\{1+2,0\} + \text{Min}\{-10+4,0\} + \text{Min}\{1+0,0\} =$$
$$= -6$$

$$j = 3 \Rightarrow \text{Min}\{1+0,0\} + \text{Min}\{-10+11,0\} + \text{Min}\{1-1,0\} =$$
$$= 0 \quad \leftarrow$$

$$j = 4 \Rightarrow \text{Min}\{1-2,0\} + \text{Min}\{-10-3,0\} + \text{Min}\{1+2,0\} =$$
$$= -14$$

$S = \{-7,\underline{5},3\}$ $z^{10} = (8,1,1,0)$

$c = \{1,2,4,6,8\}$ $\rho^{10} = (0,-2,-11,-2)$

$$y_1^{10} = 1 + 2x_1 + 0x_2 - 2x_4 - x_6 - 2x_8$$

$$y_2^{10} = 1 + 4x_1 - 11x_2 + 7x_4 - 3x_6 - x_8$$

$$y_3^{10} = 0 + 0x_1 - x_2 - x_4 + 2x_6 - x_8$$

$S = \{-7,\underline{5},3\}$ Solución posible restricción cero. Guardar $\{-\underline{7},\underline{5},3\}$

Retorno:

$S = \{-\underline{7},\underline{5},-\underline{3}\}$

$$y_1^{11} = 1 + 2x_1 + 0x_2 - 2x_4 - x_6 - 2x_8$$

$$y_2^{11} = -10 + 4x_1 - 11x_2 + 7x_4 - 3x_6 - x_8$$

$$y_3^{11} = 1 + 0x_1 - x_2 - x_4 + 2x_6 - x_8$$

$$t^{11} = (5,1,-10,1)$$

$$\rho^{11} = (0,-2,-11,-2)$$

$$\begin{bmatrix} -2 \\ -11 \\ -2 \end{bmatrix} \quad \begin{bmatrix} 2^+ & 0^+ & 2^- & 1^- & 2^- \\ 4^+ & 11^- & 7^+ & 3^- & 1^- \\ 0^+ & 1^- & 1^- & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} 1 \\ -10 \\ 1 \end{bmatrix}$$

$S = \{-\underline{7}, \underline{5}, -\underline{3}, \underline{1}\}$ Variable 1 fijada a 1 por restricción 2 guardar {-7,5,-3,-1}.

$$y_1^{12} = 3 + 0x_2 - 2x_4 - x_6 - 2x_8$$

$$y_2^{12} = -6 - 11x_2 + 7x_4 - 3x_6 - x_8$$

$$y_3^{12} = 1 - x_2 - x_4 + 2x_2 - x_8$$

$$t^{12} = (3,3,-6,1)$$

$$\rho^{12} = (0,0,-7,-2)$$

$$\begin{bmatrix} 0 \\ -7 \\ -2 \end{bmatrix} \quad \begin{bmatrix} 0^+ & 2^- & 1^- & 2^- \\ 11^- & 7^+ & 3^- & 1^- \\ 1^- & 1^- & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} 3 \\ -6 \\ 1 \end{bmatrix}$$

$S = \{-\underline{7},\underline{5},-\underline{3},\underline{1},-\underline{2}\}$ Variable 2 fijada a cero por restricción 2. Guardar $\{-\underline{7},\underline{5},-3,\underline{1},2\}$.

$$t^{13} = (3,3,-6,1)$$

$$\rho^{13} = (0,0,-7,-2)$$

$$y_1^{13} = 3 - 3x_4 - x_6 - 2x_8$$

$$y_2^{13} = -6 + 7x_4 - 3x_6 - x_8$$

$$y_3^{13} = 1 - x_4 + 2x_6 - x_8$$

$$\begin{bmatrix} 0 \\ -7 \\ -2 \end{bmatrix} \quad \begin{bmatrix} 2^- & 1^- & 2^- \\ 7^+ & 3^- & 1^- \\ 1^- & 2^+ & 1^- \end{bmatrix} \quad \begin{bmatrix} 3 \\ -6 \\ 1 \end{bmatrix}$$

S = $\{-\underline{7},\underline{5},-\underline{3},\underline{1},-\underline{2},\underline{4}\}$ Variable 4 fijada a 1 por restricción 2. Guardar $\{-7,5,-3,1,-2,-4\}$.

$$y_1^{14} = 1 - x_6 - 2x_8$$

$$y_2^{14} = 1 - 3x_6 - x_8$$

$$y_3^{14} = 0 + 2x_6 - x_8$$

S = $\{-\underline{7},\underline{5},-\underline{3},\underline{1},-\underline{2},\underline{4}\}$ solución posible.

Retorna: todos marcados fin.

6.4.2. Fase Post-Optimal

Sea el siguiente problema:

$$\text{Mín } z = 2x_1 + 5x_2 + 5x_3 + 2x_4 + 4x_5 + x_6 + 8x_7 + x_8$$

sujeto a:

$$-2x_1 + 0x_2 + 0x_3 + 2x_4 - 6x_5 + x_6 - x_7 + 2x_8 \leq -5$$

$$-4x_1 + 11x_2 - 11x_3 - 7x_4 - 4x_5 + 3x_6 - 5x_7 + x_8 \le -6$$

$$0x_1 + x_2 + x_3 + x_4 - x_5 - 2x_6 + 0x_7 + x_8 \le 0$$

Usando el algoritmo aditivo de Balas, versión Geoffrion y coleccionando las siguientes las soluciones parciales:

Solución Parcial S	z(S)	Restricción,k que causa el sondeo	γ_k (S)
$S_1 = (7,\underline{5})$	8	1	-2
$S_2 = (7,5,4)$	18	Posible	
$S_3 = (7,5,\underline{4},2)$	17	2	-1
$S_4 = (7,5,4,2,3)$	12	2	1
$S_5 = (7,5,\underline{4},2,3)$	17	Posible	
$S_6 = (\underline{7},\underline{5})$	0	1	-3
$S_7 = (\underline{7},5,3)$	9	Posible	
$S_8 = (\underline{7},5,\underline{3},\underline{1})$	4	2	-3
$S_9 = (\underline{7},5,\underline{3},1,2)$	11	2	-10
$S_{10} = (\underline{7},5,\underline{3},1,\underline{2},4)$	12	0	-3
$S_{11} = (\underline{7},5,\underline{3},1,3,4)$	6	2	-6

Esto es, podemos hacer la siguiente partición:

$\Lambda_f = \{S_2, S_5, S_7\}$

$\Lambda_1 = \{S_1, S_6\}$

$\Lambda_2 = \{S_3, S_4, S_8, S_9, S_{11}\}$

$\Lambda_3 = \{\phi\}$

6.4.2.1. <u>Análisis de Rango para</u> $c_j \wedge b_j$

Solución optimal actual; $x_3 = x_5 = 1$, $x_1 = x_2 = x_4 = x_6 = x_7 = x_8 = 0$.

