ESTUDOS SOBRE PROBLEMAS DE PROGRAMAÇÃO

VETORIAL LINEAR COM VARIÁVEIS ZERO-UM

LUZIA KAZUKO YOSHIDA

DISSERTAÇÃO APRESENTADA AO

INSTITUTO DE MATEMÁTICA E ESTATÍSTICA

DA

UNIVERSIDADE DE SÃO PAULO

PARA OBTENÇÃO DO GRAU DE MESTRE

EM

MATEMÁTICA APLICADA

ORIENTADOR:

Prof. Dr. MARCO ANTONIO BRINATI

São Paulo, novembro 1978

Ao

Masao

Agradecimentos

Ao Prof. Dr. Marco Antonio Brinati,

pela valiosa orientação, pelo interesse e apoio que foram muito significativos na realização deste trabalho.

Ao Prof. Dr. Gabriel R. Bitran,

que acompanhou com grande dedicação o desenrolar deste trabalho, pela colaboração e incentivo constante.

Ao Prof. Dr. Carlos Humes Jr.,

pelas críticas construtivas e sugestões que melhoraram a apresentação deste trabalho.

Aos amigos e funcionários do MAP,

pelo apoio durante a elaboração deste trabalho e em particular, ao Prof. Jorge Stolfi, pelas contribuições na programação dos algoritmos.

Às Srtas. Regina Helena, Luzia e Sueli pelo excelente trabalho de datilografia.

São Paulo, novembro de 1978

L.K.V.

Í N D I C E

pág.

RESUMO

CAPÍTULO I	- INTRODUÇÃO	1
CAPÍTULO II	- RESULTADOS TEÓRICOS	13
	1. Introdução	13
	2. Pontos eficientes e os mapeamentos M e N	14
	3. Os conjuntos $M(v)$, $N(v)$ e $\bar{V}$	33
	4. Os conjuntos $\mathcal{M}_0(v)$ e $\mathcal{N}(v)$	37
CAPÍTULO III	- DESCRIÇÃO DO ALGORITMO PARA O PROBLEMA (P)	54
	1. Introdução.	54
	2. O algoritmo	56
	3. Os conjuntos $\bar{V}$ e $\bar{V}_1$	64
	4. O conjunto D	74
	5. O conjunto $EF(P)$	80
	5.1 Dominância entre elementos de dois conjuntos	81
	5.2 Dominância entre elementos do próprio conjunto.	82
	5.3 Antecessores no grafo dirigido $G(DUC, \bar{V}_1)$	85
	5.4 Viabilidade dos elementos de DUC	85

CAPÍTULO IV - APLICAÇÃO DO ALGORITMO PARA DIFEREN	
TES MATRIZES DE CRITÉRIOS C	86
1. Introdução	86
2. O algoritmo simplificado para a matriz de critérios C não negativa.	89
3. Problemas com a matriz de critérios $C \geq 0$ com o conjunto F fixo.	101
4. Matriz de critérios C com uma linha não positiva	103
5. Matriz de critérios C com uma determinada porcentagem de elementos negativos.	105
6. Experiências computacionais variando os parâmetros p e m	119
7. Análise dos resultados computacionais	120
 CAPÍTULO V - CONJUNTO DE PONTOS FORTEMENTE EFICI	
ENTES E DE PONTOS FRACAMENTE EFICI	
ENTES.	123
1. Introdução.	123
2. Identificação de pontos fortemente eficientes.	125
3. Algoritmo para resolver $(PSW)^1$	135
 CAPÍTULO VI - PROBLEMAS DE PROGRAMAÇÃO VETORIAL	
COM OS ELEMENTOS DA MATRIZ DE CRITÉ	
RIOS NUM CERTO INTERVALO.	139

	pág.
1. Introdução	139
2. Resultados Teóricos	140
3. Resultados Computacionais sobre $EF(\nabla PF)$ e $EF(PF)$	146
CONCLUSÕES	150
APÊNDICE 1	152
APÊNDICE 2	163
BIBLIOGRAFIA	168

RESUMO

Neste trabalho, são apresentados resultados teóricos para os problemas de programação vetorial linear com variáveis zero-um, a partir dos quais, desenvolve-se um algoritmo para determinar o conjunto de todos os pontos eficientes para estes problemas. O algoritmo é aplicado para diferentes matrizes de critérios, cada qual apresentando uma característica particular. São apresentados extensos resultados computacionais para estes problemas.

Além disso, é apresentado um algoritmo para identificar pontos fortemente eficientes e fracamente eficientes e também para obter soluções eficientes para uma extensão do problema principal. São apresentados resultados computacionais para estes problemas.

CAPÍTULO I

INTRODUÇÃO

Os problemas de programação matemática referem-se, em geral, à distribuição eficiente de recursos limitados entre atividades competitivas, com a finalidade de atender a um determinado objetivo, por exemplo, maximização de lucros ou minimização de custos, que é expresso por uma função chamada função-objetivo.

Mas, não é raro, encontrarmos problemas em que desejamos atender vários objetivos, simultaneamente. Seja, por exemplo, o seguinte problema: "dadas duas configurações de economia A e B , onde o bem-estar social de um indivíduo, o indivíduo I , é "melhor" em A do que em B e o bem-estar social de um outro indivíduo, o indivíduo II , é "melhor" em B do que em A . Entre as configurações A e B , qual é a melhor escolha com o objetivo de atender os indivíduos I e II , simultaneamente?"

Podemos notar que não existe uma escolha que possamos chamar de ótima, pois, se decidirmos pela configuração A

estaremos prejudicando o indivíduo *II* e se decidirmos pela configuração *B* estaremos prejudicando o indivíduo *I*.

Uma configuração deste tipo, na qual se fizermos uma mudança viável não é possível "melhorar" alguém sem "piorar" um outro, chama-se Pareto* ótima, ou seja, é uma configuração ótima segundo critério de Pareto. Portanto, no problema acima, tanto a configuração *A* como a configuração *B* é Pareto ótima.

Em termos de programação matemática, podemos dizer que existem problemas em que desejamos otimizar, simultaneamente, várias funções-objetivo conflitantes. Estes problemas são tratados como problemas de programação matemática vetorial, pelo fato de podermos exibir as funções-objetivo como uma função vetorial.

As configurações que são Pareto ótimas serão chamadas de *pontos eficientes*.

Os problemas de programação matemática vetorial contínua são amplamente tratados na literatura e existem várias interpretações para este tipo de problemas: as *p* funções-objetivo são substituídas por uma única função que é

* Vilfredo Pareto, 1848-1923, economista e sociólogo italiano.

não decrescente em cada argumento, ou define-se uma ordem de preferência entre as funções-objetivo, ou as soluções do problema são os pontos eficientes. Para determinar pontos eficientes, em geral, supõe-se que cada função-objetivo seja côncava e o conjunto de pontos viáveis convexo. B. Roy, em /13/ (1971), fez uma síntese sobre as principais abordagens dadas a este tipo de problemas.

A.M. Geoffrion, J.S. Dyer e A. Feinberg, em /5/, apresentaram um procedimento com interação (o decisor é solicitado para responder algumas perguntas), para resolver problemas de programação vetorial contínua, aplicando o algoritmo de Frank-Wolfe. A principal dificuldade é que eles associam uma função de utilidade que não é conhecida explicitamente e que se supõe ser diferenciável e côncava, sobre um conjunto compacto e convexo. Esta é a razão pela qual o decisor é envolvido na execução do algoritmo respondendo perguntas do tipo: "está disposto a "sacrificar" α unidades na função-objetivo 1 para "ganhar" β unidades na função-objetivo i , $i \neq 1$?"

S. Zionts e J. Wallenius, em /19/, apresentaram também um procedimento com interação para resolver problemas de programação vetorial contínua, cujas funções-objetivo são funções côncavas sobre um conjunto convexo. Inicialmente, escolhe-se um conjunto de pesos λ_i para as funções-objetivo ,

de tal forma que $\sum_{i=1}^n \lambda_i = 1$, $\lambda_i \geq \epsilon$, $i=1, \dots, n$ (n = dimensão do espaço onde a função-objetivo está definida e ϵ = número positivo suficientemente pequeno) e resolve-se um problema de programação linear determinando-se uma solução eficiente. A partir da análise das variáveis não básicas o decisor é solicitado a responder perguntas do mesmo tipo que no procedimento anterior. Obtidas as respostas, novos pesos são calculados e o procedimento é repetido até que nenhuma variável não-básica, quando introduzida na base, leva a um ponto extremo adjacente eficiente.

R. Steuer, em /15/, analisou o problema de programação vetorial contínua, quando os pesos associados a cada função-objetivo variam num determinado intervalo. O objetivo deste estudo é tentar diminuir o número de pontos extremos eficientes que, em geral, é muito grande.

J. Shapiro, em /14/, cita vários exemplos de problemas de programação vetorial que não satisfazem a hipótese de que a função utilidade é côncava sujeita a restrições convexas. Existe uma teoria completa para caracterizar o conjunto de pontos eficientes e tem-se, também, desenvolvido algoritmos "bem comportados" para determinar este conjunto sob a hipótese acima. Porém, pouco se tem feito quando essa hipótese de convexidade não é válida, como no caso de programação vetorial inteira. Shapiro discutiu este tipo de

problemas, mostrando como os resultados da teoria da dualidade podem ser aplicados para gerar soluções eficientes. Porém, como em alguns casos, é necessário um número excessivo de cálculos, ele sugere que o método dual seja associado ao método de "branch and bound".

B. Villarreal e M. H. Karwan, em /16/, mostraram as extensões das equações recursivas de programação dinâmica em processos de decisões sequenciais com múltiplos estágios, para problemas de programação vetorial. Eles desenvolveram, também, um algoritmo para problemas de programação vetorial inteira baseado nas equações recursivas de programação dinâmica.

H. Pasternak e U. Passy, em /10/, apresentaram um algoritmo para resolver um problema particular de programação vetorial com variáveis zero-um em que a função-objetivo está definida de R^n em R^2 . Este algoritmo foi utilizado para seleção de projetos e foi desenvolvido baseando-se na resolução de dois problemas paramétricos:

- 1) $\max[\alpha f(x) + (1-\alpha) g(x)]$, tal que $x \in S$ e
- 2) $\max f(x)$, tal que $x \in S^0$,

onde $f(\cdot)$ e $g(\cdot)$ são as funções-objetivo, lineares que devem ser maximizadas (no sentido vetorial); S é o conjunto viável

vel do problema dado e $S^0 \equiv \{x \in S : g(x) \geq \theta\}$. O método consiste em resolver várias vezes o problema 1) fixando-se valores para α e utilizando-se uma variante do método de filtro de Balas /1/ e a seguir, resolver o problema 2) variando θ num determinado intervalo de tal forma que todos os pontos eficientes possam ser gerados.

S. Zionts, em /18/, apresentou um algoritmo com interação para determinar os pontos eficientes entre as alternativas discretas apresentadas. O algoritmo é semelhante ao apresentado por S. Zionts e J. Wallenius em /19/ para o caso contínuo, só que neste caso nenhum problema de programação matemática escalar é utilizado. É um algoritmo com interação, porque o decisor é solicitado a indicar a preferência entre os pontos apresentados no decorrer do algoritmo.

G. R. Bitran, em /2/, propôs uma base teórica para problemas de programação vetorial linear com variáveis zero-um. A partir dos resultados teóricos, Bitran desenvolveu um algoritmo para determinar o conjunto de todos os pontos eficientes para este tipo de problema. O algoritmo consiste em determinar, inicialmente, as direções de crescimento das funções-objetivo, no sentido vetorial, e a partir destas direções determinar todos os pontos que são eficientes. As soluções do problema são obtidas em duas etapas:

- 1) resolve-se um problema auxiliar com as mesmas

funções-objetivo, tendo como conjunto de pontos viáveis os vértices do hipercubo unitário;

2) a partir das soluções obtidas na etapa 1), determina-se o conjunto de pontos eficientes para o problema original, isto é, considerando o conjunto de pontos viáveis do problema dado. Além de apresentar vários resultados computacionais, o trabalho é rico no seu aspecto teórico, pois contém, ainda, o tratamento dos seguintes tópicos: análise paramétrica das restrições, aplicação dessa teoria na resolução de problemas de otimização de funções monotônicas com variáveis zero-um, relação entre o problema de programação vetorial contínua e discreta e uma breve discussão sobre o problema de programação vetorial mista.

O objetivo deste trabalho é estudar os problemas de programação vetorial linear com variáveis zero-um, tendo por base os resultados obtidos por Bitran em /2/ e desenvolver um algoritmo mais eficiente para determinar o conjunto de pontos eficientes para estes problemas.

No capítulo II, apresentaremos as definições básicas utilizadas em programação vetorial e os resultados teóricos visando o desenvolvimento de um algoritmo para determinar o conjunto de pontos eficientes.

No capítulo III, descreveremos um algoritmo para determinar todos os pontos eficientes num problema de pro-

gramação vetorial linear com variáveis zero-um.

No capítulo IV, o algoritmo apresentado no capítulo III será aplicado para configurações particulares da matriz dos coeficientes das funções-objetivo. Três são os casos considerados:

- 1) os coeficientes para todas as funções-objetivo são não negativos;
- 2) todos os coeficientes são não negativos, com exceção de uma função-objetivo que tem todos os coeficientes não positivos e
- 3) os coeficientes podem ser positivos, negativos ou nulos.

Serão apresentados vários resultados computacionais para cada um destes problemas. Foram resolvidos cerca de 200 problemas com programas escritos em ALGOL do computador Burroughs, B-6700.

No capítulo V, classificaremos um conjunto discreto e finito de pontos de R^p em: conjunto de pontos fortemente eficientes, pontos fracamente eficientes e pontos dominados. O interesse desta análise é impor uma restrição adicional com o objetivo de reduzir o número de soluções de um problema de programação vetorial linear com variáveis zero-um. Apresentaremos um algoritmo para efetuar essa classifi

cação, o qual tem como instrumentos básicos os métodos simplex e dual simplex de programação linear. Este algoritmo - foi programado pela Irina C. Orlow (aluna de pós-graduação da EPUSP), em ALGOL do B-6700. Foram resolvidos 180 problemas e será apresentado o número médio de elementos em cada um dos conjuntos.

No capítulo VI, analisaremos o problema em que cada coeficiente das funções-objetivo é dado por um certo intervalo de valores. Aplicaremos os resultados da análise deste problema para obter algumas soluções para o problema de programação vetorial côncava com variáveis zero-um. Apresentaremos os resultados computacionais de 45 problemas, obtidos com um programa em ALGOL do B-6700, escrito por Nicolau Sesso (aluno de graduação da EPUSP).

No apêndice 1, daremos alguns detalhes de programação dos algoritmos apresentados no capítulo IV em ALGOL do B-6700.

No apêndice 2, daremos algumas definições básicas da teoria dos grafos, sobre árvores e análise convexa que serão utilizadas no decorrer deste trabalho.

As seções dentro de cada capítulo são numeradas a partir de 1. Dentro de cada seção as definições, proposições, corolários, exemplos e tabelas terão números sequenciais -

próprias a partir de 1, porém antecedidos dos respectivos números da seção, com exceção dos corolários que são antecedidos pelo número da proposição da qual é consequência.

Ao referirmo-nos a um resultado e não mencionarmos o número do capítulo, fica subentendido que se trata de um resultado do capítulo em questão.

Em todas as experiências computacionais, os coeficientes das funções-objetivo e das restrições foram gerados aleatoriamente num certo intervalo especificado para cada caso. Os coeficientes das restrições serão todos não negativos.

A expressão "se e somente se" será abreviada por "sse".

Os conjuntos serão denotados por letras maiúsculas e os vetores por letras minúsculas. Para distinguir vetores, utilizaremos índices superiores e para distinguir componentes de um vetor, utilizaremos índices inferiores.

Em geral, os vetores serão vetores colunas, mas o sinal T de transposto não será utilizado quando o produto - escalar entre dois vetores está claro.

Os vetores nulos serão indicados por 0 quando o significado estiver claro. Em caso contrário será utilizado $\underline{0} = (0, 0, \dots, 0)^T$.

As seguintes convenções serão adotadas para as re-

lações de ordem. Se $(x, y) \in R^n \times R^n$, então:

$$x=y \quad \text{sse} \quad x_i=y_i, \quad \forall i=1, \dots, n$$

$$x \geq y \quad \text{sse} \quad x_i \geq y_i, \quad \forall i=1, \dots, n$$

$$x \geq y \quad \text{sse} \quad x_i \geq y_i, \quad \forall i=1, \dots, n \quad x \neq y$$

$$x > y \quad \text{sse} \quad x_i > y_i, \quad \forall i=1, \dots, n.$$

Note-se que, se $(x, y) \in R \times R$, então a relação $x \geq y$ é equivalente à $x > y$.

As seguintes notações serão utilizadas no decorrer deste trabalho.

$$J \equiv \{1, \dots, n\}$$

$$[X] = \text{envoltória convexa de } X$$

$$DUC \equiv \{x \in R^n : x_j \in \{0, 1\}, \quad j \in J\} \quad (\text{vértices do hipercubo unitário})$$

$$UC \equiv \{x \in R^n : 0 \leq x_j \leq 1, \quad j \in J\} \quad (UC \equiv [DUC])$$

$$\prod_{i \in J} X_i \equiv X_1 \times X_2 \times \dots \times X_n, \text{ produto cartesiano entre os conjuntos } X_i, \quad i=1, \dots, n.$$

$$X \setminus Y \equiv \{x \in X \text{ e } x \notin Y\} \equiv X - Y$$

$$e = (1, 1, \dots, 1), \quad e \in R^n$$

$$e^i = (0, \dots, 0, 1, 0, \dots, 0), \quad e^i \in R^n, \text{ isto é :}$$

$$e_j^i = 0, \quad \forall j \in J \text{ e } j \neq i \quad \text{e} \quad e_i^i = 1.$$

$$\underline{1} = (1, \dots, 1)^T$$

$$\lfloor z \rfloor = \text{maior inteiro } \leq z$$

$$\lceil z \rceil = \text{menor inteiro } \geq z$$

$$\text{card}(X) = \text{cardinalidade do conjunto } X.$$

$$\wedge = "e"$$

CAPÍTULO II

RESULTADOS TEÓRICOS

1. INTRODUÇÃO

Neste capítulo apresentaremos os resultados teóricos gerais obtidos para problemas de programação vetorial linear com variáveis zero-um a partir da teoria desenvolvida em /2/.

Na seção 2, daremos a formulação do problema de programação vetorial com variáveis zero-um e definiremos o conceito de *solução* para este tipo de problema. Baseados neste conceito, definiremos um mapeamento M tal que qualquer ponto que pertence à imagem deste mapeamento não pertence ao conjunto *solução*. De forma análoga, definiremos um mapeamento N tal que, se a intersecção do mapeamento N com o mapeamento M for vazia, então todo elemento que pertence à imagem do mapeamento N pertence ao conjunto *solução*. Basicamente, esta seção é dedicada ao estudo das relações que existem entre esses mapeamentos e das condições que devem ser estabelecidas para obtermos mapeamentos convenientes de uma forma computacionalmente eficiente.

Na seção 3, daremos uma caracterização desses mapeamentos M e N utilizando problemas de programação linear. Daremos, também, uma caracterização parcial de um conjunto de vetores obtido a partir das condições estabelecidas na seção anterior para relacionar os mapeamentos M e N .

Na seção 4, definiremos dois mapeamentos $\mathcal{M}$ e $\mathcal{N}$ baseados, respectivamente, nos mapeamentos M e N com valores no cubo unitário. Mostraremos que estes mapeamentos tem característica de face do cubo unitário e mostraremos, também, as relações que existem entre os elementos do cubo unitário e os que pertencem ao mapeamento $\mathcal{M}$ ou ao mapeamento $\mathcal{N}$.

2. PONTOS EFICIENTES E OS MAPEAMENTOS M E N .

Um problema de programação matemática vetorial pode ser definida como uma extensão natural de problemas de programação matemática escalar:

dados:

$$f: R^n \rightarrow R^p, \text{ uma função } p\text{-dimensional}$$

$$g: R^n \rightarrow R^m, \text{ uma função } m\text{-dimensional}$$

$$b \in R^m, J = \{1, 2, \dots, n\}$$

e denominando o conjunto

$$F \equiv \{x \in R^n: g(x) \leq b\}$$

de conjunto de *pontos viáveis*,
 encontre, se existir, $x^0 \in F$, tal que $f(x^0) \geq f(x), \forall x \in F$;
 ou ainda:
 encontre, se existir, $x^0 \in F: \nexists x \in F$, tal que, $f(x) \geq f(x^0)$.

Note-se que quando $f(.)$ é uma função real as duas definições são equivalentes, pois, a relação $\geq$ é equivalente à relação $>$.

No caso em que a $f(.)$ é uma função vetorial, as duas definições podem não ser equivalentes como no seguinte problema:

dados:

$$f: R^n \rightarrow R^n, \text{ tal que } f(x) = x$$

$$F \equiv \{e^i, i \in J\}$$

Pela primeira definição: $\nexists x^0 \in F: f(x^0) \geq f(x), \forall x \in F$, isto é, o conjunto solução deste problema é vazio.

Pela segunda definição: $\forall x \in F$, x é solução (solução múltipla), isto é, o próprio conjunto F é conjunto solução.

Consideremos os seguintes conjuntos:

$$X_1 \equiv \{x^0 \in F: \forall x \in F, f(x^0) \geq f(x)\}$$

$$X_2 \equiv \{x^0 \in F: \nexists x \in F: f(x) \geq f(x^0)\}$$

Fato 1. Se $X_1 \neq \emptyset$ então $X_2 \equiv X_1$, pois:

$$\begin{aligned} i) \quad x^0 \in X_1 &\Rightarrow \forall x \in F, f(x^0) \geq f(x) \\ &\Rightarrow \nexists x \in F: f(x) > f(x^0) \Rightarrow x^0 \in X_2 \end{aligned}$$

Portanto, $X_1 \subset X_2$.

$$ii) \quad x^0 \in X_2 \Rightarrow \nexists x \in F: f(x) \geq f(x^0) \Rightarrow \forall x \in F, f(x) \not\geq f(x^0).$$

$$\forall y \in X_1 \Rightarrow f(y) \geq f(z), \quad \forall z \in X_2 \subseteq F.$$

$$\text{Então, } f(y) \geq f(x^0) \text{ e } f(y) \not\geq f(x^0) \Rightarrow f(y) = f(x^0) \Rightarrow x^0 \in X_1.$$

Portanto, $X_2 \subset X_1$.

Fato 2. Se $X_1 \equiv \emptyset$ então $X_2 \equiv X_1 \Leftrightarrow F \equiv \emptyset$, pois a relação $\geq$ é uma relação de ordem parcial em R^D .

Portanto, o conceito de *solução ótima* de programação matemática não é adequado para problemas de programação vetorial, sendo necessário estendê-lo para o que denominamos de *solução eficiente*.

Expressaremos o nosso problema da seguinte forma: .
encontre, se existir, $x^0 \in F: f(x^0) = \max\{f(x): x \in F\}$.

DEFINIÇÃO 2.1 - Dado um problema de programação ve torial $\max\{f(x): x \in F\}$, dizemos que um ponto $x^0 \in F$ é um *ponto eficiente* sse não existir $x \in F$ tal que $f(x) \geq f(x^0)$.

DEFINIÇÃO 2.2 - Dado um problema de programação ve torial $\max\{f(x): x \in F\}$, dizemos que $x \in F$ *domina* $x^0 \in F$ ou que $x^0 \in F$ é *dominado* por $x \in F$ sse $f(x) \geq f(x^0)$.

Nós estudaremos apenas os problemas de programação

matemática vetorial *linear* com *variáveis zero-um*, isto é , onde $f(.)$ e $g(.)$ são funções lineares e as variáveis assumem, apenas, valores zero ou um.

Usando a notação matricial para os coeficientes de $f(.)$ e $g(.)$ reescrevemos o nosso problema da seguinte forma: encontre, se existir, o conjunto de *pontos eficientes* em

$$(P): \max \{ Cx : x \in F \}$$

onde: $F \equiv \{ x \in R^n : Ax \leq b \wedge x_j \in \{0, 1\}, j \in J \};$

C é uma matriz $p \times n$, onde os elementos da linha i , $1 \leq i \leq p$, são os coeficientes da função $f_i(.)$. Denotaremos esta matriz de *matriz de critérios* C ;

A é uma matriz $m \times n$, onde os elementos da linha i , $1 \leq i \leq m$, são os coeficientes da função $g_i(.)$.

O nosso problema consiste em determinar:

$x^0 \in F : \nexists x \in F : Cx \geq Cx^0$, conforme a definição 2.1, ou seja, o conjunto de pontos viáveis que não são dominados em (P) por nenhum outro ponto viável.

Denotaremos o conjunto de pontos eficientes em (P) por $EF(P)$.

Resolveremos este problema através de um problema auxiliar: encontre, se existir, o conjunto de *pontos eficientes* em

$$(P') : \max \{Cx : x \in DUC\},$$

$$\text{onde } DUC \equiv \{x \in R^n : x_j \in \{0, 1\}, j \in J\}.$$

Denotaremos o conjunto de pontos eficientes de (P') por $EF(P')$ e o conjunto de pontos de DUC que não são eficientes em (P') por $EF(P')^C$.

O conjunto $EF(P')$ será estudado analisando-se o seguinte conjunto:

$$V \equiv \{v \in R^n : Cv \geq 0, v_j \in \{-1, 0, 1\}, j \in J\}$$

Note-se que, dado $x \in DUC$, se $\exists v \in V$:

$y = (x+v) \in DUC \Rightarrow Cy = C(x+v) \geq Cx$. Neste caso, dizemos que y domina x na direção de v . Neste sentido, V corresponde ao conceito de direções de crescimento em programação matemática.

A seguir, estabeleceremos uma relação entre os elementos de V e os de $EF(P')^C$ através do seguinte mapeamento:

DEFINIÇÃO 2.3. M é um mapeamento,

$$M: V \rightarrow DUC$$

$$v \mapsto M(v) \equiv \{x^i, i=1, \dots, s_v\}$$

$$\text{tal que: } x_j^i = 1 \text{ se } v_j = -1$$

$$x_j^i = 0 \text{ se } v_j = 1$$

$$x_j^i \in \{0, 1\} \text{ se } v_j = 0, j \in J$$

Podemos definir o conjunto imagem deste mapeamento da seguinte forma:

DEFINIÇÃO 2.4. Seja $v \in V$. Então

$$M(v) \equiv Z_1(v) \times Z_2(v) \times \dots \times Z_n(v) = \prod_{j \in J} Z_j(v), \text{ onde:}$$

$$Z_j(v) = \begin{cases} \{1\} & \text{se } v_j = -1 \\ \{0\} & \text{se } v_j = 1 \\ \{0, 1\} & \text{se } v_j = 0, j \in J \end{cases}$$

É fácil de se ver que as definições 2.3 e 2.4 são equivalentes.

