UM ALGORITMO PARA UM PROBLEMA NÃO LINEAR MISTO-DISCRETO E ANÁLISE COMPARATIVA

HSING PEI CHIN

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

APROVADA POR:

Prof. Susana Scheimberg de Makler DsC.

Prof. Paulo Roberto Oliveira Dr.Ing

Prof. Sérgio Granville Ph.D

RIO DE JANEIRO, RJ - BRASIL

ABRIL DE 1995

HSING, PEI CHIN

Um algoritmo para um problema não linear misto-discreto e análise comparativa [Rio de Janeiro] 1995.

X, 146 p., 29.7 cm(COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1995)

Tese - Universidade Federal do Rio de Janeiro, COPPE.

Aos meus pais e
a toda minha família

Agradecimentos

À professora Susana Scheimberg de Makler , pela orientação competente e dedicada, pelo incentivo ao meu desenvolvimento acadêmico e pela sua amizade, paciência e confiança.

Aos meus amigos Sérgio Maurício e Chen Ying Ling que deram incansáveis assistências nas horas mais desesperadas na implementação da tese.

Ao Maurício David, Plácido e Ricardo Arantes, pelas suas colaborações e sugestões no trabalho da tese e na aquisição dos pacotes de programas usados na tese.

Às minhas colegas Ana Paula, Lujan, Maria Tereza e Clícia pela valiosa amizade.

A todos aqueles que de alguma forma contribuíram para a elaboração desta tese.

Resumo apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciência (M.Sc.)

Um Algoritmo Para Um Problema Não Linear Misto-Discreto e Análise Comparativa

Hsing Pei Chin

Abril , 1995

Orientador : Susana Scheimberg de Makler

Programa : Engenharia de Sistemas e Computação

Neste trabalho, pretende-se estudar, implementar e comparar três algoritmos de linearização para resolver problemas de otimização não linear, onde algumas variáveis tomam valores contínuos e outras valores discretos.

O primeiro a ser analisado é desenvolvido por Duran e Grossmann, denomina-se o *Algoritmo De Aproximação Externa*, para resolver numericamente uma classe de programação não linear nas variáveis contínuas e linear nas variáveis discretas.

O segundo algoritmo é o de *Aproximação Linear Sequencial*, deselvolvido por Loh e Papalambros, é um algoritmo para uma classe mais geral de funções.

Ambos os algoritmos serão apresentados e analisados mais detalhadamente na introdução da tese.

O terceiro algoritmo é o que nós daremos um estudo mais aprofundado, chamaremos de *Algoritmo De Aproximação Poliédrica Com Região De Confiança*, pois é elaborado a partir das vantagens existentes nos dois métodos anteriores, com o objetivo de eliminar as dificuldades que cada um dos dois algoritmos anteriores apresenta.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirement for the degree of Master of Science (M.Sc.)

The Algorithm For a Mixed-Discrete Nonlinear Problem And Comparative Analysis

Hsing Pei Chin

April , 1995

Thesis Supervisor : Susana Scheimberg de Makler

Department : Computing and Systems Engineering

In this work, there is a intention of studing, implementing and comparing three linearization algorithms destinated to a class of nonlinear optimization problems, where some of the variables can assume continuous values and the others discretes values.

The first algorithm to be analyzed is developed by Duran & Grossmann, known as an *Outer Aproximation Algorithm,* which has a objective to solve a particular class of nonlinear programming, where the underlying mathmatical structure is linearity of the integer variables and nonlinearity of the continuous variables.

The second algorithm is *Sequetial Linearization Approch,* elaborated by Loh & Papalambros.This is a algorithm developed for a more general class of functions.

Both of the algorithms will be presented and analyzed with more details at this thesis' introductionThe third algorithm is the one that we will present with more enphasis, because it is elaborated by gathering what we have of advantages of both previous algorithms, and it is also aimmed to eliminate difficulties presented by the first two algorithms.

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Serão apresentados nesta fase inicial, dois algoritmos que darão uma base fundamental para o terceiro algoritmo. Estudaremos em cada um deles os seguintes assuntos:

i) Estrutura matemática dos problemas não lineares misto-discretos.

ii) O método de aproximação.

iii) O algoritmo.

iv) A convergência do algoritmo.

1.1 Algoritmo De Aproximação Externa

i) Estrutura matemática dos problemas não lineares mistos-discretos

Este algoritmo foi apresentado por Duran-Grossmann [1] para resolver uma classe particular dos problemas de programação não linear mista-discreta, onde envolvem ambas as variáveis contínuas (x) e variáveis discretas (y). A característica principal da estrutura matemática é a separabilidade dos dois tipos de variáveis, a linearidade das variáveis inteiras e a convexidade das funções não

lineares que envolvem as variáveis contínuas. Podemos representar esta classe pelo seguinte problema da programação não linear mista-discreta (PNLMD).

$$Z = \min c^T y + f(x)$$

s.a

$$g(x) + By \leq 0 \qquad (P)$$

$$x \in \chi \subset R^N$$

$$y \in U \subset R^M_+$$

Onde $f: R^N \rightarrow R$ e aqueles que compõem a função vetor g: $R^N \rightarrow R^p$ são todas funções não lineares, contínuamente diferenciáveis e convexas sobre um conjunto aberto que contém o conjunto convexo poliédrico compacto de dimensão n $\chi = \{ x : x \in R^N, A_1 x \leq a_1 \}$; $U = \{ y : y \in Y,$ inteiro , $A_2 y \leq a_2 \}$ é um conjunto discreto finito, para a maioria das aplicações, o Y corresponde ao hipercubo unitário $Y = \{ 0 , 1 \}^M$. B, A_1, A_2 e C, a_1 , a_2 são matrizes e vetores respectivamente de dimensões adequadas.

ii) O Método De Aproximação:

A idéia básica da aproximação externa apóia-se na descrição da região de soluções viáveis de um dado problema como sendo a intersecção de uma coleção infinita de conjuntos, ou seja, o algoritmo é baseado na caracterização de um conjunto convexo através da intersecção dos hiperplanos suportes [12]. O objetivo da aproximação é de providenciar uma representação poliédrica do conjunto viável do problema (P), onde a tal representação resultará na linearidade das variáveis contínuas, permitindo então a aproximação do problema de (PNLMD) pelo problema mais simples de programaçào linear mista-discreta (PLMD).

Para podermos fazer esta substituição de (PNLMD) por (PLMD), começamos então linearizando a função objetivo. Como f é uma função convexa , então [f - μ] também é convexa , onde μ é uma variável escalar , sem perda de generalidade, podemos escrever o problema (P) da sequinte forma :

$$Z = \min c^T y + \mu$$

s. a

$$f(x) - \mu \leq 0 \qquad (P0)$$

$$g(x) + B y \leq 0$$

$$x \in \chi$$

$$y \in U$$

$$f_L \leq \mu \leq f_U$$

Onde $f_L = \min \{ f(x) : x \in \chi \}$ e $f_U = \max \{ f(x) : x \in \chi \}$. Assumimos que a condição de Slater seja satisfeita em relação à variável contínua, ou seja, existe um ponto $x \in \chi$ tal que $g(x) + B y < 0$ para cada $y \in U \cap V$, onde:

$$V = \{ y : g(x) + B y \leq 0 \text{ para algun } x \in \chi \}$$

Consideremos a função ponto-conjunto F de $U \cap V$ em $\chi \times [f_L, f_U]$, definida por :

$$F(y) = \{ (x, \mu), x \in \chi, \mu \in [f_L, f_U], f(x) - \mu \leq 0, g(x) + B y \leq 0 \}.$$

Isto é, $F(y)$ é o conjunto viável nas variáveis contínuas associado a cada $y \in U \cap V$. Podemos finalmente verificar que devido a convexidade das funções f e g, e ao fato do conjunto χ ser compacto, $F(y)$ é um conjunto convexo e fechado para cada $y \in U \cap V$. Logo $F(y)$ pode ser considerado como sendo a intersecção dos semi-espaços suportes fechados. Mais ainda, devido à convexidade e diferenciabilidade das funções, tem-se a seguinte caracterizaçào de $F(y)$:

$$F(y) = \cap_{x^i \in \chi} \{ (x, \mu) \in \chi \times [f_L, f_U] / f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \leq 0, \; g(x^i) + \nabla g(x^i)^T (x - x^i) + B y \leq 0 \}$$

Onde $\nabla f(x^i)$ e $\nabla g(x^i)$ são vetores gradientes de f e g, calculados no ponto $x^i \in \chi$.

