CONTRIBUIÇÕES AO ESTUDO DO PROBLEMA DA MOCHILA

Vera Lucia da Motta

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

Aprovada por:

Prof. Nelson Maculan Filho
(Presidente)

Prof. Paulo O.Boaventura Netto

Prof. Jair Köiller

RIO DE JANEIRO, RJ - BRASIL

DEZEMBRO DE 1981

DA MOTTA, VERA LUCIA

Contribuições ao Estudo do Problema da Mochila (Rio de Janeiro) 1981

IX, 199p. 29,7cm (COPPE-UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1981)

Tese - Univ. Fed. Rio de Janeiro. Fac. Engenharia

1. Estudos de Algoritmos de Programação Inteira a Uma Restrição I.COPPE/UFRJ II.Título (série)

AGRADECIMENTO

Agradeço especialmente ao professor Nelson Maculan Filho que, com seu encorojamento constante e orientação segura, tornou possível a realização deste trabalho. A dedicação e o carinho a mim dispensados, por voce Maculan, eu os guardo para devolver a tantos quantos de mim precisarem.

À professora e amiga Maria Luiza Villares agradeço o apoio que a mim dispensou, o qual contribuiu, sobremaneira, para que eu desenvolvesse com tranquilidade o meu trabalho.

Aos professores Jair Köiller e Paulo Oswaldo Boaventura agradeço a atenção dedicada ao meu trabalho, assim como orientação recebida.

Aos professores e colegas Antonio José de Nardi, Ronaldo Luiz Menezes de Albuquerque e Oswaldo de Souza Campos agradeço o incentivo que recebi desde o início do meu curso.

A excelente datilografia deste trabalho é devido à Sta. Suely Klajman a quem agradeço profundamente.

Finalmente o meu agradecimento ao Dr. Homero Roberto Passos Werneck de Carvalho, que me proporcionou a oportunidade de realizar meu curso, liberando-me das minhas atividades docentes da U.F.R.R.J e constantemente me incentivando com palavras de estímulo e confiança.

RESUMO

Neste trabalho nós enfocamos o problema da mochila, caracterizando sua importância, seus métodos de solução, técnicas e vantagens na transformação de um programa inteiro de várias restrições em um problema mochila, destacamos a necessidade de uma solução aproximada para socorrer problemas de grande porte e desenvolvemos um método de solução apoiado em grupos gerados por um determinado elemento e classes laterais.

É feito um estudo sobre o "cutting-stock problem", ressaltando a importância da técnica de geração de colunas, a qual resolve o embaraço causado pela quantidade exagerada de modelos de cortes. Essa técnica faz surgir a cada passo um problema mochila cuja rapidez de solução é fundamental para obtenção do ótimo do "cutting-stock problem". Desta feita a solução aproximada se torna a mais indicada. Usamos, por conseguinte, um algoritmo extremamente rápido quanto aproximado.

A técnica de programação dinâmica é desenvolvida e ilustrada com exemplos simples que nos permitem aplicar sem embaraços os algoritmos elaborados. Isto contribuiu para que se contruisse a cada passo um algoritmo mais eficiente computacionalmente. Neste caso a propriedade periódica é de relevante importância, pois minimiza, como concluímos, as computações.

A aplicação da teoria dos grupos no desenvolvimento de outra técnica de solução é precedida de uma caracterização algébrica dos elementos utilizados. Definições, proprie-

dades e teoremas da álgebra são amplamente abordados. Posteriormente à construção do algoritmo relativo a esta técnica, fazemos uma comparação entre os problemas mochila simples e o mochila grupo, dando ênfase à utilização de um ou outro processo de solução quanto ao aspecto operacional.

A importância do problema mochila ficou justificada pela sua utilização em várias situações práticas. O "cutting-stock problem" é a situação real escolhida para ressaltar a importância do problema, bem como a necessidade de busca de soluções extremamente rápidas para o problema mochila, o qual é usado como rotina muitíssimo solicitada no algoritmo elaborado.

O método dos multiplicadores de Lagrange dentro da abordagem considerável feita neste trabalho, se apresenta como método suficiente para redimensionar mochilas e estimar dados. A partir do momento em que sejam conhecidos os dados (daí não poder seu usado no "cutting-stock problem") mostramos que existem soluções fora da região viável que podem ser escolhidas com vantagens como solução aproximada. Caso exista uma solução fora da região viável mas suficientemente próxima de b pode-se pensar na possibilidade de alterar b (redimensionar a mochila).

ABSTRACT

In this work we approach the knapsack Problem ., , and characterize its importance, solution methods and the advantages in the transformation of an integer program into a knapsack problem. We point out the need for approximate solutions to help solving large-scale problems and develop a solution method based on groups generated by an element and on lateral classes.

We study the "cutting-stock problem" and focus on the importance of column-generation, that solves the difficulties brought by the exagerate number of different cutting models. This technique gives rise at each step to a knapsack problem that must be solved by a fast method. This indicates the convenience of an approximate solution.

Dynamic programming technique is presented and illustrated with some simple examples fo tacilitate the comprehension of the proposed algorithms. We presented a series of increasingly computationaly efficient algorithms using besides other properties the periodice to minimize the number of the computations.

Theoretical group methods are then presented following analgebraic characterization of the involved elements. After an algorithm using this technique is built, we compare its performance in the solution of knapsack problem with that of a simple knapsack problem algorithm.

The knapsack problem has several practical. Among them we choose the cutting stock problem to show its relevance. For this last problem we develop a process which

demands extremely fast solutions of knapsack problems.Finally Lagrange multipliers method is proved to be a efficient method for the estimation of the data and to change if necessary the size of the knapsack. This is done when an almost feasible point which can be considered as an approximate solution is found.

ÍNDICE

CAPÍTULO I

INTRODUÇÃO

1. Formulação do Problema

Consideremos a seguinte situação:

"Uma transportadora suporta uma carga máxima "b". Existem $\underline{n}$ diferentes tipos de itens a serem transportados, sendo a_j e c_j respectivamente o peso e o valor relativo do item j. Queremos selecionar esses itens, de tal modo que a carga a ser transportada seja tal que a soma de todos os valores relativos é MÁXIMA".

A modelagem é simples. Seja x_j a quantidade do item j a ser transportada e sejam os demais parâmetros inteiros. Essas considerações concorrem para a formulação do seguinte problema de programação inteira:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_j \, x_j \leq b$$

$$x_j \geq 0 \quad \text{e inteiro} \quad (j=1,2,\ldots,n)$$

onde a_j, c_j e b são inteiros.

O modelo acima, que resolve a situação criada, é do tipo:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_j \, x_j \leq b$$

$$x_j \geq 0 \quad \text{e inteiro } (j=1,2,\ldots,n) \qquad (1)$$

onde os c_j, a_j e b são inteiros, sendo que cada a_j $(j=1,2,\ldots,n)$ e b são positivos.

O problema de programação inteira com uma restrição (1) é chamado "Problema da Mochila". Este nome é devido ao fato dele estar relacionado com o enchimento de uma mochila que pode suportar um peso máximo b. O referido problema consiste em enchê-la de tal sorte que não seja excedida sua capacidade e se maximize a soma dos seus valores relativos c_j, sendo a_j o peso de cada item j. Supõe-se que existam n tipos de itens.

O problema da transportadora citado inicialmente é, portanto, um problema do tipo mochila.

Em muitas situações os c_j serão também positivos. Caso $c_j = 0$, o x_j correspondente é igualmente nulo, pois não há sentido transportar um item sem valor relativo, se queremos maximizar a soma desses valores relativos.

2. Sua Importância

Embora o problema tipo mochila seja o mais simples programa inteiro, ele merece estudo especial pois, além de ser representativo de muitas situações industriais, ele aparece em muitos algoritmos como subprograma. A busca de soluções que tirem proveito da existência de apenas uma restrição fica portanto justificada.

Antes de passarmos ao capítulo seguinte, no qual estudaremos algoritmos específicos, nos permitiremos maior motivação fazendo uma modelagem tipo mochila ao analisarmos o clássico problema de investimento que segue:

"Existem n possibilidades de investimento independentes. O problema consiste da escolha entre essas n possibilidades de investimento de tal forma a maximizar o lucro total dos investimentos sujeito a restrição do capital disponível."

Assim sendo sejam:

c_j o lucro proveniente do projeto j

a_j o custo do projeto j

b o capital disponível

$x_j = 1 \rightarrow$ o projeto é aceito

$x_j = 0 \rightarrow$ o projeto é rejeitado

O modelo básico é então:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_j\, x_j \leq b$$

$$x_j = 0 \text{ ou } 1 \ , \ j=1,2,\ldots,n$$

o qual é um problema mochila com variáveis ZERO-UM. Os parâmetros c_j , a_j e b devem ser tomados como positivos e inteiros.

Para maior esclarecimento vale ressaltar aqui duas extensões para o modelo, assunto discutivo em Lorie and Savage |21| e Weingartner |28|.

1º caso: imaginemos a situação de investimento acima dividida em vários períodos com custos e capital disponível voltados para a exigência do período. Neste caso, temos:

a_{ij} $(j=1,...,n\ ;\ t=1,...,T)$ → custo não negativo do projeto j no período t

b_t → capital disponível no período t $(t=1,...,T)$

Este caso é solucionado pelo modelo:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_{tj}\, x_j \leq b_t \qquad (t=1,...,T)$$

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

O problema acima, caso $T \neq 1$, é um programa inteiro padrão ZERO-UM com uma matriz de restrição. Txn não negativa. Caso $T = 1$ torna-se um programa mochila.

2º caso: imaginemos que as n possibilidades de investimento são particionadas em subconjuntos disjuntos $n_1,.......,$ $n_p (p \leq n)$ e é especificado que precisamente um projeto em cada subconjunto deve ser selecionado. Considerando ainda a divisão por período caracterizada no caso 1º, temos a seguinte solução:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_{tj} \, x_j \leq b_t \qquad (t=1,\ldots,T)$$

$$\sum_{j \in n_i} x_j = 1 \qquad (i=1,\ldots,p \leq n)$$

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

CAPÍTULO II

MÉTODOS DE SOLUÇÃO

1. Técnicas de Programação Dinâmica

Recordemos, inicialmente, a formulação do problema mochila para facilidade de referência:

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

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_j \, x_j \leq b$$

$$x_j \geq 0 \text{ e inteira}, \quad j=1,\ldots,n$$

onde os c_j, a_j e b são inteiros, sendo que cada a_j ($j=1,\ldots,n$) e b são positivos.

Definamos $f(k,g)$ como sendo o máximo valor da funçõa objetivo usando os primeiros k e g, onde $k=1,\ldots,n$ e $g=0,1,\ldots,b$. Feito isto, temos que:

$$f(k,g) = \text{MÁXIMO} \quad \sum_{j=1}^{k} c_j \, x_j$$

$$\text{sujeito a:} \quad \sum_{j=1}^{k} a_j \, x_j \leq g$$

$$x_j \geq 0 \text{ e inteiro } (j=1,\ldots,k)$$

Podemos, portanto, encontrar $f(k,g)$ para $k=1,\ldots,n$ e $g=0,1,\ldots,b$.

Nosso objetivo é achar $f(n,b)$ onde n é o número de itens e b é o limite máximo de peso. Isto será conseguido encontrando-se todos os $f(k,g)$ a partir de $k=1,\ldots,n$ e $g=0,1,\ldots,b$. É fácil entender que $f(k,0)=0$, $\forall k$, pois se o limite de peso é ZERO então itens não podem ser considerados e evidentemente não há o que maximizar.

1.1. Problemas com variáveis limitadas

Embora $x_j \leq [g/a_j]$ para todo j, pois a positividade de $a_j (j=1,\ldots,n)$ e de g não permitem que x_j seja maior que aquele valor, em certos modelos torna-se necessária a restrição adicional $x_j \leq r_j \leq [g/a_j]$, r_j inteiro positivo. Como exemplo podemos citar o caso de um problema 0-1, onde $r_j = 1$.

Estudo da técnica de programação dinâmica aplicável a esses casos

Suponhamos que para um determinado k $(k=2,\ldots,n)$, os valores $f(k-1,g)$ são conhecidos para todo $g=0,1,\ldots,b$. Nosso intuito é achar $f(k,g)$ para um dado g. Para tal podemos começar escrevendo:

$$f(k,g) = \text{MÁXIMO} \quad c_k x_k + \left[\begin{array}{l} \sum_{j=1}^{k-1} c_j x_j \\ \\ \sum_{j=1}^{n-1} a_j x_j \leq g - a_k x_k \\ \\ x_j \geq 0 \text{ e inteiro } (j=1,\ldots,k) \end{array} \right]$$

sujeito a:

$$x_k = 0,1,\ldots, [g/a_k]$$

obs.: se um limite superior $r_k < [g/a_k]$ existe, então ele substituirá $[g/a_k]$ em $f(k,g)$ e assim $x_k \leq r_k$.

Para um valor inteiro conhecido x_k $(0 \leq x_k \leq [g/a_k])$, $f(k,g)$ reduz-se à forma abaixo, já que x_k deixa de ser incógnita.

$$f(k,g) = c_k\, x_k + \text{MÁXIMO} \left[\begin{array}{l} \sum_{j=1}^{k-1} c_j\, x_j \\ \text{sujeito a: } \sum_{j=1}^{n-1} a_j\, x_j \leq g - a_k\, x_k \\ x_j \geq 0 \text{ e inteiro } (j=1,..,k-1) \end{array} \right]$$

Entretanto a expressão entre colchetes é precisamente $f(k-1,\ g-a_k x_k)$, a qual é conhecida visto que $g - a_k x_k$ é um inteiro não negativo não excedendo $\underline{b}$. Por exemplo: no cálculo de $f(2,4)$ aparece $f(1,4)$ que já foi calculado anteriormente. Podemos escrever, então:

$$f(k,g) = \underset{x_k = 0,1,\ldots,[g/a_k]}{\text{MÁXIMO}} \quad (c_k x_k + f(k-1,\ g-a_k x_k)) \qquad (2)$$

Para cada $k = 2,3,\ldots,n$ a equação (2) pode ser usada para achar $f(k,g)$ para $g = 0,1,\ldots,b$.

<u>obs.</u>: 1) Quando $k = 0$ então $f(k,g) = 0$, pois não existem itens. Podemos então definir, sem perda de significado, $f(0,g) = 0$ e a equação (2) pode ser usada para $k = 1$, diretamente.

2) Quando $a_k > g \rightarrow [g/a_k] = 0$. Então, por (2) temos que:

$$f(k,g) = f(k-1,g) \quad \text{com} \quad x_k = 0$$

Lembrando que $f(k,0) = 0,\ \forall k$, podemos solucionar problemas com variáveis limitadas aplicando a equação (2), para cada $k=1,2,\ldots,n$ e tendo como condições iniciais:

$f(0,g) = 0 \ , \ \forall g$

$f(k,0) = 0 \ , \ \forall k \ , \text{ com } \ x_k = 0$

EXEMPLO:

MAXIMIZAR $\quad 10x_1 + 16x_2 + x_3$

sujeito a: $\quad 2x_1 + 3x_2 + 2x_3 \leq 4$

$x_1 \leq 1 \ , \ x_2 \leq 1$

$x_1, x_2, x_3 \geq 0$ e inteiro.

Como x_1 e x_2 não podem exceder 1, substituiremos $[g/a_k]$ por 1 sempre que $[g/a_k] \geq 1$, k=1,2. Para k=3 não há restrições, o limite é dado por $[g/a_3]$.

Condições iniciais: $f(0,g)=0 \ , \ \forall g$

$f(k,0)=0 \ , \ \forall k \ , \text{ com } \ x_k = 0$

Todos os $f(k,g)$ serão obtidos, portanto, fazendo uso da equação (2).

K = 1

Temos $a_1 = 2$ e $c_1 = 10$

Obtenção dos limites para x_1 :

$$[g/a_1] = [g/2] = \begin{cases} 0 & \text{para } g = 0,1 \\ 1 & \text{para } g = 2,3,4 \end{cases}$$

Obtenção dos $f(1,g)$:

$f(1,0) = 0 \quad \text{com} \quad x_1 = 0$

$f(1,1) = 10.0 + f(0,1) \rightarrow f(1,1) = 0 \quad \text{com} \quad x_1 = 0$

$$f(1,g) = \underset{x_1 = 0,1}{\text{MÁXIMO}}(10x_1 + f(0,g - 2x_1)) = \underset{x_1 = 0,1}{\text{MÁXIMO}}(10x_1) = 10$$

$g = 2,3,4 \rightarrow f(1,2)=f(1,3)=f(1,4)=10$ com $x_1 = 1$

$\boxed{K = 2}$

Temos $a_2 = 3$ e $c_2 = 16$

Obtenção dos limites para x_2 :

$$[g/a_2] = [g/3] = \begin{cases} 0 & \text{para } g = 0,1,2 \\ 1 & \text{para } g = 3,4 \end{cases}$$

Obtenção dos $f(2,g)$:

$f(2,0) = 0$ com $x_2 = 0$

$f(2,g) = 16.0 + f(1,g-3.0) = f(1,g)$ para $g=1,2$ com $x_2 = 0$

$$\left.\begin{array}{l} f(2,1) = 0 \\ f(2,2) = 10 \end{array}\right\} \text{com } x_2 = 0$$

$$f(2,g) = \underset{x_2 = 0,1}{\text{MÁXIMO}}(16x_2 + f(1,g-3x_2)),\ g = 3,4$$

$$f(2,3) = \underset{x_2 = 0,1}{\text{MÁXIMO}}(16x_2+f(1,3-3x_2))=\text{MÁXIMO}(10,16)=16 \text{ com } x_2 = 1$$

$$f(2,4) = \underset{x_2 = 0,1}{\text{MÁXIMO}}(16x_2+f(1,4-3x_2))=\text{MÁXIMO}(10,16)=16 \text{ com } x_2 = 1$$

$\boxed{K = 3}$

Temos $a_3 = 2$ e $c_3 = 1$

Obtenção dos limites para x_3 :

$$[g/a_3] = [g/2] = \begin{cases} 0 & \text{para} \quad g = 0,1 \\ 1 & \text{para} \quad g = 2,3 \\ 2 & \text{para} \quad g = 4 \end{cases}$$

Obtenção dos $f(3,g)$:

$f(3,0) = 0$ com $x_3 = 0$

$f(3,1) = 1.0 + f(2,1-2.0) = f(2,1) = 0$ com $x_3 = 0$

$$f(3,g) = \underset{x_2 = 0,1}{\text{MÁXIMO}}(1.x_3 + f(2,g-2x_3)) \quad , \ g = 2,3$$

$$= \text{MÁXIMO}(f(2,g), 1 + f(2,g-2)) \ , \ g = 2,3$$

Assim:

$f(3,2) = \text{MÁXIMO}(f(2,2), 1 + f(2,0)) = 10$ com $x_3 = 0$

$f(3,3) = \text{MÁXIMO}(f(2,3), 1 + f(2,1)) = 16$ com $x_3 = 0$

$$f(3,4) = \underset{x_3=0,1,2}{\text{MÁXIMO}}(x_3 + f(2,4-2x_3)) = \text{MÁXIMO}(f(2,4), 1+f(2,2), 2+f(2,0))$$

$$= \text{MÁXIMO}(16,11,2) = 16 \text{ com } x_3 = 0$$

Formando um quadro com os valores encontrados temos:

g	f(1,g)	x_1	f(2,g)	x_2	f(3,g)	x_3
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	10	1	10	0	10	0
3	10	1	16	1	16	0
4	10	1	16	1	16	0

A solução ótima é $f(3,4) = 16$. Em princípio sabemos apenas que $x_3 = 0$. Precisamos conhecer os valores das demais variáveis. É fácil percebermos que os valores das variáveis x_{k-1} são obtidos por intermédio de $f(k-1,\ g - a_k x_k)$. Esta conclusão é devida ao fato de considerarmos o problema sem o item K e evidentemente com um limite de peso diminuido do peso desses itens K , ou seja, $g - a_k x_k$. Esta idéia nos permite calcular as demais variáveis correspondentes a esse ÓTIMO obtido.

Como a solução ÓTIMA é $f(3,4) = 16$ com $x_3 = 0$, temos: $k = 3$ e $g = 4$. E assim:

$f(k-1,g-a_k x_k) = f(3-1,4-a_3.0) = f(2,4) = 16$ com $x_2 = 1$

Com o mesmo raciocínio, temos: $k = 2$ e $g = 4$ e então:

$f(k-1,g-a_k x_k) = f(2-1,4-a_2.1) = f(1,1) = 0$ com $x_1 = 0$

Logo a solução ÓTIMA para o problema é:

$f(3,4) = 16$ com $x_1 = x_3 = 0$ e $x_2 = 1$.

<u>Caso em que a restrição é uma igualdade</u>

Neste caso os valores $f(k,g)$ deverão ser computados de tal forma que o x_k correspondente seja tal que não permita folga na mochila. Assim sendo, o valor $f(k-1,\ g-a_k x_k)$ que aparece no cálculo de $f(k,g)$ só deverá ser computado se $g = a_k x_k$ para um dos possíveis valores de x_k. Caso contrário, x_k não será registrado e para efeito de cálculo, como veremos, será atribuido a $f(k,g)$ o valor $-\infty$. Desta forma $f(k,g) = -\infty$ significa não haver solução que evite a folga, ou seja, que satisfaça a igualdade, e portanto o valor de x_k não é registrado.

O raciocínio acima nos torna possível resolver o problema fazendo uso da equação (2), desde que acrescentemos a condição inicial $f(0,g) = -\infty$, $g = 1,\dots,b$ além de $f(k,0) = 0$, $\forall k$, com $x_k = 0$.

EXEMPLO:

$$\begin{array}{ll} \text{MAXIMIZAR} & 10x_1 + 16x_2 + x_3 \\ \text{sujeito a:} & 2x_1 + 3x_2 + 2x_3 = 4 \\ & x_1 \leq 1 \text{ , } x_2 \leq 1 \\ & x_1, x_2, x_3 \geq 0 \text{ e inteiro} \end{array}$$

O cálculo dos limites para as variáveis x_1, x_2 e x_3 já foi efetuado no exemplo anterior e por conseguinte não será repetido.

Condições iniciais:

$$f(0,g) = -\infty \text{ , } g=1,2,3,4$$

$$f(k,0) = 0 \text{ , } \forall k, \text{com } x_k = 0$$

K=1

Temos $a_1 = 2$ e $c_1 = 10$

$$x_1 = \begin{cases} 0 & \text{para } g = 0,1 \\ 1 & \text{para } g = 2,3,4 \end{cases}$$

Obtenção dos $f(1,g)$:

$f(1,0) = 0$ com $x_1 = 0$

$f(1,1) = 10.0 + f(0,1-2.0) = f(0,1) = -\infty$

$$f(1,g) = \underset{x_1=0,1}{\text{MÁXIMO}}(10x_1 + f(0,g-2x_1)) \text{ , } g = 2,3,4$$

Assim:

$f(1,2) = \text{MÁXIMO}(-\infty,10) = 10$ com $x_1 = 1$

$f(1,3) = \text{MÁXIMO}(-\infty,10+(-\infty)) = -\infty$

$f(1,4) = \text{MÁXIMO}(-\infty,10+(-\infty)) = -\infty$

K = 2

Temos $a_2 = 3$ e $c_2 = 16$

$$x_2 = \begin{cases} 0 & \text{para } g = 0,1,2 \\ 1 & \text{para } g = 3,4 \end{cases}$$

Obtenção dos $f(2,g)$:

$f(2,0) = 0$ com $x_2 = 0$

$f(2,g) = f(1,g)$, $g = 1,2$

$f(2,1) = -\infty$

$f(2,2) = 10$ com $x_2 = 0$

$$f(2,g) = \underset{x_2=0,1}{\text{MÁXIMO}}(16x_2+f(1,g-3x_2))$$

$$= \text{MÁXIMO}(f(1,g),16+f(1,g-3)) \quad , \quad g = 3,4$$

$f(2,3) = \text{MÁXIMO}(-\infty,16+0)=16$ com $x_2 = 1$

$f(2,4) = \text{MÁXIMO}(-\infty,16+(-\infty) = -\infty$

K = 3

Temos $a_3 = 2$ e $c_3 = 1$

$$x_3 = \begin{cases} 0 & \text{para } g = 0,1 \\ 1 & \text{para } g = 2,3 \\ 2 & \text{para } g = 4 \end{cases}$$

Obtenção dos $f(3,g)$:

$f(3,0) = 0$ com $x_3 = 0$

$f(3,1) = 1.0+f(2,1-2.0) = f(2.1) = -\infty$

$$f(3,g) = \underset{x_3=0,1}{\text{MÁXIMO}}(x_3+f(2,g-2x_3))$$

$$= \text{MÁXIMO}(f(2,g), 1+f(2,g-2), \quad g = 2,3$$

$f(3,2) = \text{MÁXIMO}(f(2,2), 1+f(2,0)=10 \quad \text{com} \quad x_3 = 0$

$f(3,3) = \text{MÁXIMO}(f(2,3), 1+f(2,1))=16 \quad \text{com} \quad x_3 = 0$

$$f(3,4) = \underset{x_3=0,1,2}{\text{MÁXIMO}}(x_3+f(2,4-2x_3))$$

$= \text{MÁXIMO}(f(2,4), 1+f(2,2), 2+f(2,0))=11 \quad \text{com} \quad x_3 = 1$

Formando um quadro com esses valores, temos:

g	f(1,g)	x_1	f(2,g)	x_2	f(3,g)	x_3
0	0	0	0	0	0	0
1	$-\infty$	.	$-\infty$	.	$-\infty$	.
2	10	1	10	0	10	0
3	$-\infty$	.	16	1	16	0
4	$-\infty$	.	$-\infty$	.	11	1

A solução ótima é $f(3,4) = 11$ com $x_3 = 1$. Obteremos a partir daí os valores ótimos para x_2 e x_1.

Obtenção de x_2 : (k_3 ; $g = 4$; $x_3 = 1$)

$f(k-1, g-a_k x_k)=f(2,4-2.1)=f(2,2)=10 \quad \text{com} \quad x_2 = 0$

Obtenção de x_1: ($k = 2$; $g = 2$; $x_2 = 0$)

$f(k-1,g-a_k x_k)=f(1,2-3.0)=f(1,2)=10 \quad \text{com} \quad x_1 = 1$

Solução ótima para o problema:

$$f(3,4) = 11$$
$$x_1 = x_3 = 1$$
$$x_2 = 0$$

OBSERVAÇÃO: Embora a técnica apresentada seja importante, visto que ela pode tratar problemas com variáveis limitadas, a mesma não tira vantagem da linearidade da função objetivo e da restrição. Ela pode envolver muitos cálculos, especialmente se b é grande e se as variáveis podem tomar muitos valores. Este procedimento é consideravelmente mais eficiente quando não se apresenta limites às variáveis, como veremos a seguir.

1.2. Problemas com variáveis não limitadas

Seja $f(k-1,g)$ conhecido para todo g e $k=2,...,n$. Nosso objetivo é calcular $f(k,g)$, a partir dessa informação. Como $f(k-1,g)$ é suposto conhecido podemos compará-lo com a expressão $c_k+f(k,g-a_k)$, que representa o valor de $f(k,g)$ para $x_k=1$ (por (2)). Pode ocorrer:

(i) $c_k+f(k,g-a_k) \leq f(k-1,g)$

(ii) $c_k+f(k,g-a_k) > f(k-1,g) \quad , \quad g \geq a_k$

Se ocorre (i) então não houve melhora nas buscas implicando em $x_k=0$ e $f(k,g)=f(k-1,g)$. Se ocorre (ii) significa que houve melhora e então x_k deve ser no mínimo igual a 1 (tal que não viole a restrição) e $f(k,g)= c_k + f(k,g-a_k)$.

Assim:

(i) $c_k + f(k,g-a_k) \leq f(k-1,g) \rightarrow x_k = 0$

(ii) $c_k + f(k,g-a_k) > f(k-1,g)$ para $g \geq a_k \rightarrow x_k \geq 1$

Como $f(k,g)$ é uma solução máxima, temos:

$$f(k,g) = \text{MÁXIMO}\left(f(k-1,g), c_k + f(k,g-a_k)\right) \text{ para } g \geq a_k \tag{3}$$

A expressão (3) nos permite computar $f(k,g)$ para $k=1,\ldots,n$ e $g=0,1,\ldots,b$ com as condições iniciais citadas abaixo e já analisadas anteriormente.

$f(k,0) = 0$, $\forall k$, com $x_k = 0$

$f(0,g) = 0$, $\forall g$

EXEMPLO:

MAXIMIZAR $10x_1 + 16x_2 + x_3$

sujeito a: $2x_1 + 3x_2 + 2x_3 \leq 4$

$x_1, x_2, x_3 \geq 0$ e inteiros

$\boxed{K=1}$

Temos $a_1 = 2$ e $c_1 = 10$

$f(1,0) = 0$ com $x_1 = 0$

$f(1,g) = \text{MÁXIMO}\{f(0,g), 10+f(1,g-2) \text{ para } g \geq 2\}$, $g=1,2,3,4$

$f(1,1) = f(0,1) = 0$ com $x_1 = 0$

$f(1,2) = \text{MÁXIMO}\{f(0,2), 10+f(1,0)\} = 10$ com $x_1 \geq 1$

$f(1,3) = \text{MÁXIMO}\{f(0,3), 10+f(1,1)\} = 10$ com $x_1 \geq 1$

$f(1,4) = \text{MÁXIMO}\{f(0,3), 10+f(1,2)\} = 20$ com $x_1 \geq 1$

K = 2

Temos $a_2 = 3$ e $c_2 = 16$

$f(2,0) = 0$ com $x_2 = 0$

$f(2,g) = \text{MÁXIMO}\{f(1,g), 16+f(2,g-3)$ para $g \geq 3\}$

$f(2,1) = f(1,1) = 0$ com $x_2 = 0$

$f(2,2) = f(1,2) = 10$ com $x_2 = 0$

$f(2,3) = \text{MÁXIMO}\{f(1,3), 16+f(2,0)\} = 16$ com $x_2 \geq 1$

$f(2,4) = \text{MÁXIMO}\{f(1,4), 16+f(2,1)\} = 20$ com $x_2 = 0$

K = 3

Temos $a_3 = 2$ e $c_3 = 1$

$f(3,0) = 0$ com $x_3 = 0$

$f(3,g) = \text{MÁXIMO}\{f(2,g), 1+f(3,g-2)$ para $g \geq 2\}$

$f(3,1) = f(2,1) = 0$ com $x_3 = 0$

$f(3,2) = \text{MÁXIMO}\{f(2,2), 1+f(3,0)\} = 10$ com $x_3 = 0$

$f(3,3) = \text{MÁXIMO}\{f(2,3), 1+f(3,1)\} = 16$ com $x_3 = 0$

$f(3,4) = \text{MÁXIMO}\{f(2,4), 1+f(3,2)\} = 20$ com $x_3 = 0$

g	f(1,g)	x_1	f(2,g)	x_2	f(3,g)	x_3
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	10	1	10	0	10	0
3	10	1	16	1	16	0
4	20	1	20	0	20	0

As setas indicam as buscas horizontal e vertical efetuadas no processo de recuperação de solução ótima.

Recuperação dos Apontadores

Neste caso devemos examinar os valores de $f(k,g)$, a partir de $f(n,b)$. Se $f(k,g)$ é tal que $x_k = 0$ isto significa que não houve melhora nas buscas e assim $f(k,g) = f(k-1,g)$ e começamos o cálculo de x_{k-1} através $f(k-1,g)$, pois $x_k = 0$. Se no entanto $x_k = 1$, isto significa pelo que vimos ao determinar esta técnica, que x_k é no mínimo igual a 1. Assim sendo precisamos examinar o valor de x_k relativo a $f(k,g-a_k)$. Se for ZERO significa que x_k deve receber o valor 1, e passamos a examinar o valor de x_{k-1} através $f(k-1,g-a_k)$. Se ao contrário de $x_k = 0$ tivéssemos encontrado novamente $x_k = 1$, x_k seria incrementado de uma unidade e passaria então a valer 2. Nesse caso calculamos o novo $g = g-a_k$ (a proporção que encontramos $x_k = 1$, g é diminuido de a_k) e $f(k,g-a_k)$ é novamente examinado para verificar se x_k sofre outra incrementação (se for 1) ou se recebe o valor anterior (se for 0) nos permitindo analisar, então, x_{k-1} através $f(k-1,g-a_k)$, repetindo o mesmo raciocínio. Tendo por base a tabela estamos efetuando busca vertical quando subtraimos a_k de g e busca horizontal quando subtraimos 1 de k.

Este raciocínio nos conduz ao seguinte PROCEDIMENTO:

PASSO 1: Faça $k = n$, $g = b$, $j = 0$ e determine x_k a partir de $f(k,g)$. Se $x_k = 0$ vá para o PASSO 2. Senão vá para o PASSO 3.

PASSO 2: Faça $k = k-1$. Se $k = 0$, PARE. Senão determine x_k a partir de $f(k,g)$ e vá para o PASSO 3.

<u>PASSO 3</u>: Se $x_k = 1$ faça $g = g-a_k$, $j = j+1$ e vá para o PASSO 4. Se $x_k = 0$ faça $x_k = j$, $j = 0$ e vá para o PASSO 2.

<u>PASSO 4</u>: Determine x_k por $f(k,g)$ e volte ao PASSO 3.

<u>APLICAÇÃO AO EXEMPLO</u>

(1) $k = 3$; $g = 4$; $j = 0$

Como $x_3 = 0$ vá para o PASSO 2.

(2.1) $k = 3-1 \rightarrow k = 2$

$x_2 = 0$. Vá para o PASSO 3

(3.1) Como $x_2 = 0$ e $j = 0$ $\rightarrow x_2 = 0$ e vá para o PASSO 2

(2.2) $k = k - 1 \rightarrow k = 1$

$x_1 = 1$. Vá para o PASSO 3

(3.2) Como $x_1 = 1$ faça $g = g-a_1 \rightarrow g = 4-2 = 2$

$j = j+1 \rightarrow j = 1$ e vá para o PASSO 4

(4.1) $x_1 = 1$ e volte para o PASSO 3.

(3.3.1) Como $x_1 = 1$ faça $g = g - a_1 \rightarrow g = 0$

$j = j+1 \rightarrow j = 2$ e vá para o PASSO 4

(4.1.1) $x_1 = 0$ e volte para o PASSO 3.

(3.3.2) Como $x_1 = 0$ e $j = 2$ $\rightarrow$ $x_1 = 2$, $j = 0$ e vá para o PASSO 2.

(2.3) $k = k-1$ $\rightarrow$ $k = 0$ $\rightarrow$ PARE

Solução ótima: $f(3,4)=20$ com $x_3 = x_2 = 0$ e $x_1 = 2$

Caso em que a restrição é uma igualdade

Para resolver problemas desse tipo a única modificação natural se encontra nas condições iniciais. Como já analisamos anteriormente, devemos tornar $f(k,g) = -\infty$ e não registrar o valor correspondente de x_k sempre que a folga não puder ser evitada. Sendo assim o problema se resolve com a aplicação de (3) e das condições iniciais seguintes:

$$f(k,0) = 0, \ \forall k, \text{ com } x_k = 0$$
$$f(0,g) = -\infty \ , \ g = 1,2,\ldots,n$$

EXEMPLO:

MAXIMIZAR $10x_1 + 16x_2 + x_3$

sujeito a: $2x_1 + 3x_2 + 2x_3 = 4$

$x_1, x_2, x_3 \geq 0$ e inteiros

Aproveitando os cálculos feitos no exemplo imediatamente anterior, temos:

$\boxed{K = 1}$

Temos $a_1 = 2$ e $c_1 = 10$

$f(1,0) = 0$ com $x_1 = 0$

$f(1,1) = -\infty$

$f(1,2) = 10$ com $x_1 \geq 1$

$f(1,3) = -\infty$

$f(1,4) = 20$ com $x_1 \geq 1$

K = 2

Temos $a_2 = 3$ e $c_2 = 16$

$f(2,0) = 0$ com $x_2 = 0$

$f(2,1) = -\infty$

$f(2,2) = 10$ com $x_2 = 0$

$f(2,3) = \text{MÁXIMO}\{f(1,3), 16+f(2,0)\} = 16$ com $x_2 \geq 1$

$f(2,4) = \text{MÁXIMO}\{f(1,4), 16+f(2,1)\} = 20$ com $x_2 = 0$

K = 3

Temos $a_3 = 2$ e $c_3 = 1$

$f(3,0) = 0$ com $x_3 = 0$

$f(3,1) = -\infty$

$f(3,2) = 10$ com $x_3 = 0$

$f(3,3) = \text{MÁXIMO}\{f(2,3), 1+f(3,1)\} = 16$ com $x_3 = 0$

$f(3,4) = \text{MÁXIMO}\{f(2,4), 1+f(3,2)\} = 20$ com $x_3 = 0$

g	f(1,g)	x_1	f(2,g)	x_2	f(3,g)	x_3
0	0	0	0	0	0	0
1	$-\infty$	.	$-\infty$	.	$-\infty$	.
2	10	1	10	0	10	0
3	$-\infty$	.	16	1	16	0
4	20	1	20	0	20	0

Recuperação dos apontadores

À análise feita para o caso da desigualdade deve-se acrescentar, naturalmente, o fato de que se $f(n,b) = -\infty$ então não há solução. Se $f(n,b) \neq -\infty$ então está garantido que $f(k,g-a_k)$ é sempre diferente de $-\infty$ (busca vertical), assim como $f(k-1,g) \neq -\infty$ (busca horizontal). Isto é fácil de ser percebido, pois se na mochila de tamanho b cabem sem folga 3 itens de peso a_i, 2 itens de peso a_j e 4 de peso a_k, e é retirado, por exemplo, 1 item de peso a_i, então $f(k,g-a_i) \neq -\infty$, pois não haverá folga nessa mochila. Ela passa a comportar exatamente 2 itens de peso a_i, 2 de peso a_j e 4 de peso a_k.