EXEMPLO 2.1

Seja

$$V = \{v = (0, 1, -1)\}$$

Então,

$$M(v) = \{x^1 = (0, 0, 1), x^2 = (1, 0, 1)\}$$

Observações:

- a) $\forall v \in V \Rightarrow M(v) \neq \emptyset$
- b) $\text{card}(M(v)) > 1 \Leftrightarrow \exists j \in J : v_j = 0$
- c) se $v \in V$, então $x \in M(v) \Leftrightarrow x + v \in DUC$

PROPOSIÇÃO 2.1 / 2/. Seja

$$(P'): \max\{Cx: x \in DUC\}, \text{ então } EF(P')^C \equiv \bigcup_{v \in V} M(v).$$

Demonstração.

$$\text{Seja } x \in M(v) \text{ para algum } v \in V \Rightarrow x+v \in DUC \Rightarrow x \notin EF(P') \Rightarrow x \in EF(P')^C \Rightarrow \bigcup_{v \in V} M(v) \subset EF(P')^C.$$

$$\text{Seja } x^0 \in EF(P')^C \Rightarrow x^0 \in DUC \setminus EF(P') \Rightarrow \exists x \in DUC: Cx \geq Cx^0 \Rightarrow C(x-x^0) \geq 0. \text{ Faça } v = x-x^0. \text{ Então:}$$

$$i) \quad Cv \geq 0$$

$$ii) \text{ Como } (x_j, x_j^0) \in \{0, 1\} \times \{0, 1\} \Rightarrow v_j \in \{-1, 0, 1\}. \text{ De } i) \text{ e } ii) \\ v \in V \wedge x = (x^0 + v) \in DUC \Rightarrow x^0 \in M(v) \text{ (pela observação c) anterior)} \\ \Rightarrow EF(P')^C \subset M(v).$$

$$\text{Portanto, } EF(P')^C \equiv \bigcup_{v \in V} M(v). \quad \square$$

Definiremos a seguir um mapeamento análogo ao mapeamento M .

DEFINIÇÃO 2.5. N é um mapeamento,

$$N: V \rightarrow DUC$$

$$v \mapsto N(v) \equiv \{y^i, i=1, \dots, s_v\}$$

$$\text{tal que: } \begin{aligned} y_j^i &= 1 & \text{se } v_j = 1, \\ y_j^i &= 0 & \text{se } v_j = -1, \\ y_j^i &\in \{0, 1\} & \text{se } v_j = 0, \quad j \in J. \end{aligned}$$

De maneira análoga ao que foi feito com o conjunto $M(v)$, vamos dar uma definição do conjunto imagem deste mapeamento.

DEFINIÇÃO 2.6. Seja $v \in V$. Então

$$N(v) \equiv W_1(v) \times W_2(v) \times \dots \times W_n(v) = \prod_{j \in J} W_j(v), \text{ onde:}$$

$$W_j(v) = \begin{cases} \{1\} & \text{se } v_j = 1 \\ \{0\} & \text{se } v_j = -1 \\ \{0, 1\} & \text{se } v_j = 0, j \in J \end{cases}$$

É fácil de se ver que as definições 2.5 e 2.6 são equivalentes.

EXEMPLO 2.2

Seja

$$V = \{v = (0, 1, -1)\}.$$

Então,

$$N(v) \equiv \{y^1 = (0, 1, 0), y^2 = (1, 1, 0)\}$$

OBSERVAÇÕES

- a) $\forall v \in V \Rightarrow N(v) \neq \phi$
- b) $\text{card}(N(v)) > 1 \Leftrightarrow \exists j \in J : v_j = 0.$
- c) se $v \in V$, então $y \in N(v) \Leftrightarrow y - v \in DUC.$

Analisando o resultado da proposição 2.1 e a definição de $N(v)$ poder-se-ia conjecturar que $EF(P') \equiv \bigcup_{v \in V} N(v)$ ou que $EF(P') \subseteq \bigcup_{v \in V} N(v)$. Mas, podemos verificar, nos exemplos a baixo, que nenhuma das duas relações é válida.

EXEMPLO 2.3. Seja o problema:

$$(P'): \max \left\{ \begin{bmatrix} -1 & 2 \\ 2 & -1 \end{bmatrix}, x : x \in DUC \right\}$$

Então, $V \equiv \{v = (1, 1)\}$

$$N(v) \equiv \{y^1 = (1, 1)\}$$

$$EF(P') \equiv \{(1, 0), (1, 1), (0, 1)\}$$

Neste caso, $\bigcup_{v \in V} N(v) \subset EF(P')$, mas, geralmente, isto não é verdade, como mostramos no exemplo abaixo.

EXEMPLO 2.4. Seja o problema:

$$(P'): \max \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, x : x \in DUC \right\}$$

Então, $V \equiv \{v^1 = (0, 1), v^2 = (1, 0), v^3 = (1, 1)\}$

$$N(v^1) \equiv \{(0, 1), (1, 1)\}$$

$$N(v^2) \equiv \{(1, 0), (1, 1)\}$$

$$N(v^3) \equiv \{(1, 1)\}$$

$$\bigcup_{v \in V} N(v) \equiv \{(0, 1), (1, 0), (1, 1)\}$$

$$EF(P') \equiv \{(1, 1)\}$$

Podemos, também, notar que

$$M(v^2) \equiv \{(0, 0), (0, 1)\}$$

$$\text{e } N(v^1) \cap M(v^2) \neq \emptyset$$

Pelo fato de $EF(P')^C = \bigcup_{v \in V} M(v)$ (proposição 2.1) concluimos que $\exists y \in N(v^1) : y \in EF(P')^C$.

Contudo, vale o seguinte resultado.

PROPOSIÇÃO 2.2. Seja o problema

$$(P') : \max\{Cx : x \in DUC\}, \text{ então } \bigcap_{v \in V} N(v) \subseteq EF(P').$$

Demonstração. Seja $x \in \bigcap_{v \in V} N(v)$. Suponha que $x \notin EF(P')$. Então, $\exists y \in DUC : Cy \geq Cx \Rightarrow C(y-x) \geq 0$.

Seja $v = y - x$. Então $Cv \geq 0$ e $v_j \in \{-1, 0, 1\}$

Portanto, $v \neq 0$ e $v \in V$. Logo,

$$i) \text{ se } v_j = 1 \Rightarrow x_j = 0$$

$$ii) \text{ se } v_j = -1 \Rightarrow x_j = 1$$

$i)$ e $ii)$ $\Rightarrow x \notin N(v) \Rightarrow x \notin \bigcap_{v \in V} N(v)$, o que é uma contradição. Portanto, $x \in EF(P')$. $\square$

Os conjuntos $M(v)$ e $N(v)$ são relacionados pelo seguinte resultado.

PROPOSIÇÃO 2.3. Para cada $v \in V$, $N(v) \equiv M(v) + \{v\}$.

Demonstração. É consequência imediata das definições de $M(v)$ e $N(v)$. $\square$

A seguir apresentaremos alguns resultados sobre os conjuntos $M(v)$ e $N(v)$ que nos auxiliarão no desenvolvimento do algoritmo para determinar $EF(P)$.

PROPOSIÇÃO 2.4. Seja $(v^1, v^2) \in V \times V$, $v^1 \neq v^2$. $J_0 \equiv \{j \in J : v_j^1 = 0\}$. Então $v_j^1 = v_j^2, \forall j \in J \setminus J_0$, sse $M(v^2) \subset M(v^1)$.

Demonstração.

Condição Necessária.

$$i) \text{ se } x \in M(v^2) \Rightarrow x \in \bigcap_{j \in J} Z_j(v^2) \text{ tal que:}$$

$$Z_i(v^2) \equiv Z_i(v^1), \forall i \in J \setminus J_0 \text{ e}$$

$$Z_i(v^2) \subset \{\{0\}, \{1\}\}, \forall i \in J_0.$$

ii) se $z \in M(v^1) \Rightarrow z \in \prod_{j \in J} Z_j(v^1)$, tal que:

$$Z_i(v^1) \equiv \{0, 1\}, \quad \forall i \in J_0.$$

De i) e ii) $\prod_{j \in J} Z_j(v^2) \subset \prod_{j \in J} Z_j(v^1) \Rightarrow M(v^2) \subset M(v^1)$.

Condição Suficiente. Suponha que $\exists j \in J \setminus J_0$:

$$0 \neq v_j^1 \neq v_j^2 \Rightarrow \exists j \in J \setminus J_0 : Z_j(v^1) \neq Z_j(v^2) \text{ e } Z_j(v^2) \not\subset Z_j(v^1) \Rightarrow$$

$\Rightarrow M(v^2) \not\subset M(v^1)$, o que contradiz a hipótese. Portanto, $v_j^1 = v_j^2$, $\forall j \in J \setminus J_0$. $\square$

PROPOSIÇÃO 2.5. Seja $(v^1, v^2) \in V \times V$, $v^1 \neq v^2$.
 $J_0 \equiv \{j \in J : v_j^1 = 0\}$. Então, $v_j^1 = v_j^2$, $\forall j \in J \setminus J_0$ sse $N(v^2) \subset N(v^1)$.

Demonstração. A demonstração é análoga à da proposição 2.4, usando a definição de $N(v)$. $\square$

Os vetores do tipo $(v^1, v^2) \in V \times V$, $v^1 \neq v^2$, com $J_0 \equiv \{j \in J : v_j^1 = 0\}$ e $v_j^1 = v_j^2$, $\forall j \in J \setminus J_0$, como definidos nas proposições 2.4 e 2.5, são de extrema importância para a determinação do conjunto $EF(P')$. Portanto, damos as seguintes definições:

DEFINIÇÃO 2.7. Seja $(v^1, v^2) \in V \times V$, $v^1 \neq v^2$. Dizemos que v^1 envolve v^2 sse $v_j^1 = 0$, para $j \in J_0 \subset J$ e $v_j^1 = v_j^2$, $\forall j \in J \setminus J_0$.

DEFINIÇÃO 2.8. Dizemos que $v^1 \in V$ é envolvente sse

$\exists v \in V: v \text{ envolve } v^1$.

Denotaremos o subconjunto dos elementos *envolventes* de V por $\bar{V}$, $\bar{V} \subseteq V$. (Note-se que $\bar{V}$ não denota fecho de V ou complementar de V).

PROPOSIÇÃO 2.6. Sejam $(P') : \max\{Cx : x \in DUC\}$ e $\bar{V}$ o subconjunto de vetores *envolventes* de V . Então,

$$\bigcap_{v \in V} N(v) \subseteq \bigcap_{\bar{v} \in \bar{V}} N(\bar{v}) \subseteq EF(P').$$

Demonstração. É imediato que $\bigcap_{v \in V} N(v) \subseteq \bigcap_{\bar{v} \in \bar{V}} N(\bar{v})$, pois $\bar{V} \subseteq V$.

Seja $x \in \bigcap_{\bar{v} \in \bar{V}} N(\bar{v})$. Suponha que $x \notin EF(P')$. Então,

$$\exists y \in DUC : Cy \geq Cx \Rightarrow C(y-x) \geq 0.$$

Seja $v = y - x$. Então, $Cv \geq 0$ e $v_j \in \{-1, 0, 1\} \Rightarrow v \neq 0$ e $v \in V$.

Seja $v \in \bar{V} \Rightarrow x \notin N(\bar{v})$, contradição.

Seja $v \in V \setminus \bar{V} \Rightarrow \exists \bar{v} \in \bar{V} : \bar{v} \text{ envolve } v$, isto é,

$\bar{v}_j = v_j$, $j \in J_0 \subset J$. Então, $y_j = x_j + v_j = x_j + \bar{v}_j$, $j \in J_0 \Rightarrow x \notin N(\bar{v})$, contradição.

Portanto, $x \in EF(P')$. $\square$

PROPOSIÇÃO 2.7. Seja $(v^1, v^2) \in V \times V$. Então $N(v^1) \cap M(v^2) \equiv \phi (M(v^1) \cap N(v^2) \equiv \phi)$ sse $\exists j \in J : v_j^1 \cdot v_j^2 > 0$.

Demonstração.

$$N(v^1) \cap M(v^2) \equiv \phi \iff \bigcap_{j \in J} W_j(v^1) \cap \bigcap_{j \in J} Z_j(v^2) \equiv \phi \iff$$

$\exists j \in J : W_{j0}(v^1) \cap Z_{j0}(v^2) \equiv \phi \iff v_j^1 \cdot v_j^2 > 0$, pela própria definição de $W_j(v^1)$ e $Z_j(v^2)$. $\square$

COROLÁRIO 2.7.1. Para todo $v \in V$, $N(v) \cap M(v) \equiv \phi$.

Demonstração. É consequência imediata da proposição 2.5, pois, $\forall v \in V \Rightarrow v_j \cdot v_j \geq 0$, $\forall j \in J$. Como $0 \notin V$, então, $\forall v \in V$, $\exists j \in J : v_j \cdot v_j > 0$. $\square$

COROLÁRIO 2.7.2. Seja $(v^1, v^2) \in V \times V$, tal que v^1 envolve v^2 . Então $N(v^1) \cap M(v^2) \equiv \phi$ e $N(v^2) \cap M(v^1) \equiv \phi$.

Demonstração. Seja $J_0 \equiv \{j \in J : v_j^1 = 0\}$, $J_0 \subset J$, pois $0 \notin V$.

Como v^1 envolve v^2 , para

$$j \in J \setminus J_0, v_j^1 = v_j^2 \Rightarrow \exists j \in J : v_j^1 \cdot v_j^2 > 0.$$

Pela proposição 2.7 $N(v^1) \cap M(v^2) \equiv \phi$ e $N(v^2) \cap M(v^1) \equiv \phi$. $\square$

PROPOSIÇÃO 2.8. Seja $(v^1, v^2) \in V \times V$. Então

$$M(v^1) \cap M(v^2) \equiv \phi \text{ sse } \exists j \in J : v_j^1 \cdot v_j^2 < 0.$$

Demonstração.

$$M(v^1) \cap M(v^2) \equiv \emptyset \iff \prod_{j \in J} Z_j(v^1) \cap \prod_{j \in J} Z_j(v^2) \equiv \emptyset \iff$$

$\exists j_0 \in J : Z_{j_0}(v^1) \cap Z_{j_0}(v^2) \equiv \emptyset \iff v_{j_0}^1 \cdot v_{j_0}^2 < 0$, pela própria definição de $Z_j(v^1)$ e $Z_j(v^2)$. $\square$

PROPOSIÇÃO 2.9. Seja $(v^1, v^2) \in V \times V$. Então $N(v^1) \cap N(v^2) \equiv \emptyset$ sse $\exists j \in J : v_j^1 \cdot v_j^2 < 0$.

Demonstração. A demonstração é análoga à da proposição 2.8. $\square$

Com relação aos resultados até aqui apresentados, podemos mencionar alguns fatos importantes:

Fato 1: Uma consequência imediata das proposições 2.4 e 2.5, é que o subconjunto de elementos *envolventes* de V , denotado por $\bar{V}$, é suficiente para gerar os conjuntos $\bigcup_{v \in V} M(v)$ e $\bigcup_{v \in V} N(v)$, isto é,

$$\bigcup_{v \in V} M(v) \equiv \bigcup_{\bar{v} \in \bar{V}} M(\bar{v}) \text{ e } \bigcup_{v \in V} N(v) \equiv \bigcup_{\bar{v} \in \bar{V}} N(\bar{v}).$$

Entretanto, o exemplo abaixo mostra que o subconjunto $\bar{V}$ ainda não é o menor subconjunto de V que satisfaz esta propriedade.

EXEMPLO 2.4. Seja a seguinte matriz de critérios:

$$C = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & -1 & -1 & -1 \\ 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 \\ -1 & 0 & 2 & 1 \end{bmatrix}$$

Para esta matriz de critérios C , temos:

$$\bar{V} \equiv \{v^1 = (1, -1, 1, 0), v^2 = (1, -1, 0, 1), v^3 = (1, 0, 1, -1)\}$$

$$M(v^1) \equiv \{(0, 1, 0, 0), (0, 1, 0, 1)\}$$

$$M(v^2) \equiv \{(0, 1, 0, 0), (0, 1, 1, 0)\}$$

$$M(v^3) \equiv \{(0, 0, 0, 1), (0, 1, 0, 1)\}$$

Podemos notar que: $M(v^1) \subset M(v^2) \cup M(v^3)$.

$$N(v^1) \equiv \{(1, 0, 1, 0), (1, 0, 1, 1)\}$$

$$N(v^2) \equiv \{(1, 0, 0, 1), (1, 0, 1, 1)\}$$

$$N(v^3) \equiv \{(1, 0, 1, 0), (1, 1, 1, 0)\}$$

Podemos notar, também, que: $N(v^1) \subset N(v^2) \cup N(v^3)$.

Fato 2: Outra consequência imediata das proposições 2.4 e 2.5, é que se $x \in M(v)$ para algum $v \in V$, então

$\exists \bar{v} \in \bar{V} : x \in M(\bar{v})$ e $y \in N(\bar{v})$ com $y = x + v$. Isto mostra que o subconjunto $\bar{V}$, também é suficiente para gerar os elementos que dominam x em qualquer direção $v \in V$.

EXEMPLO 2.5. Seja a seguinte matriz de critérios C :

$$C = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}$$

Para esta matriz de critérios C , temos:

$$V \equiv \{v^1 = (0, 1), v^2 = (1, 1)\}$$

$$\bar{V} \equiv \{\bar{v} = (0, 1)\}$$

$$M(v^2) \equiv \{x = (0, 0)\}$$

$$M(\bar{v}) \equiv \{(0, 0), (1, 0)\}$$

$$N(\bar{v}) \equiv \{(0, 1), (1, 1)\}$$

$x = (0, 0) \in M(v^2)$, $v^2 \notin \bar{V}$, mas $x \in M(\bar{v})$ e $y = (1, 1) = x + v^2$ e $y \in N(\bar{v})$.

Fato 3: Se $x \in N(v^1) \cap M(v^2)$, então $\exists y \in N(v^2) : y = x + v^2$ e

$$Cy = C(x + v^2) \geq Cx, \text{ pois } Cv^2 \geq 0.$$

EXEMPLO 2.6. Seja a seguinte matriz de critérios

C :

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Para esta matriz de critérios C , temos:

$$V = \{v^1 = (1, 0), v^2 = (0, 1), v^3 = (1, 1)\}$$

$$N(v^1) = \{(1, 0), (1, 1)\}$$

$$M(v^2) = \{(0, 0), (1, 0)\}$$

$$x = (1, 0) \in N(v^1) \cap M(v^2)$$

$$N(v^2) = \{(0, 1), (1, 1)\}$$

$$y = (1, 1) = x + v^2 = (1, 0) + (0, 1)$$

Ou seja, $x \in N(v^1) \cap M(v^2)$ é uma condição suficiente para x não ser eficiente em (P^t) , porém se $x \in N(v^1)$ e $x \notin M(v^2)$, isto não implica que x seja eficiente, pois x pode pertencer a $N(v^1) \cap M(v^3)$.

Fato 4: Pela proposição 2.1, sabemos que $EF(P^t)^C \equiv \bigcup_{v \in V} M(v)$.

Se os elementos do conjunto $\bigcup_{v \in V} M(v)$ forem adjacen-

tes conexos^{*}, os pontos não eficientes em (P') seriam adjacentes conexos.

Se os elementos do conjunto $\bigcup_{v \in V} N(v)$ forem adjacentes conexos, isto simplificaria o algoritmo para determinar $EF(P)$, apresentado no capítulo III pois este é baseado na geração dos elementos de $\bigcup_{v \in V} N(v)$, tal que $\bigcup_{v \in V} N(v) \cap \bigcup_{v \in V} M(v) \equiv \phi$.

Mas, como vemos no exemplo abaixo, nenhuma das duas hipóteses é verdadeira.

EXEMPLO 2.7. Seja a seguinte matriz de critérios C :

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 \\ -1 & 2 & 0 & 1 \\ -1 & 2 & -1 & 0 \\ 1 & -2 & 0 & -1 \end{bmatrix}$$

Para esta matriz de critérios C , temos:

$$V \equiv \{v^1 = (1, 1, 1, -1), v^2 = (1, 0, -1, 1)\}$$

$$M(v^1) \equiv \{(0, 0, 0, 1)\}$$

* A definição de pontos adjacentes conexos encontra-se no apêndice 2.

$$M(v^2) \equiv \{(0, 0, 1, 0), (0, 1, 1, 0)\}$$

$\cup_{v \in V} M(v) \equiv \{(0, 0, 0, 1), (0, 0, 1, 0), (0, 1, 1, 0)\}$, não é adjacente conexo.

$$N(v^1) \equiv \{(1, 1, 1, 0)\}$$

$$N(v^2) \equiv \{(1, 0, 0, 1), (1, 1, 0, 1)\}$$

$\cup_{v \in V} N(v) \equiv \{(1, 1, 1, 0), (1, 0, 0, 1), (1, 1, 0, 1)\}$, não é adjacente conexo.

3. OS CONJUNTOS $M(v)$, $N(v)$ E $\bar{V}$.

Os conjuntos $M(v)$ e $N(v)$, $v \in V$, podem ser caracterizados como conjunto de pontos extremos que resolvem um problema de programação linear, como mostramos nas proposições seguintes:

PROPOSIÇÃO 3.1 / 2/. Para cada $v \in V$, $M(v)$ é o conjunto dos pontos extremos que resolvem:

$$(P_1) : z = \min \{vx : x \in UC\}$$

Demonstração.

i) x é ponto extremo de $UC \iff x \in DUC$

ii) Seja $J^+(v) \equiv \{j \in J : v_j = 1\}$ e $J^-(v) \equiv \{j \in J : v_j = -1\}$

Então,

$$\forall x \in UC, \sum_{j \in J} v_j x_j = \sum_{j \in J^+(v)} x_j + \sum_{j \in J^-(v)} (-x_j) \geq \sum_{j \in J^-(v)} (-1) = \bar{z}$$

iii) $\forall x \in M(v), z = \bar{z}$ é o valor ótimo do problema.

Portanto,

x é ponto extremo, solução de (P_1) $\Leftrightarrow$

x é um elemento de $DUC : \sum_{j \in J} v_j x_j = \bar{z} \Leftrightarrow$

$$\left\langle \begin{array}{l} v_j = -1 \Rightarrow x_j = 1 \\ v_j = 1 \Rightarrow x_j = 0 \\ v_j = 0 \Rightarrow x_j \in \{0, 1\} \end{array} \right\rangle \Leftrightarrow x \in M(v). \quad \square$$

PROPOSIÇÃO 3.2. Para cada $v \in V$, $N(v)$ é o conjunto dos pontos extremos que resolvem:

$$(P_2) : z = \max \{vx : x \in UC\}$$

Demonstração. É análoga à demonstração da proposição anterior, usando a definição de $N(v)$. $\square$

A seguir daremos uma caracterização parcial do subconjunto $\bar{V} \subseteq V$, Suponha que $V = \{v^i, i \in K\}$, $K = \{1, 2, \dots, k\}$. Seja $v^u \in V$ e $J(u) \subseteq K \setminus \{u\}$.

PROPOSIÇÃO 3.3. Seja $v^u \in V$ tal que v^u é combinação convexa de $\{v^i\}_{i \in J(u)}$, isto é, $v^u = \sum_{i \in J(u)} \alpha_i v^i$, $\sum_{i \in J(u)} \alpha_i = 1$, $\forall i \in J(u)$, $\alpha_i \geq 0$. Então, $\exists i \in J(u) : v^i \notin \bar{V}$ e além disto, v^u envolve v^i , $\forall i : \alpha_i > 0$.

Demonstração. Como $\sum_{i \in J(u)} \alpha_i = 1$, $J'(u) \equiv \{i \in J(u) : \alpha_i > 0\} \neq \emptyset$

e $v^u = \sum_{i \in J'(u)} \alpha_i v^i$ e $\sum_{i \in J'(u)} \alpha_i = 1$.

$$\begin{aligned} \forall j : v_j^u \neq 0 &\Rightarrow \forall i \in J'(u), v_j^i = v_j^u \text{ e } v^i \neq v^u \\ &\Rightarrow v^u \text{ envolve } v^i, \forall i \in J'(u) \\ &\Rightarrow \forall i \in J'(u), v^i \notin \bar{V}. \quad \square \end{aligned}$$

Seja $\bar{V} \equiv \{v^i : i \in L\}$, onde $L = \{1, \dots, l\}$.

COROLÁRIO 3.3.1- Todo $v \in \bar{V}$ é ponto extremo de $\bar{V}$.

Demonstração. Suponha que $v \in \bar{V}$ não seja ponto extremo de $\bar{V}$. Então, $\exists v^i, v^j \in \bar{V}$, $v^i \neq v^j$, $i, j \in L$, tais que: $v = \sum_{i \in L} \alpha_i v^i$, $\sum_{i \in L} \alpha_i = 1$, $\forall i \in L$, $\alpha_i \geq 0$. Pela proposição anterior, $\exists i \in L : v^i \notin \bar{V}$. Portanto, $v \in \bar{V}$ é ponto extremo de $\bar{V}$. $\square$

COROLÁRIO 3.3.2. Seja $v^u \in V$ e $v^u = \sum_{i \in J(u)} \alpha_i v^i$, $\sum_{i \in J(u)} \alpha_i = 1$,

$\forall i \in J(u), \alpha_i \geq 0$ e $J'(u) \equiv \{i \in J(u) : \alpha_i > 0\}$. Então $\bigcup_{i \in J'(u)} M(v^i) \subseteq M(v^u)$.

Demonstração. É uma consequência imediata das proposições 3.1 e 2.4. $\square$

A caracterização dada ao subconjunto $\bar{V}$ é uma caracterização parcial, pois, os pontos extremos de V não pertencem necessariamente ao subconjunto $\bar{V}$ e nem sequer todo ponto de $\bar{V}$ é necessariamente uma combinação convexa dos elementos de V como ilustramos no exemplo abaixo.

EXEMPLO 3.1. Seja a seguinte matriz de critérios C :

$$C = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$V \equiv \{v^1 = (0, 1), v^2 = (1, 1)\}$$

$$\bar{V} \equiv \{\bar{v} = (0, 1)\}$$

$v^2 = (1, 1)$ é ponto extremo de V , mas $v^2 \notin \bar{V}$.

$\bar{v} = (0, 1)$ não pode ser escrito como combinação convexa dos elementos de V .

4. OS CONJUNTOS $\mathcal{M}_b(v)$ E $\mathcal{N}(v)$.

DEFINIÇÃO 4.1. Dado um conjunto X convexo, uma *face* de X é um subconjunto convexo X' de X , tal que, todo segmento de reta em X com um ponto em seu interior relativo pertencente a X' , tem as extremidades em X' .

Com X e X' nas condições da definição e sendo $[a,b]$ segmento de reta em X , esta definição afirma que: X' é uma *face* $\Leftrightarrow x \in (a,b)$, $x \in X' \Rightarrow a \in X' \wedge b \in X'$.

DEFINIÇÃO 4.2. $\mathcal{M}_b$ é um mapeamento,

$$\mathcal{M}_b: V \rightarrow UC$$

$$v \mapsto \mathcal{M}_b(v) \equiv \arg \min \{vx : x \in UC\}.$$

EXEMPLO 4.1. Seja $v = (0, 1)$

$$\mathcal{M}_b(v) \equiv \{(x_1, 0) : x_1 \in [0, 1]\}$$

PROPOSIÇÃO 4.1. Para cada $v \in V$, $\mathcal{M}(v)$ é uma face de UC .

Demonstração.

i) $\mathcal{M}(v) \subseteq UC$, limitado, pela própria definição de $\mathcal{M}(v)$.

ii) $\mathcal{M}(v) \equiv \{x \in UC : v \cdot x = v \cdot x^i, \text{ para } x^i \in M(v)\}$.

