Oswaldo Fadigas Fontes Tòrres

CONTRIBUIÇÃO AO PROBLEMA DA PROGRAMAÇÃO LINEAR COM VARIÁVEIS INTEIRAS

Tese apresentada à Douta Congregação da Escola Politécnica da Universidade de São Paulo para concurso de provimento efetivo da Cátedra n. 48 - "Planejamento da Produção".

São Paulo, 1967

À memória de
João Fontes Torres,
meu pai.

Agradecimentos

Ao Prof. Dr. Ruy Aguiar da Silva Leme, pelo constante apoio e incentivo, convidando-me para participar de sua equipe do Departamento de Engenharia de Produção.

Aos colegas do curso de pós-graduação "Técnicas de Pesquisa Operacional", de 1965, que souberam criar o ambiente de trabalho científico e debate, que deu origem a esta tese.

Aos Engs. Isu Fang e Nicolau Reinhard, do Centro de Computação Eletrônica, pelo auxílio na programação do algoritmo e testes no computador.

Ao Prof. Samsão Woiler pela ajuda na obtenção de algumas referências mais recentes.

Aos funcionários do Departamento de Engenharia de Produção, D. Maria Geny Ferracini, Francisco Décio Simões Bandini e Edgard de Freitas, pelo auxílio desinteressado e expontâneo na datilografia e impressão dêste trabalho.

ÍNDICE

I N T R O D U Ç Ã O

Nos problemas de Engenharia de Produção ocorre frequentemente que sòmente têm sentido as soluções numéricas expressas por números inteiros.

Ora, a maioria das técnicas utilizadas na solução dos modelos matemáticos supõem variáveis continuas, e conduzem, via de regra, a soluções não inteiras. Isto ocorre, por exemplo, com a programação linear.

A primeira idéia que surge é, naturalmente, usar valores inteiros obtidos por um arredondamento conveniente da solução não inteira. Embora isto seja aceitável em muitos exemplos práticos, quando os valores numéricos são grandes e, portanto, a fração desprezada ou aumentada é, percentualmente, muito pequena e da ordem de grandeza da imprecisão existente nos dados originais do problema, esta solução, mesmo sendo viável, pode ser bastante diferente da solução ótima inteira obtida por métodos exatos.

Sòmente êste fato bastaria para justificar a importância do problema da programação linear com variáveis inteiras na Engenharia de Produção. Mas, um estudo mais minucioso da questão nos revela a existência de inúmeros outros problemas de grande importância, e de difícil solução, que podem ser transformados num problema de programação linear em inteiros e, portanto, resolvidos, se tivermos um algoritmo eficiente para esta última.

Dantzig (1960) cita os exemplos seguintes:

a) dicotomias, isto é, restrições alternativas;

b) otimização sôbre pares de conjuntos de restrições;

c) problemas combinatórios com variáveis discretas;

d) problemas com função-objetivo não linear;

e) restrições condicionais;

f) mínimo global de uma função côncava;

g) construção de quadrados latinos;

h) problema do viajante comercial;

i) problemas com obrigações fixas.

Ilustraremos com os dois últimos citados a maneira de formular êstes problemas como programação linear inteira.

O viajante comercial precisa visitar n cidades i(i = 1, 2,... n), partindo de sua sede i = 0, passando em cada cidade uma única vez e voltando à origem. No caso mais geral êle deve voltar à sede exatamente m vêzes, inclusive a última volta, e não deve visitar mais que p cidades em cada viagem. O problema é achar um itinerário que torne mínima a distância total percorrida. Evidentemente $mp \geq n$.

Seja $x_{ij} = 1$ ou 0, conforme êle vá da cidade i para a cidade j ou não, e d_{ij} a distância entre as duas. Podemos formular o problema como o seguinte problema de programação linear inteira:

$$Z(\text{min.}) = \sum_{o}^{n} \sum_{o}^{n} d_{ij}\, x_{ij}$$

sujeito a:

$$\sum_{i=o}^{n} x_{ij} = 1 \qquad j = 1, 2 \ldots n$$

$$\sum_{j=o}^{n} x_{ij} = 1 \qquad i = 1, 2 \ldots n$$

$$U_i - U_j + p\, x_{ij} \leq p - 1 \qquad 1 \leq i \neq j \leq n$$

$0 \leq x_{ij} \leq 1$ $\quad x_{ij}$ inteiro, $\quad U_i > 0$, $U_j > 0$

A terceira restrição destina-se a evitar a volta a uma cidade já visitada, isto é, um circuito com k arcos que não passa pela origem. Com efeito, somando as desigualdades ao longo de um circuito que não passa por zero, as diferenças $U_i - U_j$ se cancelam e temos $pk \leq (p - 1)\, k$. o que é uma contradição. Para os circuitos que passam origem 0, se fixarmos, por exemplo, $U_i = t$ se a cidade i fôr a $t^{\underline{a}}$ a ser visitada (t = 1, 2, ... p), é claro que $U_i - U_j \leq p - 1$ para todo par i, j e portanto a restrição é satisfeita para todo $x_{ij} = 0$; se $x_{ij} = 1$, temos:

$U_i - U_j + p\, x_{ij} = t - (t + 1) + p = p - 1$, que também satisfaz a restrição.

O problema das obrigações fixas é encontrado frequentemente, quando temos uma grandeza fixa, não proporcional à variável, mas que não existe se a variável fôr nula. (custos fixos, tempo de preparação de máquinas etc.)

Admitindo que os custos variáveis sejam lineares teremos o problema:

$$Z\ (\min) = \sum_{j=1}^{n} C_{T_j}$$

onde $\quad C_{T_j} = C_{F_j} + C_{v_j} x_j \quad$ se $\quad x_j > 0$

$\quad\quad C_{T_j} = 0 \quad\quad\quad\quad\quad$ se $\quad x_j = 0$,

sujeito a restrições lineares.

Admitindo que exista um limite superior L_j para a variável x_j, o que em geral acontece nos problemas reais, podemos escrever C_{T_j} sob a forma:

$$C_{T_j} = S_j\, C_{F_j} + C_{V_j}\, j$$

com as restrições adicionais:

$$x_j \leqslant S_j\, L_j$$

$$0 \leqslant S_j \leqslant 1 \quad , \quad S_j \text{ inteiro}$$

o que transforma o problema num de programação linear inteira. A primeira restrição garante que $x_j = 0$ se $S_j = 0$; embora a recíproca não seja obrigatória, isto é, se $x_j = 0$, S_j não está obrigado a ser 0, como o problema tem por objetivo mínimo custo, S_j será 0 se $x_j = 0$ na solução.

Por êstes dois exemplos vemos como a introdução de variáveis de existência, inteiros 0 ou 1, permite abordar vários problemas de grande interêsse para a Engenharia de Produção.

Embora alguns problemas particulares já tivessem sido resolvidos anteriormente com métodos ad-hoc, é a partir de 1958 que aparecem os primeiros algoritmos gerais de solução, Gomory (1958). Desde então as publicações de Pesquisa Operacional têm apresentado inúmeros exemplos de aplicação dos quais citamos abaixo alguns:

- problemas de corte de um número inteiro de comprimentos padrões de uma peça de comprimento dado, com mínimo de perdas em retalhos em Eisemann (1957), Gilmore e Gomory (1961), Gilmore e Gomory (1965).
- tempo mínimo de busca em arquivos, sem duplicação das informações em Day (1965)

- escolha de investimentos indivisíveis, maximizando o rendimento total, com restrição de capital diponível em vários períodos em Weigartner (1963), Hansmann (1961), Petersen (1967)
- Programação de manutenção preventiva em Wagner e al (1964)
- Sincronisação de sinais de trânsito de forma a tornar máximo o tempo livre, tendo em vista que o "vermelho" e o "verde" ocorrem um número inteiro de ciclos entre si, em Little (1966)
- problemas de localização industrial, considerando os custos fixos, em Efroymson e Ray (1966) Manne (1964)
- problemas de roteiro de entrega, com restrição que cada cliente deve receber todo o seu pedido, em Balinsky e Quandt (1964) Dantzig e Ramser (1959), Clarke e Wright (1964)
- número de deputados para cada estado, em Burt e Harris (1963)
- balanceamento de linhas de montagem, em Klein (1963) e Bowman (1960)
- programação de carga de máquinas, em Wagner (1959) Bowman (1959), .. Manne (1960), Dantzig (1960)
- escolha de equipamento, em Peart e al (1961)
- tamanho ótimo de uma frota, em Esopo e Lifkowitz (1964)
- distribuição de recursos no planejamento pelos métodos de caminho crítico, em Moder e Phillips (1964), Assis (1964)
- cargas indivisíveis, (problemas da mochila) em Pandit (1962)
- tamanho e número ótimo de embalagens, para ter um custo mínimo total, em Wilson (1965)
- avaliação de propostas numa concorrência, em Beale (1963)

Entretanto, embora o assunto tenha despertado tanto interêsse, os resultados práticos não têm sido plenamente satisfatórios, pois

os algoritmos existentes não se mostraram à altura.

Em vários casos a convergência tem sido muito lenta, o que tem limitado muitas aplicações a problemas de pequeno porte. (ver, por exemplo as apreciações de Balinsky (1965)). É bem verdade que ultimamente têm surgido resultados mais animadores no que se refere aos problemas com variáveis do tipo 0 ou 1, como os de Balas (1965), Glover (1965), Driebeek (1966), Woiler (1967).

A presente tese pretende ser uma contribuição para a solução do problema, apresentando um novo método baseado nas soluções múltiplas obtidas quando introduzimos uma restrição adicional paralela ao hiper plano da função-objetivo.

No Capítulo I conceituamos o problema da programação linear com variáveis inteiras e fazemos uma apresentação resumida dos algoritmos mais representativos de cada método.

No Capítulo II apresentamos os fundamentos do método das soluções múltiplas e desenvolvemos um primeiro algoritmo, utilizando combinações convexas.

No Capítulo III desenvolvemos um segundo algoritmo que nos parece mais adequado para problemas de maior porte e para a utilização de computador eletrônico, baseado num processo de enumeração implícita das soluções inteiras viáveis.

No Capítulo IV são apresentados alguns aperfeiçoamentos que tornam mais rápida a convergência do algoritmo, e estendem o seu campo de aplicação.

Encerrando a tese, além de conclusões, e bibliografia atualizada até maio de 1967, há o apêndice I, onde apresentamos a solução de alguns problemas de pequeno porte, comparando com a obtida por outros métodos. O apêndice II contém o programa em linguagem FORTRAN II, que utilizamos no computador IBM-1620 do Centro de Computação Numérica da U.S.P. para testes preliminares do algoritmo II.

Esperando ter atingido o nosso objetivo, temos a honra de submeter esta tese à consideração da Douta Congregação da Escola Politécnica da Universidade de São Paulo.

C A P Í T U L O I

DEFINIÇÕES - MÉTODOS DE SOLUÇÃO

1.0 Programação linear geral

Supomos o leitor familiarizado com a programação linear geral, tal como é exposta, por exemplo, em Dantzig (1963), Gass (1964), Hadley (1962), Simonnard (1966) ou Fadigas Torres (1967).

Em sua forma mais geral trata-se de achar o máximo ou o mínimo de uma forma linear sujeita a restrições lineares com desigualdades $\leqslant$ ou $\geqslant$ ou igualdades estritas.

Utilizaremos neste trabalho, como forma padrão do problema:

$$Z\ (\max) = \sum_{j=1}^{n} C_j\, x_j \qquad (1.1)$$

sujeito a

$$x_j \geqslant 0 \qquad j = 1,\ 2,\ \ldots\ n \qquad (1.2)$$

$$\sum_{j=1}^{n} a_{ij}\, x_j \leqslant d_i \qquad i = 1,\ 2,\ \ldots\ m \qquad (1.3)$$

A forma canônica será definida como sendo:

$$Z\ (\max) = \sum_{j=1}^{n'} C_j\, x_j \qquad (1.4)$$

sujeito a

$$x_j \geqslant 0 \qquad j = 1, 2, \ldots n' \qquad (1.5)$$

$$\sum_{j=1}^{n'} a_{ij}\, x_j = d_i \qquad i = 1, 2, \ldots n' \qquad (1.6)$$

$$d_i \geqslant 0 \qquad (1.7)$$

Para passar da forma geral para canônica ou padrão, temos:

a) $\min Z = - \max (-Z)$

b) se x_k não tem restrição de sinal, fazemos:

$$x_k = x_s - x_t \text{ , onde } x_s \geqslant 0, \quad x_t \geqslant 0$$

c) para transformar desigualdades $\geqslant$ em $\leqslant$, basta multiplicar por -1:

$$\sum_{j=1}^{n} a_{rj}\, x_j \geqslant d_r \quad \text{ou} \quad \sum_{j=1}^{n} (-a_{rj})\, x_j \leqslant - d_r$$

d) As igualdades são decompostas em duas desigualdades opostas e aplica-se a regra c:

$$\sum_{j=1}^{n} a_{kj}\, x_j = d_k \quad \text{é substituido pelas restrições}$$

$$\sum_{j=1}^{n} a_{kj}\, x_j \geqslant d_k \quad \text{e} \quad \sum_{j=1}^{n} (-a_{kj})\, x_j \leqslant - d_k$$

e) As desigualdades são transformadas em igualdades pela introdução de variáveis residuais $x_{ri} \geqslant 0$

Assim:

$$\sum_{j=1}^{n} a_{ij}\, x_j \leqslant d_i \quad \text{fica} \quad \sum_{j=1}^{n} a_{ij}\, x_j + x_{ri} = d_i$$

$$\sum_{j=1}^{n} a_{kj}\, x_j \geqslant d_k \quad \text{fica} \quad \sum_{j=1}^{n} a_{ij}\, x_j - x_{rk} = d_k$$

Notar que estas transformações podem introduzir novas variáveis ou novas restrições, de modo que o número de restrições e de variáveis pode não ser o mesmo nas diversas formas do mesmo problema.

As formas apresentadas o foram em notação algébrica.

Quando fôr conveniente podemos utilizar a notação matricial ou a notação vetorial.

Fazendo: $C = \begin{bmatrix} c_1, & c_2 \ldots\ldots & c_n \end{bmatrix}$ $\qquad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

$$A = \begin{bmatrix} a_{11} \cdots\cdots\cdots & a_{1n} \\ \cdots\cdots\cdots\cdots\cdots & \\ \cdots\cdots a_{ij} \cdots\cdots\cdots & \\ a_{m1} \cdots\cdots\cdots & a_{mn} \end{bmatrix} \quad e \quad D = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix}$$

podemos escrever a forma padrão, em notação matricial:

$$Z\ (\text{máx}) = C.X \tag{1.8}$$

sujeito a:

$$X \geqslant 0 \tag{1.9}$$

$$AX \leqslant D \tag{1.10}$$

e a forma canônica fica:

$$Z\ (\text{max}) = C.X \tag{1.11}$$

sujeito a:

$$X \geqslant 0 \tag{1.12}$$

$$AX = D \quad \text{com} \quad D \geqslant 0 \tag{1.13}$$

Como já foi dito, a matriz A da forma padrão é, em geral, diferente da matriz A da forma canônica, e o mesmo acontece com os vetores C, X e D.

Fazendo $A = [P_1, P_2 \ldots P_n]$, onde $P_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{bmatrix}$

é o vetor coluna j de A, e pondo $P_o = D$, para uniformidade de notação, podemos escrever a forma canônica em notação vetorial:

$$Z\ (max) = \sum_{1}^{n} c_j\ x_j$$

sujeito à:

$$x_j \geq 0 \qquad j = 1, 2, \ldots n \tag{1.14}$$

$$x_1 P_1 + x_2 P_2 + \ldots + x_j P_j + \ldots + x_n P_n = P_o \tag{1.15}$$

Seja o problema na forma canônica:

$$Z\ (max) = C.X$$

sujeito à:

$$X \geq 0$$

$$AX = D \qquad A \text{ é matriz } (m \times n)$$

Vamos supor que o sistema de equações AX = D não tem equações redundantes, isto é, a característica da matriz A é m: r(A)= m.

Suporemos também que m < n, isto é, o sistema possue infinitas soluções.

Definimos:

Solução Viável ou Programa - é todo o conjunto de valores x ou melhor, é um vetor X que satisfaz a tôdas as restrições do problema, inclusive a do não negatividade.

Solução não viável ou pseudo - Programa - é um vetor X, que satisfaz $AX = D$, mas não satisfaz à condição $X \geqslant 0$, isto é, tem pelo menos uma componente negativa.

Base do sistema linear $AX = D$ é todo conjunto de m vetores-coluna P_j de A, linearmente independentes. Êstes vetores determinam uma sub-matriz quadrada B, de ordem m, de A, que é não singular.

As m variáveis associadas à base B são chamadas variáveis básicas e constituem um sub-vetor X^B de X. As $n - m$ variáveis restantes são variáveis secundárias e constituem o sub-vetor X^R de X, isto é,

$$X = \begin{bmatrix} X^B \\ X^R \end{bmatrix}$$

Anulando as $n - m$ variáveis secundárias em relação a base B, isto é, fazendo $X^R = 0$, obtemos um sistema de m equações e m incógnitas que admite uma solução única $X^B = B^{-1}D$. Uma solução básica é a solução

$$X = \begin{bmatrix} X^B \\ X^R \end{bmatrix}$$

tal que $X^R = 0$, isto é, que tem no máximo m valores x_j diferentes de zero e os restantes nulos. Frequentemente, ao falar de solução básica nos referimos apenas a X^B, ficando implícito que as $n - m$ variáveis restantes da solução completa, são nulas.

Uma solução viável básica ou programa básico é uma solução básica que satisfaz também a condição $X \geq 0$.

Solução ótima ou Programa ótimo é a solução viável X que torna máxima (ou mínima se fôr o caso) a função objetivo do problema e que tem tôdas as componentes finitas.

1.0.1 - Teorema fundamental - Método do Simplex

Dantzig demonstrou o teorema fundamental da programação linear, que é o seguinte:

Dado um problema de programação linear, sob a forma canônica,

a) se êle admite uma solução viável finita, existe pelo menos uma solução básica viável;

b) Se existe uma solução ótima, existe pelo menos uma solução básica viável que é ótima.

Indicou também o método de solução, denominado método do Simplex, que supomos conhecido do leitor. Recordaremos aqui que, no algoritmo clássico do método do Simplex a apresentação dos cálculos costuma ser feita numa tabela, com (m + 1) linhas e (n + 1) colunas. As colunas são constituidas pelos n vetores P_j e o vetor da solução básica X^B, ao qual dá-se o índice zero. As linhas correspondem aos vetores da base, mais uma linha onde colocam-se os valores de Z e de $(Z_j - c_j)$. Cada iteração corresponde à troca de uma linha (vetor) na base.

Z_j é definido por $Z_j = \sum_{i=1}^{m} c_i \, a_{ij}$.

O vetor P_e que deve entrar na base é determinado pela condição:

$$|Z_e - C_e| = \max|Z_j - C_j|, \text{ com } Z_j - C_j < 0$$

O vetor P_s que sai da base é determinado pela condição:

$$\frac{x_s}{a_{se}} = \min_{a_{ie} > 0} \frac{x_i}{a_{ie}}$$

Na solução ótima temos $Z_j - C_j \geq 0$ para todo e qualquer vetor P_j. Se tivermos $Z_j - C_j = 0$ para P_j não pertencente à base, então existem soluções básicas ótimas alternativas, obtidas introduzindo êstes vetores na base. Sabemos também que tôda combinação linear convexa destas soluções básicas ótimas também será uma solução ótima (não básica) do problema

Em nosso trabalho utilizaremos também o algoritmo dual-simplex de Lemke, que parte de uma solução básica não viável ($x_i < 0$), mas satisfazendo ao critério de otimidade $Z_j - C_j \geq 0$ e vai mudando a base até chegar a uma solução viável que será ótima, pois mantemos sempre $Z_j - C_j \geq 0$.