<u>Rango para c_j</u>:

Conjunto de soluciones parciales de la función objetivo:

$z(S)$

$\Lambda_f = \{S_2, S_5, S_{11}, S_7\}\ \{7, 5, 4\}$ $|18|$

$\{7,5,-4,-2,3\}$ $|17|$

$\{-7,5,-3,1,-2,4\}$ $|12|$

$\{-7,5,3\}$ $|4|$

$w_\gamma = \text{mín}\ \{\phi(S)\}\ ;\ \phi(S) = z(S) - \bar{z} + c_\gamma\ x_\gamma$

libre en solución parcial $z(S) - \bar{z}$

Ejemplo: $x_1 = 0$ en solución optimal actual

x_1 libre en S_2, $S_5 \wedge S_7$, en S_{11} no es libre.

$w_1 = \text{mín}\{(18-9+2),(17-9+2),(12-9),(9-9+2\} =$

$= \text{mín}\ \{11,10,3,2\}=2$

$x_2 = 0$ en solución optimal actual y no fijada a cero en $(S_2) \wedge (S_7)$.

$x_4 = 0$

$w_4 = \min \{(18-9), (12-9), (9-9+6)\} = 3$

$w_6 = \min \{(18-9+1), (17-9+1), (12-9+1), (9-9+1)\} = 1$

$w_7 = \min \{(18-9), (17-9)\} = 8$

$w_8 = \min \{(18-9+1), (17-9+1), (12-9+2), (9-9+1)\} = 1$

La solución actual permancerá optima para cualquier

$c^*_\gamma > c_\gamma - w_\gamma$

$(w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8) = (2, 5, 9999, 3, 9999, 1, 8, 1)$

<u>Rango para b_j</u>:

Conjunto de soluciones parciales:

k = 1	$\gamma, z(S)$	k = 2	γ z (S)
(S_1) {7, -5}	{2,8}	(S_3) {7,5,-4,2}	[1,17]
(S_6) {-7,-5}	{3,0}	(S_4) {7,5,-4,-2,-3}	[1,12]
k = 3		(S_8) {-7,5,-3,-1}	[3,4]
No existe		(S_9) {-7,5,-3,1,2}	[10,11]
		(S_{10}) {-7,5,-3,1,-2,-4}	[6,6]

Para k = 1

Calcular $u_k = \min_{S \in \Lambda_k} \{\gamma_k^S\}$ para $z(S) \leq z^P$

$u_1 = \min \{2,3\} = 2$

Para k = 2

Calcular $u_2 = \min \{3,6\} = 3$

Notemos que para k = 2, los valores de γ_S en las soluciones parciales S_3, S_4 y S_9 no cuentan por tener z(S) mayor que 9.

Para $k = 3$

No existen soluciones parciales

$(u_1, u_2, u_3) = (2,3,999)$

La solución optimal actual parmanece todavía optima para b_k en el intervalo

$$b_k - S_k \leq b_k^* < b_k + u_k$$

donde S_k es la holgura de la restricción k del optimo actual.

$$\begin{array}{ll} S_1 = 1 & -6x_5 + S_1 = -5 \\ S_2 = 1 & -11x_3 + 4x_5 + S_2 = -6 \\ S_3 = 0 & x_3 - x_5 + S_3 = 0 \end{array}$$

$-6 \leq b_1^* < -3;\ -7 \leq b_2^* < -3;\ 0 \leq b_3^* < 9999.$

6.4.2.2. <u>Cambio de Parametros</u>

<u>Cambio en el Costo</u>

Si se quiere reducir c_4 a 2 (valor antiguo 6). Analizamos el conjunto Λ_f.

a. la variable x_4 no debe estar fijada a cero en estas soluciones potenciales. S_5 es descartada por esta razón.

b. el valor de z(S) en cada solución parcial debe ser menor que el $\overline{z}$ actual. En la solución S_2 tenemos que x_4 es fijado en 1 y en la reducción del costo c_4 la solución queda en valor z(S) = 14 que no es mayor que la S actual.

c. Generar en problema parcial (IP^S) a partir de la solución (S_{11}) y (S_9).

Para {-7, 5, -3, 1, -2, 4}

Minimizar $x_6 + x_8 + z^S$

sujeto a:

$$x_6 + 2x_8 \leq b_1^S = -5 - (-2 + 2 - 6) = 1$$

$$3x_6 + x_8 \leq b_2^S = -6 - (-4 - 7 + 4) = 1$$

$$-2x_6 + x_8 \leq b_3^S = 0 - (1 - 1) = 0$$

$$z^S = 4x_5 + 2x_1 + \boxed{2}x_4 = 8$$

nuevo valor

Minimizar $x_6 + x_8 + 8$

sujeto a:

$$x_6 + 2x_8 \leq 1$$

$$3x_6 + x_8 \leq 1$$

$$-2x_6 + x_8 \leq 0$$

Solución posible $x_6 = x_8 = 0$ luego $x_1 = x_4 = x_5 = 1$ con $\bar{z} = 8$, solución mejorada.

Cambio en b_k

Incrementemos en -2 el valor de b_1 es decir será -7.

Debemos considerar las soluciones S_1 y S_k'

S_1 : {7, -5} [2,8]

S_6 : {-7, -5} [3,0]