Portanto, $\mathcal{M}(v)$ é a intersecção de UC com um hiperplano. Logo, $\mathcal{M}(v)$ é um politopo convexo.

iii) Seja o segmento de reta $[a, b]$ em UC , $x \in (a, b)$ e $x \in \text{ri } \mathcal{M}(v)$. Então:

$$x = \lambda a + (1-\lambda)b, \quad 0 < \lambda < 1$$

$$v \cdot x = \lambda(v \cdot a) + (1-\lambda)(v \cdot b), \quad 0 < \lambda < 1$$

Da otimalidade de x :

$$v \cdot x = v \cdot a = v \cdot b = \min \{v \cdot x : x \in UC\}$$

$$(a, b) \in UC \times UC \Rightarrow a \in \mathcal{M}(v) \text{ e } b \in \mathcal{M}(v). \quad \square$$

PROPOSIÇÃO 4.2. Para cada $v \in V$, $\mathcal{M}(v) \equiv [M(v)]$.

Demonstração. Na proposição anterior mostramos

que $\mathcal{M}(v)$ é um politopo convexo limitado. Pela proposição 3.1 e definição 4.2, $M(v)$ é o conjunto dos pontos extremos de $\mathcal{M}(v)$. Portanto, por um resultado clássico de programação linear, $\mathcal{M}(v) \equiv [M(v)]$. $\square$

PROPOSIÇÃO 4.3. Sejam $J^+(v) \equiv \{j \in J : v_j = 1\}$ e $J^-(v) \equiv \{j \in J : v_j = -1\}$.

$$\mathcal{M}(v) \equiv \left[\bigcap_{j \in J^+(v)} \{x : x_j = 0\} \right] \cap \left[\bigcap_{j \in J^-(v)} \{x : x_j = 1\} \right] \cap UC.$$

Demonstração. É imediato a partir da definição 4.2 e da proposição 3.1, pois

$$\mathcal{M}(v) \equiv \{x \in UC : v \cdot x = v \cdot x^i, x^i \in M(v)\}. \quad \square$$

COROLÁRIO 4.3.1. Seja $v \in V$. Então $\dim \mathcal{M}(v) = n - \text{número de componentes não nulas de } v$, isto é, $\dim \mathcal{M}(v) = \text{número de componentes nulas de } v$.

Demonstração. É consequência imediata da proposição anterior. $\square$

Dado o seguinte problema:

$$\max \{Cx : x \in UC\}$$

mostraremos, a seguir, que $\mathcal{M}(v)$ é uma face dominada de UC em relação à matriz de critérios C , isto é, se

$$x \in \mathcal{M}(v) \Rightarrow x+v \in UC \Rightarrow C(x+v) \geq Cx.$$

PROPOSIÇÃO 4.4. Seja $v \in V$ e $x^0 \in \mathcal{M}(v)$. Então $x^0 + v \in UC$.

Demonstração.

i) UC é convexo.

ii) Seja $M(v) \equiv \{x^i, i=1, \dots, s\}$.

$$\forall x^i \in M(v) \Rightarrow x^i + v \in DUC \subset UC \Rightarrow x^i + v \in UC$$

iii) Se $x^0 \in \mathcal{M}(v) \Rightarrow x^0 = \sum_{i=1}^s \alpha_i x^i$, $\sum_{i=1}^s \alpha_i = 1$, $\alpha_i \geq 0$, $1 \leq i \leq s$

$$\Rightarrow x^0 + v = \sum_{i=1}^s \alpha_i x^i + v = \sum_{i=1}^s \alpha_i (x^i + v).$$

i), ii) e iii) $\Rightarrow x^0 + v \in UC$. $\square$

Apesar de $\mathcal{M}(v)$ ser uma face dominada em UC , o exemplo abaixo mostrará que não é necessário que todo ponto não eficiente em UC pertença a algum $\mathcal{M}(v)$.

EXEMPLO 2.4.2. Seja o seguinte problema:

$$\max \{Cx : x \in UC\}$$

onde a matriz de critérios C é:

$$C = \begin{bmatrix} -1 & 3 \\ 1 & -1 \end{bmatrix}$$

Para esta matriz de critérios C :

$$V \equiv \{v = (1, 1)\}$$

$$M(v) \equiv \{(0, 0)\}$$

$$\mathcal{M}(v) \equiv \{(0, 0)\}$$

Entretanto, $x = (1, 1/2)$ é um ponto não eficiente de UC (pois $(1, 1)$ domina $(1, 1/2)$) e $x \notin \mathcal{M}(v)$.

A seguir mostraremos as relações entre as faces $\mathcal{M}(v)$ de UC , $v \in V$. Mas, antes demonstraremos alguns resultados sobre faces de um conjunto convexo X .

Dado um conjunto convexo X , denotaremos *fronteira relativa* de X por $\text{relbd } X$ e *interior relativo* de X por $\text{ri}X$.

PROPOSIÇÃO 4.5 /12/. Seja X um conjunto convexo e seja X' uma face de X . Se Y é um subconjunto convexo de X , tal que $\text{ri}Y \cap X' \neq \emptyset$, então $Y \subset X'$.

Demonstração.

Seja $z \in \text{ri } Y \cap X'$.

Se $x \in Y$, $x \neq z \Rightarrow \exists y \in Y: z = \alpha x + (1-\alpha)y$, $0 \leq \alpha \leq 1$. Como X' é uma face de $X \Rightarrow x \in X'$ e $y \in X'$.

Portanto, $Y \subset X'$. $\square$

COROLÁRIO 4.5.1 /12/. Se X' e X'' são faces de um conjunto convexo X , tais que $\text{ri } X' \cap \text{ri } X'' \neq \emptyset$, então $X' \equiv X''$.

Demonstração.

i) pela proposição anterior, $X' \subset X''$, pois

$$\text{ri } X' \cap \text{ri } X'' \neq \emptyset$$

ii) Analogamente, $X'' \subset X'$.

De i) e ii), $X' \equiv X''$. $\square$

PROPOSIÇÃO 4.6. Sejam $(v^i, v^j) \in V \times V$, $v^i \neq v^j$. Então $\text{ri } \mathcal{M}_b(v^i) \cap \text{ri } \mathcal{M}_b(v^j) \equiv \emptyset$.

Demonstração. Suponha que $\text{ri } \mathcal{M}_b(v^i) \cap \text{ri } \mathcal{M}_b(v^j) \neq \emptyset$.

Pelo corolário anterior, $\mathcal{M}_b(v^i) \equiv \mathcal{M}_b(v^j) \Rightarrow M(v^i) \equiv M(v^j) \Rightarrow v^i = v^j$, o que contradiz a hipótese.

Portanto, $\text{ri } \mathcal{M}_b(v^i) \cap \text{ri } \mathcal{M}_b(v^j) \equiv \emptyset$. $\square$

COROLÁRIO 4.6.1. Sejam $(v^1, v^2) \in V \times V$, $v^1 \neq v^2$. Então v^1 envolve v^2 sse $\mathcal{M}(v^2) \subset_{relbd} \mathcal{M}(v^1)$.

Demonstração. Pela proposição 2.4, v^1 envolve $v^2 \Leftrightarrow \mathcal{M}(v^2) \subset \mathcal{M}(v^1)$. Portanto,

$$i) \quad v^1 \text{ envolve } v^2 \Leftrightarrow \mathcal{M}(v^2) \subset \mathcal{M}(v^1).$$

Pela proposição anterior

$$ii) \quad ri \mathcal{M}(v^1) \cap ri \mathcal{M}(v^2) \equiv \emptyset$$

$$\text{De } i) \text{ e } ii), \quad v^1 \text{ envolve } v^2 \Leftrightarrow \mathcal{M}(v^2) \subset_{relbd} \mathcal{M}(v^1).$$

□

COROLÁRIO 4.6.2. Sejam $(v^1, v^2) \in \bar{V} \times \bar{V}$, $v^1 \neq v^2$. Então, $\mathcal{M}(v^1) \cap \mathcal{M}(v^2) \equiv_{relbd} \mathcal{M}(v^1) \cap_{relbd} \mathcal{M}(v^2)$.

Demonstração. Pela proposição anterior,

$$ri \mathcal{M}(v^1) \cap ri \mathcal{M}(v^2) \equiv \emptyset.$$

Como $(v^1, v^2) \in \bar{V} \times \bar{V}$, pelo corolário anterior,

$$\mathcal{M}(v^1) \not\subset_{relbd} \mathcal{M}(v^2) \text{ e}$$

$$\mathcal{M}(v^2) \not\subset_{relbd} \mathcal{M}(v^1).$$

Portanto,

$$i) \quad \mathcal{M}(v^1) \cap \mathcal{M}(v^2) \subset_{relbd} \mathcal{M}(v^1) \cap_{relbd} \mathcal{M}(v^2)$$

$$ii) \text{relbd } \mathcal{M}(v^1) \cap \text{relbd } \mathcal{M}(v^2) \subset \mathcal{M}(v^1) \cap \mathcal{M}(v^2)$$

$$\text{De } i) \text{ e } ii), \mathcal{M}(v^1) \cap \mathcal{M}(v^2) \equiv \text{relbd } \mathcal{M}(v^1) \cap \text{relbd } \mathcal{M}(v^2). \quad \square$$

Seria natural nesse ponto se perguntássemos: o conjunto $\bigcup_{v \in V} \mathcal{M}(v)$ é conexo? A resposta é não e será ilustrada no seguinte exemplo.

EXEMPLO 2.4.3. Seja o problema:

$\max\{Cx : x \in UC\}$, com a seguinte matriz de critérios C :

$$C = \begin{bmatrix} -1 & -1 & 2 \\ 1 & 1 & -2 \\ 2 & 1 & -3 \\ 1 & 1 & 1 \end{bmatrix}$$

$$V \equiv \{v^1 = (1, 1, 1), v^2 = (1, -1, 0)\}$$

$$\mathcal{M}(v^1) \equiv \{(0, 0, 0)\}$$

$$\mathcal{M}(v^2) \equiv \{(0, 1, 0), (0, 1, 1)\}$$

$$\mathcal{M}(v^1) \equiv \{(0, 0, 0,)\}$$

$$\mathcal{M}(v^2) \equiv \{\alpha(0, 1, 0) + (1-\alpha)(0, 1, 1), 0 \leq \alpha \leq 1\}$$

PROPOSIÇÃO 4.7. Para cada $v \in V$, $N(v)$ é uma face de UC .

Demonstração. É análoga à demonstração da proposição 4.1. $\square$

De forma semelhante ao que foi feito para $N(v)$, podemos dizer que:

a) $N(v) \equiv [N(v)]$;

b) Sendo $J^+(v) \equiv \{j \in J : v_j = 1\}$ e $J^-(v) \equiv \{j \in J : v_j = -1\}$,

$$N(v) \equiv \left[\bigcap_{j \in J^+(v)} \{x : x_j = 1\} \right] \cap \left[\bigcap_{j \in J^-(v)} \{x : x_j = 0\} \right] \cap UC.$$

Consideremos o seguinte fato. Um ponto $x^0 \in UC$ é não dominado em UC na direção $v \in V$, isto é, $x^0 + \lambda v \notin UC, \forall \lambda > 0$, se e o sistema:

$$A[v] \begin{cases} x^0 + \lambda v \leq 1 \\ x^0 + \lambda v \geq 0 \\ \lambda > 0 \end{cases}$$

não tem solução, $\forall v \in V$.

Estes sistemas $A[v]$, $\forall v \in V$, podem ser reescritos como:

$$A[v] \begin{cases} \lambda v \leq 1 - x^0 \\ \lambda v \geq -x^0 \\ \lambda > 0 \end{cases} \quad \text{ou} \quad A[v] \begin{cases} \lambda v \leq 1 - x^0 \\ -\lambda v \leq x^0 \\ \lambda > 0 \end{cases}$$

Pelo lema não-homogêneo de Farkas* / 8 /, ou $A[v]$ tem solução ou um dos dois sistemas abaixo $H[v]$ ou $G[v]$ tem

* Lema não homogêneo de Farkas:

Dados:

uma matriz A , $p \times n$

os vetores $b \in R^n$, $c \in R^p$ e um escalar β ,

ou

I : $Ax \leq c$, $bx > \beta$, tem solução $x \in R^n$,

ou

$$\text{II.} \quad \left\langle \begin{array}{l} A^T y = b, \quad cy \leq \beta, \quad y \geq 0 \\ \text{ou} \\ A^T y = 0, \quad cy < 0, \quad y \geq 0 \end{array} \right\rangle$$

tem uma solução
 $y \in R^p$

mas nunca ambos.

solução, mas nunca ambos, sendo:

$$H[v] \begin{cases} vz - vw = 1 \\ (\underline{1} - x^0) z + x^0 w \leq 0 \\ z, w \geq \underline{0} \end{cases}$$

$$G[v] \begin{cases} vz - vw = 0 \\ (\underline{1} - x^0) z + x^0 w < 0 \\ z, w \geq \underline{0} \end{cases}$$

Como no nosso caso $x_j^0 \in [0, 1]$, $j \in J$, é fácil ver que $G[v]$ é sempre impossível. Portanto, o lema não-homogêneo de Farkas se reduz ao seguinte:

"Ou $A[v]$ ou $H[v]$ tem solução, mas nunca ambos".

A desigualdade $(\underline{1} - x^0) z + x^0 w \leq 0$ em $H[v]$ não pode ser estrita pois $x^0 \in UC$, portanto $x_j^0 \in [0, 1]$. Esta igualdade é, portanto, equivalente às igualdades:

$$\begin{aligned} (\underline{1} - x^0) z &= 0 \\ x^0 w &= 0 \end{aligned}$$

O sistema $H[v]$ se torna:

$$H[v] \left\{ \begin{array}{ll} vz - vw = 1 & (4.1) \\ (\underline{1} - x^0) z = 0 & (4.2) \\ x^0 w = 0 & (4.3) \\ z, w \geq 0 & (4.4) \end{array} \right.$$

As igualdades (4.2) e (4.4) implicam que $z_{j_0} = 0, \forall j_0: 0 \leq x_{j_0} < 1$ e as igualdades (4.3) e (4.4) implicam que $w_{j_0} = 0, \forall j_0: 0 < x_{j_0} \leq 1$.

Denotando por

$$J^1(x^0) \equiv \{j \in J: x_j^0 = 1\}$$

$$J^2(x^0) \equiv \{j \in J: x_j^0 = 0\}$$

as igualdades de (4.1) a (4.4) podem ser reescritas da seguinte forma:

$$H[v] \left\{ \begin{array}{ll} \sum_{j \in J^1(x^0)} v_j z_j - \sum_{s \in J^2(x^0)} v_s w_s = 1 & \\ z_j, w_s \geq 0, j \in J^1(x^0) \text{ e } s \in J^2(x^0) & \\ z_j = w_s = 0, j \notin J^1(x^0) \text{ e } s \notin J^2(x^0) & \end{array} \right.$$

Os resultados acima implicam na seguinte proposição:

PROPOSIÇÃO 4.8. Um ponto $x^0 \in UC$ não é dominado em UC , $\forall v \in V$, sse $\forall v \in V$, o sistema $H[v]$ tem solução.

Quando $x^0 \in DUC$ e estamos interessados na dominância de x^0 em DUC a proposição 4.8 pode ser enunciada da seguinte forma:

"Um ponto $x^0 \in DUC$ não é dominado em DUC , $\forall v \in V$, ou e quivalentemente, $x^0 \in EF(P')$, sse $\forall v \in \bar{V}$, o sistema $H[v]$ tem solução.

O fato de podermos considerar apenas $v \in \bar{V}$ se deve à proposição 2.4 e à definição de $\bar{V}$.

PROPOSIÇÃO 4.9. Seja $v^u \in V$. Se $\exists j_0 \in J$ tal que $v_{j_0}^u \cdot v_{j_0}^i > 0$, $\forall v^i \in \bar{V}$, então $N(v^u)$ é uma face não dominada de UC em relação ao cone CV gerado por $\{v: v \in V\}$ e em relação ao cone $C\bar{V}$ gerado por $\{\bar{v}: \bar{v} \in \bar{V}\}$.

Demonstração. Suponhamos que $v_{j_0}^u = 1$ (a demonstração para $v_{j_0}^u = -1$ é análoga). Então,

$$N(v^u) = \{x^k : x_{j_0}^k = 1 \text{ e } j_0 \in J^1(x^k), k=1, \dots, s\}$$

Se $x^0 \in \mathcal{N}(v^u) \Rightarrow x^0 = \sum_{k=1}^s \alpha_k x^k$, $\sum_{k=1}^s \alpha_k = 1$, $\alpha_k \geq 0$, $k=1, \dots, s$

$\Rightarrow x_{j_0}^0 = 1$ e $j_0 \in J^1(x^0)$.

Então, (z^0, w^0) com $z_{j_0}^0 = 1$ e $z_j^0 = 0$, $\forall j \neq j_0, j_0 \in J^1(x^0)$ e $w_s^0 = 0$, $s \in J^2(x^0)$, resolve $H[v^u]$. Como $v_{j_0} = 1$, $\forall v \in V$, (z^0, w^0) resolve $H[v]$, $\forall v \in V$.

Se $\hat{v}$ é uma combinação convexa dos elementos de v , então (z^0, w^0) resolve $H[\hat{v}]$. Portanto, x^0 não é dominado em UC em relação ao cone CV . Logo $\mathcal{N}(v^u)$ é uma face não dominada de UC em relação ao cone CV .

Como $C\bar{V} \subset CV \Rightarrow \mathcal{N}(v^u)$ é uma face não dominada de UC em relação ao cone $C\bar{V}$. $\square$

PROPOSIÇÃO 4.10. Seja $v^u \in V$ e $v^u \cdot v > 0$, $\forall v \in \bar{V}$. Então $\mathcal{N}(v^u)$ é uma face não dominada de UC em relação ao cone $C\bar{V}$ gerado por $\{v: v \in V\}$.

Demonstração. Seja $x^0 \in \mathcal{N}(v^u)$. Pela definição de $\mathcal{N}(v^u)$, $v^u x^0 \geq v^u x$, $\forall x \in UC \Rightarrow v^u x \leq v^u x^0 < v^u(x^0 + v)$, $\forall x \in UC$. Então o hiperplano $H = \{x \in R^n: v^u x = v^u x^0\}$, separa estritamente em x_{UC}^0 e o $C\bar{V} \setminus \{0\}$. Portanto, $\mathcal{N}(v^u)$ é uma face não dominada de UC em relação a $C\bar{V}$. $\square$

OBSERVAÇÃO.

A hipótese da proposição 4.8, $v^u \in V$ e $v^u \cdot v > 0$,
 $\forall v \in \bar{V} \Rightarrow$ para cada $v \in \bar{V}$, $\exists j(v) : v_j^u(v) \cdot v_j(v) > 0$. Pela proposi-
 ção 2.7, $N(v^u) \cap M(v) \equiv \emptyset$, $\forall v \in \bar{V} \Rightarrow N(v^u) \cap \left(\bigcup_{v \in \bar{V}} M(v) \right) \equiv \emptyset$. Pela pro-
 posição 2.1,

$$EF(P')^C \equiv \bigcup_{v \in V} M(v) \Rightarrow EF(P')^C = \bigcup_{v \in \bar{V}} M(v). \text{ Portanto, } N(v^u) \subseteq EF(P').$$

O que a proposição 4.8 afirma é que $N(v^u)$ é uma
 face não dominada de UC num problema de programação vetorial
 linear:

$$(\bar{P}') : \max \{ \bar{C}x : x \in UC \},$$

onde $\bar{C}$ é a matriz cujas linhas são formadas por vetores
 que geram o cone polar reverso de $C\bar{V}$. Neste caso $N(v^u)$ pode
 ser obtido por algoritmos desenvolvidos para problemas de
 programação vetorial linear com variáveis contínuas.

Portanto, condições suficientes como as das propo-
 sições 2.7 e 2.8 podem ajudar a reduzir os problemas de pro-
 gramção vetorial linear com variáveis zero-um à obtenção de
 pontos extremos eficientes em $(\bar{P}')$. Algoritmos que resolvem
 este tipo de problemas podem ser encontrados em / 4 /, / 17 /
 e / 3 /. Para testar se um ponto extremo é eficiente ou não,
 os dois primeiros resolvem problemas de programação linear
 e o último dá uma nova caracterização para eficiência de u-

ma aresta incidente ao ponto extremo eficiente e com isto evita a resolução de problemas de programação linear.

CAPÍTULO III

DESCRIÇÃO DO ALGORITMO PARA O PROBLEMA (P)

1. INTRODUÇÃO

Neste capítulo apresentaremos um algoritmo para determinar, se existir, o conjunto de *pontos eficientes* para o problema:

$$(P): \max \{Cx : x \in F\},$$

onde $F \equiv \{x : x \in R^n, Ax \leq b, x_j \in \{0, 1\}, j \in J\}$.

Este algoritmo foi desenvolvido com base, principalmente, nos elementos do subconjunto $\bar{V} \subseteq V$, $\bar{V} \equiv \{\bar{v} : \bar{v} \in V \wedge \nexists v \in V : v \text{ envolve } \bar{v}^1\}$, e nos resultados teóricos apresentados no capítulo anterior, sobre os mapeamentos M e N definidos de V em DUC . Com a definição destes mapeamentos M e N , podemos representar os elementos do conjunto DUC por meio de um grafo dirigido* cujos vértices são os elementos de DUC

* a definição de grafo dirigido encontra-se no apêndice 2.

e os arcos são os elementos de $\bar{V}$, pois, sabemos que, se $x^0 \in M(v)$, $v \in \bar{V}$, então $x^0 + v \in DUC$ ou se $x^0 \in N(v')$, $v' \in \bar{V}$ e $x^0 \notin M(v)$, $\forall v \in \bar{V}$, então $x^0 + v \notin DUC$.

Na seção 2, mostraremos a construção desse grafo dirigido e apresentaremos algumas das propriedades mais importantes para o desenvolvimento do algoritmo. Em seguida, descreveremos um algoritmo para determinar o conjunto $EF(P)$, o conjunto de pontos eficientes em (P) .

Nas seções seguintes, descreveremos os procedimentos que foram utilizados para determinar os conjuntos que foram necessários para esse algoritmo.

Na seção 3, descreveremos um esquema de enumeração implícita, através da geração de uma árvore, para determinar os elementos do conjunto $\bar{V}$.

Na seção 4, descreveremos um procedimento para determinar o conjunto

$$\{x : x \in DUC \wedge x + v \notin DUC, \forall v \in \bar{V}\} \equiv \bigcup_{v \in \bar{V}} N(v) \setminus \bigcup_{v \in \bar{V}} M(v),$$

através da geração de uma árvore binária. Com uma escolha conveniente de elementos de $\bar{V}$ e baseado numa análise destes elementos, tornamos a geração dessa árvore num esquema de enumeração implícita.

Na seção 5, apresentaremos os principais testes realizados para determinar o conjunto de pontos eficientes em

(P). Esta seção foi subdividida em 3 subseções. Na subseção 5.1, apresentaremos o teste de dominância entre os elementos de dois conjuntos distintos; na subseção 5.2, dominância entre os elementos do próprio conjunto; na subseção 5.3, a obtenção de antecessores no grafo dirigido e na subseção 5.4, apresentaremos o teste de viabilidade dos elementos de DUC.

2. O ALGORITMO

A idéia básica do algoritmo que apresentaremos é de terminar o conjunto de todos os pontos eficientes em (P), a partir dos elementos do conjunto $\bigcup_{v \in V} N(v) \equiv \bigcup_{\bar{v} \in \bar{V}} N(\bar{v})$ que não pertencem a $\bigcup_{v \in V} M(v)$, pois $EF(P') \equiv DUC \setminus \bigcup_{v \in V} M(v)$.

O fato de podermos trabalhar com o conjunto $\bar{V}$ é justificado pela proposição 2.5, do capítulo II, que afirma: se $(v^1, v^2) \in V \times V, v^1 \neq v^2$, então v^1 envolve $v^2 \iff N(v^2) \subset N(v^1)$. Ainda, conjecturamos que seja possível trabalhar com um subconjunto de $\bar{V}$ e obter o conjunto $\bigcup_{v \in V} N(v)$, pois, como mostramos no exemplo abaixo, $\exists \bar{v} \in \bar{V} : N(\bar{v}) \subset \bigcup_{v \in \bar{V}, v \neq \bar{v}} N(v)$.

EXEMPLO 2.1 - Seja a seguinte matriz de critérios:

$$C = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & -1 & -1 & -1 \\ 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 \\ -1 & 0 & 2 & 1 \end{bmatrix}$$

Para esta matriz de critérios temos:

$$\bar{V} = \{v^1 = (1, -1, 1, 0), v^2 = (1, -1, 0, 1), v^3 = (1, 0, 1, -1)\}$$

$$N(v^1) = \{(1, 0, 1, 0), (1, 0, 1, 1)\}$$

$$N(v^2) = \{(1, 0, 0, 1), (1, 0, 1, 1)\}$$

$$N(v^3) = \{(1, 0, 1, 0), (1, 1, 1, 0)\}$$

$$N(v^1) \subset N(v^2) \cup N(v^3).$$

Para este algoritmo, os vértices de um hipercubo unitário *DUC* podem ser vistos como vértices de um grafo dirigido, em que os arcos são elementos de $\bar{V}$. Dois vértices x^1 e x^2 estão ligados por um arco $v \in \bar{V}$ com origem em x^1 e extremidade em x^2 sse $x^2 = x^1 + v$. Neste caso, dizemos que x^2 é *sucessor* de x^1 e x^1 é *antecessor* de x^2 . Denotaremos este grafo dirigido por $G(DUC, \bar{V})$.

EXEMPLO 3.2. Seja

$$\bar{V} \equiv \{v^1 = (1, 0, 0), v^2 = (0, -1, 0), v^3 = (0, 1, 1)\}$$

Vamos citar algumas propriedades de $G(DUC, \bar{V})$ que serão úteis no desenvolvimento do algoritmo.

- 1) como foi visto no exemplo 2.3, do capítulo II ,
nem todo ponto de $EF(P')$ pertence a algum $N(v)$,
 $v \in V$. Portanto, em $G(DUC, \bar{V})$ podemos ter *vértices isolados*.
- 2) qualquer *vértice terminal* de $G(DUC, \bar{V})$ pertence a $EF(P')$, enquanto que os seus antecessores são ele

mentos de $\cup_{v \in \bar{V}} M(v)$. No exemplo 3.2 existe um único vértice terminal $x=(1,0,1)$.

- 3) não existem ciclos em $G(DUC, \bar{V})$, o que equivale a dizer o seguinte: a soma dos elementos de quaisquer subconjuntos de $\bar{V}$ não é zero. Suponhamos que existisse um ciclo em $G(DUC, \bar{V})$, como por exemplo:

$$\text{Então: } x^1 = x^3 + v^3 = x^2 + v^2 + v^3 = x^1 + v^1 + v^2 + v^3$$

$$\text{Portanto: } v^1 + v^2 + v^3 = 0$$

Também podemos notar que:

$$Cx^2 \geq Cx^1$$

$$Cx^3 \geq Cx^2$$

$$Cx^1 \geq Cx^3 \geq Cx^2 \geq Cx^1, \text{ o que é absurdo.}$$

Se no grafo dirigido $G(DUC, \bar{V})$ considerarmos um sub-

conjunto qualquer $\bar{V}_1$, $\bar{V}_1 \subseteq \bar{V}$, obteremos um grafo dirigido $G(DUC, \bar{V}_1)$ no qual o número de *vértices terminais* e de *vértices isolados* é maior do que em $G(DUC, \bar{V})$. A propriedade 2) mencionada acima não é válida para $G(DUC, \bar{V}_1)$, mas devido às vantagens citadas no fim da seção 3, o algoritmo será desenvolvido utilizando-se um subconjunto $\bar{V}_1$, $\bar{V}_1 \subseteq \bar{V}$.