Logo, podemos definir a região viável para o problema (P0) por :

$$
\begin{cases}
0 \geq f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \\
0 \geq g(x^i) + \nabla g(x^i)^T (x - x^i) + B\,y \qquad \text{todo } x^i \in \chi \qquad (1) \\
\\
x \in \chi\,,\ y \in U,\ \mu \in [f_L\,,\ f_U]
\end{cases}
$$

As figuras abaixo correspondem a exemplos de aproximação externa considerando um número finito de pontos :

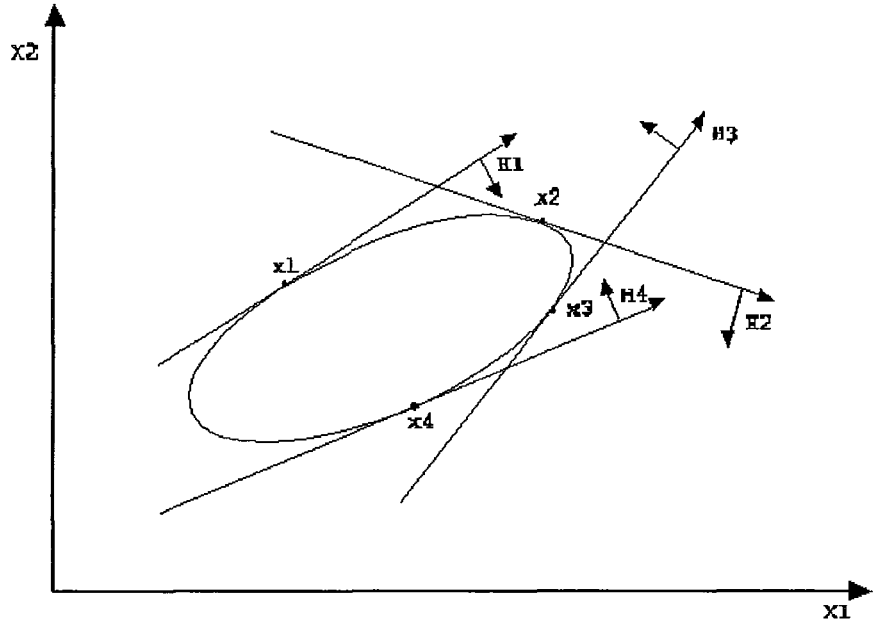


Fig 1. Aproximação externa de um conjunto convexo em duas dimensões. Onde H1, H2, H3 e H4 são semi-espaços que contém a região.

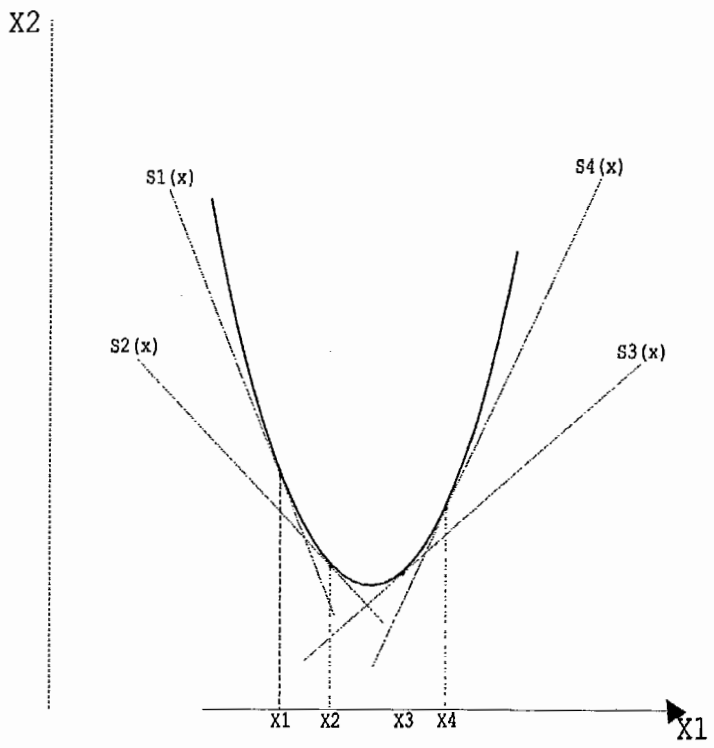


Fig 2. Exemplo aplicado a função de uma variável.

As restrições expressados em (1) é equivalente às representações das funções convexas f e g pela função máxima da coleção de seus suportes lineares. O que pode ser visto nítidamente em figura 2.

A aproximação externa do espaço viável do problema (P0) descrito sob a forma de (1) resultará na linearidade das restrições e da função objetivo de (P). Substituimos então (P0) pelo seguinte problema linear misto-inteiro infinito :

$$Z = \min c^t y + \mu$$

s.a

$$f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \leq 0 \qquad (P1)$$

$$g(x^i) + \nabla g(x^i)^T (x - x^i) + B y \leq 0 \qquad \text{todo } x^i \in \chi$$

$$x \in \chi, y \in U, \mu \in [f_L, f_U]$$

O conjunto V acima fica caracterizado por :

$V \equiv \{y : g(x^i) + \nabla g(x^i)^T (x - x^i) + B\,y \leq 0, \forall\, x^i \in \chi$, para algum x $\in \chi$ } . Este conjunto submerso no conjunto viável de P1 no sentido que , se $y \in V \Rightarrow$ então existem $x \in \chi$, $\mu = f(x)$ tal que (y , x , μ) é um ponto viável para (P1).

O seguinte lema estabelece a equivalência entre os problemas (P0) e (P1):

LEMA 1: Sob as hipóteses considerado para as funções e os conjuntos , os problemas (P0) e (P1) são equivalentes , (x , y) $\in \chi \times U \times [f_L, f_U]$ é solução de (P1).

Apesar de (P1) ser um problema de programação linear mista discreta , ele envolve um número infinito de restrições, objetivando a solução desta dificuldade e levando em consideração que o conjunto $U \cap V$ é finito e discreto, introduzimos agora a idéia de projeção do problema (P) sobre o conjunto de $U \cap V$ para podermos identificar quais são os pontos x^i que serão considerados na aproximação. A projeção do (P) sobre y é :

$$Z = \min_{y} [\ \text{ínfimo}_{x \in \chi} \{ c^T y + f(x) : g(x) + B\,y \leq 0 \}\ ]$$

s.a

$y \in U \cap V$

A parte interior dos colchetes equivale a para cada y^i pertencente ao $U \cap V$ fixo, resolver o problema :

$$Z(y^i) = c^T y^i + \min f(x)$$

s.a ($S(y^i)$)

$g(x) + B\,y^i \leq 0$

$x \in \chi$

Para y^i ser um candidato a solução ótima do problema original (P),

$S(y^i)$ tem que ser viável ($y^i \in U \cap V$) e existir um ponto x^i que é solução ótima associado ao subproblema $S(y^i)$. Pelos teoremas de caracterização dos poliedros inteiros e pela teoria de programação linear, y^i é um vértice da envoltória convexa de $U \cap V$, e x^i é um vértice da região viável linear.

Então a forma final do problema poderá ser dado da seguinte forma :

$$Z = \min c^t y + \mu$$

s.a

$$f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \leq 0 \qquad \forall i \in T \qquad (M^k)$$

$$g(x^i) + \nabla g(x^i)^T (x - x^i) + B y \leq 0$$

$$x \in \chi , y \in U , \mu \in [f_L, f_U]$$

Onde $T = \{ i , x^i$ é uma solução ótima de $S(y^i), y^i \in U \cap V \}$. Este (PLMD) chamaremos de problema " master ".

Teoricamente este problema master envolve somente um número finito de restrições associados aos finitos pontos x^i, mas existe uma dificuldade computacional , a de exigir a predeterminação de todos os pontos ótimos x^i associados à cada $S(y^i)$, $y^i \in U \cap V$. Para superar isto , na prática , usamos na verdade , uma versão relaxada do problema master , na iteração k , temos o seguinte problema :

$$Z^k = \min c^t y + \mu$$

s.a

$$(x , y) \in \Omega^K$$

$$x \in \chi , y \in U , \mu \in [f_L, f_U]$$

Onde

$$\Omega^K = \{(x , y) : f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \leq 0$$

$$g(x^i) + \nabla g(x^i)^T (x - x^i) + B y \leq 0 , \text{todo } i \in T^k \subseteq T \}$$

$T^k = \{ i : x^i$ é uma solução ótima de $S(y^i)$, i = 1,2, ...,k}

A utilização da versão relaxada do problema master implica em :

1) Resolver o problema M^k. Se a solução de $M^k(x, y^{k+1})$ não satisfizer a algum teste de parada, resolver $S(y^{k+1})$ para determinar o ponto x^{k+1} a ser considerada na nova aproximação.

2) O novo problema M^{k+1} é construido fazendo a intersecção do espaço viável de M^k com o conjunto de semi-espaços fechados associados a x^{k+1}.

iii) O Algoritmo

Antes de nós entrarmos no algoritmo propriamente dito, fazemos algumas observações teórica, a fim de poder na prática, melhorar o desempenho do mesmo.

OBSERVAÇÃO 1 : Se denotamos G e Γ^k como sendo os espaços viáveis do problema original (P0) e de M^k respectivamente, sabemos que $G \subseteq \Gamma^k$, para todo $k \geq 1$, pois os Γ^k sobreestimam o G, então :

$$G \subseteq \Gamma^k \subseteq \Gamma^{k-1} \subseteq \dots \subseteq \Gamma^1 \Rightarrow$$

$$\Rightarrow \min \{ C^t y + \mu : (x, y) \in G, \mu \in [f_L, f_U] \} \geq$$

$$\geq \min \{ C^t y + \mu : (x, y) \in \Gamma^k, \mu \in [f_L, f_U] \} \geq$$

$$\geq \dots \geq$$

$$\geq \min \{ C^t y + \mu : (x, y) \in \Gamma^1, \mu \in [f_L, f_U] \}$$

$\Rightarrow$ Os valores ótimos Z^k determinados pelo M^k, para todo k, formam uma sequência monótona não decrescente que servem como limites inferiores para o valor ótimo do problema original (P0).