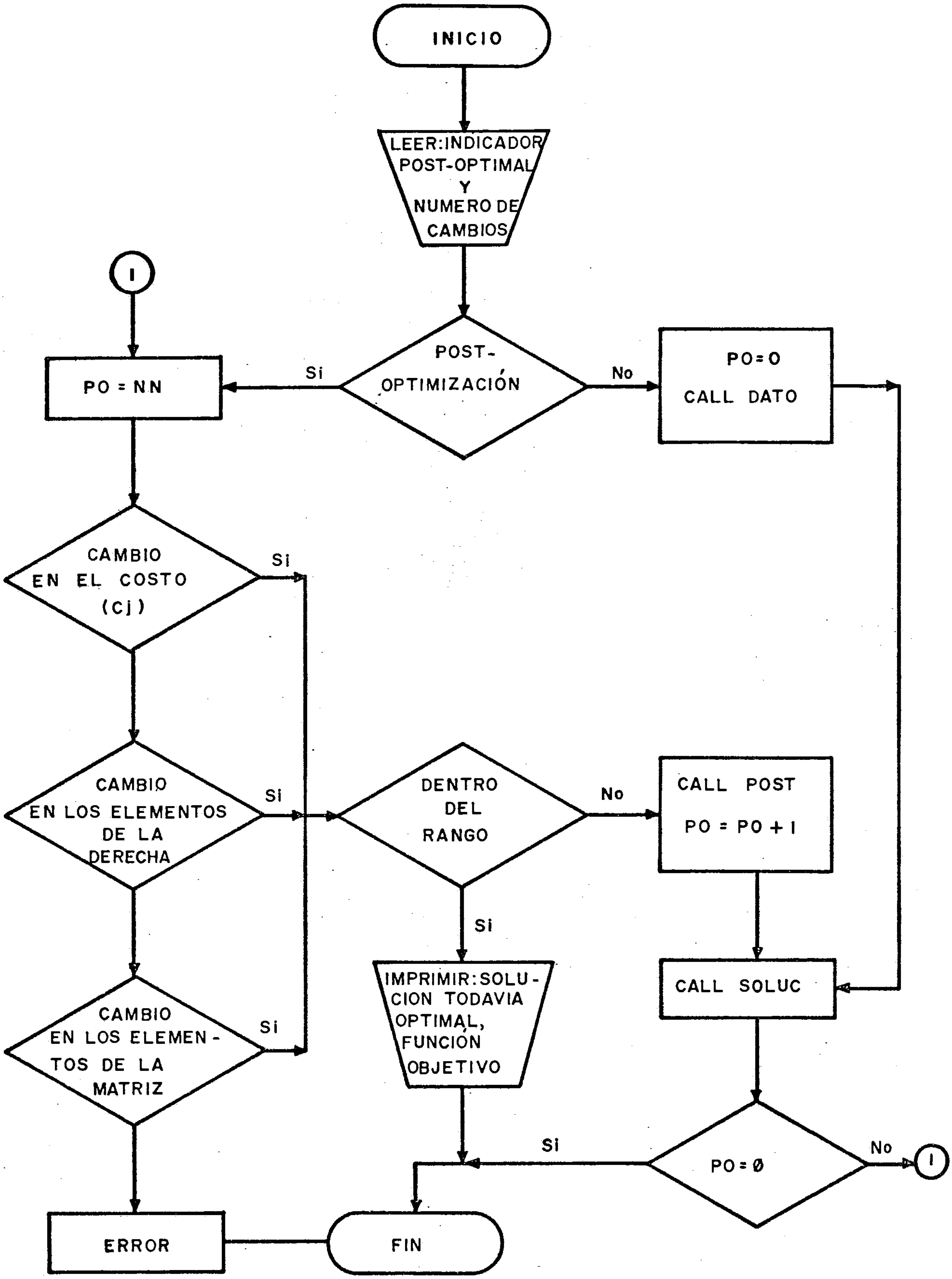
Debemos considerar los problemas parciales para estas das soluciones S_1 y S_6.

$$
(IPS_1)\left\{\begin{array}{l}
\text{Mín } 2x_1 + 5x_2 + 5x_3 + 6x_4 + x_6 + x_8 + 8 \\
\text{sujeto a:} \\
-2x_1 + 2x_4 + x_6 + 2x_8 \leq -6 \\
-4x_1 + 11x_2 - 11x_3 - 7x_4 + 3x_6 + x_8 \leq -1 \\
x_2 + x_3 + x_4 - 2x_6 + x_8 \leq 0 \\
x_j = 0 \text{ ó } 1 \qquad j = 1,2,3,4,6,8
\end{array}\right.
$$

$$
(IPS_2)\left\{\begin{array}{l}
\text{Mín } 2x_1 + 5x_2 + 5x_3 + 6x_4 + x_6 + x_8 \\
\text{sujeto a:} \\
-2x_1 + 2x_4 + x_6 + 2x_8 \leq -7 \\
-4x_1 + 11x_2 - 11x_3 - 7x_4 + 3x_6 + x_8 \leq -6 \\
x_2 + x_3 + x_4 - 2x_6 + x_8 \leq 0 \\
x_j = 0 \text{ ó } 1 \qquad j = 1,2,3,4,6,8.
\end{array}\right.
$$

sin embargo facilmente se puede verificar que estos problemas poseen una solución posible que sea mejor que el optimo original.

DIAGRAMA DE FLUJO DEL ANÁLISIS POST-OPTIMAL



NOMBRE: SYP01 (Solución y Postoptimización 0-1)

FUNCIÓN: Resuelve um problema lineal entero 0-1 de la forma

Mín. C_x

5.a

$A_x \leq b$

$x \in \{0,1\}^n$

$C = |C_1, C_2, \ldots\ldots\ldots, C_n|$

$b = |b_1, b_2, \ldots\ldots\ldots, b_m|^T$

$A = |a_{ij}|_{m \times n}$

- Acumula soluciones parciales que a partir de ellas hace un analisis de rango para cada uma de los variables. Y analisis post-optimal, esto es, cambio de parámetro en los costos, flujo exógeno y elementos de la matriz

MÉTODO: El método utilizado es el algoritmo aditivo de Balas |[1]| versión Geoffrion |[2]|, y para analisis post-optimal, Roodman |[10]|.

ESTRUTUCTURA GENERAL.

El sistema consta de um programa principal maestro, que comanda la ejecución de las diferentes opciones del programa.

Se usan cinco subrutinas que particionan el proceso, en Fase Solución y Fase Post-Optimal.

Estas subrutinas son:

DATO, SOLUC, RANKA, RANKB y RANKC.

La subrutina DATO se encarga de recolectar los datos en Tarjeta ; SOLUC, da la solución al problema 0-1, RANKA, RANKB, RANKC, realizan um analisis de Rango para los elementos de la matriz, flujo exógeno y costos respectivamente.

Ademas se utilizan tres subrutinas auxiliares para el empaquetado de la información binaria y otras para entrada post-optimal.

CARACTERISTICAS GENERALES:

Todos los sub-programas y programas principal tienen sus variables principales en COMMON con la ventaja de ahorro de memoria y en las subrutinas evitar el problema de traslado de um número excesivo de parámetros.

PROGRAMA PRINCIPAL (SYPO1)

FUNCIÓN: Es el programa maestro que dirige la ejecución del resto de programas.

CARACTERISTICAS:

Este programa lee las opciones del proceso, analisá la validez de estos opciones y transfiere el control del programa a una subrutina requerida.

SUBRUTINAS REQUERIDAS: DATO, SOLUC, POST.

RESTRICCIONES: Ninguna

SUBRUTINA DATO

FUNCIÓN: Es la encargada de leer los datos que vienen en tarjetas de la matriz $\vec{A}$ y vectores $\vec{C}$ y b

CARACTERISTICO: La forma de entrada de los datos es um dato por tarjeta para la matriz $\vec{A}$ considerando la fila y columna correspondiente, pudiendo entrar en cualquier orden obviandose la lectura de elementos cero.