De acordo com as considerações acima podemos formular a seguinte sequência de passos:

PASSO 1: Faça $k=n$, $g=b$. Se $f(k,g) = -\infty$, PARE. Senão faça $j=0$ e determine x_k a partir de $f(k,g)$. Se $x_k = 0$ vá para o PASSO 2. Senão vá para o passo 3.

PASSO 2: Faça $k = k-1$. Se $k=0$, PARE. Senão determine x_k a partir de $f(k,g)$ e vá para o PASSO 3.

PASSO 3: Se $x_k = 1$ faça $g=g-a_k$, $j=j+1$ e vá para o PASSO 4. Se $x_k = 0$ faça $x_k = j$, $j=0$ e vá para o PASSO 2.

PASSO 4: Determine x_k por $f(k,g)$ e volte ao PASSO 3.

APLICAÇÃO AO EXEMPLO

(1) $k=3$; $g=4$

Como $f(k,g) \neq -\infty$, faça $j=0$

$x_3=0$. Logo vá para o PASSO 2.

(2.1) $k=3-1 \rightarrow k=2$.

$x_2=0$. Vá para o passo 3.

(3.1) Como $x_2=0$ faça $x_2=j=0$ e vá para o passo 2

(2.2) $k=2-1 \rightarrow k=1$

$x_1=1$. Vá para o passo 3.

(3.2) Como $x_1=1$ faça $g=g-a_1=2$, $j=j+1=1$ e vá para o passo 4.

(4.1) $x_1=1$. Volte ao passo 3.

(3.3) Como $x_1=1$ faça $g=g-a_1=0$, $j=j+1=2$ e vá para o passo 4.

(4.2) $x_1=0$. Volte ao passo 3.

(3.4) $x_1=0$. Faça $x_1=j=2$, $j=0$ e vá para o passo 2.

(2.3) $k=k-1 \rightarrow k=0$. Como $k=0$, PARE.

SOLUÇÃO ÓTIMA: $f(3,4) = 20$

$x_2=x_3=0$

$x_1=2$

Obs.: As setas desenhadas na tabela indicam as buscas vertical e horizontal efetuadas.

1.3. Algoritmos I e II

Um algoritmo para o problema com variáveis não limitadas torna-se simples já que esse tipo de programa envolve a comparação entre somente dois números.

Algoritmo I

Tendo por base o estudo feito para solução de problemas com variáveis não limitadas, formulamos um algoritmo para o caso específico em que a restrição é uma desigualdade, lembrando que pode ser aplicado com mínimas modificações no caso da restrição ser uma igualdade, como foi feito anteriormente.

Sabemos que a cada valor de $f(k,g)$ computado se associa um número que caracteriza x_k. Esse número, como vimos, pode ser ZERO ou UM. Sendo UM significa que x_k é no mínimo igual a esse valor. Embora sabendo que $f(k,g)$ é não negativo, introduziremos um sinal negativo a frente de seu valor com o objetivo de fazer corresponder a essa quantidade negativa o valor UM para x_k. Assim utilizaremos a CONVENÇÃO abaixo, a qual tornará tarefa bastante simples a busca da solução ótima, após ter sido computada $f(n,b)$.

$x_k \geq 1$ se $f(k,g) < 0$

$x_k = 0$ senão

Como, segundo a convenção adotada, x_k é incrementado de uma unidade se $f(k,g) < 0$, é natural que se faça uso de um contador j para se computar o valor ótimo de x_k, após a obtenção de todos esses valores convencionais $f(k,g)$. Assim se $f(k,g) < 0$, faremos $j = j+1$. Esse processo se repete se para $g = g-a_k$, $f(k,g)$ for novamente "negativo". O valor ótimo para x_k é igual a j quando o processo das incrementações, chega a seu término ou quando não se faz necessário. Ambos os casos são caracterizados por $f(k,g)$ não negativo, segundo a convenção adotada.

Feitas essas considerações, passemos à sequência lógica de passos do algoritmo em questão. Alguns comentários acompanham a formulação.

PASSO 1: Inicialização: $k=0$

$$f(0,g)=0, \quad g=0,1,\ldots,b$$

PASSO 2: Faça $k=k+1$

Faça $f(k,g)=|f(k-1,g)|$ para todo $g < a_k$

Faça $g=a_k$

C O M E N T Á R I O: essas ordens decorrem da fórmula 3, já que $c_k + f(k,g-a_k)$ só faz sentido se $g \geq a_k$. A utilização do módulo se torna imperiosa pois neste caso $x_k=0$.

PASSO 3: Calcule $M = c_k + |f(k,g-a_k)|$

Se $M > |f(k-1,g)|$ faça $f(k,g)= -M$

Senão faça $f(k,g) = |f(k-1,g)|$

C O M E N T Á R I O: esse passo compara cada dois valores possíveis para $f(k,g)$, escolhendo o melhor. Como o sinal negativo caracteriza apenas $x_k \geq 1$, torna-se imperioso trabalhar com o módulo de determinadas expressões como fizemos.

PASSO 4: Se $g < b$, faça $g = g+1$ e vá para o passo 3.

Senão vá para o passo 5.

C O M E N T Á R I O: aqui garantimos o cálculo para um determinado k dos $f(k,g)$, $g \geq a_k$. Lembremos que os $f(k,g)$, $g < a_k$ foram computados em 2.

PASSO 5: Se $k < n$ vá para o passo 2.
Senão faça $j=0$

C O M E N T Á R I O: executado esse passo pela última vez, todos os $f(k,g)$ terão sido computados, e naturalmente obtivemos o valor procurado, qual seja: $f(n,b)$.

PASSO 6: Se $f(k,g) < 0$ faça $j=j+1$ e vá para o passo 7.
Senão vá para o passo 8.

PASSO 7: Faça $g = g-a_k$ e vá para o passo 6.

PASSO 8: Faça $x_k=j$ e $j=0$.
Se $k > 1$ faça $k=k-1$ e vá para o passo 6.
Se $k=1$, faça $f(n,b)=|f(n,b)|$ e PARE.

C O M E N T Á R I O: os passos 6,7 e 8 acham os valores ótimos para os x_k's a partir de $k=n$. Se $f(k,g) \geq 0$ então $x_n = 0$; se $f(k,g) < 0$ então passamos à fase das incrementações para obter o valor ótimo para x_n. Após obtido esse ótimo, o processo se repete para que seja computado o valor ótimo de x_{n-1}, e assim até o cálculo do ótimo para x_1. Conseguindo esse valor faremos $f(n,b) = |f(n,b)|$, pois $f(k,g)$ é não negativo, para todos k e g, e encerra-se o processamento. Alguns valores de $f(k,g)$ foram tomados negativos apenas como um artifício para caracterizar $x_k \geq 1$.

Aplicação do ALGORITMO I

MAXIMIZAR $10x_1 + 16x_2 + x_3$

sujeito a: $2x_1 + 3x_2 + 2x_3 \leq 4$

$x_1, x_2, x_3 \geq 0$ e inteiros

1. Inicialização: $k=0$

$f(0,0)=0$

$f(0,1)=0$

$f(0,2)=0$

$f(0,3)=0$

$f(0,4)=0$

2.1.0 $k=k+1 \rightarrow k=1$

$f(1,g)=f(0,g)$, $g=0,1 \rightarrow f(1,0)=0$ e $f(1,1)=0$

$g=a_1 \rightarrow g=2$

3.1.0 $M=10 + |f(1,0)| \rightarrow M=10$

Como $M > f(0,2) \rightarrow f(1,2)= -10$

4.1.0 Como $g < 4 \rightarrow g=g+1 \rightarrow g=3$ vá para 3

3.1.2 $M=10 + |f(1,1)| \rightarrow M=10$

Como $M > |f(0,3)| \rightarrow f(1,3)= -10$

4.1.2 Como $g < 3 \rightarrow g=g+1 \rightarrow g=4$ vá para 3

3.1.3 $M=10 + |f(1,2)| \rightarrow M=20$

Como $M > |f(0,3)| \rightarrow f(1,4)= -20$

4.1.3 Como $g=4$ vá para 5

5.1.0 Como $k < 3$ vá para 2

2.2.0 $k=k+1 \rightarrow k=2$

$f(2,g)=|f(1,g)|,\ g=0,1,2 \rightarrow f(2,0)=0$

$f(2,1)=0$

$f(2,2)=10$

$g=3$

3.2.0 $M=16 + |f(2,0)| \rightarrow M=16$

Como $M > |f(1,3)| \rightarrow f(2,3)=-16$

4.2.0 Como $g < 4 \rightarrow g=g+1 \rightarrow g=4$ vá para 3

3.2.1 $M=16 + |f(2,1)| \rightarrow M=16$

Como $M < |f(1,4)| \rightarrow f(2,4)=20$

4.2.1 Como $g=4$ vá para 5

5.2.0 Como $k < 3$ vá para 2

2.3.0 $k=k+1 \rightarrow k=3$

$f(3,g)=|f(2,g)|,\ g=0,1 \rightarrow f(3,0)=0$

$f(3,1)=0$

$g=2$

3.3.0 $M=1 + |f(3,0)| \rightarrow M=1$

Como $M < |f(2,2)| \rightarrow f(3,2)=10$

4.3.0 Como $g < 4 \rightarrow g=g+1 \rightarrow g=3$ vá para 3

3.3.1 $M=1 + |f(3,1)| \rightarrow M=1$

Como $M < |f(2,3)| \rightarrow f(3,3)=16$

4.3.1 Como $g < 4 \rightarrow g=g+1 \rightarrow g=4$ vá para 3

3.3.2 $M=1 + |f(3,2)| \rightarrow M=11$

Como $M < |f(2,4)| \rightarrow f(3,4)=20$

4.3.2 Como $g=4$ vá para 5

5.3.0 Como $k=3 \rightarrow j=0$

6.1.0 $f(3,4) > 0 \rightarrow$ vá para 8

8.1.0 $x_3=0$; $j=0$

Como $k > 1 \rightarrow k=k-1 \rightarrow k=2$ vá para 6

6.2.0 $f(2,4) > 0 \rightarrow$ vá para 8

8.2.0 $x_2=0$ e $j=0$

Como $k > 1 \rightarrow k=k-1 \rightarrow k=1$ vá para 6

6.3.0 $f(1,4) < 0 \rightarrow j=j+1 \rightarrow j=1$ vá para 7

7.1.0 $g=4-2 \rightarrow g=2$ vá para 6

6.3.1 $f(1,2) < 0 \rightarrow j=j+1 \rightarrow j=2$ vá para 7

7.1.1 $g=2-2 \rightarrow g=0$ vá para 6

6.3.2 $f(1,0)=0 \rightarrow$ vá para 8

8.3.0 $x_3=2$; $j=0$

Como $k=1 \rightarrow f(3,4)= 20$. PARE

ALGORITMO II

Nossa atenção agora estará voltada para a análise de um algoritmo que não se preocupa com a obtenção dos $f(k,g)$, mas sim do ótimo por peso g . Assim sendo teremos, em relação ao algoritmo anterior, kg valores a menos armazenados.

Suporte teórico

Consideremos:

$F(g)$ = valor da mochila ótima de tamanho g

$D(g) = \{k | x_k > 0$ em alguma solução ótima para o problema com mochila de tamanho $g\}$.

C O M E N T Á R I O: $D(g) = \emptyset \rightarrow$ não há solução. Fisicamente significa não ser a mochila suficiente para comportar algum item.

Supondo $F(g) \neq 0$, podemos escrever:

$$F(g) = c_k + F(g-a_k) \quad , \quad k \in D(g) \tag{A}$$

$$F(g) > c_j + F(g-a_j) \quad , \quad j \notin D(g) \quad , \quad g-a_j \geq 0 \tag{B}$$

Seja:

$$S(g) = \{j \,|\, g-a_j \geq 0\}$$

Daí segue que: $D(g) \subseteq S(g)$ (C)

Por (A), (B) e (C) podemos escrever:

$$F(g) = \begin{cases} \underset{j \in S(g)}{\text{máximo}} \; c_j + F(g-a_j) & \text{se } S(g) \neq \emptyset \\ 0 & \text{se } S(g) = \emptyset \end{cases} \tag{D}$$

C O M E N T Á R I O: esta fórmula compara tantos números quantos itens cabem na mochila, ou seja, há $|S(g)|$ comparações.

O objetivo do algoritmo é calcular $F(g)$ repetidamente para $g=1,\ldots,b$. A partir da inicialização $F(0)=0$, o processo tem seu começo.

Como visto acima o número de comparações depende do número de elementos de $S(g)$. Tendo por base diminuir o número de cálculos e comparações buscamos num conjunto $Q(g)$ tal que:

(i) $Q(g) \subseteq S(g)$

(ii) $Q(g) \cap D(g) = \emptyset \rightarrow D(g) = \emptyset$

Esse conjunto $Q(g)$ deve ser cuidadosamente escolhido, a fim de que figure entre seus elementos o item j que forneça o máximo procurado. O item (ii), satisfeito, nos garante que se $Q(g) = \emptyset$ então $F(g) = 0$, senão vejamos:

$$Q(g)=\emptyset \rightarrow Q(g) \cap D(g)=\emptyset \rightarrow D(g)=\emptyset \rightarrow \text{NÃO HÁ SOLUÇÃO}$$

Assim (D) pode ser escrito:

$$F(g) = \begin{cases} \underset{j\varepsilon Q(g)}{\text{MÁXIMO}} \quad c_j + F(g-a_j) & \text{se} \quad Q(g) \neq \emptyset \\ \\ 0 & \text{se} \quad Q(g) = \emptyset \end{cases} \tag{E}$$

A fórmula (E) requer não mais, e geralmente menos, computações que a (D).

Obtenção de um apropriado $Q(g)$

Seja:

$$d(g) = \begin{cases} \text{mínimo } \{j \mid j\varepsilon D(g)\} & \text{se} \quad D(g) \neq \emptyset \\ \\ n & \text{se} \quad D(g) = \emptyset \end{cases}$$

Note que $j\varepsilon D(g) \rightarrow D(g-a_j) \subseteq D(g)$. De acordo com a definição de $d(g)$ temos que $d(g) \leq d(g-a_j)$.

Então um apropriado $Q(g)$ é:

$$Q(g) = \{j \mid j \; S(g), \; j \leq d(g-a_j)\} \tag{F}$$

A partir da expressão (F) observamos que esse procedimento será mais eficiente quando os $d(g)$ forem pequenos, pois $Q(g)$ será composto de um menor número de elementos. Assim é mais desejável indexar as variáveis de tal forma que aquelas com pequenos índices provavelmente aparecem na solução ótima. Uma razoável ordem é obtida fazendo $\rho_j = c_j/a_j$ e indexando as variáveis tal que $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$.

C O M E N T Á R I O: de acordo com a definição de $d(g)$ e do $Q(g)$ considerados devemos ordenar as variáveis de tal forma que aquelas que forem as principais responsáveis pelo acréscimo da função objetivo tenham os menores índices. Poderíamos pensar em ordená-las segundo seus custos, ou seja: $c_1 > c_2 > \ldots > c_n$. Entretanto o exemplo abaixo prova a ineficácia desse procedimento.

Seja:

$$\text{MAXIMIZAR} \quad 10x_1 + 4x_2 + 3x_3$$

$$\text{s.a.:} \quad 7x_1 + 3x_2 + x_3 \leq 11$$

É fácil verificar que a solução ótima é obtida com $x_3=11$. Sendo assim uma ordenação conveniente seria:

$$\text{MAXIMIZAR:} \quad 3x_1 + 4x_2 + 10x_3$$

$$\text{s.a.:} \quad x_1 + 3x_2 + 7x_3 \leq 11$$

Esse exemplo torna clara a necessidade de se levar também em consideração os a_j's . Como se pode sentir, a razão c_j/a_j é uma medida bastante razoável para se conseguir tornar pequenos os índices das variáveis não nulas na solução ótima. Além disso a escolha dos $j \leq d(g-a_j)$ está justificada com essa ordenação. Em outras palavras, os itens são numerados a partir de 1, levando-se em consideração o quanto eles aumentam a função objetivo respeitados o limite g e o acréscimo que a esta mesma função objetivo provocam os demais itens. O que procuramos é deixar por último os itens que provavelmente não constarão na mochila ótima.

Passemos agora à sequência de passos do algoritmo em análise. Alguns comentários acompanham a formulação.

<u>PASSO 1</u>: Faça $F(g)=0$ e $d(g)=n$ para todo g.

Ordene as variáveis tal que $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$

Faça $g=1$ e $j=1$

<u>COMENTÁRIO</u>: A inicialização $d(g)=n$ justifica-se pois, de acordo com a ordenação, o maior j (no caso n) é o que menos colabora para o acréscimo da função objetivo. Daí começarmos com este valor para a obtenção do conjunto $Q(g)$.

<u>PASSO 2</u>: Se $g-a_j \geq 0$ e $j \leq d(g-a_j)$ vá para 3. Senão vá para 4.

<u>PASSO 3</u>: Calcule $M = F(g-a_j) + c_j$. Se $M \leq F(g)$ vá para 4. Senão faça $F(g)=M$, $d(g)=j$ e vá para 4.

<u>COMENTÁRIO</u>: O passo 2 determina se $j \in Q(g)$ e o passo 3 faz as devidas comparações para aprimorar o valor de $F(g)$.

<u>PASSO 4</u>: Se $j < n$, faça $j=j+1$ e volte para 2. Senão vá para 5.

PASSO 5: Se $g < b$, faça $j=1$, $g=g+1$ e volte para o passo 2.

Se $g=b$, vá para 6.

C O M E N T Á R I O: O passo 4 quando executado pela última vez em cada iteração encerra os cálculos para a obtenção de um determinado $Q(g)$. O passo 5 ordena o cálculo de $F(g)$ para o g imediatamente seguinte. Quando o passo 5 é executado pela última vez, $F(b)$ é encontrado.

PASSO 6: Inicialize $g=b$ e $x_j=0$, $j=1,\dots,n$.

PASSO 7: Se $F(g)=0$, PARE

Senão vá para o passo 8.

PASSO 8: Faça $x_{d(g)} = x_{d(g)} + 1$ e $g=g-a_{d(g)}$

Vá para 7

C O M E N T Á R I O: Os passos 6,7 e 8 determinam o valor ótimo das variáveis. Os $x_j \neq 0$ serão encontrados no passo 8, pois sendo $F(g) \neq 0$ o $d(g)$ corresponde ao j com $x_j \neq 0$. A variável x_j será incrementada de uma unidade quando $d(g) = d(g-a_{d(g)})$. Isto pode ser observado na aplicação do algoritmo no exemplo considerado e que vem a seguir.

Aplicação do Algoritmo II

$$\text{MAXIMIZAR} \quad 10x_1 + 16x_2 + x_3$$

$$\text{s.a.:} \quad 2x_1 + 3x_2 + 2x_3 \leq 4$$

$$x_1, x_2, x_3 \geq 0 \text{ e inteiros}$$

1. $F(4)=F(3)=F(2)=F(1)=F(0)=0$

   $d(4)=d(3)=d(2)=d(3)=d(0)=3$

   De acordo com a ordenação o problema transforma-se em:

   MAXIMIZAR $16x_1 + 10x_2 + x_3$

   s.a.: $3x_1 + 2x_2 + 2x_3 \leq 4$

   $g=1$; $j=1$

2.1.1 $g-a_1 < 0$ → vá para passo 4

4.1.1 $j < 4$ → $j=j+1=2$ → vá para passo 2

2.1.1 $g-a_2 < 0$ → vá para o passo 4

4.1.2 $j < 4$ → $j=j+1=3$ → vá para o passo 2

2.1.3 $g-a_3 < 0$ → vá para passo 4

4.1.3 $j=3$ → vá para passo 5

5.1.1 $g < 4$ faça: $j=1$

$g=g+1=2$ → vá para passo 2

2.2.1 $g-a_1 < 0$ → vá para passo 4

4.2.1 $j < 3$, faça $j=j+1=2$ → vá para passo 2

2.2.2 $g-a_2=0$ e $j < d(0)$ → vá para passo 3

3.2.1 $M = F(0) + c_2 = 10$

$M > F(2)$ → $F(2) = 10$ e $d(2)=2$ → vá para 4

4.2.2 $j < 3$ → $j=j+1=3$ → vá para passo 2

2.2.3 $g-a_3=0$ e $j=d(0)$ → vá para passo 3

3.2.2 $M = F(0) + c_3 = 1$

$M < F(2)$ → vá para passo 4

4.2.3. $j=3$ → vá para passo 5

5.2.1 $g < 4 \rightarrow j=1$ e $g=g+1=3 \rightarrow$ vá para passo 2

2.3.1 $g-a_1=0$ e $j < d(0) \rightarrow$ vá para passo 3

3.3.1 $M = F(0) + c_1 = 16$

$M > F(3) \rightarrow F(3) = 16$ e $d(3) = 1 \rightarrow$ vá para 4

4.3.1 $j < 3 \rightarrow j=j+1=2 \rightarrow$ vá para passo 2

2.3.2 $g-a_2=1$ e $j < d(1) \rightarrow$ vá para passo 3

3.3.2 $M = F(1) + c_2 = 10$

$M < F(3) \rightarrow$ vá para passo 4

4.3.2. $j < 3 \rightarrow j=j+1=3 \rightarrow$ vá para passo 2

2.3.3 $g-a_3=1$ e $j=d(1) \rightarrow$ vá para passo 3

3.3.3 $M = F(1) + c_3 = 1$

$M < F(3) \rightarrow$ vá para passo 4

4.3.3 $j=3 \rightarrow$ vá para passo 5

5.3.1 Como $g < 4$ faça $j=1$ e $g=g+1=4 \rightarrow$ vá para 2

2.4.1 $g-a_1=1$ e $j < d(1) \rightarrow$ vá para passo 3

3.4.1 $M = F(1) + c_1 = 16$

$M > F(4) \rightarrow F(4)=16$ e $d(4)=1 \rightarrow$ vá para 4

4.4.1 $j < 3 \rightarrow j=j+1=2 \rightarrow$ vá para passo 2

2.4.2. $g-a_2=2$ e $j=d(2) \rightarrow$ vá para passo 3

3.4.2 $M = F(2) + c_2 = 10 + 10 = 20$

$M > F(4) \rightarrow F(4) = 20$ e $d(4) = 2 \rightarrow$ vá para 4

4.4.2 $j < 3 \rightarrow j=j+1=3 \rightarrow$ vá para passo 2

2.4.3 $g-a_3=2$ e $j > d(2) \rightarrow$ vá para passo 4

4.4.3 $j=3 \rightarrow$ vá para 5

5.5.1 $g=4 \rightarrow$ vá para passo 6

6.1 $g=4$; $x_1=x_2=x_3=0$

7.1 $F(4) = 20 \rightarrow$ vá para passo 8

8.1 $x_{d(4)} = x_{d(4)} + 1 \rightarrow x_2=1$

$g=g-a_2=4-2=2 \rightarrow$ vá para passo 7

7.2 $F(2) = 10 \rightarrow$ vá para passo 8

8.2 $x_{d(2)} = x_{(d2)} + 1 \rightarrow x_2 = 2$

$g = g - a_2 = 0 \rightarrow$ vá para passo 7

7.3. $F(0) = 0$. PARE

1.4. Uma Propriedade Periódica

Seja o exemplo abaixo, citado em "Integer Programming de Robert S. Garfinkel e George L. Nemhauser", o qual justifica a necessidade do suporte teórico que faremos logo após sua solução.

$$\text{MAX } X_o = 11x_1 + 7x_2 + 5x_3 + x_4$$

$$\text{s.o.: } 6x_1 + 4x_2 + 3x_3 + x_4 \leq 25$$

$$x_1, x_2, x_3, x_4 \geq 0 \text{ e inteiros}$$

Verificando a tabela que apresenta a solução encontrada para o mesmo observamos que para todo $g \geq 9$, $d(g)=1$.

Esta secção preocupa-se com um determinado g^*, tal que $g \geq g^*$ é tal que $d(g)=1$. A existência, a identificação e o uso deste g^* é objetivo do estudo que segue.

TABELA DOS VALORES OBTIDOS

g	F(g)			d(g)		
0	0			4		
1	0	1		4		
2	0	2		4		
3	0	5		4	3	
4	0	7		4	2	
5	0	8		4	2	
6	0	11		4	1	
7	0	12		4	1	
8	0	13	14	4	1	2
9	0	16		4	1	
10	0	18		4	1	
11	0	19		4	1	
12	0	22		4	1	
13	0	23		4	1	
14	0	25		4	1	
15	0	27		4	1	
16	0	29		4	1	
17	0	30		4	1	
18	0	33		4	1	
19	0	34		4	1	
20	0	36		4	1	
21	0	38		4	1	
22	0	40		4	1	
23	0	41		4	1	
24	0	44		4	1	
25	0	45		4	1	

C O M E N T Á R I O:

Os número envolvidos por um círculo expressam os resultados obtidos pelo processo de recuperação dos apontadores, indicam onde há incrementação. Consultando, então, a tabela obtemos:

$F(25) = 45$ com $x_1=4$, $x_2=x_3=0$, $x_4=1$

$F(19) = 34$ com $x_1=3$, $x_2=x_3=0$, $x_4=1$

Suponhamos que as variáveis estão indexadas tal que $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$. Mostraremos que se $\rho_1 > \rho_2$, existe um pequeno g^* tal que para todo $g \geq g^*$, uma mochila ótima de tamanho g tem $x_1 \geq 1$, ou seja, $d(g)=1$.

Para uma mochila de tamanho g , uma solução viável é $x_1 = [g/a_1]$ e $x_j=0$, $j=2,\ldots,n$, senão vejamos:

Fazendo $y_j = a_j x_j$, no problema:

$$\left[\begin{array}{ll} \text{MAX} & \sum\limits_{j=1}^{n} c_j \, x_j \\ \text{s.a.:} & \sum\limits_{j=1}^{n} a_j \, x_j \leq g \\ & x_j \geq 0 \quad (j=1,\ldots,n) \end{array}\right.$$

, temos:

$$\left[\begin{array}{ll} \text{MAX} & \sum\limits_{j=1}^{n} \rho_j \, y_j \\ \text{s.a.:} & \sum\limits_{j=1}^{n} y_j \leq g \\ & y_j \geq 0 \quad (j=1,\ldots,n) \end{array}\right.$$

Como ρ_1 é o maior dos ρ_j a solução ótima é $y_1=g$ e $y_j=0$, $j=2,\ldots,n$ ou seja $x_1=[g/a_1]$ e $x_j=0$, $j=2,\ldots,n$, são soluções para o programa inteiro equivalente.

Denotando o valor da função objetivo do programa inteiro equivalente por F_1, temos:

$$F_1=c_1[g/a_1] > c_1(g/a_1-1)=c_1g/a_1-c_1=c_1/a_1(g-a_1)=\rho_1(g-a_1)$$

Segundo a ordenação das variáveis podemos afirmar que um limite superior para o valor de uma mochila de tamanho g com $x_1=0$ é:

$$F_2 = c_2g/a_2 = \rho_2 g$$

Assim sendo, uma condição suficiente para uma mochila ótima de tamanho g ter $x_1 \geq 1$ é:

$$\rho_1(g-a_1) \geq \rho_2 g \rightarrow g \geq \frac{c_1}{\rho_1 - \rho_2}$$

Acabamos de mostrar que para todo g superior ou igual a $c_1/(\rho_1 - \rho_2)$, $d(g) = 1$. Entretanto é possível que esse fato ocorra antes do valor estabelecido, já que, com o objetivo de provar sua existência, nos apoiamos no limite superior para a função objetivo, qual seja $\rho_2 g$, caso $x_1=0$. Sendo assim o limite para g^* é dado por aquela expressão, ou seja, g^* existe e é tal que:

$$g^* \leq \frac{c_1}{\rho_1 - \rho_2}$$

Consideremos, agora, uma mochila de tamanho $g+ka_1$ $k=1,2,\ldots,$ onde $g \geq g^* - a_1$ (Assim temos que $g+a_1 \geq g^*$ e consequentemente $g+ka_1 \geq g^*$). Visto que $d(g)=1$ para todo $g \geq g^*$, segue a partir da expressão:

$$F(g) = c_k + F(g-a_k) \text{ , } k \in D(g) \text{ que:}$$

$$F(g+ka_1) = kc_1 + F(g+ka_1-ka_1) \quad , \quad k=1,2,\ldots, \quad \text{ou seja:}$$

$$F(g+ka_1) = kc_1 + F(g) \quad , \quad k=1,2,\ldots \qquad \text{(I)}$$

Sendo assim, se $x_1, x_2, \ldots, x_n$ é uma solução ótima para a mochila de tamanho g, então, (x_1+k), $x_2, \ldots, x_n$ é uma solução ótima para a mochila de tamanho $g+ka_1$. Com exceção para o termo linear kc_1, o valor ótimo é periódico com período a_1 para todo $g \geq g^*$. Conhecendo-se g^* e uma solução ótima para todo $g \leq g^*$, é trivial, a partir de (I), calcular a solução ótima para um $g > g^*$. Utilizando a tabela anterior, apliquemos o resultado obtido. Seja, por exemplo, calcular $F(13)$.

Temos que $g^* = 9$, segundo a tabela. Como $13 > g^*$ podemos aplicar (I). Assim:

$F(13) = kc_1 + F(g)$ por (I)

Sendo $a_1 = 6$ podemos escrever: $13 = g+ka_1$

Podemos pensar em: $13 = 1 + 2 \times 6 \rightarrow g=1$ e $k=2$

Sendo $c_1 = 11$, temos, a partir de (I) acima:

$F(13) = 2 \times 11 + F(1) \rightarrow F(13) = 23$

Como $F(1) = 1$ e $\left[\begin{array}{l} x_1=x_2=x_3=0 \\ x_4=1 \end{array}\right.$, temos que:

$$F(13) = 23 \quad \text{e} \quad \left[\begin{array}{l} x_1' = x_1+k = 2 \\ x_2' = x_2 = 0 \\ x_3' = x_3 \\ x_4' = x_4 = 1 \end{array}\right.$$

De acordo com os dados da tabela obtivemos $g^* = 9$. Usando o limite para g^* :

$$g^* \leq \frac{c_1}{\rho_1 - \rho_2} \rightarrow g^* \leq 132$$

Podemos então afirmar que o limite dado por aquela expressão para g^* é geralmente muito ineficaz. Contudo o algoritmo II visto na secção anterior fornece g^* .

Seja $a_k = \max a_j$ e suponha $d(g)=1$, $g^o \leq g < g^o+a_k$. Então a partir da fórmula:

$$F(g) = \begin{cases} \max\limits_{j \in S(j)} \; c_j + F(g-a_j) & \text{se } s(g) \neq \emptyset \\ 0 & \text{se } s(g) = \emptyset \end{cases}$$

nós temos:

$F(g^o+a_k) = \max\limits_j c_j + F(g^o + a_k - a_j)$, pois se a_k é o o maior dos $a_j \rightarrow a_k-a_j \geq 0 \rightarrow g^o + a_k - a_j \geq 0 \rightarrow S(g^o + a_k) \neq \emptyset$

Então $d(g^o+a_k) = 1$, pois $F(g^o+a_k-a_j)$ é tal que $x_1 \geq 1$, já que $g^o \leq g^o+a_k-a_j < g^o+a_k$. Se $x_1 \geq 1$ em $F(g^o+a_k-a_j) \rightarrow x_1 \geq 1$ em $F(g^o+a_k)$, ou seja $d(g^o+a_k) = 1$.

Quer dizer, consideramos que $d(g)=1$ para $g^o \leq g < g^o+a_k$ (trabalhando com o maior intervalo, já que $a_k = \max a_j$) e provamos que $d(g^o+a_k)=1$. Enfim podemos afirmar que $d(g)=1$, para todo $g \geq g^o$, ou seja, $g^o \geq g^*$. Se $d(g^o-1) > 1$, $g^o = g^*$, pois para todo $g \geq g^o$ temos $x_1 \geq 1$.

Assim sendo quando a_k sucessivos valores de g tiverem $d(g)=1$, o algoritmo pode ser encerrado.

Utilizando a conclusão deste estudo, o algoritmo II aplicado ao último exemplo considerado pararia em $g=14$, pois $d(g)=1$ já estaria se repetindo desde $g=9$, portanto 6 vezes. Este valor 6 é obtido por $a_1=\max_j a_j$.

Como visto anteriormente o limite inferior encontrado para g^* foi 132. Assim sendo o estudo feito acima nos oferece um critério de parada bastante considerável para obtenção de g^*.

1.5. ALGORITMO III

De acordo com o estudo desenvolvido na secção anterior formularemos um novo algoritmo, o mais eficiente dentre os apresentados.

PASSO 1: Faça $F(g)=0$ e $d(g)=n$ para todo g.
Faça $a_k=\text{máximo}\ \{a_1,a_2,\ldots,a_n\}$
Ordene as variáveis tal que $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$
Faça: $g=1$, $j=1$, $i=0$

PASSO 2: Se $g-a_j \geq 0$ e $j \leq d(g-a_j)$ vá para 3.
Senão vá para 4.

PASSO 3: Calcule $M = F(g-a_j) + c_j$. Se $M \leq F(g)$ vá para 4.
Senão faça $F(g) = M$, $d(g) = j$ e vá para 4.

PASSO 4: Se $j < n$ faça $j=j+1$ e volte para 2

Senão vá para 5.

PASSO 5: Se $d(g)=1$ faça $i=i+1$

Senão faça $i=0$

C O M E N T Á R I O: Introduzimos um contador i para obter quantas vezes consecutivas $d(g)=1$

PASSO 6: Se $i=a_k$ vá para 7

Senão faça $j=1$, $g=g+1$ e volte para 2

C O M E N T Á R I O: Quando $i=a_k$ encerra-se o processo de obtenção dos $F(g)$, pois a propriedade periódica nos permite a partir daí obter qualquer $F(g_1)$, $g_1 > g$, utilizando para isso a fórmula (I) obtida na seção anterior.

PASSO 7: Faça $x_j=0$, $j=1,2,\ldots,n$

PASSO 8: Se $F(g)=0$, vá para 10

Senão vá para o passo 9

PASSO 9: Faça $x_{d(g)} = x_{d(g)} + 1$ e $g = g - a_{d(g)}$

Vá para 8

PASSO 10: Faça: $k = [b/a_1]$

$$g' = b - ka_1$$

$$F(b) = ka_1 + F(g')$$

$$x_1 = x_1 + k$$

C O M E N T Á R I O: Como $b = k_{a1} + g'$ pode ser obtido para vários valores de k e g' mas g' pode ser tal que $F(g')$ não tenha ainda sido computado, tomamos k o máximo inteiro contido em b/a_1 e evitamos que esse fato ocorra. A partir desse valor de k, g' é facilmente obtido e usando a fórmula (I) obtemos o valor de $F(b)$ e do x_1 correspondente, já que os demais x_j permanecem inalterados, segundo mostramos na secção anterior.

2. Um Algoritmo de Branch and Bound

Consideremos o programa linear:

$$\text{MAXIMIZAR} \quad \sum_{j=1}^{n} (c_j/a_j) y_j$$

$$\text{s.a.:} \quad \sum_{j=1}^{n} y_j = b$$

$$y_j \geq 0 \quad \text{e inteiro } (J=1,\ldots,n)$$

onde $y_j = a_j x_j$, $j=1,\ldots,n$ e as variáveis estão ordenadas tais que $c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$. Isto acontecendo já sabemos que a solução do programa linear é $x_1 = b/a_1$, $x_j=0$ $(j \neq 1)$ e a função objetivo é $c_1(b/a_1)$.

Mostraremos que esse resultado se estende para o caso no qual as variáveis são limitadas.