O vetor P_s a ser substituido na base é aquêle correspondente a $x_s = \min_i x_i$, com $x_i < 0$. O vetor P_e que deve entrar é dado pelo critério:

$$\frac{Z_e - C_e}{a_{se}} = \min \frac{Z_j - C_j}{|a_{sj}|}, \qquad a_{sj} < 0$$

1.0.2 - Interpretação Geométrica

A um problema de programação linear podemos associar dois espaços vetoriais. O primeiro é um espaço R^n de dimensão n, que contém, em particular os vetores X das soluções e, por esta razão, denomina-se espaço das soluções.

O segundo é um espaço R^m, de dimensão m, que contém, em particular, os vetores P_j da matriz A e denomina-se espaço das restrições.

No espaço das soluções temos uma coordenada para cada incógnita (sem considerar as residuais). Cada restrição dá origem um hiperplano e o conjunto das soluções viáveis é um poliedro convexo.

As soluções básicas corresponden aos vértices. A função objetivo corresponde a uma família hiperplanos paralelos. Interpretado geomètricamente o método do simplex consiste em caminhar de um vértice a outro adjacente no sentido de aumentar o valor de Z, até alcançar um vértice que tangencia o hiperplano máximo de Z.

Se a tangência se der segundo uma ou mais arestas teremos várias soluções básicas ótimas alternativas, correspondendo aos extremos destas arestas; qualquer ponto combinação linear convexa dêstes extremos será também uma solução ótima do problema.

No espaço das restrições cada coordenada é uma restrição e o conjunto das combinações lineares positivas dos vetores P_j constitui um cone poliédrico convexo; cada base corresponde a uma face do cone

Neste trabalho utilizaremos unicamente a interpretação do problema no espaço das soluções.

1.1 - Programação linear em inteiros

Para facilitar a linguagem, usaremos neste trabalho as expressões: ponto inteiro, vetor inteiro, matriz inteira, para significar o fato que as suas coordenadas ou componentes são expressas por números inteiros.

Definimos como programação linear em inteiros a programação linear geral com restrição adicional que uma ou mais variáveis x_j devem ser inteiras.

Distinguiremos dois tipos de problemas de programçaõa linear em inteiros:

a) O problema parcial ou misto, em que apenas parte das variáveis da forma padrão são restritas a serem inteiras;

b) o problema total em que tôdas as variáveis da forma padrão têm restrição de inteireza.

As variáveis residuais, introduzidas na forma canônica, não têm esta restrição; evidentemente se a matriz A e o vetor P_o forem inteiros, as variáveis residuais, no problema total, também resultarão inteiras.

1.2 - Métodos de Solução

Podemos inicialmente classificar os métodos que têm sido propostos para resolver o problema de programação linear em inteiros em três grandes grupos:

a) enumeração implícita;
b) corte
c) penetração.

Os dois últimos são baseados no método do simplex.

1.2.1 - Métodos de enumeração implícita

É natural considerar a enumeração de tôdas as soluções viáveis de um problema em inteiros (ou discreto, no caso geral) quando há um número finito de possibilidades. Entretanto êste número finito pode ser muito grande e daí a necessidade de imaginar esquemas de enumeração implícita, que permitam evitar a inspeção de soluções òbvia - mente não ótimas.

Para conceituar melhor a idéia, consideraremos um problema com variáveis bivalentes, 0 ou 1. O conjunto das soluções possíveis está em correspondência bi-unívoca com o conjunto dos vértices terminais de uma árvore definida da seguinte forma. Ao primeiro nível corresponderá a variável x_1; o ramo (arco) da esquerda corresponde ao valor $x_1 = 1$ e o da direita a $x_1 = 0$.

No segundo nível se encontram portanto dois vértices, um extremidade do arco $x_1 = 1$ e o outro extremidade do arco $x_1 = 0$. A êste nível associamos a variável x_2; os ramos da esquerda que saem de cada um dos vértices que corresponden a x_1 têm o valor $x_2 = 1$; os da direita $x_2 = 0$. No terceiro nível temos, portanto, quatro vértices, correspondendo respectivamente da esquerda para a direita aos valores (1,1), (1,0), (0,1) e (0,0) de (x_1, x_2).

A êste terceiro nível associamos x_3 e assim por diante. A idéia fundamental dos métodos de enumeração implícita é explorar a árvore, partindo de A.(Fig. 1.1).

Chegando em B, calculamos o valor da função objetivo (correspondente a 1.1.1), Z_1. Prosseguindo na exploração, chegamos em C e calculamos o novo valor Z_2. Se $Z_2 < Z_1$, conservamos o valor Z_1 (estamos procurando o máximo) e a solução correspondente e passamos para D.

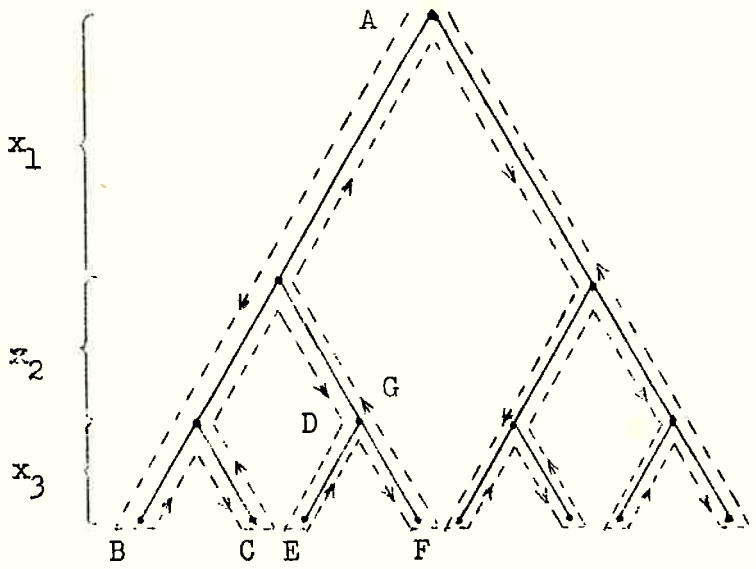


Fig. 1.1

Quando chegamos em D, já temos fixado $x_1 = 1$ e $x_2 = 0$ para todo o trajeto D E F G. Se tivermos maneira de estabelecer um limite superior para o valor de Z a partir dos valores já fixados para as incógnitas, isto é, se fôr possível afirmar que, se $x_1 = 1$ e $x_2 = 0$, Z será no máximo $m < Z_1$, então é inutil efetuar o percurso D E F G e podemos passar diretamente de D a G, sem explorar a parte da árvore que sai do vértice DG.

Está claro que o raciocínio vale para o caso de cada variável x poder tomar vários valores discretos.

O método é bastante geral e permite resolver problemas não lineares com variáveis discretas. A rigor o próprio método do simplex pode ser considerado como de enumeração implícita, pois sabemos pelo teorema fundamental que a solução ótima é básica, o que permite enumerar tôdas as soluções básicas., que são em número finito. O simplex passa de uma básica para outra que aumenta o valor de Z, eliminando de consideração aquelas que teriam valor Z menor.

A programação dinâmica de R. Bellman é outro exemplo bastante sugestivo de enumeração implícita.

Os métodos de enumeração implícita têm recebido ultimamente grande atenção na literatura técnica, principalmente os referentes ao caso de variáveis bi-valentes, isto é, que só podem tomar os valores 0 ou 1. Citaremos entre os mais importantes os trabalhos de Balas (1965), Roy e al.(1965), Little e al. (1963), Lawler e Wood (1966), Woiler (1967).

Êstes métodos podem ser generalizados para o caso de variáveis inteiras quaisquer, desde que se conheça um limite superior de seu valor, utilizando a representação no sistema binário, mas êste artifício pode levar a um número muito grande de variáveis.

Com efeito $x \leq 2^k - 1$ pode ser expresso pela soma de k variáveis bivalentes $x_1, x_2, \ldots x_k$, por meio da relação

$$x = x_1 + 2x_2 + \ldots + 2^{k-1}x_k.$$

1.2.2 - Métodos de Corte

O grande sucesso do método do simplex na programação linear geral levou, naturalmente, a tentar usá-lo na solução dos problemas com variáveis inteiras. Consideremos, para raciocinar, o problema total.

Quando impomos a restrição de inteireza, o poliedro das soluções viáveis degenera, reduzindo-se aos pontos de um reticulado e não é mais convexo. O problema, a rigor, deixa de ser programação linear.

Porém se acrescentarmos restrições ligando os pontos "externos" do reticulado e considerarmos um poliedro convexo com os novos limites, teremos um novo problema de programação linear geral, em que:

a) a nova região viável contém todos os pontos inteiros da região anterior;
b) cada ponto extremo da região é um ponto inteiro.

Como a solução ótima corresponde a um ponto extremo, a solução do novo problema será necessàriamente inteira. Alguns problemas de programação linear já apresentam de início esta condição de pontos extremos inteiros. Dantzig mostrou em 1949 que os problemas tipo transporte têm sempre solução inteira, se as disponibilidades e necessidades iniciais forem inteiras. Balinsky (1966) demonstra o seguinte teorema, devido a Hoffman e Kruskal: "um problema de programação linear tem todos os pontos extremos inteiros se, e àpenas se, a matriz A gozar da propriedade unimodular", isto é, se todos os determinantes menores de A tiverem valor 0, + 1 ou - 1.

Geralmente a determinação da região viável inteira é feita por aproximação sucessiva, introduzindo uma a uma as novas restri

ções denominadas "cortes".

Dos algoritmos baseados em cortes os mais conhecidos são devidos a Gomory. No seu primeiro algoritmo, das formas inteiras, para o problema total, Gomory (1958) resolve inicialmente o problema sem levar em conta a exigência de solução inteira. Se a solução fôr inteira, o problema está terminado.

Se algum x_i não é inteiro vamos introduzir uma nova restrição (corte) que obrigue x_i a ser inteiro. Podemos escrever diretamente da linha i da tabela da solução ótima do simplex:

$$x_i = a_{io} + \sum_{1}^{n} a_{ij} (-x_j) \geqslant 0 \qquad (1.16)$$

Designando por $[a]$ o maior inteiro contido em a, isto é:

$$a = [a] + r \qquad 0 \leqslant r < 1 \qquad (1.17)$$

observamos que r é a parte fracionária de a se êste fôr positivo, e o seu complemento para 1 se a fôr negativo. Como x_i deve ser inteiro devemos ter, de (1.16), que:

$$x_i = [a_{io}] + r_{io} + \Sigma [a_{ij}] (-x_j) + \Sigma r_{ij}(-x_j) \geqslant 0 \qquad (1.18)$$

ou simplificando

$$\sum_{1}^{n} r_{ij} x_j = r_{io} + E \qquad E = \text{nº inteiro.}$$

Como $\sum_{1}^{n} r_{ij} x_j \geqslant 0$, o inteiro E tem de ser não negativo e portanto

$$\Sigma r_{ij} x_j \geqslant r_{io} \qquad (1.19)$$

que é a restrição adicional do algoritmo I de Gomory. Devido a problemas de aproximação do cálculo numérico Gomory (1963) introduziu o algoritmo II, todo inteiro, no qual todos os pivôs de transformação da tabela são unitários, conservando desta forma a matriz A sempre inteira. Dividindo (1.16) por $\lambda > 0$, temos, introduzindo o maior inteiro, e transpondo,

$$\frac{x_i}{\lambda} + \frac{1}{\lambda} \sum_{1}^{n} r_{ij}\, x_j = \left\{ \left[\frac{a_{io}}{\lambda} \right] + \sum_{1}^{n} \left[\frac{a_{ij}}{\lambda} \right] (-x_j) \right\} + \frac{r_{io}}{\lambda} \geqslant 0 \qquad (1.20)$$

Como $0 \leqslant \frac{r_{io}}{\lambda} < 1$ e o têrmo entre colchetes é inteiro, segue-se que êste último é um inteiro positivo, isto é,

$$\left[\frac{a_{io}}{\lambda} \right] + \sum_{1}^{n} \left[\frac{a_{ij}}{\lambda} \right] (-x_j) \geqslant 0 \qquad (1.21)$$

que é a restrição usada no algoritmo II de Gomory.

A constante λ é fixada de forma que o pivô seja sempre 1.

Para o problema parcial ou misto, Gomory (1960) define o corte da seguinte maneira: seja I o conjunto dos índices das variáveis inteiras. Então, para x_i, com $i \in I$, temos de (1.18), decompondo em duas partes os somatórios:

$$\sum_{j \in I} r_{ij}\, x_j + \sum_{j \notin I} a_{ij}\, x_j = \left\{ \left[a_{io} \right] + \sum_{j \in I} \left[a_{ij} \right] (-x_j) - x_i \right\} + r_{io} \qquad (1.22)$$

No lado esquerdo, o primeiro somatório é sempre positivo, mas o segundo pode ser positivo ou negativo. Se todo o têrmo da esquerda fôr positivo, então a expressão entre colchetes é um inteiro positivo ou nulo, isto é, neste caso

$$\sum_{j \in I} r_{ij}\, x_j + \sum_{j \notin I} a_{ij}\, x_j \geqslant r_{io}$$

e, "a fortiori", considerando apenas os têrmos positivos do segundo somatório:

$$\sum_{j\in I} r_{ij}\, x_j + \sum_{\substack{j\notin I \\ a_{ij}\geq 0}} a_{ij}\, x_j \geq r_{io} \qquad (1.23)$$

Se a expressão (1.22) fôr negativa, como $r_{io} \geq 0$ o têrmo entre colchetes é um inteiro menor que -1 e temos:

$$\sum_{j\in I} r_{ij}\, x_j + \sum_{j\notin I} a_{ij}\, x_j \leq -1 + r_{io}$$

e "a fortiori"

$$\sum_{\substack{j\notin I \\ a_{ij}<0}} a_{ij}\, x_j \leq -1 + r_{io}$$

ou

$$\sum_{\substack{j\notin I \\ a_{ij}<0}} (-a_{ij})\left(\frac{r_{io}}{1-r_{io}}\right) x_j \geq r_{io} \qquad (1.24)$$

Como todos os coeficientes de (1.23) e (1.24) são positivos temos:

$$-r_{io} + \sum_{j\notin j} (-r_{ij})(-x_j) + \sum_{\substack{j\notin j \\ a_{ij}\geq 0}} (-a_{ij})(-x_j) + \sum_{\substack{j\notin j \\ a_{ij}<0}} \left(\frac{r_{io}\, a_{ij}}{1-r_{io}}\right)(-x_j) \geq 0 \qquad (1.25)$$

que é a restrição usada no algoritmo III de Gomory, para problemas mistos.

Gomory demonstrou que a convergência de seus algoritmos dá-se num número finito de iterações, sendo que o algoritmo III (problema misto) só tem convergência garantida se a função-objetivo Z ou x_o fôr inteira.

No apêndice I apresentamos um exemplo de Balinsky (1965), resolvido pelos diversos algoritmos, para comparação com a solução pelo algoritmo proposto nesta tese.

Martin (1963) propôs alteração na maneira de efetuar os cortes no algoritmo I. Glover (1964,1965) apresenta outras variantes de algoritmos de corte.

1.2.3 - Métodos de Penetração

Nos métodos de penetração utilizamos planos de corte baseados na função-objetivo. O primeiro algoritmo dêste tipo foi proposto por Land e Doig (1960). Na realidade o seu método é híbrido, pois a solução é obtida por enumeração implícita. O algoritmo é o mesmo para o problema misto ou para o problema total.

Inicialmente resolve-se o problema sem restrição de inteireza. Se a solução não é inteira o valor γ^o da função é um limite superior da solução: $\max Z \leq \gamma^o$.

Uma variável x_k que deve ser inteira, mas não o é na solução corrente, deve ser tornada inteira e, portanto, ou diminuida para $\lfloor x_k \rfloor$ ou aumentada para $\lfloor x_k \rfloor + 1$ que são os pontos inteiros vizinhos. Realmente Land e Doig determinam todo o intervalo de valores inteiros possíveis, resolvendo os dois problemas de programação linear: x_k (max) e x_k (min), com as restrições do problema original.

Fixando então x_k num valor inteiro possível, resolve-se o problema original (que agora tem uma variável a menos) determinando $\max Z = \gamma^1$ que será um novo limite superior para a função-objetivo, e assim por diante.

A enumeração das combinações possíveis é feita por meio de uma árvore, com a raiz (vértice 0) correspondendo à solução X^o, Y^o e os ramos (arcos) (0,i) correspondendo à solução X^i, Y^i, com x_k fixado num valor x_k^i inteiro, aplicando-se o mesmo procedimento sempre que as variáveis com restrição de inteireza não forem inteiras.

Nesta árvore um caminho de 0 a qualquer vértice define uma sequência de valores inteiros fixados para certas variáveis, com restrição de inteireza. Cada vértice terminal ou corresponde a uma solução inteira viável ou a uma sequência que não é solução viável. O vértice viável com max. Z é a solução ótima, pois tôdas as soluções viáveis foram enumeradas.

Land e Doig investigam apenas um sub-conjunto destas soluções, desenvolvendo apenas a parte da árvore que contém a solução ótima baseados em que, cada vértice deve ter um valor de Z não maior que o de seu predecessor, pois a ramificação significa impôr uma restrição adicional (nova variável fixada), e, num vértice X^i, se x_k^i - não é inteiro, o valor que produz menor decréscimo de Z é ou $\lceil x_k^i \rceil$ ou $\lceil x_k^i \rceil + 1$.

Segundo informa Balinsky (1966), até 1964 êste algoritmo não havia sido testado em computadores de grande porte, a experiência sôbre seu uso restringindo-se a problemas resolvidos manualmente, a principal dificuldade parecendo ser a grande exigência de espaço na memória.

Harris (1964) e Thompson (1964) apresentam outros algoritmos de penetração.

No apêndice I fazemos uma comparação entre êstes algoritmos e o proposto nesta tese.

C A P Í T U L O II

O MÉTODO DAS SOLUÇÕES MÚLTIPLAS. ALGORITMO I

2.0 - Considerações Geométricas

Sabemos que no espaço das soluções de dimensão n, as restrições formam um conjunto convexo, ou mais particularmente, um poliedro convexo.

Cada valor da função objetivo está associado a um hiperplano, que em geral corta o poliedro convexo e, no caso especial do valor ótimo, é tangente.

A interseção do hiperplano com o poliedro convexo é um outro conjunto convexo de dimensão (n - 1) no máximo. A figura 2.1 representa esta interseção no caso de n = 3.

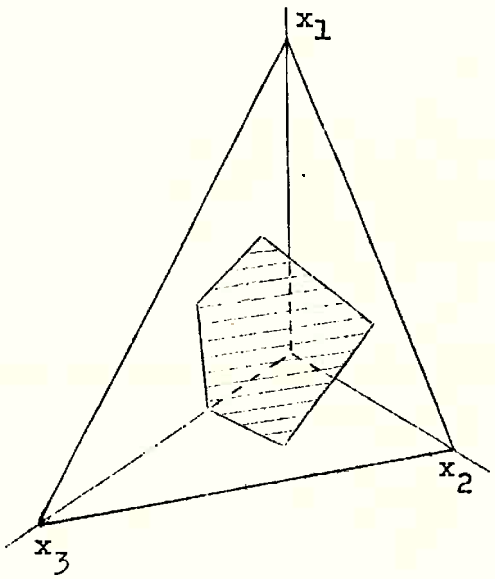


Fig. 2.1

Os pontos x_1, x_2 e x_3 são os interceptos do hiperplano com os três eixos e a área hachurada representa o conjunto convexo limitado pelas restrições do problema e o hiperplano.