RESTRICIONES: Máximo orden de la matriz

$N = 100$; $M = 50$

SUBRUTINA SOLUC

FUNCIÓN: Este subprograma es usado en la fase solución y post-optimales, encontrando soluciones optimales y post-optimales.

CARACTERISTICAS: Usado en la fase solución y enlazado com la subrutina DATO se encarga de Recolectar y almacenar información tipo solucines parciales. Usado en fase post-optimal y enlazado con la subrutina POST, da una nueva solución al problema si es que existe.

SUBRUTINAS REQUERIDAS: EMPAQ, RANKA, RANKB y RANKC.

SUBRUTINAS RANKA

FUNCIÓN: Esta subrutina hace un analisis de rango para la matriz $A = |a_{ij}|$ esto es, los coeficientes de los restricciones

CARACTERISTICAS: Encuentra rangos de variación donde la solución todavia es optimal, utiliza la información almacenada en la fase solución, es decir, la soluciones parciales.

SUBRUTINAS REQUERIDA: BINA

SUBRUTINA RANKB

FUNCIÓN: Se encarga de dar el rango de variación para el flujo exógeno, donde el problema actual permanece optimo.

CARACTERISTICAS: Utiliza soluciones parciales almacenadas en Fase Solución y obtiene un vector de variación.

SUBRUTINA REQUERIDA: BINA

SUBRUTINA RANKC

FUNCIÓN: Encargada de hacer un analisis da rango para el vector costo del problema.

CARACTERISTICAS: A partir de las soluciones parciales, es dado un rango de variación a los costos que indica si la solución continua siendo óptima.

SUBRUTINA REQUERIDA: BINA

SUBRUTINA DECI

FUNCIÓN: Convierte un arreglo de valores 0-1, a un valor decimal almacenado en por lo menos en una palabra.

CARACTERISTICAS: La conversión la hace de 20 en 20 caracteres, es decir por ejemplo si se tiene un arreglo de 35 valores binarios los 20 primeros (de derecha a iz quierda) dan un valor decimal y a partir da la posición 21 al 35 otro valor deci mal.

RESTRICIONES: El maximo valor considerado, esto es, el tamañ del arreglo es de 100.

FORMATO: DECI (XXX, VALOR, N)

XXX: Arreglo que almacena el dato binario

VALOR: Arreglo que contiene el valor binario

N: Número de elementos binarios a ser transformado.

SUBRUTINA BINA

FUNCIÓN: Convierte un valor decimal a binário en un arreglo.

CARACTERISTICAS: El número de variables indicará la dimensión del arreglo necesario y al número de palabras a decodificar.

RESTRICCIONES: Maxima dimensión del arreglo igual 100.

FORMATO: BINA (XXX, VALOR, N)

XXX: Arreglo que se almacenará el valor binario.

VALOR: Arreglo que contiena al valor decimal.

N: Número de elementos binarios obtenidos.

SUBRUTINA EMPAQ

FUNCIÓN: Empaqueta la información de una solución parcial.

CARACTERISTICAS: Almacena en disco la solución parcial, variables libres, variables fijas, valor de la función objetivo para la solución parcial, valor de la holgura de Pas restricciones y la restricción que causó al sondeo.

SUBRUTINA REQUERIDA: DECI

SUBRUTINA POST

FUNCIÓN: Da entrada a una solución parcial para que genere una nueva solución.

CARACTERISTICAS: Formula un nuevo problema a partir de una solución parcial, actualiza el vector de variables libres y da un nuevo valor para b(1), siendo este el valor de Z actual.

ENTRADA y SALIDA DE DATOS

ENTRADA: Existem tres tipos de entradas todas por tarjetas;una para control de Post-optimización, otra para cambios de parámetros y otra para datos del modèlo es decir costos,matriz de restricciones y flujo exógeno

- TARJETA DE CONTROL DE POST-OPTIMIZACIÓN (Tarjeta única y obligatoria)

Si tiene una parforación 9 en columna 2 indica que se hace post-optimización, caso contrario es, fase solución simple.

En caso de ser 9 en las columnas 4 y 5 se perforará el número de tarjetas de cambios que le siguen, siendo una tarjeta por cambio.

FORMATO: Entero (I2,1x, I2)

- TARJETA DE CAMBIO DE PARAMETROS

FORMATO: Entero (2I2, I5)

CC 1-2: Número que indica la columna

CC 3-4: Número que indica la fila

CC 5-9: Nuevo valor del parámetro

Observación:En este caso para mayor facilidad se considera una matriz de ordem (M + 1 x N + 1) donde la 1^{a} fila esta representando los valores de los costos. Y la ultima columna el vector flujo exógeno.

	C_1	C_2	C_3	C_n			
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$			
	Col_1	Col_2	Col_3	Col_n	Col_{n+1}		$\leftarrow b_1$
fila 1	a_{11}	a_{12}	a_{13}	a_{1n}	$a_{1,n+1}$		$\leftarrow b_2$
fila 2	a_{21}	a_{22}	a_{23}	a_{2n}	$a_{2,n+1}$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
fila m+1	$a_{m+1,1}$	$a_{m+1,2}$	$a_{m+1,3}$	$a_{m+1,n}$	$a_{m+1,n+1}$		$\leftarrow b_m$

Caso la tarjeta de post-optimización indique que se entra a fase Solución, entonces a ésta tarjeta le siguen las tarjetas de datos del modelo.

TARJETAS DE DATOS

TARJETAS ORDEN DE LA MATRIZ (Tarjeta única)

FORMATO: Entero (2I3)

CC 1-3: Número de filas (ajustado a la derecha)

CC 4-6: Número de columnas (ajustado a la derecha)

TARJETA ELEMENTOS DE LA MATRIZ (VARIAS)

A esta tarjeta le siguen las tarjetas de elementos de la matriz, una tarjeta por elemento.

FORMATO: Entero(3I3)

CC 1-3: Número que indica la fila

CC 4-6: Número que indica la columna

CC 7-9: Valor del elemento de la matriz

Observación:-No es necesario perforar los elementos ceros.

-Fin de este tipo de tarjeta, una tarjeta en blanco.

TARJETA DE FLUJO EXOGENO

FORMATO:Entero (20I3)

CC 1-3: Valor de b (1)

CC 4-6: Valor de b (2)

CC 7-9: ....

⋮ ⋮

TARJETA DE COSTOS

FORMATO:Entero (20I3)

CC 1-3: Valor de c (1)

CC 4-6: Valor de c (2)

CC 7-9: ....