Por outro lado, para cada $y^i \in U \cap V$ fixo, temos :

$$Z = \min \{ C^t y + f(x) : (x, y) \in G \} \leq Z(y^i) =$$

$$= \min \{ C^t y + f(x) : x \in \chi, g(x) + B y^i \leq 0 \}.$$

Isto é $Z(y^i)$ é um limite superior para o valor ótimo de (P), Z^*.

Usando estas duas propriedades , podemos substituir a restrição $\mu \in [f_L , f_U]$ por :

$$Z^{k-1} \leq C^t y + \mu < Z^*$$

Onde $Z^{k-1} = \min \{ C^t y + \mu : (x , y) \in \Gamma^{k-1} \}$ e Z^* é o melhor limite superior atualizado , $Z^* = \min \{ Z (y^i) , i = 1, 2,....., k \}$.

OBSERVAÇÃO 2 : Como foi visto antes , a condição de $y \in U \cap V$ é necessária para que S (y) seja viável . Mas com o problema master relaxado , os subconjuntos V^k de V são gerados automáticamente pelo processo do algoritmo , sob a forma de:

$V^k = \{ y \in U : g (x^i) + \nabla g (x^i)^T (x - x^i) + B y \leq 0 ,$ para todo $i \in T^k$, para algum $x \in \chi \}$.

Sendo :

$V = \{ y \in U : g (x^i) + \nabla g (x^i)^T (x - x^i) + B y \leq 0 ,$ para todo $i \in T$, para algum $x \in \chi \}$.

Para cada iteração k , V^k é uma sobrestimativa de V , então não é possível garantir que y selecionado no problema master M^k faça com que o subproblema não linear associado $S (y^i)$ seja viável . Quando isso acontecer , um dos procedimentos normais para manter as propriedades dados na observação 1 é de eliminar este y^i. A maneira mais simples de fazer isso é acrescentar ao V^k, as restrições de semi-espaços para construir um V^{k+1} mais restrito . O próprio subproblema $S (y^i)$ serve para testar se $y^i \in V^k$ está ou não em V . Mesmo $S (y^i)$ não sendo viável , qualquer algoritmo de programação não linear fornecerá um x^i associado . Então o conjunto de restrições que será acrescentado ao V^k para formar V^{k+1} é :

$$f (x^i) + \nabla f (x^i)^T (x - x^i) - \mu \leq 0$$

$$g (x^i) + \nabla g (x^i)^T (x - x^i) + B y \leq 0$$

Eliminando assim o ponto (x^i, y^i) de Ω^{k+1}. Mas o ponto y^i pode voltar a ser considerado novamente . Para garantir que ele não seja selecionado de novo nas futuras iterações , o método mais utilizado é um corte inteiro [1].

Feitas estas duas observações , podemos agora analisar melhor o algoritmo:

DEFINIÇAO : Para $x^i \in R^n$

$$C(x^i) = \{ x, y : f(x^i) + \nabla f(x^i)^T (x - x^i) - \mu \leq 0$$
$$g(x^i) + \nabla g(x^i)^T (x - x^i) + B y \leq 0, \mu \in R^1 \}$$

PASSO 1 : $\Omega^0 = R^n \times R^m$

limite inferior $Z^0 = - \infty$

limite superior $Z^* = + \infty$

$i := 1$

selecionar uma combinação inteira $y^1 \in U$, $y^1 \in U \cap V$ se for possível.

PASSO 2 : Resolver o subproblema não linear $S(y^i)$.

$$Z(y^i) = c^T y^i + \min f(x)$$

s.a

$$g(x) + B y^i \leq 0$$

$$x \in \chi$$

Se $S(y^i)$ for viável então $Z^* = \min \{ Z^*, Z(y^i) \}$

se $Z^* = Z(y^i)$ então $y^* = y^i$, $x^* = x^i$, $\Omega^i = \Omega^{i-1} \cap C(x^i)$

e ir ao passo 3.

Senão adicione ao M^i o corte de inteiro para eliminar o y^i

faça $\Omega^i = \Omega^{i-1} \cap C(x^i)$

onde x^i está associada ao problema não viável $S(y^i)$.

PASSO 3 : Resolver o problema master relaxado M^i (PLMD) :

$$Z = \min c^t y + \mu$$

s.a

$$(x, y) \in \Omega^i$$

$$Z^{k-1} \leq C^t y + \mu \leq Z^*$$

$$x \in \chi, y \in U, \mu \in R^1$$

$y \in$ (conjunto de cortes de inteiros)

Se o problema M^i não tiver solução viável mista-inteira então PARE.

A solução ótima para o problema original P é dado pelo atual limite superior Z^* e o vetor (x^*, y^*) correspondente a solução ótima do subproblema não linear definido no passo 2 .

Senão M^i tem solução ótima correspondente (Z^i, x, y). Z^i é um elemento da sequência monótona dos limites inferiores da solução ótima do (PLMD) original .

Faça $y^{i+1} = y$, $i = i + 1$ e retorne ao passo 2.

iv) Convergência

A convergência do algoritmo da aproximação externa está baseado em dois critérios :

1) O primeiro deve-se à propriedade de limites inferiores do processo. A condição $Z^{i-1} \leq C^t y + \mu \leq Z^*$ no i-ésimo problema master relaxado M^i é violada no passo 3 , se não houver solução viável mista discreta , o que implica que $c^t y + \mu \geq Z^* \Rightarrow Z^i \geq Z^*$, indicando o cruzamento do limite inferior com o limite superior , convergindo assim o algoritmo.

2) Devido ao conjunto discreto U ser finito , o que implica que existe apenas um número finito de combinações inteiro y , fazendo com que o algoritmo de aproximação externa termine em um número finito de iterações, pois como foi

observado anteriormente , os pontos são relacionados como máximo uma única vez .

1.2 Algoritmo de Aproximação Linear Sequencial

i) Estrutura matemática do problema não linear misto discreto

O algoritmo de aproximação linear sequencial destina-se a resolver problemas de programação não linear mista discreta que nào tem uma estrutura especial explícita em relação às funções . Para analisarmos melhor o modelo de PNLMD apresentado por Loh e Papalambros [4] , é necessário introduzir antes a seguinte definição:

DEFINIÇÃO 1 : Seja S um conjunto aberto não vazio de R^n .

Uma função $f : S \subset R^n \rightarrow R$ função diferenciável em S ,

é pseudoconvexa se para todo x1, x2 $\in$ S , tem-se :

(a) $\nabla f(x1)^t (x2 - x1) \geq 0 \Rightarrow f(x2) \geq f(x1)$

A função f é estritamente pseudoconvexa se para todo x1, x2 verifica-se :

(b) $\nabla f(x1)^t (x2 - x1) \geq 0 \Rightarrow f(x2) > f(x1)$

Ou equivalentemente :

(a') $f(x2) < f(x1) \Rightarrow \nabla f(x1)^t (x2 - x1) < 0$

(b') $f(x2) \leq f(x1) \Rightarrow \nabla f(x1)^t (x2 - x1) < 0$

Observamos que o conceito de pseudoconvexidade é mais fraca que o de convexidade , isto é , se uma $f : S \subset R^n \rightarrow R$ diferenciável é convexa (estritamente) então é pseudoconvexa (estritamente) .

Tendo esta noção de pseudoconvexidade , podemos agora apresentar o modelo matemático de Loh e Papalambros , dado por :

$$
\begin{aligned}
&\text{Minimizar } f(x, y) \\
&\text{s.a} \\
&\quad g_j(x, y) \leq 0 \qquad j = 1, \ldots\ldots, p \\
&\quad L_i \leq x_i \leq U_i \qquad i = 1, \ldots\ldots, n \qquad (P) \\
&\quad l_k \leq y_k \leq u_k \qquad k = 1, \ldots\ldots, m \\
&\quad x \in R^n, y \in D \subseteq S \subseteq R^m_+
\end{aligned}
$$

Onde $S \subseteq R^m_+$ é um conjunto aberto que contém o conjunto discreto D. A vizinhança aberta S que contém o espaço discreto foi acrescentado ao trabalho original do Loh e Papalambros , para possibilitarmos trabalhar com as funções $f : S \times R^n \rightarrow R$ e $g_j : S \times R^n \rightarrow R$ $j = 1, \ldots\ldots, p$ diferenciáveis de classe C^1 . Nenhuma outra exigência foi feita às funções restrições $g_j(x, y)$. No entanto precisamos notar que no caso destas restrições não serem convexas , o algoritmo poderá falhar. Experimentalmente foram testados problemas do tipo (P) com restrições não convexas , alguns foram resolvidos com sucesso e outros não.