Neste poliedro convexo de dimensão (n - 1) haverá um valor máximo e um valor mínimo para cada coordenada, consistente com as restrições do problema original, como ilustra a fig. 2.2.

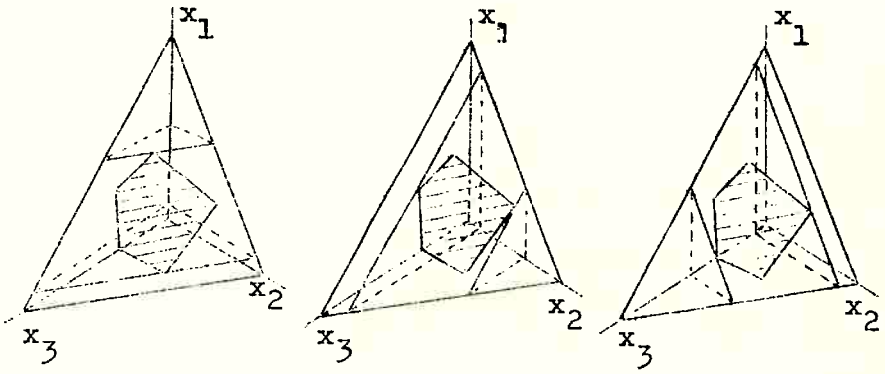


Fig. 2.2

Se o hiperplano é aquêle associado com a solução ótima do problema sem restrição de inteireza, êste poliedro convexo reduz-se, normalmente a um ponto único (a não ser que o problema original tenha soluções ótimas alternativas), mas mesmo assim podemos afirmar que existem um valor máximo e mínimo (que podem coincidir) para cada variável x_j.

Conhecendo, para um dado valor de Z, êstes valores máximo e mínimo, podemos determinar se há possibilidade de um valor inteiro, para cada variável, nesta posição do hiperplano.

Se não existir pelo menos um valor inteiro possível para cada variável x_j, então podemos afirmar imediatamente que não há solução inteira para êste valor de Z. Infelizmente a recíproca não é verdadeira, pois a interseção destas coordenadas inteiras pode não estar sôbre o hiperplano, pois, realmente, estamos pesquisando os inteiros dentro de um hiperparalelepípedo, limitado pelos hiperplanos paraleloes que passam pelo máximo e mínimo de cada variável.

Até êste ponto seguimos de perto Land e Doig (1960). Agora porém, em vez de fixar valores inteiros para algumas variáveis e resolver os problemas auxiliares $\max x_j$ e $\min x_j$, como fazem os autores citados, vamos proceder de modo diverso.

Consideremos inicialmente o problema total, isto é, com tôdas as variáveis inteiras. Podemos sempre impor, sem perda de generalidade a condição que todos os coeficientes c_j de $Z = \sum_1^n c_j x_j$ sejam inteiros e primos entre si. Neste caso as soluções inteiras darão valores também inteiros a Z, de modo que, se Z não fôr inteiro, podemos logo afirmar que não há solução inteira para esta posição do hiperplano.

Partindo, portanto, da posição $Z = \gamma^o$ do problema sem restrição de inteireza, vamos penetrando progressivamente o hiperplano da função objetivo, até atingir o primeiro valor inteiro $Z = \gamma^1 < \gamma^o$. Obteremos então uma interseção que é um poliedro convexo com (n - m) pontos extremos, todos êles dando a função objetivo o mesmo valor $Z = \gamma^1$, visto que estão sôbre o hiperplano.

Ora, êstes pontos extremos podem ser facilmente obtidos na tabela do simplex original com a restrição adicional $\sum_1^n c_j x_j \leqslant \gamma^1$, pois cada um dêles é uma das (n - m) soluções básicas ótimas do novo problema.

Destas (n - m) soluções podemos obter diretamente max x_j e min x_j e verificar se pode haver algum ponto inteiro incluido. Se não houver, fazemos nova penetração até o próximo valor inteiro $Z=\gamma^2=\gamma^1-1$ e repetimos o procedimento, até encontrar um ponto inteiro possível.

Para verificar se êste é realmente uma solução do problema basta lembrar que, havendo múltiplas soluções básicas ótimas, qualquer combinação linear convexa é também uma solução ótima. Necessitamos apenas pesquisar se existe uma combinação linear convexa que corresponda ao ponto inteiro.

Ocorre inclusive que algum dos pontos extremos pode já ser inteiro e, portanto, uma solução ótima do problema.

Se não existir combinação linear convexa inteira fazemos nova penetração até o próximo valor inteiro de Z, e repetimos o procedimento.

Como sòmente penetramos até um valor $Z = \gamma^k$ depois de ter verificado que não há solução inteira para valores de Z maiores que γ^k, o algoritmo necessàriamente converge, se existir solução inteira do problema, após $[Z^o] - Z^* + 1$ iterações, no máximo, onde Z^* é o valor de Z na solução inteira ótima.

Para o caso do problema parcial ou misto, o método é o mesmo; apenas a penetração é feita com um hiperplano $Z' = \sum_{k \in E} c_k x_k$, E sendo o conjunto de índices das variáveis inteiras, isto é, penetramos com um hiperplano que é paralelo a Z no sub-espaço das variáveis inteiras.

Também neste caso supomos os coeficientes c_k inteiros e primos entre si, de modo que às soluções com x_k inteiro correspondam

também valores inteiros de Z', valendo, portanto, tôdas as considerações anteriores, inclusive sôbre convergência. A verificação do ponto inteiro incluido é restrita, òbviamente, ao sub-espaço das variáveis inteiras.

A única limitação do método proposto nesta tese para o problema parcial é a exigência de haver pelo menos uma variável inteira x_k com coeficiente $c_k \neq 0$ na função objetivo. Se todos os coeficientes de variáveis inteiras, na função objetivo, forem nulos, o método das soluções múltiplas não pode ser aplicado, pela impossibilidade de fixar a profundidade de penetração. Felizmente tais casos são bastante raros na prática.

2.1 - Algoritmo I

Suporemos, neste capítulo que o problema original, sem restrição de inteireza, admite uma única solução ótima. O caso em que o problema original já apresenta, de início, soluções múltiplas, será abordado no capítulo III.

Suporemos , também, que as variáveis com restrição de inteireza são as r primeiras, isto é, $E = 1, 2 \ldots r$. No caso do problema total, evidentemente, $r = n$.

O algoritmo I será explicado com o auxílio do diagrama de bloco da fig. 2.3.

(1)- Resolver o problema original de programação linear por meio de um dos algoritmos do simplex.

(2)- Se a solução ótima obtida fôr inteira, o problema está terminado. Se não, passe para (3).

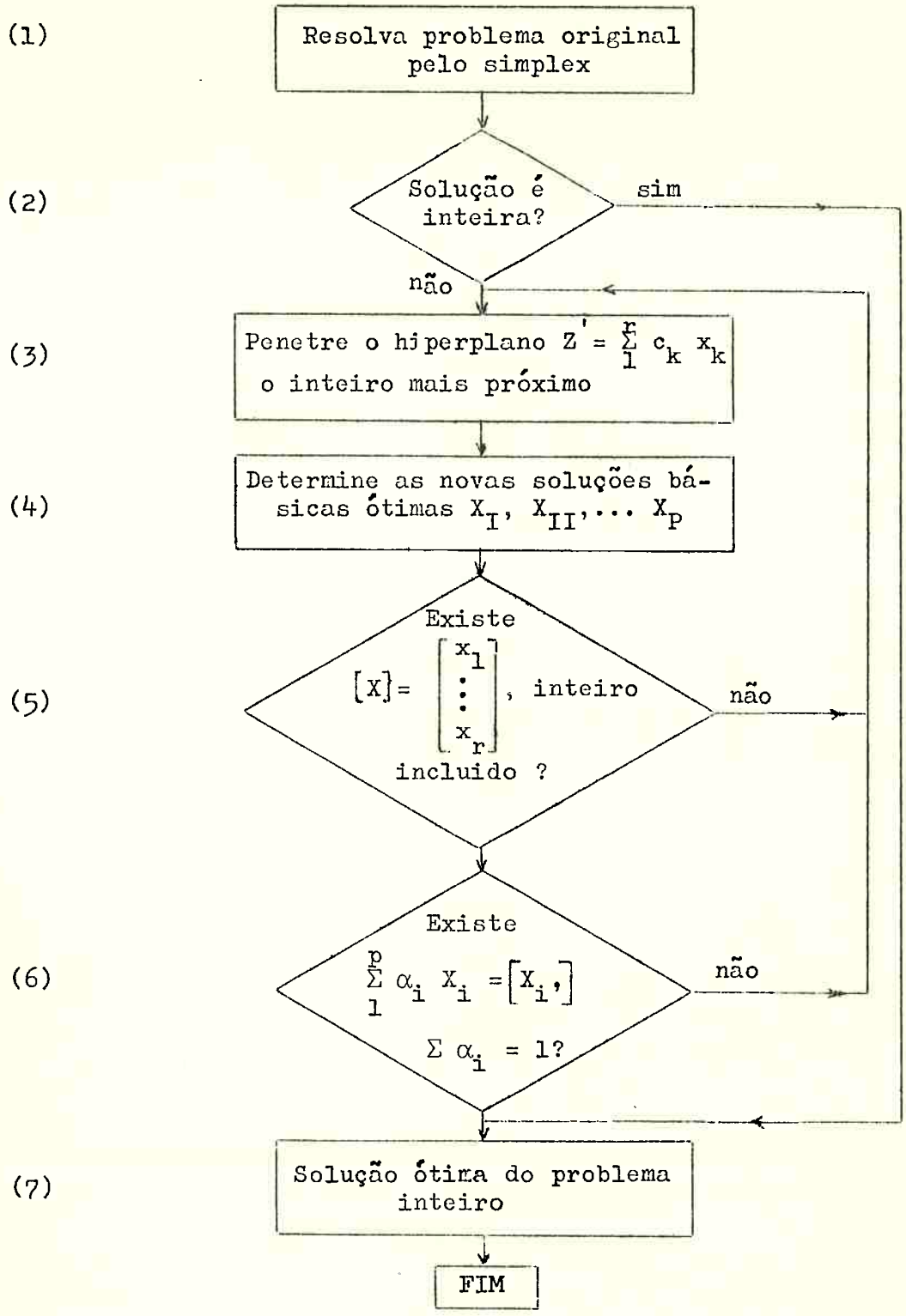


Fig. 2.3 - Algoritmo I

(3)- Penetração do hiperplano $Z' = \sum_{1}^{r} c_k x_k$ até o inteiro mais próximo. É obtida introduzindo-se a restrição adicional:

$$Z' = \sum_{1}^{r} c_k x_k \leq \gamma^1 \qquad (2.1)$$

onde γ^1 é o maior inteiro contido em $Z^{o'}$, se êste não fôr inteiro, ou $Z^{o'} - 1$ se $Z^{o'}$ fôr inteiro.

Evidentemente, como $\gamma^1 < Z^{o'}$ a restrição adicional (2.1) não será satisfeita pela solução obtida em (1).

Para introduzir a restrição diretamente na tabela da solução ótima do simplex basta lembrar que, de modo geral, as variáveis básicas podem ser escritas:

$$x_i = a_{io} + \sum_{j} a_{ij} (- x_j) \qquad (2.2)$$

onde x_j são as variáveis fora da base. Também:

$$Z' = Z^{o'} + \sum_{j} (Z'_j - c_j) (-x_j) \qquad (2.3)$$

Introduzindo-se em (2.1) a variável residual x_{n+1} e substituindo em (2.3), temos:

$$\Sigma (Z'_j - c_j)(- Z_j) + x_{n+1} = \gamma' - Z^{o'} = - \Delta \qquad (2.4)$$

onde Δ é a penetração feita.

A tabela do novo problema fica, portanto:

Base	P_0	P_1	...	P_m	...	P_j	P_k	...	P_{n+1}
x_1	a_{1o}	1		0		a_{1j}	a_{1k}		0
.	.	.		.		.	.		.
.	.	.		.		.	.		.
.	.	.		.		.	.		.
x_m	a_{mo}	0		1		a_{mj}	a_{mk}		0
x_{n+1}	$-\Delta$	0		0		$-(Z'_j-c_j)$	$-(Z'_k-c_k)$		1
	Z^o	0		0		Z_j-c_j	Z_k-c_k		0

(2.5)

Como a solução anterior era ótima, todos os $Z_j - c_j > 0$, para j fora da base. Logo temos tôdas as condições para aplicar o algoritmo dual simplex. O vetor que sai da base é evidentemente, x_{n+1}. O que entra é determinado pelo critério do dual simplex:

$$\min \frac{Z_k - c_k}{|Z'_k - c_k|}, \quad \text{com } Z'_k - c_k < 0, \quad Z_j - c_j > 0$$

No caso do problema total todos os vetores fora da base têm o mesmo valor 1 para êste quociente, ocorrendo empate. No problema parcial é possível que haja um únioo vetor para entrar na base, mas também pode ocorrer empate entre vários.

(4)- Determinar as novas soluções básicas ótimas.

Um vetor x_k que entra na base dará uma nova solução X_k.

Base	P_o	P_j	P_{n+1}
x_1	$a_{1o} - \frac{a_{1k}}{Z'_k - c_k} \Delta$	$a_{1j} - \frac{a_{1k}}{Z'_k - c_k} \cdot (Z'_j - c_j)$	$- \frac{a_{1k}}{Z'_k - c_k}$
. . .	.....	.....	.....
x_m	$a_{mo} - \frac{a_{mk}}{Z'_k - c_k} \Delta$	$a_{mj} - \frac{a_{mk}}{Z'_k - c_k} (Z'_j - c_j)$	$- \frac{a_{mk}}{Z'_k - c_k}$
x_k	$\frac{1}{Z'_k - c_k} \Delta$	$\frac{Z'_j - c_j}{Z'_k - c_k}$	$- \frac{1}{Z'_k - c_k}$
	$Z^o - \Delta$	$(Z_j - c_j) - \frac{Z'_j - c_j}{Z'_k - c_k} (Z_k - c_k)$	$\frac{Z_k - c_k}{Z'_k - c_k}$

(2.6)

Como só precisamos das soluções básicas e em cada penetração muda apenas o valor Δ, permanecendo os demais elementos da tabela (desde que não ocorra nenhum $x_i < 0$ o que indicaria a necessidadade de mudança de base), não é preciso calcular tôda a tabela.

Podemos escrever uma solução X_k sob forma paramétrica

$$X_k = \left[\begin{array}{l} x_1 = a_{1o} - \dfrac{a_{1k}}{Z_k' - c_k}\,\Delta \\ \cdots\cdots\cdots\cdots\cdots \\ x_m = a_{mo} - \dfrac{a_{mk}}{Z_k' - c_k}\,\Delta \\ x_k = \dfrac{1}{Z_k' - c_k}\,.\,\Delta \end{array}\right] \qquad (2.7)$$

onde os índices (i .... m) indicam os vetores da base ótima inicial e k um vetor inicialmente fora da base original e que satisfez ao critério de entrada do dual simplex. Lembramos que no caso do problema total todos os vetores fora da base inicial são candidatos à entrada.

Podemos observar que as m primeiras componentes de (2.7) podem ser escritos sob a forma

$$x_i = a_{io} + \bar{a}_{i,n+1}\,\Delta \qquad (2.8)$$

onde $\bar{a}_{i,\,n+1}$ é a componente i do vetor P_{n+1}, residual da nova restrição.

Desta forma calculamos tôdas as soluções alternativas X_I, X_{II} ... X_p do problema com penetração.

(5)- Pesquisar se existe uma solução inteira incluida

Determinamos o máximo e o mínimo de cada variável x_k inteira, e verificamos se existe algum inteiro entre êstes dois valores, êles incluidos.

Se não existir voltamos para (3), penetrando o hiperplano Z' até o próximo valor inteiro.

(6)- Pesquisa de uma combinação linear convexa

Existindo um ponto inteiro possível $y_1, y_2 \ldots y_r$ no sub-espaço das variáveis inteiras, precisamos verificar se tem solução o sistema de (r + 1) equações:

$$\begin{aligned}
&\alpha_I x_{I1} + \alpha_{II} x_{II1} + \ldots + \alpha_P x_{P1} = y_1 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\alpha_I x_{Ir} + \alpha_{II} x_{IIr} + \ldots + \alpha_P x_{Pr} = y_r \\
&\alpha_I + \alpha_{II} + \ldots + \alpha_P = 1 \\
&\alpha_i \geqslant 0 \qquad i = 1, \ldots p
\end{aligned} \tag{2.9}$$

No caso do problema total com matriz A inteira a determinação é facilitada pelo fato de, sendo tôdas as variáveis inteiras, cada variável x_k introduzida na base permite determinar os valores possíveis do α_k correspondente, pois α_k tem um campo de variação discreto:

$$\alpha_k = 0, \frac{1}{x_k}, \frac{2}{x_k} \ldots \frac{[x_k]}{x_k}$$

Êste método é bastante eficiente se todos os valores x_k forem pequenos.

Caso o sistema (2.9) não admita solução voltamos a (3) e realizamos nova penetração.

Se existir mais de uma solução do sistema (2.9) teremos múltiplas soluções inteiras ótimas.

2.2 - Exemplo de Aplicação

Vamos resolver o seguinte problema, que se encontra resolvido em Carr e Howe (1964), página 222 pelo algoritmo I de Gomory:

$$Z\ (\max) = 2\,x_1 + x_2$$

$$20\,x_1 + 10\,x_2 \geqslant 75$$

$$12\,x_1 + 7\,x_2 \leqslant 55$$

$$25\,x_1 + 10\,x_2 \leqslant 90$$

$$x_1 \geqslant 0 \quad \text{inteiro}$$

$$x_2 \geqslant 0 \quad \text{inteiro}$$

Introduzindo as variáveis residuais x_3, x_4 e x_5 respectivamente, e resolvendo pelo simplex temos a seguinte tabela de solução ótima sem restrição de inteireza, dada por Carr e Howe (1964)

Base	P_o	P_1	P_2	P_3	P_4	P_5
x_2	295/55	0	1	0	25/55	- 12/55
x_3	425/55	0	0	1	50/55	20/55
x_1	80/55	1	0	0	-10/55	7/55
$Z_j - c_j$	455/55	0	0	0	5/55	2/55

A nova restrição será dada por:

$$-\Delta = 5/55\ (-x_4) + 2/55\ (-x_5) + x_6$$

e temos, iniciando com $\Delta = 15/55$

Base	P_o	P_1	P_2	P_3	P_4	P_5	P_6
x_2	295/55	0	1	0	25/55	-12/55	0
x_3	425/55	0	0	1	50/55	20/55	0
x_1	80/55	1	0	0	-10/55	7/55	0
x_6	-15/55	0	0	0	- 5/55	-2/55	1
$Z_j - c_j$	8	0	0	0	5/55	2/55	0

Introduzindo na base P_4 e P_5 temos as duas soluções

X_I

$$x_1 = \frac{80}{55} + \frac{10}{55} \cdot \frac{55}{5} \cdot \frac{15}{55} = \frac{110}{55} = 2$$

$$x_2 = \frac{295}{55} - \frac{25}{55} \cdot \frac{55}{5} \cdot \frac{15}{55} = \frac{220}{55} = 4$$

$$x_3 = \frac{425}{55} - \frac{50}{55} \cdot \frac{55}{5} \cdot \frac{15}{55} = \frac{275}{55} = 5$$

$$x_4 = \frac{55}{5} \cdot \frac{15}{55} = \frac{15}{5} = 3$$

$$x_5 = 0 = 0$$

X_{II}

$$x_1 = \frac{80}{55} - \frac{7}{55} \cdot \frac{15}{2} = 1/2$$

$$x_2 = \frac{295}{55} + \frac{12}{55} \cdot \frac{15}{2} = 7$$

$$x_3 = \frac{425}{55} - \frac{20}{55} \cdot \frac{15}{2} = 5$$

$$x_4 = 0 = 0$$

$$x_5 = \frac{55}{2} \cdot \frac{15}{55} = \frac{15}{2}$$

Temos $\max\ x_1 = 2$, $\min\ x_1 = 1/2$; logo $y_1 = 2$ ou 1

$\max\ x_2 = 7$ $\min\ x_2 = 4$ $y_2 = 7$ ou 6 ou 5 ou 4

$\max\ x_3 = 5$ $\min\ x_3 = 5$ $y_3 = 5$

$\max\ x_4 = 3$ $\min\ x_4 = 0$ $y_4 = 3$ ou 2 ou 1 ou 0

$\max\ x_5 = \frac{15}{2}$ $\min\ x_5 = 0$ $y_5 = 7, 6, 5, 4, 3, 2, 1$

$$\left.\begin{aligned} \alpha_I X_I + \alpha_{II} X_{II} &= y \\ \alpha_I + \alpha_{II} &= 1 \end{aligned}\right\}$$ Pesquisa das Combinações Convexas.