⋮ ⋮

Observación:El problema inicial se considera con el siguente esquema

	Col_1	Col_2	......			Col_n	$Col\vec{b}$
fila $\vec{c}$	C_1	C_2	C_3	C_4	C_5	C_n	
fila 1	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{1n}	b_1
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
fila m	a_{m1}	a_{m2}	a_{m3}	a_{m4}	a_{m5}	a_{mn}	b_m

SALIDA - El sistema da los siguientes salidas en impresora: En fase solución, imprime los datos almacenados , costos, flujo exogeno y matriz de restricciones; Al final de la fase solución se imprimirá la solución óptima (si existe), función objetiva y rango de los costo, flujo exógeno y matriz de restricciones; caso contrario se imprimirá "IMPOSIBLE". En la fase post-optimal, se imprimá el cambio y la solución mejorada si existe, caso contrario "SOLUCIÓN ACTUAL TODAVIA OPTIMAL".

B I B L I O G R A F I A

|1| BALAS, EGON - "An Additive Algoritms for Solving Linear Programming with Zero-One Variables", Operations Research 13,(1965) 517-546.

|2| GEOFFRION, A.M. - "Integer Programming by Implicit Enumeration and Balas' Method", Siam Review , Vol. 9, nº 2, Abril 1967.

|3| BALINSKI, M.L. - "Integer Programming, Methods, Uses, Computations", Management Science, 12 (1965), pp. 235-313.

|4| BOWMAN, V. and F. GLOVER - "A Note on Zero-One Integer and Concave Programming", Operations Research 20(1), 182-189 (1972).

|5| WATTERS, L. - "Reduction of Polynomial Programming Problem to Zero-One Linear Programming Problem", Operations Research 15(6), 1171-1174 (1967).

|6| GLOVER, F. and R. WOOLSEY - "Futher Reduction of Zero-One Polynomial Programming Problems", Operations Research 21(1), 156-161 (1973).

|7| GLOVER F. - "Converting the Zero-One Polynomial Programming Problems to a Zero-One Linear Programming", Operations Research 22(1), 180-182, (1974).

|8| SALKIN HARVEY M. - "Integer Programming", Addison-Wesley Publishing Company (1975).

|9| JENSEN R. - "Sensitivity Analisis and Integer Linear Programming", Accounting Review 43(3), 425-446 (1968).

|10| ROODMAN G. - "Post-Optimal Analisis in Zero-One Programming by Implicit Enumeration", Amos Tuck School of Business Administration Dartmouth College, January, 1972.

|11| PIPER C. and ZOLTNERS A. - "Implicit Enumeration Based Algorithms for Post Optimizing Zero-One Programs", Naval Research Logistcs Quarterly Dez. (1975), Vol. 22, nº 4.

|12| KNUTH, D.E. - "The Art of Computer Programming Volume I / Fundamental Algorithms", Addison Wesley, Reading, Mass. 1969.

|13| LANG AND DOIG - "An Automatic Method for Solving Discrete Programming Problems", Econometrica 28(3), 497-520 (1960).