O ponto contínuo x é n-dimensional e y é a variável discreta de dimensão m . R denota o espaço real contínuo . E cada componente x_i e y_k são limitados inferiormente e superiormente por L_i , U_i e l_k , u_k respectivamente .

ii) O método de aproximação

DEFINIÇÃO 2 : Dado um problema não linear misto discreto (P) seu modelo linear misto discreto (PLMD) associado relativo ao um ponto (x_0, y_0) é dado por :

$$\text{Minimizar } \nabla f(x_0, y_0)^t (x - x_0, y - y_0)$$

s. a

$$(\text{PLMD}(x_0, y_0)) \qquad g_j(x_0, y_0) + \nabla g_j(x_0, y_0)^t (x - x_0, y - y_0) \leq 0$$

$$j = 1, \ldots\ldots\ldots, p$$

$$x \in R^n$$

$$y \in D \subseteq S \subset R^m_+$$

O algoritmo técnico básico SLP 1 é o seguinte :

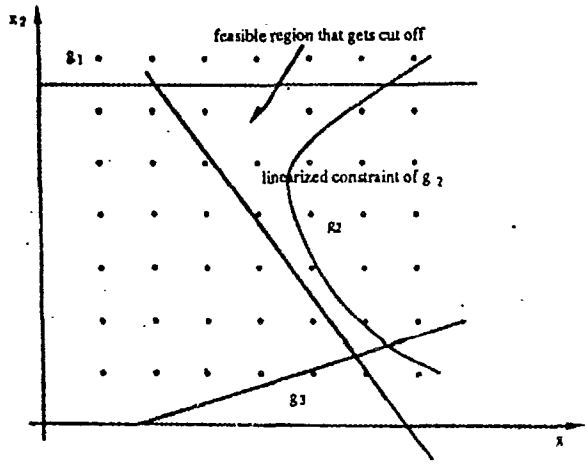
1. Dado (x_0, y_0) $k = 0$, $\delta > 0$.
2. Construir o PLMD (x_k, y_k).
3. Resolver o PLMD (x_k, y_k). Usando o método simplex e o branch and bound de Dakin [2] .

   Seja (x_{k+1}, y_{k+1}) a solução ótima de PLMD (x_k, y_k).
4. Se $||(x_{k+1}, y_{k+1}) - (x_k, y_k)|| < \delta$, PARE .

   Senão seja $k = k+1$ e volta ao passo 2 .

A idéia principal deste algoritmo é tentar achar a solução ótima do problema original (P) através de sucessivas linearizações da função objetivo e das restrições que limitam a região das soluções viáveis . Diferentes linearizações das restrições resultam em diferentes regiões viáveis . Quando as restrições são lineares , as linearizações são as próprias restrições , mantendo assim a região de soluções viáveis . Quando as restrições são funções convexas , as linearizações são os hiperplanos suporte das restrições no ponto considerado (x_k, y_k) do problema original relaxado , isto é , considerando todas as variáveis contínuas . Neste caso , a região viável do (PLMD) contém a região viável do problema original . Mas quando as restrições não são convexas , o caso é mais complexo , pois quando fizermos uma linearização deste tipo , uma parte da região viável original pode não ser considerada .

Olhe a figura abaixo :



g2 linearizad

Fig 3.

A parte hachurada é a parte da região de soluções viáveis que vai deixar de ser considerada , podemos correr o risco de não achar a solução ótima se estiver contido nela . Entretanto , este algoritmo oferece uma solução para este problema pelo seu próprio procedimento , pois é possível recuperar a região de soluções viáveis não levada em conta num passo na linearização de uma outra iteração , devido ao algoritmo ter voltado ao PNLMD original e linearizá-lo em relação ao um outro ponto , dando a possibilidade assim de reconsiderar a região anterior que foi cortada , mas sacrificando uma outra parte da região .

Podemos visualizar melhor o funcionamento do algoritmo SLP 1 , dando um exemplo :

Exemplo 1 : Cosidere o seguinte problema : (Grupta , 1980)

$$\text{Minimize } f(y) = (y1-8)^2 + (y2-2)^2$$

s.a

$$g1(y) = 0.1\, y1^2 - y2 \leq 0 \qquad (1)$$

$$g2(y) = 1/3\, y1 + y2 - 4.5 \leq 0$$

$$y1 , y2 \text{ inteiros}$$

A solução contínua ótima com 3 casas decimais do problema relaxado é $(5.245 , 2.75)^t$. Podemos dar um y0 inicial como sendo o arredondamento para o inteiro mais próximo $(5 , 3)^t$.

Este ponto não é viável para (1) . Linearizando (1) em relação ao ponto $(5 , 3)^t$ resultará o seguinte problema :

Minimize -6 y1 + 2 y2

s.a

$y1 - y2 - 2.5 \leq 0$ (2)

$1/3\ y1 + y2 - 4.5 \leq 0$

y1 , y2 inteiros

Resolvendo este problema encontramos a solução no ponto viável $(4 , 2)^t$ com $f(4 , 2) = 16$.

Voltando ao (1) , linearizando-o em relação ao $(4 , 2)^t$, teremos o seguinte problema :

Minimize -8 y1

s.a

$0.8\ y1 - y2 - 1.6 \leq 0$ (3)

$1/3\ y1 + y2 - 4.5 \leq 0$

y1 , y2 inteiros

Resolvendo este problema , achamos a solução ótima novamente o ponto $(4 , 2)^t$, o que implica o término do processo de linearização . Pelo teorema de convergência que vai ser estudado depois , este ponto será o minimizador global do problema misto discreto original .

o procedimento feito pelo algoritmo poderá ser analisado gráficamente para este exemplo :

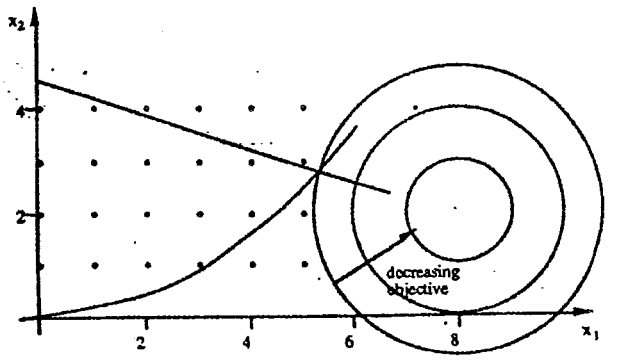


Fig . 4.1 Problema não linear

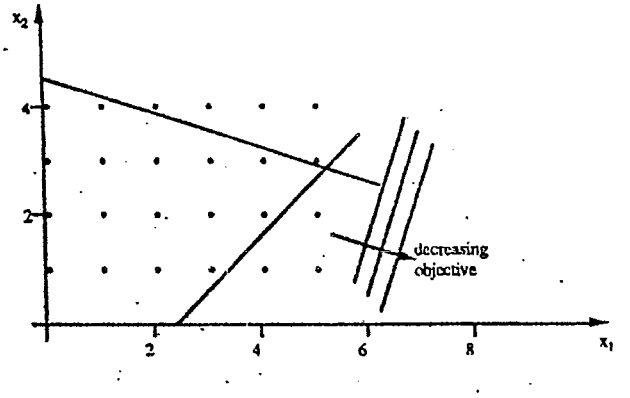


Fig . 4.2 Aproximação linear
no ponto $(5 , 3)^t$.

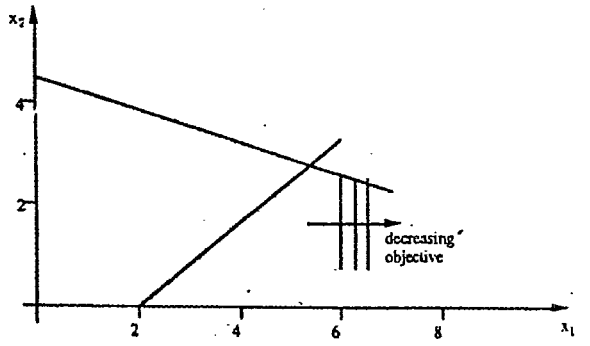


Fig . 4.3 Aproximação liner no ponto (4 , 2)^t.

Este algoritmo SLP 1 não é muito eficiente , pois para certos problemas , ele pode entara em ciclos , oscilando entre dois pontos fixos , consequentemente o algoritmo não vai finalizar em um determinado ponto , como é mostrado no exemplo seguinte de uma dimensão :

Minimizar $f(y) = (y - 3)^2$

s.a

$y \leq 4$

$y \geq 2$

y inteiro

É muito fácil de verificar que o algoritmo SLP 1 entrará em cíclos nos pontos y = 2 e y = 4 . Notemos que a função objetiva nestes dois pontos tem o mesmo valor . Para evitar que isto aconteça , pode-se fazer duas modificações que garantem que o procedimento gere pontos diferentes .

1) Durante o procedimento de linearização sucessiva , movemos de um ponto x1 para um ponto x2 apenas se houver um decréscimo na função objetivo não linear , ou seja , somente se f (x2) < f (x1) .

2) Incorporar limite de passo decrescente.