Evidentemente, como X_I é inteiro, uma solução é $\alpha_I = 1$, $\alpha_{II} = 0$.

Como x_4 deve ser inteiro, os valores possíveis para α_I são $0, \frac{1}{3}, \frac{2}{3}, 1$.

Como x_5 deve ser inteiro, os valores possíveis para α_{II} são $0, \frac{2}{15}, \frac{4}{15}, \frac{6}{15}, \frac{8}{15}, \frac{10}{15}, \frac{12}{15}, \frac{14}{15}$

A única combinação que dá $\alpha_I + \alpha_{II} = 1$, $\alpha_I = \frac{1}{3}$, $\alpha_{II} = \frac{2}{3}$, que nos fornece a solução:

$$x_1 = \frac{1}{3} \cdot 2 + \frac{2}{3} \cdot \frac{1}{2} = 1$$

$$x_2 = \frac{1}{3} \cdot 4 + \frac{2}{3} \cdot 7 = 6$$

$$x_3 = \frac{1}{3} \cdot 5 + \frac{2}{3} \cdot 5 = 5$$

$$x_4 = \frac{1}{3} \cdot 3 + \frac{2}{3} \cdot 0 = 1$$

$$x_5 = \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot \frac{15}{2} = 5$$

O problema admite portanto duas soluções ótimas inteiras:

$$X_I = (x_1 = 2, \quad x_2 = 4)$$

$$X_{II} = (x_1 = 1, \quad x_2 = 6)$$

É interessante comparar com a solução apresentada por Carr e Howe (1964), que é igual à nossa solução X_{II}, e exigiu a introdução de quatro cortes, com as correspondentes tabelas completas.

2.3 - Mudança de base durante a penetração

À medida que vamos penetrando o hiperplano Z' as soluções vão se modificando e podem, num certo instante, deixar de serem viáveis, isto é, conterem valores negativos.

Da fórmula de transformação de uma variável qualquer da base:

$$x_i = a_{io} - \frac{a_{ik}}{Z'_k - c_k} \Delta \qquad (2.10)$$

concluimos que, sendo $Z'_k - c_k > 0$ e $\Delta > 0$, a única possibilidade de $x_i \leqslant 0$ é se $a_{ik} > 0$ e $\Delta > \frac{a_{io}}{a_{ik}}(Z'_k - c_k)$.

Evidentemente, se todos os $a_{ik} \leqslant 0$, para qualquer i e k, não haverá troca de base. Se tomarmos:

$$\Delta \text{ crítico} = \min_{\substack{i,k \\ a_{ik} > 0}} \left[\frac{a_{io}}{a_{ik}} (Z'_k - c_k) \right] \qquad (2.11)$$

teremos o limite superior de Δ compatível com a base inicial, isto é, a base inicial só é viável no intervalo $0 < \Delta \leqslant \Delta$ crítico.

Se $\Delta > \Delta$ crítico, devemos mudar de base, se quizermos que as nossas soluções múltiplas X_I, X_{II}, ... X_P sejam sempre viáveis. Geomètricamente esta mudança de base significa que uma das interseções do poliedro com o hiperplano Z' é agora numa face diferente daquela que estavamos considerando.

Feita a mudança de base, calculamos novo Δ crítico, que dará o limite superior de Δ para validade desta base, e assim por diante.

Ora, se estamos trabalhando com a solução sob forma paramétrica, teriamos de calcular tôda a tabela do simplex correspondente à penetração feita, para podermos fazer a substituição de base, utilizando, como temos feito, o algoritmo dual simplex.

Devemos notar, entretanto, que o algoritmo das combinações convexas não exige que os pontos extremos sejam todos viáveis, e vale da mesma forma se houver algumas coordenadas negativas.

Desde que haja ponto inteiro positivo incluído o nosso algoritmo nos fornecerá a solução.

O único incoveneiente, neste caso, é a possibilidade de virmos a examinar mais pontos inteiros desnecessàriamente, pois estamos pesquisando num campo maior, no sentido que o conjunto (agora não viável) da base antiga contém o conjunto que corresponderia à nova base.

2.3.1 - Exemplo

Para ilustrar êste problema de mudança de base durante a pentração, onsiderenos o problema seguinte, apresentado por Hadley (1964), p.285.

$$Z(\max) = 0{,}25\, x_1 + x_2$$

sujeito a:

$$0{,}50\, x_1 + x_2 \leq 1{,}75$$

$$x_1 + 0{,}30\, x_2 \leq 1{,}50$$

$$x_1,\ x_2 \geq 0,\ \text{inteiros}$$

Para aplicar o nosso algoritmo, Z deve ter coeficientes inteiros. É também conveniente no problema totalmente inteiro ter a matriz A inteira, embora não seja indispensável ao algoritmo.

Transformamos, portanto, o problema original no equivalente já sob forma canônica:

$$Z\ (\max) = x_1 + 4\, x_2$$

sujeito a:

$$2\, x_1 + 4\, x_2 + x_3 = 7$$

$$10\, x_1 + 3\, x_2 + x_4 = 15$$

com $x_j \geq 0$, inteiro

Temos:

Base	P_0	P_1	P_2	P_3	P_4
x_3	7	2	4	1	0
x_4	15	10	3	0	1
$Z_j - c_j$	0	-1	-4	0	0

Entra o P_2 e sai o P_3:

Base	P_0	P_1	P_2	P_3	P_4
x_2	7/4	1/2	1	1/4	0
x_4	39/4	17/2	0	-3/4	1
$Z_j - c_j$	7	1	0	1	0

Solução ótima sem restrição.

Como x_2 e x_4 não são inteiros, embora Z o seja, vamos penetrar uma profundidade Δ, introduzindo a restrição:

$$-\Delta = -x_1 - x_3 + x_5$$

e temos as duas soluções paramétricas.

X_I	X_{II}
$x_1 = \Delta$	$x_1 = 0$
$x_2 = 7/4 - 1/2\ \Delta$	$x_2 = 7/4 - 1/4\ \Delta$
$x_3 = 0$	$x_3 = \Delta$
$x_4 = 39/4 - 17/2\ \Delta$	$x_4 = 39/4 + 3/4\ \Delta$

Δ crítico = min (7/2, 7, 39/34) = 1,147

Inicialmente tomamos $\Delta = 1$ e temos:

$\Delta = 1$
$Z = 6$

	X_I^1	X_{II}^1	inteiro incluido
x_1	1	0	sim
x_2	5/4	6/4	não
x_3	0	1	sim
x_4	42/4	5/4	sim

Como os valores de x_2 não incluem um inteiro, passamos para $\Delta = 2$, e temos:

$\Delta = 2$

$Z = 5$

	X_I^2	X_{II}^2	inteiro incluido
x_1	2	0	sim
x_2	3/4	5/4	sim
x_3	0	2	sim
x_4	-29/4	45/4	sim

Conforme era previsto, a solução X_I não é viável, pois estamos com $\Delta > \Delta$ crítico. Mas, como dissemos, continua valendo o algoritmo das combinações convexas. Com efeito, temos a combinação $\alpha_I = 1/2$, $\alpha_{II} = 1/2$, que dá a solução inteira:

	$1/2\ X_I^2 + 1/2\ X_{II}^2$
x_1	$1/2 \cdot 2 + 1/2 \cdot 0 = 1$
x_2	$1/2 \cdot 3/4 + 1/2 \cdot 5/4 = 2$
x_3	$1/2 \cdot 0 + 1/2 \cdot 2 = 1$
x_4	$1/2(-29/4) + 1/2 \cdot 45/4 = 2$

Vamos, entretanto, efetuar a mudança de base, como ilustrar. A tabela completa da base X_I é:

Base	P_0	P_1	P_2	P_3	P_4	P_5
x_2	3/4	0	1	-1/4	0	1/2
x_4	-29/4	0	0	-37/4	1	17/2
x_5	2	1	0	1	0	-1
$Z_j - c_j$	5	0	0	0	0	1

Sai x_4 e entra x_3, dando a tabela:

Base	P_0	P_1	P_2	P_3	P_4	P_5
x_2	35/37	0	1	0	1/37	10/37
x_3	29/37	0	0	1	-4/37	-34/37
x_1	45/37	1	0	0	4/37	- 3/37
Z_j-c_j	5	0	0	0	0	1

As soluções são agora:

	$X_I^{2'}$	$X_{II}^{2'}$
x_1	45/37	0
x_2	35/37	5/4
x_3	29/37	
x_4	0	45/37

$\Delta = 2$

$Z = 5$

Neste caso a combinação convexa é dada por $\alpha_1 = 37/45$, $\alpha_2 = 8/37$, resultando na mesma solução inteira anterior:

$$x_1 = 1$$
$$x_2 = 2$$
$$x_3 = 1$$
$$x_4 = 2$$

A figura 2.3 nos mostra o que acontece no espaço das soluções quando mantemos a base não viável.

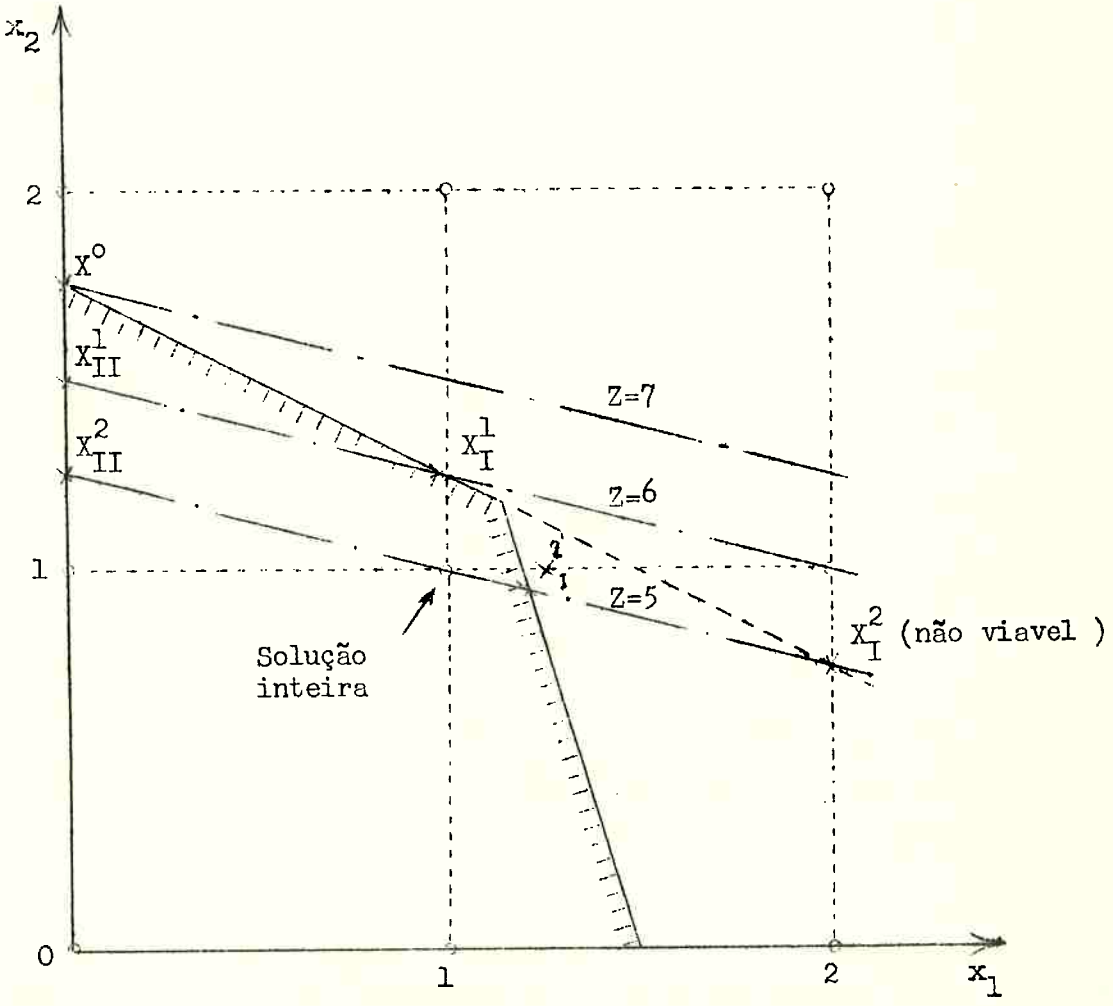


Fig. 2.3

2.4 - Variáveis Bivalentes

O algoritmo não oferece qualquer dificuldade de aplicação, mas a eficiência é maior se empregarmos na solução do problema sem restrição de inteireza o algoritmo do simplex para variáveis com limite superior, de forma a ter, já no início da penetração,

$$0 \leqslant x_i \leqslant 1$$

para as variáveis bivalentes.

2.5 - Apreciação Geral sôbre o Algoritmo I

O algoritmo I, das combinações lineares convexas, foi usado com sucesso na solução manual de vários problemas de pequeno porte, apresentados por diversos autores como ilustração didática, e dos quais reproduzimos mais alguns exemplos no Apêndice I para comparação.

Êle é também fàcilmente adaptável a qualquer um dos diversos algoritmos do simplex, e especialmente ao simplex revisado, onde podemos introduzir a restrição adicional logo de início, sem permitir a sua saida da base na fase I (solução sem restrição de inteireza).

Entretanto é preciso reconhecer o aspecto negativo, de que a solução do sistema de equações (2.9) das combinações convexas torna-se bastante trabalhosa à medida que aumenta o porte do problema.

Outra questão dificil de contornar é a perda de precisão no cálculo numérico. A não ser que a solução seja tôda calculada com

frações ordinárias, o que é impraticável num computador eletrônico, auxiliar indispensável para resolver problemas grandes, será impossível obter a solução exata do sistema (2.9).

Estas razões nos levaram a modificar o algoritmo I em sua segunda parte, substituindo a determinação das combinações convexas por um processo de enumeração implícita. Êste algoritmo II é apresentado no capítulo seguinte.

C A P Í T U L O III

ALGORITMO II

3.0 - Considerações Iniciais

A grande facilidade do computador eletrônico em fazer comparação de valores, conjugada com as dificuldades de obter uma combinação linear convexa de muitas soluções e a perda de precisão do cálculo numérico, nos levaram a modificar a segunda parte do algoritmo I, substituindo-a por um processo de enumeração implícita.

Tôda a parte inicial do algoritmo permanece conceitualmente a mesma; apenas fizemos pequenas modificações, eliminando algumas passagens que se tornaram desnecessárias.

Com efeito, na enumeração implícita é supérfluo calcular tôdas as soluções básicas X_I, X_{II}, ... X_P bastando determinar, para todo x_k que deva ser inteiro, o seu valor máximo e mínimo no conjunto das soluções.

3.1 - Determinação dos valores limites das Variáveis

Na expressão da solução básica X_k sob forma paramétrica, tinhamos

$$x_i = a_{io} - \frac{a_{ik}}{Z'_k - c_k} \Delta \qquad (3.1)$$

$$x_k = \frac{1}{Z'_k - c_k} \cdot \Delta \qquad (3.2)$$

onde x_i é uma variável da solução básica original e x_k é uma variável introduzida pela penetração do hiperplano.

Definamos:

$$D\ max_i = - \min_k \frac{a_{ik}}{Z'_k - c_k} \qquad (3.3)$$

$$D\ min_i = - \max_k \frac{a_{ik}}{Z'_k - c_k} \qquad (3.4)$$

$$D\ max_k = \frac{1}{Z'_k - c_k} \qquad (3.5)$$

$$D\ min_k = 0 \qquad (3.6)$$

Neste caso podemos escrever diretamente:

$$max\ x_j = a_{jo} + D\ max_j \cdot \Delta \qquad (3.7)$$

$$min\ x_j = a_{jo} + D\ min_j \cdot \Delta \qquad (3.8)$$

expressão válida para qualquer j.

Consideremos um x_j que deva ser inteiro e indiquemos, como anteriormente, por $[x]$ o maior inteiro contido em x.

Para determinar o número h_j de inteiros existentes no intervalo $(max\ x_j, min\ x_j)$, precisamos distinguir dois casos, conforme $min\ x_j$ seja ou não inteiro. Temos:

$$h_j = \lceil \max x_j \rceil - \lceil \min x_j \rceil \quad \text{se } \min x_j \neq \lceil \min x_j \rceil \qquad (3.9)$$

$$h_j = \lceil \max x_j \rceil - \lceil \min x_j \rceil + 1 \quad \text{se } \min x_j = \lceil \min x_j \rceil \qquad (3.10)$$

Se, para tôdas as variáveis x_j que devem ser inteiras, tivermos $h_j \geqslant 1$, então há possibilidade de existir uma solução inteira.

3.2 - Enumeração Implícita das Soluções Inteiras

Devemos então testar tôdas as combinações possíveis de valores inteiros de cada uma das variáveis. Ora, êste é um problema combinatório que pode fàcilmente atingir dimensões enormes, pois o número de combinações é dado pelo produto $\prod_{1}^{r} h_i$, o que torna impraticável verificar tôdas as possibilidades.

Felizmente um algoritmo de enumeração implícita permitirá eliminar a verificação das combinações que òbviamente não satisfazem ao problema, conforme discutimos de modo geral no Capítulo I.

Consideremos, agora, o problema totalmente inteiro. Vamos então enumerar implìcitamente tôdas as "soluções" possíveis através de uma árvore, definida pelo índice j da variável correspondente a cada nível, a qual pode tomar os h_j valores inteiros.

$$\lceil \max x_j \rceil \ , \ \lceil \max x_j \rceil - 1, \ \ldots \ \lceil \max x_j \rceil - h_j + 1.$$

Tomando inicialmente $j = 1$ vamos construíndo a árvore, até chegarmos a $j = n$ quando teremos enumerado tôdas as possíveis soluções inteiras.

Uma solução deve satisfazer às restrições:

$$\sum_{1}^{n} c_j x_j = Z = \gamma^o - \Delta \qquad (3.11)$$

$$\sum_{1}^{n} a_{ij} x_j \leqslant a_{io}\ , \qquad i = 1, 2 \ldots m \qquad (3.12)$$

A equação (3.11) exprime a condição que o ponto deve estar sôbre o hiperplano Z na sua posição corrente, $\gamma^o - \Delta$; as demais são as restrições originais do problema.

Para colocar tôda a verificação da solução numa única rotina, vamos fazer $c_j = a_{oj}$ e $\gamma^o - \Delta = a_{oo}$, de modo que podemos escrever as restrições sob a forma:

$$\sum_{1}^{n} a_{ij} x_j \leqslant a_{io} \qquad i = 0, 1, \ldots m \qquad (3.13)$$

notando que para $i = 0$ deve valer a igualdade:

Sabemos que uma solução corresponde a um caminho partindo do nível 0 até um vértice n. Vamos supor que seguindo êste caminho, atingimos o nível j, isto quer dizer que já fixamos os valo- $x_1, x_2, \ldots x_j$ das j primeiras variáveis.