<u>1º caso</u>: Existe limite superior inteiro

Chamemos esse limite superior inteiro de μ_j. Sendo assim $x_j \leq \mu_j$. Como $x_j = y_j/a_j$, temos:

$$y_j/a_j \leq \mu_j \rightarrow y_j \leq a_j \mu_j$$

(i) Quando $a_1\mu_1 > b$ concluimos que $y_1=b$ e $y_j=0$, $j \neq 1$ pois:

(1º) y_1 é a variável correspondente ao maior coeficiente, já que $c_1/a_1 \geq \ldots \geq c_n/a_n$

(2º) Fazendo $j=1 \rightarrow y_1 \leq a_1 u_1$

Logo: $\left[\begin{array}{l} y_1 \leq a_1\mu_1 \\ a_1\mu_1 > b \end{array}\right. \rightarrow y_1=b$ e $y_j=0$, $j \neq 1$

Expressando esse resultado em termo de x_j, vem:

$X = (b/a_1, 0, \ldots, 0)$, pois $y_1=x_1a_1 \rightarrow b=x_1a_1 \rightarrow x_1=b/a_1$

(ii) Quando $a_1\mu_1 \leq b$, a solução ótima do programa linear é encontrada fazendo-se $y_j=a_j\mu_j$ $(j=1,\ldots,t \geq 1)$, $y_{t+1} = b - \sum_{j=1}^{t} a_j\mu_j$ e $y_j=0$ $(j=t+2,\ldots,n)$. Este raciocínio evita folga na mochila, já que $a_1\mu_1 \leq b$. Assim sendo t é escolhido de tal forma que seja o menor índice tal que:

$$b - \sum_{j=1}^{t+1} a_j\mu_j < 0$$

Expressando esse resultado em termos de x_j:

$$X = (\mu_1, \ldots, \mu_t, (1/a_{t+1})(b - \sum_{j=1}^{t} a_j\mu_j), 0, \ldots, 0) \quad , \quad \text{pois}$$

(1º) $y_j = a_j\mu_j \rightarrow x_ja_j = a_j\mu_j \rightarrow x_j=\mu_j$

(2º) $y_{t+1} = b - \sum_{j=1}^{t} a_j\mu_j \rightarrow a_{t+1}\, x_{t+1} = b - \sum_{j=1}^{t} a_j\mu_j$

$$\rightarrow x_{t+1} = (1/a_{t+1})(b - \sum_{j=1}^{t} a_j\mu_j)$$

<u>2º caso</u>: Existe limite inferior inteiro positivo

Se o limite inferior inteiro positivo μ_j existe, então uma desigualdade da forma $x_j \geq \mu_j$ pode ser tratada implicitamente introduzindo uma variável de complemento $\bar{x}_j = x_j - \mu_j \geq 0$ substituindo x_j por $\mu_j + \bar{x}_j$ na restrição do problema mochila e na função objetivo. Isto simplesmente reduz b de $a_j\mu_j$. Se $b - a_j\mu_j < 0$ o problema não tem solução, visto que $a_j > 0$, $\forall_j$. Se $b - a_j\mu_j \geq 0$ nós temos um problema mochila padrão e um programa linear associado. Vejamos:

A restrição original é:

$$a_1x_1 + \ldots + a_jx_j + \ldots + a_nx_n = b$$

Fazendo $x_j = \mu_j + \bar{x}_j$, temos:

$$a_1x_1 + \ldots + a_j\mu_j + a_j\bar{x}_j + \ldots + a_nx_n = b$$

$$a_1x_1 + \ldots + a_j\bar{x}_j + \ldots + a_nx_n = b - a_j\mu_j$$

Sendo $b - a_j\mu_j \geq 0$ a solução do programa associado é:

$$x_1 = (b - a_j\mu_j)/a_1$$

$$\bar{x}_j = 0 \rightarrow x_j = \mu_j$$

$$x_k = 0 \text{ , } k \neq 1 \text{ e } k \neq j$$

Mostramos então que um problema mochila com ou sem variáveis limitadas pode ser resolvido como um programa linear por inspeção. Então um procedimento natural para resolver o programa inteiro é por uma enumeração de "branch and bound". Se nós adotamos o "Dakin branch and bound" o valor de $x_1 = b/_{a_1}$ é inicialmente examinado. Se ele é inteiro, o problema mochila está resolvido. Caso contrário dois nós são criados - o primeiro por introduzir a restrição $x_1 \leq [b/a_1]$ e o segundo impondo a restrição $x_1 \geq [b/a_1] + 1$. O programa linear para cada um desses nós é resolvido por inspeção, e o processo continua. Isto é, nós são criados sempre que eles podem produzir uma solução inteira melhorada; por exemplo, quando a solução do PPL não é inteira e seus valores excedem o valor da melhor solução inteira até então. Como $x_1 = [b/a_1]$, $x_j=0$ $(j \neq 1)$ é uma solução inteira, um limite inferior inicial na função objetivo é $c_1[b/a_1]$. Também, observe que se a solução do PPL não é inteira, há exatamente uma variável (denominada x_1 ou x_1+1) a qual é fracionária e então é escolhida para criar nós.

Façamos agora uma relação dos critérios a serem utilizados na solução de um problema mochila, tomando por base o estudo feito anteriormente. A enumeração desses critérios visa esclarecer ainda mais a técnica desenvolvida nesta seção.

(1) <u>INÍCIO</u>: Fazer $X = (b/a_1, 0, 0, \ldots, 0)$

Obter o limite inferior inicial para a função objetivo.

<u>I M P O R T A N T E</u>: Se x_1 é inteiro o problema mochila está resolvido.

(2) <u>LIMITES DA VARIÁVEL FRACIONÁRIA</u>

Considera-se $\left[\begin{array}{l} x_j \leq [x_i] \\ x_j \geq [x_i] + 1 \end{array}\right.$, onde i é o índice

da variável fracionária no nó anterior

(3) OBTENÇÃO DE UM NÓ RESULTANTE DE UMA EXPANSÃO

(i) Fazer o retrospectro dos possíveis valores $x_i^{'}$s para ver se há viabilidade;

(ii) Se existe limite superior inteiro μ_j , isto é, para $x_j \leq \mu_j$, tem-se que:

$$X = (b/a_1, 0, \ldots, 0) \text{ se } a_1 u_1 > b$$

$$X = (u_1, \ldots, u_t, (1/a_{t+1})(b - \sum_{j=1}^{t} a_j \mu_j), 0, \ldots, 0) \text{ se } a_1 \mu_1 \leq b$$

(iii) Se existe limite inferior inteiro μ_j , isto é, para $x_j \geq \mu_j$, tem-se que:

$$x_1 = (b - a_j \mu_j)/a_1$$

$$x_j = \mu_j$$

$$x_k = 0 \quad , k \neq 1 \text{ e } k \neq j$$

OBS.: CASO UMA VARIÁVEL x_k, $k \notin \{1,j\}$, JÁ ESTEJA DETERMINADA QUANDO DA OBTENÇÃO DE UM NÓ, O VALOR PARA x_1 PASSA A SER:

$$x_1 = (b - a_j \mu_j - a_k x_k)/a_1$$

CASO HAJA MAIS DE UMA VARIÁVEL JÁ DETERMINADA, EXISTIRÁ MAIS DE UMA PARCELA $a_k x_k$ A SER SUBTRAÍDA.

(VER OBTENÇÃO DO NÓ 17, NO EXEMPLO RESOLVIDO ADIANTE, NO QUAL ESSE RACIOCÍNIO SE FAZ NECESSÁRIO).

(4) CRITÉRIO PARA FORMAÇÃO DA LISTA DE NÓS ATIVOS

Após obtermos a F.O.L. e a F.O.I* correspondentes a um determinado NÓ, ele será classificado como NÓ ATIVO se:

F.O.L. > F.O.I*

Obs.:

Chamamos de F.O.L o valor da função objetivo correspondente ao valor de X obtido em um nó.

Chamamos de F.O.I o valor da função objetivo obtida tomando-se o máximo inteiro contido em cada x_i de X.

Chamamos de de F.O.I* o máximo valor das F.O.I até o nó considerado.

C O M E N T Á R I O:

Como os nós ativos são aqueles que poderão ser expandidos, ou seja, aqueles que poderão oferecer uma melhor solução para o problema, torna-se claro o critério usado para formação da lista de nós ativos, já que se um nó oferece uma F.O.L inferior a F.O.I*, ele nada acrescenta em termos de solução mais favorável.

(5) CRITÉRIO DE CORTES NA LISTA DE NÓS ATIVOS

A proporção que os nós se sucedem, a F.O.I* cresce em valor e determinados nós ativos perdem em interesse. Assim sendo a lista deve ser a cada iteração submetida a cortes. Todo nó cujo máximo inteiro contido em F.O.L. for igual ou menor que F.O.I*, não pode oferecer soluções mais favoráveis que esta, e deve então ser desprezado. Ou seja, todo nó tal que verifique a desigualdade abaixo será cortado.

F.O.L. - F.O.I* < 1

Obs.: a afirmação sublinhada deve-se ao fato dos custos serem inteiros.

(6) CRITÉRIO PARA ESCOLHA DO PRÓXIMO NÓ A SER EXPANDIDO

"Escolha o mais recente nó criado pertencente a lista dos nós ativos com a mais alta solução linear."

(7) CRITÉRIO DE PARADA

Não há mais nó ativo

EXEMPLO: (Hu,T.; Integer Programming and Network Flows)

MAXIMIZAR $18x_1 + 14x_2 + 8x_3 + 4x_4$

sujeito a: $15x_1 + 12x_2 + 7x_3 + 4x_4 \leq 33$

$x_1, x_2, x_3, x_4 \geq 0$ e inteiros

SOLUÇÃO

1) INÍCIO:

Obtenção do nó 1: $x_1 = b/a_1 \rightarrow x_1 = 33/15$

$X = (33/15,0,0,0)$; F.O.L $= 39\frac{3}{5}$; F.O.I $= 36$

2) EXPANSÃO DO NÓ 1:

$x_1 \leq [33/15] \rightarrow x_1 \leq 2 \rightarrow$ NÓ 2

$x_1 \geq [33/15] + 1 \rightarrow x_1 \geq 3 \rightarrow$ NÓ 3

Obtenção do nó 2: $x_1 \leq 2$

Existe o limite superior inteiro $\mu_1 = 2$ e $a_1\mu_1 < b$. Assim:

$X = (\mu_1, \frac{1}{a_2}(b-a_1\mu_1),0,0) \rightarrow X = (2,1/4,0,0)$

F.O.L $= 39\frac{1}{2}$; F.O.I $= 36$; F.O.I* $= 36$

<u>Obtenção do Nó 3</u>: $x_1 \geq 3$

INVIÁVEL

3) <u>EXPANSÃO DO NÓ 2</u>:

$x_2 \leq [1/4] \rightarrow x_2 \leq 0 \rightarrow$ NÓ 4

$x_2 \geq [1/4] \rightarrow x_2 \geq 1 \rightarrow$ NÓ 5

<u>Obtenção do nó 4</u>: $x_2=0$; $x_1 \leq 2$

Existe o limite superior inteiro $\mu_2=0$ e $a_1\mu_1 < b$.

Assim:

$$X = (\mu_1, \mu_2, \frac{1}{a_3}(b - \sum_{j=1}^{2} a_j\mu_j), 0) \rightarrow X = (2,0,3/7,0)$$

F.O.L = $39\frac{3}{7}$; F.O.I = 36 ; F.O.I* = 36

<u>Obtenção do nó 5</u>: $x_2 \geq 1$, $x_1 \leq 2$

Existe limite inferior inteiro $\mu_2=1$. Então:

$$x_1 = (b - a_2\mu_2)/a_1 \rightarrow x_1(33-12.1)/15 \rightarrow x_1 = 21/15$$

$x_2 = \mu_2 \rightarrow x_2=1$ e $x_3=x_4=0$. Assim: $X = (\frac{21}{15},1,0,0)$

F.O.L = $39\frac{1}{5}$; F.O.I = 32 ; F.O.I* = 36

PELO CRITÉRIO DE ESCOLHA DOS NÓ, O NÓ 4 SERÁ EXPANDIDO

4) <u>EXPANSÃO DO NÓ 4</u>:

$x_3 \leq [3/7] \rightarrow x_3 \leq 0 \rightarrow$ NÓ 6

$x_3 \geq [3/7] + 1 \rightarrow x_3 \geq 1 \rightarrow$ NÓ 7

<u>Obtenção do nó 6</u>: $x_3=0$, $x_2=0$, $x_1 \leq 2$

Existe limite superior inteiro $\mu_3=0$ e $a_1\mu_1 < b$.

Assim:

$$X = (\mu_1,\mu_2,\mu_3, \frac{1}{a_4}(b-\sum_{j=1}^{3} a_j\mu_j)) \rightarrow X = (2,0,0,3/4)$$

F.O.L = 39 ; F.O.I = 36 ; F.O.I* = 36

<u>Obtenção do nó 7</u>: $x_3 \geq 1$, $x_2=0$, $x_1 \leq 2$

Existe limite inferior inteiro $\mu_3=1$. Então:

$x_1 = (b - a_3\mu_3)/a_1 \rightarrow x_1 = (33 - 7.1)/15 \rightarrow x_1 = 26/15$

$x_3 = \mu_3 \rightarrow x_3=1$ e $x_2=x_4=0$. Assim: $X=(\frac{26}{25},0,1,0)$

F.O.L = $39\frac{1}{5}$; F.O.I = 36 ; F.O.I* = 36

PELO CRITÉRIO DE ESCOLHA DOS NÓS, O NÓ 7 SERÁ EXPANDIDO

5) <u>EXPANSÃO DO NÓ 7</u>:

$x_1 \leq [26/15] \rightarrow x_1 \leq 1 \rightarrow$ NÓ 8

$x_1 \geq [26/15] + 1 \rightarrow x_1 \geq 2 \rightarrow$ NÓ 9

<u>Obtenção do nó 8</u>: $x_1 \leq 1$, $x_3 \geq 1$, $x_2=0$

Existe limite superior inteiro $\mu_1=1$ e $a_1\mu_1 < b$.

Assim:

$$X = (\mu_1,\mu_2, \frac{1}{a_3}(b-\sum_{j=1}^{2} a_j\mu_j),0) \rightarrow X = (1,0,18/7,0)$$

F.O.L. = $38\frac{4}{7}$; F.O.I = 34 ; F.O.I* = 36

Obtenção do nó 9: $x_1=2$, $x_3 \geq 1$, $x_2=0$

INVIÁVEL

PELO CRITÉRIO DE ESCOLHA DOS NÓS, O NÓ 5 SERÁ EXPANDIDO

6) EXPANSÃO DO NÓ 5:

$x_1 \leq [21/15] \rightarrow x_1 \leq 1 \rightarrow$ NÓ 10

$x_1 \geq [21/15] + 1 \rightarrow x_1 \geq 2 \rightarrow$ NÓ 11

Obtenção do NÓ 10: $x_1 \leq 1$, $x_2 \geq 1$

Existe limite superior inteiro $\mu_1=1$ e $a_1\mu_1 < b$.

Assim:

$$X = (\mu_1, \frac{1}{a_2}(b-a_1\mu_1),0,0) \rightarrow X = (1,3/2,0,0)$$

F.O.L = 39 ; F.O.I = 32 ; F.O.I* = 36

Obtenção do NÓ 11: $x_1=2$, $x_2 \geq 1$

INVIÁVEL

PELO CRITÉRIO DE ESCOLHA DOS NÓS, O NÓ 10 SERÁ EXPANDIDO

7) EXPANSÃO DO NÓ 10:

$x_2 \leq [3/2] \rightarrow x_2 \leq 1 \rightarrow$ NÓ 12

$x_2 \geq [3/2] + 1 \rightarrow x_2 \geq 2 \rightarrow$ NÓ 13

Obtenção do NÓ 12: $x_1 \leq 1$, $x_2=1$

Existe liminte superior inteiro $\mu_2=1$ e $a_1\mu_1 < b$.

Assim:

$$X = (\mu_1, \mu_2, \frac{1}{a_3}(b - \sum_{j=1}^{2} a_j\ _j), 0) \rightarrow X = (1, 1, 6/7, 0)$$

$F.O.L = 38\frac{6}{7}$; $F.O.I = 32$; $F.O.I^* = 36$

Obtenção do nó 13: $x_1 \leq 1$, $x_2 \geq 2$

Existe limite inferior inteiro $\mu_2 = 2$

$x_1 = (b - a_2\mu_2)/a_1 \rightarrow x_1 = (33 - 12.2)/15 \rightarrow x_1 = 3/5$

$x_2 = \mu_2 \rightarrow x_2 = 2$ e $x_3 = x_4 = 0$. Assim: $X = (3/5, 2, 0, 0)$

$F.O.L = 38\frac{4}{5}$; $F.O.I = 28$; $F.O.I^* = 36$

PELO CRITÉRIO DE ESCOLHA DOS NÓS, O NÓ 6 SERÁ EXPANDIDO

8) EXPANSÃO DO NÓ 6:

$x_4 \leq [3/4] \rightarrow x_4 \leq 0 \quad \rightarrow$ NÓ 14

$x_4 \geq [3/4] + 1 \rightarrow x_4 \geq 1 \quad \rightarrow$ NÓ 15

Obtenção do NÓ 14: $x_4 = 0$, $x_1 = 0$, $x_2 = 0$, $x_1 \leq 2$

Existe limite superior inteiro μ_4 e $a_1\mu_1 < b$.

Assim:

$X = (\mu_1, \mu_2, \mu_3, \mu_4) \rightarrow X = (2, 0, 0, 0)$

$F.O.L = 36$; $F.O.I = 36$; $F.O.I^* = 36$

Obtenção do NÓ 15: $x_4 \geq 1$, $x_3 = 0$, $x_2 = 0$, $x_1 \leq 2$

Existe limite inferior inteiro $\mu_4 = 1$

$x_1 = (b - a_4\mu_4)/a_1 \rightarrow x_1 = (33 - 4.1)/15 \rightarrow x_1 = 29/15$

$x_2 = x_3 = 0$ e $x_4 = \mu_4 = 1$. Assim: $X = (29/15, 0, 0, 1)$

$F.O.L = 38\frac{4}{5}$; $F.O.I = 22$; $F.O.I^* = 36$

PELO CRITÉRIO DE ESCOLHA DOS NÓS, O NÓ 12 SERÁ EXPANDIDO

9) <u>EXPANSÃO DO NÓ 12</u>:

$x_3 \leq [6/7] \rightarrow x_3 \leq 0 \quad \rightarrow$ NÓ 16

$x_3 \geq [6/7] + 1 \rightarrow x_3 \leq 1 \quad \rightarrow$ NÓ 17

<u>Obtenção do NÓ 16</u>: $x_3=0$, $x_2=1$, $x_1 \leq 1$

Existe limite superior inteiro $\mu_3=0$ e $a_1\mu_1 < b$.

Assim:

$$(\mu_1, \mu_2, \mu_3, \frac{1}{a_4}(b-\sum_{j=1}^{3} a_j\mu_j)) \rightarrow X = (1,1,0,3/2)$$

$F.O.L = 38$; $F.O.I = 32$; $F.O.I^* = 36$

<u>Obtenção do NÓ 17</u>: $x_3 \geq 1$, $x_1 \leq 1$, $x_2=1$

Existe limite inferior inteiro $\mu_3=1$

$$x_1 = (b - a_3\mu_3)/a_1 \rightarrow x_1 = (33-7.1)/15 \rightarrow x_1 = 26/15$$

Entretanto, como $x_2=1$, temos:

$$x_1 = \frac{26 - a_2x_2}{15} \rightarrow x_1 = 14/15$$

Assim:

$$x_1 = 14/15 \; ; \; x_2 = 1 \; ; \; x_3 = \mu_3 = 1 \; ; \; x_4 = 0 \rightarrow X=(14/15,1,1,0)$$

$F.O.L = 38\frac{4}{5}$; $F.O.I = 22$; $F.O.I^* = 36$

PELO CRITÉRIO DE ESCOLHA DE NÓS, O NÓ 17 SERÁ EXPANDIDO

10) EXPANSÃO DO NÓ 17:

$x_1 \leq [14/15] \rightarrow x_1 \leq 0 \qquad \rightarrow$ NÓ 18

$x_1 \geq [14/15] + 1 \rightarrow x_1 \geq 1 \quad \rightarrow$ NÓ 19

Obtenção do NÓ 18: $x_1=0$; $x_3 \geq 1$; $x_2=1$

Existe limite superior $\mu_1=0$ e $a_1\mu_1 < b$. Assim:

$$X = (\mu_1, \mu_2, \frac{1}{a_3}(b - \sum_{j=1}^{2} a_j\mu_j), 0) \rightarrow X = (0,1,3,0)$$

F.O.L = 38 ; F.O.I = 38 ; F.O.I* = 38

OBS.: (1) DE ACORDO COM OS VALORES ENCONTRADO PARA F.O.L e F.O.I*, ESSE NÓ INGRESSA NA LISTA DE NÓS ATIVOS.

(2) COMO F.O.I* = 38 E AS F.O.L ANTERIORES SÃO TAIS QUE F.O.L - F.O.I* < 1 , OS NÓS QUE CONSTAM DA LISTA DE NÓS ATIVOS SÃO CORTADOS.

Obtenção do NÓ 19: $x_1=1$, $x_2=1$, $x_3 \geq 1$

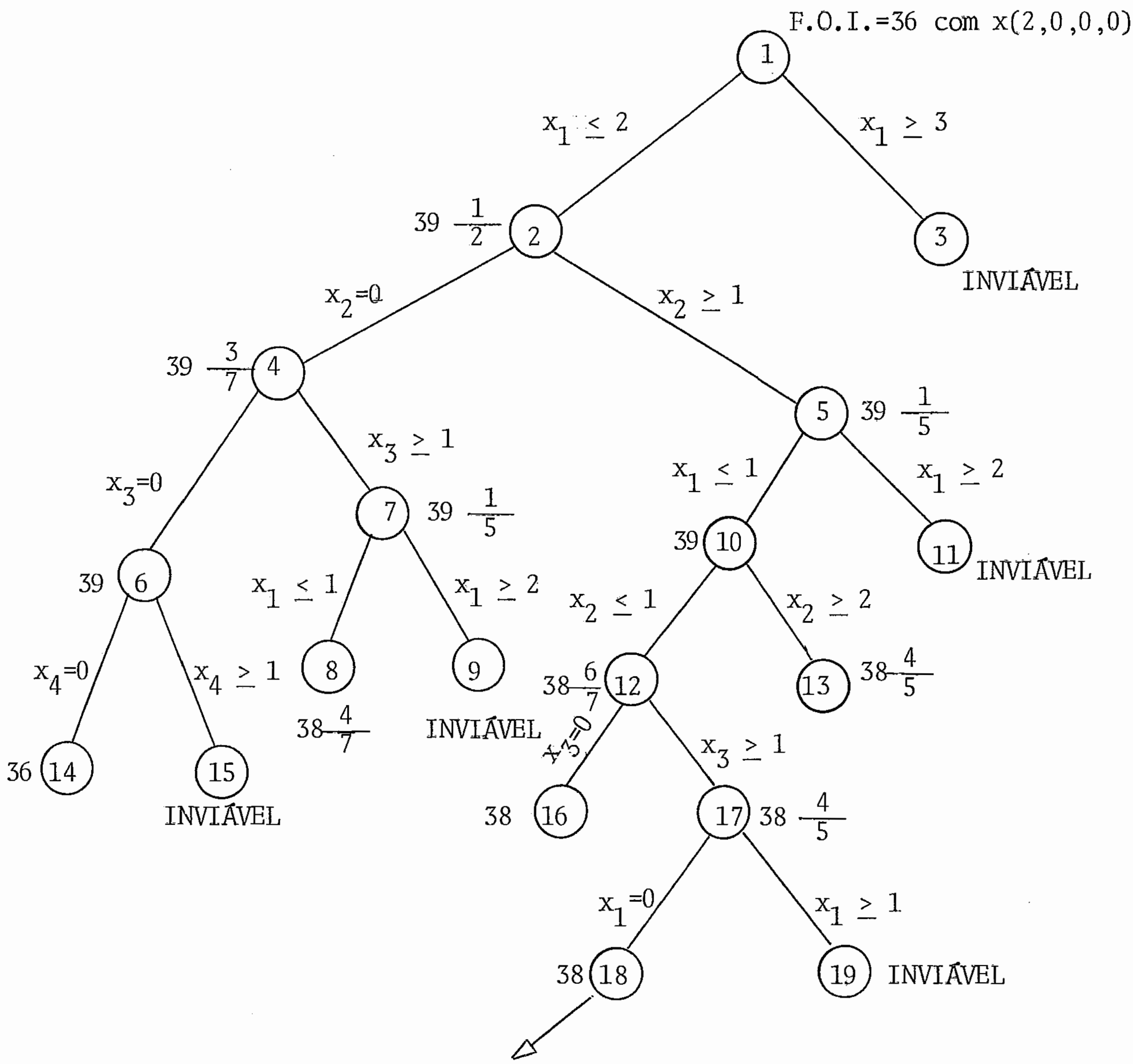
INVIÁVEL

COMO A LISTA DE NÓS ATIVOS ESTÁ VAZIA, MAIS NENHUM NÓ HÁ PARA SER EXPANDIDO. A SOLUÇÃO ÓTIMA FOI ENCONTRADA NO NÓ 18.

F.O. = 38

$x_1 = 0$; $x_2 = 1$; $x_3 = 3$; $x_4 = 0$

ARBORESCÊNCIA ASSOCIADA ÀS COMPUTAÇÕES



SOLUÇÃO INTEIRA ÓTIMA
X=(0,1,3,0) com F.O=38

3. Método dos Multiplicadores de Lagrange

Um resultado devido a Everett | 9 | é agora apresentado permitindo-nos resolver um programa inteiro, e então um problema mochila impondo a exigência da integridade e não negatividade das variáveis.

TEOREMA: Seja o problema linear inteiro:

$$\begin{array}{ll} \text{MAXIMIZAR} & f(\underline{x}) \\ \text{sujeito a:} & g(\underline{x}) \leq b \qquad , \ \underline{x} = (x_1, x_2, \ldots, x_n) \\ & \underline{x} \geq 0 \text{ e inteiro} \end{array}$$

Seja $\lambda \geq 0$. Se $\underline{x}^o$ resolve o problema:

$$\begin{array}{ll} \text{MAXIMIZAR} & f(\underline{x}) - \lambda\, g(\underline{x}) \\ \text{sujeito a:} & \underline{x} \geq 0 \text{ e inteiro} \quad , \end{array}$$

então ele também resolve o problema:

$$\begin{array}{ll} \text{MAXIMIZAR} & f(\underline{x}) \\ \text{sujeito a:} & g(\underline{x}) \leq g(\underline{x}^o) \\ & \underline{x} \geq 0 \text{ e inteiro} \end{array}$$

OBS.: Se λ é encontrado tal que $g(\underline{x}^o) = b$, o problema original está resolvido, com $\underline{x}^o$ sendo o ÓTIMO.

DEMONSTRAÇÃO:

Se $\underline{x}^o$ maximiza a FUNÇÃO LAGRANGEANA $f(\underline{x}) - \lambda\, g(\underline{x})$, então:

$$f(\underline{x}^o) - \lambda\, g(\underline{x}^o) \geq f(\underline{x}) - \lambda\, g(\underline{x}) \quad , \quad \forall \underline{x} \geq 0$$

Logo:

$$f(\underline{x}^o) \geq f(\underline{x}) + \lambda [g(\underline{x}^o) - g(\underline{x})] \quad , \quad \forall \underline{x} \geq 0$$

Se a desigualdade acima é verdadeira para todo $\underline{x} \geq 0$, então ela é satisfeita para $g(\underline{x}^o) - g(\underline{x}) \geq 0$.

Sendo $\lambda \geq 0$ e como $g(\underline{x}^o) - g(\underline{x}) \geq 0$ satisfaz a desigualdade anterior, concluímos que:

$$f(\underline{x}^o) \geq f(\underline{x}) \quad , \quad \forall \underline{x} \geq 0$$

Como vimos acima $f(\underline{x}^o) \geq f(\underline{x})$ é satisfeita para $g(\underline{x}^o) - g(\underline{x}) \geq 0$, ou seja: $f(\underline{x}^o) \geq f(\underline{x})$ mesmo que $g(\underline{x}^o) \geq g(\underline{x})$. Ou como queríamos demonstrar, $\underline{x}_o$ satisfaz o problema:

$$\begin{array}{ll} \text{MAXIMIZAR} & f(\underline{x}) \\ \text{sujeito a:} & g(\underline{x}) \leq g(\underline{x}^o) \\ & \underline{x} \geq 0 \text{ e inteiro} \quad \text{(c.q.d)} \end{array}$$

Se o multiplicador λ é selecionado tal que $b = g(\underline{x}^o)$, nós podemos resolver o programa inteiro sem a restrição desigualdade. Este problema é consideravelmente mais fácil para resolver.

A dificuldade com a aproximação LAGRANGEANA é que nem sempre é possível encontrar λ tal que $g(x^o) = b$. Everett provou esta afirmativa e demonstrou um Teorema que nos garante o seguinte: SE $\underline{x}^*$ É PRÓXIMO DO ÓTIMO DA LAGRANGEANA, $\underline{x}^*$ TAMBÉM É PRÓXIMO DO ÓTIMO DO PROBLEMA ORIGINAL COM RESTRIÇÃO.

Esse resultado nos permite formular o seguinte algoritmo:

<u>PASSO 1</u>: Escolha $\lambda \geq 0$

<u>PASSO 2</u>: Ache o $\underline{x}$ que maximiza a FUNÇÃO LAGRANGEANA para este valor de λ

PASSO 3: Se $g(\underline{x})$ é próximo SUFICIENTEMENTE de b, PARE. Senão escolha novo $\lambda \geq 0$ e volte ao passo 2.

Um processo de solução por inspeção

De acordo com o teorema estudado nosso problema é:

$$\text{MAXIMIZAR} \quad \sum_{j=1}^{n} c_j\, x_j - \lambda \sum_{j=1}^{n} a_j\, x_j$$

$$x_j \geq 0 \quad \text{e inteiro} \quad , \quad j=1,\ldots,n$$

ou então:

$$\text{MAXIMIZAR} \quad \sum_{j=1}^{n} (c_j - \lambda\, a_j)x_j$$

$$x_j \geq 0 \quad \text{e inteiro} \quad , \quad j=1,\ldots,n$$

Como nosso objetivo é maximizar a expressão $(c_1 - \lambda\, a_1)x_1 + \ldots + (c_n - \lambda\, a_n)x_n$ podemos fazer:

(i) $x_j=0$ se $c_j - \lambda\, a_j < 0$

(ii) $x_j=\mu_j$ se $c_j - \lambda\, a_j > 0$, onde μ_j é o máximo valor que x_j assumir

(iii) $x_j \in [0,\mu_j]$ se $c_j - \lambda\, a_j = 0$ (isto é devido ao fato de, neste caso, o valor de x_j não interferir na função objetivo)

A partir daí apenas dois fatos passam a interessar: a) o sinal de $c_j - \lambda\, a_j$

b) o máximo valor que x_j pode assumir

a) Para definir o sinal de $c_j - \lambda\, a_j$ precisamos avaliar os valores de λ .

Suponhamos que c_j e a_j são positivos e que as variáveis estão ordenadas de tal forma que $c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$. Os valores assumidos por λ serão : 0, c_n/a_n, c_{n-1}/a_{n-1}, ..., c_1/a_1, onde c_1/a_1 é o máximo valor que λ assume. Como $\underline{x}$ varia somente quando $c_j - \lambda a_j = 0$, ou seja, quando $\lambda = c_j/a_j$, o valor para $\underline{x}$ permanece constante nos intervalos:

$$0 \leq \lambda < c_n/a_n \;,\; c_n/a_n \leq \lambda < c_{n-1}/a_{n-1}, \ldots, c_2/a_2 \leq \lambda < c_1/a_1, c_1/a_1 \leq \lambda < +\infty$$

Assim obtemos $\underline{x}$ para $\lambda = 0$ e sabemos então que este $\underline{x}$ se conserva constante até que λ assuma o valor c_n/a_n, implicando então em um novo valor para o vetor $\underline{x}$, que se conservará constante até que λ assuma o valor c_{n-1}/a_{n-1}, e assim por diante. Como dissemos a pouco, os únicos valores a serem testados se correspondem com os seguintes valores assumidos por λ : 0, $c_n/a_n, \ldots, c_2/a_2$, c_1/a_1.

b) <u>Valores máximo μ_j que os x_j podem assumir</u>

Caso $c_j - \lambda a_j > 0$ então $x_j = \mu_j$ que é o maior valor possível para x_j. Vejamos uma forma de estimá-lo.

Se os limites superiores μ_j não são conhecidos, eles podem ser tomados como $[b/a_j]$. Entretanto ao considerarmos $\sum a_j x_j^o$, este valor pode ser consideravelmente maior que b. Acrescido do fato de que x_j pode tomar qualquer valor desde o menor até o maior, se $c_j - \lambda a_j = 0$, então aquele somatório cresce ainda mais. Desta feita os multiplicadores de Lagrange são normalmente usados quando são conhecidos os valores máximos das variáveis, por exemplo $x_j \in \{0,1\}$

EXEMPLO

MAXIMIZAR $15x_1 + 13x_2 + 7x_3 + 5x_4$

sujeito a: $17x_1 + 11x_2 + 7x_3 + 4x_4 \leq 37$

$x_1, x_2, x_3, x_4 \in \{0,1\}$

Ordenando as variáveis, temos:

MAXIMIZAR $5x_1 + 13x_2 + 7x_3 + 15x_4$

sujeito a: $4x_1 + 11x_2 + 7x_3 + 17x_4 \leq 37$

$x_1, x_2, x_3, x_4 \in \{0,1\}$

Valores assumidos por λ : 0,15/17, 7/7, 13/11, 5/4

(1) Para $\lambda = 0$ ou para $\lambda \in [0,15/17)$

SINAL DE $c_j - \lambda\, a_j$: sempre positivo

Valores dos x_j : $x_1=x_2=x_3=x_4=1 \rightarrow \underline{x} = (1,1,1,1)$

$$\sum_{j=1}^{4} a_j x_j = 4.1+11.1+7.1+17.1 = 39$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-39 = -2$$

$$\sum_{j=1}^{4} c_j x_j = 5.1+13.1+7.1+15.1 = 40$$

(2) Para $\lambda = 15/17$ ou para $\lambda \in [15/17,1)$

Sinal de $c_j - \lambda a_j$: positivo para $j=1,2,3$

nulo para $j=4$

Valores dos x_j : $x_1=x_2=x_3=1$ $\underset{\sim}{x} = (1,1,1,0)$

$x_4=0$ ou 1 ou

$\underset{\sim}{x} = (1,1,1,1)$

$$\sum_{j=1}^{4} a_j x_j = 39 \text{ para } \underset{\sim}{x} = (1,1,1,1)$$

$$\sum_{j=1}^{4} a_j x_j = 22 \text{ para } \underset{\sim}{x} = (1,1,1,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-39 = -2 \text{ para } \underset{\sim}{x} = (1,1,1,1)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-22 = 15 \text{ para } \underset{\sim}{x} = (1,1,1,0)$$

$$\sum_{j=1}^{4} c_j x_j = 5.1+13.1+7.1+15.1 = 40 \text{ para } \underset{\sim}{x} = (1,1,1,1)$$

$$\sum_{j=1}^{4} c_j x_j = 5.1+13.1+7.1+15.0 = 25 \text{ para } \underset{\sim}{x} = (1,1,1,0)$$

(3) Para $\lambda = 1$ ou para $\lambda \in [1,13/11)$

Sinal de $c_j - \lambda a_j$: positivo para $j=1,2$

nulo para $j=3$

negativo para $j=4$

Valores dos x_j : $$\left[\begin{array}{l} x_1=x_2=1 \\ x_3=0 \text{ ou } 1 \\ x_4=0 \end{array}\right. \rightarrow \left[\begin{array}{l} \underset{\sim}{x} = (1,1,0,0) \\ \text{ou} \\ \underset{\sim}{x} = (1,1,1,0) \end{array}\right.$$

$$\sum_{j=1}^{4} a_j x_j = 22 \quad \text{para} \quad \underset{\sim}{x} = (1,1,1,0)$$

$$\sum_{j=1}^{4} a_j x_j = 15 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-22 = 15 \quad \text{para} \quad \underset{\sim}{x} = (1,1,1,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-15 = 22 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

$$\sum_{j=1}^{4} c_j x_j = 25 \quad \text{para} \quad \underset{\sim}{x} = (1,1,1,0)$$

$$\sum_{j=1}^{4} c_j x_j = 18 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

(4) Para $\lambda = 13/11$ ou para $\lambda \in [13/11, 5/4]$

Sinal de $c_j - \lambda a_j$: $\left[\begin{array}{l} \text{positivo para } j=1 \\ \text{nulo para } j=2 \\ \text{negativo para } j=3,4 \end{array}\right.$

Valores dos x_j: $\left[\begin{array}{l} x_1=1 \\ x_2=0 \text{ ou } 1 \\ x_3=x_4=0 \end{array}\right. \rightarrow \left[\begin{array}{l} \underset{\sim}{x} = (1,0,0,0) \\ \qquad \text{ou} \\ \underset{\sim}{x} = (1,1,0,0) \end{array}\right.$

$$\sum_{j=1}^{4} a_j x_j = 4 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

$$\sum_{j=1}^{4} a_j x_j = 15 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-4 = 33 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-15 = 22 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

$$\sum_{j=1}^{4} c_j x_j = 18 \quad \text{para} \quad \underset{\sim}{x} = (1,1,0,0)$$

$$\sum_{j=1}^{4} c_j x_j = 5 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

(5) Para $\lambda = 5/4$ ou para $\lambda \varepsilon [5/4,+\infty]$

Sinal de $c_j - \lambda\, a_j$: $\left[\begin{array}{l}\text{nulo para } j=1\\ \text{negativo para } j=2,3,4\end{array}\right.$

Valores dos x_j : $\left[\begin{array}{l} x_1=0 \text{ ou } 1 \\ \\ x_2=x_3=x_4=0 \end{array}\right. \rightarrow \left[\begin{array}{l} \underset{\sim}{x} = (0,0,0,0) \\ \qquad \text{ou} \\ \underset{\sim}{x} = (1,0,0,0) \end{array}\right.$

$$\sum_{j=1}^{4} a_j x_j = 0 \quad \text{para} \quad \underset{\sim}{x} = (0,0,0,0)$$

$$\sum_{j=1}^{4} a_j x_j = 4 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-0 = 37 \quad \text{para} \quad \underset{\sim}{x} = (0,0,0,0)$$

$$b - \sum_{j=1}^{4} a_j x_j = 37-4=33 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

$$\sum_{j=1}^{4} c_j x_j = 0 \quad \text{para} \quad \underset{\sim}{x} = (0,0,0,0)$$

$$\sum_{j=1}^{4} c_j x_j = 5 \quad \text{para} \quad \underset{\sim}{x} = (1,0,0,0)$$

SOLUÇÕES ENCONTRADAS

$$F(x) = 40 \quad \text{com} \quad b - \sum_{j=1}^{4} a_j x_j = -2$$

$$F(x) = 25 \quad \text{com} \quad b - \sum_{j=1}^{4} a_j x_j = 15$$

$$F(x) = 18 \quad \text{com} \quad b - \sum_{j=1}^{4} a_j x_j = 22$$

$$F(x) = 5 \quad \text{com} \quad b - \sum_{j=1}^{4} a_j x_j = 33$$

$$F(x) = 0 \quad \text{com} \quad b - \sum_{j=1}^{4} a_j x_j = 37$$

Como verificamos existe a possibilidade de $\sum_j a_j x_j$ cair fora da região viável. No exemplo acima constatamos, de acordo com a solução apresentada pela técnica utilizada, que a solução mais favorável exige um aumento no tamanho da mochila de 37 para 40, o que tornaria a solução exata.