Então modelamos o PNLMD na seguinte forma :

Minimizar $\nabla f(x_0, y_0)^t (x - x_0, y - y_0)$

s. a

$$g(x_0, y_0) + \nabla g(x_0, y_0)^t (x - x_0, y - y_0) \leq 0 \quad (M)$$

$$|| (x - x_0, y - y_0) ||_\infty \leq t_0$$

$$x \in R^n$$

$$y \in D \subseteq S \subset R^m_+$$

O valor de t_0 decresce no decorrer do processo enquando não houver decréscimo do valor da função objetivo , e este modelo (M) é denominado de programação linear mista discreta restrita (PLMDR) .

Então o algoritmo SLP 1 melhorado , o SLP 2 é o seguinte :

1) Dado (x_0, y_0) , t_0 , $r_t > 0$, $\delta > 0$, seja $k = 0$.
2) Construir PLMDR (x_k, y_k) .
3) Resolver o PLMDR (x_k, y_k) usando simplex e Dakin's branch and bound .
   Seja $Z = (Z_x, Z_y)$ solução ótima de PLMDR (x_k, y_k) .
4) Se $|| Z - (x_k, y_k) || < \delta$ PARE .
5) Se $f(Z) < f(x_k, y_k)$, faça $k = k + 1$ e $Z = (x_k, y_k)$. Volte ao passo 2.
   Senão faça $t_0 = t_0 / r_t$, e volte ao passo 2 .

Podemos entender melhor o algoritmo SLP 2 através de um exemplo ilustrando o procedimento .

Exemplo : Considere o problema :

$$\text{Minimize } f(y) = 7\,y1^2 + 6\,y2^2 + 8\,y3^2 - 6\,y1\,y3 + 4\,y2\,y3 - 15.8\,y1 - 93.2\,y2 - 62\,y3 + 500$$

s. a

$$g1(y) = 142\,y1 + 172\,y2 + 118\,y3 \leq 1992$$

$$g2(y) = 98\,y1 + 114\,y2 + 44\,y3 \leq 1162$$

$$g3(y) = 40\,y1 + 72\,y2 + 34\,y3 \leq 703$$

$y1$, $y2$, $y3$ inteiros.

Seja $y0 = (3 , 6 , 3)^t$ viável e $f (3 , 6 , 3) = 83.4$, $t_0 = 5$, e $r_t = 2$. linearizando em y0 , obtemos :

Minimize 8.221 y1 - 9.164 y2 - 8.976 y3

s. a

$g1(y) = 142\ y1 + 172\ y2 + 118\ y3 \leq 1992$

$g2(y) = 98\ y1 + 114\ y2 + 44\ y3 \leq 1162$

$g3(y) = 40\ y1 + 72\ y2 + 34\ y3 \leq 703$

$-5 < y1 - 3 < 5$

$-5 < y2 - 6 < 5$

$-5 < y3 - 3 < 5$

y1 , y2 , y3 inteiros.

A solução ótima deste problema é o ponto $(1 , 5 , 8)^t$ sendo o valor da função objetivo original $f (1 , 5 , 8) = 296.8$. Logo , rejeitamos este ponto e consideramos $t_0 = 5/2 = 2.5$. Voltemos ao passo 3 do algorítmo . Resolvendo de novo o problema achamos um novo ponto $(1 , 7 , 4)^t$ sendo o valor da função objetivo não linear $f (1 , 7 , 4) = 96.8$. Rejeitamos novamente este ponto e reduzimos a vizinhança $t_0 = 2.5/2 = 1.25$. Voltamos ao passo 3 do algoritmo , resolvendo novamente o problema sem mudar o ponto y0 e com o novo limite $t_0 = 1.25$, obtemos como a solução ótima o ponto $(2 , 7 , 3)^t$ com $f (2 , 7 , 3) = 69$, aceitamos este ponto porque houve um decréscimo no valor da função objetivo . Linearizamos neste novo ponto , o novo problema será :

Minimize -5.786 y1 + 2.842 y2 + 1.024 y3

s. a

$g1(y) = 142\ y1 + 172\ y2 + 118\ y3 \leq 1992$

$g2(y) = 98\ y1 + 114\ y2 + 44\ y3 \leq 1162$

$g3(y) = 40\ y1 + 72\ y2 + 34\ y3 \leq 703$

$-5 < y1 - 2 < 5$

$-5 < y2 - 7 < 5$

$-5 < y3 - 3 < 5$ y1 , y2 , y3 inteiros.

A solução ótima é o ponto $(7 , 2 , 1)^t$ com $f(7 , 2 , 1) = 421$. Rejeitamos este ponto , decresce o t_0 para 2.5 , voltamos ao passo 3 do algoritmo e achamos um novo ponto $(4 , 5 , 1)^t$ com $f(4 , 5 , 1) = 173.2$, rejeitamos de novo , decresce t_0 para 1.25 , o novo ponto agora é $(3 , 6 , 2)^t$ com $f(3 , 6 , 2) = 90.4$, novamente , rejeitamos este ponto e decresce t_0 para 0.75 , onde finalmente achamos o ponto $(2 , 7 , 3)^t$ com $f(2 , 7 , 3) = 69$. O algoritmo convergiu .

iii) O algoritmo

Para analisarmos o algoritmo , precisamos introduzir um novo conceito de viabilidade aproximada . Quando a função é pseudoconvexa e as restrições g_j são convexas , as variáveis (x , y) viáveis do problema não linear original também são viáveis do modelo linear , o que não acontece se for o contrário . Visando em superar esta dificuldade introduzimos o conceito de ε-viabilidade , aplicado especialmente para o caso de variáveis mistas-discretas . Este conceito também nos ajudará na convergência do algoritmo .

DEFINIÇÃO 3 : A soma de inviabilidade (SUMINF) de um ponto é definido como :

$$\text{suminf}(x_i, y_i) = \sum g_j (x_i, y_i) \ , g_j (x_i, y_i) > 0 .$$

DEFINIÇÃO 4: Um ponto x1 é chamado de ε-viável para um problema de PNLMD se satisfiser :

$$\text{suminf}(x_i, y_i) \leq \varepsilon \text{ . Para um } \varepsilon > 0 .$$

Durante o processo de linearização sequencial , dado um $\varepsilon > 0$, um novo ponto só será aceito se este melhorar a ε-viabilidade do ponto atual , ou então o valor da função objetivo do ponto novo é melhor em relação ao ponto atual . Para ε fixo , existem apenas um número finito de pontos que podem resultar um melhor

valor da função objetivo . Com a diminuição progressiva do ε ($\varepsilon \rightarrow 0$) , os pontos são forçados a se tornarem PNLMD viáveis e aqueles pontos que são viáveis em aproximações lineares mas PNLMD inviáveis são rejeitadas .

O algoritmo final :

PASSO 1 : Dados δ , ε_0 , ε_f , t_0, r_t , $r_\varepsilon \in R_+$ e o ponto inicial x0.

Faça $\varepsilon = \varepsilon_0$, $t = t_0$.

PASSO 2 : Se (x_0 , y_0) é um ponto tal que y_0 é discreto, então faça (x_b , y_b) = (x_0 , y_0)

Senão (x_b , y_b) = round (x_0 , y_0) .

PASSO 3 : Se suminf (x_b, y_b) > ε_0 , então faça FASE = 1

Senão FASE = 2 .

PASSO 4 : Construir PLMDR (x_b , y_b) .

PASSO 5 : Resolver o PLMDR (x_b , y_b) e obter uma solução (x_k , y_k) .
Usando o simplex e branch and bound .

PASSO 6 : Se o problema for misto discreto , fixamos as variáveis discretas de (x_k , y_k) do passo 5 e resolvemos um subproblema não linear contínuo com a finalidade de convergir aos melhores valores da função objetivo em relação às variáveis contínuas . Usando o método do gradiente reduzido .

PASSO 7 : Faça dx = (x_k , y_k) - (x_b , y_b)

Se $|| dx_i ||_\infty < \delta$. PARE .

PASSO 8 : Se FASE = 2 ir para o passo 10 .

PASSO 9 : Se suminf (x_k , y_k) < suminf (x_b , y_b) .

Faça (x_b , y_b) = (x_k , y_k) e volte ao passo 3 .

Senão ir para passo 13 .

PASSO 10 : Se suminf (x_k , y_k) > ε ir para o passo 13 .

PASSO 11 : Se f (x_k , y_k) $\geq$ f (x_b , y_b) ir para o passo 13 .

Senão faça (x_b , y_b) = (x_k , y_k).

PASSO 12 : Faça ε = max (suminf (x_k , y_k) / $r_\varepsilon , \varepsilon_f$) .

Faça FASE =1 e volte para o passo 3 .

PASSO 13 : Faça t = t / r_t .

Se t < ε . PARE com a mensagem de erro .

Senão volte ao passo 4 .

OBSERVAÇÃO 1 : No passo 6 do algoritmo apresentado , foi feito um artifício para melhorar a eficiência do procedimento , pois a solução ótima mista discreta achada pelo passo 5 , provavelmente nas suas variáveis contínuas podem não estarem nos melhores valores , visto isto , fixamos as variáveis discretas como parâmetros e resolvemos o subproblema contínuo usando o método de gradiente reduzido . Se os valores contínuos achados no subproblema melhorarem a função objetivo , então substituimos esses valores no (x_k , y_k) , senão os valores das variáveis contínuas achados no passo 5 serão mantidos .

iv) Convergência

A prova de convergência dado pelo Loh e Papalambros consiste em provar a convergência relativa dos algoritmos básicos SLP 1 e SLP 2 , pois elas não asseguram que o algorítmo chegue até o ponto ótimo . Para o caso de restrições lineares , a prova é bastante simples , pois neste caso , a região de soluções viáveis dos aproximações lineares é exatamente idêntica ao região de soluções viáveis do problema original . A convergência se resume em dois teoremas abaixo enuciados , cujas demonstrações podem ser encontradas no " Tecnical Report " de Loh e Papalambros [4] .