Apresentaremos a seguir, a descrição do algoritmo - que determina o conjunto de *pontos eficientes* para o problema (P), o conjunto $EF(P)$.

Observemos que o conjunto F utilizado na descrição do algoritmo é o conjunto de pontos viáveis:

$$F \equiv \{x \in R^n : Ax \leq b \wedge x_j \in \{0, 1\}\}$$

e o conjunto DUC é:

$$DUC \equiv \{x \in R^n : x_j \in \{0, 1\}\}$$

O algoritmo será dividido, basicamente, em três partes:

Parte I - determine um subconjunto $\bar{V}_1$ de $\bar{V}$.

Parte II- determine:

$D \equiv \{x \in DUC : x+v \notin DUC, \forall v \in \bar{V}_1\}$, que corresponde ao conjunto de vértices do hipercubo unitário que, ou são vértices isolados ou são

vértices terminais em $G(DUC, \bar{V}_1)$.

Parte III-Sejam $E_1 \equiv \emptyset$, $E_2 \equiv \emptyset$ e $EF(P) \equiv \emptyset$.

Passo 1 - determine:

$$ED \equiv \{x \in D \cap F : \nexists y \in D \cap F : Cy \geq Cx\}$$

Passo 2 - determine:

$$E_1 \equiv \{x \in ED : \nexists z \in D \setminus F : Cz \geq Cx\}$$

$$E_2 \equiv ED \setminus E_1$$

Passo 3 - determine:

$$E_3 \equiv \{x \in D \setminus F : \nexists w \in (E_1 \cup E_2) : Cw \geq Cx\}$$

Passo 4 - se $E_3 \equiv \emptyset$, faça $EF(P) \leftarrow E_1$ e pare. Em caso contrário, determine:

$$E_4 \equiv \{y : y = z - v \in F, v \in \bar{V}_1, z \in E_3\}$$

e

$$E_5 \equiv \{y' : y' = z - v \in DUC \setminus F, v \in \bar{V}_1, z \in E_3\}$$

Passo 5 - determine:

$$E_6 \equiv \{y \in E_4 : \nexists w \in (E_1 \cup E_2 \cup E_4) : Cw \geq Cy\}$$

Passo 6 - determine:

$$E_2 \equiv \{x \in E_2 : \nexists y \in E_6 : Cy \geq Cx\} \text{ e}$$

faça:

$$E_2 \leftarrow E_2 \cup E_6$$

Passo 7 - determine:

$$E_3 \equiv \{y' \in E_5 : \nexists w \in (E_1 \cup E_2) : Cw \geq Cy'\}$$

Passo 8 - determine:

$$E'_2 \equiv \{x \in E_2 : \nexists y \in E_3 : Cy \geq Cx\}$$

e faça:

$$E_1 \leftarrow E_1 \cup E'_2$$

$$E_2 \leftarrow E_2 \setminus E'_2$$

e volte para o passo 4.

Vamos mostrar que este algoritmo resolve o problema:

$$(P) : \max \{Cx : x \in F\},$$

onde $F \equiv \{x \in R^n : Ax \leq b, x_j \in \{0, 1\}\}$, isto é, determina somente pontos eficientes e além disto, determina todos os pontos eficientes.

Observe-se que:

1) $\forall x \in DUC \setminus D$ pode ser atingido percorrendo em "sentido contrário" os diversos caminhos do grafo dirigido $G(DUC, \bar{V}_1)$;

2) evidentemente, para $\forall x \in DUC \setminus D$, $\exists y \in D: Cy \geq Cx$ e $\forall y \in D$, $\nexists x \in DUC \setminus D: Cx \geq Cy$.

O algoritmo termina quando $E_3 = \emptyset$. Cabe mostrar que ele termina e que quando termina o conjunto E_1 é o conjunto de pontos eficientes em (P) .

Temos que considerar dois casos:

i) o algoritmo termina na primeira vez que o passo 4 é executado.

Se $E_3 = \emptyset \Rightarrow E_2 = \emptyset$ e $\forall x^0 \in E_1, \nexists x \in D: Cx \geq Cx^0$.

Como $\forall x^0 \in E_1, x^0 \in F$, então, $x^0 \in EF(P)$.

ii) o algoritmo não termina na primeira vez que o passo 4 é executado.

Neste caso, geramos os antecessores dos elementos de E_3 e determinamos o conjunto E'_2 , tal que:

$\forall x^0 \in E'_2, \nexists x \in D: Cx \geq Cx^0$. Como $\forall x^0 \in E'_2, x^0 \in F$, podemos tomar $E_1 \leftarrow E_1 \cup E'_2$.

Enquanto $E_3 \neq \emptyset$, repete-se o mesmo raciocínio, de terminando sempre o conjunto E'_2 . Num número finito de iterações, teremos $E_3 = \emptyset$, pois no grafo dirigido $G(DUC, \bar{V}_1)$ não existem ciclos e existe um número finito de pontos em DUC .

Quando $E_3 = \emptyset$, recaímos no caso i).

3. OS CONJUNTOS $\bar{V}$ E $\bar{V}_1$.

Daremos a seguir a descrição do procedimento utilizado para a obtenção dos elementos do conjunto $\bar{V}_1$. Como, para seleccionar os vetores de V que pertencem a $\bar{V}$, são utilizados muitos testes, a geração desses elementos será descrita em três etapas. Na etapa 1 será mostrado um esquema de enumeração dos 3^n vetores v , tais que: $v \in R^n$ e $v_j \in \{0, 1, -1\}$, $j \in J$. Na etapa 2, aplicaremos os testes básicos para a escolha dos elementos de $\bar{V}$ e a definição de $\bar{V}$. Na etapa 3, descreveremos dois testes adicionais para apressar a eliminação dos elementos no processo de enumeração implícita.

Etapa 1.

O esquema utilizado para a enumeração dos 3^n vetores citados acima é o da geração de uma árvore cuja raiz é o vetor 0 e os nós v^{ij} de nível i e $j=1, 2, \dots, 2 \times \binom{n}{i}$, são os vetores que tem i componentes não nulas, $i=1, 2, \dots, n$. Um e-

lemento $v^{(i+1)}$ diz-se *descendente* (filho) de um elemento v^i sse $v_j^{(i+1)} = v_j^i$, $j=1, \dots, k < n$ e se existe um único ℓ , $1 \leq \ell \leq n-k$, tal que $v_{j+\ell}^{(i+1)} \neq v_{j+\ell}^i = 0$. O elemento v^i é chamado de *antecedente* (pai) do elemento $v^{(i+1)}$.

Os elementos desta árvore devem ser gerados por nível, a partir do nível 1.

O seguinte exemplo ilustrará o esquema de enumeração para $n=3$.

EXEMPLO 3.1.

Algumas observações podem ser feitas quanto à geração dessa árvore.

- 1) o vetor $v=0$ não satisfaz $Cv \geq 0$. Portanto, não é necessário gerá-lo.
- 2) é conveniente que a geração dos elementos de nível 1 seja feita à parte pelas seguintes razões:
 - como são vetores do tipo $\pm e^i$, $i=1, \dots, n$, não é necessário construí-los explicitamente;
 - para testar se $Ce^i \geq 0$ ($C(-e^i) \geq 0$), basta verificar se a coluna e^i da matriz C é $e^i \geq 0$ ($-e^i \geq 0$).

Etapa 2.

Para descrever esta etapa, necessitamos de um conjunto auxiliar onde serão colocados os vetores que não pertencem ao conjunto $\bar{V}$, mas cujos descendentes podem pertencer a $\bar{V}$. Denotaremos este conjunto por NV .

Para cada v^{ij} que for gerado na etapa 1, devem ser aplicados dois testes para verificar se pertence a $\bar{V}$ ou a NV .

Teste 1 - verificar se $C.v^{ij} \geq 0$

Teste 2 - verificar se existe $v \in \bar{V}$: v envolve v^{ij} .

Assim, teremos os conjuntos:

$$\bar{V} \equiv \{v^{ij} : C v^{ij} \geq 0 \wedge \nexists v \in \bar{V} : v \text{ envolve } v^{ij},$$

$$j=1, \dots, 2 \times \binom{n}{i}, i=1, \dots, n\} \text{ e}$$

$$NV \equiv \{v^{ij} : C v^{ij} \neq 0 \wedge \nexists v \in \bar{V} : v \text{ envolve } v^{ij},$$

$$j=1, \dots, 2 \times \binom{n}{i}, i=1, \dots, n\}.$$

Para efeito de programação do algoritmo no computador, lembremos que:

- 1) é conveniente guardar a posição da última componente não nula de cada elemento de NV , para poder gerar, facilmente, os seus descendentes.
- 2) é conveniente guardar separadamente as posições das componentes $+1$ e -1 para cada elemento de NV , para poder verificar, facilmente, se um elemento *envolve* o outro.
- 3) sendo v_k^{ij} a última componente do vetor v^{ij} que foi definida com $+1$ ou -1 , para verificar se v^{ij} é *envolvido* por algum $v \in \bar{V}$, basta considerar os elementos $v \in \bar{V}$ tais que: $v_k = v_k^{ij}$. Isto se justifica pelo seguinte motivo: se um vetor v tal que $v_k = 0$ *envolver* v^{ij} , então já teria feito com o seu

anterior.

Note-se que nesse esquema de enumeração, se um certo nó v^{ij} passar a pertencer a $\bar{V}$ ou for envolvido por algum elemento de $\bar{V}$ então a subárvore de raiz v^{ij} será descartada.

Suponhamos que na árvore do exemplo 3.1 o nó $(0, -1, 0) \in \bar{V}$. Então, a subárvore a ser gerada será:

Do ponto de vista teórico, essas duas etapas são suficientes para determinar o conjunto $\bar{V}$. Na etapa 3, são realizados testes adicionais para apressar a eliminação de subárvores com raiz no nível mais baixo possível. Antes da des

criação desta etapa, introduziremos nomenclatura e notações para enunciar de forma mais clara as propriedades em que se baseiam os testes mencionados acima.

Diremos que cada vetor v é composto de duas partes: uma parte fixa (v^f) e a outra parte variável (v^v) , onde $v^v = (0, \dots, 0)$. Denotaremos o vetor v por $v = (v^f, v^v)$. Suponhamos que k seja a posição da última componente não nula de v . Então, $(v^v) = (v_1, v_2, \dots, v_k)$ e $(v^v) = (v_{k+1}, \dots, v_n) = (0, \dots, 0)$.

EXEMPLO 3.2.

Seja $v = (1, 0, -1, 0, 0, 0, 0)$

$$(v^f) = (1, 0, -1)$$

$$(v^v) = (0, 0, 0, 0).$$

Sejam:

i) v um vetor candidato para ser elemento do conjunto NV na etapa 2;

ii) $y = Cv$ e $I^-(v) \equiv \{i : 1 \leq i \leq p \wedge y_i < 0\}$;

iii) C_f uma submatriz de C tal que

$C_f = (c_{ij})$, $i \in I^-(v)$ e $j = 1, \dots, k$, onde k é a última componente de (v^f) ;

$$\text{iv) } S_f(v) = e(C_f \cdot v^f)$$

Propriedade 3.1. Seja v tal que $I^-(v) \neq \emptyset$. Se $S_f(v) + \sum_{j=k+1}^n \left| \sum_{i \in I^-(v)} c_{ij} \right| = S(v) < 0$, então, não existe descendente $\bar{v}$ de v , tal que $C\bar{v} \geq 0$.

Calcular $\sum_{j=k+1}^n \left| \sum_{i \in I^-(v)} c_{ij} \right|$, significa definir a parte variável (v^v) com valores $+1$ ou -1 de modo a obter condições mais favoráveis para que um descendente $\bar{v}$ de v possa satisfazer $C\bar{v} \geq 0$.

EXEMPLO 3.3. Seja a seguinte matriz de critérios:

$$C = \begin{bmatrix} -2 & 1 & -3 \\ 5 & 2 & 1 \\ -11 & 4 & 2 \end{bmatrix}$$

Então: $v = (1, 0, 0)$ é tal que $Cv \neq 0$,

pois

$$C_v = \begin{bmatrix} -2 \\ 5 \\ -11 \end{bmatrix}$$

Neste caso:

$$I^-(v) \equiv \{1, 3\}$$

$$S_f = [1, 1] \begin{bmatrix} -2 \\ -11 \end{bmatrix} [1] = -13$$

$$\sum_{j=k+1}^n \left| \sum_{i \in I^-(v)} c_{ij} \right| = |1+4| + |-3+2| = 6$$

$$S(v) = -13+6 < 0$$

Portanto, não existe $\bar{v}$, descendente de v , tal que $C\bar{v} \geq 0$.

Suponha que v^j , $j=1, \dots, q$, sejam os descendentes de v tais que $Cv^j \geq 0$, $j=1, \dots, q$. Sejam $k(1), \dots, k(q)$, respectivamente, a última posição fixada em v^j e $v^j_{k(j)} = \delta_{k(j)}$, onde $\delta_{k(j)} \in \{-1, 1\}$. Seja $v^s \neq v^j$, $j=1, \dots, q$, descendente de v , tal que, $y = Cv^s \neq 0$, $I^-(v^s) \equiv \{i: 1 \leq i \leq p \wedge y_i < 0\}$ e $k(s)$ a última posição fixada em v^s . Seja

$$K^- \equiv \{k(j): (\sum_{i \in I^-(v^s)} c_{ik(j)}) \times \delta_{k(j)} < 0, j=1, \dots, q\}.$$

Propriedade 3.2. Seja v^s um descendente de v , nas condições acima. Se

$$S_f(v^s) + \left| \sum_{i \in I^-(v^s)} c_{ik(j)} \right| + \left| \sum_{i \in I^-(v^s)} c_{il} \right| \equiv S_s < 0,$$

para $k(j) \in K^-$ e $k(s)+1 \leq l \leq n$, $l \neq k(j)$, $j=1, \dots, q$, então, não existe descendente $\bar{v}$ de v^s , tal que $C\bar{v} \geq 0$.

Neste caso, podemos dizer que, para v^s existem algumas posições na sua parte variável que já foram definidas, isto é, os outros descendentes do mesmo antecedente já pertencem a $\bar{V}$ ou foram envolvidos por algum elemento de $\bar{V}$. Portanto, se essas mesmas posições forem definidas com os mesmos valores, este elemento será envolvido. Calcular

$$\left| \sum_{i \in I^-(v^s)} c_{ik(j)} \right| + \left| \sum_{i \in I^-(v^s)} c_{il} \right| \text{ é equivalente a im-}$$

por as condições mais favoráveis para que algum descendente $\bar{v}$ de v^s possa satisfazer $C\bar{v} \geq 0$.

EXEMPLO 3.4. Seja a seguinte matriz de critérios:

$$C = \begin{bmatrix} 2 & -5 & 5 & -10 & 8 \\ -3 & 20 & -25 & 7 & -10 \\ 4 & 2 & 6 & -7 & 5 \end{bmatrix}$$

Sejam: $v = (-1, 1, 0, 0, 0) \Rightarrow Cv \neq 0$

$$v^1 = (-1, 1, 0, 0, 1) \Rightarrow Cv^1 \geq 0$$

$$v^2 = (-1, 1, 0, -1, 0) \Rightarrow Cv^2 \geq 0$$

$$v^3 = (-1, 1, 1, 0, 0) \Rightarrow Cv^3 \neq 0$$

$$I^-(v^3) = \{1, 2\}$$

$$\sum_{i \in I^-(v^3)} c_{i5} \times \delta_5 = -2 \times 1 = -2 < 0$$

$$\sum_{i \in I^-(v^3)} c_{i4} \times \delta_4 = -3 \times (-1) = 3 > 0$$

$$S_f(v^3) = -4$$

$$S_s = -4 + |-2| = -2 < 0$$

Portanto, não existe descendente $\bar{v}$ de v^3 tal que ,
 $C\bar{v} \geq 0$.

Etapa 3.

Se v é tal que $Cv \neq 0$ e satisfaz a propriedade 3.1 ou 3.2, então elimine-o. Em caso contrário, $v \in NV$.

Para problemas com $n \geq 14$, o tempo de processamento na geração de $\bar{V}$ pode chegar a alguns minutos com um programa escrito em ALGOL, no computador B-6700. No entanto, experiências computacionais mostraram que o algoritmo é mais eficiente ao gerarmos apenas um subconjunto qualquer de $\bar{V}$ que chamamos de $\bar{V}_1$. Ao trabalharmos com $\bar{V}_1$ estaremos trabalhando com o grafo dirigido $G(DUC, \bar{V}_1)$ cujos arcos são os elementos de $\bar{V}_1$. Desde que $\bar{V}_1 \subseteq \bar{V}$ o número de vértices isolados e de vértices terminais de $G(DUC, \bar{V}_1)$ tendem a aumentar, afetando as

sim, as partes II e III do algoritmo. Ainda assim, experiências computacionais mostraram que o aumento no tempo de processamento dessas partes não é tão significativo quanto a economia conseguida na parte I (vide tabela 4.1 apresentada no capítulo seguinte).

4. O CONJUNTO D

Se $\bar{V}_1 \equiv \bar{V}$ o conjunto D obtido na parte II do algoritmo será o próprio conjunto $EF(P')$, pois os seus elementos são os vértices do hipercubo unitário *não dominados* em nenhuma das direções v de $\bar{V}$. São os vértices isolados ou os vértices terminais do grafo dirigido $G(DUC, \bar{V})$. Se $\bar{V}_1 \subset \bar{V}$, então $EF(P') \subset D$.

Descreveremos a seguir o procedimento utilizado para a obtenção dos elementos do conjunto D . A idéia inicial é de obter os elementos de DUC através de uma árvore binária em que cada elemento tem descendentes 0 e 1. O exemplo abaixo mostra a subárvore de raiz 0. A subárvore de raiz 1 pode ser obtida de forma semelhante.

EXEMPLO 4.1.

A geração desta árvore pode tornar-se um esquema de enumeração implícita se partirmos de algumas arestas fixadas a priori e analisarmos os descendentes em cada nível.

A regra adotada para fixar essas arestas e o procedimento utilizado para o desenvolvimento dessa árvore binária serão descritos a seguir, em etapas.

Na etapa 1, escolheremos, convenientemente, os vetores de $\bar{V}_1$ para dar origem aos vetores que contribuirão na fi

ção das arestas iniciais.

Na etapa 2, descreveremos o procedimento para obter essas arestas a partir dos vetores selecionados na etapa 1.

Na etapa 3, descreveremos o procedimento para obter os elementos de D através da árvore binária e a partir das arestas já fixadas na etapa 2.

Etapa 1.

DEFINIÇÃO 4.1. Seja $J_1(v) \equiv \{j \in J : v_j \neq 0\}$. Se $(v^1, v^2) \in \bar{V}_1 \times \bar{V}_1$ e $J_1(v^1) \cap J_1(v^2) \equiv \emptyset$, chamamos os vetores v^1 e v^2 de *vetores base*.

Nesta etapa determinamos um conjunto de *vetores base* de $\bar{V}_1$, isto é, determinamos o conjunto:

$$V_{base} \equiv \{v^i \in \bar{V}_1 : \bigcap_{v \in \bar{V}_1} J_1(v) \equiv \emptyset\} \equiv \{v^{bi}, i=1, \dots, nb\}.$$

Os vetores deste conjunto podem ser selecionados na parte I do algoritmo à medida que os elementos de $\bar{V}_1$ vão sendo gerados.

Lembremos que $x \in D$ sse $x+v \notin DUC$, $\forall v \in \bar{V}_1$.

Vamos obter estes vetores x a partir dos *vetores base* da etapa 1. Seja v^{bi} um elemento de V_{base} . Se definirmos x^{i0} tal que:

$$x_j^{i0} = 1 \text{ se } v_j^{bi} = -1$$

$$x_j^{i0} = 0 \text{ se } v_j^{bi} = 1, j \in J_1(v^{bi}),$$

então $x_j^{i0} + v_j^{bi} \in DUC$. Mas qualquer outro x_j^{ik} com $x_j^{ik} \in \{0, 1\}$ para $j \in J_1(v^{bi})$ é tal que $x_j^{ik} + v_j^{bi} \notin DUC, k=1, \dots, 2^{n_1-1}$, onde n_1 é a cardinalidade de $J_1(v^{bi})$. Mostramos abaixo, um exemplo destes elementos.

EXEMPLO 4.2. Indiquemos por $\bullet$ os valores de x_j^{ik} para $j \in J_1(v^{b1})$.

$$\text{Seja: } v^{b1} = (1, 0, -1, 0)$$

$$\text{Então: } x^{10} = (0, \bullet, 1, \bullet), x^{10} + v^{b1} \in DUC$$

$$x^{11} = (0, \bullet, 0, \bullet), x^{11} + v^{b1} \notin DUC$$

$$x^{12} = (1, \bullet, 0, \bullet), x^{12} + v^{b1} \notin DUC$$

$$x^{13} = (1, \bullet, 1, \bullet), x^{13} + v^{b1} \notin DUC$$

Denotaremos por $nr(n_1)$ o número $2^{n_1} - 1$.

Aos vetores $x^{ij}, j=1, 2, \dots, nr(n_1)$, como os obtidos acima, a partir de $v^{bi}, i=1, \dots, nb$, chamamos de radicais obtidos de v^{bi} ou simplesmente radicais quando v^{bi} estiver subentendido.

Sejam $RD(i), i=1, \dots, nb$, os conjuntos de radicais obtidos de cada v^{bi} e $rd(i)$ a cardinalidade de cada $RD(i)$. Cada radical $x^{ij} \in RD(i)$ é tal que, $x^{ij} + v^{bi} \notin DUC, i=1, \dots, nb$ e

$$j=1, \dots, rd(i).$$

Etapa 2.

Nesta etapa, determinamos os conjuntos

$$RD(i) = \{x^{ij}, j=1, \dots, rd(i)\}, i=1, \dots, nb.$$

e

$$RDF \equiv \prod_{i=1}^{nb} RD(i) \text{ ou}$$

$$RDF \equiv \{x^k = (x^{1j(1)}, x^{2j(2)}, \dots, x^{nbj(nb)}),$$

$$x^{ij(i)} \in RD(i), i=1, \dots, nb\}, k=1, \dots, \prod_{i=1}^{nb} rd(i).$$

A cada $x^k \in RDF$ chamamos de *radical final*. Denotaremos por J_f o conjunto dos índices j tais que $j \in \bigcup_{i=1}^{nb} J(v^{bi})$

Determinamos, assim, o conjunto dos radicais finais que constituem as arestas fixas da árvore binária.

Mostramos no exemplo abaixo a obtenção desses conjuntos.

EXEMPLO 4.3. Indiquemos por $\emptyset$ os valores de x_j^{ik} , para $j \notin J_1(v^{bi})$, $i=1, 2$.

$$\text{Seja: } v_{base} \equiv \{v^{b1} = (0, 0, 0, 1), v^{b2} = (1, 0, -1, 0)\}.$$

Então:

$$RD(1) = \{x^{11} = (0, 0, 0, 1)\} \text{ e } rd(1) = 1.$$

$$RD(2) = \{x^{21} = (0, 0, 0, 0), x^{22} = (1, 0, 0, 0), x^{23} = (1, 0, 1, 0)\}$$

$$\text{e } rd(2) = 3.$$

$$RDF \equiv RD(1) \times RD(2) = \{(x^{11}, x^{21}), (x^{11}, x^{22}), (x^{11}, x^{23})\}$$

$$RDF \equiv \{(0, 0, 0, 1), (1, 0, 0, 1), (1, 0, 1, 1)\} \text{ e } J_f \equiv \{1, 3, 4\}.$$

Obtido o conjunto RDF , como os radicais x^k tem algumas posições com valores já fixados podemos verificar se $x^k + v \notin DUC$, $\forall v \in \bar{V}_1$. Se isto acontecer,

$$\forall x: x_j = x_j^k, \text{ para } j \in J_f \text{ e } x_\ell \in \{0, 1\}, \forall \ell \notin J_f, x \in D.$$

A cada elemento obtido a partir de x^k chamamos de *descendente do radical* x^k ou simplesmente *descendente* quando o radical x^k estiver subentendido.

Etapa 3.

Consideremos um elemento $x^k \in RDF$. Se $x^k + v \notin DUC, \forall v \in \bar{V}_1$, então todos os seus descendentes são elementos de D .

Seja $V_1(x^k) \equiv \{v \in \bar{V}_1 : x^k + v \in DUC, x^k \in RDF\}$. Se $\exists v \in V_1(x^k) : v_j = 0, \forall j \notin J_f$, então descarte este radical x^k , pois

$x^k + v \in DUC$, $\forall x_j^k \in \{0, 1\}$, $j \notin J_f$. Em caso contrário, definimos mais uma componente $x_{j_0}^k$, $j_0 \notin J_f$, $x_{j_0}^k \in \{0, 1\}$. Se $(x^k + v) \notin DUC$, $\forall v \in V_1(x^k)$, então, todos os seus descendentes são elementos de D . Se ainda $\exists v \in V_1(x^k) : (x^k + v) \in DUC$, fazemos $V_1(x^k) \leftarrow V_1(x^k) \setminus \{v \in V_1(x^k) : (x^k + v) \notin DUC\}$. Definimos mais uma componente $x_{j_1}^k$, $j_1 \notin J_f$, $j_1 \neq j_0$ e $x_{j_1}^k \in \{0, 1\}$. Assim, fixamos duas componentes $(x_{j_0}^k, x_{j_1}^k) \in \{0, 1\} \times \{0, 1\}$. Repetimos este processo até que $V_1(x^k) \equiv \emptyset$ e todos os descendentes de x^k são elementos de D . Se todas as componentes x_j^k , $j \notin J_f$ já foram definidas e $V_1(x^k) \neq \emptyset$ então eliminamos esse radical.

Repetindo-se esse processo para todo $x^k \in RDF$, obtemos o conjunto D :

$$D \equiv \{x \in DUC : x + v \notin DUC, \forall v \in \bar{V}_1\}$$

5. O CONJUNTO $EF(P)$.

A obtenção do conjunto $EF(P)$ a partir do conjunto D obtido na parte II do algoritmo baseia-se, principalmente, em:

- a) fazer testes de *dominância* entre os elementos de D e seus antecessores;
- b) determinar os *antecessores* dos vértices no grafo

dirigido $G(DUC, \bar{V}_1)$;

c) fazer testes de *viabilidade* dos elementos de DUC .

5.1. DOMINÂNCIA ENTRE OS ELEMENTOS DE DOIS CONJUNTOS

Sejam dois conjuntos E e H , tal que:

$$E \equiv \{x^i \in R^n : eCx^i \geq eCx^{i+1}, i=1, 2, \dots, ne\}$$

Admita-se que queiramos determinar o subconjunto de elementos de H que não são *dominados* pelos elementos de E . Seja S este subconjunto.

Descreveremos a seguir, por passos, o procedimento utilizado para a obtenção de S .