OBS.: ESTE PROCEDIMENTO É ESPECIALMENTE USADO QUANDO OS DADOS (em particular b) ESTÃO SENDO ESTIMADOS.

A melhor solução encontrada, tal que $b - \sum a_j x_j$ está mais próximo de ZERO, é:

$$F^*(x) = 40 \quad \text{com} \quad \underset{\sim}{x} = (1,1,1,1)$$

Apliquemos, agora a técnica dos multiplicadores de Lagrange na solução do exemplo abaixo, no qual abandonamos a exigência da positividade dos c_j e a_j.

MAXIMIZAR $-18x_1 + 14x_2 + 8x_3 + 4x_4$

sujeito a: $-15x_1 + 12x_2 + 7x_3 + 4x_4 \leq 33$

$x_j \in \{0,1\}$

Temos, então:

MAX $(-18+15\lambda)x_1 + (14-12\lambda)x_2 + (8-7\lambda)x_3 + (4-4\lambda)x_4$

sujeito a: $x_j \in \{0,1\}$

Identicamente ao problema anterior basta analisar:

$\lambda=0$; $\lambda=1$; $\lambda=8/7$; $\lambda=14/12$; $\lambda=18/15$

O quadro abaixo apresenta os valores encontrados:

	$\underset{\sim}{x}$	$\sum a_j x_j$	$b - \sum a_j x_j$	$\sum c_j x_j$
$0 \leq \lambda < 1$	(1,1,1,1)	8	25	8
$1 \leq \lambda < 8/7$	(1,1,1,0)	4	29	4
	(1,1,1,1)	8	25	8
$8/7 \leq \lambda < 14/12$	(1,1,0,0)	-3	36	-4
	(1,1,1,0)	4	29	4
$14/12 \leq \lambda < 18/15$	(1,0,0,0)	-15	48	-18
	(1,1,0,0)	- 3	36	- 4
$\lambda \geq 18/15$	(0,0,0,0)	0	33	0
	(1,0,0,0)	-15	48	-18

A melhor solução encontrada é $F(x) = 8$ com $\underset{\sim}{x} = (1,1,1,1)$. Entretanto ve-se claramente ser $\underset{\sim}{x} = (0,1,1,1)$ o ponto ótimo com $F(x)=26$. Este fato deixa clara a necessidade da positividade dos a_j e c_j para utilização do método descrito nessa secção.

4. Comentário sobre os métodos apresentados

O procedimento de programação dinâmica, embora seja consideravelmente mais eficiente quando não se apresenta limites às variáveis, tem como aspecto negativo a quantidade de valores que necessitam ser amazenados na memória, do computador. Mesmo no ALGORITMO II, onde há kg valores a menos para serem armazenados (k=nº de variáveis e g=tamanho da mochila), persiste este impasse. Portanto para problemas com muitas variáveis e b de tamanho expressivo esta técnica, de fácil programação e entendimento pelo usuário, torna-se INEFICIENTE COMPUTACIONALMENTE. O uso da propriedade periódica (caso do ALGORITMO III) quando possível, pode diminuir consideravelmente a quantidade de valores a serem armazenados, entretanto se o número k de variáveis for muito grande vai implicar em um bom tempo de processamento, o que torna o mecanismo ineficiente para problemas de grande porte.

Para problemas de pequeno porte esta técnica é altamente eficiente. Neste caso o ALGORITMO II é bastante eficaz. Se a propriedade periódica puder ser utilizada (o tamanho da mochila e o custo c_1 podem tornar nula essa vantagem) a eficiência aumenta.

No caso de problemas de médio porte o ALGORITMO III se puder ser usado não é de todo ineficaz; a não ser que o problema mochila em questão seja um subprograma bastante solicitado.

A técnica de "branch and bound" pelos mesmos motivos expostos acima deve apenas ser usada em problemas de pequeno porte e, talvez, de médio porte.

O método dos multiplicadores de Lagrange pode ser aproveitado no caso de ser suficiente uma solução aproximada. O tempo de processamento e a necessidade de armazenamento são bastante favoráveis. Torna-se ineficaz, entretanto, caso a solução encontrada caia fora da região viável e $\underline{b}$ não possa ser modificado para absorve-la. No caso de ser usado como técnica para um subprograma muito solicitado, a possibilidade da solução cair fora da região viável pode ser consideravelmente negativa.

Como pudemos observar, cada problema requer um cuidado especial na escolha da técnica de solução. No caso de problemas de grande porte a necessidade da busca de soluções aproximadas aplicáveis a cada caso surge até como um desafio. Podemos citar "The cutting stock problem" para o qual formularemos um algoritmo para obtenção de uma solução aproximada para os problemas mochila que surgem como subprogramas. Isto será feito em capítulo futuro.

CAPÍTULO III

TRANSFORMAÇÃO DE UM PROBLEMA DE PROGRAMAÇÃO INTEIRA COM m RESTRIÇÕES EM UM PROBLEMA MOCHILA

1. <u>O problema da transformação de um sistema de equações lineares em uma única equação linear</u>

Considere as "m" equações lineares:

$$\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ - - - - - - - - - - - - - - - - \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \end{array} \qquad (1)$$

com a_{ij} e b_i inteiros.

Multiplicando as "m" equações respectivamente por w_i , i=1,...,m , obtemos a equação matricial:

$$\begin{bmatrix} w_1a_{11} \\ w_2a_{21} \\ ---- \\ w_na_{m1} \end{bmatrix} x_1 + \begin{bmatrix} x_1a_{12} \\ w_2a_{12} \\ ---- \\ w_ma_{m2} \end{bmatrix} x_2 + \dots + \begin{bmatrix} w_1a_{1n} \\ w_2a_{2n} \\ ---- \\ w_na_{mn} \end{bmatrix} x_3 = \begin{bmatrix} w_1b_1 \\ w_2b_2 \\ ---- \\ w_nb_m \end{bmatrix}$$

Adicionando as "m" equações lineares resultantes desta equação matricial e expressando sob a forma de somatório, temos:

$$\sum_{j=1}^{n} \left(\sum_{i=1}^{m} w_i \; a_{ij} \right) . x_j = \sum_{i=1}^{m} w_i \; b_i \qquad (2)$$

Observamos que as equações (1) implicam (2), toda solução (e particularmente inteira e positiva) para (1) é uma solução para (2). Atribuindo valores arbitrários para os w_i's , contudo, o conjunto dos inteiros não negativos

$\underline{x} = (x_1,\ldots,x_n)$ satisfazendo 2 é geralmente maior que o conjunto satisfazendo 1.

Como exemplo citamos abaixo o problema constante em Balas | 2 |.

O problema inicial tinha três restrições désigualdades com cinco variáveis ZERO-UM. Para obter as equações adicionamos variáveis de folga não negativas x_6, x_7 e x_8, as quais devem ser inteiras se os coeficientes também inteiros. O resultado é:

$$
\begin{array}{ll}
\text{MINIMIZAR} & 5x_1 + 7x_2 + 10x_3 + 3x_4 + x_5 \\
\text{s.a.:} & -x_1 + 3x_2 - 5x_3 - x_4 + 4x_5 + x_6 = -2 \\
 & 2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 + x_7 = 0 \\
 & x_2 - 2x_3 + x_4 + x_5 + x_8 = -1 \\
 & x_j = 0 \text{ ou } 1 \quad (j=1,2,3,4,5) \\
 & x_j \geq 0 \text{ e inteiro } (j=6,7.8)
\end{array}
$$

Existem $2^5=32$ possíveis valores $\underline{x} = (x_1,x_2,x_3,x_4,x_5)$. (Arranjo completo de 2, 5 a 5. Exemplo: (0,0,0,1,0) e (0,0,0,0,1)). Por enumeração completa, as únicas soluções binárias são (0,1,1,0,0) e (1,1,1,0,0). Se nós arbitrariamente escolhemos os valores $w_1=1$, $w_2=1$ e $w_3=1$, teremos, aplicado (2):

$$x_1 - 2x_2 - 4x_3 + 2x_4 + 3x_5 + x_6 + x_7 + x_8 = -3$$

a qual tem as duas soluções ZERO-UM acima mais 5 outras, a saber: (0,0,1,0,0), (0,1,1,0,1), (0,1,1,1,0), (1,0,1,0,0) e (1,1,1,1,0).

Assim sendo nosso problema se resume em encontrar valores para os w_i's tal que cada solução inteira e positiva da equação linear resultante seja também solução do sistema que lhe deu origem.

Evidentemente, se o problema mochila resultante é mais fácil para resolver que o programa inteiro inicial e sua construção pode ser efetuada em um número razoável de tempo, a transformação é proveitosa.

2. Teorema de Mathews; Processo de agregação I

Este processo tem por base um teorema originário em Mathews |22|, o qual encunciaremos e provaremos a seguir.

Teorema:

Considere um sistema de duas equações:

$$s_1 \equiv \sum_{j=1}^{n} a_{1j}x_j = b_1 \qquad (E_1)$$

$$s_2 \equiv \sum_{j=1}^{n} a_{2j}x_j = b_2 \qquad (E_2)$$

com coeficientes inteiros a_{ij} $(i=1,2$ e $j=1,2,\ldots,n)$ e estritamente positivos.

(a) se existem valores não negativos $x_1,\ldots,x_n$ satisfazendo E_1 e E_2 , então:

$$b_2 a_{1j}/a_{2j} \geq b_1 \quad \text{para no mínimo um} \quad j(1 \leq j \leq n)$$

(b) se w é um inteiro positivo tal que

$$w > b_2 \ \text{máximo} \ \{a_{1j}/a_{2j}\}$$

então o conjunto solução de E_1 e E_2 em variáveis inteiras não negativas é o mesmo da única equação:

$$s_1 + w\ s_2 = b_1 + w\ b_2 \qquad (E_3)$$

Demonstração:

(a) Suponha b_2a_{ij}/a_{2j} para cada j. Como $a_{1j} > 0$ e $a_{2j} > 0$, para $x_j \geq 0$, temos que:

$$b_2a_{1j}x_j < b_1a_{2j}x_j$$

O somatório sobre j nos dá:

$$b_2 \sum_{j=1}^{n} a_{ij}x_j < b_1 \sum_{j=1}^{n} a_{2j}\ x_j \quad , \quad \text{para todo } x_j \geq 0$$

Mas, por hipótese, existe no mínimo uma solução não negativa, x_j^o, para E_1 e E_2, tal que:

$$\sum_{j=1}^{n} a_{1j}x_j^o = b_1 \quad \text{e} \quad \sum_{j=1}^{n} a_{2j}x_j^o = b_2$$

Substituindo-se na última inequação obtemos $b_2b_1 < b_1b_2$, o que é uma CONTRADIÇÃO.

(b) Para toda solução de E_1 e E_2 nós temos $s_1=b_1$ e $s_2=b_2$, o que significa que $ws_2=wb_2$ e $s_1+ws_2 = b_1 + wb_2$, ou seja, é também uma solução para E_3.

Para provar a recíproca (solução de $E_3 \rightarrow$ solução de E_1 e E_2), consideremos uma solução inteira não negativa, x_j^o, para E_3. Se esta solução x_j^o não é uma solução inteira para E_1 e E_2, então $s_2 \neq b_2$, pois se $s_2=b_2$ seguiria que $ws_2 = wb_2$, e subtraindo esta última de E_3 teríamos $s_1=b_1$, logo x_j^o seria solução inteira para E_1 e E_2.

Assim sendo se provarmos que uma solução inteira não negativa para E_3 é tal que $s_2=b_2$ estará provado que esta é também uma solução para E_1 e E_2.

Se $s_2 = \sum_{j=1}^{n} a_{2j}x_j^o \neq b_2$ então há um inteiro não nulo q tal que $s_2 = b_2+q$. Substituindo-se esta expressão em E_3, com $x_j=x_j^o$, temos:

$$s_1 + w(b_2+q) = b_1 + w\,b_2 \quad \rightarrow \quad s_1 = b_1 - wq$$

Se provarmos que $q=0$ concluiremos, então, que, uma solução inteira não negativa para $E_3 \rightarrow s_2 = b_2$, e consequentemente é solução para E_1 e E_2

Sabemos que se os coeficientes a_{1j} são positivos:

$$s_1 = b_1 - wq \geq 0$$

para qualquer solução não negativa (no caso inteira), pois

$$s_1 \equiv \sum_{j=1}^{n} a_{1j}x_j \; .$$

(1) Verificaremos agora se q é positivo

Para $q > 0$ a desigualdade $b_1-w_q \geq 0$ $\rightarrow$ $\rightarrow b_1 \geq wq$. Mas por (a) temos que $w > b_1$. Logo $b_1 \geq wq$

só pode ser verdadeiro se $0 < q < 1$. Como q é inteiro temos que $b_1 \geq wq$ é uma CONTRADIÇÃO. Assim sendo q não é positivo.

(2) Verificaremos agora se q é negativo

Quando $q < 0$, ele deve ser $-1,-2,-3$, etc, e então $s_1 = b_1 - wq \rightarrow s_1 \geq b_1 + w$, pois o menor valor da expressão $b_1 - wq$ é $b_1 + w$.

Mas, por definição: $w > b_2 \max\limits_j a_{1j}/a_{2j}$

Daí, temos: $w > b_2\, a_{1j}/a_{2j}$, $j=1,\ldots,n$ $\rightarrow$ $wa_{2j} > b_2\, a_{1j}$, $j=1,\ldots,n$. Podemos também escrever:

$w\, a_{2j}\, x_j > b_2\, a_{1j}\, x_j$, ou, também:

$$w \sum_{j=1}^{n} a_{2j}\, w_j > b_2 \sum_{j=1}^{n} a_{1j}\, x_j \rightarrow w\, s_2 > b_2\, s_1$$

Se $b_1 b_2 \geq 0$ então: $w\, s_2 > b_2\, s_1 - b_1 b_2$

Substituindo-se s_1 por $b_1 + w$ a desigualdade acima continua válida pois vimos que $s_1 \geq b_1 + w$. Assim:

$$w\, s_2 > b_2(b_1 + w) - b_1 b_2$$

$$w\, s_2 > b_2\, w \rightarrow s_2 > b_2$$

Mas $s_2 = b_2 + q$ e $q < 0$. Logo $s_2 > b_2$ é uma CONTRADIÇÃO.

De acordo com os resultados de (1) e (2) concluímos que $q=0$. Assim sendo provamos que $s_2 = b_2$ para uma solução inteira não negativa de E_3 . Logo esta é também uma solução inteira não negativa para E_1 e E_2.

Apliquemos o resultado desse teorema no exemplo:

$$2x_1 + 3x_2 = 5 \quad (I)$$
$$x_1 + 2x_2 = 3 \quad (II)$$

De acordo com o exemplo, temos:

$s_1 = 2x_1 + 3x_2$

$s_2 = x_1 + 2x_2$

$b_1 = 5$

$b_2 = 3$

Obtenção do limite inferior para o inteiro w:

$w > b_2$ máximo $\{a_{11}/a_{21}, a_{12}/a_{22}\}$

$w > 3$ máximo $\{2/1, 3/2\}$

$w > 6$

Seja,então: $w=7$. Assim:

$$s_1 + w\, s_2 = b_1 + w\, b_2 \rightarrow 2x_1+3x_2+7x_1+14x_2 = 5 + 7 \times 3$$

$$9x_1 + 17x_2 = 26 \quad (III)$$

A solução inteira de (III) é $x_1=x_2=1$, mesma solução de (I) e (II).

<u>OBSERVAÇÃO 1</u>:

O teorema estudado só pode ser aplicado, como vimos, as duas equações com coeficientes inteiros e estritamente positivos. Mathews observou, entretanto, que duas equações com coeficientes inteiros e não negativos (mas não necessariamente todos positivos) $s_1=b_1$ e $s_2=b_2$ podem, por intermédio da adaptação sugerida abaixo, tornarem-se equações com coeficientes estritamente positivos. A referida adaptação é a seguinte:

$$s_1 + s_2 = b_1 + b_2 \qquad ("E_1")$$

$$s_1 + 2s_2 = b_1 + 2b_2 \qquad ("E_2")$$

As equações "E_1" e "E_2" tem coeficientes estritamente positivos visto que elas são da forma $a_{1j} + a_{2j}$ (em "E_1") e $a_{1j} + 2a_{2j}$ (em "E_2"), e para cada $j(j=1,\ldots,n)$, $a_{ij} > 0$ para no mínimo $i(i=1,2)$. Então, estas equações podem ser chamadas de E_1 e E_2 quando da aplicação do teorema de Mathews. Também, note que uma solução para "E_1" e "E_2" é uma solução para E_1 e E_2 (e reciprocamente). Para ver isto basta subtrair "E_1" de "E_2" para obter $s_2=b_2$ e então, por "E_1" , $s_1=b_1$.

OBSERVAÇÃO 2:

Elmaghraby e Wig | 8 | notaram que o Teorema de Mathews pode ser usado para unir um sistema de equações com coeficientes não negativos usando repetidamente a construção "E_1" e "E_2" . Isto é, as duas primeiras equações do sistema são substituidas por "E_1" e "E_2" e então unidas de acordo com o teorema de Mathews. A equação agregada vem a ser a primeira equação e substitui as duas primeiras, tal que o sistema agora tem uma equação a menos. O processo é então repetido até que uma única equação é achada.

OBSERVAÇÃO 3: Nós também podemos usar o processo de união de forma par acima para equações contendo coeficientes negativos desde que limites superiores para as correspondentes variáveis possam ser achados. Em particular, se $a_{1j} < 0$ e é sabido que $x_j \leq \mu_j$ (μ_j é um limite superior positivo), então substituindo x_j por $\mu_j - \overline{x}_j$ na restrição 1, onde a variável complementar $\overline{x}_j = \mu_j - x_j$, muda-se o sinal de a_{1j}. Observe que este fáto e as exigências do teorema de Mathews significam que todo programa inteiro o qual tem uma limitada região viável da programação linear com no mínimo um ponto inteiro pode ser transformado em um equivalente problema mochila.

De acordo com a teoria acima, a sequência de passos abaixo pode ser usada para reduzir a uma as várias restrições de um programa linear inteiro.

<u>1º PASSO</u>: Introduzir variáveis complementares, tornando positivos os coeficientes negativos.

<u>C O M E N T Á R I O</u>: este passo, e evidentemente este algoritmo, só pode ser efetuado se as variáveis correspondentes aos coeficientes negativos tiverem um limite superior inteiro μ_j..

<u>2º PASSO</u>: Substituir as duas primeiras equações por "E_1" e "E_2"

$$s_1 + s_2 = b_1 + b_2 \qquad (\text{"}E_1\text{"})$$

$$s_1 + 2s_2 = b_1 + 2b_2 \qquad (\text{"}E_2\text{"})$$

3º PASSO: Selecionar $w > b_2 \max_j \{a_{ij}/a_{2j}\}$, inteiro, e agregar "$E_1$" e "$E_2$".

$$E^* = "E_1" + w."E_2"$$

C O M E N T Á R I O: as duas primeiras equações dão origem a uma única equação. A cada iteração haverá, portanto, uma equação a menos.

4º PASSO: Se só houver uma equação vá para o 5º PASSO.
Senão vá para o passo 2.

C O M E N T Á R I O: a primeira equação do sistema é agora E^* e a segunda equação é a primeira que ainda não foi solicitada.

5º PASSO: Substitua as variáveis complementares $\bar{x}_j$ por $1 - x_j$ e PARE.

APLICAÇÃO DESTE ALGORITMO

Seja o exemplo da secção 1 deste capítulo:

$$\begin{array}{ll}
\text{MINIMIZAR} & 5x_1+7x_2+10x_3+3x_4+x_5 \\
\text{s.a.:} & -x_1+3x_2-5x_3-x_4+4x_5+x_6 = -2 \\
& 2x_1-6x_2+3x_3+2x_4-2x_5+x_7 = 0 \\
& x_2-2x_3+x_4+x_5+x_8 = -1 \\
& x_j=0 \text{ ou } 1 \quad (j=1,2,3,4,5) \\
& x_j \geq 0 \text{ e inteiro } (j=6,7,8)
\end{array}$$

<u>1º PASSO</u>: Como as cinco primeiras variáveis são 0 ou 1, nós temos $\mu_j=1$ $(j=1,\dots,5)$. Para eliminar os coeficientes negativos nas três restrições, faremos $x_j = 1 - \bar{x}_j$, para j tal que a_j seja negativo, ou seja:

$$
\begin{aligned}
-(1-\bar{x}_1)+3x_2-5(1-\bar{x}_3)-(1-\bar{x}_4)+4x_5+x_6 &= -2\\
2x_1-6(1-\bar{x}_2)+3x_3+2x_4-2(1-\bar{x}_5)\quad +x_7 &= 0\\
x_2-2(1-\bar{x}_3)+x_4+x_5\quad +x_8 &= -1
\end{aligned}
$$

Sendo assim as restrições tomam o aspecto:

$$
\begin{aligned}
\bar{x}_1 + 3x_2 + 5\bar{x}_3 + \bar{x}_4 + 4x_5 + x_6 &= 5 \qquad E_1\\
2x_1 + 6\bar{x}_2 + 3x_3 + 2x_4 + 2\bar{x}_5 + x_7 &= 8 \qquad E_2\\
x_2 + 2\bar{x}_3 + x_4 - x_5 + x_8 &= 1 \qquad E_3
\end{aligned}
$$

<u>2º PASSO</u>: Obtenção de "E_1" e "E_2".

$$"E_1" = 2x_1+\bar{x}_1+3x_2+6\bar{x}_2+3x_3+5\bar{x}_3+2x_4+\bar{x}_4+4x_5+2\bar{x}_5+x_6+x_7 = 13$$

$$"E_x" = 4x_1+\bar{x}_1+3x_2+12\bar{x}_2+6x_3+5\bar{x}_3+4x_4+\bar{x}_4+4x_5+4\bar{x}_5+x_6+2x_7=21$$

<u>3º PASSO</u>: Obtenção de w e E*

$$w > 21 \quad \max_j \{a_{ij}/a_{2j}\} = 21 \quad (1/1)$$

Seja $w=22$. Então $E^* = "E_1" + 22 \cdot "E_2"$, fica:

$$E^* = 90x_1+23\bar{x}_1+69x_2+270\bar{x}_2+135x_3+115\bar{x}_3+90x_4+23\bar{x}_4+92x_5+90\bar{x}_5+ 23x_6+45x_7 = 475$$

4º PASSO: Como sobraram duas equações, E^* e E_3, voltaremos a efetuar o PASSO 2.

2º PASSO (2ª ITERAÇÃO)

A esta altura temos que:

$"E_1" = E^* + E_3$ e $"E_2" = E^* + 2E_3$. Obtemos, assim:

$$"E_1" = 90x_1+23\overline{x}_1+70x_2+270\overline{x}_2+135x_3+117\overline{x}_3+91x_4+23\overline{x}_4+93x_5+90\overline{x}_5+23x_6+45x_7+x_8 = 476$$

$$"E_2" = 90x_1+23\overline{x}_1+71x_2+270\overline{x}_2+135x_3+119\overline{x}_3+92x_4+23\overline{x}_4+94x_5+90\overline{x}_5+23x_6+45x_7+2x_8 = 477$$

3º PASSO (2ª ITERAÇÃO)

$$w > 477 \max_j\{a_{1j}/a_{2j}\} = 477.(1) = 477$$

Seja $w=478$. Então $E^{**} = "E_1" + 478. "E_2"$, fica:

$$E^{**} = 43110x_1+11017\overline{x}_1+34008x_2+129330\overline{x}_2+64665x_3+56999\overline{x}_3 + \\ +56999\overline{x}_3+44067x_4+11017\overline{x}_4+45025x_5+43110\overline{x}_5+11017x_6 + \\ +21555x_7+957x_8 = 228482$$

4º PASSO (2ª ITERAÇÃO)

Como resta uma única equação passamos ao 5º passo.

5º PASSO: Subsituindo $\overline{x}_j$ por $1-x_j$ para $j=1,2,3,4,5$ temos:

$$E^{**} = 32093x_1-95322x_2+7666x_3+33050x_4+1915x_5+11017x_6 + \\ + 21555x_7+957x_8 =-22991$$

Existe somente duas soluções (x_1,x_2,x_3,x_4,x_5) para a equação acima, as quais são as mesmas das três equações originais a cinco variáveis: $(0,1,1,0,0)$ e $(1,1,1,0,0)$

OBSERVAÇÃO 1ª:

Vimos, por intermédio desse exemplo, que a equação resultante do processo de agregação pode ter coeficientes negativos. Então, se o programa inteiro é para ser resolvido como um problema mochila com valores positivos para os coeficientes das restrições, limites superiores devem existir para as variáveis correspondentes a esses coeficientes negativos, tais que seus complementos possam ser introduzidos. Tendo o problema uma solução não negativa, o valor de b vem a ser positivo após a complementação.

Na equação resultante do exemplo desenvolvido tínhamos $-95322 < 0$ e $x_2 \leq 1$. Introduzindo $1-\bar{x}_2$ no lugar de x_2 , vem:

MINIMIZAR $5x_1+7\bar{x}_2+10x_3+3x_4+x_5+7$

s.a.: $32093x_1+95322\bar{x}_2+7666x_3+33050x_4+1915x_5 + 11017x_6+21555x_7+957x_8 = 72331$

$x_1 \leq 1 \ , \ \bar{x}_2 \leq 1 \ , \ x_3 \leq 1 \ , \ x_4 \leq 1 \ , \ x_5 \leq 1$

$x_1,\ldots,x_8 \geq 0$ e inteiros

Como MIN F(x) = -MAX(-F(x)) , temos:

$$\text{MIN } 5x_1+7\bar{x}_2+10x_3+3x_4+x_5+7 = -\text{MAX } -5x_1+7\bar{x}_2-10x_3-3x_4-x_5-7$$

Ou seja:

$$\text{MIN } 5x_1-7\bar{x}_2+10x_3+3x_4+x_5+7 = 7-\text{MAX}-5x_1+7\bar{x}_2-10x_3-3x_4-x_5$$

Nosso problema tornou-se um programa de maximização. Note que este problema tem valores negativos para os custos e limites superiores para certas variáveis. Assim, se ele deve ser resolvido como um problema mochila, a técnica de solução deve permitir pois valores negativos para os custos e limites superiores para as variáveis. Também deve permitir uma igualdade, preferivelmente do que uma inequação, como restrição.

OBSERVAÇÃO 2ª :

Os coeficientes de x_6, x_7 e x_8 são respectivamente os multiplicadores das restrições E_1,E_2 e E_3 do programa inteiro inicial tais que fornecem a equação resultante desejada. Senão, vejamos:

$$E^{**} = (E^* + E_3) + 478(E^* + 2\,E_3) = 479\,E^* + 957\,E_3 =$$

$$= 479[(E_1 + E_2) + 22(E_1 + 2E_2)] + 957\,E_3 =$$

$$= 11017\,E_1 + 21555\,E_2 + 957\,E_3$$

OBSERVAÇÃO 3ª:

Vimos que esse recurso de agregação acarreta um crescimento muito rápido dos coeficientes. De acordo com matéria vista em Glover e Woolsey |14|, o resultado de repetidas agregações de m equações fornecerá uma equação tal que b excede $2^u b^v$, onde $\mu = 3(2^{m-2})$ e $v = 2^{m-1}$. Este número é claramente astronômico para não mais que 7 equações. Estes números enormes causam erros de arredondamento e superabundância (overflow) no computador. Sendo assim, o processo estudado perde em eficiência.

3. Teorema de Glover; Processo de agregação II

Vários artigos têm aparecido discutindo técnicas de agregação as quais tendem a fornecer menores gastos computacionais. Apresentaremos agora um resultado devido a Glover |13|, o qual, oferece um favorável procedimento de redução. É parecido com o anterior no que se refere a união aos pares.

Teorema:

Considere um sistema de duas equações:

$$s_1 \equiv \sum_{j=1}^{n} a_{1j}x_j = b_1 \qquad (E_1)$$

$$s_2 \equiv \sum_{j=1}^{n} a_{2j}x_j = b_2 \qquad (E_2)$$

onde todos os coeficientes a_{ij} e todos os b_i são inteiros, e no mínimo um dos b_i é não nulo. Sejam w_1 e w_2 inteiros primos entre si. Se existe no mínimo uma solução não negativa para E_1 e E_2, então toda solução inteira não negativa para:

$$w_1s_1 + w_2s_2 = w_1b_1 + w_2b_2 \qquad (E_3)$$

é uma solução inteira não negativa para E_1 e E_2, e reciprocamente, desde que:

$$w_1 \, a_{1j} + w_2 \, a_{2j} \geq |b_2 \, a_{1j} - b_1 \, a_{2j}| \qquad \text{(i)}$$

para $j=1,\ldots,n$, e (i) mantenha-se "estritamente uma inequação" para $j \in J$, onde J é um subconjunto não vazio de $\{1,2,\ldots,n\}$ tal que toda solução não negativa para E_3 satisfaz $x_j > 0$ para no mínimo um $j \in J$.

COMENTÁRIO: Se existem cinco variáveis e (0,1,0,0,2) é uma solução, J é diferente de $\{1,3,4\}$,pois $x_1=x_3=x_4=0$.

Demonstração:

Está claro que E_1 e E_2 implicam em E_3 . Assim toda solução (inteira não negativa) de E_1 e E_2 satisfaz E_3 . Para mostrar a recíproca, escolha solução inteira não negativa $x_j (j=1,\ldots,n)$ para E_3 e multiplique ambos os lados de (i) por $x_j = |x_j|$ tal que:

$$(w_1 a_{1j} + w_2 a_{2j}) x_j \geq |b_2 a_{1j} - b_1 a_{2j}| \, |x_j| = |(b_2 a_{1j} - b_1 a_{2j}) x_j|$$

$$w_1 a_{1j} x_j + w_2 a_{2j} x_j \geq |b_2 a_{1j} x_j - b_1 a_{2j} x_j|$$

Passando ao somatório sobre j , temos:

$$w_1 s_1 + w_2 s_2 > |b_2 s_1 - b_1 s_2| \qquad \text{(ii)}$$

onde a estrita inequação segue da definição de J .

(Por (i) cada coeficiente da restrição agregada é não negativo. Então $x_j=0$ $(j=1,\ldots,n)$ é a única solução se e somente se $w_1 b_1 + w_2 b_2 = 0$. Nós estamos ignorando esta possibilidade, logo $x_j > 0$ para algum j em cada solução para E_3).

Agora suponhamos $s_1 \neq b_1$, o qual implica $s_2 \neq b_2$ (e vice-versa). Então sejam α e β todos os inteiros tais que:

$$s_1 = b_1 + \alpha \quad \text{e} \quad s_2 = b_2 + \beta \qquad (*)$$

Substituindo-se esses valores em E_3, temos:

$$\cancel{w_1 b_1} + w_1\alpha + \cancel{w_2 b_2} + w_2\beta = \cancel{w_1 b_1} + \cancel{w_2 b_2}$$
$$w_1\alpha = -w_2\beta \qquad (**)$$

Como, por hipótese, w_1 e w_2 são primos entre si, é necessário que façamos:

$\alpha = q\, w_2$ e $\beta = q\, w_1$, q inteiro, a fim de que $(**)$ seja verdadeira.

Substituindo-se α e β em (*), temos:

$$s_1 = b_1 + q\, w_2 \quad \text{e} \quad s_2 = b_2 - q\, w_1 \qquad (***)$$

Substituindo-se, agora, essas novas expressões de s_1 e s_2 em (ii), vem:

$$w_1 b_1 + \cancel{w_1 q w_2} + w_2 b_2 - \cancel{w_2 q w_1} > |\cancel{b_2 b_1} + b_2 q w_2 - \cancel{b_1 b_2} + b_1 q w_1|$$
$$w_1 b_1 + w_2 b_2 > |q|\ |w_1 b_1 + w_2 b_2|$$

Sendo assim, $|q| = 0$ ou $q = 0$. Logo, por (***) $s_1 = b_1$ e $s_2 = b_2$.

<u>CONCLUSÃO</u>:

Escolhemos uma solução x_j para E_3. Consideramos a expressão (i) obedecida e a partir dela obtivemos uma nova expressão que chamamos de (ii). A partir daí supomos $s_1 \neq b_1$ e $s_2 \neq b_2$. Utilizando a equação E_3 fomos forçados a definir novas expressões para s_1 e s_2, as quais quando substituidas em (ii) implicaram em $s_1 = b_1$ e $s_2 = b_2$. Ou seja x_j solução de E_3 implicam em $s_1 = b_1$ e $s_2 = b_2$. Logo x_j é solução de E_1 e E_2, como queríamos demonstrar.

<u>OBSERVAÇÃO</u>:

(1ª.) Através da expressão:

$$w_1 s_1 + w_2 s_2 = \sum_j w_1 a_{1j} w_j + \sum_j w_2 a_{2j} x_j = \sum_j (w_1 a_{1j} + w_2 a_{2j}) x_j$$

observamos ser o coeficiente de x_j em E_3 da forma $w_1 a_{1j} + w_2 a_2 j$.

(2ª.) O teorema implica por (ii) que w_1 e w_2 sejam escolhidos tais que $w_1 b_1 + w_2 b_2 > 0$. Também, por (i) os coeficientes em E_3, isto é, $w_1 a_{1j} + w_2 a_{2j}$ para $j=1,\ldots,n$, são não negativos. Entretanto, cada solução inteira não negativa para E_3 deve ter $x_j > 0$ para no mínimo um j onde seu coeficiente em E_3 é positivo. Consequentemente o conjunto J deve consistir daqueles j para os quais $w_1 a_{1j} + w_2 a_{2j} > 0$.

(3ª.) Por (i), uma condição suficiente mas não necessária para existência dos primos relativos w_1 e w_2 é que os coeficientes a_{ij} sejam não negativos. Para esclarecer, suponhamos que para um determinado $j \in J$ tenhamos a_{1j} e a_{2j} negativos. Logo $w_1 a_{1j} + w_2 a_{2j}$ é negativo e a equação (i) não é satisfeita para nenhum w_1, w_2 inteiro e positivos. Como isto pode não ocorrer, a condição não é necessária.

Este estudo nos permite formular o seguinte ALGORITMO:

1º PASSO: Introduzir variáveis complementares, tornando positivos os coeficientes negativos.

C O M E N T Á R I O: Este passo favorece a obtenção dos multiplicadores w_1 e w_2 .

2º PASSO: Obter os multiplicadores w_1 e w_2 a partir de:

$w_1 a_{1j} + w_2 a_{2j} > |b_2 a_{1j} - b_1 a_{2j}|$, para $j \in J$ e tais que $w_1 b_1 + w_2 b_2 > 0$ e w_1 e w_2 primos entre si.

3º PASSO: Obtenção da equação agregada:

$$E^* = w_1 E_1 + w_2 E_2$$

C O M E N T Á R I O: assim como no algoritmo anterior, E* passa a ser a primeira equação do sistema resultante após a substituição das duas primeiras equações.

4º PASSO: Se só houver uma equação vá para o 5º passo.
Senão volte ao 2º passo.

5º PASSO: Substitua as variáveis complementares $\bar{x}_j$ por $1 - x_j$ e PARE.

APLICAÇÃO DESTE ALGORITMO

Seja o mesmo exemplo da secção anterior cujas restrições são:

$$\begin{array}{l}
-x_1 + 3x_2 - 5x_3 - x_4 + 4x_3 + x_6 = -2 \\
2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 + x_7 = 0 \\
x_2 - 2x_3 + x_4 + x_5 + x_8 = -1 \\
x_j \in \{0,1\} \ , \ j=1,2,3,4,5 \\
x_j \geq 0 \text{ e inteiro } , \ j=6,7,8
\end{array}$$

1º PASSO: Sendo igual ao do algoritmo anterior podemos passar diretamente ao resultado já conseguido:

$$\begin{array}{ll}
\bar{x}_1 + 3x_2 + 5\bar{x}_3 + \bar{x}_4 + 4x_5 + x_6 = 5 & E_1 \\
2x_1 + 6\bar{x}_2 + 3x_3 + 2x_4 + 2\bar{x}_5 + x_7 = 8 & E_2 \\
x_2 + 2\bar{x}_3 + x_4 + x_5 + x_8 = 1 & E_3
\end{array}$$

2º PASSO: Obtenção dos multiplicadores w_1 e w_2

Seja $J = \{1,\ldots,7\}$

j	$\lvert b_2a_{1j}-b_1a_{2}j\rvert$	$w_1a_{1j}+w_2a_{2j} > \lvert b_2a_{1j}-b_1a_{2j}\rvert$
1	10	$0w_1 + 2w_2 > 10 \rightarrow w_2 > 5$
$\bar{1}$	8	$w_1 + 0w_2 > 8 \rightarrow w_1 > 8$
2	24	$3w_1 + 0w_2 > 24 \rightarrow w_1 > 8$
$\bar{2}$	30	$0w_1 + 6w_2 > 30 \rightarrow w_2 > 5$
3	15	$0w_1 + 3w_2 > 15 \rightarrow w_2 > 5$
$\bar{3}$	40	$5w_1 + 0w_2 > 40 \rightarrow w_1 > 8$
4	10	$0w_1 + 2w_2 > 10 \rightarrow w_2 > 5$
$\bar{4}$	8	$w_1 + 0w_2 > 8 \rightarrow w_1 > 8$
5	32	$4w_1 + 0w_2 > 32 \rightarrow w_1 > 8$
$\bar{5}$	10	$0w_1 + 2w_2 > 10 \rightarrow w_2 > 5$
6	8	$w_1 + 0w_2 > 8 \rightarrow w_1 > 8$
7	5	$0w_1 + w_2 > 5 \rightarrow w_2 > 5$

A partir daí verificamos que:

$w_1 = 8 + n_1$ e $w_2 = 5 + n_2$, $n_1, n_2 \in \{1,2,3,\ldots\}$

n_1	n_2	sinal de $w_1b_1+w_2b_2$	w_1 e w_2 são primos entre si?
1	1	> 0	NÃO
2	1	> 0	NÃO
1	2	> 0	SIM

Logo $w_1 = 9$ e $w_2 = 7$

3º PASSO: Obtenção da equação agregada E^*

$$E^* = w_1E_1 + w_2E_2 \rightarrow E^* = 9E_1 + 7E_2$$

$$E^* = 14x_1+9\overline{x}_1+27x_2+42\overline{x}_2+21x_3+45\overline{x}_3+14x_4+14x_4+9\overline{x}_4 + 36x_5+14\overline{x}_5+9x_6+7x_7 = 101$$

4º PASSO: Como há ainda duas equações, quais sejam E^* e E_3 voltaremos ao passo 2.