TEOREMA 1 : Seja $f : R^n \times S \rightarrow R$ uma função pseudoconvexa e $g_i : R^n \times S \rightarrow R$ funções lineares para todo i , i = 1 ,....., p . Se a solução do SLP 1 do problema :

$$\begin{aligned} &\text{Minimizar } f(x, y) \\ &\text{s.a} \\ &\quad g_j(x, y) \leq 0 \qquad j = 1, \ldots\ldots, p \\ &\quad x \in R^n, y \in S \subseteq R^m_+ \end{aligned}$$

convergir para um ponto (x1 , y1) então (x1 , y1) é o minimizador discreto global .

A palavra convergir está um pouco forçosamente colocada , pois o teorema não garante uma convergência real do algoritmo , garante apenas que se o algoritmo finalizar em um ponto este é o minimizador discreto global .

TEOREMA 2 : O SLP 2 é um algoritmo decrescente e que termina .

O teorema 2 apenas garante que o algorítmo SLP 2 finalize mas não afirma que este ponto seja , um minimizador discreta global .

No caso das restrições serem funções convexas , o teorema enfraquece mais ainda .

TEOREMA 3 : Seja $f : R^n \times S \rightarrow R$ uma função pseudoconvexa e $g_i : R^n \times S \rightarrow R$ são funções convexas para todo i , i = 1 ,....., p .Se a solução de SLP 1 do problema :

$$\begin{aligned} &\text{Minimizar } f(x, y) \\ &\text{s.a} \\ &\quad g_j(x, y) \leq 0 \qquad j = 1, \ldots\ldots, p \\ &\quad x \in R^n, y \in S \subseteq R^m{}_+ \end{aligned}$$

finalizar para um ponto (x1 , y1) PNLMD viável , então (x1 , y1) é o minimizados discreto global .

TEOREMA 4 : O resultado do teorema 3 se aplica ao algoritmo SLP 2 modoficado pelas regras de $\in$-viabilidade .

Os teoremas 3 e 4 forem apenas citados no "Tecnical Report" de Loh e Papalambros [4] , não houve uma demonstração teórica mais convincente , foram afirmados devido aos resultados experimentais .

Capítulo 2

O Terceiro Método

2.1 Introdução

Nesta parte da tese será apresentada o terceiro algoritmo destinado a resolver o problema não linear misto discreto , desenvolvido a partir dos dois algorítmos apresentados anteriormente . Denominamo-lo de *Algoritmo de Aproximação Poliédrica com Região de Confiança* .Considere o seguinte problema :

$$
\begin{aligned}
&\text{Min } f(x, y) \\
&\quad \text{s.a} \\
&\qquad g_i(x, y) \leq 0 \qquad i = 1, \ldots\ldots\ldots, p \qquad (P) \\
&\qquad x \in X \\
&\qquad y \in Y
\end{aligned}
$$

Onde as funções $f : R^n \times R^m \rightarrow R$ e $g : R^n \times R^m \rightarrow R$ são todas contínuamente diferenciáveis em um aberto que contém $X \times Y$, x é a variável contínua e y a variável discreta . X é um subconjunto convexo e compacto em R^n e Y subconjunto discreto finito em R^m . Sem perda de generalidade , podemos considerar que Y é totalmente ordenado .

O nosso objetivo principal é de obter a resolução dos problemas onde o número de de restrições é muito grande em relação ao número de variáveis

discretas por um método de descida . Esta propriedade do método ser decrescente é muito importante , pois para certos problemas mistos discretos , o importante não é obter a solução do (P) , mas sim um ponto tal que melhore o ponto viável inicial .

O algoritmo é do tipo sequencial . Em cada iteração nós consideramos dois problemas : a aproximação linear do problema (P) e o problema não linear na variável contínua x . (problema de projeção sobre as variáveis discretas) .

Apresentamos as propriedades básicas tais como a convergência finita e as condições de otimalidade para o caso convexo . Esse tipo de problema surgem em várias áreas de interesse prático , como o problema de síntese no projeto de processo químico .

Para o método de aproximação linear sequencial de Geoffrion [3] e Duran e Grossmann [1] , considera-se a aproximação do plano de corte , este método é eficiente para um pequeno número de restrições , pois a cada iteração , o número de restrições lineares aumenta . Loh e Papalambros (Tecnical Report 89) [4] considerou a cada iteração a aproximação linear da função objetiva e das restrições com a região de confiança , este método tem implementação mais simples para problemas de tamanho grande . Mas este algoritmo pode terminar em um ponto que não seja a solução do problema (P) . O método discutido nesta terceira parte localiza-se entre estes dois tipos de aproximação . De fato , nós consideramos as aproximações lineares com região de confiança até encontramos um ponto de mínimo local ou um ponto isolado (supondo que o conjunto pode não ser uniformemente distribuído) ou chegar a conclusão que deve ser melhorado a aproximação linear . Neste caso , acrescentamos cortes do tipo feito por Duran-Grossmann [1] ao problema linear para podermos sair até um outro ponto , continuando a busca do ponto de mínimo global .

Motivado pelos problemas apresentados pelo Loh e Papalambros [4] e pelo fato de trabalhar com aproximação poliedral local , nós esperamos que este algorítmo tenha um bom performance para o caso de funções não convexas com propriedades monótonas . De fato , a parte da região de soluções viáveis que seriam eliminadas em uma iteração , poderão ser reconsideradas nas iterações seguintes , o que não acontece no método do plano de corte .

2.2 O Procedimento da aproximação

Ao longo desta secção estamos assumindo que o problema original (P) tem pelo menos uma solução . E as normas utilizadas no método são normas infinito .

Introduzimos finalmente o conceito de problema projetado já utilizado nos métodos anteriores :

DEFINIÇÃO 1 : O problema projetado de (P) em relação à variável discreta y_0 é:

$$\underset{x \in X}{\text{Min}} \{ f(x, y_0) \text{ sujeito à } g(x, y_0) \leq 0 \} \qquad P(y_0)$$

isto é , se $x(y_0)$ é a solução de $P(y_0)$ e $F(y_0) = f(x(y_0), y_0)$, então temos :

$$\underset{y}{\text{Min}} \{ F(y) \text{ sujeito à } y \in Y \cap V \} \qquad (P0)$$

onde $V = \{ y \in R^m : g(x, y) \leq 0 \text{ para algun } x \in X \}$

O seguinte teorema mostra que (P) e (P0) são equivalentes no seguinte sentido .

TEOREMA 1 : Se (x^*, y^*) é ótimo para (P) então y^* é ótimo para (P0) . Se y^* é ótimo para (P0) e x^* for o mínimo conseguido em $P(y)$ com $y = y^*$, então (x^*, y^*) é a solução ótima de (P) . O problema (P) é inviável se e somente se o mesmo é verdade para o (P0) .

A prova deste teorema poderá ser encontrada no trabalho do Duran e Grossmann [1].

Notamos que o ponto y é viável de (P0) se e somente se o problema parametrizado P (y) é viável .

DEFINIÇÃO 2 : O problema principal misto discreto inicial associado à cada (PNLMD) na iteração k é o seguinte :

Minimizar μ

s.a

$$f(u^k) + \nabla f(u^k)^t (u - u^k) - \mu \leq 0 \qquad (M^k{}_1)$$

$$g_l(u^k) + \nabla g_l(u^k)^t (u - u^k) \leq 0 \qquad l = 1, \ldots\ldots, p$$

$$||u - u^k||_\infty \leq \rho^k$$

$$u \in X \times Y^k$$

onde $\rho^k = +\infty$, $Y^k = Y^{k-1} - \{ y^k \}$, $u^k = (x^k , y^k)$.