|14| DAY RICHARD - Operations Research" 1965.

```
   JOB PERIKITO  CLASS=2  USER=
    COS04001/823
   NAME=148 BEGIN
    COMPILE  CCC FORTRAN DATA
$ RESET SINGLE
FILE  3=IMPRE,UNIT=PRINTER
FILE  2=LECTU,UNIT=READER
FILE  4=DISK1,UNIT=DISK,AREA=500,RECORD=5,SAVE=2
FILE  8=DISK2,UNIT=DISK,AREA=101,RECORD=51,SAVE=2
      DIMENSION AA(30,30)
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
   50 READ(2,1)I,NUOPP
    1 FORMAT(2I2)
      IF(I-9)2,3,88
    2 CALL DATO
      NUOP=0
      CALL SOLUC
      GO TO 50
    3 READ(8=1)Z,M,N,IQ
      MX=M+1
      DO 41 L=2,MX
      K=L-1
      READ(8=L)(A(K,I),I=1,N),B(K)
   41 CONTINUE
      DO 42 I=1,N
   42 C(I)=A(1,I)
      DO 40 I=1,IQ
      READ(4=I)XPQ(I),VLPQ(I),GAPQ(I),CXPQ(I),NNPQ(I)
   40 CONTINUE
      CALL RANKA(AA)
      CALL RANKB
      CALL RANKC
      NUOP=1
      WRITE(3,47)
   47 FORMAT(1H1,'***** ANALISIS POST-OPTIMAL ***** ')
      DO 25 ITES=1,NUOPP
      READ(8=1)Z,M,N,IQ
      JH=0
```

```
      READ(2,4)J,K,KIT
    4 FORMAT(2I2,I3)
      IF(J-1)6,5,6
    5 IF(K-N)7,8,8
    7 FOL=C(K)-KIT
      C(K)=KIT
      IF(C(K)-W(K))9,10,10
   10 WRITE(3,11)Z,(XR(I),I=1,N)
   11 FORMAT(1X,'F.O.',F8.0,2X,'SOLUCION TODAVIA OPTIMAL',20I3)
      GO TO 25
    9 WRITE(3,28)J,K,KIT
   28 FORMAT(1X,'CAMBIO EN EL COSTO',//,' FILA',I2,//,' COLUMNA',I2,//
     1' NUEVO COSTO =',I3)
      LU=2
      L=1
      GO TO 20
    8 J=2
    6 IF(J-M)30,30,31
   31 WRITE(3,32)
   32 FORMAT(1X,'ERROR EN EL DATO')
      GO TO 88
   30 IF(K-N)33,34,34
   33 WRITE(3,35)J,K,KIT
   35 FORMAT(1X,'CAMBIO EN EL COEFICIENTE DE LA MATRIZ',//,' FILA',I2,//
     1 'COLUMNA ',I2,//,' NUEVO VALOR =',I3)
      LU=2
      AR=-1
      IF(A(J,K)-AA(J,K))36,36,20
   36 WRITE(3,11)Z,(XR(I),I=1,N)
      GO TO 25
   34 WRITE(3,29)J,K,KIT
   29 FORMAT(1X,'CAMBIO EN EL FLUJO EXOGENO',//,' FILA',I2,//,' COLUMNA'
     1 I2,//,' NUEVO VALOR =',I3)
      LU=0
      B(J)=KIT
      IF(B(J)-U(J))17,17,18
   17 WRITE(3,11)Z,(XR(I),I=1,N)
      GO TO 25
   18 L=J
   20 DO 12 IJ=1,13
      IF(ANPO(IJ)-L)12,15,12
```

```
   15 XPP=XPQ(IJ)
      VAM=VLPQ(IJ)
      CALL BINA(CL,VAM,N)
      CALL BINA(X,XPP,N)
      IF(LU)16,22,16
   16 IF(CL(K))22,22,21
   21 IF(X(K))12,12,19
   19 IF(AR+1)37,22,37
   37 CXPQ(IJ)=CXPQ(IJ)-FOL
   22 IF(CXPQ(IJ)-Z)14,12,12
   14 MX=M+1
      DO 60 L=2,MX
      K=L-1
   60 READ(8=L)(A(K,I),I=1,N),B(K)
      IF(LU-2)52,53,53
   52 B(J)=KIT
      GO TO 54
   53 IF(L-1)56,55,56
   55 C(K)=KIT
      GO TO 54
   56 A(J,K)=KIT
   54 CALL POST
      JH=JH+1
      WRITE(3,51) JH,(X(I),I=1,N)
   51 FORMAT(1H0,I2,'.- SOLUCION PARCIAL'  20I3)
      CALL SOLUC
   12 CONTINUE
   25 CONTINUE
   88 CALL EXIT
      END

      SUBROUTINE DATO
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NAPQ(50),P,J1(30),J0(30)
      COMMON Z,M,N,RUOP,IO,IS,ICV,ZZ
C*****LECTURA DEL ORDEN DE LA MATRIZ
    1 FORMAT(2I3)
      READ(2,1)M,N
      M=M+1
      DO 66 I=1,M
```

```
      DO 66 J=1,N
   66 A(I,J)=0
C*****LECTURA DE DATOS , ELEMENTO POR ELEMENTO
    3 READ(2,2)I,J,KIT
    2 FORMAT(3I3)
      IF(I)4,4,44
   44 I=I+1
      A(I,J)=KIT
      GO TO 3
C*****LECTURA DEL FLUJO EXOGENO
    4 READ(2,6)(B(I),I=2,M)
    6 FORMAT(20I3)
      DO 5 I=2,M
    5 Y(I)=B(I)
C*****LECTURA DE COSTOS
      READ(2,6)(C(J),J=1,N)
      DO 200 J=1,N
  200 A(1,J)=C(J)
      ICV=0
      ZZ=0
      DO 7 J=1,N
      IF(C(J))71,72,72
   71 C(J)=-C(J)
      ZZ=ZZ-C(J)
      ICV=ICV+1
      CV(ICV)=J
      DO 73 I=1,M
      A(I,J)=-A(I,J)
   73 Y(I)=A(I,J)+Y(I)
   72 CL(J)=0
      SS(J)=0
    7 X(J)=0
      WRITE(3,704)(C(I),I=1,N)
  704 FORMAT(1H1,'***** DATOS ALMACENADOS *****',///,2X,' VECTOR COSTOS'
     1 //,20I5)
      WRITE(3,705)
  705 FORMAT(//,2X,' MATRIZ DE RESTRICCIONES',5X,'  ...  ULTIMA COLUMNA,
     1 FLUJO EXOGENO ... '//)
      DO 703 I=2,M
  703 WRITE(3,702)(A(I,J),J=1,N),B(I)
  702 FORMAT(1H0,21I5)
```

```
      IS=0
      RETURN
      END

      SUBROUTINE SOLUC
      DIMENSION AA(30,30)
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      IQ=0
      IJ0=0
      IJ1=0
      P=0
      SKIP=1
      Z=9999
      B(1)=Z
      Y(1)=Z
   28 CX=0
      DO 8 J=1,N
    8 CX=CX+C(J)*X(J)
  288 II=0
C***  CALCULO DE T(I),RO(I), AMA(I)
  600 DO 101 I=1,M
      RO(I)=0
  103 DO 105 LL=1,N
      IF(CL(LL))105,107,105
  107 IF(A(I,LL))108,105,105
  108 RO(I)=RO(I)+A(I,LL)
  105 CONTINUE
  101 GAMA(I)=Y(I)-RO(I)
C***** SON TODOS LOS Y(I) POSITIVOS
      DO 9 I=1,M
      IF(Y(I))10,9,9
   10 II=II+1
      NH(II)=I
    9 CONTINUE
      IF(II)11,11,12
C***** SI CX Z ENTONCES Z=CX Y XR=X
   11 IF(NUOP-1)200,201,200
  200 IND=1
```

```
      CALL EMPAQ(IND,CX)
  201 IF(CX-Z)14,14,30
   14 Z=CX
      B(1)=Z
      Y(1)=Z
      P=P+1
      DO 15 J=1,N
   15 XR(J)=X(J)
      GO TO 30
C**** CALCULO DEL CONJUNTO T QUE ES  TODA VARIBLE LIBRE TAL QUE
C*****CX+C(J) Z Y A(I,J) 0 PARA ALGUN I TAL QUE Y(I) 0
   12 IF(SKIP)901,900,901
  901 IT=0
      SKIP=0
      DO 16 J=1,N
      IF(CL(J))16,29,16
   29 IF(CX+C(J)-Z)17,16,16
   17 DO 18 K=1,II
      I=NH(K)
      IF(-A(I,J))18,19,19
   19 IT=IT+1
      T(IT)=J
      K=II
   18 CONTINUE
   16 CONTINUE
      IF(IT)900,900,50
C*****SI  Y(I)+(SUMATORIA DE J PERTENECIENTE A T ) MAX (0,A(I,J))   0
C***** PARA ALGUN I TAL QUE Y(IN  0 ENTONCES VA PARA 30 SINO  DO 24
   50 DO 20 K=1,II
      I=NH(K)
      TEMP=Y(I)
      DO 21 L=1,IT
      J=T(L)
      IF(-A(I,J))21,21,23
   23 TEMP=TEMP-A(I,J)
   21 CONTINUE
      IF(TEMP)30,20,20
   20 CONTINUE
      MAX=-9999
C***** AUMENTAR AL CONJUNTO S UN J PERTENECIENTE A T TAL QUE SEA EL
C***** MAXIMO DE  SUMATORIA DE MIN (Y(I)+A(I,J),0) PARA TODO J DE T
```

```
      DO 24 L=1,IT
      J=T(L)
      TT(L)=0
      DO 25 I=1,M
      IF(Y(I)-A(I,J))26,25,25
   26 TT(L)=TT(L)+Y(I)-A(I,J)
   25 CONTINUE
      IF(MAX-TT(L))27,27,24
   27 MAX=TT(L)
      JJ=J
   24 CONTINUE
      IS=IS+1
      X(JJ)=1
      S(IS)=JJ
      CL(JJ)=1
      DO 36 I=2,M
   36 Y(I)=Y(I)-A(I,JJ)
      Y(1)=Y(1)+A(1,JJ)
      GO TO 28
  900 DO 106 I=2,M
      SKIP=1
      DO 109 LL=1,N
      IF(CL(LL))109,110,109
  110 RT=RO(I)+IABS(A(I,LL))
      IF(RT-Y(I))109,109,111
  111 CL(LL)=1
      IS=IS+1
      SS(IS)=2