Conhecemos portanto o valor da soma parcial

$$\text{SIGMA } (i,j) = \sum_{1}^{j} a_{ij} x_j \qquad i = 0, 1, \ldots m \qquad (3.14)$$

que pode ser calculado fàcilmente a partir de

$$\left.\begin{aligned} \text{SIGMA } (i,j) &= \text{SIGMA } (i,j-1) + a_{ij} x_j \\ \text{SIGMA } (i,0) &= 0 \end{aligned}\right\} \qquad (3.15)$$

Vamos calcular agora um limite superior SUP (i,j) de cada restrição i no nível j, tal que, se

$$\text{SIGMA}\ (i,j) > \text{SUP}(i,j)$$

para uma restrição i, não existe combinação de valores das variáveis restantes, $x_j + 1, \ldots x_n$ que permita satisfaer a restrição i e portanto é inútil prosseguir neste caminho.

Para determinar êstes limites superiores, suponhamos SUP (i,j) já calculado, e vamos determinar SUP (i,j-1).

A diferença entre os dois níveis deve ser a menor contribuição possível da variável x_j.

Se $a_{ij} \geq 0$ esta corresponderá ao menor valor inteiro permitido para x_j, isto é, $[\max x_j] - h_j + 1$.

Se $a_{ij} < 0$ a menor contribuição será dada pelo maior valor inteiro possível para x_j, isto é, $[\max x_j]$.

Portanto, podemos calcular a matriz dos SUP (i,j) por recorrência, partindo de:

$$\text{SUP}\ (i,n) = a_{io} \qquad i = o, 1 \ldots m \tag{3.16}$$

$$\left.\begin{aligned} \text{SUP}(i,j-1) &= \text{SUP}(i,j) - a_{ij}\,[\max x_j] \quad , \quad a_{ij} < 0 \\ \text{SUP}(i,j-1) &= \text{SUP}(i,j) - a_{ij}([\max x_j] - h_j + 1) \quad a_{ij} \geq 0 \end{aligned}\right\} \tag{3.17}$$

Chegando ao nível j da árvore, duas situações podem ocorrer:

1) SIGMA (i,j) $\leqslant$ SUP (i,j).

Neste caso avançamos ao nível j+1, fixando

$x_{j+1} = \lceil \max x_{j+1} \rceil$ e calculamos a nova soma parcial, pela relação

$$SIGMA(i,j+1) = SIGMA(i,j) + a_{ij+1}\, x_{j+1} \qquad (3.18)$$

e testamos se, no novo nível

SIGMA(i,j+1) $\leqslant$ SUP (i,j+1) , e assim por diante.

2) SIGMA (i,j) $>$ SUP (i,j)

Neste caso devemos retroceder ao nível imediatamente anterior j-1 e tentar escolher um novo valor de x_j que permita satisfazer ao teste.

Como o problema é linear e os valores de x_j variam de 1 em 1, vamos utilizar êste fato para simplificar nossa busca. Como a árvore é construida de modo que a variável x_j é ordenada segundo valores decrescentes, a primeira possibilidade de variar x_j é no ramo à direita, com $\bar{x}_j = x_j - 1$, o que dá o novo valor

$$\overline{SIGMA\ (i,j)} = SIGMA\ (i,j) - a_{ij}$$

Se $a_{ij} > 0$, então $\overline{SIGMA}\ (i,j) < SIGMA\ (i,j)$ e, é possível que agora $\overline{SIGMA}\ (i,j) \leqslant SUP\ (i,j)$.

Se ainda $\overline{SIGMA}\ (i,j) > SUP\ (i,j)$ deslocamos para o próximo ramo à direita e repetimos, ou até encontrar um valor

SIGMA (i,j) $\leqslant$ SUP (i,j) (que permitirá retomar a descida da árvore), ou até chegar ao menor valor possível de x_j (ramo mais à direita), que é $x_j = \left[\max x_j\right] - h_j + 1$.

Se ainda neste caso SIGMA (i,j) $\geqslant$ SUP (i,j), então devemos retroceder ao nível j-2, onde mantemos a escolha de x_1, x_2 ... x_{j-2} e tentamos, diminuindo 1 do valor corrente de x_{j-1}, prosseguir a descida.

Se $a_{ij} < 0$ então o deslocamento para a direita irá aumentar o valor de SIGMA (i,j) e portanto, podemos retroceder imediatamente ao nível (j-2) e passar à explorar o ramo da direita de x_{j-1}, tentando prosseguir na descida.

Se conseguirmos chegar ao nível n, com

$$\text{SIGMA } (0,n) = \text{SUP } (0,n)$$

$$\text{SIGMA } (i,n) \leqslant \text{SUP } (i,n) \qquad i = 1, 2 \ldots m$$

teremos encontrado uma solução que satisfaz às restrições do problema.

Se ao retroceder chegarmos ao nível j=0 podemos concluir que não existe solução inteira para aquêle valor de Z; fazemos nova penetração e recomeçamos.

Pode acontecer que existam várias soluções inteiras do problema. Para determiná-las basta prosseguir a busca, depois de ter achado uma solução, retrocedendo ao nível n-1 e repetindo o procedimento, até chegar a retroceder ao nível j=0 quando a busca estará terminada.

3.2.1 - Problema Parcial ou Misto

Neste caso a árvore das soluções inteiras possíveis é montada apenas com as variáveis inteiras x_1, x_2 ... x_r, e a verificação de uma solução é feita em duas fases.

Na fase I verificamos pelo procedimento anterior se o ponto inteiro x_1, x_2 ... x_r está sôbre o hiperplano.

$$Z'^{*} = \sum_{1}^{r} c_j x_j = \gamma^{o'} - \Delta \qquad (3.19)$$

Na fase II cada sub-conjunto x_1^*, x_2^* ... x_r^* que satisfaz a equação (3.19) é substituido no problema original e temos um problema de programação linear ordinária com (n-r) variáveis.

$$\left.\begin{aligned} & Z\ (\max) = Z'^{*} + \sum_{r+1}^{n} c_j x_j \\ & \text{sujeito a:} \\ & \sum_{r+1}^{n} a_{ij} x_j \leqslant a_{io} - \sum_{1}^{r} a_{ij} x_j^* \qquad i = 1, 2 \ldots n \\ & x_j \geqslant 0 \qquad j = r+1, \ldots n \end{aligned}\right\} \qquad (3.20)$$

Se o problema (3.20) tiver uma solução viável x_{r+1}^*, ... x_n^* a solução do problema original será, evidentemente

$$x_1^*, x_2^*, \ldots, x_r^*, x_{r+1}^* \ldots x_n^*$$

Se o problema (3.19) não tiver solução viável então faremos nova penetração do hiperplano $Z' = \sum_{1}^{r} c_j x_j$ e repetimos o procedimen-

to anterior.

3.3 - Erros de Aproximação no Cálculo Numérico

Como no algoritmo II apenas precisamos enquadrar os valores inteiros de x_i, podemos fàcilmente eliminar o êrro devido as aproximações do cálculo numérico, se aumentarmos de 2σ a largura da faixa entre $\max x_i$ e $\min x_j$.

Para isto, modificamos as fórmulas (3.3) a (3.6) para

$$D \max x_i = - \min_k \frac{a_{ik}}{Z'_k - c_k} + \sigma \qquad (3.21)$$

$$D \min x_i = - \max_k \frac{a_{ik}}{Z'_k - e_k} - \sigma \qquad (3.22)$$

$$D \max x_k = \frac{1}{Z'_k - c_k} + \sigma \qquad (3.23)$$

$$D \min x_k = 0 \qquad (3.24)$$

Para $D \min x_k$ não há modificação, pois o valor é exato. Desta forma não há perigo de passarmos por cima de um ponto inteiro sem reconhecê-lo. Apenas teremos, eventualmente, de pesquisar mais pontos inteiros que os estritamente necessários.

Em nossos cálculos no computador eletrônico temos usado $\sigma = 0{,}01$.

3.4 - Algoritmo II para o Problema Total

Podemos agora formalizar o algoritmo, conforme o diagrama de bloco da figura 3.1.

(1) Resolva o problema original por um dos algoritmos do simplex.

(2) Se a solução é inteira, o problema está terminado.

(3) Se a solução não é inteira calcule para cada variável os acréscimos.

$$D\max_j = -\min_k \frac{a_{jk}}{Z_k - c_k} + \sigma \qquad j \in \text{base}$$

$$D\min_j = -\max_k \frac{a_{jk}}{Z_k - c_k} - \sigma \qquad j \in \text{base}$$

$$D\max_k = \frac{1}{Z_k - c_k} + \sigma \qquad k \notin \text{base}$$

$$D\min_k = 0 \qquad k \notin \text{base}$$

(4) Penetre o hiperplano até o valor inteiro mais próximo fazendo:

$$\Delta = Z^o - [Z^o] \qquad \text{se} \quad Z^o > 0$$

$$\Delta = Z^o - [Z^o] + 1 \qquad \text{se} \quad Z^o < 0$$

(5) Determine os valores limites de cada variável

$$\max x_j = a_{jo} + D \max_j \cdot \Delta$$

$$\min x_j = a_{jo} + D \min_j \cdot \Delta \qquad j = 1, \ldots, n$$

$$h_j = \lceil \max x_j \rceil - \lceil \min x_j \rceil \qquad \text{se} \quad \min \neq \lceil \min x_j \rceil$$

$$h_j = \lceil \max x_j \rceil - \lceil \min x_j \rceil + 1 \qquad \text{se } \min x_j = \lceil \min x_j \rceil$$

(6) Se todos os $h_j > 0$ existem pontos inteiros incluídos

(7) Cálculo dos limites superiores das restrições

Para $i = 0, 1 \ldots m$ e $j = N, n-1, \ldots 1$ cálcule

$$SUP\ (i, j-1) = SUP\ (i,j) - a_{ij} \lceil \max_j \rceil \quad \text{se} \quad a_{ij} \leq 0$$

$$SUP\ (i, j-1) = SUP\ (i,j) - a_{ij} (\lceil \max_j \rceil - h_j + 1) \text{ se } a_{ij} > 0$$

$$SUP\ (i, n) = a_{io}$$

$$SUP\ (0, n) = a_{oo} = Z$$

(8) Efetue a enumeração implícita da árvore das soluções possíveis. O diagrama de bloco da fig. 3.2 índica detalhadamente como proceder.

(9) Achada uma solução inteira, para pesquisar as eventuais outras, proceda como indica o diagrama de bloco da fig. 3.3.

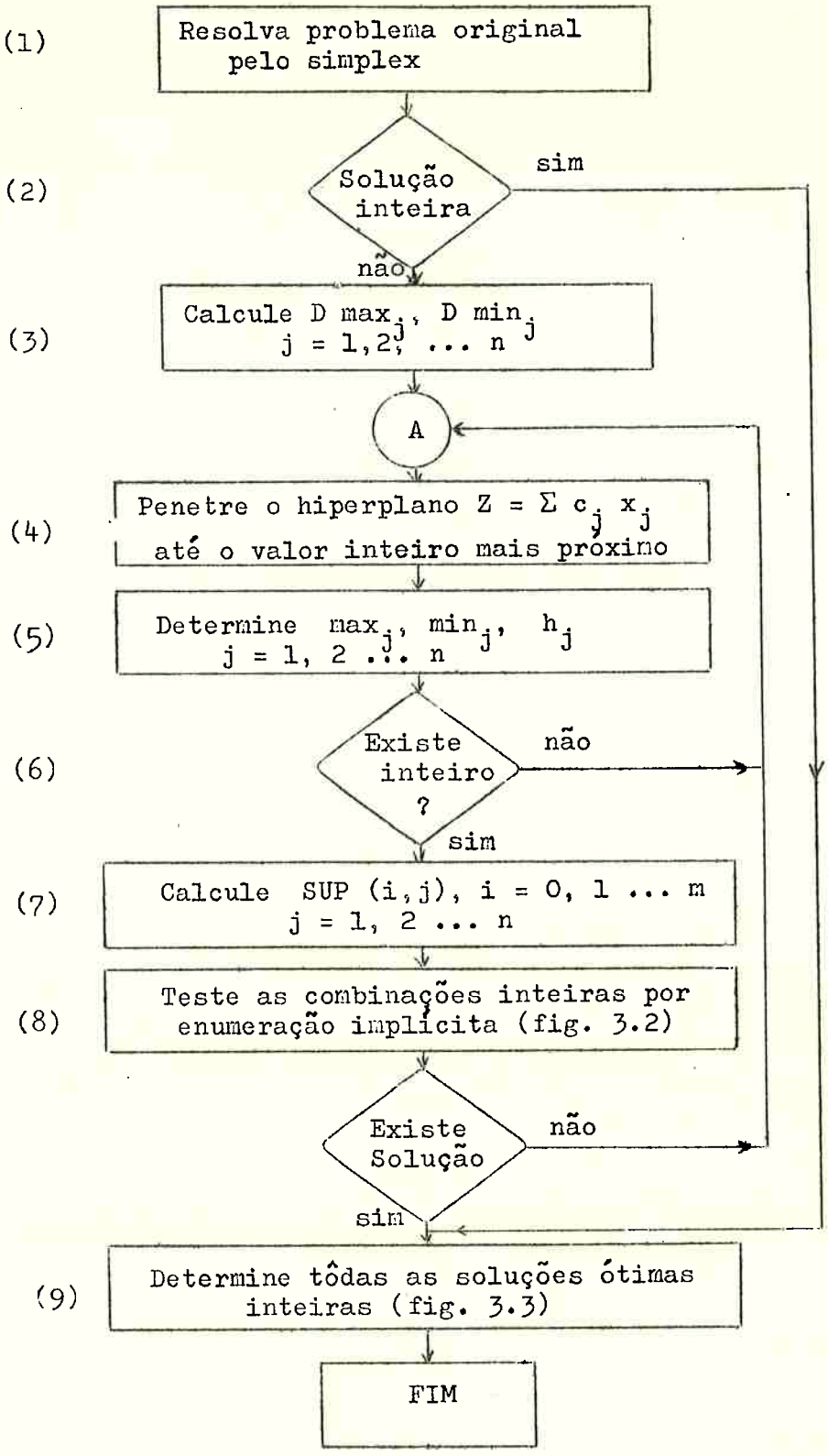


Fig. 3.1 - Algoritmo II para o problema total

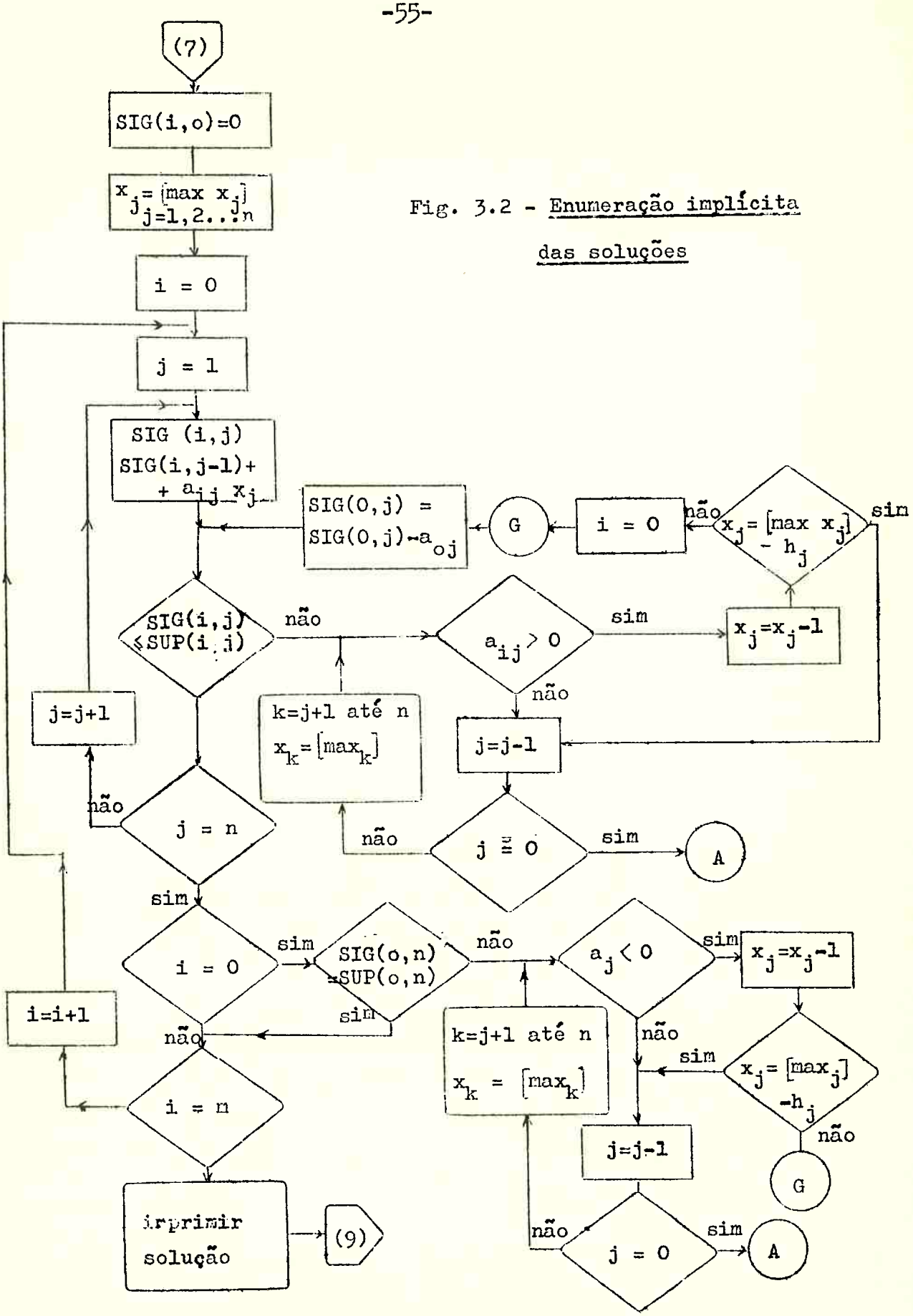


Fig. 3.2 - Enumeração implícita das soluções

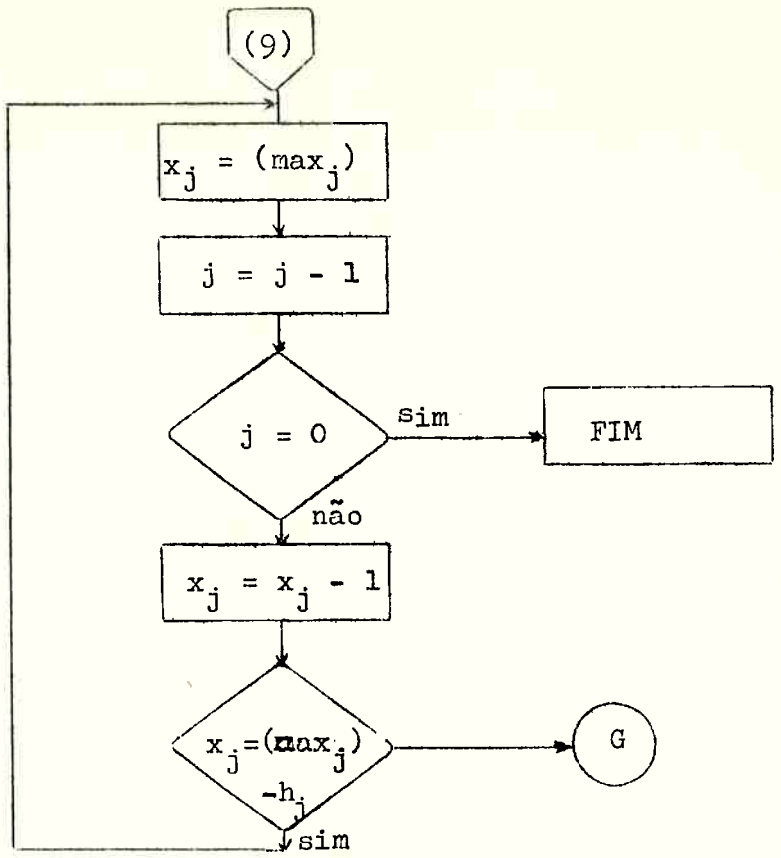


Fig. 3.3 - Pesquisa de outras soluções inteiras

3.4 - Exemplo de Aplicação

Vamos resolver o mesmo problema 2.2 anterior pelo algoritmo II. Da tabela final do simplex da pg. 30, temos

$$D\ max_1 = \frac{10}{55} \cdot \frac{55}{5} = 2 \qquad D\ min = -\frac{7}{55} \cdot \frac{55}{2} = -\frac{7}{2}$$

$$D\ max_2 = \frac{12}{55} \cdot \frac{55}{2} = 6 \qquad D\ min_2 = -\frac{25}{55} \cdot \frac{55}{5} = 5$$

$$\Delta = \frac{15}{55} \qquad Z = 8$$

$\max_1 = \frac{80}{55} + 2 \cdot \frac{15}{55} = 2$ $\quad \min x_2 = \frac{80}{55} - \frac{7}{2} \cdot \frac{15}{55} = 0{,}5$ $\quad h_1 = 2$