A idéia básica do método é a seguinte :

a) Resolver o problema principal de aproximação linear ou poliédrica (P) com o objetivo de obter um ponto discreto y^{k+1} viável .

b) Resolver o problema não linear projetado P (y^{k+1}) .

c) Se y^{k+1} é um canditado no sentido de que y^{k+1} é viável e F (y^{k+1}) < F (y^k) , onde F (y^{k+1}) é a solução do problema parametrizado P (y^{k+1}) , então y^{k+1} é o novo ponto da sequência (passo sério) e vai ser considerado para a seguinte aproximação linear .

d) Se P (y^{k+1}) for inviável ou se F (y^{k+1}) $\geq$ F (y^k) , então nós não podemos considerar y^{k+1} como o novo ponto , devemos fazer o algoritmo de tal maneira que ache um novo canditado $y^{'k+1}$ e que o antigo y^{k+1} não seja analisado de novo ao longo do desenvolvimento da aproximação . Isto pode ser feito pela reformulação do conjunto das variáveis de inteiro ou acrescentando o corte dos inteiros [13]. Ou seja , uma vez resolvido o problema master $M^k{}_1$, seja $y^{k+1}{}_1$, a parte inteira da solução . Se $y^{k+1}{}_1$ for inviável , o problema projetado P ($y^{k+1}{}_1$)

não tem solução ,então devemos reconsiderar o M^k_1 com $Y^k_1 = Y^k_1 - \{ y^{k+1}_1 \}$. No caso contrário, seja $v^{k+1} = (x^{k+1}_1 , y^{k+1}_1)$ onde x^{k+1}_1 é a solução de P (y^{k+1}_1). Se $f(v^{k+1}) \geq f(u^k)$, diminui-se o valor de ρ^k e volta-se a calcular M^k_1 .O processo repete-se até obter um novo ponto v^{k+1} tal que $f(v^{k+1}) < f(u^k)$, ou atingir a vizinhança mínima estipulada ($\rho^k < \rho_L$) . Uma vez o valor decrescente é obtido , iremos para a nova com $u^{k+1} := v^{k+1}$, senão é conveniente melhorarmos a aproximação linear , tomando a aproximação poliedral considerando no último ponto v^{k+1} como sendo u^k_2 , e então repetimos o processo partindo com $\rho^k_2 = +\infty$ e assim sucessivamente .

Então , poderemos ter depois da i tentativa na iteração k :

$$
\begin{aligned}
&\text{Minimizar } \mu \\
&\text{s.a} \\
&\quad f(u^k_j) + \nabla f(u^k_j)^t (u - u^k_j) - \mu \leq 0 \qquad j = 1 ,\ldots\ldots, i \\
(M^k_i) \qquad &\quad g_l(u^k_j) + \nabla g_l(u^k_j)^t (u - u^k_j) \leq 0 \qquad l = 1 ,\ldots\ldots, p \\
&\quad \| u - u^k_1 \|_\infty \leq \rho^k_i \\
&\quad u \in X \times Y^k_i
\end{aligned}
$$

onde u^k_j corresponde à solução do subproblema (P^k_{j-1}) com vizinhança mínima . E i corresponde ao contador dentro de uma iteração k , onde o problema master é modificado e resolvido procurando achar um ponto que melhore o valor da função objetivo para poder passar a outra iteração k+1 .

2.3 O algoritmo

No algoritmo , Γ representará o subconjunto discreto Y.

PASSO 1 : Dado ρ_L , $\beta \in (0 , 1)$, $y^1 \in \Gamma$, $y^1_1 := y^1$, $\rho^1_1 := +\infty$, $k = i = 1$,

passo sério : = true , $\Gamma^1_1 := \Gamma - \{ y^1 \}$, iteração : = 0 ;

PASSO 2 : Problema projetado P (y^1) .

Se y^1 é inviável então escolher um $y^m \in \Gamma$ e

faça $y^1 := y^m$ e $\Gamma^1_1 := \Gamma - \{ y^m \}$ e volte ao problema projetado ;

Se y^1 é viável , então seja x^1_1 uma solução do problema .

Seja $u^1 := u^1_1 := (x^1_1 , y^1)$.

PASSO 3 : Teste de parada 1 .

Se $\Gamma^k_i := \varnothing$ então PARE . u^k é solução de (P).

Senão

Se iteração $\neq 0$ e passo sério : = false então

faça $d := || u^k_{i+1} - u^k ||_\infty$

Se $d \leq \rho_L$ então $\rho^k_{i+1} := +\infty$, $i := i + 1$

Senão ($d > \rho_L$) $\rho^k := \beta d$ $\Gamma^k_i := \Gamma^k_{i+1}$;

PASSO 4 : Resolver o problema master (M^k_i)

iteração : = iteração + 1 ;

Se não existe solução então PARE , u^k mínimo ou ponto isolado .

Senão $u^k = (v , w)$ solução de M^k_i , $y^k_{i+1} := w$.

PASSO 5 : Problema projetado P (y^k_{i+1})

Se não existe solução então $\Gamma^k_i := \Gamma^k_i - \{ w \}$, passo serio : = true ;

volte ao passo 3 .

Senão seja x^k_{i+1} uma solução do problema projetado então

$u^k_{i+1} := (x^k_{i+1} , y^k_{i+1})$; $\Gamma^k_{i+1} := \Gamma^k_i - \{ w \}$.

PASSO 6 : Teste de parada 2 .

Se $< \nabla f (u^k) , (u - u^k) > \geq 0$ então PARE .

$\rho^k_i = +\infty$ solução de (P)

$\rho^k_i < +\infty$ mínimo local de (P) ou ponto isolado .

PASSO 7 : Ponto novo

Se $f(u^k) \leq f(u^k_{i+1})$ então passo sério : = false , e volte para passo 3

Senão $f(u^k) > f(u^k_{i+1})$

$u^{k+1} := u^k_{i+1}$, $u^{k+1}_1 := u^{k+1}$,

$\Gamma^{k+1} := \Gamma^k_{i+1}$

$\Gamma^{k+1}_1 := \Gamma^k_{i+1}$

$\rho^{k+1} := +\infty$, $\rho^{k+1}_1 := \rho^{k+1}$, $i := 1$, $k := k+1$,

passo sério : = true,

e volte ao passo 3 ;

Observações:

PASSO 1 : É o passo de inicialização , ρ_L indica o limite mínimo de vizinhança , y^1 é um ponto discreto inicial de preferência viável e Γ^1_1 é o espaço inicial tirando o ponto inicial .

PASSO 2 : Teste de parada para o ponto discreto inicial . O algoritmo só sairá deste passo se achar um primeiro ponto discreto viável . Notamos que se o problema P (y) não for viável , então este ponto y também não será viável para o problema original (P) .

PASSO 3 : Teste de parada . Se satisfizer a primeira condição de parada , isto significa que nós temos considerados todos os pontos possíveis , não há mais nenhum a ser analisado , então o último valor u^k é a solução mínima do problema (P) .

Neste passo também fazemos a modificação da região de confiança e teste de limite inferior . De acordo com o resultado do teste , atualizamos ρ^k_i para um valor menor (isto é , mudamos a região de confiança , mantendo a mesma aproximação) ou para infinito para poder resolver o problema (P^k_{i+1}).Neste caso o objetivo é achar uma outra solução com índice i incrementado , aumentando assim o número de restrições (modificamos a aproximação) .Observamos que o algoritmo, na primeira iteração não passa por esta parte do passo 3 .

PASSO 4 : Se (M^k_i) for inviável , então mostraremos que u^k é um ponto isolado ou um mínimo global .

PASSO 5 : Nesta etapa verificamos a viabilidade do ponto discreto da solução obtida no passo 4 , resolvendo o problema não linear projetado .

PASSO 6 : Se o teste de parada for satisfeito pela segunda condição , significa que o ponto u^k é uma solução ótima de (P) (com $\rho = +\infty$) , ou uma solução de mínimo local do (P) (com $\rho < +\infty$) ou um ponto isolado .

PASSO 7 : Teste de atualização . Obtém-se um novo ponto , se a solução obtida no passo 5 faz a função objetivo decrescer de valor , neste caso tem-se um "passo sério" . Atualizamos as variáveis , domínio e limite da região de confiança para voltar a uma nova iteração . Caso contrário tem-se um "passo nulo"; volta-se ao passo 3 onde define-se a aproximação a ser feita (redução da região de confiança ou modificação da aproximação poliedral).

Notemos que logo após de um passo sério o algoritmo volta a considerar uma aproximação linear , partes da região viável que possivelmente tenha sido eliminadas anteriormente , no processo de convexificação do modelo podem voltar a serem examinadas. Este fato torna o algoritmo interessante quando o problema (P) for não convexo.

2.4 Convergência

Antes de mostrarmos os teoremas que comprovam a convergência do algoritmo e as condições de optimalidade , introduzimos primeiramente as seguintes definições :

DEFINIÇÃO 1 : Um ponto u^* é considerado um ponto misto discreto viável isolado do problema (P) , relativo à norma considerada se :

a) u^* é um ponto viável de (P) .

b) $\exists\ \rho > \rho_m$ onde ρ_m é o tamanho mínimo de discretização da variável discreta y , tal que :

$\forall\ u : || u - u^* || < \rho \Rightarrow$ u não é um ponto viável de (P).

DEFINIÇÃO 2 : O ponto u^* é um ponto de mínimo local de (P) relativo à norma considerada , se existir um $\rho \geq \rho_m$ tal que :

a) $\exists\, u : 0 < \| u - u^* \| \leq \rho$, u ponto viável do problema (P).

b) $\forall$ u ponto viável tal que $\| u - u^* \| < \rho$ tem se $f(u^*) \leq f(u)$.

LEMA 1 : Se f é uma função convexa diferenciável então f é uma função pseudoconvexa , ou seja , $f(v) < f(u) \Rightarrow \langle \nabla f(u), v - u \rangle < 0$.

Notemos que para um ρ_L suficientemente grande ($\rho_L \to \infty$) . O método proposto torna-se o método de cortes de Duran e Grossmann . No outro extremo quando o modelo poliedral é sempre linear resulta ser o algoritmo de Loh e Papalambros . Em geral , é uma modificação deste método mas convergente .