      S(IS)=LL
      IF(A(I,LL))114,112,112
  112 IJ0=IJ0+1
      J0(IJ0)=LL
      IF(NUOP-1)204,288,204
  204 X(LL)=1-X(LL)
      IND=I
      CXX=CX+C(LL)
      GAMA(IND)=GAMA(IND)-A(IND,LL)
      CALL EMPAQ(IND,CX)
      X(LL)=1-X(LL)
      GO TO 288
  114 IJ1=IJ1+1
```

```
      J1(IJ1)=LL
      X(LL)=1
      IF(NUOP-1)203,122,203
  203 IND=I
      X(LL)=1-X(LL)
      CXX=CX-C(LL)
      GAMA(IND)=GAMA(IND)+A(IND,LL)
      CALL EMPAQ(IND,CXX)
      X(LL)=1-X(LL)
      GO TO 122
  109 CONTINUE
  106 CONTINUE
      GO TO 30
  122 DO 133 I=2,M
  133 Y(I)=Y(I)-A(I,LL)
      Y(1)=Y(1)+A(1,LL)
      B(1)=Y(1)
      GO TO 28
C*****          **********   RETORNO   **********
C**** LOCALIZACION DEL ELEMENTO MAS A LA DERECHA DE S EL CUAL NO ESTA
C****  MARCADO  SI NO EXISTE TERMINA , EN OTRO CASO  REEMPLAZAR EL
C****  ELEMENTO POR EL COMPLEMENTO  MARCANDOLO Y ABANDOÑAR LOS  OTROS
C****  ELEMENTOS DE LA DERECHA
   30 J=S(IS)
   33 IF(SS(IS))32,31,32
   32 CL(J)=0
      SS(IS)=0
      IF(X(J))98,99,98
   98 DO 38 I=2,M
   38 Y(I)=Y(I)+A(I,J)
      Y(1)=Y(1)-A(1,J)
      X(J)=1-X(J)
   99 IS=IS-1
      IF(IS)88,88,30
   31 SS(IS)=2
      X(J)=1-X(J)
      DO 37 I=2,M
   37 Y(I)=Y(I)+A(I,J)
      Y(1)=Y(1)-A(1,J)
      SKIP=0
      GO TO 28
```

```
 88 IF(ICV)79,79,89
 89 DO 91 J=1,ICV
    L=CV(J)
 91 XR(L)=1-XR(L)
 79 IF(P)81,80,81
 80 WRITE(3,82)
 82 FORMAT(1X,'SOLUCION IMPOSIBLE')
    IF(NUOP-1)801,800,801
800 RETURN
801 CALL EXIT
 81 IF(NUOP-1)206,205,206
205 WRITE(3,207)
207 FORMAT(1X,'NUEVA SOLUCION OPTIMA' //)
    GO TO 210
206 WRITE(3,208)
208 FORMAT(1X,'SOLUCION OPTIMA'//)
    Z=Z+ZZ
210 DO 219 J=1,N
219 WRITE(3,209)J,XR(J)
209 FORMAT(1HO,' VARIABLE X',I2,' IGUAL A',I2)
    WRITE(3,35)Z
 35 FORMAT(1HO,'VALOR DE LA FUNCION OBJETIVO =',F10.0)
    IF(NUOP-1)211,212,211
212 RETURN
211 WRITE(8=1)Z,M,N,IQ
    DO 230 K=1,M
    J=K+1
230 WRITE(8=J)(A(K,I),I=1,N) ,B(K)
    CALL RANKC
    CALL RANKB
    CALL RANKA(AA)
    RETURN
    END

    SUBROUTINE BINA(XXX,VALOR,NN)
    DIMENSION XXX(30)
    COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
    COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
    COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
    COMMON Z,M,N,NUOP,IQ,IS,IC/,ZZ
    MM=NN+1
```

```
      DO 2 I=1,NN
      L=VALOR/2
      RESTO=VALOR-L*2
      MM=MM-1
      XXX(MM)=RESTO
    2 VALOR=L
      RETURN
      END

      SUBROUTINE DECI(XXX,VALOR,NN)
      DIMENSION XXX(30)
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      VALOR=0
      DO 1 I=1,NN
    1 VALOR=VALOR+XXX(I)*2**(NN-I)
      RETURN
      END

      SUBROUTINE EMPAQ(IND,CX)
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      CALL DECI(X,XDEC,N)
      CALL DECI(CL,XCLD,N)
      IQ=IQ+1
      XPQ(IQ)=XDEC
      VLPQ(IQ)=XCLD
      GAPQ(IQ)=GAMA(IND)
      CXPQ(IQ)=CX
      NNPQ(IQ)=IND
      WRITE(4=IQ)XPQ(IQ),VLPQ(IQ),GAPQ(IQ),CXPQ(IQ),NNPQ(IQ)
      RETURN
      END


      SUBROUTINE RANKC
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
```

```
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPG(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      DO 1 I=1,N
      MIN=9999
      IF(XR(I))10,2,10
    2 DO 3 J=1,IQ
      IF(NNPQ(J)-1)3,8,3
    8 VAM=VLPQ(J)
      CALL BINA(CL,VAM,N)
      IF(CL(I))5,4,5
    4 FI=CXPQ(J)-Z+C(I)
      GO TO 6
    5 XPP=XPQ(J)
      CALL BINA(X,XPP,N)
      IF(X(I))9,3,9
    9 FI=CXPQ(J)-Z
    6 IF(FI-MIN)7,3,3
    7 MIN=FI
    3 CONTINUE
   10 W(I)=MIN
      W(I)=C(I)-W(I)
    1 CONTINUE
      WRITE(3,900)
  900 FORMAT(1H1,' RANGO DEL COSTO DONDE LA SOLUCION ACTUAL ES TODAVIA
     1OPTIMAL')
      DO 898 I=1,N
  898 WRITE(3,899)I,W(I)
  899 FORMAT(1H0,'  VARIABLE C',I2,' MAYOR QUE',I5)
      RETURN
      END

      SUBROUTINE RANKB
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPG(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      DO 5 NV=2,M
      MIN=9999
      DO 1 J=1,IQ
      IF(NNPQ(J)-NV)1,3,1
```

```
 3 IF(CXPQ(J)-Z)8,1,1
 8 GA=-1-GAPQ(J)
   IF(GA-MIN)4,1,1
 4 MIN=GA
 1 CONTINUE
   U(NV)=MIN
   U(NV)=B(NV)+U(NV)
 5 CONTINUE
   DO 14 I=2,M
   Y(I)=0
   DO 14 J=1,N
14 Y(I)=Y(I)-A(I,J)*XR(J)
   WRITE(3,12)
12 FORMAT(1H1,' RANGO DEL FLUJO EXOGENO DONDE LA SOLUCION ACTUAL ES
  1TODAVIA OPTIMAL')
   DO 7 NV=2,M
   NNVV=NV-1
 7 WRITE(3,6)NNVV,U(NV),Y(NV)
 6 FORMAT(1H0,' VARIABLE B',I2,' MENOR QUE',I5,' Y MAYOR QUE',I5)
   RETURN
   END

   SUBROUTINE POST
   COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
   COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
   COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPQ(50),P,J1(30),J0(30)
   COMMON Z,M,N,NUOP,IG,IS,ICV,ZZ
   DO 1 I=2,M
   DO 1 J=1,N
   B(I)=B(I)-A(I,J)*X(J)
   Y(I)=B(I)
 1 CONTINUE
   IS=0
   DO 5 J=1,N
   IF(CL(J)-1)5,6,5
 6 IS=IS+1
   S(IS)=J
 5 CONTINUE
   RETURN
   END
```

```
      SUBROUTINE RANKA(AA)
      DIMENSION AA(30,30)
      COMMON Y(20),S(70),X(30),SS(30),TT(30),T(30),XR(30),NH(20),CL(30)
      COMMON GAMA(30),RO(30),W(100),A(30,30),B(30),C(30),CV(30),U(30)
      COMMON XPQ(50),VLPQ(50),GAPQ(50),CXPQ(50),NNPG(50),P,J1(30),J0(30)
      COMMON Z,M,N,NUOP,IQ,IS,ICV,ZZ
      DO 10 NV=2,M
      DO 1 I=1,N
      MIN=9999
      DO 2 J=1,IQ
      IF(NNPG(J)-NV)2,3,2
    3 IF(CXPQ(J)-Z)12,2,2
   12 VAM=VLPQ(J)
      CALL BINA(CL,VAM,N)
      IF(CL(I))5,4,5
    4 IF(A(NV,I))6,7,7
    7 GA=-1-GAPQ(J)+A(NV,I)
      GO TO 8
    5 XPP=XPQ(J)
      CALL BINA(X,XPP,N)
      IF(X(I))2,2,6
    6 GA=-1-GAPQ(J)
    8 IF(GA-MIN)11,2,2
   11 MIN=GA
    2 CONTINUE
      AA(NV,I)=MIN
      AA(NV,I)=A(NV,I)-AA(NV,I)
    1 CONTINUE
   10 CONTINUE
      WRITE(3,16)
   16 FORMAT(1H1,' RANGO DE LA MATRIZ DE RESTRICCIONES DONDE LA SOLUCION
     1 ES TODAVIA OPTIMAL ')
      DO 15 I=2,M
   15 WRITE(3,14)(AA(I,J),J=1,N)
   14 FORMAT(1H0,20I6)
      RETURN
      END
 DATA LECTU
7 1
 3  8
 1  1 -2
```

```
  1   4   2
  1   5  -6
  1   6   1
  1   7  -1
  1   8   2
  2   1  -4
  2   2  11
  2   3 -11
  2   4  -7
  2   5   4
  2   6   3
  2   7  -5
  2   8   1
  3   2   1
  3   3   1
  3   4   1
  3   5  -1
  3   6  -2
  3   8   1

 -5 -6
  2   5   5   6   4   1   8   1
 9 2
 1 4  2
 3 9  1
99
  END JOB
 X
```