Passo 1 - Tome um elemento y^j de H . Seja $y^j = y^1$.

Passo 2 - Tome um elemento x^i de E . Seja $x^i = x^1$.

Passo 3 - Se $eCx^i \geq eCy^j$ e $Cx^i \neq Cy^j$, então tome outro $x^i \in E$, se existir, e volte para o passo 3. Se não existir, $y^j \in S$.

Passo 4 - Tome outro $y^j \in H$, não testado, se existir, e volte para o passo 2. Se não existir, pare.

5.2. DOMINÂNCIA ENTRE OS ELEMENTOS DO PRÓPRIO CONJUNTO

Suponha que queremos testar a *dominância* entre os elementos do conjunto,

$$E \equiv \{x^i, i=1, 2, \dots, ne\},$$

que é particionado em ne subconjuntos unitários E_i tal que $E_i = \{x^i\}$, $i=1, 2, \dots, ne$. Então, $E \equiv \bigcup_{i=1}^{ne} E_i \equiv \bigcup_{i=1}^{ne} \{x^i\}$

A dominância será testada entre os elementos de cada dois subconjuntos E_i e E_{i+1} , conforme o procedimento descrito abaixo:

Passo 1 -

Faça: $i+1; j+1;$

então
se $Cx^i \geq Cx^{i+1}$

então x^{i+1} é domi-
nado por x^i
e é elimina-
do. Coloque
 x^i em S_j ;

senão x^{i+1} não é do-
minado por
 x^i e vice-
versa.
Coloque x^i
e x^{i+1} , nes-
ta ordem, em
 S_j ;

Passo 2 -

Se $eCx^i \geq eCx^{i+1}$

senão
se $Cx^{i+1} \geq Cx^i$

então x^i é domina-
do por x^{i+1}
e é elimina-
do.
Coloque x^{i+1}
em S_j ;

senão x^i não é do-
minado por
 x^{i+1} e vi-
ce-versa.
Coloque x^{i+1}
e x^i , nessa
ordem, em S_j ;

Passo 3 -

Se ainda existem pelo menos dois subconjuntos a serem analisados, então, faça: $i+i+2$; $j+j+1$ e volte para o passo 2 (quando os subconjuntos não são unitários, o incremento da- do ao i deve ser modificado de forma conveniente). Se exis- tir apenas um subconjunto a ser analisado faça: $j+j+1$; $i+i+2$, coloque x^i em S_j e pare. Se não existe mais nenhum subconjun- to, pare.

Oteve-se, assim, um subconjunto S de E , particiona- do em $\lceil ne/2 \rceil$ subconjuntos S_j de um elemento, x^i ou x^{i+1} , ou de dois elementos x^i e x^{i+1} dispostos em ordem decrescente segundo eCx^i e eCx^{i+1} . Então, S poderia ser, por exemplo:

$$S \equiv \{x^1\} \cup \{x^3, x^4\} \cup \{x^6\} \cup \dots \cup \{x^{ne}\} ,$$

dependendo da dominância entre os elementos de E_i e E_{i+1} .

Se repetirmos esse procedimento, testando a dominân- cia entre os elementos de S_i e S_{i+1} , obteremos um subconjun- to de E particionado em subconjuntos de um a quatro elemen- tos. Ao repetirmos esse procedimento por $\lceil ne/2 \rceil$ vezes, obte- remos um único subconjunto de E que será o subconjunto de e- lementos não dominados pelos elementos do próprio E .

5.3. ANTECESSORES NO GRAFO DIRIGIDO $G(DUC, \bar{V}_1)$

Dado um vértice $x \in G(DUC, \bar{V}_1)$, determinar o seu *antecessor* significa determinar um outro vértice y , tal que $y = x - v$, $v \in \bar{V}_1$ e $y \in DUC$.

O procedimento utilizado para obter estes antecessores será discutido para cada tipo de problemas apresentado no capítulo seguinte.

5.4. VIABILIDADE DOS ELEMENTOS DE DUC

Para verificar se um elemento x é viável ou não, isto é, se $Ax \leq b$, basta fazer adição de colunas a^j de A , $j \in J_1$, $J_1 \equiv \{j \in J : x_j = 1\}$ e verificar se $\sum_{j \in J_1} a^j \leq b$.

OBSERVAÇÃO.

Mais detalhes sobre programação do algoritmo em ALGOL do B-6700, podem ser encontrados no apêndice 1.

CAPÍTULO IV

APLICAÇÃO DO ALGORITMO PARA DIFERENTES MATRIZES DE CRITÉRIOS C

1. INTRODUÇÃO

Neste capítulo, apresentaremos o algoritmo para diferentes matrizes de critérios C . Em cada caso particular, devido à estrutura da matriz de critérios, introduzimos algumas simplificações para melhorar a eficiência computacional. Apresentaremos, também, os resultados dos testes computacionais, obtidos com a aplicação do algoritmo para cada um destes casos.

Nas experiências computacionais os elementos da matriz de critérios C e da matriz A (todos não negativos) foram gerados aleatoriamente no intervalo $[0,100]$. Cada elemento do vetor b foi calculado como uma fração da soma dos elementos da linha correspondente em A , isto é, cada

$$b_i = \alpha \sum_{j=1}^n a_{ij}, \quad i=1, \dots, m.$$

Foram utilizados os seguintes valores para α : 0.75, 0.50, 0.25.

Na seção 2, analisaremos o caso em que a matriz de critérios C tem todos os elementos não negativos, com pelo

menos um elemento positivo em cada coluna*. Denominaremos uma matriz com esta característica, de matriz de critérios C não negativa e denotaremos por $C \geq 0$. Esta é a hipótese sob a qual conseguimos introduzir mais simplificações ** no algoritmo, as quais influíram significativamente no tempo de processamento computacional, como podem ser observados nos resultados dos testes apresentados.

Na seção 3, apresentaremos mais alguns resultados teóricos que permitem analisar os problemas com a matriz de critérios $C \geq 0$, num conjunto de pontos viáveis F fixo. Neste caso, ainda, o algoritmo simplificado da seção anterior pode ser aplicado.

Na seção 4, apresentaremos o caso em que a matriz de critérios C tem uma linha não positiva e todas as outras não negativas. Denominaremos uma matriz com esta característica de matriz de critérios C com uma linha não positiva. Neste caso, não obtivemos nenhuma simplificação, no algoritmo em si, mas, notamos que existe um subconjunto $\bar{V}_1 \subseteq \bar{V}$, com um número de elementos que pode ser considerado ótimo em função do tempo total de processamento.

Na seção 5, apresentaremos o caso em que a matriz de

* vide observação 2) no final da seção 2.

** convém notar que, quando $C \geq 0$, podem ser introduzidas simplificações semelhantes. Vide observação 1) no final da seção 2.

critérios C tem uma certa porcentagem de elementos negativos. Nas experiências computacionais usamos aproximadamente 20% de elementos negativos. Denotaremos por $C_{20\%}$, uma matriz com esta característica. Neste caso, podemos estudar o problema decompondo a matriz de critérios C em $C=[C_1, C_2, C_3]$, sendo C_1 uma matriz $p \times n_1$, não negativa, C_2 uma matriz $p \times n_2$, não positiva e C_3 uma matriz $p \times (n - n_1 - n_2)$ com elementos positivos, negativos ou nulos. É fácil ver que, com esta decomposição, as matrizes C_1 e C_2 são matrizes com características apresentadas na seção 2, para as quais obtivemos muitas simplificações no algoritmo. Estas simplificações permitiram que utilizássemos apenas a matriz C_3 , de dimensão menor que a da matriz C , nas duas primeiras partes do algoritmo o que o tornou, computacionalmente, mais eficiente.

Na seção 6, apresentaremos os resultados de alguns testes computacionais para problemas com a matriz de critérios $C_{20\%}$, variando-se os parâmetros p e m . O objetivo destes testes é observar a variação do tempo total gasto pelo algoritmo em função do número de funções-objetivo e do número de restrições associadas ao problema.

Na seção 7, analisaremos os resultados computacionais apresentados nas seções anteriores.

2. O ALGORITMO SIMPLIFICADO PARA A MATRIZ DE CRITÉRIOS C NÃO NEGATIVA.

A principal simplificação conseguida no algoritmo é devida ao seguinte lema:

LEMA 2.1. / 2 /. Se $C \geq 0$ os únicos n vetores de $\bar{V}$ necessários para construir $EF(P') \equiv DUC - \sum_{v \in \bar{V}} M(v)$ são $v^i = e^i$, $i \in J$.

Note-se que com este resultado a parte I do algoritmo pode ser eliminada, pois $\bar{V}_1 \equiv \bar{V} \equiv \{e^i\}_{i=1}^n$. Com este conjunto $\bar{V}_1$ podemos facilmente verificar que no grafo dirigido $G(DUC, \bar{V}_1)$ não existem vértices isolados e existe um único vértice terminal $e = (1, 1, \dots, 1)$. Portanto, a parte II do algoritmo também pode ser eliminada e o conjunto $D \equiv \{e = (1, 1, \dots, 1)\}$.

A figura 2.1 mostra para $n=4$, um subgrafo de $G(DUC, \bar{V}_1)$ que é suficiente e conveniente para gerar os antecessores. Cada algarismo do número dado a cada vértice representa a posição j , $1 \leq j \leq n$, tal que $x_j = 0$, para $x \in DUC$.

Figura 2.1.

A parte III do algoritmo consiste, então, em analisar este tipo de subgrafo que também é uma *árvore*. A cada vez que o passo 4 do algoritmo é executado, os nós de um nível são gerados. É fácil ver que, se o nó 0 da árvore acima for viável, isto é, $(1,1,1,1) \in F$ então $EF(P) \equiv \{(1,1,1,1)\}$. No caso geral, $EF(P) \equiv \{e=(1,\dots,1)\}$. Portanto, no algoritmo pode-

mos considerar o conjunto $E_1 \equiv \emptyset$.

Apresentaremos a seguir o algoritmo simplificado. Conservaremos as diferentes partes e os números referentes aos passos da parte III da descrição original do capítulo anterior, para futuras referências. Os passos que não forem mencionados não serão necessários neste caso particular.

Parte I: $\bar{V}_1 \equiv \{e^i\}_{i=1}^n$

Parte II: $D \equiv \{e = (1, 1, \dots, 1)\}$

Parte III: seja $E_2 \equiv \emptyset$

Passo 1: se $e \in F$, então $EF(P) \leftarrow \{e\}$. Pare.

Em caso contrário, $e \in E_3$.

Passo 4: se $E_3 \equiv \emptyset$, então $EF(P) \leftarrow E_2$. Pare.

Em caso contrário, determine:

$$E_4 \equiv \{y : y = z - v \in F, v \in \bar{V}_1, z \in E_3\}$$

e

$$E_5 \equiv \{y' : y' = z - v \in DUC \setminus F, v \in \bar{V}_1, z \in E_3\}$$

Passo 5: determine:

$$E_6 \equiv \{y : y \in E_4 \wedge \nexists w \in E_2 \cup E_4 : Cw \geq Cy\}$$

Passo 6: determine:

$$E_2 \equiv \{x : x \in E_2 \wedge \nexists y \in E_6 : Cy \geq Cx\} \quad e$$

faça $E_2 \leftarrow E_2 \cup E_6$.

Passo 7: determine:

$$E_3 \equiv \{y' : y' \in E_5 \wedge \nexists w \in E_2 : Cw \geq Cy'\} \quad e$$

volte para o passo 4.

Seja $z \in E_3$. No passo 4 do algoritmo, para determinarmos um *antecessor* de z , estamos determinando $y \in DUC$, tal que $y = z - e^i$, $i \in J$. Analisando a árvore da figura 2.1, podemos notar que, para obtermos y a partir do vértice z , basta fazermos uma componente $y_{k+l} = 0$, para $1 \leq l \leq n-k$, sendo k o índice da última componente nula de z . Com isto queremos dizer que no passo 4, o antecessor y pode ser obtido sem efetuar explicitamente a diferença $z - e^i$, $i \in J$.

Da mesma forma, verificar se $Ay \leq b$, para $y = z - e^i$, $i \in J$, também torna-se mais fácil, pois se o produto Az já foi calculado, basta subtrairmos deste produto a coluna a^i da matriz A , correspondente à componente e_i^i do vetor e^i .

Quanto aos testes de dominância não há nada a simplificar.

Os resultados dos testes computacionais para problemas com $C \geq 0$, utilizando o algoritmo acima, são mostrados nas tabelas 2.1 a 2.3, no final desta seção.

Podemos ainda encontrar o seguinte problema. Como $D \equiv \{e = (1, 1, \dots, 1)\}$, se o primeiro ponto viável tiver poucas componentes iguais a 1, então o passo 4 do algoritmo dever ser executado várias vezes até encontrar o primeiro ponto viável. Para apressarmos este processo, testamos o seguinte procedimento:

resolver um problema auxiliar de programação matemática linear:

$$z = \max \{ex : Ax \leq b, x \leq 1 \wedge x \geq 0\} ,$$

pelo método *simplex revisado* e determinar o conjunto dos vértices de DUC com $\lfloor z \rfloor$ componentes iguais a 1. Seja $\bar{D}$ este conjunto.

No algoritmo, substituímos D por $\bar{D} \cup (D + \bar{D})$ e as seguintes modificações são introduzidas na parte III do algoritmo.

Passo 1: determine:

$$E_2 \equiv \{x : x \in \bar{D} \cap F \wedge \nexists y \in \bar{D} \cap F : Cy \geq Cx\}$$

Passo 3: determine:

$$E_3 \equiv \{z : z \in \bar{D} \setminus F \wedge \nexists x \in E_2 : Cx \geq Cz\}$$

Os passos seguintes continuam sem nenhuma alteração.

Experiências computacionais mostraram que a resolução desse problema auxiliar é útil somente no caso de $b_i = 0.25 \cdot \sum_{j=1}^n a_{ij}$, $i=1, \dots, m$, conforme os resultados apresentados nas tabelas 2.4, 2.5 e 2.6, comparados, respectivamente, com os das tabelas 2.1, 2.2 e 2.3. No problema 12, $n=10$, o problema auxiliar não se mostrou útil.

OBSERVAÇÕES

- 1) Quando a matriz de critérios C tem todos os elementos não positivos, com pelo menos um elemento negativo em cada coluna, podemos introduzir simplificações semelhantes ao caso em que $C \geq 0$, lembrando que, $\bar{V}_1 = \{-e^i\}_{i=1}^n$ e o conjunto $D \equiv \{(0, 0, \dots, 0)\}$.
- 2) Quando uma coluna c^k da matriz de critérios C tem todos os elementos nulos, o algoritmo pode ser aplicado. Porém, após ter obtido o conjunto solução, se algum ponto x com a componente $x_k = 1$ for solução, então o ponto x' tal que $x'_k = 0$ e $x'_j = x_j$, $\forall j \neq k$, $1 \leq j \leq n$ também será um ponto eficiente, se todos os elementos da matriz A são não negativos. Em caso contrário, deverá ser testada a viabilidade para o ponto x' .

Tabela 2.1

$n=10$, $m=4$, $p=3$, $C \geq 0$, sem *Simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	Nº de elementos em $EF(P)$	Tempo Total (segs)
1	.75	7	8	.64
2	.75	7	10	.84
3	.75	7	7	.91
4	.75	7	8	.68
5	.75	7	10	.82
6	.50	5	9	1.53
7	.50	5	11	1.99
8	.50	5	7	1.53
9	.50	5	7	1.59
10	.50	5	11	1.64
11	.25	4	1	1.88
12	.25	3	2	1.92
13	.25	3	3	2.01
14	.25	2	3	2.14
15	.25	2	5	2.13

Tabela 2.2

$n=14$, $m=4$, $p=3$, $C \geq 0$, sem *simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	Nº de elementos em $EF(P)$	Tempo Total (segs)
16	.75	10	20	5.93
17	.75	11	19	3.67
18	.75	10	14	4.18
19	.75	11	24	6.85
20	.75	11	21	3.33
21	.50	8	28	20.09
22	.50	8	9	20.20
23	.50	8	23	19.75
24	.50	7	20	20.41
25	.50	8	12	17.63
26	.25	4	16	28.29
27	.25	4	14	27.87
28	.25	4	13	28.08
29	.25	4	10	28.13
30	.25	4	10	28.14

Tabela 2.3

$n=18$, $m=4$, $p=3$, $C \geq 0$, sem *simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	Nº de elementos em $EF(P)$	Tempo Total (segs)
31	.75	14	53	39.39
32	.75	14	52	25.85
33	.75	14	87	63.36
34	.75	14	15	10.66
35	.75	14	19	44.25
36	.50	10	60	339.07
37	.50	10	20	236.61
38	.50	10	52	285.49
39	.50	10	23	212.47
40	.50	10	44	251.88
41	.25	Somente 1 destes problemas foi processado <i>sem simplex</i> pois o tempo total gasto para este problema foi bem maior (~500 segs) que o gasto <i>com simplex</i> (Tabela 2.6)		
42	.25			
43	.25			
44	.25			
45	.25			

Tabela 2.4

$n=10$, $m=4$, $p=3$, $C \geq 0$, com *simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	$[z]$ obtido pelo Simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
1	.75	7	7	8	1.90
2	.75	7	8	10	2.06
3	.75	7	7	7	2.02
4	.75	7	7	8	1.90
5	.75	7	7	10	2.12
6	.50	5	5	9	2.14
7	.50	5	5	11	2.29
8	.50	5	6	7	2.49
9	.50	5	5	7	2.11
10	.50	5	5	11	2.29
11	.25	4	4	1	1.90
12	.25	3	3	2	1.37
13	.25	3	3	3	1.03
14	.25	2	2	3	1.04
15	.25	2	3	5	1.29

Tabela 2.5

$n=14$, $m=4$, $p=3$, $C \geq 0$, com *Simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	$[z]$ obtido pelo Simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
16	.75	10	11	20	10.13
17	.75	11	11	19	7.46
18	.75	10	11	14	8.16
19	.75	11	11	24	10.91
20	.75	11	11	21	7.46
21	.50	8	8	28	23.99
22	.50	8	8	9	23.19
23	.50	8	8	23	23.37
24	.50	7	8	20	24.06
25	.50	8	8	12	22.41
26	.25	4	4	16	5.93
27	.25	4	4	14	5.73
28	.25	4	4	13	5.69
29	.25	4	4	10	6.15
30	.25	4	4	10	5.80

Tabela 2.6

$n=18$, $m=4$, $p=3$, $C \geq 0$, com *Simplex*

Nº do Probl.	Valor de α	Nº de l's do 1º pto. viável	$ z $ obtido pelo Simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
31	.75	14	14	53	57.2
32	.75	14	14	52	48.2
33	.75	14	14	87	85.2
34	.75	14	14	15	31.5
35	.75	14	14	19	65.1
36	.50	10	10	60	384.5
37	.50	10	10	20	287.9
38	.50	10	10	52	333.7
39	.50	10	10	23	273.1
40	.50	10	10	44	310.8
41	.25	5	6	19	83.8
42	.25	5	5	25	36.4
43	.25	5	5	21	34.5
44	.25	5	6	6	83.3
45	.25	5	5	12	34.5

3. PROBLEMAS COM A MATRIZ DE CRITÉRIOS $C \geq 0$ COM O CONJUNTO F FIXO.

Os seguintes resultados teóricos serão úteis quando o problema:

$$(P): \max \{Cx : x \in F \wedge C \geq 0\},$$

deve ser resolvido para um conjunto fixo F e diferentes matrizes de critérios $C \geq 0$. Este é o caso em que se deseja analisar a variação do conjunto de pontos eficientes quando se altera a matriz de critérios para o mesmo conjunto F .

PROPOSIÇÃO 3.1. Seja o problema

$(PI): \max \{I_n x : x \in F\}$, onde I_n é a matriz identidade de ordem n . Sejam, $EF(PI)$ o conjunto de pontos eficientes em (PI) e $EF(P)$ o conjunto de pontos eficientes em (P) . Então $EF(P) \subseteq EF(PI)$.

Demonstração. Suponha que $x^1 \in F$ e $x^1 \notin EF(PI)$. Então, $\exists x^2 \in F : I_n x^2 \geq I_n x^1 \Rightarrow x^2 \geq x^1 \Rightarrow Cx^2 \geq Cx^1 \Rightarrow x^1 \notin EF(P)$.

Portanto, $EF(P) \subseteq EF(PI)$. $\square$

Esta proposição nos mostra que, qualquer que seja a matriz de critérios $C \geq 0$, desde que o conjunto F seja fixo, o conjunto de pontos eficientes em $P, EF(P)$, pode ser obtido a partir de $EF(PI)$.

COROLÁRIO 3.1.1. Se $x^1 \in EF(PI)$ e $x^1 \notin EF(P)$, então existe $x^2 \in EF(PI)$, tal que $Cx^2 \geq Cx^1$.

Demonstração. Se $x^1 \notin EF(P) \Rightarrow \exists x^2 \in EF(P) : Cx^2 \geq Cx^1$. Pela proposição anterior, $x^2 \in EF(PI)$. $\square$

Deste resultado concluímos que podemos obter $EF(P)$, testando a dominância em relação à matriz $C \geq 0$ entre os elementos de $EF(PI)$.

Então, o algoritmo simplificado para $C \geq 0$ pode ser aplicado da seguinte forma:

Parte I: $\bar{V}_1 \equiv \{e^i\}_{i=1}^n$

Parte II: $D \equiv \{e = (1, 1, \dots, 1)\}$

Parte III: seja $E_2 \equiv \emptyset$

Passo 1: se $e \in F$, então $EF(P) + \{e\}$. Pare.

Em caso contrário, $e \in E_3$.

Passo 4: se $E_3 \equiv \emptyset$, vá para o passo 8. Em caso contrário, determine:

$$E_4 \equiv \{y : y = z - v \in F, v \in \bar{V}_1, z \in E_3\}$$

e

$$E_5 \equiv \{y' : y' = z - v \in DUC \setminus F, v \in \bar{V}_1, z \in E_3\}$$

Passo 6: faça: $E_2 \leftarrow E_2 \cup E_4$

Passo 7: faça: $E_3 \leftarrow E_5$ e volte para o passo 4.

Passo 8: determine:

$$EF(P) \equiv \{x : x \in E_2 \wedge \nexists y \in E_2 : Cy \geq Cx\} \text{ e pare.}$$

4. MATRIZ DE CRITÉRIOS C COM UMA LINHA NÃO POSITIVA

Quando a matriz de critérios C tem uma linha de elementos não positivos, nenhuma simplificação no algoritmo pôde ser introduzida.

Para este tipo de matriz de critérios, experiências computacionais mostraram que o tempo gasto para gerar todos os elementos de $\bar{V}$ pode chegar a alguns minutos (programa em ALGOL do B.6700) para problemas de dimensão relativamente pequena. Esta dificuldade nos levou a realizar vários testes com o objetivo de reduzir o tempo de processamento da parte I do algoritmo.

Vários testes foram realizados classificando as colunas da matriz C em diferentes ordens. Assim, as colunas da

matriz C foram dispostas em ordem crescente segundo a soma $\sum_{i=1}^p c_{ij}$, $j = 1, \dots, n$, em ordem decrescente segundo a mesma soma e em ordem decrescente segundo os elementos da linha não positiva. Porém, o melhor resultado foi obtido com as colunas da matriz C dispostas em ordem crescente segundo os elementos da linha não positiva.

Como os resultados ainda não eram satisfatórios, surgiu a idéia de desenvolver o algoritmo com um subconjunto de $\bar{V}$, o conjunto $\bar{V}_1$. Entretanto, uma questão ainda continua em aberto: "qual é o número ótimo de elementos para o conjunto $\bar{V}_1$, em função do tempo total de processamento?".

Foram testados quatro problemas, dois com $n=14$ e dois com $n=18$, fixando-se diferentes valores para a cardinalidade de $\bar{V}_1$, que será indicada por $\text{card}(\bar{V}_1)$. Cada elemento do vetor b foi obtido fazendo-se $b_i = 0.75 \sum_{j=1}^n a_{ij}$, $i=1, \dots, m$, para todos os problemas. Os resultados apresentados na tabela 4.1 sugerem que o tempo total do algoritmo é uma função convexa de $\text{card}(\bar{V}_1)$. Nos problemas testados, podemos notar que o mínimo foi atingido em torno de $\text{card}(\bar{V}_1)=30$, para $n=14$ e $\text{card}(\bar{V}_1)=45$ para $n=18$.

Com o algoritmo apresentado no capítulo anterior, fixando-se $\text{card}(\bar{V}_1)=30$, foram resolvidos quinze problemas para $n=10$ e quinze para $n=14$. Os resultados são mostrados nas tabelas 4.2 e 4.3.

Quando estes mesmos problemas foram resolvidos sem fixar $\text{card}(\bar{V}_1)$, com exceção do problema 51, cuja $\text{card}(\bar{V}_1)=29$, o tempo total de processamento em cada problema era bem maior que os apresentados nessas tabelas.

Para $n=10$, experiências computacionais com variação da cardinalidade de $\bar{V}_1$, talvez sugerissem um outro valor diferente de 30 que nos levasse a melhores resultados que os apresentados na tabela 4.2.

5. MATRIZ DE CRITÉRIOS C COM UMA DETERMINADA PORCENTAGEM DE ELEMENTOS NEGATIVOS

Quando a matriz C tem uma determinada porcentagem de elementos negativos, existe a possibilidade de aparecerem colunas sô de elementos não negativos, com pelo menos um elemento positivo em cada coluna ou colunas sô de elementos não positivos com pelo menos um elemento negativo em cada coluna. Denotaremos por C_1 e C_2 , respectivamente, as submatrizes formadas pelas colunas acima mencionadas. Denotaremos por C_3 a submatriz formada pelas colunas de C que não pertencem a C_1 ou C_2 .

Podemos introduzir algumas simplificações no algoritmo, generalizando o resultado do lema 2.1, sobre a matriz de critérios C com todos os elementos não negativos.