2º PASSO: (2ª ITERAÇÃO)

Obtenção dos multiplicadores w_1 e w_2 . Seja $J = \{1,\ldots,8\}$.

j	$\lvert b_2a_{1j}-b_1a_{2j}\rvert$	$w_1a_{2j}+w_2a_{2j} > \lvert b_2a_{1j}-b_1a_{2j}\rvert$
1	14	$14w_1 + 0w_2 > 14 \rightarrow w_1 > 1$
$\overline{1}$	9	$9w_1 + 0w_2 > 9 \rightarrow w_1 > 1$
2	74	$27w_1 + w_2 > 74$
$\overline{2}$	42	$42w_1 + 0w_2 > 42 \rightarrow w_1 > 1$
3	21	$21w_1 + 0w_2 > 21 \rightarrow w_1 > 1$
$\overline{3}$	157	$45w_1 + 2w_2 > 157$
4	87	$14w_1 + w_2 > 87$
$\overline{4}$	9	$9w_1 + 0w_2 > 9 \rightarrow w_1 > 1$
5	65	$36w_1 + w_2 > 65$
$\overline{5}$	14	$14w_1 + 0w_2 > 14 \rightarrow w_1 > 1$
6	9	$9w_1 + 0w_2 > 9 \rightarrow w_1 > 1$
7	7	$7w_1 + 0w_2 > 7 \rightarrow w_1 > 1$
8	101	$0w_1 + w_2 > 101 \rightarrow w_2 > 101$

Assim:

$$w_1 = 1+n_1 \quad \text{e} \quad w_2 = 101+n_2 \quad , \quad n_1,n_2 \in \{1,2,3,\ldots\}$$

n_1	n_2	sinal de $w_1b_1+w_2b_2$	w_1 e w_2 são primos entre si ?
1	1	> 0	NÃO
2	1	> 0	NÃO
1	2	> 0	SIM

Logo $w_1=2$ e $w_2=103$, os quais satisfazem as demais inequações da primeira tabela.

<u>3º PASSO</u>: (2ª ITERAÇÃO)

$$E^{**} = w_1\, E^* + w_2\, E_3 \;\rightarrow\; E^{**} = 2\, E^* + 103\, E_3$$

$$E^{**} = 28x_1+18\bar{x}_1+157x_2+84\bar{x}_2+42x_3+296\bar{x}_3+131x_4+18\bar{x}_4 + 175x_5+28\bar{x}_5+18x_6+14x_7+103x_8 = 305$$

<u>4º PASSO</u>: (2ª ITERAÇÃO)

Como resta uma única equação executaremos o último passo.

<u>5º PASSO</u>: Substituindo $\bar{x}_j$ por $1-x_j$, para $j=1,2,3,4,5$ vem a equação desejada:

$$10x_1 + 73x_2 - 254x_3 + 113x_4 + 147x_5 + 18x_6 + 14x_7 + 103x_8 = -139$$

<u>OBSERVAÇÃO 1ª</u> : Esta técnica nos oferece uma equação resultante com coeficientes bem menores que os da equação resultante quando da aplicação da técnica anterior. Mais computações,no entanto, são necessárias levando-se em conta os cálculos para obtenção dos multiplicadores.

<u>OBSERVAÇÃO 2ª</u> : Os coeficientes de x_6, x_7, x_8 são respectivamente os multiplicadores das restrições E_1, E_2 e E_3 do programa inteiro inicial tais que fornecem a equação resultante desejada. Senão vejamos:

$$E^{**} = 2\ E^* + 103\ E_3 = 2(9\ E_1 + 7\ E_2) + 103\ E_3 =$$
$$= 18\ E_1 + 14\ E_2 + 103\ E_3$$

OBSERVAÇÃO 3ª : A substituição $x_j = 1 - \bar{x}_j$, com o objetivo de tornar não negativos os coeficientes a_{ij} , pode contribuir para um acréscimo nos valores dos coeficientes da última equação agregada. Isto porque os valores dos b_i's podem aumentar e, consequentemente, a expressão $|b_2 a_{ij} - b_1 a_{2j}|$, que aparece na desigualdade (i), pode sofrer um acréscimo que provocaria maiores valores para os multiplicadores w_i's. Entretanto aquela substituição garante a existência desses multiplicadores. Mesmo assim, como vimos, essa técnica ainda fornece uma equação final bem mais favorável que a obtida pelo processo anterior.

OBSERVAÇÃO 4ª : Substituindo x_3 por $1 - \bar{x}_3$ na equação obtida, com o objetivo de tornar a_3 e b positivos, encontramos:

$$10x_1 + 73x_2 + 254\bar{x}_3 + 113x_4 + 147x_5 + 18x_6 + 14x_7 +$$
$$+ 103\ x_8 = 115$$

Caso não pudéssemos tornar a_3 e b positivos (se a_3 fosse menor que b) a inviabilidade do problema estaria comprovada.

CAPÍTULO IV

TEORIA DOS GRUPOS EM PROGRAMAÇÃO INTEIRA A UMA RESTRIÇÃO

1. Introdução

1.1. Congruências em Z (Z - conjunto dos inteiros)

Definição: Sejam a e b número inteiros e m inteiro positivo. Dizemos que a e b são congruentes segundo o módulo m se, e somente se, a diferença a-b é divisível por m .

Escreve-se:

$a \equiv b(\mathrm{mod}\ m)$

Le-se: a é congruente a b módulo m .

Portanto, temos:

$a \equiv b(\mathrm{mod}\ m) \leftrightarrow \forall k \in Z \ , \ a = b + km$

Exemplos:

$8 \equiv 2 \ (\mathrm{mod}\ 3)$

$3 \equiv 0 \ (\mathrm{mod}\ 3)$

Em particular, se a é mulitiplo de m , escreve-se:

$a \equiv 0 \ (\mathrm{mod}\ m)$, pois $a = km \ , \ k \in Z$

Teorema:

A relação em Z definida por $a \equiv b \ (\mathrm{mod}\ m)$ é uma relação de equivalência em Z, denominada CONGRUÊNCIA MÓDULO m .

Demonstração:

Uma relação chama-se de equivalência se, e somente se, é ao mesmo tempo reflexiva, simétrica e transitivo. Devemos então demonstrar que:

(i) $a \equiv a \pmod{m}$

(ii) $a \equiv b \pmod{m} \rightarrow b \equiv a \pmod{m}$

(iii) $a \equiv b \pmod{m}$ e $b \equiv c \pmod{m} \rightarrow a \equiv c \pmod{m}$

(i) $\forall a \in Z : a \equiv a \pmod{m}$, pois $a-a = 0$ e 0 é divisível por m.

(ii) $\forall a,b \in Z : a \equiv b \pmod{m} \rightarrow a-b$ divisível por $m \rightarrow$ $-(a-b)$ divisível por $m \rightarrow b-a$ divisível $m \rightarrow b \equiv a \pmod{m}$

(iii) $\forall a,b,c \in Z$

$a \equiv b \pmod{m} \rightarrow a - b$ divisível por m

$b \equiv c \pmod{m} \rightarrow b - c$ divisível por m

Assim: $(a-b)+(b-c)$ divide $m \rightarrow a-c$ divide $m \rightarrow$ $a \equiv c \pmod{m}$

<u>Propriedades das congruências módulo m</u>

(i) $a \equiv b \pmod{m} \rightarrow \begin{cases} a+c \equiv b+c \pmod{m} \\ c+a \equiv c+b \pmod{m} \end{cases}$

(ii) $a \equiv b \pmod{m} \rightarrow \begin{cases} a\,c \equiv bc \pmod{m} \\ c\,a \equiv cb \pmod{m} \end{cases}$

(iii) $a \equiv b \pmod{m}$ e $c \equiv d \pmod{m} \rightarrow ac \equiv bd \pmod{m}$

Demonstração:

$$\left.\begin{array}{l} a \equiv b \pmod{m} \rightarrow ac \equiv bc \pmod{m} \\ \\ c \equiv d \pmod{m} \rightarrow bc \equiv bd \pmod{m} \end{array}\right] \rightarrow ac \equiv bd \pmod{m}$$

Classes de equivalência em Z ; Conjunto Quociente Z_m

Definição:

Chama-se classe de equivalência segundo uma congruência módulo m de um elemento $a \in Z$, ao conjunto de todos os elementos $x \in Z$, tais que $a \equiv x \pmod{m}$.

Notação: $\bar{a} \rightarrow$ classe de equivalência de a

Assim:

$$\bar{a} = \{x \in Z \mid a \equiv x \pmod{m}\}$$

OBSERVAÇÃO:

Cada congruência módulo $\underline{m}$ define $\underline{m}$ classes de equivalência distintos. Vejamos:

$$a \equiv x \pmod{m} \rightarrow x \equiv a \pmod{m} \rightarrow x = a + km \text{ , } k \in Z$$

Assim $\underline{a}$ é o resto da divisão de $\underline{x}$ por $\underline{m}$ e tal $0 \leq a < m$.

De acordo com a observação acima, a congruência módulo 3 define 3 classes de equivalência: $\bar{0}$, $\bar{1}$, $\bar{2}$.

Exemplo:

Seja construir as classes de equivalência:

$$\bar{a} = \{x \in Z \mid a \equiv x \pmod{3}\}$$

Sabemos que são 3 classes assim definidas:

$$\bar{0} = \{x \in Z \mid x = 0 + 3k \text{ , } k \in Z\}$$

$$\bar{1} = \{x \in Z \mid x = 1 + 3k \text{ , } k \in Z\}$$

$$\bar{2} = \{x \in Z \mid x = 2 + 3k \text{ , } k \in Z\}$$

Logo:

$\overline{0} = \{\ldots,-9,-6,-3,0,3,6,9,\ldots\}$

$\overline{1} = \{\ldots,-8,-5,-2,1,4,7,10,\ldots\}$

$\overline{2} = \{\ldots,-7,-4,-1,2,5,8,11,\ldots\}$

OBSERVAÇÃO:

Verificamos que: $\overline{0} \cap 1 = \overline{0} \cap 2 = \overline{1} \cap \overline{2} = \emptyset$ e $\overline{0} \cup \overline{1} \cup \overline{2} = Z$. Este fato se apoia no TEOREMA FUNDAMENTAL SOBRE AS RELAÇÕES DE EQUIVALÊNCIA cujo enunciado segue:

"A toda relação de equivalência em um conjunto corresponde uma decomposição do conjunto em classes de equivalência disjuntas duas a duas e cuja união reproduz o conjunto".

Definição:

Chama-se conjunto-quociente Z_m ao conjunto das classes de equivalência de Z segundo uma congruência módulo m .

Exemplos:

$Z_3 = \{\overline{0},\overline{1},\overline{2}\}$

$Z_m = \{\overline{0},\overline{1},\overline{2},\ldots,\overline{m-1}\}$

OBSERVAÇÃO: Se chamássemos a congruência módulo m de relação R , o conjunto Z_m poderia ser representando por Z/R e chamado conjunto das classes de equivalência de Z segundo R . Esta nomenclatura está mais de acordo com a denominação de CONJUNTO-QUOCIENTE.

ADIÇÃO EM Z_m

Seja $m \geq 1$ um número inteiro e consideremos o conjunto quociente Z_m de Z pela congruência módulo m:

$$Z_m = \{\overline{0},\overline{1},\overline{2},\ldots,\overline{m-1}\}$$

onde cada elemento inteiro entre 0 e m-1 indica a classe de restos módulo m determinada por esse inteiro.

Adição módulo m:

Sejam $\overline{a},\overline{b} \in Z_m$. A operação adição em Z_m é assim definida:

$$\overline{a} + \overline{b} = \overline{a+b}$$

Redução módulo m: indicando-se por "r" o resto da divisão de "a" por "m", tem-se:

$$\overline{a} = \overline{r}$$

Como exemplo, seja $m=10$, $a=13$. Logo $r=3$. Em Z_{10} as classes de equivalência do elemento 13 e do elemento 3 são iguais, ou seja: $\overline{13} = \overline{3}$.

Utilizando a REDUÇÃO MÓDULO m, a soma $\overline{a} + \overline{b}$ pode ser feita desprezando-se os múltiplos de m, como veremos abaixo.

Sejam r e p, respectivamente, as restos da divisão euclidiana de a e b por m. Assim, temos:

$$\overline{a} = \overline{r} \quad \text{e} \quad \overline{b} = \overline{p}$$

Seja, agora, v o resto da divisão euclidiana de r+p por m. Temos:

$\overline{r+p} = \overline{v}$

Logo, podemos escrever:

$\overline{a}+\overline{b} = \overline{r} + \overline{p} = \overline{r + p} = \overline{v}$

Portanto, $\overline{a} + \overline{b}$ pode ser feita desprezando-se os múltiplos de m, razão pela qual é dita SOMA MÓDULO m e a operação $+$ em Z_m é dita ADIÇÃO MÓDULO m.

Exemplo:

Em Z_{10}, temos: $\overline{14} + \overline{29} = \overline{4} + \overline{9} = \overline{13} = \overline{3}$

Em outra notação, podemos escrever:

$14(\text{mod } 10)+29(\text{mod } 10)=4(\text{mod } 10)+9(\text{mod } 10)=13(\text{mod } 10)=3(\text{mod } 10)$

1.2. Operação Binária; Grupo Abeliano

Definição: Uma OPERAÇÃO BINÁRIA em um conjunto não vazio V é uma aplicação $f: V \times V \rightarrow V$.

Portanto a cada par ordenado $(v_i, v_j) \in V \times V$ corresponde um, e somente um $v_k \in V$.

Sendo $*$ uma operação binária, $\forall v_i, v_j \in V$ então $v_i * v_j \in V$.

Definição: Seja $V = \{v_0, v_1, \ldots, v_{|V|-1}\}$ um conjunto finito e seja $*$ uma operação binária. Dizemos que o conjunto V munido da operação $*$ é um GRUPO ABELIANO se a operação binária $*$ satisfaz as seguintes propriedades:

(i) $(v_i * v_j) * v_k = v_i * (v_j * v_k) = v_i * v_j * v_k \ , \ \forall v_i, v_j, v_k \ \varepsilon \ V$

(ii) $\exists v_o \ \varepsilon \ V$ tal que $v_o * v_i = v_i \ , \ \forall v_i \ \varepsilon \ V$

(iii) $\exists v_k \ \varepsilon \ V$ tal que $v_k * v_i = v_o \ , \ \forall v_i \ \varepsilon \ V$

(iv) $v_i * v_j = v_j * v_i \ , \ \forall v_i, v_j \ \varepsilon \ V$

OBSERVAÇÃO: A quarta propriedade caracteriza o GRUPO ABELIANO.

O grupo acima é denotado por $G = (V, *)$ e sua ordem (número de elementos de V) por $|G| = |V|$.

EXEMPLO: $(Z_m, +)$ é um GRUPO ABELIANO, denominado GRUPO ADITIVO DOS INTEIROS MÓDULO m . Senão, vejamos.

A operação + em Z_m é uma operação binária, pois: $\overline{0} \leq (i+j)(\mathrm{mod}\ m) < \overline{m}$. As quatro propriedades se verificam:

(i) $((i+j)(\mathrm{mod}\ m)+k)(\mathrm{mod}\ m) = (i+(j+k)(\mathrm{mod}\ m))(\mathrm{mod}\ m) =$

$= (i+j+k)(\mathrm{mod}\ m)$

(ii) $\overline{0}$ é tal que $\overline{0}+i(\mathrm{mod}\ m) = i(\mathrm{mod}\ m)$

(iii) $((m-i)+i)(\mathrm{mod}\ m) = \overline{0}$

(iv) $(i+j)(\mathrm{mod}\ m) = (j+i)(\mathrm{mod}\ m)$

OBSERVAÇÃO: $\overline{i} = i \ (\mathrm{mod}\ m)$

1.3. Grupo particular G(m)

Aplicando Teoria dos Grupos em Programação Inteira, somente o grupo G(m), que iremos definir, e certos grupos derivados dele precisam ser considerados.

Seja $V = \{v_o, v_1, \ldots, v_{m-1}\}$ e $v_i * v_j = v_{(i+j)(\text{mod } m)}$.

Mostraremos que $G = (V,*)$ é um grupo abeliano, o qual será denotado particularmente por $G(m)$.

A operação $*$ é binária, pois $0 \leq (i+j)(\text{mod } m) < m$

Vejamos as propriedades:

(i) $(v_i * v_j) * v_k = v_{(i+j)(\text{mod } m)} * v_k = v_{((i+j)(\text{mod } m)+k)(\text{mod } m)}$

$= v_i * (v_j * v_k) = v_i * v_j * v_k$ (a partir da demonstração da propriedade (i) para o grupo $(Z_m, +)$)

(ii) $v_o * v_i = v_{(0+i)(\text{mod } m)} = v_i$

(iii) $v_{(m-i)} * v_i = v_{m(\text{mod } m)} = v_o$

(iv) $v_i * v_j = v_{(i+j)(\text{mod } m)} = v_{(j+i)(\text{mod } m)} = v_j * v_i$.

Portanto $G(m) = (V,*)$, onde $V = \{v_o, v_1, \ldots, v_{m-1}\}$ e $v_i * v_j = v_{(i+j)(\text{mod } m)}$, é um GRUPO ABELIANO.

1.4. Subgrupos; Subgrupos de G(m)

Definição: Seja o grupo $G = (V,*)$ e $M \subseteq V$. Se M munido da operação binária de G é um grupo então $H = (M,*)$ é dito subgrupo de $G = (V,*)$.

<u>Teorema</u>: Seja $G = (V,*)$ e $M \subseteq V$. Dizemos que $H = (M,*)$ é um subgrupo de G se, e somente se:

(i) $\forall v_i, v_j \in M \rightarrow v_i * v_j \in M$

(ii) $v_o \in M$, onde v_o é tal que $v_o * v_i = v_i$, $\forall v_i \in M$

(iii) $\forall v_i \in M \rightarrow v_k \in M$, onde v_k é tal que $v_k * v_i = v_o$

<u>OBSERVAÇÃO</u>: A demonstração do teorema é imediata a partir da definição de subgrupo.

<u>Exemplo</u>: Seja o grupo $G(8)$ e $M = \{v_o, v_2, v_4, v_6\}$. Então $H = (M,*)$, onde $*$ é a operação binária de G, é um subgrupo de $G(8)$, a partir do teorema anterior.

<u>Subgrupo gerado por um elemento v_i de V</u>

Defina:

(1) $n\ v_i = v_i * v_i * \dots * v_i$ (n termos)

(2) $0\ v_i = v_o$

Em $G(m)$, a partir da definição acima, temos:

$$n\ v_i = v_{(ni)(\text{mod } m)}$$

Seja agora:

$$M(v_i) = \{v_j \mid v_j = n\ v_i \text{ para algum } n\}$$

<u>Afirmação</u>: $H(v_i) = (M(v_i),*)$ é um subgrupo de $G(m)$.

Prova: Basta que sejam verificados os 3 itens do teorema anterior.

(i) Sejam $v_j = n_1 v_i$ e $v_k = n_2 v_i$. De acordo com a definição (1) podemos escrever:

$$v_j * v_k = n_1 v_i * n_2 v_i = \underbrace{v_i * \ldots * v_i}_{n_1 \text{ termos}} * \underbrace{v_i * \ldots * v_i}_{n_2 \text{ termos}} =$$

$$= (n_1 + n_2) v_i$$, o que nos permite concluir que:

$$\forall v_j,\ v_k \in M(v_i) \rightarrow v_i * v_j \in M(v_i)$$

(ii) Para $n=0$, temos: $n\, v_i = v_o$, a partir da definição (2), o que nos permite afirmar que $v_o \in M(v_i)$.

(iii) Seja $v_j = \frac{n}{i} v_i$. Fazendo $v_k = \frac{m-n}{i} v_i$, temos:

$$v_j * v_k = \frac{n}{i} v_i * \frac{m-n}{i} v_i = \frac{m}{i} v_i = v_{(\frac{m}{i} i) (\text{mod } m)} =$$

$$= v_{m (\text{mod } m)} = v_o$$

OBSERVAÇÃO: $H(v_i)$ é chamado SUBGRUPO GERADO por v_i. A ordem de $H(v_i)$ é chamada de ordem de v_i e é denotada por $|v_i|$

EXEMPLO:

Consideremos o grupo $G(m)$ para $m=8$, ou seja:

$$G(8) = (V,*), \text{ onde: } \left[\begin{array}{l} V = \{v_o,v_1,v_2,v_3,v_4,v_5,v_6,v_7\} \\ \qquad\qquad e \\ v_i * v_j = v_{(i+j)(\text{mod } 8)} \end{array}\right.$$

Vimos que $(M,*)$, $M = \{v_o,v_2,v_4,v_6\}$ é subgrupo de $G(8)$. Além disso temos que:

$$\left.\begin{array}{l} v_o = 4v_2 = v_{(4x2)(\text{mod } 8)} \\ v_4 = 2v_2 = v_{(2x2)(\text{mod } 8)} \\ v_6 = 3v_2 = v_{(3x2)(\text{mod } 8)} \end{array}\right[\begin{array}{l} \rightarrow \forall v_j \in M \text{ existe } n \text{ tal} \\ \text{que } v_j = nv_2 \end{array}$$

O subgrupo $(M,*)$ pode ser, então, denotado por $H(v_2) = (M(v_2),*)$, ou seja subgrupo gerado por v_2 . Como podemos observar, a ordem de $H(v_2)$ é $|v_2| = 4$.

<u>OBSERVAÇÃO</u>: A ordem de v_i , gerador de $(M(v_i),*)$, é um divisor da ordem de $G(m)$. Isto implica em:

$$|G(m)|.v_i = v_o$$

<u>Definição</u>: Um grupo $G = (V,*)$ é dito CÍCLICO se existe $v_i \in V$, tal que v_i gera G .

<u>Afirmação</u>: $G(m)$ é CÍCLICO.

<u>prova</u>: Sendo $G(m) = (V,*)$, temos a partir da definição da operação binária $*$, que qualquer $v_j \in V$ é tal que $v_j = n\, v_1$. Assim sendo v_1 gera o grupo $G(m)$. Logo $G(m)$ é CÍCLICO para todo m .

1.5. Grupo particular $G(m_1, m_2, \ldots, m_n)$

Sejam:

$$V = \{(a_i, b_k) \mid i=0,\ldots,m_1-1 \quad e \quad k=0,\ldots,m_2-1\}$$

$$(a_i, b_k)*(a_j, b_\ell) = (a_{(i+j)(\mathrm{mod}\ m_1)}, b_{(k+\ell)(\mathrm{mod}\ m_2)})$$

Afirmação: $G(m_1, m_2) = (V, *)$ define um grupo de ordem $m_1 m_2$ dito SOMA DIRETA de $G(m_1)$ e $G(m_2)$

prova: A operação * definida para os elementos de V é uma operação binária pois quaisquer que sejam os elementos (a_i, b_k) e (a_j, b_ℓ) de V então $(a_i, b_k)*(a_j, b_\ell)$ também é um elemento de V . As demais propriedades de grupo se verificam, como podemos ver a seguir.

(i) $[(a_i, b_k)*(a_j, b_\ell)]*(a_m, b_n) = (a_i, b_k)*[(a_j, b_\ell)*(a_m, b_n)] =$

$= (a_i, b_k)*(a_j, b_\ell)*(a_m, b_n)$

(ii) $\exists\, (a_o, b_o) \in V$ tal que $(a_o, b_o)*(a_i, b_k) = (a_i, b_n)$ para todo (a_i, b_k) de V

(iii) $\exists (a_{m1}-i, b_{m2}-k) \in V$ tal que:

$(a_{m1}-i, b_{m2}-k)*(a_i, b_k) = (a_o, b_o)$

Logo $G(m_1, m_2) = (V, *)$ tem estrutura de grupo.

Sendo m_1 o número de elementos a_i e m_2 o número de elementos b_k temos um total de $m_1 m_2$ elementos

(a_i, b_k).

OBSERVAÇÃO: $G(m_1, m_2)$ é um grupo abeliano.

Seguindo o mesmo raciocínio podemos contruir o grupo soma direta $G(m_1, m_2, \ldots, m_n)$ cuja ordem é:

$$|G(m_1, m_2, \ldots, m_n)| = \prod_{i=1}^{n} m_i$$

NOTAÇÃO: Os elementos (a_i, b_k) serão denotados, daqui por diante, por $v_{i,k}$. Sendo assim:

$$V = \{v_{i,k} \mid i=0, \ldots, m_1-1 \text{ e } k=0, \ldots, m_2-1\}$$

EXEMPLO: Seja o grupo $G(2,3)$. Neste caso temos:

$$V = \{v_{0,0}, v_{1,0}, v_{0,1}, v_{1,1}, v_{0,2}, v_{1,2}\}$$

Para construir a tabela de $G(2,3)$ lembremos que:

$$v_{i,k} * v_{j,\ell} = v_{(i+j)(\text{mod } 2),(k+\ell)(\text{mod } 3)}$$

*	$v_{0,0}$	$v_{1,0}$	$v_{0,1}$	$v_{1,1}$	$v_{0,2}$	$v_{1,2}$
$v_{0,0}$	$v_{0,0}$	$v_{1,0}$	$v_{0,1}$	$v_{1,1}$	$v_{0,2}$	$v_{1,2}$
$v_{1,0}$	$v_{1,0}$	$v_{0,0}$	$v_{1,1}$	$v_{0,1}$	$v_{1,2}$	$v_{0,2}$
$v_{0,1}$	$v_{0,1}$	$v_{1,1}$	$v_{0,2}$	$v_{1,2}$	$v_{0,0}$	$v_{1,0}$
$v_{1,1}$	$v_{1,1}$	$v_{0,1}$	$v_{1,2}$	$v_{0,2}$	$v_{1,0}$	$v_{0,0}$
$v_{0,2}$	$v_{0,2}$	$v_{1,2}$	$v_{0,0}$	$v_{1,0}$	$v_{0,1}$	$v_{1,1}$
$v_{1,2}$	$v_{1,2}$	$v_{0,2}$	$v_{1,0}$	$v_{0,0}$	$v_{1,1}$	$v_{0,1}$

TABELA DE G(2,3)

A partir da construção da tabela para $n\, v_{ik}$ com $n=0,1,\ldots,|G|-1$ verificamos que $G(2,3)$ é cíclico, visto que é gerado por $v_{1,1}$ e $v_{1,2}$.

n	0	1	2	3	4	5
$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$
$v_{1,0}$	$v_{0,0}$	$v_{1,0}$	$v_{0,0}$	$v_{1,0}$	$v_{0,0}$	$v_{1,0}$
$v_{0,1}$	$v_{0,0}$	$v_{0,1}$	$v_{0,2}$	$v_{0,0}$	$v_{0,1}$	$v_{0,2}$
$v_{1,1}$	$v_{0,0}$	$v_{1,1}$	$v_{0,2}$	$v_{1,0}$	$v_{0,0}$	$v_{1,2}$
$v_{0,2}$	$v_{0,0}$	$v_{0,2}$	$v_{0,1}$	$v_{0,0}$	$v_{0,2}$	$v_{0,1}$
$v_{1,2}$	$v_{0,0}$	$v_{1,2}$	$v_{0,1}$	$v_{1,0}$	$v_{0,2}$	$v_{1,1}$
	Tabela de $n\, v_{i,k}$ para $G(2,3)$					

OBSERVAÇÃO: Provamos anteriormente que $G(m)$ é CÍCLICO. Entretanto o mesmo não pode ser feito para o grupo $G(m_1,m_2,\ldots,m_n)$, salvo exce ções.

Generalizando podemos denotar os elementos do grupo $G(m_1,m_2,\ldots,m_n) = (V,*)$ da seguinte forma:

$$v_{i_1,i_2,\ldots,i_n} \in V \ , \text{ onde } \ 0 \leq i_k < m_k \quad , \quad k=1,2,\ldots,n$$

EXEMPLO: Considerando o grupo $G(2,3,4)$ seus elementos são, então, denotados como:

$$v_{i_1,i_2,i_3} \ , \text{ onde: } \left[\begin{array}{l} 0 \leq i_1 < 2 \\ 0 \leq i_2 < 3 \\ 0 \leq i_3 < 4 \end{array}\right.$$

Ao operarmos dois elementos de $G = (m_1, m_2, \ldots, m_n)$ efetuaremos "n" adições módulo m_i, $i=1,2,\ldots,n$.

1.6. Isomorfismo

Definição: Sejam os grupos $G = (A,*)$ e $G' = (B,\circ)$. Um ISOMORFISMO de G em G' é toda função bijetora $f:G \to G'$ tal que se tem:

$$f(a_1 * a_2) = f(a_1) \circ f(a_2) \quad , \quad \forall a_1 \, , \, a_2 \in A$$

A existência de ao menos um isomorfismo entre dois grupos significa que, do ponto de vista algébrico, eles têm o mesmo comportamento. Neste caso, os grupos são ditos ISOMORFOS. Costuma-se dizer também que eles são iguais "a menos de um isomorfismo".

EXEMPLO: $G(6)$ é isomorfo a $G(2,3)$. Senão vejamos:

$$v_0 \leftrightarrow f(v_0) = v_{0,0}$$

$$v_1 \leftrightarrow f(v_1) = v_{1,2}$$

$$v_2 \leftrightarrow f(v_2) = v_{0,1}$$

$$v_3 \leftrightarrow f(v_3) = v_{1,0}$$

$$v_4 \leftrightarrow f(v_4) = v_{0,2}$$

$$v_5 \leftrightarrow f(v_5) = v_{1,1}$$

$$f(v_i * v_j) = f(v_i) * f(v_j) \quad , \quad \forall v_i \, , \, v_j \in G(6)$$

Sendo $G(6)$ isomorfo a $G(2,3)$ então $G(6)$ sendo CÍCLICO implica em G(2,3) CÍCLICO.

CONTRA-EXEMPLO: Seja a tabela de $n\ v_{i,k}$ para o grupo G(2,2), abaixo:

n	0	1	2	3
$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$	$v_{0,0}$
$v_{1,0}$	$v_{0,0}$	$v_{1,0}$	$v_{0,0}$	$v_{1,0}$
$v_{0,1}$	$v_{0,0}$	$v_{0,1}$	$v_{0,0}$	$v_{0,1}$
$v_{1,1}$	$v_{0,0}$	$v_{1,1}$	$v_{0,0}$	$v_{1,1}$

Verificamos que nenhum elemento gera o grupo. Logo G(2,2) não é CÍCLICO. Como G(4) é cíclico então G(2,2) não é isomorfo a G(4).

OBSERVAÇÃO: Nota-se, sem embaraços, que o grupo $G(1,\dots,1, m_1,\dots,m_n)$ é ISOMORFO a $G(m_1,\dots,m_n)$

Um importante resultado teórico de grupos para PROGRAMAÇÃO INTEIRA é que cada grupo soma direta $G(m_i,\dots,m_n)$ de ordem maior que 1 é isomorfo a um único grupo soma direta $G(s_1,\dots,s_k)$ tal que:

(i) $\prod_{i=1}^{n} m_i = \prod_{i=1}^{k} s_i$

(ii) $1 < s_1 \leq \dots \leq s_k$

(iii) s_i é um divisor de s_{i+1} , $i=1,\dots,k-1$

(iv) $k \leq n$

O grupo $G(s_1,\ldots,s_k)$ é chamado representação CANÔNICA de todos os grupos isomorfos a ele. Mais a frente será apresentado um método para obter a forma canônica. Note que, a partir da propriedade (iv), a representação canônica envolve um número mínimo de adições módulo sobre todas as representações isomorfas.

1.7. Classes Laterais

Definição: Seja $H = (M,*)$ um subgrupo de $G = (V,*)$. Uma classe lateral à esquerda de H em G é um conjunto cujos elementos são da forma $v*v_j$, $v \varepsilon G$ e $v_j \varepsilon H$, $0 \leq j \leq |H|-1$. Denotaremos uma classe lateral à esquerda por $v*H$.

EXEMPLO: Seja o subgrupo de $G(m)$ gerado por v_i, $H(v_i) = (M(v_i),*)$. A classe lateral à esquerda representada pelo elemento $v \varepsilon G$ é o conjunto:

$$M_v(v_i) = \{r \text{ tal que } r = v*n\,v_i \;,\; 0 \leq n \leq |v_i|-1\}$$

Teorema: Seja $H = (M,*)$ subgrupo de $G = (V,*)$ e consideremos dois elementos $v_1,v_2 \varepsilon V$. Sendo $v_1 \neq v_2$ então ou $v_1*H = v_2*H$ ou $v_1*H \cap v_2*H = \emptyset$. (Da mesma forma: ou $H*v_1 = H*v_2$ ou $H*v_1 \cap H*v_2 = \emptyset$).

Demonstração:

Suponhamos $v_1*H \cap v_2*H \neq \emptyset$. Seja $x \varepsilon v_1*H \cap v_2*H$. Então $x = v_i*v_j = v_2*v_k$, v_j e $v_k \varepsilon H$, pois $x \varepsilon v_1*H$ e

$x \varepsilon v_2*H$.

$$v_2^{-1}*(v_1*v_j)*v_j^{-1} = v_2^{-1}*(v_2*v_k)*v_j^{-1}$$

$$v_2^{-1}*v_1 = v_k*v_j^{-1}$$

Como $v_k*v_j^{-1} \varepsilon H$ então $v_2^{-1}*v_1 \varepsilon H$

$$v_2^{-1}*v_1 \varepsilon H \rightarrow (v_2^{-1}*v_1)*H = H \rightarrow v_2^*(v_2^{-1}*v_1)*H = v_2*H \rightarrow$$

$$\rightarrow v_1*H = v_2*H$$

Aplicação: A união de todas as classes laterais à esquerda (ou à direita) de $H = (M,*)$ em $G(V,*)$ reproduz o conjunto V .

$$\bigcup_{v \varepsilon V} M_v(v_i) = V$$

Outros resultados importantes: 1º) Seja H um subconjunto de G . O número de classes laterais à esquerda de H em G coincide com o número de classes laterais à direita de H em G.

2º) Sendo H um subgrupo de G , o número de classes laterais à esquerda distintas de H em G é: $|G|/|H|$.

De acordo com os resultados apresentados as diferentes classes laterais à esquerda (ou à direita) de $M(v_i)$ em $G(m)$ fornecem uma partição de V em $|G(m)|/|v_i|$ subconjuntos. Cada um desses subconjuntos tem cardinalidade $|v_i|$. Por exemplo, em $G(2,2)$, temos:

$$G(2,2) = (\{v_{0,0}, v_{0,1}, v_{1,0}, v_{1,1}\}, *)$$

$$H(v_{0,1}) = (\{v_{0,0}, v_{0,1}\}, *) \quad \text{subgrupo de} \quad G(2,2)$$

As classes laterais à esquerda de $H(v_{0,1})$ em $G(2,2)$ são:

$$H_{v_{0,0}}(v_{0,1}) = \{v_{0,0}*v_{0,0}, v_{0,0}*v_{0,1}\} = \{v_{0,0}, v_{0,1}\}$$

$$H_{v_{0,1}}(v_{0,1}) = \{v_{0,1}*v_{0,0}, v_{0,1}*v_{0,1}\} = \{v_{0,1}, v_{0,0}\}$$

$$H_{v_{1,0}}(v_{0,1}) = \{v_{1,0}*v_{0,0}, v_{1,0}*v_{0,1}\} = \{v_{1,0}, v_{1,1}\}$$

$$H_{v_{1,1}}(v_{0,1}) = \{v_{1,1}*v_{0,0}, v_{1,1}*v_{0,1}\} = \{v_{1,1}, v_{1,0}\}$$

Observamos que a união dessas classes laterais reproduz o conjunto V de $G(2,2)$.

Verificamos também que o número de classes laterais distintas de $H(v_{0,1})$ em $G(2,2)$ é dado por:

$$\frac{|G(2,2)|}{|H(v_{0,1})|} = \frac{4}{2} = 2$$

2. Formulação e Estudo do Problema Grupo Mochila

2.1. Formulação do Problema Grupo Mochila

Sejam: (i) $G = (V,*)$ um grupo abeliano finito não necessariamente cíclico;

(ii) $(V',*)$, onde $V' = \{v_{i1}, v_{i2}, \ldots, v_{is}\} \subseteq V$, não necessariamente um subgrupo de G.

Dado $v^o \varepsilon V$ e $I = \{i_1, i_2, \ldots, i_s\}$ nosso objetivo é achar inteiros não-negativos r_{ip} , $p=1,\ldots,s$, tais que:

$$\overset{*}{_{j\varepsilon I}} r_j v_j = r_{i_1} v_{i_1} * r_{i_2} v_{i_2} * \ldots * r_{i_s} v_{i_s} = v^0 \qquad (I)$$

Temos que, para inteiros a e n tais que $a + n|v_k| \geq 0$:

$$(a + n|v_k|)v_k = av_k * n|v_k|v_k = av_k * nv_o = av_k * v_o = av_k$$

Sendo assim, se (I) tem uma solução com $r_k = a$, existem soluções com $r_k = a + n|v_k|$ para todo n tal que $a + n|v_k| \geq 0$.