***** DATOS ALMACENADOS *****

VECTOR COSTOS

2	5	5	6	4	1	8	1

MATRIZ DE RESTRICCIONES ... ULTIMA COLUMNA, FLUJO EXOGENO ...

-2	0	0	2	-6	1	-1	2	-5
-4	11	-11	-7	4	3	-5	1	-6
0	1	1	1	-1	-2	0	1	0

SOLUCION OPTIMA

VARIABLE X 1 IGUAL A 0
VARIABLE X 2 IGUAL A 0
VARIABLE X 3 IGUAL A 1
VARIABLE X 4 IGUAL A 0
VARIABLE X 5 IGUAL A 1
VARIABLE X 6 IGUAL A 0
VARIABLE X 7 IGUAL A 0
VARIABLE X 8 IGUAL A 0

VALOR DE LA FUNCION OBJETIVO = 9.

RANGO DEL COSTO DONDE LA SOLUCION ACTUAL ES TODAVIA OPTIMAL

VARIABLE C 1 MAYOR QUE 0
VARIABLE C 2 MAYOR QUE 0
VARIABLE C 3 MAYOR QUE-9994
VARIABLE C 4 MAYOR QUE 3
VARIABLE C 5 MAYOR QUE-9995
VARIABLE C 6 MAYOR QUE 0
VARIABLE C 7 MAYOR QUE 0
VARIABLE C 8 MAYOR QUE 0

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
NÚCLEO DE COMPUTAÇÃO ELETRÔNICA

```
RANGO DEL FLUJO EXOGENO DONDE LA SOLUCION ACTUAL ES  TODAVIA OPTIMAL
VARIABLE B 1 MENOR QUE   -4 Y MAYOR QUE     6
VARIABLE B 2 MENOR QUE   -4 Y MAYOR QUE     7
VARIABLE B 3 MENOR QUE 9999 Y MAYOR QUE     0
```

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
NÚCLEO DE COMPUTAÇÃO ELETRÔNICA

```
RANGO DE LA MATRIZ DE RESTRICCIONES DONDE LA SOLUCION ES TODAVIA OPTIMAL

   -3    -1    -1    -1-10005    -1    -2    -1
   -9     2-10010    -9     2    -2-10004    -2
-9999 -9998 -9998 -9998-10000-10001 -9999 -9998
```

```
***** ANALISIS POST-OPTIMAL *****

CAMBIO EN EL COSTO

FILA 1

COLUMNA 4

NUEVO COSTO =  2

 1.- SOLUCION PARCIAL   1   0   0   1   1   0   0   0
NUEVA SOLUCION OPTIMA

 VARIABLE X 1 IGUAL A 1
 VARIABLE X 2 IGUAL A 0
 VARIABLE X 3 IGUAL A 0
 VARIABLE X 4 IGUAL A 1
 VARIABLE X 5 IGUAL A 1
 VARIABLE X 6 IGUAL A 0
 VARIABLE X 7 IGUAL A 0
 VARIABLE X 8 IGUAL A 0
VALOR DE LA FUNCION OBJETIVO =          8.
CAMBIO EN EL FLUJO EXOGENO

FILA 3

COLUMNA 9

NUEVO VALOR =  1

 1.- SOLUCION PARCIAL   0   0   0   0   1   0   0   0
SOLUCION IMPOSIBLE
```