$\max_2 = \frac{295}{55} + 6 \cdot \frac{15}{55} = 7$ $\quad \min x_2 = \frac{295}{55} + 5 \cdot \frac{15}{55} = 4$ $\quad h_2 = 4$

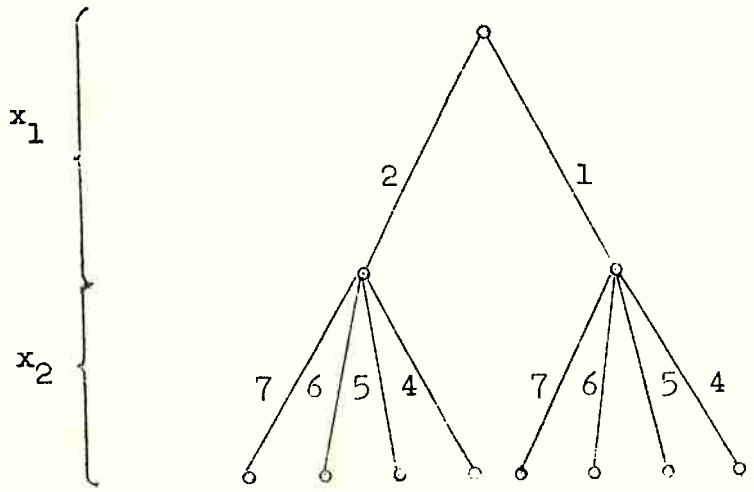
SUP (0,2) = 8 SUP (0,1) = 4

SUP (1,2) = - 75 SUP (1, 1) = - 50

SUP (2,2) = 55 SUP (2, 1) = 27

SUP (3, 2) = 90 SUP (3, 1) = 50

A árvore de soluções é:



Soluções ótimas inteiras:

$x_1 = 2 \quad x_2 = 4 \quad Z = 8$

$x_1 = 1 \quad x_2 = 6 \quad Z = 8$

3.6 - Algoritmo II para o problema parcial

Seja E = (1, 2, ... r) o conjunto de índices das variáveis inteiras.

(1) Resolva o problema original por um dos algoritmos do simplex.

(2) Se todo x_j, $j \in E$ fôr inteiro, o problema está terminado

(3) Caso contrário calcule para cada variável x_j, $j \in E$ os acréscimos

$$\left.\begin{aligned} D\max_j &= -\min_k \frac{a_{jk}}{Z'_k - c_k} + \sigma \\ D\min_j &= -\max_k \frac{a_{jk}}{Z'_k - c_k} + \sigma \end{aligned}\right\} \quad Z'_k - c_k > 0\,, \quad j \in \text{base}$$

$$\left.\begin{aligned} D\max_j &= \frac{1}{Z'_j - c_j} + \sigma \\ D\min_j &= 0 \end{aligned}\right\} \quad Z'_j - c_j > 0 \qquad j \in \text{base}$$

(4) Penetre o hiperplano $Z' = \sum_1^r c_j x_j$, até o valor inteiro mais próximo, fazendo

$$\Delta = Z'^{,o} - \left[Z'^{,o}\right] \qquad \text{se} \quad Z'^{,o} > 0$$

$$\Delta = Z'^{,o} - \left[Z'^{,o}\right] + 1 \qquad \text{se} \quad Z'^{,o} < 0$$

(5) Determine os valores limites de cada variável $j \in E$

$$\max x_j = a_{jo} + D \max_j \Delta$$

$$\min x_j = a_{jo} + D \min_j \Delta \qquad j = 1, \ 2 \ldots r$$

$$h_j = [\max x_j] - [\min x_j] \quad \text{se } \min x_j \neq [\min x_j]$$

$$h_j = [\max x_j] - [\min x_j] + 1 \text{ se } \min x_j = [\min x_j]$$

(6) Se todos $h_j > 0$, existem pontos inteiros incluídos.

(7) Calcule $SUP\ (0, r) = Z'$

$$SUP\ (0, j-1) = SUP(0, j) - C_j\ [\max_j] \text{ se } C_j \leqslant 0$$

$$SUP\ (0, j-1) = SUP(0, j) - C_j([\max_j] - h_j + 1) \text{ se } C_j > 0$$

(8) Verifique, por enumeração implícita, se existe ponto inteiro sôbre Z'. O procedimento é o mesmo indicado no diagrama de bloco das figs. 3.2 e 3.3, limitando o índice i ao valor zero, isto é, fazendo-se $m = 0$.

(9) Achada uma solução parcial $x_1^*, x_2^* \ldots x_r^*$ que está sôbre Z', substituir êstes valores nas equações do problema original e resolver o problema de programação linear ordinária com (n-r) variáveis:

$$Z\ (max) = Z'^{*} + \sum_{r+1}^{r} c_j\ x_j$$

sujeito a:

$$\sum_{r+1}^{n} a_{ij}\ x_j = a_{io} - \sum_{r+1}^{r} a_{ij}\ x_j^{*} \qquad i = 1,\ 2 \ldots n$$

$$x_j \geq 0 \qquad j = r+1,\ \ldots\ n$$

Exemplo de aplicação é apresentado no apêndice I, problemas A-4 e A-5.

C A P Í T U L O IV

APERFEIÇOAMENTO DOS ALGORITMOS

4.0 - Considerações Iniciais

A experiência que tivemos da utilização de nossos algoritmos I e II, embora pequena, indicou-nos a possibilidade de alguns refinamentos que tornam mais rápida a convergência. Discutiremos também neste capítulo o caso em que o problema original apresenta solução múltipla.

4.1 - Penetração Inicial

Como sabemos, a solução paramétrica nos fornece as expressões:

$$\max x_j = a_{jo} + D \max_j \Delta$$
$$\min x_j = a_{jo} + D \min_j \Delta \qquad (4.1)$$

Em nossos algoritmos temos feito a penetração inicial Δ igual à parte fracionária de Z ou Z', conforme o caso, e depois vamos aumentando Δ de um em um até achar uma solução inteira.

Na expressão (4.1) podemos determinar os valores de Δ que correspondem ao primeiro valor inteiro encontrado em $\max x_j$ e $\min x_j$. Com efeito, êste primeiro inteiro só pode ser $[a_{jo}] + 1$ ou $[a_{jo}]$ conforme $D > 0$ ou $D < 0$.

Fazendo $a_{jo} = [a_{jo}] + r_j$ obtemos:

para $\max x_j$ ser inteiro:

$$\left.\begin{array}{ll} \Delta \text{ int } \max_j = \dfrac{1 - r_j}{D \max_j} & \text{se } D \max_j > 0 \\ \Delta \text{ int } \max_j = \dfrac{- r_j}{D \max_j} & \text{se } D \max_j < 0 \\ \Delta \text{ int } \max_j = 0 & \text{se } r_j = 0 \end{array}\right\} \qquad (4.2)$$

e expressão inteiramente análoga para $\min x_j$.

Portanto a penetração inicial que dá inteiro possível em tôdas as variáveis será:

$$\Delta_i = \max_j \left\{ \min \left(\Delta \text{ int } \max_j, \Delta \text{ int } \min_j \right) \right\} \qquad (4.3)$$

Lembrando que êste valor deve ser ajustado para dar um valor inteiro a Z ou Z', tomaremos:

$$\Delta \text{ inicial} = \left[\Delta_i - r_z \right] + r_z + 1 \qquad (4.4)$$

Nos exemplos A-3 e A-5 do apêndice I teremos ocasião de aplicar esta idéia.

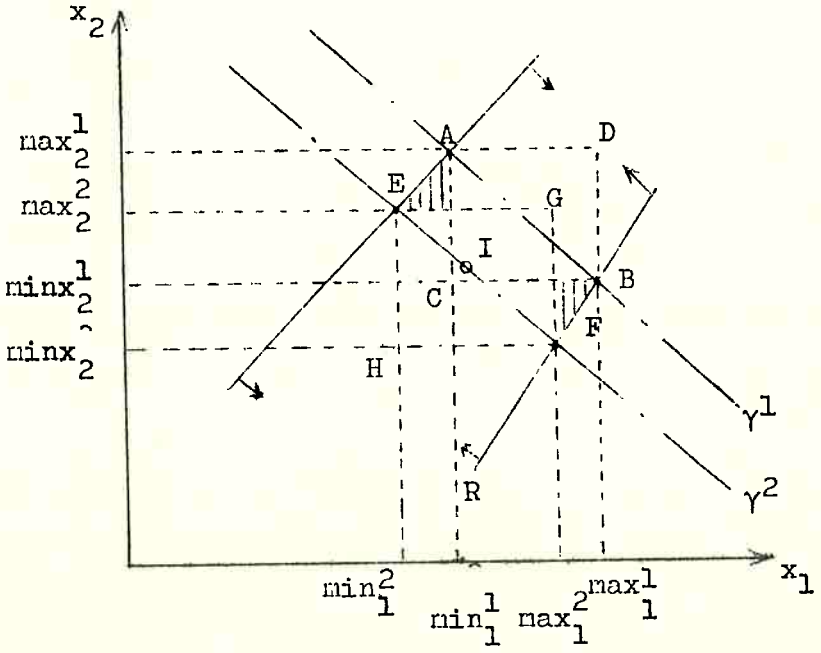
4.2 - Penetração Acelerada

Ocorre muitas vêzes que ao testar os primeiros pontos inteiros obtemos um valor $Z = \gamma^2$ muito inferior ao valor corrente γ^1, indicando um inteiro muito abaixo do hiperplano.

Como a busca é feita dentro do hiperparalelepípedo definido pelos máximos e mínimos, êste fato é uma indicação que há grande probabilidade da solução inteira corresponder a êste valor γ^2.

Consideremos, para visualizar melhor o problema um exemplo em duas dimensões, cuja região viável está indicada na figura 4.1.

Fig. 4.1



O hiperplano γ^1 provocou uma busca na região ABCD que indicou um ponto inteiro em I.

Passando para γ^2, teremos que procurar os inteiros dentro do retângulo EFGH. Ora, ficariam de fora as áreas hachuradas, onde poderia haver pontos inteiros.

Como a variação de max_j^1 para max_j^2 é ao longo de uma aresta, podemos ter a certeza de que incluíremos todos os pontos in-

teiros dentro do poliedro viável entre os planos γ^1 e γ^2, penetrando diretamente até γ^2, desde que tomemos como limites:

$$\max x_j = \max (\max_j^1, \max_j^2)$$
$$\min x_j = \min (\min x_j^1, \min x_j^2) \qquad (4.5)$$

Se encontrarmos um ponto inteiro que dê $Z = \gamma$, $\gamma^2 < \gamma < \gamma^1$, retrocedemos até o nível $Z = \gamma$, e verificamos se êste ponto está sôbre o hiperplano γ. O exemplo A-5 do apêndice ilustra a aplicação da penetração acelerada.

4.3 - Modificação da ordem das restrições

As restrições cuja variável residual aparece na base ótima do simplex têm grande probabilidade de serem satisfeitas por todos os pontos inteiros examinados. A enumeração implícita será, portanto, mais rápida se testarmos em primeiro lugar as restrições cuja variável residual não está na base ótima sem restrição de inteireza. Isto é obtido fàcilmente renumerando o índice i das equações mantendo i = 0 para a função-objetivo e colocando em primeiro lugar as restrições que não tinham folga na solução X^o, e em último as com folga, na ordem crescente de folgas.

4.4 - Problemas com Solução Múltipla Inicial

Se a solução inicial do problema é múltipla, devemos testar inicialmente se há inteiro incluído e verificar, pela combinação linear convexa ou pela enumeração implícita se há solução com êste valor de $Z = \gamma^o$.

Evidentemente, se γ^o não fôr inteiro podemos fazer imediatamente a penetração do hiperplano até o inteiro mais próximo.

Entretanto, aparecem algumas dificuldades:

a) havendo alguns $Z_j - c_j = 0$, fora da base, não podemos aplicar diretamente as nossas fórmulas (3.3) a (3.6) para o cálculo das $D\ max_j$ e $D\ min_j$.

b) havendo várias soluções, teremos várias tabelas de solução ótima a considerar.

Se fizermos tôdas as tabelas de transformação de base possíveis, compatíveis com a condição $Z = \gamma^o - \Delta$, podemos tirar destas tabelas os valores $D\ max_j$ e $D\ min_j$, mas é um processo trabalhoso.

Ora, sabemos que, quando há solução múltipla é porque, num sub-espaço das soluções o hiperplano da função-objetivo é paralelo ao hiperplano de uma das restrições. Êste hiperplano é fàcilmente identificável, porque a sua variável residual é necessàriamente nula em tôdas as soluções múltiplas, e há proporcionalidade entre pelo menos dois dos coeficientes, isto é

$$\frac{a_{ij}}{c_j} = \frac{a_{ik}}{c_k} \qquad (4.6)$$

para, pelo menos um par (j, k).

Se eliminarmos temporàriamente estas restrições e resolvermos o problema, teremos uma solução única, que embora não satisfazendo as restrições eliminadas, nos permite escrever a equação do hiperplano Z para a penetração sob a forma

$$-\Delta = \Sigma\ (Z_j - c_j)\ (-x_j) + x_{n+1} \qquad (4.7)$$

onde x_{n+1} é a residual correspondente.

Recolocando as restrições que haviam sido retiradas, e resolvendo pelo dual-simplex teremos, na coluna correspondente a x_{n+1} os valores dos acréscimos conforme foi exposto na página 28, equação (2.8), donde podemos calcular os D max_j e D min_j e prosseguir normalmente no uso do algoritmo.

4.5 - Outros Aperfeiçoamentos

Na solução pelo algoritmo I acontece, às vêzes que a solução inteira é determinada imediatamente, pois está num dos pontos extremos. Na resolução pelo algoritmo II tal fato é ignorado e a solução inteira é pesquisada dentre as várias combinações possíveis.

Além disto, pode ocorrer, também que na mesma solução básica X_k do algoritmo I tenhamos associado max x_j e min x_i. Usando esta informação poderíamos eliminar o exame de várias combinações inteiras.

Estas considerações nos sugerem uma combinação dos algoritmos I e II, usando o primeiro como indicador para eliminar uma parte da árvore. Estamos pesquisando esta possibilidade, mas ainda não temos resultados concretos.

Outra idéia interessante que estamos estudando é eliminar a reconsideração dos caminhos já pesquisados numa penetração anterior que não resultou em solução inteira. Com efeito, a interseção das árvores de penetrações concecutivas em geral não é vazia, e pelo contrário, contém um número grande de elementos que se não forem mais considerados resultarão numa convergência rápida para a solução.

SUMÁRIO E CONCLUSÕES

Nesta tese são apresentados dois algoritmos para solução de problemas de programação linear com variáveis inteiras, total ou parcial, baseados no método da penetração do hiperplano da função objetivo.

O algoritmo I emprega uma combinação linear convexa de soluções básicas múltiplas, enquanto que o algoritmo II utiliza um processo de enumeração implícita.

A avaliação da eficiência de um algoritmo só pode ser feita através da sua utilização. A experiência na solução manual de problemas pequenos foi bastante promissora, parecendo indicar que os algoritmos propostos comparam favoràvelmente com os apresentados por outros autores.

Infelizmente não foi ainda possível testar a resolução de problemas grandes em computadores de alta velocidade, para efeito de comparação.

Apresentamos um programa inicial, não otimizado em linguagem FORTRAN II, utilizado em alguns testes no computador IBM 1620.

A P Ê N D I C E I

ALGUNS PROBLEMAS RESOLVIDOS

A-1 - Exemplo de Balinsky (1965)

$$Z\ (max) = -\ 4\ x_1 - 5\ x_2$$

sujeito a:

$$-\ 3\ x_1 - x_2 + x_3 = -\ 2$$

$$-\ x_1 - 4\ x_2 + x_4 = -\ 5$$

$$-\ 3\ x_1 - 2\ x_2 + x_5 = -7$$

$$x_j \geqslant 0 \quad \text{inteiro}$$

A tabela final da solução ótima não inteira é:

Base	P_0	P_1	P_2	P_3	P_4	P_5
x_3	42/10	0	0	1	3/10	-11/10
x_2	8/10	0	1	0	-3/10	1/10
x_1	18/10	1	0	0	7/10	11/10
$Z_j - c_j$	-12/10	0	0	0	7/10	11/10

Algoritmo I :

Introduzimos a restrição

$$x_6 = - \Delta - 7/10\ x_4 - 11/10\ x_5$$

e temos as soluções:

X_I	X_{II}
$x_1 = 18/10 - 2/7\ \Delta$	$x_1 = 18/10 + 4/11\ \Delta$
$x_2 = 8/10 + 3/7\ \Delta$	$x_2 = 8/10 - 1/10\ \Delta$
$x_3 = 42/10 - 3/7\ \Delta$	$x_3 = 42/10 + \Delta$
$x_4 = 0 + 10/7\ \Delta$	$x_4 = 0$
$x_5 = 0$	$x_5 = 0 + 10/11\ \Delta$

Primeira penetração: $\Delta = 8/10$, $Z = - 12$

	X_I	X_{II}	inteiro?
$x_1 =$	11/7	23/11	sim
$x_2 =$	8/7	8/11	sim
$x_3 =$	27/7	5	sim
$x_4 =$	8/7	0	sim
$x_5 =$	0	8/11	sim

Não existe, porém, combinação linear de X_1 e X_2 que dê inteiro, pois $\alpha_I = 7/8$ ou 0 e α_{II} só pode ser 0, devido a x_5

Segunda penetração: $\Delta = 18/10$, $Z = -13$

	X_I	X_{II}	inteiro?
x_1	9/7	27/11	sim
x_2	11/7	7/11	sim
x_3	24/7	6	sim
x_4 =	18/7	0	sim
x_5 =	0	18/11	sim

$\alpha_I = 7/18, 14/18, 0$

$\alpha_{II} = 11/18, 0$

A combinação $\alpha_I = 7/18$, $\alpha_{II} = 11/18$ tem $\alpha_I + \alpha_{II} = 1$ e dá $x_1 = 2$, $x_2 = 1$, $x_3 = 5$, $x_4 = 1$, $x_5 = 5$

Pelo algoritmo II:

$D\ max_1 = 4/11$ $\quad$ $D\ min_1 = -2/7$

$D\ max_2 = 3/7$ $\quad$ $D\ min = -1/11$

Primeira penetração:

$\Delta = 8/10$ $\quad$ $Z = -12$

$max_1 = 23/11$ $\quad min_1 = 11/7$ $\quad h_1 = 1$ $\quad [max_1] = 2$

$max_2 = 8/7$ $\quad min_2 = 8/11$ $\quad h_2 = 1$ $\quad [max_2] = 1$

Teste: $Z = -4.2 - 5.1 = -13$

Logo penetramos para $Z = -13$, $\Delta = 18/10$

$max_1 = 27/11$ $\quad min_1 = 9/7$ $\quad h_1 = 1$ $\quad [max_1] = 2$

$max_2 = 11/7$ $\quad min_2 = 7/11$ $\quad h_2 = 1$ $\quad |max_2| = 1$

Teste: $Z = -4.2 - 5.1 = -13$

$-3.2 - 1.1 = -7 \leqslant -2$

$-1.2 - 4.1 = -6 \leqslant -5$

$-3.1 - 2.1 = -8 \leqslant -7$

Solução ótima $x_1 = 2$, $x_2 = 1$

Balinsky mostra a solução por diversos algoritmos. O algoritmo I de Gomory exigiu a introdução de dois cortes, num total de cinco tabelas. O algoritmo II de Gomory chega à solução com 3 tabelas apenas.