Podemos, assim, formular o seguinte problema de programação linear:

$$\begin{aligned} f(s,v^o) &= \min \sum_{j\varepsilon I} b_j r_j \\ &\overset{*}{_{j\varepsilon I}} r_j v_j = v^o \\ &r_j \geq 0 \text{ e inteiro, } j \varepsilon I \end{aligned} \qquad (II)$$

Os valores de $b_j \geq 0$ são dados para todo $j \varepsilon I$.

<u>IMPORTANTE</u>: (1) Sendo os $b_j \geq 0$, então, se (II) tem uma solução, ele tem uma solução ótima com $r_j < |v_j|$ para todo $j \varepsilon I$.

EXEMPLO: Consideremos $G(2,4)$ e $V' \subseteq V = \{v_{0,2}, v_{0,3}, v_{1,1}\}$. Seja o problema:

$$f(3,v_{1,3}) = \text{MIN} \quad 2r_{0,2} + 3r_{0,3} + 2r_{1,1}$$

$$\text{s.a.: } r_{0,2}v_{0,2} * r_{0,3}v_{0,3} * r_{1,1}v_{1,1} = v_{1,3}$$

$$r_j \geq 0 \text{ e inteiro, } j \in \{(0,2),(0,3),(1,1)\}$$

$$|v_{0,2}| = 2 \quad ; \quad |v_{0,3}| = 4 \quad ; \quad |v_{1,1}| = 4$$

Temos que $r_{0,2} = r_{0,3} = 0$ e $r_{1,1} = 7$ é tal que:

$$r_{0,2}v_{0,2} * r_{0,3}v_{0,3} * r_{1,1}v_{1,1} = v_{1,3}$$

Entretanto: $r_{0,2} = r_{1,1} = 1$ e $r_{0,3} = 0$ é uma solução ótima para o problema considerado e tal que $r_{0,2} < |v_{0,2}|$ e $r_{1,1} < |v_{1,1}|$.

(2) Se algum b_j é negativo e (II) tem uma solução, ele é ILIMITADO.

Suponhamos que no problema sugerido anteriormente tenhamos $b_{0,3} < 0$. Consideremos o solução $r_{0,2} = r_{0,3} = 0$ e $r_{1,1} = 7$. Vimos que sendo $r_{0,3} = 0$ uma solução, então existem soluções com $r_{0,3} = 0 + n|v_{0,3}|$ para todo n tal que $0 + n|v_{0,3}| \geq 0$. Logo $(0,4n,7)$ é a solução que torna mínima a função $2r_{0,2} + b_{0,3}r_{0,3} + 2r_{1,1}$ com $b_{0,3}<0$. Sendo assim o problema é ILIMITADO, pois $4n$ é sempre positivo para $n > 0$.

2.2. Técnicas de Solução

Mostraremos que o recurso de PROGRAMAÇÃO DINÂMICA pode ser usado para o caso do PROBLEMA GRUPO MOCHILA.

2.2.1. Técnica I

Seja $I_k = \{i_1, \ldots, i_k\}$, $k=1,\ldots,s$. Para qualquer que seja v , temos:

$$f(k,v) = \text{MIN} \sum_{j \varepsilon I_k} b_j r_j$$

$$\mathop{*}_{j \varepsilon I_k} r_j v_j = v \qquad \text{(III)}$$

$$r_j \geq 0 \text{ e inteiro, } j \ \varepsilon \ I_k$$

Observamos que (II) tem uma solução com $r_{ik} = a$ se, e somente se:

$$\mathop{*}_{j \varepsilon I_{k-1}} r_j v_j = v *(-a\ v_{i_k}),\ r_j \geq 0 \quad \text{e inteiro, } j \ \varepsilon \ I_{k-1}$$

tem uma solução (IV)

Combinando (III) e (IV) obtemos para todo v e $k=2,\ldots,n$:

$$f(k,v) = \mathop{\text{MÍNIMO}}_{r_{i_k}=0,\ldots,|v_{ik}|-1} (b_{i_k} r_{i_k} + f(k-1, v^* - (r_{i_k} \ v_{i_k}))) \qquad \text{(V)}$$

Se v pertence ao subgrupo gerado por v_{i_1} , então:

$f(1,v) = b_{i_1} r^o_{i_1}$, onde $r^o_{i_1}$ é o menor inteiro não negativo tal que $r_{i_1} v_{i_1} = v$. Caso contrário: $f(1,v) = M$, M arbitrariamente grande.

Para extender (V) no caso de $k=1$, inicialize:

$$f(0,v_{0,0}) = 0$$

$$f(0,v) = M$$

Como um problema simples de mochila, (III) pode ser resolvido para todo v achando $f(s,v)$ para todo v .

A técnica decorrente de (III) é análoga a desenvolvida para:

$$f(k,g) = \underset{x_k=0,1,\ldots,[g/a_k]}{\text{máximo}} (c_k x_k + f(k-1, g-a_k x_k))$$ no caso do problema mochila simples.

Para um problema mochila grupo, a eficiência de um algoritmo depende da estrutura do grupo. A técnica decorrente de (V) é vantajosa se $|v_j|$, $j \in I$, é pequena, como em $G(2,2,\ldots,2)$.

<u>EXEMPLO</u>: Seja definir a fórmula (V) considerando o grupo $G(2,2)$..

$|v_{i_k}| = 2$, pois: $v_{1,0} + v_{1,0} = v_{0,0} = v_0$

$$v_{0,1} + v_{0,1} = v_{0,0} = v_0$$

$$v_{1,1} + v_{1,1} = v_{0,0} = v_0$$

A fórmula (V) assume, então, o aspecto:

$$f(k,v) = \underset{r_{i_k}=0,1}{\text{MÍNIMO}} \left(b_{i_k} r_{i_k} + f(k-1, v^* - r_{i_k} v_{i_k})\right) \qquad \text{(VI)}$$

Como para este grupo: $\begin{bmatrix} v_j * v_j = v_0 \\ v_j = -v_j \end{bmatrix}$, temos:

$$f(k,v) = \text{MÍNIMO}(f(k-1,v),\ b_{ik} + f(k-1, v^* v_{i_k})) \quad \begin{bmatrix} k=1,\ldots,r \\ i_k \in I_k \end{bmatrix}$$

<u>APLICAÇÃO</u>: Consideremos o problema:

$$\text{MINIMIZAR} \quad 5r_{1,0} + 3r_{0,1} + r_{1,1}$$

$$r_{1,0}v_{1,0} * r_{0,1}v_{0,1} * r_{1,1}v_{1,1} = v_{1,0}$$

$$r_{i,j} \geq 0 \quad \text{e} \quad \text{inteiro}$$

sobre $G(2,2)$

<u>INICIALIZAÇÃO</u>:

$f(0, v_{0,0}) = 0$

$f(0,v) = M$, $v \neq v_{0,0}$

K=1

$I_k = \{(1,0)\} \rightarrow i_k = (1,0)$

$f(1,v_{0,0}) = \text{MIN}\{f(0,v_{0,0}),\ b_{1,0} + f(0, v_{0,0} * v_{1,0})\} = 0$

$f(1,v_{1,0}) = \text{MIN}\{f(0,v_{1,0}),\ b_{1,0} + f(0, v_{1,0} * v_{1,0})\} =$

$= \text{MIN}\{M,\ 5 + f(0,v_{0,0})\} = 5$

$$f(1,v_{0,1}) = MIN\{f(0,v_{0,1}),\ b_{1,0} + f(0,v_{0,1}{}^*v_{1,0})\} =$$

$$= MIN\{M,\ 5 + f(0,v_{1,1})\} = MIN\{N,M\} = M$$

$$f(1,v_{1,1}) = MIN\{f(0,v_{1,1}),\ b_{1,0} + f(0,v_{1,1} * v_{1,0})\} = M$$

K = 2

$$I_k = \{(1,0),(0,1)\} \rightarrow i_k \in \{(1,0),(0,1)\}$$

$$f(2,v_{0,0}) = MIN\{f(1,v_{0,0}),b_{1,0}+f(1,v_{0,0}{}^*v_{1,0}),b_{0,1}+f(1,v_{0,0}{}^*v_{0,1})\}$$

$$= MIN\{0,\ 5 + f(1,v_{1,0}),\ 3 + f(1,v_{0,1})\} =$$

$$= MIN\{0,\ 5+5,\ 3+M\} = 0$$

$$f(2,v_{1,0}) = MIN\{f(1,v_{1,0}),b_{1,0}+f(1,v_{1,0}{}^*v_{1,0}),b_{0,1}+(1,v_{1,0}{}^*v_{0,1})\}$$

$$= MIN\{5,5+0,3+M\} = 5$$

$$f(2,v_{0,1}) = MIN\{f(1,v_{0,1}),b_{1,0}+f(1,v_{0,1}{}^*v_{1,0}),b_{0,1}+f(1,v_{0,1}{}^*v_{0,1})\}$$

$$= MIN\{M,5+M,3+0\} = 3$$

$$f(2,v_{1,1}) = MIN\{f(1,v_{1,1}),b_{1,0}+f(1,v_{1,1}{}^*v_{1,0}),b_{0,1}+f(1,v_{1,1}{}^*v_{0,1})\}$$

$$= MIN\{M,5+M,3+5\} = 8$$

K=3

$$I_k = \{(1,0),(0,1),(1,1)\} \rightarrow i_k \in \{(1,0),(0,1),(1,1)\}$$

$$f(3,v_{0,0}) = MIN\{f(2,v_{0,0}),b_{1,0}+f(2,v_{0,0}{}^*v_{1,0}),b_{0,1}+f(2,v_{0,0}{}^*v_{0,1}),$$

$$b_{1,1}+f(2,v_{0,0}{}^*v_{1,1})\} = MIN\{0,5+5,3+3,1+8\} = 0$$

$$f(3,v_{1,0}) = \text{MIN}\{f(2,v_{1,0}), b_{1,0}+f(2,v_{1,0}*v_{1,0}), b_{0,1}+f(2,v_{1,0}*v_{0,1}),$$
$$b_{1,1} + f(2,v_{1,0}*v_{1,1})\} = \text{MIN}\{5, 5+0,\ 3+8,\ 1+3\} = 4$$

<u>Obtenção dos valores dos r_{ik}</u>

A solução $f(3,v_{1,0}) = 4$ é obtida para $i_k=(1,1)$. Sendo assim $r_{1,1}$ é a variável correspondente a esta solução. De acordo com a fórmula (VI), $r_{1,1}$ pode ser 0 ou 1 . É <u>0</u> se o mínimo for o elemento $f(k-1,\ v_{1,0})$ e <u>1</u> se o mínimo for o elemento $b_{ik} + f(k-1,\ v_{1,0}*v_{ik})$. Como neste caso o mínimo é o elemento $b_{ik} + f(k-1,\ v_{1,0}*v_{ik})$, onde $i_k = (1,1)$, $r_{1,1}$ assume o valor 1 . A fim de obtermos os demais valores para os r_{ik} procederemos como no caso do problema de mochila simples. Assim, temos:

$f(3,v_{1,0}) = 4$ com $r_{1,1} = 1$

$f(k-1,v_{1,0}*(-r_{1,1}v_{1,1})) = f(2,v_{1,0}*-v_{1,1}) = F(2,v_{1,0}*v_{1,1}) =$

$= f(2,v_{0,1}) = 3$. Como esta solução é obtida para $i_k = (0,1)$, então $r_{0,1}$ é a variável correspondente, e neste caso $r_{0,1} = 1$.

$f(k-1,v_{0,1}*(-r_{0,1}v_{0,1}))=f(1,v_{0,1}*v_{0,1})=f(1,v_{0,0}) = 0$. Como esta solução é obtida para $i_k = (1,0)$, então $r_{1,0}$ é a variável correspondente, e neste caso $r_{1,0} = 0$.

Assim, temos:

$f(3,v_{1,0}) = 4$; $r_{1,1} = r_{0,1} = 1$ e $r_{1,0} = 0$

2.2.2. Técnica II

Se $r_{i_k} = 0$ resolve (V) para um dado k e $v \in V$, então $f(k,v) = f(k-1,v)$. Por outro lado, se $r_{ik} \geq 1$ é uma solução ótima para (V), então:

$$f(k,v) = b_{i_k} + f(k, v * -v_{i_k})$$

Desta feita, analogamente à fórmula obtida no estudo das técnicas de programação dinâmica para o problema mochila simples, temos para todo v e $k=1,\ldots,n$:

$$f(k,v) = \text{MIN}\ \{f(k-1,v), b_{i_k} + f(k, v^* - v_{i_k})\} \qquad \text{(VII)}$$

A condição inicial continua sendo:

$$f(0,v) = \begin{cases} 0 & \text{se} \quad v = v_{0,0} \\ M & \text{se} \quad v \neq v_{0,0} \end{cases}$$

Para computar $f(k,v)$ a partir de VII, $f(k, v^* - v_{i_k})$ deve ser conhecido e os elementos do grupo devem estar ordenados tais que $v^* - v_{i_k}$ precede v. Embora, em geral, uma ordenação para todo $v \in V$ não seja possível, nós podemos ordenar os elementos de uma classe lateral. As computações de $f(k,v)$ são feitas separadamente para cada classe distinta e pode ser dividida em dois casos.

1º CASO: $v \in M(v_{i_k})$, subgrupo gerado por v_{i_k}.

Então existe n, $0 \leq n < |v_{i_k}|$ tal que

$v = n\, v_{ik}$. Assim:

$v * - v_{i_k} = n\, v_{i_k} * - v_{i_k} = (n-1) v_{i_k}$ e para todo

$v \varepsilon M(v_{i_k})$, VII pode ser escrita como:

$$f(k, n\, v_{i_k}) = \text{MIN} \{f(k-1, n\, v_{i_k}),\ b_{i_k} + f(k, (n-1) v_{i_k})\} \quad \text{(VIII)}$$

Visto que $f(k, v_{0,0}) = 0$, $\forall k$, (VIII) pode ser usada para calcular $f(k,v)$, $\forall v \varepsilon M(v_{i_k})$, começando com $n = 1$ e terminando com $n = |v_{i_k}| - 1$.

<u>Obs.</u>: "Se $|v_{i_k}| = |G|$, VIII fornece $f(k,v)$ para todo v" .

<u>2º CASO</u>: $v \not\varepsilon M(v_{i_k})$

Consideremos a classe lateral:

$M_v(v_{i_k}) = \{h | h = v * n\, v_{ik}\ ,\ 0 \leq n < |v_{i_k}|\}$

Para todo $h \varepsilon M_v(v_{i_k})$, temos:

$h * - v_{i_k} = v * n\, v_{i_k} * - v_{i_k} = v * (n-1) v_{i_k}$

Então, para $n = 1, \ldots, |v_{i_k}| - 1$, (VII) pode ser escrita:

$$f(k, v * n\, v_{i_k}) = \text{MIN}\{f(k-1, v * n\, v_{i_k}),\ b_{i_k} + f(k, v * (n-1) v_{i_k})\} \quad \text{(IX)}$$

Para começar a aplicar IX, $f(k, v * n\, v_{i_k})$ deve ser conhecido para algum n. Demonstraremos a seguir um lema que nos fornece uma forma de obter este valor.

<u>LEMA</u>: Se $f(k-1, v * n^o v_{i_k}) = \underset{0 \le n < |v_{i_k}|}{\text{MÍNIMO}} f(k-1, v * n\, v_{i_k})$ (X)

então: $f(k, v * n^o v_{i_k}) = f(k-1, v * n^o v_{i_k})$.

<u>Demonstração</u>:

A partir de (V) podemos escrever:

$$f(k, v * n^o v_{i_k}) = \underset{r_{i_k}=0,\ldots,|v_{i_k}|-1}{\text{MÍNIMO}} (b_{i_k} r_{i_k} + f(k-1, v * n^o v_{i_k} * - r_{i_k} v_{i_k})) \qquad (A)$$

A hipótese deste lema nos permite escrever:

$$f(k-1, v * n^o v_{i_k}) \le f(k-1, v * n\, v_{i_k}) \ , \quad 0 \le n < |v_{i_k}|$$

Como $n^o v_{i_k} * - r_{i_k} v_{i_k} \ \varepsilon \ M_v(v_{i_k})$, temos:

$f(k-1, v * n^o v_{i_k}) \le f(k-1, v * n^o v_{i_k} * - r_{i_k} v_{i_k})$, com $r_{i_k} = 0, \ldots, |v_{i_k}|-1$, pois $n^o v_{i_k} * - r_{i_k} v_{i_k} = (n^o - r_{i_k}) v_{i_k}$ e $0 \le n^o - r_{i_k} < |v_{i_k}|$

Sendo assim, o mínimo em (A) é obtido para:

$$f(k-1, v * n^o v_{i_k} * - r_{i_k} v_{i_k}) = f(k-1,\ v * n^o v_{i_k})$$

e

$$b_{i_k} = 0$$

Logo: $f(k, v * n^o v_{i_k}) = f(k-1, v * n^o v_{i_k})$

A partir desse LEMA o caminho óbvio para determinar n^o é resolvendo (X). Então (IX) poderia ser resolvida para $n = n^o+1, \ldots, n^o + |v_{ik}|-1$. Isto forneceria $f(k, v * nv_{i_k})$, $n = 0, \ldots, |v_{i_k}|-1$, visto que:

$$f(k, v * nv_{i_k}) = f(k, v*(n+|v_{i_k}|)v_{i_k})$$

IMPORTANTE

O LEMA demonstrado acima embora ofereça recurso para determinação do valor de $f(k, v*(n-1)v_{i_k})$, sem o qual a fórmula IX é não funcional, implica em um procedimento pouco eficaz computacionalmente. Teríamos que realizar para cada cálculo de $f(k, v*n\ v_{i_k})$ uma sequência de comparações para a obtenção de n^o , com o qual, segundo o LEMA, ficaria determinado o termo desconhecido da fórmula IX, possibilitando a obtenção de $f(k, v*n\ v_{i_k})$. Isto nos leva a utilizar um outro procedimento, o qual não envolve tantas computações.

RECURSO ESPECIAL

O procedimento que estudaremos a seguir determina n^o , sem que sejam feitas tantas computações como sugere o LEMA, e simultaneamente calcula os valores de $f(k,v * n\ v_{i_k})$. Sendo assim ele é DUPLAMENTE VANTAJOSO.

Seja $f(k,v) = f(k-1,v)$. A partir desta consideração obtemos, usando a fórmula IX, valores, então, ESTIMADOS para $f(k,v * n\ v_{i_k})$, que denotaremos $f'(k,v * n\ v_{i_k})$. Isto fornece:

PARA n = 1:

$$f'(k,v*v_{i_k}) = \text{MIN}\{f(k-1,v*v_{i_k}), b_{i_k} + f(k,v*(1-1)v_{i_k})\}$$

Como: $f(k,v*v_{0,0}) = f(k,v)$ e $f(k,v) = f(k-1,v)$ pela consideração inicial, temos:

$$f'(k,v*v_{i_k}) = \text{MIN}\{f(n-1,v*v_{i_k}), b_{i_k} + f(k-1,v)\} \qquad \text{(XI)}$$

PARA $n = 2,\ldots,|v_{i_k}|-1$

$$f'(k,v*n\ v_{i_k}) = \text{MIN}\{f(k-1,v*n\ v_{i_k}), b_{i_k} + f'(k,v*(n-1)v_{i_k})\}$$

$$\text{(XII)}$$

Como $f(k,v) \leq f(k-1,v)$, segue que para todo n:

$f'(k,v*n\ v_{i_k}) \geq f(k,v*n\ v_{i_k})$, pois os valores ESTIMADOS são obtidos assumindo-se que $f(k,v) = f(k-1,v)$.

Para $n = n^o$ (XII) torna-se:

$f'(k,v*n\ v_{i_k}) = f(k-1,v*n\ v_{i_k})$ (XIII), (lembremos que n^o é o menor n tal que o MÍNIMO em (XII) é

$f(k-1, v^* n\, v_{i_k}))$

<u>AFIRMAÇÃO</u>: Para $n \geq n^o$, (XIII) implica em:

$$f'(k, v^* n\, v_{i_k}) = f(k, v^* n\, v_{i_k}) \qquad \text{(XIV)}$$

<u>PROVA</u>:

Pelo lema isto é verdade para $n = n^o$, ou seja:

$$f'(k, v^* n^o v_{i_k}) = f(k, v^* n^o v_{i_k}) \qquad \text{(A)}$$

<u>Para $n = n^o + 1$</u>, (XII fica):

$$f'(k, v^*(n^o+1) v_{i_k}) = \text{MIN}\{f(k-1, v^*(n^o+1) v_{i_k}, b_{ik} + f'(k, v^* n^o v_{i_k})\}$$

A partir de (A) a expressão acima fica:

$$f'(k, v^*(n^o+1) v_{i_k} = \text{MIN}\{f(k-1, v^*(n^o+1) v_{i_k}, b_{ik} + f(k, v^* n^o v_{i_k})\} \qquad \text{(B)}$$

Por (IX) podemos escrever:

$$f'(k, v^*(n^o+1) v_{i_k}) = \text{MIN}\{f(k-1, v^*(n^o+1) v_{i_k}), b_{i_k} + f(k, v^* n^o v_{i_k}\} \qquad \text{(C)}$$

Comparando (B) e (C) comprovamos que:

$$f'(k, v^*(n^o+1) v_{i_k} = f(k, v^*(n^o+1) v_{i_k}) \qquad \text{(D)}$$

<u>Para $n = n^o + 2$</u>, (XII) fica:

$$f'(k, v^*(n^o+2) v_{i_k}) = \text{MIN}\{f(k-1, v^*(n^o+2) v_{i_k}, b_{i_k} + f'(k, v^*(n^o+1) v_{i_k})\} \qquad \text{(E)}$$

A partir de (D) a expressão acima assume o aspecto:

$$f'(k, v^*(n^o+2)v_{i_k} = MIN\{f(k-1, v^*(n^o+2)v_{i_k}, b_{i_k} + f(k, v^*(n^o+1)v_{i_k})\}$$

Obtendo o valor de $f(k, v^*(n^o+2)v_{i_k}$ pela fórmula IX, como foi feito em (C), concluímos que:

$$f'(k, v^*(n^o+2)v_{i_k} = f(k, v^*(n^o+2)v_{i_k}$$

Continuando este raciocínio para $n = n^o+3, \ldots, n^o+|v_{i_k}|-1$, concluimos que:

$$f'(k, v^*n\ v_{i_k}) = f(k, v^*n\ v_{i_k}) \quad , \quad n \geq n^o$$

<u>OBSERVAÇÃO</u>:

Este RECURSO ESPECIAL referente ao 2º caso desta TÉCNICA II, ou seja, quando $v \notin M(v_{i_k})$, nos sugere o seguinte procedimento para obtenção dos $f(k, v^*n\ v_{i_k})$:

<u>PROCEDIMENTO ESPECIAL</u>

(1) ESCOLHA UM $v \notin M(v_{i_k})$

(2) ASSUMA QUE $f(k,v) = f(k-1,v)$

(3) CALCULE n^o APLICANDO (XI) e (XII)

<u>OBS.</u>:

Como n^o é o menor n tal que o mínimo em (XII) é $f(k-1, v^*n\ v_{i_k})$, iniciamos estes cálculos com $n=1$ e para-mos quando o MÍNIMO referente à fórmula XII for este primeiro elemento. O n tal que isto ocorrer é o valor de n^o.

(4) CALCULE OS VALORES DE $f(k, v * n\ v_{i_k})$

OBS.:

Duas situações podem ocorrer e, portanto, dois caminhos distintos surgirão para obtenção dos valores de $f(k, v * n\ v_{i_k})$.

1ª SITUAÇÃO: $n^o < |v_{i_k}|$

(4.1) Faça $f(k, v*n^o v_{i_k}) = f'(k, v*n^o v_{i_k})$ e então use (IX) para calcular $f(k, v*nv_{i_k})$ para $n = n^o+1, \ldots,$ $n^o + |v_{i_k}|-1$. Observe que no cálculo de $f(k, v*(n^o+1)v_{i_k}) =$ $= \mathrm{MIN}\{f(k-1,\ v*(n^o+1)v_{i_k}), b_{i_k} + f(k, v*n^o v_{i_k})\}$, já não mais existe o embaraço de $f(k, v*n_o v_{i_k})$ ser desconhecido.

2ª SITUAÇÃO: $n^o = |v_{i_k}|$

CAMINHO A SEGUIR:

(4.2) Faça $f(k, v*n\ v_{i_k}) = f'(k, v*n\ v_{i_k})$, $1 \leq n \leq |v_{i_k}|$

OBS.: Neste caso os valores do n^o e $f(k, v*n\ v_{i_k})$ são calculados simultaneamente. Abaixo este procedimento é justificado.

Como para $n \geq n^o$, $f'(k, v*n\ v_{i_k}) = f(k, v*n\ v_{i_k})$, temos:

PARA $n = n^o$

$$f'(k,v^*|v_{i_k}|v_{i_k}) = f(k,v^*|v_{i_k}|v_{i_k})$$

ou

$$f'(k,v^*v_0) = f(k,v^*v_0) \rightarrow f'(k,v) = f(k,v)$$

Sendo assim, $n^o = |v_{i_k}|$ é equivalente a $n^o = 0$, pois:

$$f'(k,v^*0.v_{i_k}) = f(k,v^*0.v_{i_k}) \rightarrow f'(k,v) = f(k,v)$$

Logo deveríamos aplicar a fórmula (IX) para calcular $f(k,v^*n\ v_{i_k})$, $n = n^o+1,\ldots,n^o+|v_{i_k}|$. Como $n^o = |v_{i_k}|$ é equivalente a $n^o = 0$ os cálculos seriam feitos para $n = 1,\ldots,|v_{i_k}|$. Entretanto como $n^o = |v_{i_k}|$, todos os valores $f'(k,v^*n\ v_{i_k})$ para $n = 1,\ldots,|v_{i_k}|$ já foram calculados. Sendo, para $n \geq n^o$, $f(k,v^*n\ v_{i_k}) = f'(k,v^*n\ v_{i_k})$, mais nenhum cálculo necessita ser efetuado; basta tornar reais os valores estimados, ou seja:

$$f(k,v^*n\ v_{i_k}) = f'(k,v^*n\ v_{i_k})$$

<u>I M P O R T A N T E</u>

Caso exista algum v_i não pertencente à classe lateral $M_V(v_{i_k})$, para o v escolhido, nova escolha deverá ser feita, implicando em outra classe lateral. O PROCEDIMENTO ESPECIAL é então repetido. (Este fato ocorre no exemplo resolvido a seguir).

APLICAÇÃO DA TÉCNICA II

Consideremos o seguinte problema sobre G(6).

$$\text{MINIMIZAR} \quad 3r_3 + 10r_2 + 5r_5$$
$$r_3v_3 * r_2v_2 * r_5\ddot{v}_5 = v_1$$
$$r_3, r_2, r_5 \geq 0 \quad \text{e inteiro}$$

A TÉCNICA II CONSISTE EM OBTER PARA CADA v_{i_k} O SUBGRUPO $M(v_{i_k})$ E APLICAR O 1º CASO OU O 2º CASO, CONFORME v_1, v_2, v_3, v_4, v_5 PERTENÇAM OU NÃO A ESTE SUBGRUPO. Os v_{i_k}'s SÃO CONSIDERADOS NA ORDEM APRESENTADA PELO PROBLEMA. ASSIM SENDO A ORDEM A SER OBEDECIDA É: v_3, v_2, v_5 .

Condições iniciais:
$$\left[\begin{array}{ll} f(0,v) = M , & v \neq v_0 \\ f(0,v) = 0 , & v = v_0 \\ f(k,v_0) = 0 , & \forall k \end{array}\right.$$

K=1

$$i_k = 3 \quad ; \quad b_{ik} = b_3 = 3$$

Subgrupo gerado por $v_3 = \{v_0, v_3\}$. Logo na obtenção de $f(1,v_0)$ e $f(1,v_3)$ utilizaremos a fórmula VIII, 1º CASO.

$f(1,v_0) = 0$ (pela inicialização, sem necessidade de

VIII)

$$f(1,v_3) = \text{MIN}\{f(0,v_3),\ 3 + f(1,0 \cdot v_3)\} \rightarrow \quad f(1,v_3) = 3$$

Como v_1, v_2, v_4 e v_5 não pertencem ao subgrupo gerado por v_3, o cálculo de $f(1,v_i)$, $i = 1,2,4$ e 5 é obtido pelo PROCEDIMENTO ESPECIAL relativo ao 2º CASO.

(1) Escolha: $v = v_2$

(2) $f(1,v_2) = f(0,v_2) = M$

(3) Cálculo de n^o

$$n = 1 \rightarrow f'(1,v_2*v_3) = \text{MIN}\{f(0,v_2*v_3),\ 3 + f(0,v_2)\} \rightarrow$$

$$\rightarrow f'(1,v_5) = M$$

$$n = 2 \rightarrow f'(1,v_2*2v_3) = \text{MIN}\{f(0,v_2*2v_3),\ 3 + f'(1,v_2*v_3)\} =$$

$$= \text{MIN}\{f(0,v_2),\ 3 + f'(1,v_5)\} \rightarrow$$

$$\rightarrow f'(1,v_2) = M$$

Como $v_2*3v_3 = v_5$ e $v_2*4v_3 = v_2$ verificamos que v_2 e v_5 são os únicos elementos da classe lateral $M_{v_2}(v_3)$. Obteremos então os valores de $f(1,v_5)$ e $f(1,v_2)$ e, posteriormente faremos nova escolha de v o que nos fornecerá uma outra classe lateral.

Podemos considerar $n^o = 1$

(4) Cálculo de $f(1,v_2*n\,v_3)$

Como $n^o < |v_3| = 2$, seguiremos o procedimento indicado em 4.1, relativo à 1ª SITUAÇÃO.

$$f(1,v_5) = f'(1,v_5) \quad \rightarrow \quad f(1,v_5) = M$$

Usaremos, então, IX para obter $f(1,v_2*n\ v_3)$ para $n = 2$. Assim:

$$f(1,v_2*2v_3) = \mathrm{MIN}\{f(0,v_2*2v_3),\ 3 + f(1,v_2*v_3)\}$$

$$= \mathrm{MIN}\{M,\ 3 + M\} \rightarrow \quad f(1,v_2) = M$$

Como faltam os valores de $f(1,v_1)$ e $f(1,v_4)$ nova escolha será feita para v.

(1) Escolha: $v = v_1$

(2) $f(1,v_1) = f(0,v_1) = M$

(3) Cálculo de n^o :

$$n = 1 \rightarrow f'(1,v_1*v_3)=\mathrm{MIN}\{f(0,v_1*v_3),3+f(0,v_1)\} \rightarrow f'(1,v_4)= M$$

$$n = 2 \rightarrow f'(1,v_1*2v_3)=\mathrm{MIN}\{f(0,v_1*2v_3),3+f'(1,v_1*v_3)\}\rightarrow f'(1,v_1)=M$$

Podemos considerar $n^o = 1$

(4) Cálculo de $f(1,v_1*n\ v_3)$

Como $n^o < |v_3| = 2$, seguiremos o procedimento incidado em 4.1, relativo à 1ª SITUAÇÃO.

$$f(1,v_4) = f'(1,v_4) \rightarrow \quad f(1,v_4) = M$$

Usaremos, então, IX para obter $f(1,v_1*n\ v_3)$ para $n = 2$. Assim:

$$f(1,v_1*2v_3)=\mathrm{MIN}\{f(0,v_1*2v_3),\ 3 + f(1,v_1*v_3)\}$$

$$=\mathrm{MIN}\{M,\ 3 + M\} \rightarrow \quad f(1,v_1) = M$$

OBSERVAÇÃO:

Como para $v \neq v_0$ ou v_3 , $v^{*n}\ v_3 \neq v_0$ para todo n , $f(1,v_i) = M$ para $i=1,2,4,5$ já podia ser considerado, sem os cálculos anteriores, pois (e basta examinar as fórmulas XI e XII) a comparação para obtenção do mínimo é sempre feita entre M e $b_{i_k} + M$ fornecendo portanto M. Os cálculos serviram, no entanto, para comprovar este fato.

Também estes cálculos mostraram a necessidade de uma nova escolha para v , e consequentemente a repetição do PROCEDIMENTO ESPECIAL, já que v_1 e v_4 não pertenciam à primeira classe lateral obtida com a primeira escolha de v .

$K = 2$

$i_k = 2$; $b_{i_k} = b_2 = 10$

Subgrupo gerado por $v_2 = \{v_0, v_2, v_4\}$. Logo na obtenção $f(2,v_0)$, $f(2,v_2)$, $f(2,v_4)$ utilizaremos a fórmula VIII, 1º CASO.

$f(2,v_0) = 0$ (pela inicialização)

$n = 1$:

$f(2,v_2) = \text{MIN}\{f(1,v_2), 10 + f(2,0.v_2) = \text{MIN}\{M,10\} \rightarrow f(2,v_2) = 10$

$n = 2$:

$$f(2,2v_2)=\text{MIN}\{f(1,2v_2),10 + f(2,v_2)\}=\text{MIN}\{M,10+10\} \rightarrow \quad f(2,v_4) = 20$$

Como $v_1, v_3, v_5 \notin M(v_2)$, o cálculo de $f(2,v_i)$, $i=1,3,5$ é obtido pelo PROCEDIMENTO ESPECIAL relativo ao 2º CASO.

(1) Escolha: $v = v_1$

(2) $f(2,v_1) = f(1,v_1) = M$

(3) Cálculo de n^o:

$$n = 1 \rightarrow f'(2,v_1{*}v_2)=\text{MIN}\{f(1,v_1{*}v_2),\ 10 + f(1,v_1)\} \rightarrow f'(2,v_3)=3$$

Logo: $n^o = 1$

(4) Cálculo de $f(2,v_1{*}n\ v_2)$

Como $n^o < |v_2| = 3$, seguiremos o procedimento indicado em 4.1, relativo à 1ª SITUAÇÃO.

$$f(2,v_3) = f'(2,v_3) \quad \rightarrow \quad f(2,v_3) = 3$$

Usaremos, então, IX para obter $f(2,v_1{*}n\ v_2)$ para $n=2,3$. Assim:

$$f(2,v_1{*}2v_2) = \text{MIN}\{f(1,v_1{*}2v_2),10 + f(2,v_1{*}v_2)\}$$
$$= \text{MIN}\{f(1,v_5),10 + f(2,v_3)\} \rightarrow \quad f(2,v_5) = 13$$

$$f(2,v_1{*}3v_2) = \text{MIN}\{f(1,v_1{*}3v_2),10 + f(2,v_1{*}2v_2)\}$$
$$= \text{MIN}\{f(1,v_1),10 + f(2,v_5)\} \rightarrow \quad f(2,v_1) = 23$$

<u>OBSERVAÇÃO</u>: Repetindo esse PROCEDIMENTO ESPECIAL para a escolha $v = v_3$, temos:

(1) Escolha: $v = v_3$

(2) $f(2,v_3) = f(1,v_3) = 3$

(3) Cálculo de n^o :

$n = 1 \rightarrow f'(2,v_3{*}v_2) = \mathrm{MIN}\{f(1,v_3{*}v_2),\ 10 + f(1,v_3)\} \rightarrow f'(2,v_5) = 13$

$n = 2 \rightarrow f'(2,v_3{*}2v_2) = \mathrm{MIN}\{f(1,v_3{*}2v_2),\ 10 + f(2,v_3{*}v_2)\} \rightarrow f'(2,v_1) = 23$

$n = 3 \rightarrow f'(2,v_3{*}3v_3) = \mathrm{MIN}\{f(1,v_3{*}3v_2),\ 10 + f(2,v_3{*}2v_2)\} \rightarrow f'(2,v_3) = 3$

Logo $n^o = 3$

(4) Cálculo de $f(2,v_3{*}n\ v_2)$

Como $n^o = |v_{i_k}|$, seguiremos o procedimento indicado em 4.2, relativo à 2ª SITUAÇÃO. Sendo assim:

$f(2,v_5) = f'(2,v_5) \rightarrow f(2,v_5) = 13$

$f(2,v_1) = f'(2,v_1) \rightarrow f(2,v_1) = 23$

$f(2,v_3) = f'(2,v_3) \rightarrow f(2,v_3) = 3$

<u>CONCLUSÃO</u>: DEPENDENDO DA ESCOLHA A OBTENÇÃO DE $f(k,v{*}n\ v_{i_k})$ É MAIS SIMPLES.

K=3

$i_k = 5$; $b_{i_k} = b_5 = 5$

Verificamos que $M(v_5) = \{v_0,v_1,v_2,v_3,v_4,v_5\}$. As sim sendo o cálculo de $f(3,n\ v_5)$ é obtido pela fórmula VIII, 1º CASO.

$n = 0 : f(3,0.v_5) = f(3,v_0) = 0$ (pela inicialização)

$n = 1 : f(3,1.v_5) = MIN\{f(2,1.v_5), 5 + f(3,0.v_5)\} \rightarrow f(3,v_5) = 5$

$n = 2 : f(3,2.v_5) = MIN\{f(2,2.v_5), 5 + f(3,1.v_5)\} \rightarrow f(3,v_4) = 10$

$n = 3 : f(3,3.v_5) = MIN\{f(2,3.v_5), 5 + f(3,2.v_5)\} \rightarrow f(3,v_3) = 3$

$n = 4 : f(3,4.v_5) = MIN\{f(2,4.v_5), 5 + f(3,3.v_5)\} \rightarrow f(3,v_2) = 8$

$n = 5 : f(3,5.v_5) = MIN\{f(2,5.v_5), 5 + f(3,4.v_5)\} \rightarrow f(3,v_1) = 13$

De acordo com a fórmula IX temos que:

$$f(k,v^*n\ v_{i_k}) = f(k-1,v^*n\ v_{i_k}) \quad \text{com} \quad r_{i_k} = 0$$

ou

$$f(k,v^*n\ v_{i_k}) = f(k,v^*(n-1)v_{i_k}) \quad \text{com} \quad r_{i_k} \geq 1$$

Podemos assim formar a tabela abaixo que envolve as computações relativas ao exemplo acima desenvolvido:

v	f(1,v)	r_3	f(2,v)	r_2	f(3,v)	r_5
v_0	0	0	0	0	0	0
v_1	M	0	23	1	13	1
v_2	M	0	10	1	8	1
v_3	3	1	3	0	3	0
v_4	M	0	20	1	10	1
v_5	M	0	13	1	5	1

Recuperação da Solução Ótima

Procede-se de modo análogo ao caso do problema de mochila simples, onde há buscas verticais e horizontais.