Tabela 4.1

$$b_i = 0.75, \sum_{j=1}^n a_{ij}, i=1, \dots, m$$

Problema	card($\bar{V}_1$)	card (D)	Tempo Total (secs)
$n=14$ $m=4$ $p=3$ card EF(P)=169	120	217	144.5
	90	276	101.8
	60	548	43.6
	45	773	42.8
	30	802	37.9
	15	2080	46.9
	5	6144	92.5
$n=14$ $m=4$ $p=3$ card EF(P)=180	120	240	148.2
	90	346	113.7
	60	568	86.9
	45	706	46.2
	30	908	43.7
	15	1356	41.5
	5	5376	72.8
$n=18$ $m=4$ $p=3$ card EF(P)=298	120	888	400.5
	90	1311	332.6
	60	1990	270.6
	45	2970	154.9
	30	18880	443.7
	15		>500
	5		>600
$n=18$ $m=4$ $p=3$ card EF(P)=182	120	349	496.2
	90	463	418.0
	60	873	206.0
	45	2809	223.5
	30	13600	334.7
	15		>500
	5		>600

Tabela 4.2

$n=10$, $m=4$, $p=3$, C com uma linha não-positiva

Nº do Probl.	Valor de α	$\text{card}(\bar{V}_1)$	$\text{card}(D)$	Nº de elementos em $EF(P)$	Tempo Total (segs)
46	.75	30	128	70	7.5
47	.75	30	78	51	7.5
48	.75	30	119	69	5.9
49	.75	30	124	83	9.6
50	.75	30	120	86	9.1
51	.50	29	184	65	34.1
52	.50	30	142	60	9.8
53	.50	30	103	50	12.9
54	.50	30	76	28	9.7
55	.50	30	81	25	10.0
56	.25	30	136	9	10.3
57	.25	30	76	6	9.4
58	.25	30	84	14	13.9
59	.25	30	102	17	11.0
60	.25	30	93	11	10.5

Tabela 4.3

$n=14, m=4, p=3, C$ com uma linha não-positiva

Nº do Probl.	Valor de α	$\text{card}(\bar{V}_1)$	$\text{card}(D)$	Nº de elementos em $EF(P)$	Tempo Total (seg's)
61	.75	30	571	142	20.3
62	.75	30	734	269	84.2
63	.75	30	656	159	23.6
64	.75	30	476	142	22.0
65	.75	30	486	164	42.8
66	.50	30	1288	139	107.5
67	.50	30	836	117	163.1
68	.50	30	428	69	129.4
69	.50	30	1607	161	132.5
70	.50	30	612	80	91.0
71	.25	30	788	16	123.4
72	.25	30	444	18	133.8
73	.25	30	1372	48	128.6
74	.25	30	994	32	121.6
75	.25	30	432	24	127.1

PROPOSIÇÃO 5.1. Seja $C=[C_1, C_2, C_3]$, onde C_1 é uma submatriz $p \times n_1$, C_2 uma submatriz $p \times n_2$ e C_3 uma submatriz $p \times (n-n_1-n_2)$, como definidos acima. Então

$\bar{V} \equiv \{e^i\}_{i=1}^{n_1} \cup \{-e^i\}_{i=n_1+1}^{n_1+n_2} \cup \bar{V}(C_3)$, onde $\bar{V}(C_3)$ é o conjunto de vetores envolventes do conjunto

$$V \equiv \{v=(0, \bar{v}), 0 \in R^{n_1+n_2}, \bar{v} \in R^{n-n_1-n_2}: \bar{v}_j \in \{0, 1, -1\},$$

$$j=1, \dots, n-n_1-n_2 \text{ e } C_3 \bar{v} \geq 0\}$$

Demonstração. É fácil ver que

$$\{e^i\}_{i=1}^{n_1} \cup \{-e^i\}_{i=n_1+1}^{n_1+n_2} \subset \bar{V}.$$

Seja $v' \in \bar{V}$, $v' \neq e^i$ e $v' \neq -e^i$.

Se $v'_j = 1$, $1 \leq j \leq n_1$, então $\exists e^j \in \bar{V}: e^j$ envolve v' .

Se $v'_j = -1$, $1 \leq j \leq n_1$, então $v'': v''_j = 0$, para $1 \leq j \leq n_1$ e

$$v'' = v'_j, \text{ para } n_1+1 \leq j \leq n, \quad v'' \in V \text{ e } v'' \text{ envolve } v'.$$

De forma análoga, $v'_j \notin \{1, -1\}$ para $n_1+1 \leq j \leq n_1+n_2$.

Portanto, $\forall v' \in \bar{V}$ e $v' \notin \{e^i\}_{i=1}^{n_1} \cup \{-e^i\}_{i=n_1+1}^{n_1+n_2}$ é

tal que: $v'_j = 0, \forall j, 1 \leq j \leq n_1+n_2 \Rightarrow$ as $(n-n_1-n_2)$ componentes de v' definem um elemento de $\bar{V}(C_3)$.

Logo, $\forall v: v \in (0, \bar{v}), 0 \in R^{n_1+n_2}$ e $\bar{v} \in \bar{V}(C_3) \Rightarrow v \in \bar{V}$, pois não é envolvido por nenhum elemento de $\bar{V} \setminus \bar{V}(C_3)$. $\square$

Com a aplicação do resultado acima ao algoritmo, a parte I se resume em determinar $\bar{V}_1(C_3)$, isto é, podemos trabalhar com a matriz C_3 cujo número de colunas é $(n-n_1-n_2) \leq n$. Na parte II também é suficiente trabalharmos com $\bar{V}_1(C_3)$ e de terminarmos o conjunto que denominaremos de D' , cujos elementos pertencem a $R^{n-n_1-n_2}$, substituindo o conjunto D . Como na parte III, temos que trabalhar com a matriz C original, para obtermos os elementos x de D , $x \in R^n$, basta definirmos, para cada elemento $x' \in D'$, $x_j = 1, \forall j, 1 \leq j \leq n_1, x_j = 0, \forall j, n_1+1 \leq j \leq n_1+n_2$ e $x_j = x'_j, \forall j, n_1+n_2+1 \leq j \leq n$.

Para a obtenção do conjunto $\bar{V}_1(C_3)$, foram feitos testes, colocando-se as colunas da matriz C_3 em ordem crescente e em ordem decrescente segundo a soma $\sum_{i=1}^p c_{ij}, j=1, \dots, n$, obtendo-se melhores resultados com a ordem decrescente.

Mesmo com a matriz de critérios decomposto em $C=[C_1, C_2, C_3]$ e ordenando as colunas da matriz C_3 em ordem de crescente segundo a soma acima, o algoritmo não se mostrou satisfatório para determinar $\bar{V}_1(C_3)$, no caso em que $n=18$ e $b_i = 0.50, \sum_{j=1}^n a_{ij}, i=1, \dots, m$. Para alguns problemas testados,

900 segundos de processamento (programa em ALGOL do B-6700) não foram suficientes para determinar $EF(P)$. A razão do mal desempenho é que normalmente o número de componentes iguais a 1 nos elementos do conjunto D é bem maior que o número máximo de 1's de um ponto viável. Como consequência, o passo 4 do algoritmo é executado várias vezes, até encontrar o primeiro ponto viável.

Para contornar esta dificuldade, foi introduzida uma modificação no algoritmo.

Seja $C=[C_1, C_2, C_3]$, sendo C_1, C_2 e C_3 como na proposição 5.1.

Sejam:

q o número máximo de 1's que um ponto viável pode apresentar;

$q_1(x)$ o número de 1's de $x \in D'$, $x \in R^{n-n_1-n_2}$, D' obtido a partir de $\bar{V}_1(C_3)$.

Denotaremos por $X(n, q)$ o conjunto de vértices de DUC que tem exatamente q componentes iguais a 1.

A modificação no algoritmo baseia-se no seguinte fato:

"se $\bar{V}_1(C_3) \subseteq \bar{V}(C_3)$, tal que:

$$\bar{V}_1(C_3) = \{v \in \bar{V}(C_3) : \sum_{j \in J} v_j = 1\},$$

então o primeiro *antecessor* em

$$G(DUC, \bar{V}_1(C_3) \cup \{e^i\}_{i=1}^{n_1} \cup \{-e^i\}_{i=n_1+1}^{n_1+n_2}), \text{ de qualquer}$$

$x \in D$ com $q - q_1(x) \leq n_1$, viável, pertence a $X(n, q)$ ".

Então, o algoritmo modificado será apresentado a seguir:

Parte I- determine:

$$\bar{V}_1(C_3) \equiv \{v \in \bar{V}(C_3) : \sum_{j \in J} v_j = 1\}$$

Parte II - determine:

$$D' \equiv \{x' : x' \in DUC \cap R^{n-n_1-n_2} \wedge x' + v \notin DUC, \\ \forall v \in \bar{V}_1(C_3)\}$$

$$D \equiv \{x : x \in R^n, x = (1, 0, x'), 1 \in R^{n_1}, 0 \in R^{n_2} \wedge \\ x' \in D'\}$$

$$\bar{D} \equiv X(n, q) \cup \{x \in D : q - q_1(x) > n_1\}$$

Parte III - seja $E_3 \equiv \emptyset$. Determine:

$$\bar{\bar{V}}(C_3) \equiv \{\bar{v} : \bar{v} = (0, v), 0 \in R^{n_1+n_2}, v \in \bar{V}(C_3)\}$$

$$\bar{V}_1 \equiv \{e^i\}_{i=1}^{n_1} \cup \{-e^i\}_{i=n_1+1}^{n_1+n_2} \cup \bar{\bar{V}}(C_3)$$

Passo 2 - determine:

$$E_2 \equiv \{x : x \in \overline{D} \cap F \wedge \nexists y \in \overline{D} \cap F : Cy \geq Cx\}$$

Passo 3 - determine:

$$E_3 \equiv \{y : y \in \overline{D} \setminus F \wedge \nexists x \in E_2 : Cx \geq Cy\}$$

Passo 4 - se $E_3 \equiv \emptyset$, $EF(P) \leftarrow E_2$. Pare. Em caso contrário, determine:

$$E_4 \equiv \{z : z \models y - v \in F, v \in \overline{V}_1, y \in E_3\}$$

e

$$E_5 \equiv \{z' : z' \models y - v \in DUC \setminus F, v \in \overline{V}_1, y \in E_3\}$$

Passo 5 - determine:

$$E_6 \equiv \{z : z \in E_4 \wedge \nexists w \in E_2 \cup E_4 : Cw \geq Cz\}$$

Passo 6 - determine:

$$E_2 \equiv \{x : x \in E_2 \wedge \nexists z \in E_6 : Cz \geq Cx\} \text{ e}$$

$$\text{tome } E_2 \leftarrow E_2 \cup E_6.$$

Passo 7 - determine:

$$E_3 \equiv \{z' : z' \in E_5 \wedge \nexists x \in E_2 : Cx \geq Cz'\} \quad \text{e}$$

volte para o passo 4.

Os elementos do conjunto $X(n, q)$ foram gerados usando-se a estrutura de árvore ilustrada na figura 2.1. A geração é feita por níveis e como a cada nível, em cada nó, apenas uma componente se altera, os valores de $(Ax-b)$ e Cx podem ser calculados simultaneamente com os elementos de $X(n, q)$. A cada nível, faz-se apenas subtração das colunas das matrizes A e C , respectivamente, a^j e c^j , correspondentes à componente j que se alterou.

É fácil ver que no caso de $b_i = 0.25 \sum_{j=1}^n a_{ij}, i=1, \dots, m$, a geração dos elementos $X(n, q)$ por árvore não é eficiente. Para estes problemas, verificamos que a resolução de um problema auxiliar, um problema de programação matemática linear

$$z = \max\{ex : Ax \leq b, x \leq 1 \text{ e } x \geq 0\},$$

como no caso de ter $C \geq 0$, é útil para determinar diretamente o conjunto $X(n, [z])$. Sendo assim, os valores de $(Ax-b)$ e Cx devem ser calculados para cada $x \in X(n, [z])$.

As tabelas 5.1, 5.2 e 5.3, mostram os resultados de 90 problemas resolvidos para $n=10$, $n=14$ e $n=18$, com a matriz C contendo aproximadamente 20% de elementos negativos. A ra-

ção pela qual apresentamos uma amostra maior para cada valor de n (amostra de 30 problemas) é para compensar o efeito aleatório da densidade de elementos negativos em C que afeta o número de colunas de C_3 .

OBSERVAÇÕES

1) Como o número médio de colunas de C com todos os elementos não positivos é pequeno para a densidade de 20% de elementos negativos, nos testes realizados não se distinguiu a matriz C_2 da matriz C_3 . A matriz C foi particionada da seguinte forma: $C=[C_1, C_3]$, onde C_3 inclui todas as colunas com pelo menos um elemento negativo.

2) Para os problemas com $n=18$ e $b_i=0.50 \sum_{j=1}^n a_{ij}$, $i=1, \dots, m$, a cardinalidade de $V_1(C_3)$ foi fixada em 5 como sugeriram alguns testes realizados mas não relatados aqui. Mais testes, talvez, sugerissem um outro número. Este é mais um tópico para futuras pesquisas.

Tabela 5.1

$n=10$, $m=4$, $p=3$, $C20\%$

Nº do Probl.	Valor de α	Nº de colunas em C_1	card $(\bar{V}_1(C_3))$	card $(\bar{D})$	Nº de l's do 1º pto. viável	$ s $ obtido pelo simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
76	.75	7	1	123	7	↑ FOI PROCESSADO SEM SIMPLEX ↓	5	1.8
77	.75	6	0	121	7		12	1.9
78	.75	6	2	121	7		13	1.4
79	.75	3	0	184	7		64	4.2
80	.75	2	10	187	7		43	5.3
81	.75	6	1	121	7		17	2.1
82	.75	5	2	125	7		23	1.6
83	.75	6	0	125	8		26	2.5
84	.75	6	0	50	8		11	1.6
85	.75	5	0	126	7		17	2.2
86	.50	7	0	252	5	↑ FOI PROCESSADO SEM SIMPLEX ↓	16	2.7
87	.50	5	1	252	5		8	2.7
88	.50	4	0	253	5		19	2.9
89	.50	6	0	252	5		8	2.7
90	.50	5	3	252	5		6	3.0
91	.50	3	3	259	5		27	4.5
92	.50	6	0	252	5		26	3.6
93	.50	3	4	258	5		16	3.0
94	.50	7	0	252	5		29	3.0
95	.50	9	0	252	5		17	2.8
96	.25	4	5	45	2	2	7	2.4
97	.25	1	20	127	2	3	11	8.3
98	.25	5	0	120	3	3	11	2.7
99	.25	5	2	120	2	3	4	2.4
100	.25	4	1	120	3	3	4	2.6
101	.25	1	10	125	3	3	15	4.8
102	.25	6	0	120	2	3	10	2.1
103	.25	6	0	120	3	3	7	2.3
104	.25	4	2	120	2	3	10	2.3
105	.25	2	4	121	3	3	12	3.1

Tabela 5.2

$n=14$, $m=4$, $p=3$, $C20\%$

Nº do Probl.	Valor de α	Nº de colunas em C_1	card $(\bar{V}_1(C_3))$	card $(\bar{D})$	Nº de l's do 1º pto. viável	$ z $ obtido pelo Simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
106	.75	8	2	379	11	↑ FOI PROCESSADO SEM SIMPLEX ↓	31	7.5
107	.75	7	5	398	11		48	9.2
108	.75	9	0	370	11		51	9.1
109	.75	8	0	386	11		25	5.5
110	.75	9	0	390	11		36	15.1
111	.75	6	6	1048	10		36	13.3
112	.75	8	1	380	11		34	3.9
113	.75	8	0	1008	10		69	15.8
114	.75	8	2	1007	10		68	14.8
115	.75	8	1	1007	10		19	7.5
116	.50	4	2	3167	8	FOI PROCESSADO SEM SIMPLEX	27	27.6
117	.50	9	1	3003	8		43	30.4
118	.50	8	0	3003	8		17	23.3
119	.50	4	21	3447	7		108	57.8
120	.50	8	2	3003	8		17	28.1
121	.50	9	0	3003	8		97	48.2
122	.50	6	2	3012	8		26	24.1
123	.50	6	1	3433	7		45	34.7
124	.50	7	1	3432	7		59	33.0
125	.50	8	1	3003	8		64	42.8
126	.25	9	0	1001	4	4	20	10.8
127	.25	7	3	1001	3	4	29	11.3
128	.25	8	1	1001	4	4	9	10.4
129	.25	6	2	1001	4	4	7	9.7
130	.25	7	4	1001	4	4	20	11.3
131	.25	9	0	1001	4	4	4	8.5
132	.25	9	0	1001	4	4	14	9.6
133	.25	11	0	1001	3	4	11	9.7
134	.25	3	31	1002	4	4	38	47.2
135	.25	5	6	2002	5	5	23	23.0

Tabela 5.3

$n=18, m=4, p=3, C=20\%$

Nº do Probl.	Valor de α	Nº de colunas em C_1	Card ($V_1(C_3)$)	Card ($\bar{D}$)	Nº de l's do 1º pto. Viável	l's obtido pelo Simplex	Nº de elementos em $EF(P)$	Tempo Total (segs)
136	.75	11	0	3089	14	↑ FOI PROCESSADO SEM SIMPLEX ↓	47	30.5
137	.75	8	8	3220	14		45	23.9
138	.75	10	11	3108	14		67	86.2
139	.75	6	31	4127	14		129	99.4
140	.75	11	2	3083	14		87	134.4
141	.75	11	0	3089	14		72	23.8
142	.75	7	75	3345	14		116	105.7
143	.75	10	7	3101	14		52	35.6
144	.75	8	22	3194	14		102	56.4
145	.75	15	0	3060	14		30	39.7
146	.50	9	4	48620	9	↑ FOI PROCESSADO SEM SIMPLEX ↓	119	449.8
147	.50	9	5	43759	10		46	291.0
148	.50	10	5	43759	10		46	278.0
149	.50	7	5	31937	11		94	299.2
150	.50	8	5	43766	10		44	271.4
151	.50	10	5	43758	10		86	537.3
152	.50	10	5	43758	10		52	323.3
153	.50	12	4	43758	10		89	529.1
154	.50	13	1	48620	9		73	364.2
155	.50	8	5	43765	10		70	440.0
156	.25	10	3	8568	19	5	5	73.3
157	.25	10	7	8568	20	5	5	66.6
158	.25	5	94	8568	56	5	5	151.8
159	.25	11	2	18564	39	6	6	190.4
160	.25	12	3	18564	17	6	6	161.3
161	.25	10	1	8568	21	5	5	68.8
162	.25	9	11	8568	83	5	5	97.2
163	.25	8	13	18564	22	6	6	197.5
164	.25	7	24	18564	19	5	6	242.2
165	.25	10	7	8568	50	5	5	82.5

6. EXPERIÊNCIAS COMPUTACIONAIS VARIANDO OS PARÂMETROS p E m .

Nesta seção apresentaremos os resultados obtidos das experiências computacionais, resolvendo-se 60 problemas para $n=14$ com um número variável de restrições e de funções-objetivo. Cada elemento do vetor b , em todos problemas, é $b_i = 0.50 \cdot \sum_{j=1}^n a_{ij}$, $i=1, \dots, m$. A matriz de critérios C considerada, tem aproximadamente 20% de elementos negativos e portanto a matriz C foi particionada em $C=[C_1, C_3]$ (vide observação 1) da seção anterior). Para cada par de parâmetros (p, m) o tamanho da amostra é de 10 problemas.

Tabela 6.1

(p, m)	Média Nº de colunas em C_1	Média Card $\bar{V}_1(C_3)$	Média Card $\bar{D}$	Média Nº de ele- mentos em $EF(P)$	Média Tempo Total (segs)
(2, 4)	9.1	1.7	3217.7	8.6	22.9
(3, 4)	6.9	3.1	3150.6	50.3	35.0
(4, 4)	5.6	9.3	3300.5	89.7	54.2
(5, 4)	4.5	15.5	3326.7	227.9	122.7
(3, 3)	7.0	5.4	3101.5	47.0	36.5
(3, 4)	6.9	3.1	3150.6	50.3	35.0
(3, 5)	7.6	2.3	3261.6	27.6	28.1
(3, 6)	7.2	6.5	3405.0	44.5	40.2

7. ANÁLISE DOS RESULTADOS COMPUTACIONAIS

Concluindo este capítulo discutiremos os resultados apresentados nas tabelas de 2.1 a 2.6, de 4.1 a 4.3, de 5.1 a 5.3 e 6.1.

- 1) Em todos os problemas, quando o vetor b é tal que

$b_i = 0.50 \cdot \sum_{j=1}^n a_{ij}, i=1, \dots, m$, o tempo total de processamento é maior do que quando $b_i = 0.75 \cdot \sum_{j=1}^n a_{ij},$

$i=1, \dots, m$. Isto se deve ao fato de, no primeiro caso, o número de vértices em $G(DUC, \bar{V}_1)$ que tem o número de componentes não nulas igual ao número máximo de uns que um ponto viável pode apresentar, ser maior que no segundo caso.

- 2) No caso em que a matriz C tem uma linha não posi-

tiva e $b_i = 0.25 \cdot \sum_{j=1}^n a_{ij}, i=1, \dots, m$, o tempo de processamento é maior do que quando $b_i = 0.75 \cdot \sum_{j=1}^n a_{ij},$

$i=1, \dots, m$. Isto porque uma parte significativa da árvore deve ser gerada para obtermos o primeiro ponto viável. A solução do problema auxiliar, um problema de programação matemática linear

$z = \max\{ex : Ax \leq b, x_j \leq 1, x_j \geq 0, j \in J\}$ e a construção do

conjunto de vértices de $G(DUC, \bar{V}_1)$ tendo $|z|$ componentes iguais a um, evitaria a geração de uma parte da árvore mencionada acima.

Portanto, resultados melhores que os apresentados nas tabelas 4.2 e 4.3 poderiam ser obtidos, incorporando esse processo ao algoritmo, quando C tem uma linha não positiva, da mesma forma que foi feito para $C20\%$. Entretanto, o conjunto $\bar{V}_1$ (lembre-se que para $C20\%$, $v \in \bar{V}_1(C_3)$ sse $\sum_{j=1}^n v_j = 1$) e $\text{card}(\bar{V}_1)$ devem ser cuidadosamente estudados, desde que, a escolha feita para construir o conjunto $\bar{V}_1(C_3)$ pode não ser adequada para este caso. Este é um tópico para futuras pesquisas.

- 3) O tempo total de processamento para resolver (P) com a matriz de critérios C tendo uma linha não positiva, aparentemente é maior do que para a matriz $C \geq 0$ ou $C20\%$.
- 4) O número de pontos eficientes em (P) para problemas quando a matriz C tem uma linha não positiva é, aparentemente, maior do que quando $C \geq 0$ ou $C20\%$.
- 5) Uma atenção maior poderia ser dedicada aos testes de *dominância* e às operações para determinar os

antecessores e tentar torná-los mais eficientes , pois uma parte significativa do tempo total é gasto nessas operações.

6) Os problemas com a matriz de critérios $C \geq 0$ e a matriz C com uma linha não positiva podem ser considerados como casos particulares do problema em que a matriz C tem uma determinada porcentagem de elementos negativos.

7) O tempo total de processamento do algoritmo parece ser mais sensível às variações de p do que m .

CAPÍTULO V

CONJUNTO DE PONTOS FORTEMENTE EFICIENTES E DE PONTOS FRACAMENTE EFICIENTES

1. INTRODUÇÃO

Em otimização vetorial discreta os pontos eficientes em (P) :

$$(P) : \max \{ Cx : x \in F \},$$

onde F é o conjunto de pontos viáveis, não maximizam necessariamente um funcional do tipo λCx , com $\lambda \in R^p$, $\sum_{i=1}^p \lambda_i = 1$, $\lambda_i \geq 0$, $i = 1, 2, \dots, p$, sobre o conjunto de pontos viáveis.

De fato, se temos o seguinte problema:

$$(P) : \max \left\{ \begin{bmatrix} 1 & -2 \\ -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} : x \in F \wedge x_j \in \{0, 1\}, j=1, 2 \right\}$$

$$F \equiv \{x^1 = (0, 0), x^2 = (1, 0), x^3 = (1, 1), x^4 = (0, 1)\}.$$

$$EF(P) \equiv F.$$

Consideremos $x^1 \in EF(P)$. Suponhamos que x^1 maximiza λCx sobre F para algum λ nas condições acima. Então, $\lambda Cx^1 \geq \lambda Cx^j$, $j=2, 3, 4 \Rightarrow \lambda C(x^1 - x^j) \geq 0$, $j=2, 3, 4$, donde obtemos o seguinte sistema:

$$\begin{aligned} -\lambda_1 + \lambda_2 &\geq 0 \\ \lambda_1 - 2\lambda_2 &\geq 0 \\ 2\lambda_1 - 3\lambda_2 &\geq 0 \\ \lambda_1 &> 0, \lambda_2 > 0 \end{aligned}$$

cuja única solução é $\lambda_1 = \lambda_2 = 0$. Isto mostra que não existe ne num λ , tal que, $\sum_{i=1}^p \lambda_i = 1$, $\lambda_i \geq 0$, $i=1, 2, \dots, p$ e que x^1 maximi-ze o funcional λCx .

Assim, neste capítulo, daremos uma forma de clas-sificar os elementos de $EF(P)$, conjunto de pontos eficien-tes em (P) , em pontos que chamaremos de *fortemente eficien-tes* e *fracamente eficientes*.

Na seção 2, apresentaremos resultados teóricos que permitem identificar os pontos que são eficientes em (P) e fortemente eficientes. Com a utilização do *Método Simplex* ou do *Método Dual Simplex*, mostramos quais as condições num certo "tableau" para identificar se um ponto é forte ou fracamente eficiente ou se é um ponto dominado (classifica-remos assim, um certo conjunto de pontos que não são neces-

sariamente eficientes em (P)).

Como consequência desses resultados, descreveremos na seção 6, um algoritmo para identificar esses três conjuntos. Apresentaremos, também, os resultados de testes computacionais, considerando-se conjuntos com 100 e 200 elementos e classificando-os em pontos fortemente eficientes, fracamente eficientes ou pontos dominados.

2. IDENTIFICAÇÃO DE PONTOS FORTEMENTE EFICIENTES

Sejam:

$$S \equiv \{ \lambda \in R^p : \sum_{i=1}^p \lambda_i = 1, \lambda_i \geq 0, i=1, \dots, p \}$$

e o problema:

$$(P\lambda) : \max \{ \lambda Cx : x \in F \},$$

onde F é o conjunto de pontos viáveis.

Como em otimização vetorial discreta, se $x^0 \in EF(P)$ não implica necessariamente que $\exists \lambda \in S : x^0$ resolve $(P\lambda)$ seguem as seguintes definições:

DEFINIÇÃO 2.1 - Seja x^0 um ponto eficiente em (P) . Ao conjunto $S(x^0)$, tal que:

$$S(x^0) \equiv \{\lambda \in S : x^0 \text{ resolve } (P\lambda)\},$$

chamamos de *conjunto indiferença* de x^0 .

DEFINIÇÃO 2.2 - Um ponto eficiente x^0 é *fortemente eficiente* sse $S(x^0) \neq \emptyset$.

Denotaremos o conjunto de pontos fortemente eficientes por $\overline{EF}(P)$.

Para a determinação do conjunto $\overline{EF}(P)$, é importante a caracterização dada pelo seguinte teorema, para o conjunto indiferença $S(x^0)$.

TEOREMA 2.1 / 2 /. O conjunto $S(x^0)$ é o conjunto de soluções do sistema $\lambda C(x^0 - x^i) \geq 0$, $\forall x^i \in \overline{EF}(P)$, $\lambda \in S$.

DEFINIÇÃO 2.3 - Um ponto eficiente x^0 é *fracamente eficiente* sse $S(x^0) \neq \emptyset$.

Podemos formular o problema para determinar o conjunto de pontos forte e fracamente eficientes, o qual denotaremos por (PSW) , da seguinte forma:

(PSW) : "Determinar o conjunto de pontos forte e fracamente eficientes em $\max\{Cx : x \in \bar{F}\}$, onde $\bar{F}$ é qualquer subconjunto discreto de R^n ".

Nós nos limitaremos a resolver (PSW) para o caso em que $\bar{F}$ é um conjunto discreto conhecido, $\bar{F} = \{x^i, i \in Q\}$, sendo $Q = \{0, 1, \dots, q\}$.

Para simplificar a exposição dos resultados, daremos a seguir uma formulação equivalente ao problema (PSW), o qual denominaremos por (PSW)'.

(PSW)': "Determinar o conjunto de pontos forte e fracamente eficientes em $\max\{I_p y : y \in Y\}$, onde $Y = \{y^i : y^i = Cx^i, i \in Q\}$ e I_p é a matriz identidade de ordem p ".