O de Land e Doig conduziu ao exame dos seguintes pontos da árvore:

vértice 0:(-112/10, 18/10, 8/10, 42/10, 0, 0) - solução inicial

vértice 1:(- 14, 1, 2, 3, 4, 2)

vértice 2:(-47/4, 2, 3/4, 19/4, 3/4, 1/2)

vértice 3:(- ∞, ...)

vértice 4:(-13, 2, 1, 5, 1, 1)

vértice 5:(-29/2, 3, 1/2, 15/2, 0, 3)

A solução é o vértice 4.

O nosso algoritmo exigiu 3 tabelas para chegar à solução - ótima não inteira, como Gomory I e Land e Doig, e depois 2 penetrações.

A - 2 - Exemplo de Balinsky (1965)

Mesmo problema anterior, com a restrição que apenas x_1 e Z devem ser inteiros.

Como Z é inteiro, a penetração é feita pelo próprio Z, como no problema A-1. A única diferença é que devemos pesquisar apenas as combinações lineares que tornam x_1 inteiro. Na primeira penetração:
$\Delta = 8/10 \quad Z = -12$

Inteiro $x_1 = 11/7\ \alpha + 23/11\ (1 - \alpha)$ que dá a solução:

$\alpha_I = 7/40$, $(1 - \alpha) = 33/40$ e

$x_1 = 2$, $x_2 = 4/5$, $x_3 = 24/5$ $x_4 = 1/5$, $x_5 = 3/5$ $Z=-12$

A solução pelo algoritmo III de Gomory exigiu 3 cortes, com 6 tabelas no total.

A restrição Z = inteiro é exigida para garantir a convergência do algoritmo III de Gomory. Vamos levantar esta restrição, especificando que apenas x_1 deve ser inteiro.

Neste caso a penetração é feita com $Z' = - x_1$ e a restrição correspondente é:

$$x_6' = - \Delta + 2/10 \; x_4 - 4/10 \; x_5$$

Sòmente pode entrar na base x_5, pelo critério do dual-simplex, e temos uma única solução:

$$X_I$$

$$x_1 = 18/10 + \Delta$$

$$x_2 = 8/10 - 1/4 \; \Delta$$

$$x_3 = 42/10 + 11/4 \; \Delta$$

$$x_4 = 0$$

$$x_5 = 10/4 \; \Delta$$

$$Z = - 112/10 - 11/4 \; \Delta$$

Fazendo

$\Delta = 2/10$, temos a solução procurada $x_1 = 2$, $x_2 = 3/4$, $x_3 = 19/4$, $x_4 = 0$, $x_5 = 1/2$, $Z = - 47/4$

A-3 - Exemplo de Balas (1965) - Variáveis bi-valentes

$$Z\ (\min) = 5\,x_1 + 7\,x_2 + 10\,x_3 + 3\,x_4 + x_5$$

$$x_1 - 3\,x_2 + 5\,x_3 + x_4 - 4\,x_5 \geqslant 2$$

$$2\,x_1 + 6\,x_2 - 3\,x_3 - 2x_4 + 2\,x_5 \geqslant 0$$

$$- x_2 + 2\,x_3 - x_4 - x_5 \geqslant 1$$

$$x_j = 0 \text{ ou } 1$$

Colocando sob forma padrão e resolvendo pelo simplex, temos a solução ótima não inteira:

Base	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
x_2	1/3	-4/9	1	0	-7/9	1/9	0	-3/9	-2/9
x_3	2/3	-2/9	0	1	-8/9	-4/9	0	-6/9	-1/9
x_6	1/3	-7/9	0	0	-28/9	13/9	1	-21/9	1/9
$Z_j - c_j$	-9	93/9	0	0	156/9	42/9	0	81/9	24/9

Usaremos o algoritmo II, com penetração inicial rápida:

$max_1 = 4/93\ \Delta$ $\quad min_1 = 0$ $\quad \Delta_1 = min(93/4,\ 0) = 0$

$max_2 = 1/3 + 1/12\ \Delta$ $\quad min_2 = 1/3 - 1/42\ \Delta$ $\quad \Delta_2 = min(8,\ 14) = 8$

$max_3 = 2/3 + 4/42\ \Delta$ $\quad min_3 = 2/3 + 2/93\ \Delta$ $\quad \Delta_3 = min(14/4,\ 31/2) = 14/4$

$max_4 = 9/156\ \Delta$ $\quad min_4 = 0$ $\quad \Delta_4 = min(156/9, 0) = 0$

$max_5 \quad 9/42\ \Delta$ $\quad min_5 = 0$ $\quad \Delta_5 = min(42/9,\ 0) = 0$

Δ inicial = max $\Delta_i = 8$, pois Z^o é inteiro, e temos:

$\Delta = 8$, $\quad Z = -17$

max_1	=	0,344	min_1	= 0	h_1	= 1	$[max_1] = 0$
max_2	=	1,000	min_2	= 0,143	h_2	= 1	$[max_2] = 1$
max_3	=	1,429	min_3	= 0,839	h_3	= 1	$[max_3] = 1$
max_4	=	0,461	min_4	= 0	h_4	= 1	$[max_4] = 0$
max_5	=	1,714	min_5	= 0	h_5	= 2	$[max_5] = 1$

1o. teste: $x_1 = 0,\ x_2 = 1,\ x_3 = 1,\ x_4 = 0,\ x_5 = 1$

Verificação:

$Z = -5.0 - 7.1 - 10.1 - 3.0 - 1.1 = -18$ não serve

2o. teste: $x_1 = 0,\ x_2 = 1\ x_3 = 1,\ x_4 = 0,\ x_5 = 0$

Verificação:

$Z = -5.0 - 7.1 - 10.1 - 3.0 - 1.0 = -17$

$$-3 + 5 = 2 \geq 2$$

$$6 - 3 = 3 \geq 0$$

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

logo é a solução procurada.

Além de Balas (1965), o mesmo problema é apresentado por Roy e outros em (1965).

A-4 - Exemplo de Harris (1964)

Trata-se do problema parcial ou misto:

$$Z\ (\max) = 11\,x_1 + 2\,x_2 + 6\,x_3 + x_4$$

restrito a:

$$6\,x_1 + 4\,x_2 + x_3 - 3\,x_4 + x_5 = 6$$

$$x_1 + 3\,x_2 + 2\,x_3 + x_4 + x_6 = 3$$

$$x_j \geq 0 \qquad j = 1, 2 \ldots 6 \qquad x_1 \text{ e } x_2 \text{ inteiros}$$

A solução ótima não inteira é dada pela tabela

Base	P_0	P_1	P_2	P_3	P_4	P_5	P_6
x_1	5/3	1	13/9	7/9	0	1/9	1/3
x_4	4/3	0	14/9	11/9	1	-1/9	2/3
$Z_j - c_j$	59/3	0	139/9	34/9	0	10/9	13/3

A restrição é agora $Z' \leqslant 11\ x_1 + 2\ x_2 = 55/3$, ou em função das variáveis fora da base:

$$x_7 = -\Delta + 125/9\ (-x_2) + 77/9\ (-x_3) + 11/9\ (-x_4) + 11/3(-x_3)$$

donde:

$$max_1 = 5/3 - 1/11\ \Delta \qquad min_1 = 5/3 - 13/125\ \Delta$$

$$max_2 = 13/125\ \Delta \qquad min_2 = 0$$

Penetração inicial rápida $\Delta_i = max\ .\ min\ (22/3,\ 90/13) = 90/13$

$$\Delta \text{ inicial} = [90/13 - 1/3] + 1/3 + 1 = 22/3$$

$$\Delta = 22/3 \qquad Z' = 11$$

$$max_1 = 1 \qquad min_1 = 0{,}904 \qquad h_1 = 1 \qquad [max_1] = 1$$

$$max_2 = 0{,}763 \qquad min_2 = 0 \qquad h_2 = 1 \qquad [max_1] = 0$$

Como $Z' = 11.1 + 2.0 = 11$, o ponto está sôbre o hiperplano.

Resolvemos agora o problema de programação linear sem restrição inteira:

$$Z\ (max) = 11 + 6\ x_3 + x_4$$

sujeito a:

$$x_3 - 3\ x_4 + x_5 = 0$$

$$2\ x_3 + x_4 + x_6 = 2$$

$$x_j \geqslant 0$$

cuja solução é:

$$x_3 = 6/7 \qquad x_4 = 2/7 \qquad Z = 11 + 38/7$$

donde a solução completa do problema:

$$x_1 = 1, \quad x_2 = 0, \quad x_3 = 6/7, \quad x_4 = 2/7, \quad Z = 115/7$$

A solução de Harris exigiu 9 tabelas de simplex, enquento que a nossa necessitou 6 para resolver os dóis problemas de programação linear.

A-5 - Exemplo de Land e Doig (1960)

$$Z\ (\max) = 779\ x_1 + 768\ x_2 + 896\ x_3 + 971\ x_4 + 313\ x_5$$

sujeito a:

$$10{,}9\ x_1 + 3{,}6\ x_2 - 40{,}8\ x_3 + 43{,}9\ x_4 + 7{,}1\ x_5 + x_6 = 82{,}3$$

$$-86{,}8\ x_1 + 32{,}7\ x_2 + 24{,}3\ x_3 + 13{,}8\ x_4 - 12{,}6\ x_5 + x_7 = 77{,}3$$

$$60{,}9\ x_1 + 68{,}9\ x_2 + 69{,}0\ x_3 - 56{,}9\ x_4 + 22{,}5\ x_5 = 86{,}5$$

$$x_1, x_2, x_3 \geq 0 \text{ inteiras} \quad x_4, x_5, x_6, x_7 \geq 0$$

A penetração será feita com

$$Z' = 779\ x_1 + 768\ x_2 + 896\ x_3$$

A tabela da solução ótima não inteira é, incluindo já a linha $Z'_j - c_j$:

Base	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7
x_1	1,4960	1	0,5271	0	0	0,2389	0,0125	0,0038
x_4	6,1697	0	1,9260	0	1	0,4615	0,0401	0,0174
x_3	5,0210	0	2,1256	1	0	0,4124	0,0220	0,0177
$Z'_j - c_j$	5664,2	0	1547,15	0	0	555,61	29,45	12,90

Temos:

$\max_1 = 1,4960 + 0,000424\ \Delta$ $\quad \min_1 = 1,4960 - 0,000\ 430\ \Delta$

$\max_2 = 0,000646\ \Delta$ $\quad \min_2 = 0$

$\max_3 = 5,0210 - 0,000747\ \Delta$ $\quad \min_3 = 5,0210 - 0,001374\ \Delta$

$\Delta_i = \max\ (\min\ (1153,5,\ 1187?4);\ \min(1540,\ 0);\ \min(28,1,\ 15,3)\ ).$

Δ inicial = 1154,2 com $Z' = 4510$

$\max_1 = 1,985$ $\quad \min_1 = 1,000$ $\quad h_1 = 1$

$\max_2 = 0,746$ $\quad \min_2 = 0$ $\quad h_2 = 1$

$\max_3 = 4,159$ $\quad \min_3 = 3,435$ $\quad h_3 = 1$

A solução inteira $x_1 = 1$, $x_2 = 1$, $x_3 = 1$ dá $Z' = 4363$

Façamos então a penetração rápida para êste nível

$\Delta = 1301,2$ $\qquad Z = 4363$

$\max_1 = \max(1,985;\ 2,048) = 2,048$ $\quad \min_1 = \min(1,000;\ 0,9365) = 1,000$

$\max_2 = \max(0,746;\ 0,841) = 0,841$ $\quad \min_2 = 0$

$\max_3 = \max(4,159;\ 4,049) = 4,159$ $\quad \min_3 = \min(3,435;\ 3,323) = 3,435$

Os pontos inteiros incluidos são:

$$x_1 = 2, \quad x_2 = 0, \quad x_3 = 4$$

e $\quad x_1 = 1, \quad x_2 = 0 \quad x_3 = 4$

O primeiro dá $Z' = 5142 > 4510 = \gamma^1$ rejeitado. Como o segundo é o que nos levou à penetração até $Z' = 4363 = \gamma^2$ êle é aceito, pois já sabemos que está sôbre o hiperplano γ^2.

Substituindo no problema original temos o novo problema sem restrição inteira:

$$Z\ (\max) = 4363 + 971\ x_4 + 313\ x_5$$

sujeito a:

$$43,9\ x_4 + 7,1\ x_5 + x_6 = 234,6$$

$$13,8\ x_4 - 12,6\ x_5 + x_7 = 66,9$$

$$56,9\ x_4 - 22,5\ x_5 = 250,4$$

$$x_j \geqslant 0$$

cuja a solução completará a solução do problema misto.

APÊNDICE II - PROGRAMA PARA COMPUTADOR

```
C       PROGRAMACAO LINEAR DE INTEIROS
C       ------ -
C       PROGRAMA 1 - INICIALIZACAO
C                                        C.C.E. - U.S.P.  REINHARD
C
C       DEFINICAO DE AREAS
        DIMENSION NBAS(8),NNBAS(20),NX(8),XMAX(8),XMIN(8),TESTX(8),
       1A(11,08),D(10),ZC(20),DMAX(8),DMIN(8)
        COMMON NEQ,NR,Z,FRACZ,NX,XMAX,XMIN,D,DMAX,DMIN,A,NEQM1,NOSOL
C       LEITURA DAS MATRIZES
   10   READ 1,NEQ,NTVAR,Z,EPSIL
    1   FORMAT(2I4,2E14.7)
        READ 2,((A(I,J),I=1,NEQ),J=1,NTVAR)
        READ 2,(D(J),J=1,NEQ)
        READ 2,(ZC(I),I=1,NTVAR)
        READ 4,(NBAS(J),J=1,NEQ)
        READ 4,(NNBAS(I),I=1,NTVAR)
        READ 1,NR
        READ 4,(NX(I),I=1,NR)
    2   FORMAT(5E14.7)
    4   FORMAT (I4)
        NEQM1=NEQ+1
        NOSOL=1
C       TESTE SOLUCAO UNICA
        DO 11 J=1,NTVAR
        IF(ZC(J))12,13,11
C       SOLUCAO NAO OTIMA
   12   STOP 100
C       SOLUCAO MULTIPLA
   13   CALL LINK (PROLI3)
   11   CONTINUE
C       TESTE SOLUCAO INTEIRA
        DO 120 I=1,NR
        KKK=NX(I)
        DO 121 J=1,NEQ
        IF(NBAS(J)-KKK)121,122,121
  121   CONTINUE
  120   CONTINUE
        GO TO 200
  122   IF(D(J)-DRH(D(J)))130,120,130
C       CALCULO DAS VARIACOES LIMITES
  130   DO 14 J=1,NR
        DO 137 JJ=1,NEQ
        IF(NBAS(JJ)-NX(J))137,138,137
  137   CONTINUE
        XMAX(J)=0.
```

```
      XMIN(J)=0.
      GO TO 136
  138 XMAX(J)=D(JJ)
      XMIN(J)=D(JJ)
C     VERIFICACAO J PERTENCE A BASE
  136 DO 131 JJ=1,NEQ
      IF(NBAS(JJ)-NX(J))131,132,131
  131 CONTINUE
      GO TO 133
C     VARIAVEL J E BASICA
C     CALCULO DO MAXIMO
  132 DMAX(J)=1.0E+98
      DO 17 K=1,NTVAR
      VAR=A(JJ,K)/ZC(K)
      IF(DMAX(J)-VAR)17,17,18
   18 DMAX(J)=VAR
   17 CONTINUE
      DMAX(J)=-DMAX(J)
C     CALCULO DO MINIMO
      DMIN(J)=-1.0E+98
      DO 20 K=1,NTVAR
      VAR=A(JJ,K)/ZC(K)
      IF(DMIN(J)-VAR)21,20,20
   21 DMIN(J)=VAR
   20 CONTINUE
      DMIN(J)=-DMIN(J)
      GO TO 180
C     VARIAVEL NAO BASICA
C     PESQUISA DO INDICE NA MATRIZ
  133 DO 134 JJ=1,NTVAR
      IF(NNBAS(JJ)-NX(J))134,135,134
  134 CONTINUE
      STOP
  135 DMAX(J)=1./ZC(JJ)
      DMIN(J)=0.
C     AJUSTE ERRO ARREDOND.
  180 IF(DMAX(J))171,171,172
  172 DMAX(J)=DMAX(J)+EPSIL
  171 IF(DMIN(J))201,14,14
  201 DMIN(J)=DMIN(J)-EPSIL
   14 CONTINUE
C     TESTE DE Z INTEIRO
      IF(Z-DRH(Z))22,23,22
   22 IF(Z)25,25,24
   23 FRACZ=1.
```

```
      GO TO 26
   24 FRACZ=Z-DRH(Z)
      GO TO 26
   25 FRACZ=Z-DRH(Z)+1.
   26 READ 5,I,J,XIJ
    5 FORMAT(2I4,F10.5)
      IF(I)263,264,263
  263 IF(J)261,262,261
  261 A(I,J)=XIJ
      GO TO 26
  262 D(I)=XIJ
      GO TO 26
  264 CALL LINK(PROLI2)
  200 PUNCH 400,NOSOL,Z
  400 FORMAT(11HSOLUCAO NO,I3,17HFUNCAO OBJETIVO =,E14.7,/)
      NOSOL=NOSOL+1
      PUNCH 500
  500 FORMAT(8HVARIAVEL,5X,5HVALOR)
      PUNCH 300,(NX(I),D(I),I=1,NR)
  300 FORMAT(2X,I5,E14.7)
      STOP
      END
```

```
C     PROGRAMACAO LINEAR DE INTEIROS
C     --------
C     PROGRAMA 2 - PROCESSO DE PESQUISA NA ARVORE
C
C     DEFINICAO DE AREAS
      DIMENSION NX(8),XMAX(3),XMIN(8),A(11,8),D(10),DMAX(8),
     1DMIN(8),SIGMA(11,8),H(8),X(8),SUP(11,8),TESTX(8)
      COMMON NEQ,NR,Z,FRACZ,NX,XMAX,XMIN,D,DMAX,DMIN,A,NEQM1,NOSOL
      EQUIVALENCE (XMAX(1),TESTX(1))
C     NOVO VALOR FUNCAO OBJETIVO
   27 Z=Z-FRACZ
C     CALCULO DOS VALORES LIMITES DAS VARIAVEIS
      IF(NOSOL-1)271,271,80
  271 DO 33 J=1,NR
      XMAX(J)=XMAX(J)+DMAX(J)*FRACZ
      XMIN(J)=XMIN(J)+DMIN(J)*FRACZ
      IF(XMIN(J))29,30,30
   29 XMIN(J)=0.
```

```
   30 IF(XMIN(J)-DRH(XMIN(J)))31,32,31
C     NUMERO DE INTEIROS CONTIDOS NO INTERVALO
   31 H(J)=DRH(XMAX(J))-DRH(XMIN(J))
      GO TO 33
   32 H(J)=DRH(XMAX(J))-DRH(XMIN(J))+1.
C     VALORES DE PARTIDA
   33 TESTX(J)=DRH(XMAX(J))
C     VERIFICAR EXISTENCIA DE INTEIROS
      DO 35 J=1,NR
      IF(H(J))34,34,35
   34 FRACZ=1.
      GO TO 27
   35 CONTINUE
      SUP(1,NR+1)=Z
C     CALCULO DOS SUPREMOS PARA O ULTIMO NIVEL
      DO 36 I=2,NEQM1
   36 SUP(I,NR+1)=D(I)
      J=NR
C     SAIDA INTERMEDIARIA OPCIONAL
C     INDICA A DIMENSAO DA ARVORE
      IF(SENSE SWITCH 1)361,40
  361 PUNCH 300, NOSOL,Z
      PUNCH 300,(NX(I),H(I),I=1,NR)
C     CALCULO DOS SUPREMOS PARA OS NIVEIS INTERMEDIARIOS
   40 DO 37 I=1,NEQM1
      IF(A(I,J))38,38,39
   39 SUP(I,J)=SUP(I,J+1)-A(I,J)*(TESTX(J)-H(J)+1.)
      GO TO 37
   38 SUP(I,J)=SUP(I,J+1)-A(I,J)*TESTX(J)
   37 CONTINUE
      J=J-1
      IF(J-1)41,41,40
C     SOMATORIAS NULAS - INICIO
   41 DO 50 I=1,NEQM1
   50 SIGMA(I,1)=0.
C     TESTE DA ARVORE
      DO 601J=1,NR
  601 X(J)=TESTX(J)
      DO 70 I=1,NEQM1
      DO 60 J=1,NR
      SIGMA(I,J+1)=SIGMA(I,J)+A(I,J)*X(J)
   51 IF(SIGMA(I,J+1)-SUP(I,J+1))60,60,52
   60 CONTINUE
      IF(I-1)70,53,70
   70 CONTINUE
```

```
      GO TO 200
   52 IF(A(I,J))54,54,55
   54 X(J)=TESTX(J)
      J=J-1
      IF(J)56,56,52
   56 FRACZ=1.
      GO TO 27
   55 X(J)=X(J)-1.
      IF(X(J)-TESTX(J)+H(J))54,54,58
C     RECOMECA A VERIFICACAO
   58 I=1
      SIGMA(I,J+1)=SIGMA(I,J+1)-A(I,J)
      GO TO 51
C     VERIFICACAO SE O PONTO PERTENCE AO PLANO DE CORTE
   53 IF (SIGMA(1,NR+1)-Z)62,70,62
   61 IF(A(I,J))72,71,71
   71 X(J)=TESTX(J)
      J=J-1
      IF(J)73,73,61
   73 FRACZ=1.
      GO TO 27
   62 J=J-1
      GO TO 61
   72 X(J)=X(J)-1.
      IF(X(J)-TESTX(J)+H(J))71,71,74
   74 SIGMA(1,J+1)=SIGMA(1,J+1)+A(1,J)
      GO TO 51
C     SAIDA DA SOLUCAO
  200 PUNCH 400,NOSOL,Z
  400 FORMAT(11HSOLUCAO NO,I3,3X,17HFUNCAO OBJETIVO =,E14.7,/)
      NOSOL=NOSOL+1
      PUNCH 500
  500 FORMAT(8HVARIAVEL,5X,5HVALOR)
      PUNCH 300,(NX(I),X(I),I=1,NR)
  300 FORMAT(2X,I5,E14.7)
C    PESQUISA DE OUTRAS SOLUCOES
      J=NR+1
      I=NEQ+1
   75 X(J)=TESTX(J)
      J=J-1
      IF(J)80,80,76
   76 X(J)=X(J)-1.
      IF(X(J)-TESTX(J)+H(J))75,75,92
   92 J1=J+1
      GO TO 58
C     PARADA FINAL
   80 STOP 999
      END
ZZZZZ
```