Neste caso começamos com $f(3,v_1)$, onde $n=3$ $b = v_1$, se quizermos fazer uma comparação com a mochila simples na qual fazemos $n = k$ e $g = b$.

Como $f(3,v_1) = 13$ com $r_5 = 1$, iniciamos uma busca vertical, subtraindo seguidamente v_5 de v_1 até obtermos $r_5 = 0$, dando início à busca horizontal.

$$f(3,v_1{}^*\text{-}v_5) = f(3,v_1{}^*v_1) = f(3,v_2) = 8 \quad \text{com} \quad r_5 = 1$$

Obs.: $-v_5 = v_1$, pois $v_5{}^*v_1 = v_0$

$$f(3,v_2{}^*\text{-}v_5 = f(3,v_2{}^*v_1) = f(3,v_3) = 3 \quad \text{com} \quad r_5 = 0$$

Logo: $r_5 = 1+1 \rightarrow r_5 = 2$

A busca horizontal se caracteriza pela subtração de uma unidade de k. Assim devemos investigar $f(3-1,v_3) = f(2,v_3)$.

$f(2,v_3) = 3$ com $r_2 = 0$

Logo $r_2 = 0$ e nova busca horizontal se inicia.

Então devemos investigar: $f(2-1,v_3) = f(1,v_3)$.

$f(1,v_3) = 3$ com $r_3 = 1$. Assim iniciamos nova busca vertical, subtraindo v_3 de v_3 até obtermos $r_3 = 0$.

$$f(1,v_3{}^*\text{-}v_3) = f(1,v_3{}^*v_3) = f(1,v_0) = 0 \quad \text{com} \quad r_3 = 0$$

O processo de recuperação se encerra,pois subtraindo uma unidade deste novo k, obtemos $k = 0$.

SOLUÇÃO ÓTIMA : $f(3,v_1) = 13$

$r_3 = r_2 = 0$

$r_5 = 2$

OBS.: As setas desenhadas na tabela indicam as buscas vertical e horizontal efetuadas.

3. Formulação do algoritmo para solução do Programa Grupo Mochila

Este algoritmo é análogo ao ALGORITMO I formulado para o problema mochila simples. De forma que, embora sabendo que $f(k,v) \geq 0$, utilizaremos a "CONVENÇÃO" de torná-lo negativo com o objetivo de fazer corresponder a essa quantidade negativa o valor 1 para r_{i_k} . Logo $f(k,v) < 0$ implica em incrementação no valor de r_{i_k} .

$$\begin{cases} r_{i_k} \geq 1 & \text{se} \quad f(k,v) < 0 \\ r_{i_k} = 0 & \text{senão} \end{cases}$$

Faremos uso portanto de um contador j para obter os valores ótimos das variáveis r_{i_k} .

ALGORITMO

PASSO 1: Inicialização: $f(0,v) = M$ para $v \neq v_0$

$f(0,v_0) = 0$

$k = 0$

PASSO 2: Faça $k = k+1$

$n = 1$

$f(k,v_0) = 0$

PASSO 3: Calcule $w = b_{i_k} + |f(k,(n-1)v_{i_k})|$. Se $w < |f(k-1,n\ v_{i_k})|$ faça $f(k,n\ v_{i_k}) = -w$. Senão faça $f(k,n\ v_{ik}) = |f(n-1,n\ v_{i_k})|$

C O M E N T Á R I O: Como o sinal negativo, caracteriza apenas $r_{i_k} \geq 1$ é necessário trabalhar com o módulo de determinadas expressões como fizemos. Como este algoritmo se baseia em um problema de minimização ocorre que:

$f(k,v) = f(k-1,v) \rightarrow x_k = 0$ (NÃO HOUVE MELHORA NAS BUSCAS)

$f(k,v) < f(k-1,v) \rightarrow x_k \geq 1$ (HOUVE MELHORA NAS BUSCAS)

Por este motivo temos que $r_{i_k} \geq 1$ se $b_{i_k} + f(k,(n-1)v_{i_k}$ for menor que $f(k-1,n\ v_{i_k})$, o que é representado no algoritmo por $f(k,n\ v_{i_k}) = -w$

PASSO 4: Se $n < |v_{i_k}|-1$, faça $n = n+1$ e volte ao PASSO 3. Se $n = |v_{i_k}|-1$, vá para o PASSO 5.

PASSO 5: Se $f(k,v)$ não foi computado para algum v , vá para o PASSO 6 . Caso contrário vá para o PASSO 10.

C O M E N T Á R I O: Os passo 3,4,5 referem-se se ao 1º CASO. Se algum v for tal que não pertença ao subgrupo gerado por v_{i_k} o 2º caso será utilizado e por conseguinte a classe lateral desse elemento será construída. Iremos portanto para o PASSO 6 que dá início a esse 2º caso. Entretanto se todos os $f(k,v)$ tiverem sido computados, isto é, se todo v pertence a $M(v_{i_k})$ iremos para o PASSO 11 onde se inicia a busca da solução ótima (recuperação dos apontadores).

PASSO 6: Escolha um v tal que $f(k,v)$ não tenha sido computado. Ponha $f(k,v) = |f(k-1,v)|$, $n = 1$ e $n^o = 0$.

C O M E N T Á R I O : Este passo caracteriza o início do PROCEDIMENTO ESPECIAL (2º caso). Pelo fato da convenção usada para caracterizar $r_{i_k} = 0$ ou $r_{i_k} \geq 1$, tornou se necessário a tomada do módulo de $f(k-1,v)$. É necessário zerar n^o , pois caso seja preciso contruir mais de uma classe lateral, n^o tem que ser novamente calculado.

PASSO 7: Se $n \leq |v_{i_k}|$ vá para o PASSO 8. Se $|v_{i_k}| < n < n^o + |v_{i_k}|$ vá para o passo 9. Se $n = n^o + |v_{i_k}|$ vá para o PASSO 5.

PASSO 8: Calcule $w = b_{i_k} + |f(k,v*(n-1)v_{i_k})|$. Se $w < |f(k-1,v*n\ v_{i_k})|$, faça $f(k,v*n\ v_{i_k}) = -w$, $n = n+1$ e volte ao PASSO 7. Caso contrário ponha $f(k,v*n\ v_{i_k}) = |f(k-1,v*n\ v_{i_k})$, e, se $n^o = 0$, ponha $n^o = n$ e volte ao passo 7.

PASSO 9: Faça $n' = n - |v_{i_k}|$. Calcule $W = b_{i_k} + |f(k,v*(n'-1)v_{i_k})|$. Se $w < |f(k-1,v*n'v_{i_k})|$, faça $f(k,v^*n'v_{i_k}) = -w$. Caso contrário ponha $f(k,v^*n'v_{i_k}) = |f(k-1,v^*nv_{i_k})|$, faça $n=n+1$ e volte para 7.

C O M E N T Á R I O S O B R E O S P A S S O S 7, 8, 9 : O PASSO 8 determina n^o, calcula $f'(k,v^*n\, v_{i_k})$, $1 \leq n < n^o$ e calcula $f(k,v^*n\, v_{i_k})$, $n^o \leq n \leq |v_{i_k}|$. Por este motivo o PASSO 7 ordena a passagem ao PASSO 8 desde que $n \leq |v_{i_k}|$. O PASSO 9 calcula $f(k,v^*n\, v_{i_k})$, $|v_{i_k}| < n < n^o + |v_{i_k}|$ justificando a ordem dada ao PASSO 7. Como podem haver ainda $f(k,v^\bullet)$ desconhecidos, isto é, $v^\bullet$ não pertence à classe lateral obtida, o PASSO 7 ordena que para $n = n^o + |v_{i_k}|$ retorne-se ao PASSO 5, onde esta verificação é feita.

Observa-se que, se $n^o = |v_{i_k}|$, os valores de $f(k,v^*n\, v_{i_k})$ para $n \neq n^o$ senão todos obtidos pelo PASSO 9 e, se $n^o = 1$, os valores de $f(k,v^*n\, v_{i_k})$ serão todos obtidos no PASSO 8. Verificamos assim a necessidade do PASSO 9 para $n^o > 1$, pois até $n = n^o - 1$ os valores de $f(k,v^*n\, v_{i_k})$ são estimados e representados por $f'(k,v^*n\, v_{i_k})$.

Os módulos das expressões tornam-se necessários pela convenção usada para caracterizar os valores de r_{i_k} (0 ou ≥ 1).

PASSO 10: Se $K < N$, volte ao PASSO 2

Se $K = N$, vá para o PASSO 11

C O M E N T Á R I O: Quando este passo for realizado pela última vez, teremos calculados todos os valores dos $f(k,v)$.

PASSO 11: Faça $v = v^o$ e $j = 0$. Vá para o PASSO 12.

C O M E N T Á R I O: O valor de v^o se corresponde com o valor de b na mochila simples (Ver fórmula II). Utilizamos também aqui um contador j que acompanha as incrementações de uma unidade das variáveis r_{i_k} .

PASSO 12: Se $f(k,v) < 0$ faça $j=j+1$ e vá para o PASSO 13.
Senão vá para o PASSO 14.

PASSO 13: Faça $v = v^* - v_{i_k}$ e vá para o PASSO 12.

C O M E N T Á R I O: Este passo caracteriza a busca vertical.

PASSO 14: Faça $r_{i_k} = j$ e $j = 0$

Se $k > 1$ faça $k=k-1$ e volte ao PASSO 12.

Se $k=1$, faça $f(N,v^o) = |f(N,v^o)|$ e PARE.

C O M E N T Á R I O: Ficam os valores das variáveis r_{i_k} determinadas neste passo. Após o cálculo de cada uma dessas variáveis, é feita a busca horizontal $(k=k-1)$ e sugerida uma nova busca vertica (se $k > 1$) ou encerram-se os cálculos $(k=1)$. Devemos tomar o módulo de $f(N,v^o)$ pois essa quantidade pode estar acompanhada de um sinal negativo por causa da convenção usada.

4. Relação entre o Problema Mochila Simples e o Problema Mochila Grupo

Considerando o problema mochila simples, temos a seguinte formulação:

$$\text{MAXIMIZAR} \quad X_o = \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_j x_j \leq b$$

$$x_j \geq 0 \text{ e inteiro }, \quad j=1,\ldots,n$$

Adicionando a variável de folga x_{n+1} a fim de obter uma igualdade como restrição, temos:

$$\text{MAXIMIZAR} \quad X_o = \sum_{j=1}^{n+1} c_j x_j$$

$$\sum_{j=1}^{n+1} a_j x_j = b \tag{1}$$

$$x_j \geq 0 \text{ e inteiro}, \ j=1,\ldots,n+1$$

$$a_{n+1} = 1 \quad , \quad c_{n+1} = 1$$

Fazendo $\rho_i = c_i/a_i$ e assumindo que as variáveis estão ordenadas de tal forma que $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$, uma solução ótima para o problema linear correspondente a (1) é:

$$x_1 = b/a_1$$

$$x_o = \rho_1 \, b$$

$$x_j = 0 \quad , \quad j \neq 1$$

Escrevendo x_1 em termos das variáveis restantes, vem:

$$\sum_{j=1}^{n+1} a_j x_j = b \rightarrow \quad x_1 = \frac{b}{a_1} - \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} \tag{1a}$$

Logo:

$$X_o = C_1\left(\frac{b}{a_1} - \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1}\right) + \sum_{j=2}^{n+1} c_j x_j$$

$$X_o = \rho_1 b - c_1 \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} + \sum_{j=2}^{n+1} c_j x_j$$

$$X_o = \rho_1 b - \left(\sum_{j=2}^{n+1} \rho_1 a_j x_j - \sum_{j=2}^{n+1} c_j x_j\right)$$

$$X_o = \rho_1 b - \sum_{j=2}^{n+1} (\rho_1 a_j - c_j) x_j \tag{1b}$$

onde $\rho_1 a_j - c_j \geq 0$ para $j=2,\ldots,n+1$, pois:

$$\rho_1 a_j - c_j = c_j(\rho_1 a_j / c_j - 1) = c_j(\rho_1/\rho_j - 1) \qquad \text{e}$$

$\rho_1/\rho_j - 1 \geq 0$, $j=2,\ldots,n+1$

Suponhamos que b/a_1 não é inteiro, caso contrário a solução do programa linear resolve o programa inteiro (1). De acordo com as novas expressões para x_1 e x_o podemos escrever (1) como:

$$\text{MINIMIZAR} \quad \sum_{j=2}^{n+1} (c_1 a_j - c_j a_1) x_j$$

$$\frac{b}{a_1} - \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} \geq 0 \quad \text{e} \quad \text{inteiro} \tag{2}$$

$$x_j \geq 0 \quad \text{e inteiro, } j=2,\ldots,n+1$$

Observe que:

$$\text{MAX}\left[\rho_1 b - \sum_{j=2}^{n+1} (\rho_1 a_j - c_j) x_j\right] \to \text{MIN} \sum_{j=2}^{n+1} (\rho_1 a_j - c_j) x_j \quad \to$$

$$\to \text{MIN} \sum_{j=2}^{n+1} (c_1 a_j - c_1 a_1) x_j$$

Suponhamos agora que b é suficientemente grande tal que x_1 é certamente positivo na solução ótima. Este fato é caracterizado por $d(g) = 1$. (A obtenção de um g^* tal que todo $b > g^*$ implique em $x_1 > 0$ foi vista no capítulo II, seção 1.4). Sendo assim a não negatividade de x_1 pode ser abandonada. Desta feita a 2a. restrição do nosso problema, torna-se:

$$\frac{b}{a_1} - \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} \quad \text{inteiro} \tag{3}$$

Sabemos que se dois números a e b são congruentes módulo m então $a - b = km$, k inteiro. Sendo $a-b$ um número inteiro então podemos escrever $a-b = k.1$, ou seja $a \equiv b \pmod 1$. Assim por (3) podemos escrever:

$$\frac{b}{a_1} = \sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} \pmod 1$$

Como a congruência módulo m é uma relação de equivalência, temos:

$$\sum_{j=2}^{n+1} \frac{a_j x_j}{a_1} \equiv \frac{b}{a_1} \pmod 1$$

Então:

$$\frac{1}{a_1}\left(\sum_{j=2}^{n+1} a_j x_j - b\right) = k.1 \quad , \quad k = \text{inteiro}$$

ou

$$\sum_{j=2}^{n+1} a_j x_j - b = k.a_1$$

O que equivale a:

$$\sum_{j=2}^{n+1} a_j x_j \equiv b \pmod{a_1}$$

ou

$$\sum_{j=2}^{n+1} (a_j (\text{mod } a_1)) x_j \equiv b (\text{mod } a_1) \tag{4}$$

Sendo assim (2) pode ser escrito:

$$\text{MINIMIZAR} \quad \sum_{j=2}^{n+1} (c_1 a_j - c_j a_1) x_j$$

$$\sum_{j=2}^{n+1} (a_j (\text{mod } a_1)) x_j \equiv b (\text{mod } a_1) \tag{5}$$

$$x_j \geq 0 \text{ e inteiro} \quad , \quad j=2,\ldots,n+1$$

Desenvolvendo (4), vem:

$$a_2(\text{mod } a_1)x_2 + \dots + a_{n+1}(\text{mod } a_1)x_{n+1} \equiv b(\text{mod } a_1) \qquad (6)$$

Podemos escrever (6) como uma equação de grupo sobre o grupo $G(a_1)$ da seguinte forma:

$$v_{a_2(\text{mod } a_1)}t_{a_2(\text{mod } a_1)} * \dots * v_{a_{n+1}(\text{mod } a_1)}t_{a_{n+1}(\text{mod } a_1)} = v_{b(\text{mod } a_1)} \qquad (7)$$

Ou seja:

$$\sum_{j=2}^{n+1}(a_j(\text{mod } a_1))x_j \equiv b(\text{mod } a_1)$$

é equivalente a:

$$\sum_{j=2}^{n+1} v_{a_j(\text{mod } a_1)} \cdot\cdot\, t_{a_j(\text{mod } a_1)} = v_{b(\text{mod } a_1)} \qquad (8)$$

onde $t_p = \sum_{j\varepsilon T_p} x_j$, $T_p = \{j \mid a_j(\text{mod } a_1) = p \ , \ j=2,\dots,n+1\}$

Esta expressão para $t_{a_j(\text{mod } a_1)}$, a qual nos oferece uma relação entre as variáveis x e t, é baseada no seguinte: Suponhamos que para $j=k$ e $j=\ell$ ocorre $a_k(\text{mod } a_1) = a_\ell(\text{mod } a_1) = h$. Então $v_h t_h$ aparece duas vezes no desenvolvimento de (8), e assim temos $2v_h t_h = v_{2h} t_h$ implicando em $t_h = x_k + x_\ell$.

Pela observação feita acima verificamos que a variável t pode determinar diretamente x ou pode representar a soma de várias variáveis x . Se ocorrer o segundo caso basta lembrar que se:

$c_1a_k - c_ka_1 < c_1a_\ell - c_\ell a_1 \rightarrow x_\ell = 0$, pois nosso intuito é minimizar $\sum_{j=2}^{n+1} (c_1a_j - c_ja_1)x_j$.

Nosso objetivo agora será obter uma expressão equivalente para a função objetivo:

$$\sum_{j=2}^{n+1} (c_1a_j - c_ja_1)x_j$$

Como desejamos minimizar $\sum_{j=2}^{n+1} (c_1a_j - c_ja_1)x_j$, podemos transformar esta expressão em:

$$\text{MINIMIZAR} \sum_{j=2}^{n+1} d_{a_j(\text{mod } a_1)} \, t_{a_j(\text{mod } a_1)} \tag{9}$$

onde $d_{a_j(\text{mod } a_1)}$ será agora analisado.

Lembrando que pode ocorrer para $j=k$ e $j=\ell$: $a_{k}(\text{mod } a_1) = a_\ell(\text{mod } a_1) = h$, teríamos $d_h \, t_h$ surgindo duas vezes como parcela em (9), o que é equivalente a $2d_ht_h$. Como $d_{a_j(\text{mod } a_1)}$ representa a expressão $c_1a_j-c_ja_1$, que valor deverá ser atribuído a j neste caso: k ou ℓ ? Como o problema é de minimização este impasse se desfaz desde que estabeleçamos que:

$$d_p = \min_{j \varepsilon T_p} (c_1 a_j - c_j a_1) \tag{9a}$$

Sendo assim j receberá, por exemplo, o valor k se $c_1 a_1 - c_k a_1$ for menor que $c_1 a_\ell - c_\ell a_1$.

Podemos então escrever (5) como um problema de mochila grupo (desde que b seja suficientemente grande) da seguinte forma:

$$\text{MINIMIZAR} \quad \sum_{j=2}^{n+1} d_{a_j(\text{mod } a_1)} \, t_{a_j(\text{mod } a_1)}$$

$$\sum_{j=2}^{n+1} v_{a_j(\text{mod } a_1)} \, t_{a_j(\text{mod } a_1)} = v_{b(\text{mod } a_1)} \tag{10}$$

$$t_{a_j(\text{mod } a_1)} \geq 0 \quad \text{e inteiro, } j=2,\ldots,n+1$$

A partir da solução ótima de (10) podemos obter a solução ótima de (1), pois há uma equivalência entre esses dois problemas como vimos, desde que b seja suficientemente grande.

IMPORTANTE:

A vantagem em transformar (1) em (10) é que sendo a ordem do grupo a_1 , onde $a_1 \leq b$, pode acarretar em um número menor de cálculos, já que em (1) o número de cálculos é proporcional a b e em (10) é proporcional a a_1 .

APLICAÇÃO:

Consideremos o exemplo:

$$\text{MAXIMIZAR} \quad X_0 = 11x_1 + 7x_2 + 5x_3 + x_4$$
$$6x_1 - 4x_2 + 3x_3 + x_4 \leq 25$$
$$x_1, x_2, x_3, x_4 \geq 0 \quad \text{e inteiros}$$

Como $a_4 = 1$ o problema não se altera se tornarmos a restrição numa igualdade. A solução ótima do problema linear correspondente é:

$$x_1 = 25/6 \quad ; \quad x_0 = 275/6 \quad ; \quad x_j = 0 \quad \text{para} \quad j=2,3,4$$

Não sendo esta solução inteira e sendo $b=25$ suficientemente grande, como vimos no capítulo II parágrafo 1.4, passemos a transformá-lo em um problema de mochila grupo. Para tal o escreveremos na forma intermediária fornecida por (5):

$$\text{MINIMIZAR} \quad 2x_2 + 3x_3 + 5x_4$$
$$4(\text{mod } 6)x_2 + 3(\text{mod } 6)x_3 + 1(\text{mod } 6)x_4 \equiv 1(\text{mod } 6)$$
$$x_2, x_3, x_4 \geq 0 \quad \text{e inteiro}$$

O problema de mochila grupo correspondente é fornecido por (10):

$$\text{MINIMIZAR} \quad d_4t_4 + d_3t_3 + d_1t_1$$
$$v_4t_4 + v_3t_3 + v_1t_1 = v_1$$
$$t_4, t_3, t_1 \geq 0 \quad \text{e inteiros}$$

Precisamos determinar ainda d_4, d_3, d_1 , o que nos é fornecido por (9a), e obter a correspondência entre os x_j's e os t_i's.

$T_4 = \{j \mid a_j (\mathrm{mod}\ a_1) = 4 \ , \ j=2,3,4\} = \{2\}$

Como $t_4 = \sum_{j \varepsilon T_4} x_j \rightarrow t_4 = x_2$, t_4 é correspondente a x_2 .

Como $d_4 = \min_{j \varepsilon T_4} (c_1 a_j - c_j a_1) \rightarrow d_4 = 2$

Continuando os cálculos acima para $p=3$ e $p=1$, obtemos:

$$
\begin{aligned}
\text{MINIMIZAR}\quad & 2t_4 + 3t_3 + 5t_1 \\
& v_4 t_4 + v_3 t_3 + v_1 t_1 = v_1 \\
& t_4, t_3, t_1 \geq 0 \ \text{ e inteiros}
\end{aligned}
$$

com t_4 correspondente a x_2 , t_3 a x_3 e t_1 a x_4 .

Por simples inspeção, uma solução ótima para este problema é:

$t_3 = t_4 = 0$ e $t_1 = 1$ com custo 5

Logo:

$x_2 = x_3 = 0$ e $x_4 = 1$

Por (1a) e (1b), temos:

$$x_1 = \frac{25}{6} - \frac{1}{6} \rightarrow x_1 = 4$$

$$X_0 = \frac{275}{6} - \frac{5}{6} = 45$$

<u>O B S E R V A Ç Ã O</u>: 1) Outra solução ótima é $t_3 = t_4 = 1$ e $t_1 = 0$, o que acarreta $x_2 = x_3 = 1$ e $x_4 = 0$, com $x_1 = \frac{25}{6} - \frac{4}{6} - \frac{3}{6} = 3$ e $X_0 = \frac{275}{6} - \frac{2}{6} - \frac{3}{6} = 45$.

2) Verificamos que o problema foi resolvido de forma mais simples utilizando a transformação para mochila grupo. Entretanto isto somente foi possível por ser b suficientemente grande, o que garante a viabilidade da solução, ou seja, x_1 é CERTAMENTE POSITIVO.

CAPÍTULO V

TRANSFORMAÇÃO DE UM PROBLEMA DE PROGRAMAÇÃO INTEIRA EM UM PROBLEMA MOCHILA GRUPO

1. Idéia do Processo

Seja um problema de programação inteira. Relaxando-se a integridade das variáveis obtem-se um problema linear cuja solução ótima é conseguida por intermédio do MÉTODO SIMPLEX. Diante desta solução ótima (x_B, x_N) a restrição do problema inicial é reescrita como:

$$B\ x_B + N\ x_N = b \quad \text{ou} \quad B\ x_B = b - N\ x_N$$

Esta equação é transformada na equivalente:

$\hat{B}\ y = R(b - Nx_N)$, sendo $\hat{B}$ matriz diagonal, o que é suportado por dois teoremas enunciados e demonstrados neste capítulo, que além disso oferecem subsídios para análise e formulação de um algoritmo que diagonaliza B , calculando como veremos as matrizes $\hat{B}$ e R .

Como y deve ser inteiro, isto ocorre se e somente se:

$R(b - Nx_N) \equiv 0 \pmod{\lambda}$, λ vetor cujas coordenadas são a diagonal principal de $\hat{B}$.

$$\hat{B} = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_m \end{bmatrix}$$

Estas congruências módulo λ_i permitem transformar o então problema em um MOCHILA sobre o GRUPO $G(\lambda_1, \lambda_2, \ldots, \lambda_m)$.

2. Suporte Teórico

TEOREMA 1:

Seja A uma matriz unimodular de ordem m. Para cada vetor inteiro y, existe um único vetor inteiro x tal que $y = Ax$.

Demonstração:

Sendo A matriz unimodular então, A é inteira, existe A^{-1} e $\det A = 1$.

Tendo que:

$$A^{-1} = \frac{\text{adj } A}{\det A} \rightarrow A^{-1} = \text{adj } A$$

Como a adjunta de uma matriz inteira é uma matriz inteira concluímos que A^{-1} é inteira.

Se $y = Ax$ então $A^{-1}y = x$. Logo dado y inteiro x é inteiro e único, pois resulta de um produto entre duas matrizes inteiras.

TEOREMA 2:

Sejam R e C matrizes unimodulares de ordem m e seja $RBC = \hat{B}$. Então $Bx = b$, x inteiro, e $\hat{B}y = Rb$, y inteiro, são representações equivalentes.

Demonstração:

Multiplicando-se $Bx = b$ à esquerda por R, vem:

$$RBx = Rb \qquad (I)$$

Visto que C é unimodular, C^{-1} existe e por tando:

$$RBC = \hat{B} \rightarrow RB = \hat{B}\ C^{-1} \tag{II}$$

Substituindo (II) em (I), temos:

$$\hat{B}\ C^{-1}\ x = R\ b \tag{III}$$

Consideremos agora:

$$y = C^{-1}\ x \tag{IV}$$

Sendo C uma matriz unimodular então C^{-1} é unimodular. Por isso, a partir do TEOREMA 1, concluímos que existe uma correspondência um a um entre os valores inteiros de x e y em (IV).

Substituindo (IV) em (III), vem:

$$\hat{B}y = R\ b \tag{V}$$

Um vetor inteiro y^* resolve (V) se e somente se $x^* = C\ y^*$ é tal que $R\ B\ x^* = Rb$. Como R é não singular então existe R^{-1} e por conseguinte $RBx^* = Rb \rightarrow$ $\rightarrow Bx^* = b$. Logo y^* resolve (V) se e somente se $x^* = Cy^*$ é tal que $Bx^* = b$.

3. Transformação I

Suponhamos um problema de programação inteira. Seja a solução ótima, relaxando a integridade, (x_B, x_N) correspondente à base ótima B. A partir daí podemos escrever o conjunto das restrições do problema inteiro da seguinte forma:

$$Bx_B + Nx_N = b$$
$$x_B \geq 0 \quad e \quad \text{inteiro}$$
$$x_N \geq 0 \quad e \quad \text{inteiro}$$

ou

$$Bx_B = b - Nx_N$$
$$x_B \geq 0 \quad e \quad \text{inteiro} \qquad \text{(VI)}$$
$$x_N \geq 0 \quad e \quad \text{inteiro}$$

Sejam R e C matrizes inversíveis tais que:

$B = R^{-1} \hat{B} C^{-1}$ onde $\hat{B}$ é matriz diagonal.

Deste modo a equação $Bx_B = b - Nx_N$ pode ser escrita como:

$$R^{-1} \hat{B} C^{-1} x_B = b - Nx_N \qquad \text{(VII)}$$

Multiplicando (VII) à esquerda por R , temos:

$$\hat{B} C^{-1} x_B = R(b - Nx_N)$$

O conjunto de restrições (VI) pode então ser escrito:

$$\hat{B} C^{-1} x_B = R(b - Nx_N$$
$$x_B \geq 0 \quad e \quad \text{inteiro} \qquad \text{(VII)}$$
$$x_N \geq 0 \quad e \quad \text{inteiro}$$

Observando que $\hat{B} C^{-1} = R B$ (basta multiplicar $B = R^{-1} \hat{B} C^{-1}$ à esquerda por R), então:

$$\hat{B}C^{-1}x_B = R(b-Nx_N) \rightarrow RBx_B = R(b-Nx_N) \rightarrow Bx_B = b - Nx_N ,$$

o que garante a não alteração do conjunto solução de (VI).

Seja C^{-1} uma matriz unimodular. Logo, pelo TEOREMA 1, dado y inteiro existe um único x inteiro tal que $y = c^{-1}x$. Portanto (VII) pode ser escrito:

$$\begin{aligned} &\hat{B}\, y = R(b - Nx_N) \\ &y \text{ inteiro} \\ &x_N \geq 0 \quad \text{e} \quad \text{inteiro} \end{aligned} \qquad \text{(VIII)}$$

Exemplo:

Consideremos o seguinte problema:

$$\begin{array}{lrl} \text{MAXIMIZAR} & x_0 = & 2x_1 + x_2 \\ \text{sujeito a:} & & x_1 + x_2 + x_3 = 5 \\ & & -x_1 + x_2 + x_4 = 0 \\ & & 6x_1 + 2x_2 + x_5 = 21 \\ & & x_i \geq 0 \ \text{ e inteiro}, \quad i=1,2,3,4,5 \end{array}$$

Apliquemos a esse exemplo a TRANSFORMAÇÃO acima apresentada.

O relaxamento da integridade das variáveis implica em um problema linear cuja solução ótima é:

$$x_B^t = (x_1, x_2, x_4) = (11/4, 9/4, 1/2)$$

$$x_N^t = (x_3, x_5) = (0,0)$$

correspondente ao quadro ótimo:

		$-x_5$	$-x_3$
x_0	31/4	1/4	1/2
x_1	11/4	1/4	-1/2
x_2	9/4	-1/4	3/2
x_3	0	0	-1
x_4	1/2	1/2	-2
x_5	0	-1	0

Sendo x_1 , x_2 e x_4 as variáveis básicas da solução ótima, a base ótima B correspondente é:

$$B = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \\ 6 & 2 & 0 \end{bmatrix}$$

Escrevendo as restrições do problema inteiro como em (VI), vem:

$$\begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \\ 6 & 2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \\ 21 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_5 \end{bmatrix}$$

$x_B \geq 0$ e inteiro

$x_N \geq 0$ e inteiro

Para escrevermos estas restrições como em (VIII) necessitamos de $\hat{B}$, matriz diagonal obtida a partir de B , e de R , também obtida a partir de B . Um algoritmo para obtenção dessas matrizes será desenvolvido a frente. Adiantaremos, para atingirmos nosso objetivo, que:

$$\hat{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} \quad \text{e} \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix}$$

Com esses elementos podemos escrever as restrições em questão como em VIII . Temos:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ 0 \\ 21 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_5 \end{bmatrix}$$

y inteiro

$x_N \geq 0$ e inteiro

Isto é, nosso conjunto de restrições torna-se:

$$\begin{aligned} y_1 &= 5 - x_3 \\ y_2 &= 5 - x_3 \\ 4y_3 &= -9 + 6x_3 - x_5 \\ &y_1, y_2, y_3 \quad \text{inteiros} \\ &x_3, x_4 \geq 0 \quad \text{e} \quad \text{inteiros} \end{aligned}$$

4. <u>Transformação II</u>

A partir das restrições escritas como em (VIII), onde:

$$\hat{B} = \begin{bmatrix} \lambda_1 & & 0 \\ & & \\ 0 & & \lambda_m \end{bmatrix}$$

verificamos que y é inteiro se e somente se:

$$R(b - Nx_N) \equiv 0 \pmod{\lambda}$$

Aplicando ao nosso exemplo, onde $\lambda_1 = \lambda_2 = 1$ e $\lambda_3 = 4$, temos y_1, y_2 e y_3 inteiros se e somente se:

$$5 - x_3 \equiv 0 \pmod{1} \qquad \text{(a)}$$

$$-9 + 6x_3 - x_5 \equiv 0 \pmod{4} \qquad \text{(b)}$$

Observando as demais restrições do problema, concluímos que (a) é redundante, pois exige x_3 inteiro, o que já faz parte do conjunto de restrições do problema.

A partir da observação acima verificamos que a restrição CONGRUÊNCIA MÓDULO λ_i pode ser abandonada se $\lambda_i = 1$.

Desta forma as restrições do nosso exemplo reduzem-se a:

$$-9 + 6x_3 - x_5 \equiv 0 \pmod 4$$
$$x_3, x_5 \geq 0 \quad \text{e inteiros}$$

ou

$$6x_3 - x_5 \equiv 9 \pmod 4$$
$$x_3, x_5 \geq 0 \quad \text{e inteiros}$$

ou

$$2x_3 + 3x_5 \equiv 1 \pmod 4$$
$$x_3, x_5 \geq 0 \quad \text{e inteiros}$$

Observando o QUADRO ÓTIMO relativo à solução linear temos a seguinte função objetivo: $1/2x_3 + 1/4x_5$. Multiplicando esta função por 4 , temos $2x_3 + x_5$. Como os pares (x_3, x_5) que minimizam ambas as funções são os mesmos, podemos escrever nosso problema inteiro, como:

$$\begin{array}{ll} \text{MINIMIZAR} & 2x_3 + x_5 \\ \text{sujeito a:} & 2x_3 + 3x_5 \equiv 1 \pmod 4 \\ & \quad x_3, x_5 \geq 0 \quad \text{e inteiros} \end{array}$$

que pode ser visto como um problema do tipo mochila sobre o grupo G(4).

O problema acima pode então ser escrito:

$$\text{MINIMIZAR} \quad d_{a_3(\text{mod } 4)}\, t_{a_3(\text{mod } 4)} + d_{a_5(\text{mod } 4)}\, t_{a_5(\text{mod } 4)}$$

$$\text{s.a.:} \quad v_{a_3(\text{mod } 4)}\, t_{a_3(\text{mod } 4)} + v_{a_5(\text{mod } 4)}\, t_{a_5(\text{mod } 4)} = v_{b(\text{mod } 4)}$$

$$t_{a_j(\text{mod } 4)} \geq 0 \text{ e inteiro }, \; j=3,5$$

$$\text{MINIMIZAR} \quad d_2 t_2 + d_3 t_3$$

$$\text{s.a.:} \quad v_2 t_2 + v_3 t_3 = v_1$$

$$t_2, t_3 \geq 0 \quad \text{e inteiros}$$

5. Procedimento para obtenção da matriz $\hat{B}$

Como R e C são unimodulares, portanto inteiras, concluímos ser $\hat{B}$ também inteira pois $\hat{B}$ = RBC , onde B também é inteira, já que resulta de um problema inteiro. Desta forma as matrizes obtidas em cada passo da diagonalização de B devem ser inteiras. Desta feita as operações elementares linha utilizados na obtenção de $\hat{B}$ devem ser :

(i) TROCA DE LINHA (COLUNA).

(ii) SUBSTITUIR UMA LINHA (COLUNA) i POR ELA SOMADA COM UM MÚLTIPLO INTEIRO DE OUTRA

(iii) MULTIPLICAR UMA LINHA (COLUNA) POR -1 .

Estas operações também se justificam pelo fato de estarmos trabalhando dentro do contexto de um Z-módulo, pois G é um módulo sobre Z . (TODO GRUPO ABELIANO G É UM MÓDULO SOBRE O ANEL DOS INTEIROS).

O cuidado na formulação dessas propriedades nos garante que λ_i é inteiro para todo i, o que nos possibilita considerar, sem embaraços, as congruências módulo λ_i.

<u>Obtenção de $\hat{B}$ a partir de B</u>

$$\text{Seja} \quad B = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Trocando linhas e ou colunas posicionamos o menor elemento em valor absoluto e diferente de zero em (1,1). (isto pela conveniência de se obter λ_i menor possível, o que implica em um grupo com menos elementos). Supondo já realizada essa operação, fazemos:

$c = ma + n$, $0 \leq n < a$, n inteiro

$b = m'a + n'$, $0 \leq n' < a$, n' inteiro

(<u>1º</u>) Suponhamos $n = n' = 0$. Desta forma $c = ma$ e $b = m'a$, isto é:

$$B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{array}{c} \begin{bmatrix} a & m'a \\ ma & d \end{bmatrix} \end{array} \begin{array}{l} L_1 \\ L_2 \end{array}$$

Aplicando a operação linha (ii) em B, vem:

$$B^* = \begin{bmatrix} a & m'a \\ 0 & d-mm'a \end{bmatrix} \begin{array}{l} L_1 \\ L_2 = L_2 - mL_1 \end{array}$$

Sujeitando B^* à operação coluna (ii), vem:

$$\hat{B} = \begin{bmatrix} a & 0 \\ 0 & d-mm'a \end{bmatrix}$$
$$\begin{array}{cc} \downarrow & \downarrow \\ c_1 & c_2 = c_2 - m'c_1 \end{array}$$

(2º) Suponhamos, agora, $n \neq 0$. Desta forma, temos:

$$B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a & b \\ ma+n & d \end{bmatrix} \begin{array}{l} L_1 \\ L_2 \end{array}$$

Aplicando a operação linha (ii) de forma a substituir L_2 por $L_2 - m\,L_1$, obtemos:

$$\begin{bmatrix} a & b \\ n & d-mb \end{bmatrix} \begin{array}{l} L_1 \\ L_2 = L_2 - mL_1 \end{array}$$

Após esta operação temos a certeza de que existe pelo menos um elemento (n) menor em valor absoluto do que "a", pois ele é resto da divisão por "a" . É possível até, o que é mais conveniente ainda, que "d-mb" seja menor do que "n" em valor absoluto.