É importante notar que, se $EF(P) \subseteq \bar{F}$, então $x^i \in \overline{EF}(P)$ se y^i é fortemente eficiente em (PSW)'.

No que segue, vamos supor que $y^i \neq y^j, i \neq j, \forall i, j \in Q$.

Para verificar se um ponto eficiente y^0 é fortemente eficiente em (PSW)' é suficiente determinar uma solução para o seguinte sistema:

$$\left. \begin{array}{l} \lambda(y^0 - y^i) \geq 0, \quad i \in Q \setminus \{0\} \\ \lambda I_p \geq 0 \end{array} \right\} \quad (2.1)$$

De acordo com o teorema de alternativas de

Tucker*, ou o sistema (2.1) tem solução, ou o seguinte sistema:

$$\left. \begin{aligned} I_p z + \sum_{i=1}^q (y^0 - y^i) \gamma_i &= 0 \\ z > 0, \gamma_i &\geq 0, i \in Q \setminus \{0\} \end{aligned} \right\} \quad (2.2)$$

tem solução z e γ , mas nunca ambos.

O sistema (2.2) pode ser reescrito da seguinte forma:

$$\left. \begin{aligned} y^0 \left(\sum_{i=1}^q \gamma_i \right) &< \sum_{i=1}^q y^i \gamma_i \\ \gamma_i &\geq 0, i \in Q \setminus \{0\} \end{aligned} \right\} \quad (2.3)$$

De (2.3), conclui-se que $\sum_{i=1}^q \gamma_i > 0$. Tomando-se

* Teorema de alternativas de Tucker / 8 /. Sejam dadas as matrizes B, C e D , sendo B não vazia. Então, ou o sistema

I. $Bx \geq 0, Cx \leq 0, Dx = 0$ tem solução x

ou

II. $\left\langle \begin{aligned} B^T y_2 + C^T y_3 + D^T y_4 &= 0 \\ y_2 > 0, y_3 &\geq 0 \end{aligned} \right\rangle$ tem solução y_2, y_3, y_4

mas não ambos.

$$\alpha_i = \frac{\gamma_i}{\sum_{i=1}^q \gamma_i}, \text{ obtemos:}$$

$$\left. \begin{aligned} y^0 &< \sum_{i=1}^q y^i \alpha_i \\ \sum_{i=1}^q \alpha_i &= 1, \quad \alpha_i \geq 0, \quad i \in Q \setminus \{0\} \end{aligned} \right\} \quad (2.4)$$

Pelo fato de que, para y^0 ser fortemente eficiente, é também necessário que o sistema (2.1) tenha solução, os resultados acima implicam na seguinte proposição:

PROPOSIÇÃO 2.1 - Um ponto y^0 é *fortemente eficiente* se e somente se é eficiente e não existe $\alpha_i \geq 0, i=1, \dots, q, \sum_{i=1}^q \alpha_i = 1$, tal que $\sum_{i=1}^q \alpha_i y^i > y^0$.

A partir desta proposição podemos dar uma outra definição para pontos fortemente eficientes.

DEFINIÇÃO 2.4 - Seja $Y \equiv \{y^i = Cx^i, i \in Q\}$. Um ponto eficiente y^0 é *fortemente eficiente* se não é estritamente dominado por uma combinação convexa dos $y^i, i \in Q \setminus \{0\}$.

Deste modo, o sistema (2.4) fornece uma condição - para verificar se um ponto $y^0 \in Y$ é fortemente eficiente ou não. A seguinte proposição apresentará este sistema de uma

forma mais operacional.

PROPOSIÇÃO 2.2 - O sistema (2.4) tem solução sse o seguinte sistema:

$$\left. \begin{aligned} -y^0 &= - \sum_{i=0}^q \beta_i y^i + I_p s \\ 0 &= I_p s - I_p t - \theta e \\ 1 &= \sum_{i=0}^q \beta_i \\ s_i, t_i &\geq 0, \quad i=1, \dots, p \\ \beta_i &\geq 0, \quad i=0, 1, \dots, q \\ \theta &\geq 0 \\ \theta \in R, \quad s, t \in R^p \end{aligned} \right\} \quad (2.5)$$

tem solução com $\theta > 0$.

Demonstração.

Condição Suficiente: Seja $(\beta_0^0, \beta_1^0, \dots, \beta_q^0, s^0, t^0, \theta^0)$ uma solução para (2.5) com $\theta^0 > 0$. Então, $s^0 > 0$ e $\beta_0^0 < 1$.

Portanto,

$$\sum_{i=1}^q \frac{\beta_i^0}{1 - \beta_0^0} y^i > y^0$$

$$\text{com } \sum_{i=1}^q \frac{\beta_i^0}{1-\beta_0^0} = 1$$

$$\text{e } \frac{\beta_i^0}{1-\beta_0^0} \geq 0, \quad i \in Q \setminus \{0\}$$

Logo, o sistema (2.4) tem solução.

Condição Necessária - Suponha que $(\alpha_1^0, \alpha_2^0, \dots, \alpha_q^0)$ resolva o sistema (2.4).

Sejam, $s^0 = \sum_{i=1}^q \alpha_i y^i - y^0$, $\beta_0^0 = 0$, $\theta^0 = \min_{j=1, \dots, p^j} s_j^0$ e $t^0 = s^0 - \theta^0$. Então, $s^0 > 0$, $\theta^0 > 0$ e $(\beta_0^0, \alpha_1^0, \alpha_2^0, \dots, \alpha_q^0, s^0, t^0, \theta^0)$ resolve o sistema (2.5). $\square$

Denotaremos por $F(y^j)$ o conjunto de soluções do sistema (2.5) quando o termo independente da primeira equação deste sistema é y^j , $j \in Q$. Note-se que $F(y^j) \neq \emptyset$, uma vez que $\beta_j = 1$, $\beta_i = 0$, $i \neq j$, $i \in Q$, $s = 0$, $t = 0$ e $\theta = 0$ resolvem (2.5).

Consideremos o seguinte problema:

$$P(y^j) : w^j = \max \{ \theta : (\beta_0, \beta_1, \dots, \beta_q, s, t, \theta) \in F(y^j) \}$$

Pela proposição 2.2, um ponto y^j é fortemente eficiente sse for eficiente e $w^j = 0$.

Poder-se-ia chegar a mesma conclusão resolvendo -se um problema de programação matemática linear tendo como restrições o sistema (2.1). Entretanto, o problema $P(y^j)$ apresenta as seguintes vantagens:

Fato 1: em geral, o sistema (2.5) tem menos restrições que o sistema (2.1), uma vez que, normalmente, $\text{card}(Q) > 2p+1$.

Fato 2: os coeficientes do termo independente do sistema de equações que formam as restrições no problema $P(y^j)$ são iguais aos coeficientes de β_j .

Fato 3: os problemas $P(y^i)$ e $P(y^j)$ são muito semelhantes, $\forall i, j \in Q, i \neq j$. Eles diferem apenas nas p primeiras componentes do termo independente no sistema de equações que formam as restrições. Portanto, tendo resolvido o problema $P(y^i)$, pelo método Simplex ou pelo método Dual Simplex, para resolvermos o problema $P(y^j)$, basta substituímos o termo independente do sistema pelos coeficientes de β_j (fato 2) no "tableau" ótimo e aplicar o método Dual Simplex se necessário. Este fato será útil para o algoritmo que resolve (PSW)', descrito adiante.

PROPOSIÇÃO 2.3 - Suponha que o problema $P(y^j)$ é resolvido pelo método *Simplex* ou *Dual Simplex*. Sejam $\bar{y}^u$ e $\bar{c}^u$, respectivamente, a coluna de coeficientes e custo marginal correspondentes à variável β_u em um dado "tableau":

- a) se em alguma iteração da fase II do método *Simplex* ou em alguma iteração do método *Dual Simplex*, $\bar{y}^u \geq 0$ e $\bar{c}^u < 0$, para algum $u \in Q$, então y^u não é ponto *fortemente eficiente*. Se, adicionalmente, β_z , $z \in Q \setminus \{u\}$, é a única variável básica entre os β_i , $i \in Q$, então $y^z > y^u$.
- b) se no "tableau" ótimo $\bar{y}^z$ tem pelo menos uma componente positiva e $\bar{c}^z = 0$, para algum $z \in Q$, então, ou y^z é *fortemente eficiente* ou é *dominado* por um ponto *fortemente eficiente*.

Demonstração.

- a) Para o "tableau" desta iteração, substitua a coluna correspondente aos termos independentes do sistema, incluindo o coeficiente da linha correspondente à função objetivo, por $(\bar{c}^u, \bar{y}^u)$. Sendo $\bar{y}^u \geq 0$ e $\bar{c}^u < 0$, o problema $P(y^u)$ admite uma solução viável com $\theta > 0$, isto é, o sistema (2.5) admite uma solução com $\theta > 0$. Portanto, y^u não é *fortemente*

eficiente. Se β_z , $z \in Q \setminus \{u\}$, é a única variável básica, então $\beta_z = 1$, $\theta > 0$ e $s > 0$ é uma solução básica para $P(y^u)$. Portanto, $y^z > y^u$.

b)

i) Se $\bar{y}^z \geq 0$, substituindo a coluna correspondente aos termos independentes do sistema, incluindo o coeficiente da linha correspondente à função objetivo, por $(\bar{c}^z = 0, \bar{y}^z \geq 0)$, o "tableau" resultante é ótimo para o problema $P(y^z)$ e o valor ótimo é $w^z = 0$.

Portanto, pela proposição (2.2),

$\nexists$ solução para o sistema (2.4), quando y^z substitui $y^0 \Rightarrow$
 $\nexists$ solução para o sistema (2.1) correspondente $\Rightarrow$
 y^z resolve $(P\lambda)$ para algum $\lambda \in S$.

Se y^z é eficiente $\Rightarrow y^z$ é fortemente eficiente. Caso contrário, qualquer ponto eficiente y^i que domina y^z , também resolve $(P\lambda) \Rightarrow y^i$ é fortemente eficiente. Portanto, y^z é dominado por um ponto fortemente eficiente.

ii) Se $\nexists k$, $1 \leq k \leq p$, tal que $\bar{y}_k^z > 0$, então "pivotamos" no elemento $\bar{y}_r^z$ tal que:

$$\frac{\bar{y}_r^i}{\bar{y}_r^z} = \min_{\bar{y}_k^z > 0} \left\{ \frac{\bar{y}_k^i}{\bar{y}_k^z} \right\}$$

introduzindo β_z na base, obtendo uma solução ótima alternativa para $P(y^i)$. Como neste "tableau" ótimo todos os coeficientes de β_z são não negativos, recaímos no caso i). $\square$

3. ALGORITMO PARA RESOLVER $(PSW)'$.

A proposição 2.3 justifica o algoritmo seguinte que resolve o problema $(PSW)'$, particionando o conjunto dado $Y \equiv \{y^i : y^i = Cx^i, i=0, \dots, r\}$ em conjuntos de pontos forte mente eficientes, fracamente eficientes e pontos dominados.

Vamos supor que os elementos y^i do conjunto Y estejam em ordem decrescente segundo o valor de ey^i , isto é, $ey^i \geq ey^{i+1}$, $i=0, \dots, r-1$.

Inicialização - $Y \equiv \{y^i : y^i = Cx^i, i=0, \dots, r\}$,

$$j=0, \quad \bar{Y}_1 \equiv \phi, \quad \bar{Y}_2 \equiv \phi, \quad \bar{Y}_3 \equiv \phi;$$

Passo 1 - determine o conjunto

$$E \equiv \{y^i : y^i \in Y \wedge y^0 \neq y^i\} \equiv \{y^i, i=0, \dots, q\} \equiv \{y^i, i \in Q\}$$

$$\bar{Y}_3 \equiv Y \setminus E.$$

Passo 2 - resolva o problema $P(y^j)$ pelo método Simplex ou pelo Dual Simplex.

Se em algum "tableau", $\bar{y}^u \geq 0$ e $\bar{c}^u < 0$, para algum $u \in Q$, então:

a) se β_z , para algum $z \in Q \setminus \{u\}$, é a única variável básica entre os β_i , $i \in Q$, então elimine β_u do "tableau". Faça $\bar{Y}_3 \equiv \bar{Y}_3 \cup \{y^u\}$ e $Q \equiv Q \setminus \{u\}$. Em caso contrário, faça $\bar{Y}_2 \equiv \bar{Y}_2 \cup \{y^u\}$ e $Q \equiv Q \setminus \{u\}$.

b) se o "tableau" é ótimo e $\bar{c}^z = 0$ e $\exists k$, $1 \leq k \leq p: \bar{y}_k^z > 0$, para algum $z \in Q$, faça $\bar{Y}_1 \equiv \bar{Y}_1 \cup \{y^z\}$ e $Q \equiv Q \setminus \{z\}$. Não elimine β_z do "tableau".

Passo 3 - se $Q \equiv \emptyset$, vá para o passo 4. Caso contrário, tome o menor índice em Q . Seja u este índice. Substitua o lado esquerdo do último "tableau" ótimo por $(\bar{c}^u, \bar{y}^u)$, faça $j = u$ e volte para o passo 2.

Passo 4 - determine os conjuntos:

$$Y_1 \equiv \{y : y \in \bar{Y}_1 \wedge \nexists \bar{y} \in \bar{Y}_1 : \bar{y} \geq y\}$$

$$Y_2 \equiv \{y : y \in \bar{Y}_2 \wedge \nexists \bar{y} \in \bar{Y}_1 \cup \bar{Y}_2 : \bar{y} \geq y\}$$

$$Y_3 \equiv \bar{Y}_3 \cup (\bar{Y}_2 \setminus Y_2) \cup (\bar{Y}_1 \setminus Y_1)$$

Pare.

No passo 1 do algoritmo, identificamos o conjunto E formado pelos elementos de Y que não são dominados por

y^0 e o conjunto $\bar{Y}_3$ de elementos de Y dominados por y^0 .

Pela proposição 2.3, parte a), os y^u 's identificados no passo 2 a) não são fortemente eficientes. Portanto, se $y^z > y^u, y^u \in \bar{Y}_3$; em caso contrário, $y^u \in \bar{Y}_2$. No passo 2 b) , $y^z \in \bar{Y}_1$, o que é justificado pela parte b) da proposição 2.3.

No passo 4, identificamos os elementos dominados em $\bar{Y}_1$ e $\bar{Y}_2$ e colocamo-los em Y_3 . Portanto, Y_1 , é o conjunto de pontos *fortemente eficientes*, Y_2 é o conjunto de pontos *fracamente eficientes* e Y_3 é o conjunto de pontos *dominados*. O algoritmo termina num número finito de iterações, pois E tem um número finito de pontos.

A tabela 3.1 mostra os resultados computacionais obtidos pela aplicação do algoritmo para resolver o problema (PSW)':

(PSW)': "determinar os pontos forte e fracamente eficientes em

$$\max \{I_p y : y \in Y\}"$$

Para cada par (p, q) , o tamanho da amostra é de 30 problemas. Todos os elementos do conjunto Y foram gerados, aleatoriamente, no intervalo $[0, 10]$.

Tabela 3.1

Valores de (q,p)	Média Nº de ele mentos em Y_1	Desvio Padrão	Média Nº de ele mentos em Y_2	Desvio Padrão	Média Nº de ele mentos em Y_3	Desvio Padrão
(99,2)	3.2	1.1	.6	.9	96.2	1.5
(99,4)	15.1	3.4	9.3	4.7	75.6	7.1
(99,6)	34.1	4.8	24.5	6.5	41.4	9.4
(199,2)	3.6	1.0	.5	.7	195.9	1.2
(199,4)	20.8	3.7	15.1	5.6	164.1	7.7
(199,6)	49.7	6.2	46.4	8.8	103.9	11.4

CAPÍTULO VI

PROBLEMAS DE PROGRAMAÇÃO VETORIAL COM OS ELEMENTOS DA MATRIZ DE CRITÉRIOS NUM CERTO INTERVALO

1. INTRODUÇÃO

Quando se trata de problemas reais é difícil especificar precisamente os valores de cada elemento da matriz de critérios. No entanto, poderíamos determinar um intervalo de variação para cada um destes elementos.

Neste capítulo analisaremos como obter o conjunto de pontos eficientes em relação a toda matriz de critérios, cujos elementos variam num certo intervalo.

Na seção 2, apresentaremos alguns resultados teóricos relacionando o conjunto de pontos eficientes desse problema com o conjunto de pontos eficientes em problemas afins. Desta forma, será possível desenvolver, a partir de um algoritmo apresentado no capítulo IV, um algoritmo para determinar o conjunto de pontos eficientes do problema em questão. Como aplicação desta metodologia, apresentaremos um procedimento para determinar um subconjunto de pontos eficientes para o problema de programação vetorial côncava com variáveis

zero-um.

Na seção 3, apresentaremos alguns resultados computacionais para analisar a relação entre o número total de pontos eficientes num problema de programação vetorial côncava e o número de pontos eficientes obtidos para este mesmo problema, utilizando um algoritmo apresentado na seção 2.

2. RESULTADOS TEÓRICOS

Seja F o conjunto de pontos viáveis. Consideremos o seguinte problema:

"determinar, se existir, o conjunto de pontos de F eficientes em relação a toda matriz $C=(c_{ij})$, tal que $c_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}]$, $1 \leq i \leq p$, $1 \leq j \leq n$ ".

Este problema pode ser formulado da seguinte maneira: encontre, se existir, o conjunto de pontos eficientes em

$$(PC'): \max \{Cx : x \in F \wedge C \in \mathcal{C}\},$$

onde $\mathcal{C}$ é o conjunto das matrizes $p \times n$, cujos elementos

$$c_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}], \quad 1 \leq i \leq p, \quad 1 \leq j \leq n.$$

Definimos como conjunto de pontos eficientes em (PC') , o conjunto de pontos de F eficientes em relação a toda matriz $C \in \mathcal{C}$. Denotaremos este conjunto por $EF(PC')$.

O conjunto $\mathcal{C}$ é convexo. Se $\underline{c}_{ij} \neq \bar{c}_{ij}$, $\forall i, 1 \leq i \leq p, \forall j, 1 \leq j \leq n$,

então o conjunto $\mathcal{C}$ tem $2^{p \times n}$ pontos extremos que são as matrizes $p \times n$, onde cada elemento c_{ij} assume o valor mínimo ou máximo no intervalo $[\underline{c}_{ij}, \bar{c}_{ij}]$. Como para cada i fixo, $1 \leq i \leq p$, a i -ésima linha de um ponto extremo de $\mathcal{C}$ pode assumir 2^n valores distintos (cada elemento assumindo ou o valor mínimo ou o valor máximo), podemos formar uma matriz $(p \times 2^n) \times n$, cujas linhas são formadas por linhas distintas de todas as matrizes que são pontos extremos de $\mathcal{C}$. Denotaremos esta matriz por $\tilde{C}$.

Considerando-se essa matriz de critérios $\tilde{C}$, podemos formular o seguinte problema:

encontre, se existir, o conjunto de pontos eficientes em

$$(PC) : \max \{ \tilde{C}x : x \in F \},$$

onde F é o conjunto de pontos viáveis.

Denotaremos o conjunto de pontos eficientes em (PC) por $EF(PC)$.

PROPOSIÇÃO 2.1. Sejam $EF(PC')$ e $EF(PC)$, os conjuntos de pontos eficientes em (PC') e (PC) , respectivamente. Então, $EF(PC') \subseteq EF(PC)$.

Demonstração. Seja $x^0 \notin EF(PC) \Rightarrow \exists C \in \mathcal{C} : x^0$ não é eficiente em relação a $C \Rightarrow x^0 \notin EF(PC')$.

Portanto, $EF(PC') \subseteq EF(PC)$. $\square$

DEFINIÇÃO 2.1. Seja $v \in R^n$, tal que $v_j \in \{-1, 0, 1\}, \forall j \in J$. Definimos uma matriz $U_v = (u_{ij})$, $i=1, \dots, p$ e $j=1, \dots, n$, onde:

$$u_{ij} = \begin{cases} \underline{c}_{ij} & \text{se } v_j = -1 \\ \bar{c}_{ij} & \text{se } v_j = 1 \\ \tilde{c}_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}] & \text{se } v_j = 0 \end{cases}$$

PROPOSIÇÃO 2.2. Sejam $(x^0, x^1) \in F \times F$ e $v = x^1 - x^0$. Se $Cv \geq 0$ para algum $C \in \mathcal{C}$ então $U_v \cdot v \geq 0$.

Demonstração

Se $C \in \mathcal{C} \Rightarrow c_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}]$.

Pela definição da matriz $U_v: \sum_{j \in J} u_{ij} v_j \geq \sum_{j \in J} c_{ij} \cdot v_j \geq 0$, com pelo menos uma desigualdade estrita, pois $Cv \geq 0$.

Portanto, $U_v \cdot v \geq 0$. $\square$

As proposições 2.1 e 2.2 sugerem o seguinte processo para determinar $EF(PC')$:

- resolver o problema (PC) e determinar $EF(PC)$
- determinar $v = x^1 - x^0$, $(x^1, x^0) \in EF(PC) \times EF(PC)$ e construir U_v .
- verificar se $U_v \cdot v \geq 0$. Se isto ocorrer, x^0 é dominado por x^1 em relação à matriz $U_v \in \mathcal{C}$. Portanto, $x^0 \notin EF(PC')$. Se x^0 não é dominado por x^1 em relação

ã matriz $U_v \in \mathcal{C}$, então não será dominado por x^1 em relação a nenhuma matriz $C \in \mathcal{C}$.

Em geral, a matriz $\hat{C}$ tem muitas linhas. Entretanto, quando $\bar{c}_{ij} > c_{ij} \geq 0$, $\forall i, 1 \leq i \leq p, 1 \leq j \leq n$, vamos resolver o seguinte problema auxiliar:

$$(PI): \max\{I_n x : x \in F\},$$

onde I_n é matriz identidade de ordem n .

Utilizando o algoritmo apresentado no capítulo IV, seção 2, para matriz de critérios $C \geq 0$, determinamos o conjunto $EF(PI)$, o conjunto de pontos eficientes em (PI) . Pela proposição 3.1, do capítulo IV, $EF(P) \subseteq EF(PI)$, então, $EF(PC) \subseteq EF(PI)$. Pela proposição 2.1, $EF(PC') \subseteq EF(PC) \subseteq EF(PI)$. Portanto, podemos determinar $EF(PC')$, testando se $U_v \cdot v \geq 0$, para $v = x^1 - x^0$, onde $(x^1, x^0) \in EF(PI) \times EF(PI)$.

Uma aplicação imediata do problema (PC') é resolver o problema de programação vetorial côncava com variáveis zero-um, apresentado na seguinte forma:

$$(PF): \max\{f(x) : x \in F\}$$

onde: $f(x)$ é uma função vetorial côncava diferenciável, definida sobre um conjunto aberto contendo DUC e

$$F \equiv \{x \in DUC : Ax \leq b\}.$$

As componentes de $f(\cdot)$ serão denotadas por $f_i(\cdot)$,

$i=1, 2, \dots, p$.

Seja $x^0 \in F$. O seguinte problema é uma aproximação linear em x^0 , para o problema (PF):

$$(\nabla PF x^0) : \max \{ \nabla f(x^0) x : x \in F \}.$$

PROPOSIÇÃO 2.3. Sejam $EF(PF)$ e $EF(\nabla PF x^0)$ os conjuntos de pontos eficientes em (PF) e $(\nabla PF x^0)$, respectivamente. Então, $EF(\nabla PF x^0) \subseteq EF(PF)$.

Demonstração. Seja $x^0 \notin EF(PF)$. Então,

$$\exists x \in F : f(x) \geq f(x^0).$$

Como $f(x)$ é diferenciável e côncava *

$$\nabla f(x^0)(x - x^0) \geq f(x) - f(x^0) \geq 0.$$

Portanto, $x^0 \notin EF(\nabla PF x^0)$. $\square$

A principal desvantagem de se resolver o problema $(\nabla PF x^0)$ para obtermos um subconjunto de $EF(PF)$ é que este problema depende de um ponto particular.

* Teorema / 8/. Seja f uma função definida sobre um conjunto aberto $X^0 \subset R^n$, diferenciável em $\bar{x} \in X^0$. Se f é côncava em $\bar{x} \in X^0$, então:

$$f(x) - f(\bar{x}) \leq \nabla f(\bar{x})(x - \bar{x}), \text{ para cada } x \in X^0.$$

Esta inconveniência sugere o uso do problema (PC') da seguinte forma:

$$(\forall PF) : \max \{ Cx : x \in F \wedge C \in \mathcal{C} \} ,$$

onde: $\mathcal{C}$ é o conjunto das matrizes $p \times n$, cujos elementos $c_{ij} \in [\underline{c}_{ij}, \bar{c}_{ij}]$ e os elementos $\underline{c}_{ij}$ e $\bar{c}_{ij}$ são definidos como:

$$\underline{c}_{ij} = \min_{x \in DUC} \frac{\partial f_i(x)}{\partial x_j} , \quad i=1, \dots, p \quad \text{e} \quad j=1, \dots, n$$

e

$$\bar{c}_{ij} = \max_{x \in DUC} \frac{\partial f_i(x)}{\partial x_j} , \quad i=1, \dots, p \quad j=1, \dots, n.$$

A proposição seguinte mostra um resultado semelhante ao da proposição 2.1.

PROPOSIÇÃO 2.4. Sejam $EF(\forall PF)$ e $EF(PF)$ os conjuntos de pontos eficientes em $(\forall PF)$ e (PF) , respectivamente. Então, $EF(\forall PF) \subseteq EF(PF)$.

Demonstração.

Se $x^0 \in EF(\forall PF) \Rightarrow x^0 \in EF(\forall PF x^0) \subseteq EF(PF)$ (proposição anterior) $\Rightarrow x^0 \in EF(PF)$.

Portanto, $EF(\forall PF) \subseteq EF(PF)$. $\square$

Como o problema $(\forall PF)$ é uma variante do problema (PC') ,

podemos utilizar o processo que determina $EF(PC')$ para determinar o conjunto $EF(\forall PF)$, isto é:

- resolver o problema (PI) e determinar o conjunto $EF(PI)$
- eliminar de $EF(PI)$, os elementos que são dominados em relação a algum $C \in \mathcal{C}$, aplicando a proposição 2.2.

A questão a ser respondida, neste ponto, é se $EF(\forall PF)$ representa um subconjunto relevante de $EF(PF)$. Para analisar esta questão, apresentamos a seguir os resultados computacionais obtidos para estes problemas.

3. RESULTADOS COMPUTACIONAIS SOBRE $EF(\forall PF)$ E $EF(PF)$.

O problema (PF) e o correspondente problema $(\forall PF)$ foram resolvidos para analisarmos a relação entre as cardinalidades dos conjuntos $EF(\forall PF)$ e $EF(PF)$.

O problema (PF) foi resolvido usando-se a definição de *ponto eficiente*, isto é, determinando-se, os pontos $x^0 \in F: \nexists x \in F: f(x) \geq f(x^0)$.

O problema $(\forall PF)$ foi resolvido pelo processo que acabamos de descrever, isto é, eliminando de $EF(PI)$ os pontos dominados em relação alguma matriz $C \in \mathcal{C}$.