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

AGIN, N., (1966) - Optimun Seeking with Branch and Bound in Management Science, v.13,n.4, p. B/76 B/85.

ASSIS, S.L.O.(1964) - Contribuição ao estudo do método de percurso crítico. São Paulo, U.S.P., Escola Politécnica - Tese de Livre-Docência.

BALAS, E. (1965) - An Additive Algorithm for Solving Linear Programs with Zero-One Variables, in Operations Research, v.13,n.4, p.517-546.

BALINSKY, M.L.(1961) - Fixed Cost Transportation Problems, in Naval Research Logistics Quarterly, v.8,n.1,p.41-54.

_____ (1965) - Integer Programming - Methods, Uses, Computations, in Management Science, v.12, n.3, p.253-313.

_____ (1966) - On Finding Integer Solutions to Linear Programs, in Proceeding of the IBM Scientific Conjunting Symposium on Combinatorial Problems - March 16 18, 1964. IBM Data Processing Division, White Plains, New York.

BALINSKY, M.L. e R.E. Quandt, (1964) - On an Integer Program for a Delivery Problem, in Operations Research, v.13,n.6, p. 946-959.

BEALE, E.M.L. (1963) - Two Transportation Problems - Comunicação ao 3o. Congresso Internacional de Pesquisa Operacional, Oslo (1963).

BELLMAN, R. (1957) - Dynamic Programming. Princeton University Press Princeton. N.Y.

BELLMAN, R. e S.E. Dreyfuss (1962) - Applied Dynamic Programming. Princeton University Press, Princeton N.Y.

BENDERS, J.F., R.A. Catchpole e L.C. Kuiken (1959) - Discrete Variable Optimization Problems - Comunicação ao Simpósio de Programação Matemática - Rand Corp. St. Monica.

BOWMAN, E.H. (1959) - The Schedule Sequencing Problem, in Operations Research, v.7, n.5, p.621-624.

———— (1960) - Assembly Line Balancing by Linear Programming, in Operations Research, v.8, n.3, p.385-389.

BOWMAN, E.H. e R.B. Fetter (1957) - Analisys for Produc - tion Management Richard D. Irwin, Inc., Homewood, Ill.

BURT, O. e C.C. Harris Jr. (1963) - Apportionment of the U.S. House of Representative: a Minimum Range, Integer Solution Allocation Problem, in Letter to the Editor, Operations Research, v.11, n.4, p.648-652.

CARR, C.R. e C.W. Howe (1964) - Quantitative Decision Proceduras in Management and Economics - McGraw Hill Book Co, N.Y.

CHARNES, A. e W.W. Cooper (1961) - Management Models and Industrial Applications of Linear Programming - 2 volu mes - J. Wiley and Sons, Inc., New York.

CHARNES, A. e M.H. Miller (1956) - A Model for the Optimal Programming of Railway Freight Train Movements, in Management Science, v.3, n.1, p. 74-92.

CLARKE, G. e J.W. Wright (1964) - Scheduling of Uehicles from a Central Depot to a Number of Delivery Points, in Operations Research, v.12, n.4, p.568-581.

DALTON, R.E. e R.W. Llewellyn (1966) - An Extension of the Gomory Mixed Integer Algorithm to Mixed Discrete Varia bles, in Management Science, v.12, n.7, p.569-575.

DANTZIG, G.B. (1957) - Discrete Variable Extremum Problems in Operations Research, v.5, n.2, p.266-277.

———— (1960a) - On the Significance of Solving Li- near Programming Problems with Some Integer Variables in Econometrica, v.28, n.1, p.30-44.

———— (1960b) - A Machine Job Scheduling Model, in Management Science, v.6, n.2. p.191-196.

DANTZIG, G.B. (1963) - Linear Programming and Extensions. Princeton University Press, Princeton, N.Y.

DANTZIG, G.B., D.R. Fulkerson e S.M. Johnson (1954) - Solution of a Large Scale Travelling Salesman Problem, in Journal of the Operations Research Society of America, v.2, n.4, p.393-410.

_____ (1959) - On a Linear Programming Combinatorial Approach to the Travelling Salesman Problem, in Operations Research, v.1, n.1, p.58-66.

DANTZIG, G.B. e J.H. Ramser (1959) - The Truch Dispatching Problem, in Management Science, v.6, n.1, p.80-91.

DAY, R.H. (1965) - On Optimal Extracting from a Multiple File Data Storage: an Application of Integer Programming, Letter the Editor, in Operations Research, v.13, n.1, p.94-120.

D'ESOPO, D.A. e B. Lefkowitz (1964) - Note on an Integer Linear Programming Model for Determining a Minimum Embarkation Fleet, in Naval Research Logistics Quarterly, v.11, n.1, p.79-82.

DARN, W.S. (1963) - Non Linear Programming a Survey, in Management Science, v.9, n.2, p.171-208.

DRIEBEEK, N.J. (1966) - An Algorithm for the Solution of Mixed Integer Programming Problems, in Management Science, v.12, n.7, p.576-587.

DUBY, J.J. (1966) - An Example d'application de la Methode "Branch and Bound": Determination de Variable de Décision, in Gestion, ano 9, Maio 1966, p.304-381.

EFROYMSON, M.A. e T.L. Ray - A Branch - Bound Algorithm for Plant Location, in Operations Research, v.14, n.3, p. 361-368.

EISEMANN, K. (1957) - The Trim Problem, in Management Science, v.3, n.3, p.279-284.

ELMAYHRABY, S.E. e A.S. Ginsberg (1964) - A Dynamic Model for the Optimal Loading of Linear Multi-Operation Shops, in Management Technology, v.4, n.1. p.47-58.

FADIGAS TORRES, O. (1967) - Programação Linear: Introdução teórica e Método Simplex, Departamento de Eng. de Produção, Escola Politécnica, U.S.P.

FAURE, R. e Y. Malgrange (1963) - Une Méthode Booléiene pour la Resolution des Programmes Linéaires en Nombres Entiers, in Gestion, abril 1963, p.250-260.

---------- (1965) Nouvelles Recherches sur la Resolution des Programmes Linéaires en Nombres Entiers in Gestion, Juin 1965, p.371-375.

FLEISCHMAN, B. (1967) - Computational Experience with the Algorithm of Balas, Letter to the Editor, in Operations Research, v.15, n.1, p.153-155.

FORTET, R. (1960) - Applications de L'algébre de Boole en Recherche Opérationelle, in Revue Française de Recherche Opérationelle, v.4, n.1, p.17-25.

---------- (1961) Les Programmes Lineaires en Nombres Entiers in Huard (1964) p.71-91.

FREEMAN, R. (1966) - Computational Experience with a "balasian" Integer Programming Algorithm, Letter to the Editor, in Operations Research, v.14, n.5, p.935-940.

GASS, S. (1964) - Linear Programming - 2a. Edição - McGraw - Hill Book Co, N.Y.

GILBERT, E. J. e A.J. Schatz (1964) - An ill Conceived Proposal for Apportionment of the U.S. House of Representatives Letter to the Editor, in Operations Research, v.12, n.5, p.768-773.

GILMORE, P.C. e R.E. Gomory (1961) - A Linear Programming Approach to the Cutting Stock Problem, in Operations Research, v.9, n.6, p.849-859.

GILMORE, P.C. e R.E. Gomory (1963) - A Linear Programming Approach to the Cutting Stock Problem, Part II, in Operations Research, v.11, n.6, p.849-859.

———————————— (1965) - Multistage Cutting Stock Problems of Two and More Dimensions, in Operations Research, v.13, n.1, p.94-120.

GIGLIO, R.J. e H.M. Wagner (1964) - Approximate Solutions to the Three Machine Scheduling Problem, in Operations Research, v.12, n.2, p.305-324.

GLOVER, F. (1964) - A Bound Escalation Method for the Solution of Integer Programs, in Cahiers du Centre d'Études de Recherche Operérationelle, Bruxelles, v.6, n.3, p.131 168.

———— (1965a) - A Hybrid Dual Integer Programming Algorithm, in Cahiers du Centre d'Études de Recherche Opérationelle, Bruxelles, v.7, n.1, p.5-23.

———— (1965b) - A Multiphase Dual Algorithm for the Zero-One Integer Programming Problem, in Operation Research v.13, n.6, p.879-919.

———— (1966) - Generalized Cut in Diophantine Programming, in Management Science, v.13, n.3, p.254-268.

GLOVER, F. e S. Zionts (1964) - A Note on the Additive Algorithm of Balas, in Operations Research, v.13, n.4, p.546-549.

GOLOMB, S.W. e L.D. Baumert (1965) - Bactrack Programming in, Journal of the Association for Computing Machinery v.12, n.4, p.516-524.

GOMORY, R.E. (1958) - An Algorithm for Integer Solutions to Linear Programs. Princeton -IBM Mathematics Research Project, Technical Report, n. 1; também em Graves e Wolfe (1963), p.269-302.

———— (1960a)- All Integer Programming Algorithm em Muth e Thompson (1963) p.193-206, (originalmente publicado como IBM Research Center, Report R.C. 189.)

GOMORY, R.E. (1960b) - An Algorithm for the Mixed Integer Problem apud Balinsky (1965) (originalmente publicado como Research Memo 2597 - Rand Corp.

GRAVES, R. L. e P. Wolfe (1963) - Recent Advances Mathematical Programming - McGraw Hill Book Co, N.Y.

HADLEY, G. (1961) - Linear Algebra = Addison Wesley Publ. Co. Reading, Mass.

———— (1962) - Linear Programming - Addison Wesley Publ. Co. Reading Mass.

———— (1964) - Non Linear and Dynamic Programming - Addison Wesley Publ.Co. Reading, Mass.

HALDI, J. (1964) - 25 Integer Programming Test Problems Working Paper n.43, Graduate school of Business - Stanford University.

HALDI, J. e L.M. Isaacson (1965) - A computer code for Integer Solutions of Linear Programs, in Operations Research, v.13, n.6, p.946-959.

HANSMANN, F. (1961) - Operational Research in the National Planning of Underdeveloped Countries, in Operations Research, v. 9, n.1, p.203-248.

HARRIS, P.M.J. (1964) - An Algorithm for solving Integer Linear Programmes, in Operations Research Quarterly v.15, n.2, p.117-132.

HEALY, W.C.Jr. (1964) - Multiple Choice Programming, in Operations Research, v.12, n.1, p.122-138.

HERVÉ, P. (1967) - Resolution des Programmes Linéaires a Variables Mixtes par la Procédure S.E.P., in Metra, v.VI, n.1, p.77-91.

HILLIER, F.S. (1966a) - Efficient Sub-optimal Algorithms for Integer Linear Programming with an Interior. Technical Report, n.2, August 1966, Dept of Industrial Engineering, Stanford University.

———— (1966b) - An Optimal Bound-and-Scan Algorithm for Integer Linear Programming. Tech Report, n.3 August 1966, Dept of Industrial Engineering, Stanford University.

HUARD, P. (1964) - Mathématique des Programmes Économiques Monographies de Recherche Operationelle, n.1, Dunod, Paris.

KAUFMANN, A., M. Denis Pappin e R. Faure (1963) - Cours de Calcul Booléien - Albin Michel, Paris.

KLEIN, M. (1963) - On Assembly Line Balancing, in Operations Research, v.11, n.2, p.274-281.

LAMBERT, F. (1962) - Programmes en Nombres Entiers et Programmes Mixtes, in Metra, v.1, n.1, p.93-112.

LAND, A.H. e A.G. Doig (1960) - An Automatic Method of Solving Discrete Programming Problems, in Econometrica, v.28,n.3 p.497-520.

LAWLER, E.L. e D.E. Wood (1966) - Branch and Bound Methods - A Survey, in Operations Research, v.14, n.4, p.699-719.

LE GARF, A. e Y. Malgrange (1963) - Resolution des Programmes Linéaires a Valeurs Entières por une méthode Booleiene "Compacte", in Proceedings of the 3rd International Conference, in Operations Research, Oslo, 1963, Dunod Paris.

LITTLE, J.D.C. (1966) - The Synchronization of Traffic Signals by Mixed Integer Linear Programming, in Operations Research, v.14, n.4, p.568-594.

LITTLE, J.D.C., K.G. Murt, D.W. Sweeney e C. Karel (1963) - An Algorithm for the Travalling Selesman Problem, in Operations Research, v.11, n.6, p.972-989.

MANNE, A.S. (1960) - On the Job -Shop Scheduling Problem,in Operations Research, v.8, n.2, p.219-223.

———— (1964) - Plant Location Under Economies of Scale Decentralization and Computation, in Management Science, v.11, n.2, p.213-235.

MARTIN, G.T. (1963) - An Accelerated Euclidean Algorithm for Integer Linear Programming, p.311-318 de Groves e Wolf (1963).

MODER, J.J. e C.R. Phillips (1964) - Project Management with CPM and PERT. Reinhold Publishing Co. N.Y.

MUTH, J.F e G.L. Thompson (1963) - Industrial Scheduling Prentice Hall Inc., Inglewood Cliffs, N.Y.

PANDIT, S.N. (1962) - The Loading Problem, in Operations Research, v.10, n.5, p.639-646.

PEART, R.M., C.E. French e W.G.Isaacs (1961) - Optimizing Systems when Components have Disentinuons Cost Functions in Operations Research, v.9, n.4, p.468-478.

PETERSEN, C.C. (1967) - Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R. and D. Projects, in Management Science, v.13, n.9, p. 736-750.

POLLACK, M. e W. Wiebenson (1960) - Solutions of the Shortest Route-Problem - A Review, in Operations Research, v.8, n.2, p.224-230.

ROY, B. (1963) - Programration Mathematique et Description Segmentée, in Metra, v.II, n.4, p.523-535.

ROY, B., T.P. Nghien e P. Bertier (1965) - Programmes Linéaires en Nombres Entiers et Procedure S.E.P., in Metra, v.IV, n. 3, p.441-460.

SIMONNARD, M. (1966) - Linear Programming - Prwntice Hall, Inc. Englewood Cliffs, N.Y.

SRINIVASAN, A.V. (1965) - An Investigation on Some Computational Aspects of Integer Programming, im Journal of the Association for Computing Machinery, v.12, n.4, p.525-535.

THOMPSON, G.L. (1964) - The Stopped Simplex Method: I Basic Theory for Mixed Integer Programming, in Revue Francaise de Recherche Opérationelle, v. 8. n.31, p.159-182.

WAGNER, H.M. (1959) - An Integer Linear Programming Model for Machine Shop Scheduling, in Naval Research Logistics Quarterly, v.6, n.2, p.131-140.

WAGNER, H.M., R. Giglio e R.G. Glaser (1964) - Preventive Maintenance Scheduling by Mathematical Programming, in Management Science, v.10, n.2, p.316-334.

WEINGARTNER, H.M. (1963) - Mathematical Programming and the Analysis of Capital Budgeting Problems. Prentice Hall Inc, Englewood Cliffs, N.Y.

WULSON, R.B. (1967) - Stronger Cuts in Gomory's all Integer Programming Algorithm, Letter The Editor, in Operations Research, v.15, n.1, p.155-157.

WILSON, R.C. (1965) - A Packaging Problem, in Management Science, v.12, n.4, p.B135-B145.

WOILER, S. (1967) - Implicit Enumeration Algorithms for Discrete Optimization Problems. Department of Industrial Engineering Tech. Report, n.4 - May 1967 - Stanford University.