Transpondo desta feita linhas e ou colunas posicionamos este menor elemento em valor absoluto em (1,1). Supondo seja "n" este elemento, temos, traspondo L_1 e L_2 :

$$\begin{bmatrix} n & d-mb \\ a & b \end{bmatrix} \begin{array}{l} L_1 = L_2 \\ L_2 = L_1 \end{array}$$

A partir daí, recomeçamos o processo, verificando se toda linha e toda coluna de n é múltipla inteira de n . Caso seja, recaímos no 1º caso. Caso contrário repetimos o procedimento deste 2º caso.

Existe também a possibilidade de "a" ser múltiplo inteiro de "n" e "d-mb" não o ser. Neste caso a operação linha (ii) aplicada no início deste 2º caso é substituída por uma operação coluna (ii), e o processo de recolocação do menor elemento em valor absoluto na posição (1,1) é iniciado.

CONCLUSÃO: A PARTIR DO MOMENTO EM QUE O MENOR ELEMENTO EM VALOR ABSOLUTO E DIFERENTE DE ZERO ESTÁ POSICIONADO EM (1,1), A PROBABILIDADE DOS DEMAIS ELEMENTOS DE SUA LINHA E COLUNA SEREM SEUS MÚLTIPLOS INTEIROS É MUITO PEQUENA. SENDO ASSIM O PROCEDIMENTO DESENVOLVIDO NO 2º CASO TENDE A SER REPETITIVO. ISSO PODE CONDUZIR A RESTOS "n" CADA VEZ MENORES (CASO NÃO SE CONSIGA OS TAIS MÚLTIPLOS ENFOCADOS PELO 1º CASO) O QUE NA PIOR DAS HIPÓTESES, TRABALHANDO COM RESTOS CADA VEZ MENORES, CHEGA-SE A UM RESTO 1, O QUE GARANTE A CONVERGÊNCIA DO PROCESSO.

6. Algoritmo para obtenção da matriz $\hat{B}$

Sendo $\hat{B}$ = RBC , podemos considerar inicialmente o produto IBI e, por intermédio de operações linha e coluna aplicados a B, obter R e C , extendendo as operações linha à primeira matriz identidade e as operações coluna à segunda matriz identidade. Isto porque toda operação linha aplicada a uma matriz A é equivalente a aplicar esta operação linha na matriz identidade e, em seguida, multiplicar esta nova matriz por A . No caso da operação coluna, a equivalência é obtida pela aplicação desta operação na matriz identidade e, em seguida, efetuar a multiplicação de A por essa nova matriz conseguida.

Este algoritmo, de acordo com o estudo feito no item anterior, usará operações linha e coluna sobre a matriz B e as que lhe sucedem. Poderemos submeter as matrizes identidade a estas operações, como vimos no último parágrafo, caso queiramos obter R e C .

ALGORITMO

PASSO 1: Faça $k=1$

Se b_{11} é o menor elemento em valor absoluto da matriz e diferente de zero vá para o PASSO 2.

Senão, usando linhas e/ou colunas, coloque b_{11} igual ao menor elemento em valor absoluto da matriz.

PASSO 2: Se b_{11} divide b_{1t} , $1 < t \leq n$, vá para o PASSO 3 .

Senão, para cada b_{1t} não divisível por b_{11} faça $b_{1t} = k_t b_{11} + r_t$, r_t resto da divisão de b_{1t} por b_{11}. Realize então a operação $c_t - k_t c_1$, onde c_1 e c_t são respectivamente as colunas 1 e t da matriz. Volte ao PASSO 1.

COMENTÁRIO: As operações $c_t - k_t c_1$, se necessárias, tornam os novos b_{1t} iguais aos restos r_t .

PASSO 3: Se b_{11} divide b_{t1} , $1 < t \leq n$, vá para o PASSO 4.

Senão, para cada b_{t1} não divisível por b_{11} faça $b_{t1} = k_t b_{11} + r_t$, r_t resto da divisão de b_{t1} por b_{11}. Realize então a operação $L_t - K_t L_1$, onde L_1 e L_t são respectivamente as linhas 1 e t da matriz. Volte ao PASSO 1.

C O M E N T Á R I O: As operações $L_t - K_t L_1$, se necessárias, tornam os novos b_{t1} iguais aos restos r_t

PASSO 4: Faça : $c_t = c_t - b_{1t} c_1$, $1 < t \leq n$

$$L_t = L_t - b_{11} L_1 \quad , \quad 1 < t \leq n$$

C O M E N T Á R I O: Este passo transforma a matriz B em:

$$B = \begin{bmatrix} b_{11} & 0 & 0 & \cdots & 0 \\ 0 & b_{22} & b_{23} & \cdots & b_{2n} \\ 0 & b_{32} & b_{33} & \cdots & b_{3n} \\ - & - & - & - & - \\ 0 & b_{n2} & b_{n3} & \cdots & b_{nn} \end{bmatrix}$$

PASSO 5: Se $b_{11} < 0$ multiplique a linha 1 por -1 .

Faça: $d_{kk} = b_{11}$

$n = n - 1$

$B_{nxn} = B_{(n-1)x(n-1)}$

$k = k + 1$

C O M E N T Á R I O: Este passo ordena o armazenamento dos elementos da diagonal e reduz a matriz, abandonando a primeira linha e a primeira coluna.

PASSO 6: Se B_{nxn} tem ordem 1 e $b_{11} > 0$ faça $d_{kk} = b_{11}$ e vá para o PASSO 7.

Se B_{nxn} tem ordem 1 e $b_{11} < 0$, multiplique b_{11} por -1, faça $d_{kk} = b_{11}$ e vá para o PASSO 7.

Se B_{nxn} tem ordem diferente de 1, volte ao PASSO 1.

C O M E N T Á R I O: Este passo ordena que o algoritmo se repita até que a matriz se reduza ao único elemento d_{nn}.

PASSO 7: Faça:

$$\hat{B} = \begin{bmatrix} d_{11} & & \underset{\sim}{0} \\ & d_{22} & \\ \underset{\sim}{0} & & \ddots \; d_{nn} \end{bmatrix} \quad \text{e} \quad \text{PARE} .$$

Uso do ALGORITMO para obter R e $\hat{B}$

Seja a matriz B do exemplo considerado anteriormente, qual seja:

$$B = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \\ 6 & 2 & 0 \end{bmatrix}$$

Necessitamos, para chegar a (VIII) da transformação I, das matrizes $\hat{B}$ e R . Vimos que:

$$\begin{bmatrix} \; I \; \end{bmatrix}\begin{bmatrix} \; B \; \end{bmatrix}\begin{bmatrix} \; C \; \end{bmatrix} \rightarrow \begin{bmatrix} \; R \; \end{bmatrix}\begin{bmatrix} \; \hat{B} \; \end{bmatrix}\begin{bmatrix} \; C \; \end{bmatrix}$$

Como precisamos obter apenas R e $\hat{B}$, consideramos somente [I] [B] e estendemos a I e suas derivadas apenas as operações linha aplicados a B e suas derivadas.

$$\begin{bmatrix} \; I \; \end{bmatrix}\begin{bmatrix} \; B \; \end{bmatrix} \rightarrow \begin{bmatrix} \; R \; \end{bmatrix}\begin{bmatrix} \; \hat{B} \; \end{bmatrix}$$

<u>Obtenção de R e $\hat{B}$</u>

(1.1) $k=1$; $n = 3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \\ 6 & 2 & 0 \end{bmatrix}$$

Como b_{11} é o menor elemento em valor absoluto diferente de ZERO, vá para o PASSO 2

(2.2) b_{11} divide b_{1t} , $t = 2,3$. Vá para o PASSO 3 .

(3.1) b_{11} divide b_{t1} , $t = 2,3$. Vá para o PASSO 4 .

(4.1) $c_2 = c_2 - 1 \cdot c_1 \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ -1 & 2 & 1 \\ 6 & -4 & 0 \end{bmatrix}$

$$\begin{matrix} L_2 = L_2 - (-1)L_1 \\ \\ L_2 = L_3 - 6\ L_1 \end{matrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & -4 & 0 \end{bmatrix}$$

Vá para o PASSO 5

(5.1) $d_{11} = 1$

$n = 2$

$$B = \begin{bmatrix} 2 & 1 \\ -4 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ -4 & 0 \end{bmatrix}$$

$k = 2$. Vá para o PASSO 6

(6.1) Como B tem ordem 2 , volte ao PASSO 1

(1.2) Fazendo $c_2 = c_1 \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & -4 \end{bmatrix}$

Vá para o PASSO 2.

(2.2) Como b_{11} divide b_{12} vá para o PASSO 3

(3.2) Como b_{11} divide b_{21} vá para o PASSO 4

(4.2) Fazendo $c_2 = c_2 - 2c_1 \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -4 \end{bmatrix}$

Vá para o PASSO 5

(5.2) $d_{22} = 1$

$n = 1$

$$B = \begin{bmatrix} -4 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix} \begin{bmatrix} -4 \end{bmatrix}$$

$k = 3$. Vá para o PASSO 6

(6.2) Como $b_{11} < 0$ multiplique-o por -1. Logo $b_{11} = 4$

$d_{33} = 4$

Vá para o PASSO 7

(7.1) $$\hat{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} \quad ; \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -6 & 0 & 1 \end{bmatrix}$$

7. Caracterização do Processo

Dado o problema de programação inteira:

$$\begin{aligned} \text{MAXIMIZAR} \quad & CX \\ & AX = b \\ & X \geq 0 \text{ e inteiro} \end{aligned}$$

obter a solução ótima de:

$$\begin{aligned} \text{MAXIMIZAR} \quad & CX \\ & AX = b \\ & X \geq 0 \end{aligned}$$

Seja (x_B, x_N) a solução ótima procurada. O problema inteiro fica então:

$$\begin{aligned} \text{MAXIMIZAR} \quad & C_B X_B + C_N X_N \\ & BX_B + NX_N = b \\ & X_B, X_N \geq 0 \quad \text{e inteiros} \end{aligned}$$

A equação $BX_B + NX_N = b \rightarrow X_B = B^{-1}(b - NX_N)$. Substituindo na função objetivo, temos:

$$C_B B^{-1}(b-NX_N) + C_N X_N = C_B B^{-1}b - (C_B B^{-1}N - C_N)X_N$$

Mas:

$$\text{MAX} \quad C_B B^{-1}b - (C_B B^{-1}N-C_N)X_N = \text{MIN}(C_B B^{-1}N - C_N)X_N$$

O problema inteiro pode então ser escrito:

$$\text{MINIMIZAR} \quad (C_B B^{-1}N - C_N)X_N$$
$$BX_B + NX_N = b$$
$$X_B, X_N \geq 0 \text{ e inteiros}$$

Pela transformação I:

$$BX_B + NX_N = b \quad \rightarrow \quad \hat{B}y = R(b - NX_N)$$

Então passamos a ter:

$$\text{MINIMIZAR} \quad (C_B B^{-1}N - C_N)X_N$$
$$\hat{B}y = R(b - NX_N)$$
$$y \text{ inteiro}$$
$$X_N \geq 0 \quad \text{e inteiro}$$

Como: $\hat{B} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_m \end{bmatrix}$ o problema assume a forma:

$$\text{MINIMIZAR} \quad (C_B B^{-1}N - C_N)X_N$$
$$R(b - XN_N) \equiv 0(\text{mod } \lambda)$$
$$X_N \geq 0 \text{ e inteiro}$$

sendo $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$

ou ainda:

$$\text{MINIMIZAR} \quad (C_B B^{-1} N - C_N) X_N$$

$$R_j (b - N X_N) \equiv 0 \pmod{\lambda_j} \quad , \; j=1,\ldots m$$

$$X_N \geq 0 \quad \text{e inteiro}$$

sendo R_j a j-ésima linha de R.

<u>OBSERVAÇÃO</u>: Sempre que $\lambda_j = 1$ a restrição pode ser abandonada pois é redundante. Este fato ocorreu no exemplo apresentado.

Finalmente as restrições do problema dão origem a uma única equação sobre o grupo $G(\lambda_1, \lambda_2, \ldots, \lambda_m)$. Caso, por intermédio das transformações, atingíssemos as restrições:

$$2x_3 + x_4 \equiv 0 \pmod 2$$

$$5x_3 + 7x_4 \equiv 15 \pmod{17}$$

a equação grupo seria:

$$v_{0,5} \, t_1 * v_{1,7} \, t_2 = v_{0,15} \quad \text{sobre o grupo } G(2.17)$$

CAPÍTULO VI

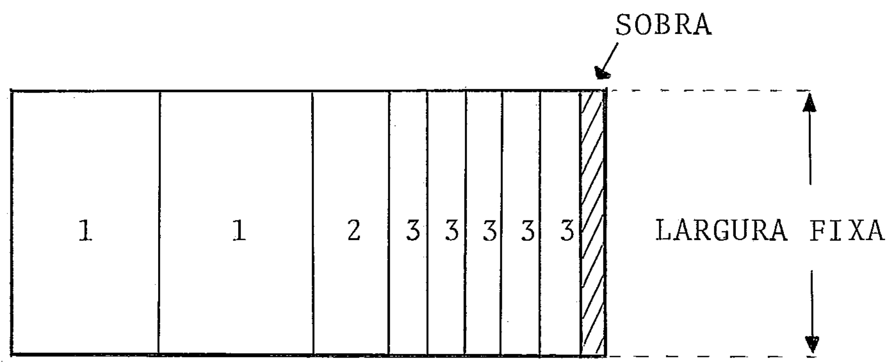
UMA APLICAÇÃO: "THE CUTTING STOCK PROBLEM"

1. Em que consiste o problema

Existem numerosas situações que consistem em uma determinada quantidade de material, que se encontra em tamanhos padrões, ter que ser cortada de acordo com certas exigências: que se perca o mínimo possível de material e seja satisfeito o número de cada pedaço de comprimento ℓ_i demandado.

Trataremos do caso que envolve cortes verticais. O material em estoque não difere em largura. Existem comprimentos $\ell_i (i=1,\ldots,m)$, para atender a demanda de mercado.

Um tipo de modelo de corte para um comprimento L é visto abaixo:



Para este modelo de corte j podemos definir o seguinte <u>VETOR CORTE</u> :

$$\underline{a_j} = \begin{bmatrix} a_{ij} \\ a_{2j} \\ a_{3j} \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 5 \end{bmatrix}$$

onde a_{ij} significa o número de pedaços de comprimento ℓ_i obtidos no modelo de corte j .

Já podemos concluir que existem "n" possibilidades de corte que cresce intensamente à proporção que o número "m" de comprimentos ℓ_i cresce. Este crescimento é ainda mais intenso se tivermos mais de um comprimento padrão.

2. Formulação do Problema

Sejam:

- $N_i (i=1,\ldots,m)$ o número de pedaços de comprimento ℓ_i demandados.

- $x_j (j=1,\ldots,n)$ o número de vezes que o j-ésimo modelo de corte é usado.

- c_j o custo do material L que sofreu o j-ésimo corte.

  Obs.: os c_j podem ser todos iguais desde que haja apenas um tamanho padrão L.

- a_{ij} o número de pedaços de comprimento ℓ_i produzidos cada vez que o j-ésimo modelo de corte é usado.

O problema que se propõe determinar o número de vezes que cada modelo de corte deve ser usado, tal que a demanda do número de pedaços de cada comprimento ℓ_i seja satisfeita e o custo total do material seja mínimo (o que implica em minimizar a PERDA DE MATERIAL) é:

$$\text{MINIMIZAR} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_{ij} x_j \geq N_i \qquad (i=1,\ldots m)$$

$$x_j \geq 0 \text{ e inteiro } (j=1,\ldots,n)$$

A expressão $\sum_{j=1}^{n} a_{ij}x_j$ representa a quantidade de pedaços ℓ_i determinada por todos os modelos de corte.

Existem tantas restrições quanto forem os pedaços de comprimento ℓ_i . Ou seja existem "m" restrições.

Nosso problema é inteiro como "m" restrições.

3. Dificuldades Apresentadas

Há duas dificuldades:

(1ª) O número de modelos de corte é excessivamente grande. Este fato raramente permite que o programa seja resolvido "diretamente". Como exemplo suponhamos placas de 200 cm de comprimento e demanda para 40 diferentes comprimentos desde 20 cm a 80 cm. Neste caso o número de modelos de corte pode exceder 100 milhões. Isto nos poderia oferecer uma função objetivo com 100 milhões de parcelas para ser minimizada, o que é impraticável.

(2ª) O valor de x_j inteiro causa muita dificuldade no processo de solução. O arredondamento dos valores fracionários não causa grandes danos ao programa. Especialmente no caso de problemas com demanda alta, os valores ótimos fracionários de x_j são bastante grandes, tal que perde-se pouco ao arredondá-los para inteiros. Neste caso a dificuldade apresentada tem solução simples.

Como vimos acima a segunda dificuldade não nos causa embaraços. Encontra-se contorno imediato para o problema surgido, ficando nosso programa na forma linear, qual seja:

$$\begin{array}{lll} \text{MINIMIZAR} & \sum_{j=1}^{n} c_j x_j & \\ \text{sujeito a} & \sum_{j=1}^{n} a_{ij} x_j \geq N_i & (i=1,\ldots,m) \\ & x_j \geq 0 & (j=1,\ldots,n) \end{array}$$

Quanto à primeira dificuldade, o número excessivo de modelos de corte, iremos contorná-a resolvendo o problema linear acima com as colunas a_j "geradas" à proporção que se tornam necessárias. Isto será visto com mais detalhes adiante. Desta feita não precisaremos definir previamente todos os modelos de corte.

Importante atentarmos para o fato de que estamos trabalhando no espaço $\mathbb{R}^m$, ou seja, nossos vetores a_j possuem "m" elementos e definem um modelo de corte:

$$\underline{a_j} = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{bmatrix}$$

Ao iniciarmos o problema conheceremos "m" , que se refere aos tipos de tamanhos ℓ_i . Não teremos, entretanto o valor de "n" , pois não definiremos os "a_j" previamente. O processo de solução a ser aqui desenvolvido não usará todos os modelos de corte possíveis.

4. Estudo para obtenção da solução a ser dada

Nosso problema é:

MINIMIZAR $\sum_{j=1}^{n} c_j x_j$

sujeito a: $\sum_{j=1}^{n} a_{ij} x_j \geq N_i$

$x_j \geq 0$

Suponhamos conhecida uma solução viável básica $\underline{x_B}$ para a equação:

$$\sum_{j=1}^{n} a_{ij} x_j - s_i = N_i \qquad (i=1,\ldots,m)$$

onde s_i é uma variável de folga não negativa.

Usando o SIMPLEX REVISADO, o quadro inicial tem a forma:

BASE	1	$\underline{\pi}$	z	
A_B^t	$\underline{0}$	B^{-1}	$\underline{x_B}$	

$\underline{\pi}\,\underline{a_j} - c_j = z_j - c_j$

$\underline{Y} = B^{-1}.a_j$

Lembremos que: $\underline{\pi} = \underline{c_B^t} \,.\, B^{-1}$

$$z = \underline{c_B^t} \,.\, \underline{x_B}$$

$$x_B = B^{-1} \,.\, \underline{b}$$

Para obtermos os valores para $\underline{\pi}\,\underline{a_j} - c_j$ e

$B^{-1}\,\underline{a_j}$ precisamos determinar o $\underline{a_j}$ que entra na base. Neste caso o critério de entrada na base (OTIMALIDADE) é:

$$z_k - c_k = \text{MÁXIMO}\{\underline{\pi}\ \underline{a_j} - c_j \text{ tal que } \underline{\pi}\ \underline{a_j} - c_j > 0,\ \underline{a_j} \in X\}$$

Se todos $\underline{\pi}\ \underline{a_j} - c_j \leq 0 \rightarrow$ ÓTIMO . PARE

Caso contrário entra na base o $\underline{a_j}$ tal que $j=k$.

(As colunas $\underline{a_j}$ que determinam um modelo de corte têm uma estrutura bem definida e pertencem a um determinado conjunto X)

SURGE AQUI O PROBLEMA CAUSADO PELA GRANDE QUANTIDADE DE MODELOS DE CORTE: É IMPRATICÁVEL A DEFINIÇÃO DE TODOS OS ELEMENTOS DO CONJUNTO X, O QUE COMPROMETE A MAXIMIZAÇÃO DOS $\underline{\pi}\ \underline{a_j} - c_j$.

<u>SOLUÇÃO ENCONTRADA</u>: <u>GERAÇÃO DE COLUNAS ATRAVÉS SOLUÇÃO DE PROBLEMAS MOCHILA</u>

Essa técnica consiste em maximizar $\underline{\pi}\ \underline{a_j} - \underline{c_j}$, sem a necessidade de se dispor do conjunto X . Vamos a ela.

O conjunto X compreende as colunas $\underline{a_j}$ com estrutura bem definida, ou seja, tais que:

$$\sum_{i=1}^{m} \ell_i\, a_{ij} \leq L$$

$$a_{ij} \geq 0 \text{ e inteiro}$$

L $\rightarrow$ comprimento da placa

ℓ_i $\rightarrow$ comprimento do pedaço i

a_{ij} → número de pedaços de comprimento ℓ_i do modelo j .

Como desejamos obter $\underline{a}_j$ que maximiza $\underline{\pi}\ \underline{a}_j - c_j$ é suficiente resolver o problema mochila:

$$\text{MAXIMIZAR} \quad \sum_{i=1}^{m} \pi_i\, a_{ij}$$

$$\text{sujeito a:} \quad \sum_{i=1}^{m} \ell_i a_{ij} \leq L$$

$$a_{ij} \geq 0 \text{ e inteiro}$$

Este problema mochila nos fornece a coluna $\underline{a}_j$ que entra na base e, portanto, nos permite completar o QUADRO SIMPLEX com os valores $(\underline{\pi}\ \underline{a}_j - c_j)$ e $B^{-1}\ \underline{a}_j$, caso não seja atingido o ÓTIMO.

Nosso "CRITÉRIO DE ENTRADA NA BASE" passa a ser o seguinte:

(1º) resolve-se o problema mochila criado a partir do quadro simplex

(2) Se $\underline{\pi}\ \underline{a}_j - c_j \leq 0$ → QUADRO ÓTIMO. PARE .

Se $\underline{\pi}\ \underline{a}_j - c_j > 0$ → $\underline{a}_j$ entrará na base

OBSERVAÇÕES: (1) Considerando π_i o custo, a cada iteração,do pedacinho ℓ_i , fica perfeitamente justificado o critério de paradas. O custo total $\underline{\pi}\ \underline{a}_j$ vem se aproximando do custo c_j da placa. Enquanto se desenvolve esse processo de aproximação, que é caracterizado por $\underline{\pi}\ \underline{a}_j > c_j$, vão se alterando os $\underline{a}_j$ da base. No momento em que $\underline{\pi}\ \underline{a}_j \leq c_j$ atingimos o QUADRO ÓTIMO. Caso $\underline{\pi}\ \underline{a}_j = c_j$ não há perda de material; o aproveitamento da chapa é total. O vetor $\pi = (\pi_1, \ldots, \pi_m)$ é a solução dual correspondente ao programa linear.

(2) No caso de haver mais de um tamanho padrão de chapa, teremos também mais de um problema mochila para resolver. Neste caso o QUADRO ÓTIMO só é atingido quando $\underline{\pi}\ \underline{a}_j - c_j \leq c_j$ para cada problema mochila. Do contrário a coluna $\underline{a}_j$ que fornecer o maior valor positivo para $\underline{\pi}\ \underline{a}_j - c_j$ entra na base. Isto está em perfeito acordo com o critério de entrada do SIMPLEX. Aliás não poderia ser de outra forma.

Seja, então, $\underline{a}_k$ a próxima coluna a entrar na base. Nosso QUADRO SIMPLEX toma a forma:

	BASE	1	$\underline{\pi}$	z	$\underline{\pi} \cdot \underline{a}_k - c_k$
$\underline{a}_k \rightarrow$ $\underline{a}_s \leftarrow$	A_B^t	$\underline{0}$	B^{-1}	$\underline{x}_B$	$\underline{Y} = B^{-1} . \underline{a}_k$

QUEM SAI DA BASE PARA DAR ENTRADA A $\underline{a}_k$?

Obtem-se o índice s , tal que a_s sai da base, a partir do critério:

$$\frac{x_{Bs}}{Y_s} = \underset{i}{\text{MÍNIMO}} \left\{ \frac{x_{Bi}}{Y_i} \text{ tal que } Y_i > 0 \right\} \rightarrow j = s$$

Obtido o $\underline{a}_s$ que sai da base dando entrada para $\underline{a}_k$, iniciamos o pivoteamento em torno de Y_s . Terminado os cálculos repete-se o procedimento de saída e entrada na base, até a obtenção do QUADRO ÓTIMO.

Todo esse estudo se baseou numa solução inicial $\underline{x_B}$, suposta conhecida. Nossa preocupação agora está voltada para a obtenção desse $\underline{x_B}$. Para obter essa solução básica inicial introduziremos variáveis artificiais x_{ai} não negativas e usaremos o MÉTODO DAS DUAS FASES.

A partir da formulação do problema, podemos escrever:

$$
\begin{array}{lcl}
z - c_1x_1 - \dots - c_nx_n & & = 0 \\
 & z_a + x_{a1} + \dots + x_{am} & = 0 \\
a_{11}x_1 + \dots + a_{1n}x_n - s_1 & + x_{a1} & = b_1 \\
\vdots \qquad \vdots \qquad \ddots & \ddots & \vdots \\
a_{m1}x_1 + \dots + a_{mn}x_n \qquad - s_m & + x_{am} & = b_m
\end{array}
$$

<u>FASE I</u>:

MINIMIZAR $\quad Z_a = \sum_{i=1}^{m} x_{a_i}$

sujeito a: $\quad \sum_{j=1}^{n} a_{ij}x_j - s_i + x_{ai} = N_i$

$s_i \geq 0 \ , \ x_{ai} \geq 0 \quad (i=1,\dots,m)$

$x_j \geq 0 \quad (j=1,\dots,n)$

QUADRO INICIAL:

	BASE	1	0	$\underline{\pi}$	z	$\triangleleft\!\!-\; \underline{\pi}\ \underline{a_k} - c_k = z_k - c_k$
		0	1	$\underline{\pi_I}$	z_a	$\triangleleft\!\!-\; \underline{\pi_I}\ \underline{a_k} = (z_k - c_k)a$
$\underline{a_s}$ $\underline{a_k}$	A_B^t	$\underline{0}$	$\underline{0}$	B^{-1}	$\underline{x_B}$	$\triangleleft\!\!-\; \underline{y} = B^{-1}.\underline{a_k}$

$$\underline{\pi_I} = \underline{C_I^t} \cdot B^{-1}$$

$$Z_a = \underline{C_I^t} \cdot \underline{x_B}$$

Para determinarmos esta última coluna, ou seja, o "k" relativo a quem entra na base precisamos conhecer também todos os modelos de corte $\underline{a_j}$, pois:

$$(z_k - c_k)_a = \text{MÁXIMO}\{\underline{\pi_I}\ \underline{a_j} \text{ tal que } \underline{\pi_I}\ \underline{a_j} > 0 \ , \ \underline{a_j} \in X\}$$

Utilizamos também aqui a técnica de geração de colunas, resolvendo o problema mochila:

$$\text{MAXIMIZAR} \quad \sum_{i=1}^{n} \pi_{I_i}\ a_{ij}$$

$$\text{sujeito a:} \quad \sum_{i=1}^{m} \ell_i\ a_{ij} \leq L$$

$$a_{ij} \geq 0 \text{ e inteiro}$$

Se $\underline{\pi_I}\ \underline{a_j} \leq 0 \rightarrow$ FIM DA FASE I
TESTE PARA VER SE HÁ SOLUÇÃO

Se $\underline{\pi_I}\ \underline{a_j} > 0 \rightarrow \underline{a_j}$ entra na base

<u>OBS.</u>: Como a FASE 1 termina quando $(z_j - c_j) \leq 0$ para os vetores não básicos fica justificado o critério usado.

<u>TESTE</u>

Três casos podem ocorrer:

(1) $Za > 0$

(2) $Za = 0$ e nenhum vetor artificial permenece na base

(3) $Za = 0$ e um ou mais vetores artificiais permanecem na base com $x_{ai} = 0$

Ocorrendo (1) o <u>PROBLEMA ORIGINAL NÃO TEM SOLUÇÃO</u>. Se ocorrer (2) ou (3) então <u>ATINGIMOS SOLUÇÃO INICIAL BÁSICA</u> .

<u>OBS.</u>: Como adicionamos a restrição $Z_a + \sum x_{ai} = 0$, não corremos o risco de x_{ai} , que pode permanecer na base a nível zero no final da FASE 1, vir assumir valores positivos nas iterações seguintes.

No critério de saída da base não há nada de novo a acrescentar.

FASE II:

Nossa atenção volta-se para a função objetivo Z. O critério de entrada na base é o usual, ou seja, a função objetivo do problema mochila volta a ser:

$$\sum_{i=1}^{m} \pi_i . a_{ij}$$

5. Técnica Aproximada para Solução dos Problemas Mochila - Algoritmo Guloso

Pela análise da solução do problema em questão, feita na secção anterior, concluímos que a eficiência do algoritmo que o soluciona depende crucialmente da rapidez da solução dos problemas mochila. Usaremos uma técnica, que embora forneça uma resposta aproximada, tem como características principais a rapidez e a simplicidade de solução.

O algoritmo aproximado que usaremos é denominado guloso no sentido de Edmonds |18|.

ALGORITMO GULOSO

Seja o problema mochila:

$$\text{MAXIMIZAR} \quad Z = \sum_{j=1}^{n} c_j x_j$$

$$\text{sujeito a:} \quad \sum_{j=1}^{n} a_j x_j \leq b$$

$$x_j \geq 0 \text{ e inteiro }, \ j=1,\ldots,n$$

onde as variáveis estão ordenadas de forma que:

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \dots \geq \frac{c_n}{a_n}$$

Algoritmo:

PASSO 1: $j=1$

PASSO 2: $x_j = [b/a_j]$

PASSO 3: Faça $b = b - a_j x_j$

PASSO 4: Se $j=n$, PARE

Senão faça $j = j+1$ e vá para o PASSO 2

OBS.: O algoritmo procura dar o valor máximo possível primeiramente para x_1 (que está associado a c_1/a_1), após, e com o que sobra na mochila, o mesmo é feito com x_2 (que está associado a $c_2/a_2 < c_1/a_1$) e assim em diante.

APLICAÇÃO

MAXIMIZAR $10x_1 + 16x_2 + x_3$

s.a.: $2x_1 + 3x_2 + 2x_3 \leq 4$

$x_1, x_2, x_3 \geq 0$ e inteiro

(1.1) $j=1$

(1.2) $x_1 = [4/2] \rightarrow x_1 = 2$

(1.3) $b = 4 - 2.2 \rightarrow b = 0$

(1.4) $j = j + 1 \rightarrow j = 2$

(2.2) $x_2 = [0/3] \rightarrow x_2 = 0$

(2.3) $b = 0 - 3.0 \rightarrow b = 0$

(2.4) $j = 2 + 1 \rightarrow j = 3$

(3.2) $x_3 = [0/2] \rightarrow x_3 = 0$

(3.3) $b = 0 - 2.0 \rightarrow b = 0$ $\qquad x_1 = 2$

(3.4) $j = 3 \rightarrow$ PARE $\qquad x_2 = x_3 = 0$

Nem sempre, entretanto, a solução viável obtida é ÓTIMA. Precisamos então ter uma idéia quanto à aproximação obtida.

<u>Relação entre a solução ótima e a solução do Guloso</u>

Sejam:

Z^* → valor da função objetivo da solução ótima

$\overline{Z}$ → valor da função objetivo da solução do Guloso

Sabemos que:

$Z^* \leq c_1 \cdot \frac{b}{a_1}$, pois $x_1 = \frac{b}{a_1}$ e $x_j = 0$, $j=2,\ldots,n$ é

a solução do programa linear correspondente pelo relaxamento da integridade das variáveis:

Temos que:

$$\overline{Z} \geq c_1 \left[\frac{b}{a_1}\right]$$

Podemos também escrever:

$$\frac{\overline{Z}}{Z^*} \geq \frac{c_1 \left[\frac{b}{a_1}\right]}{c_1 \frac{b}{a_1}} \rightarrow \frac{\overline{Z}}{Z^*} \geq \frac{[b/a_1]}{b/a_1}$$

Suporemos $b/a_1 \geq 1$, caso contrário faríamos $x_1 = 0$ e passaríamos para um problema de menor tamanho. Se todos os $b/a_j < 1$ então a única solução viável é $x_j = 0$, $j = 1,\ldots,n$.

Sabemos que se x é tal que $x \geq 1$ então:

$$\frac{x}{[x]} > \frac{1}{2}$$

Logo:

$$\frac{[b/a_1]}{b/a_1} > \frac{1}{2} \quad \text{e então:}$$

$$\frac{\overline{Z}}{Z^*} > \frac{1}{2} \quad \rightarrow \quad \overline{Z} > \frac{1}{2} Z^*$$

Finalmente:

$$\frac{1}{2} Z^* < \overline{Z} \leq Z^*$$

<u>OBSERVAÇÃO</u>: A solução $\overline{Z}$ do Guloso pode servir de limite para um método de enumeração do tipo "branch and bound".

REFERÊNCIAS BIBLIOGRÁFICAS

|1| Anthonisse, J.; "A Note on Reducing a System to a Single Equation", Stichting Mathematisch Centrum preliminary report, BN 1/70 Amsterdam, December 1970.

|2| Bellman, R.; "Notes on the Theory of Dynamic Programming IV - Maximization over Discrete Sets", Naval Research Logistics Quarterly 3, 67-70 (1956).

|3| Bellman, R.; "Dynamic Programming, Princeton University Press, 1957.

|4| Bellman, R., and S. Dreyfus; "Applied Dynamic Programming, Princeton University Press, 1962.

|5| Bradley, G.; "Equivalent Integer Programs and Canonical Problems", Management Science 17(5), 354-366 (1971).

|6| Bradley, G.; "Transformation of Integer Programs to Knapsack Problems", Discrete Mathematics 1, 29-45 (1971).

|7| Brooks, R., and A. Geoffrion; "Finding Everett's Lagrange Multipliers by Linear Programming", Operations Research 14(16), 1149-1153 (1966).

|8| Elmaghraby, S., and M. Wig ; "On the Treatment of Stock Cutting Problems as Diophantine Programs", Operations Research Center Report Nº 61, North Carolina State University at Raleigh, May 11, 1970.

|9| Everett III, H. ; "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources", Operations Research 11(3), 399-471 (1963).

|10| Gass, S.I., "Linear Programming: Methods and Applications", McGraw-Hill, New York, 1958.

|11| Gilmore, P., and R. Gomory ; "A Linear Programming Approach to the Cutting Stock Problem I", Operations Research 9(6), 849-858 (1961).

|12| Gilmore, P., and R. Gomory ; "The Theory and Computation of Knapsack Functions", Operations Research 14(6), 1045-1074 (1966).

|13| Gilmore, P., and R. Gomory ; "New Results for Reducing Linear Programs to Knapsack Problems, 'Management Science Report Series 72-7, The University of Colorado at Boulder, April 1972.

|14| Glover, F., and R. Woolsey ; "Aggregating Diophantine Equations", Management Science Report Series 70-4, The University of Colorado at Boulder, October 1970.

|15| Hadley, G.F., "Linear Programming", Addison-Wesley, Reading Massachusetts, 1975.

|16| Hernstein, I.N., "Tópicos de Álgebra", Editora Polígono S.A., São Paulo, 1970.

|17| Hoffman K., and Kunze R. ; "Álgebra Linear", Editora Polígono, São Paulo, 1971.

|18| J. Edmonds, "Matroids and the Greedy Algorithm", Mathematical Programming 1, 1971.

|19| Kendell, K., and S. Zionts ; "Solving Integer Programming Problems by Aggregating Constraints", School of Manegement Working Paper Nº 155, S.U.N.Y. at Buffalo, November 1972.

|20| Lasdon, L. ; "Optimization Theory for Large Systems", New York, Macmillan, 1970.

|21| Lorie, J., and L. Savage, "Three Problems in Capital Rationing", Journal of Business 28, 229-239 (1955).

|22| Mathews, G., "On the Partition of Numbers", Proceedings of the London Mathematical Society 28, 486-490 (1897).

|23| Nelson Maculan Filho, "Programação Linear Inteira", PDD 17/78, Comissão de Publicações, COPPE, Universidade Federal do Rio de Janeiro.

|24| Nelson Maculan Filho e Maria Luiza Villares, "Comentários sobre um Algoritmo para a Solução Aproximada do Problema da Mochila", 1º Simpósio de Combinatória - São José do Rio Preto - 17 e 18 de Agosto de 1979.

|25| Padberg. M. ; "Equivalent Knapsack Type Formulations of Bounded Integer Linear Programs". Management Science Research Report Nº 227, Carnegie-Mellon University, September 1970.

|26| Salkin H.M., "Integer Programming , Addison-Wesley Publishing Company, 1975.

|27| Simonnard, M.A., Programmation Linéaire : Technique du Calcul Économique", Dunond, Paris, 1973, 1º volume.

|28| Weingartner, H.; Mathematical Programming and the Analysis of Capital Budgeting Problems, Englewood Cliffs, N.Y., Prentice Hall, 1963.

|29| White, W. ; "On a Group Theoretic Approach to Linear Integer Programming", Operations Research Center Report 66-27, The University of California at Berkeley, 1966.