Foram resolvidos 45 problemas, cada qual com três cri

térios

$$f_1(x) = \sum_{i=1}^n s_i (1 - \exp(-m_i x_i))$$

$$f_2(x) = \sum_{i=1}^n (s_i x_i - m_i x_i^2)$$

$$f_3(x) = \sum_{i=1}^n s_i (x_i + c_i) / (x_i + m_i),$$

com $s_i > 0$, $m_i > 0$, $m_i > c_i$ e $s_i > 2m_i$, $i=1, \dots, n$.

O conjunto viável considerado é:

$$F \equiv \{x \in DUC : Ax \leq b\},$$

onde: os elementos a_{ij} , da matriz A , foram gerados aleatoriamente no intervalo $[0, 100]$ e $b_i = 0.5 \cdot \sum_{j \in J} a_{ij}$, $i=1, \dots, m$.

Os coeficientes da função de critérios foram escolhidos de forma que em cada problema a relação $\bar{c}_{ij}/c_{ij}$ assumisse um mesmo valor para $\forall i=1, \dots, p$ e $\forall j=1, \dots, n$. Foram adotados nas experiências computacionais três valores para esta relação: 1.1, 1.2, 1.3. Isto significa que o valor máximo das derivadas em DUC é 10%, 20%, 30% maior que o seu valor mínimo em DUC.

Foram considerados três valores para n : $n=10$, $n=12$ e $n=14$. Para cada valor de n , foram gerados cinco conjuntos de

pontos viáveis F .

A tabela 3.1 mostra, para cada cinco problemas, o número médio de pontos em $EF(\nabla PF)$ e $EF(PF)$, o quociente entre estes valores, denotado por $\text{card}_1/\text{card}_2$ e o número médio de pontos viáveis.

Da análise da coluna de card_2 podemos notar que o número médio de pontos em $EF(PF)$ não varia quando a relação $\bar{c}_{ij}/c_{ij}$ varia de 1.1 para 1.3. Entretanto, o quociente $\text{card}_1/\text{card}_2$ decresce com a variação de $\bar{c}_{ij}/c_{ij}$.

Para finalizar cabe discutir uma questão mencionada anteriormente: o conjunto $EF(\nabla PF)$ representa um subconjunto relevante de $EF(PF)$? Em termos de número de pontos em $EF(PF)$, a resposta, em geral, parece ser não, principalmente se estamos interessados em obter todos os elementos de $EF(PF)$. Contudo, pode ser considerado um método útil para determinar facilmente alguns pontos de $EF(PF)$.

Tabela 3.1

n	$\bar{c}_{ij}/c_{ij}$	Nº médio de pontos em $EF(\sqrt{PF})$ Card ₁	Nº médio de pontos em $EF(PF)$ Card ₂	Card ₁ /Card ₂	Nº médio de pontos em F
10	1.1	7.8	12.6	.619	245.2
12	1.1	6.6	14.6	.452	985.4
14	1.1	14.8	27.8	.532	4012.0
10	1.2	4.4	12.6	.349	245.2
12	1.2	4.4	14.6	.301	985.4
14	1.2	7.8	27.8	.280	4012.0
10	1.3	3.4	12.6	.269	245.2
12	1.3	2.2	14.6	.150	985.4
14	1.3	3.0	27.8	.107	4012.0

CONCLUSÕES

O mapeamento N introduzido no capítulo II, seção 2, permitiu desenvolver vários resultados teóricos e algoritmos. Este mapeamento é mais adequado para analisar os problemas tratados neste trabalho, do que o mapeamento M introduzido por G. R. Bitran em /2/. Esta conclusão está implícita na proposição 2.3, do capítulo II, uma vez que $N(v) \equiv M(v) + \{v\}$ implica que cada ponto de $N(v)$ *domina* um ponto de $M(v)$. Uma parte significativa da teoria foi dirigida para a análise do problema (P') . A razão disto, é que o problema simplificado ajuda a compreender melhor o problema (P) e desenvolver algoritmos mais eficientes.

O algoritmo apresentado no capítulo III, é uma versão melhorada do algoritmo apresentado por Bitran. Entretanto, ainda não é adequado para resolver grandes problemas como se pode notar comparando os tempos de processamento para problemas com $n=10$ e $n=18$. Portanto, devem ser feitos muitos estudos sobre estes problemas. Neste trabalho, algumas questões foram levantadas, como por exemplo, em relação à cardinalidade de $\bar{V}_1$ e sobre qual o conjunto de menor cardinalidade que satisfaz $\bigcup_{v \in V} N(v) \equiv \bigcup_{\bar{v} \in \bar{V}} N(\bar{v})$. Ainda, outras questões poderiam ser levantadas, como por exemplo, que subconjunto

de V escolher, para que $\bigcap_{v \in V} N(v) \neq \emptyset$, uma vez que
 $\bigcap_{v \in V} N(v) \subseteq \bigcap_{\tilde{v} \in \tilde{V}} N(\tilde{v}) \subseteq EF(P')$ (proposição 2.6, capítulo II).

Outro tópicos a ser explorado seria em relação à técnica de decomposição que permitisse resolver o problema (P) através de uma sequência de pequenos problemas permitindo - reduzir, significativamente, o tempo de processamento. Caracterização de pontos fracamente eficientes, bem como processos para identificar sua existência, num dado problema poderiam melhorar os futuros algoritmos. Também, testes eficientes para identificar subconjuntos de D ou $\bar{D}$ com elementos que não são eficientes em (P) ou não geram antecessores eficientes em (P) , seriam úteis.

Certamente, ainda existem muitos aspectos a serem considerados em problemas de programação vetorial com variáveis zero-um.

APÊNDICE 1

Apresentaremos neste apêndice alguns detalhes de programação dos algoritmos apresentados no capítulo IV, em ALGOL do computador B-6700.

Mencionamos inicialmente os operadores booleanos, operadores de relação, funções intrínsecas e o subprograma que foram utilizados para executar certas operações do algoritmo.

Operadores booleanos: and, or, not

Operadores de relação: is, isnt

Os parâmetros das funções intrínsecas serão indicados por $\langle ae \rangle$, $\langle ae_1 \rangle$ ou $\langle ae_2 \rangle$ quando são expressões aritméticas e por $\langle arrayrow \rangle$ quando é vetor.

Funções intrínsecas:

- firstone* ($\langle ae \rangle$) - determina o "bit" mais à esquerda de $\langle ae \rangle$ com valor 1;
- ones* ($\langle ae \rangle$) - determina o número total de "bits" em $\langle ae \rangle$ com valor 1;

integert (<ae>) - determina o l<ae>];

listlookup (<ae₁₂₂₁

time (2) - determina o tempo de processamento, com unidade 60 vezes o segundo.

Subprograma:

ZX3LP (A, IA, B, C, N, M₁, M₂, S, PSOL, DSOL, RW,IW,
IER)

resolve: max cx

s.a. Ax ≤ b

x ≥ 0

pelo método Simplex Revisado, onde:

A = matriz das restrições, (M₁+ M₂+ 2) x N;

IA = número de linhas da matriz de restrições como foi especificado no programa de chamada,

IA < (M₁+ M₂+ 2);

B = vetor de comprimento (M₁+ M₂+ 2);

C = vetor da função - objetivo de comprimento N ;

M_1 = número de restrições com desigualdade;

M_2 = número de restrições com igualdade;

S = valor da função objetivo;

$PSOL$ = solução do problema primal;

$DSOL$ = solução do problema dual;

RW = vetor de trabalho de comprimento

$$(M_1 + M_2 + 2) * (M_1 + M_2 + 2) + 2 * M_1 + M_2 + 2;$$

IW = vetor de trabalho de comprimento

$$2 * M_2 + 3 * M_1 + 4;$$

IER = indicador de erro.

Devido à linguagem utilizada, a manipulação de palavras parciais é relativamente simples. Portanto, tantos os elementos do conjunto $\bar{V}_1$ como os elementos de DUC foram armazenados numa palavra de 48 bits. Consegue-se com isto, efetuar determinadas operações mais rapidamente por meio de operadores booleanos ou de relação, citados anteriormente. Porém, o programa fica restrito para problemas de dimensão $n \leq 48$.

Descreveremos a seguir, os recursos utilizados em cada parte do algoritmo.

PARTE I

A geração do conjunto

$\bar{V} \subseteq V \equiv \{v \in R^n: Cv \geq 0, v_j \in \{-1, 0, 1\}, j \in J\}$, o conjunto de vetores *envolventes* de V , apresenta na prática muitas dificuldades, principalmente, por causa do tempo de processamento, pois teoricamente temos 3^n elementos que são candidatos a V e muitas comparações devem ser feitas para se lecionar os elementos de $\bar{V}$.

Vários processos foram testados e o que se mostrou mais eficiente foi o descrito no capítulo III, seção 3, que usa a estrutura de árvore, cuja geração é feita por níveis. Neste processo, o inconveniente é que, o número de vértices para os quais os descendentes ainda devem ser gerados cresce exponencialmente. Isto nos levou a estudar uma maneira de armazená-los e manipulá-los com certa facilidade. Várias formas foram testadas, inclusive utilizando-se da representação de números na base 3, pelo fato de termos os algarismos 0, 1 e -1, mas os resultados não foram satisfatórios. A melhor forma encontrada foi a de representar cada vetor em dois vetores v^1 e v^2 , tal que, $v = v^1 - v^2$.

Na prática, não geramos o vetor v , mas geramos os vetores v^1 e v^2 , separadamente. Note-se que, quando uma componente $v_j = 1$ deve ser gerada, coloca-se 1 em v_j^1 e quando

$v_j = -1$, coloca-se 1 em v_j^2 , o que permite que cada componente v_j seja representada num "bit" da palavra.

Com esta representação, para verificar se $Cv \geq 0$, basta testar se $(Cv^1 - Cv^2) \geq 0$. Na realidade, não há necessidade de efetuarmos as multiplicações. Identificamos as componentes não nulas de v^1 e v^2 , através da função *firstone* e por meio de adição e subtração de colunas da matriz C teremos efetuado Cv .

Para testar se um vetor *envolve* o outro, é útil que sejamos capazes de identificar todos os vetores que tem os mesmos valores numa determinada posição j , $1 \leq j \leq n$. Assim, para cada novo vetor em consideração, basta verificarmos se os vetores que tem o mesmo valor na última posição definida para o novo vetor o *envolve*. Podemos utilizar os operadores *and* e *is* para fazer esta verificação, pois, sendo $v = v^1 - v^2$ e $\bar{v} = \bar{v}^1 - \bar{v}^2$, se v^1 *and* $\bar{v}^1$ *is* v^1 e v^2 *and* $\bar{v}^2$ *is* v^2 , então, v *envolve* $\bar{v}$.

EXEMPLOS

- 1) Sejam $v = (0, -1, 0, 0, 1, 0)$, $\bar{v} = (1, -1, 1, 0, 1, 0)$
 $v^1 = (0, 0, 0, 0, 1, 0)$, $\bar{v}^1 = (1, 0, 1, 0, 1, 0)$
 $v^2 = (0, 1, 0, 0, 0, 0)$, $\bar{v}^2 = (0, 1, 0, 0, 0, 0)$

$$v^1 \text{ and } \bar{v}^1 = (0, 0, 0, 0, 1, 0) \text{ is } v^1$$

$$v^2 \text{ and } \bar{v}^2 = (0, 1, 0, 0, 0, 0) \text{ is } v^2$$

Portanto, v envolve $\bar{v}$.

$$2) \text{ Sejam } v = (0, -1, 0, 0, 1, 0), \quad \bar{v} = (0, 1, 1, 0, 1, 0)$$

$$v^1 = (0, 0, 0, 0, 1, 0), \quad \bar{v}^1 = (0, 1, 1, 0, 1, 0)$$

$$v^2 = (0, 1, 0, 0, 0, 0), \quad \bar{v}^2 = (0, 0, 0, 0, 0, 0)$$

$$v^1 \text{ and } \bar{v}^1 = (0, 0, 0, 0, 1, 0) \text{ is } v^1$$

$$v^2 \text{ and } \bar{v}^2 = (0, 0, 0, 0, 0, 0) \text{ is not } v^2$$

Portanto, v não envolve $\bar{v}$.

Devido às características dos vetores *base* que são selecionados para a parte II do algoritmo, podemos fazer uso dos operadores *or*, *and* e *is* para esta seleção. Devemos ter um *identificador* que guarda as posições das componentes não nulas dos vetores que pertencem ao conjunto V_{base} . Seja h este identificador. Para testar se um certo vetor pertencerá a V_{base} basta verificarmos se $h \text{ and } (v^1 \text{ or } v^2) \text{ is } 0$.

EXEMPLO. Vamos supor que já temos:

$$h = (1, 1, 0, 0, 0, 1, 0) \text{ (isto significa que os vetores } v, \text{ tais que, } v_j \neq 0, \text{ para } j=1, 2, 6, \text{ já pertencem a } V_{\text{base}}).$$

Queremos testar se $v=(0,0,1,0,-1,0,0)$ pertence a V_{base} .

$$v^1 = (0,0,1,0,0,0,0)$$

$$v^2 = (0,0,0,0,1,0,0)$$

$$v^1 \text{ or } v^2 = (0,0,1,0,1,0,0)$$

$$n \text{ and } (v^1 \text{ or } v^2) = (0,0,0,0,0,0,0) \text{ is } 0.$$

Portanto, $v \in V_{\text{base}}$.

PARTE II

Para construirmos os *radicais* a partir de um elemento v de V_{base} , podemos gerar consecutivamente os números $0, 1, 2, \dots, 2^{n_0} - 1$, onde n_0 é o número de componentes não-nulas de v e colocarmos cada algarismo 0 ou 1 (o número está na base 2) nas posições j , $1 \leq j \leq n$, tais que $v_j \neq 0$. Todas as posições não-nulas de todos os vetores de V_{base} devem ser guardadas através do operador *or*, pois serão as posições fixas do radical final. Para se obter n_0 , podemos utilizar a função *ones*.

Para se obter um elemento de RDF, um radical final x^k , basta aplicarmos o operador *or* para os radicais $x^{ij(i)}$ que formam x^k .

Por exemplo: $x^1 = x^{11} \text{ or } x^{21} \text{ or } x^{31} \text{ or } x^{41}$.

Para verificar quais são os vetores $v \in \bar{V}_1$, tais que $x+v \notin DUC$, sendo x um descendente de x^k , temos que achar uma componente v_ℓ de v , tal que : $v_\ell = 1$ se $x_\ell^k = 1$ ou $v_\ell = -1$ se $x_\ell^k = 0$. Isto sugere que tenhamos identificados na parte I do algoritmo, quais são os vetores de $\bar{V}_1$ que tem valores $+1$ ou -1 em cada uma das posições j , $1 \leq j \leq n$.

Antes de analisarmos qualquer descendente de x^k , devemos "eliminar" todos os v 's pertencentes a $\bar{V}_1$, tais que $v+x^k \notin DUC$. Se x^k "eliminar" todos os elementos de $\bar{V}_1$ então todos os seus descendentes pertencem a D . Em caso contrário, para cada descendente de x^k , examinamos os vetores v 's que ainda não foram eliminados pelos seus antecedentes. Nesse ponto, a função *listlookup* foi utilizada.

Quando foi utilizado o problema de programação linear auxiliar

$z = \max\{ex : Ax \leq b, x \leq 1 \wedge x \geq 0\}$, este foi resolvido pelo subprograma $ZX3LP(A, IA, B, C, N, M_1, M_2, S, PSOL, DSOL, RW, IW, IER)$, que usa o *Método Simplex Revisado*.

Para determinar $[z]$ usamos a função *integert*.

PARTE III

Nesta parte do algoritmo, para testar a viabilidade e

a dominância entre os elementos, foi utilizado o processo de identificar as posições cujos valores são não nulos, através da função *firstone* e usar apenas adição ou subtração de colunas da matriz *A* ou da matriz *C*.

Quando a matriz *C* é não negativa, o passo 4 do algoritmo pode ser simplificado, de tal forma que não há necessidade de efetuar a operação $y=x-v$, para determinar o antecesor *y* do sucessor *x*. Suponha que *k* seja o último índice tal que $x_k = 0$. Para obter o seus antecessores, basta anularmos uma componente x_{k+j} de cada vez, $j=1,2,\dots,n-k$. Este é o processo de geração da árvore por níveis, mencionado no capítulo IV, seção 2. Com este processo evitamos também a repetição de antecessores.

Quando a matriz *C* é de outro tipo, essa mesma simplificção não pode ser utilizada. Neste caso, as operações tornam-se mais complexas, pois, dado um elemento *x* temos que verificar se $x-v \in DUC$. Esta verificação e a operação $x-v$ foram feitas através dos operadores *and*, *or*, *not*, *is* e *isnt*, da seguinte forma:

Seja $v=v^1-v^2$. Se *x and v² is 0* e *x or v¹ is x*, então $x-v \in DUC$ e $y=x-v$ pode ser calculado da seguinte forma:

$$y = (x \text{ or } v^2) \text{ and not } v^1.$$

EXEMPLO.

1) Sejam: $x = (1, 1, 0, 1, 0)$
 $v = (1, 0, -1, 1, 0)$
 $v^1 = (1, 0, 0, 1, 0)$
 $v^2 = (0, 0, 1, 0, 0)$

Então:

x and $v^2 = (0, 0, 0, 0, 0)$ is 0
 x or $v^1 = (1, 1, 0, 1, 0)$ is x

Portanto, $x - v \in DUC$.

x or $v^2 = (1, 1, 1, 1, 0)$
not $v^1 = (0, 1, 1, 0, 1)$
 $(x$ or $v^2)$ and not $v^1 = (0, 1, 1, 0, 0)$
 $y = (0, 1, 1, 0, 0)$

2) Sejam: $x = (1, 1, 0, 1, 0)$
 $v = (1, -1, 0, 0, 0)$
 $v^1 = (1, 0, 0, 0, 0)$
 $v^2 = (0, 1, 0, 0, 0)$

Então:

x and $v^2 = (0, 1, 0, 0, 0)$ isnt 0

Portanto, $x - v \notin DUC$.

Além disso, ainda existe a possibilidade de termos antecessores repetidos que devem ser controlados assim que são gerados, para evitar outros problemas.

Para testar a dominância entre os elementos de dois conjuntos, os elementos do vetor *e.C* foram classificados em ordem decrescente usando-se o algoritmo de classificação *quicksort* apresentado pelo D.E.Knuth em / 7 /.

O tempo gasto em cada problema apresentado nas tabelas, é o tempo dado pela função *time* (2) transformado em segundos.

APÊNDICE 2

1. GRAFOS

Se T é um conjunto finito, denotaremos por $T^{(2)}$ o conjunto dos pares não ordenados de elementos distintos de T , isto é, $T^{(2)} = \{\{u, v\} \subseteq T : u \neq v\}$.

Um *grafo* G é um par ordenado (T, E) formado por conjunto finito T e um subconjunto E de $T^{(2)}$. Os elementos de T são chamados de *vértices* do grafo e os elementos de E de *arestas* do grafo.

Se G é um grafo, denotaremos por VG o conjunto de seus vértices e EG o conjunto de suas arestas.

Sejam G e H grafos. Diz-se que H é *subgrafo* de G sse $VH \subseteq VG$ e $EH \subseteq EG$.

Um grafo G é *desconexo* sse existem subgrafos H e K tais $VG = VH \cup VK$, $EG = EH \cup EK$, $VH \neq \emptyset$ e $VK \neq \emptyset$. Caso contrário, G é *conexo*.

Se $\alpha = \{u, v\}$ é uma aresta, diz-se que u e v são *vértices adjacentes*. Um vértice que não é adjacente a nenhum outro será denominado de *vértice isolado*.

Dado $S_n = \{(a_1, a_2, \dots, a_n) : a_i \in \{0, 1\}\}$, define-se o grafo G , tal que, $VG = S_n$ sendo que, dois vértices em VG são *adj*

centes sse diferem em exatamente uma posição, como sendo o grafo associado a S_n .

Veja por exemplo o grafo G abaixo, para $n=3$.

Um *grafo dirigido* D é um par ordenado (V, A) , formado por um conjunto finito V de vértices e um conjunto finito A de pares ordenados de elementos distintos de V , chamados *arcos*.

Se $\alpha = (u, v)$ é um arco, na representação do grafo dirigido, α é indicado por um segmento orientado de u para v , unindo os pontos que representam os vértices u e v . Dizemos que u é *origem* de α e v é *extremidade* de α .

Se um vértice não é origem de nenhum arco dizemos que é um *vértice terminal*.

Sejam $VD \equiv \{v^0, v^1, \dots, v^n\}$ e $ED \equiv \{\alpha_0, \alpha_1, \dots, \alpha^m\}$. Um *ca-*

minho de G é uma sequência finita $(v^0, \alpha^0, \dots, v^k)$ com $k \geq 0$, onde os v^i 's são vértices distintos de D e $\alpha^i = (v^i, v^{i+1})$, $0 \leq i \leq k-1$, são arcos de D .

2. ÁRVORES

Uma *árvore* é um conjunto finito T de um ou mais elementos chamados de *nós*, tal que:

- a) existe um nó especial que pertence a T , que se chama *raiz* de T ;
- b) os nós restantes (excluindo a raiz) são particionados em $m \geq 0$ conjuntos disjuntos $T_1, T_2, \dots, T_m$, cada um dos quais é por sua vez uma *árvore*.

As árvores $T_1, T_2, \dots, T_m$, são chamadas *subárvores* da raiz.

Cada raiz se diz *pai* das raízes de suas subárvores e estas se dizem *filhos* daquela.

Define-se *nível* de um nó de uma árvore pela relação de recorrência:

- a) o nível da raiz é zero;
- b) se o nível de um nó é n , o nível de seus filhos é $n+1$.

Uma *árvore binária* é um conjunto finito de nós que, ou é vazio, ou consiste de uma raiz e duas árvores binárias disjuntas chamadas de subárvore à esquerda e subárvore à direita da raiz.

3. CONJUNTOS CONVEXOS

Um subconjunto S de R^n é chamado de *conjunto afim* se $(1-\lambda)x + \lambda y \in S$, $\forall x \in S, y \in S$ e $\lambda \in R$. A intersecção de todos os conjuntos afins que contêm S é chamada de *envoltória afim* de S e será denotada por $\text{aff } S$.

Um subconjunto C de R^n é chamado de *conjunto convexo* se $(1-\lambda)x + \lambda y \in C$, $\forall x \in C, y \in C$ e $0 < \lambda < 1$.

Para qualquer $a \in R^n$, a bola B com raio $\epsilon > 0$ e centro em a é dada por:

$$\{x: d(x, a) \leq \epsilon\} = a + \epsilon B,$$

onde $d(x, a)$ é a *distância Euclidiana* entre os pontos x e a .

Para qualquer conjunto C em R^n , o conjunto de pontos x cuja distância de C não excede ϵ será indicado por:

$$\{x: \exists y \in C, d(x, y) \leq \epsilon\} = \{y + \epsilon B: y \in C\} = C + \epsilon B.$$

O *fecho* de C , que será denotado por $\text{cl } C$, é dado pelo seguinte conjunto:

$$cl\ C = \cap \{C + \epsilon B : \epsilon > 0\}.$$

O interior relativo de C , que será denotado por riC é dado pelo seguinte conjunto:

$$riC = \{x \in aff\ C : \exists \epsilon > 0, (x + \epsilon B) \cap (aff\ C) \subset C\}.$$

A diferença $(cl\ C) \setminus (riC)$ é chamada de *fronteira relativa* de C e será denotada por $relbd\ C$.

Uma *face* de C é um subconjunto convexo C' de C , tal que todo segmento de reta em C com um ponto em seu interior relativo pertencente a C' , tem as extremidades em C' .

Um ponto $x \in C \subset R^n$ é um *ponto extremo* sse não pode ser expresso como uma combinação convexa $(1-\lambda)y + \lambda z$, tal que $y \in C, z \in C$ e $0 < \lambda < 1$.

BIBLIOGRAFIA

- /1/ - Balas, E., "Discrete Programming by the Filter Method", Operations Research, Vol. 15 (1967), 915-957.

- /2/ - Bitran, G.R., "Linear Multiple Objective Programs with Zero-one Variables", Mathematical Programming, 13 (1977), 121-139.

- /3/ - Ecker, J.G. e Kouada, I.A., "Finding All Efficient Extreme Points for Multiple Objective Linear Programs", Mathematical Programming, 14 (1978) 249-261.

- /4/ - Evans, J.P. e Steuer, R.E., "A Revised Simplex Method for Linear Multiple Objective Programs", Mathematical Programming, 5 (1973), 54-72.

- /5/ - Geoffrion, A.M., Dyer J.S. e A. Feinberg, " An Interactive Approach for Multi-Criterion Optimization, with an Application to the Operation of an Academic Department", Management Science, Vol.

19, Nº 4, Dezembro (1972), 357-369.

- /6/ - Harary, F., Graph Theory, Addison-Wesley, Reading ,
Mass., (1972).
- /7/ - Knuth, D.E., The Art of Computer Programming, Vol.1,
Addison-Wesley, New York, (1975).
- /8/ - Mangasarian, O.L., Nonlinear Programming, McGraw Hill,
New York, (1969).
- /9/ - Pasternak, H. e Passy, U., "Bicriterion Functions in
Annual Activity Planning", Sixth International
Conference on Operations Research, Dublin (1972).
- /10/ - Pasternak, H. e Passy, U., "Bicriterion Mathematical
Programs with Boolean Variables", Multiple Cri-
teria Decision Making, J.L. Cochrane and M.Zeleny,
eds., University of South Carolina Press, (1973) ,
327-348.
- /11/ - Pasternak, H. e Passy, U., "Finding Global Optimum of
Bicriterion Mathematical Programs", Cahiers Centre
Etudes Rech. Oper., Vol. 16, (1974), 67-80.
- /12/ - Rockafellar, R.T., Convex Analysis, Princeton Univer-

sity Press, (1970).

- /13/ - Roy, B., "Problems and Methods with Multiple Objective Functions", Mathematical Programming 1, (1971), 239-266.
- /14/ - Shapiro, J.F., "Multiple Criteria Public Investment Decision Making by Mixed Integer Programming", Proceedings of the Conference on Multi-criterion Decision Making, Jouy En Josas, Maio (1975).
- /15/ - Steuer, R.E., "Multiple Objective Linear Programming with Interval Criterion Weights", Management Science, Vol. 23, Nº 3, Novembro, (1976) 305-316.
- /16/ - Villareal, B. e Karwan, M.H., "Dynamic Programming Approaches for Multicriterion Integer Programming", Technical Report Nº 78-3, State University of New York at Buffalo, Maio (1978).
- /17/ - Yu, P.L., e Zeleny, M., "The Set of All Nondominated Solutions in Linear Cases and a Multicriteria Simplex Method", Journal of Mathematical Analysis and Applications, 49, (1975), 430-468.
- /18/ - Zionts, S., "An Interactive Method for Evaluating

Discrete Alternatives Involving Multiple Criteria", Working Paper Nº 271, State University of New York at Buffalo, Julho (1976).

/19/ - Zionts, S. e Wallenius, J., "An Interactive Programming Method for Solving the Multiple Criteria Problem", Management Science, Vol. 22, Nº 6, Fevereiro (1976), 652 - 663.

/20/ - Zionts, S., e Wallenius, "Identifying Efficient Vectors: some theory and computational results", Working Paper Nº 257, State University of New York at Buffalo, Março (1976).