Assumindo que as funções f e g são convexas diferenciáveis temos :

TEOREMA 1 : Seja u a solução de (M^k_i) tal que verifica $\langle \nabla f(u^k), u - u^k \rangle \geq 0$ e $\rho^k_i < +\infty$ então u^k é um mínimo local ou um ponto viável isolado .

Dem :

Suponha que a conclusão não seja verdadeira , ou seja , u^k não é nem um ponto de mínimo local nem um ponto isolado .

Seja $U^k_i = \{ u \in R^{n \times m} : 0 < \| u - u^k \|_\infty \leq \rho^k_i \} \neq \varnothing$

e $\exists\, u^* \in U^k_i$, $u^* \in X \times Y^k_i$ tal que $f(u^*) < f(u^k)$ e $g(u^*) \leq 0$

Se u é uma solução de M^k_i então o valor ótimo do problema vem dado por :

$$\mu = \max_{1 \leq j \leq i} [\, f(u^k_j) + \nabla f(u^k_j)^t (u - u^k_j) \,] \geq f(u^k_1) + \nabla f(u^k_1)^t (u - u^k_1)$$

Mas pelo passo 7 , $u^k_1 = u^k$, e a partir da hipótese tem-se:

$$\mu \geq f(u^k) + \nabla f(u^k)^t (u - u^k) \geq f(u^k)$$

isto é : $\mu \geq f(u^k)$ (*)

por outro lado , se u^* viável para o problema (P) então o ponto

(u^*, μ^*) com $\mu^* = f(u^*)$ satisfaz as restrições de M^k_i. De fato:

$f(u^k_j) + \nabla f(u^k_j)^t (u^* - u^k_j) \leq f(u^*)$ $\quad j = 1, \ldots\ldots, i$

$g_l(u^k_j) + \nabla g_l(u^k_j)^t (u^* - u^k_j) \leq g_l(u^*) \leq 0$ $\quad l = 1, \ldots\ldots, p$

$||u^k - u^*||_\infty \leq \rho^k_i$

Como μ é o valor mínimo de (M^k_i), deve ser $\mu \leq f(u^*) < f(u^k)$, isto é : $\mu < f(u^k)$ $\quad (**)$

De $(*)$ e $(**)$:

Contradição

COROLÁRIO 1 : Se a solução u do problema (M^k_i) com $\rho^k_i = +\infty$ verifica $< \nabla f(u^k), u - u^k > \geq 0$ então u^k é um mínimo global de (P).

TEOREMA 2 : Se o (M^k_i) é um problema inviável então u^k é um ponto viável isolado ou é uma solução global de (P).

Dem :

Nós definimos :

$D = \{ u \in X \times Y : u$ é viável de $(P) \}$ e

$D^k_i = \{ u \in X \times Y^k_i : u$ é viável de $(P) \}$

Seja $D \neq \{ u^k \}$, pois se $D = \{ u^k \}$ o teorema é obviamente válido .

Se $D^k_i = \varnothing \Rightarrow u^k$ é solução de (P) pois a sequência $\{ u^k \}$ é de descida . Em efeito : dado $u \in X \times Y$, $u = (x, y)$. se $y = y^l$ para algum $l < k \Rightarrow f(u) \geq \min_{z \in X} f(z, y^l) = f(u^l) > f(u^k)$

Se $y \neq y^l$, para $l=1,.., k \Rightarrow y = y^l_i$ para algun $l \leq k$ em certa iteração intermediária i , então : $f(u) \geq \min_{z \in X} f(z, y^l_i) = f(u^l_i)$

$\geq f(u^k)$ ou y^l_i é inviável .

Se $D^k_i \neq \varnothing$ para cada $u \in D^k_i$, nós temos :

$f(u^k_j) + \nabla f(u^k_j)^t (u - u^k_j) \leq f(u)$ $\quad j = 1, \ldots\ldots, i$

$$g_l(u^k_j) + \nabla g_l(u^k_j)^t (u - u^k_j) \leq g(u) \leq 0 \quad j = 1, \ldots\ldots, i$$
$$l = 1, \ldots\ldots, p$$

com $\mu = f(u)$, u é também viável para as restições lineares de (M^k_i). Mas como (M^k_i) é inviável, deve ser $\| u - u^k \|_\infty > \rho^k_i$ então u^k é um ponto isolado.

COROLÁRIO 2 : Se o problema (M^k_i) é inviável e $\rho^k_i = +\infty$ (i = 1) então u^k é um mínimo global .

TEOREMA 3 : Se o problema (P) tem solução e não tem pontos isolados então o algoritmo atinge uma solução global ou local de (P) em um número finito de iterações .

Dem :

O algoritmo tem três tipos de paradas possíveis :

1) Na iteração k , a solução $u = (v, w) \in X \times Y^k_i$ do problema (M^k_i) verifica $\langle \nabla f(u^k), u - u^k \rangle \geq 0$ e $\rho^k_i = +\infty$ então do corolário 1 , nós obtemos que u^k é a solução de (P) , ou se $\rho^k_i < +\infty$ temos do teorema 1 que u^k é solução local .

2)Na iteração k , não existe solução do problema (M^k_i) , pelo teorema 2 o ponto u^k é mínimo global . Todos os pontos discretos foram examinados e como o algoritmo tem a propriedade decrescente , nós concluimos que o último ponto da sequência finita $\{ u^k \}$ é a solução do problema (P) .

Para completar a prova , temos :

3) Finalmente notemos que o conjunto de pontos discretos é finito e cada ponto discreto é no máximo considerado uma vez .

CAPÍTULO 3

Os Testes Computacionais

3.1 Os exemplos usados para os testes

Foram utilizados três exemplos para fazer os testes computacionais . Os dois primeiros correspondem aos problemas testes 1 e 2 do trabalho de Duran e Grossmann [1] , onde apresentam características exigidas pelo Algoritmo de Aproximação Externa , ou sejam , a separabilidade dos dois tipos de variáveis que envolvem o problema : a linearidade das variáveis inteiras e a convexidade das funções não lineares que estão associadas às variáveis contínuas .Ambos problemas são de minimização , com algumas restrições não lineares e outras lineares . São dados também os pontos iniciais para as variáveis inteiras e os limites superiores e inferiores para as variáveis contínuas . Os dois exemplos serão apresentados logo abaixo . Para as uma descrição mais detalhada , basta olhar a referência bibliográfica [1] . O terceiro exemplo foi retirado do trabalho de Loh e Papalambros [4] . Este é um problema não convexo tal que o Algoritmo de Linearização Sequencial conseguiu convergir , ele foi selecionado com o objetivo de observar o comportamento dos outros dois algoritmos para solucionar um problema não convexo .

Os três problemas testes são descritos a seguir :

Problema teste 1 :

minimizar $z = 5y1 + 6y2 + 8y3 + 10x1 - 7x3 - 18\ln(x2+1) - 19.2\ln(x1-x2+1) + 10$

sujeito a :

$$0.8\ln(x2+1) + 0.96\ln(x1-x2+1) - 0.8x3 \geq 0$$
$$x2 - x1 \leq 0$$
$$x2 - 2y1 \leq 0$$
$$x1 - x2 - 2y2 \leq 0$$
$$\ln(x2+1) + 1.2\ln(x1-x2+1) - x3 - 2y3 \geq -2$$
$$y1 + y2 \leq 1$$

$$y \in \{ 0 , 1 \}^3 , x \in R^3$$
$$0 \leq x1 \leq 2$$
$$0 \leq x2 \leq 2$$
$$0 \leq x3 \leq 1$$

ponto inicial inteiro : $y0 = (1 , 0 , 1)$

Problema teste 2 :

minimizar $z = 5y1 + 8y2 + 6y3 + 10y4 + 6y5 - 10x1 - 15x2 - 15x5 + 15x3 + 5x4 - 20x6 + \exp(x1) + \exp(x2/1.2) - 60\ln(x3+x4+1) + 140$

sujeito a :

$$-\ln(x3+x4+1) \leq 0$$
$$-x1 - x2 - 2x5 + x3 + x2 \leq 0$$
$$-x1 - x2 - 0.75x5 + x3 + x2 \leq 0$$
$$x5 - x6 \leq 0$$
$$2x5 - x3 - 2x6 \leq 0$$
$$-0.5x3 + x4 \leq 0$$
$$0.2x3 - x4 \leq 0$$
$$\exp(x1) - 10y1 \leq 1$$
$$\exp(x2/1.2) - 10y2 \leq 1$$
$$1.2x5 - 10y3 \leq 0$$
$$x3 + x4 - 10y4 \leq 0$$

$$-2x5 + 2x6 - 10y5 \leq 0$$

$$y1 + y2 = 1$$

$$y4 + y5 \leq 1$$

$$y \in \{ 0 , 1 \}^5 , x \in R^6$$

$$0 \leq x1 \leq 2$$

$$0 \leq x2 \leq 2$$

$$0 \leq x3 \leq 2$$

$$0 \leq x4$$

$$0 \leq x5$$

$$0 \leq x6 \leq 3$$

ponto inicial inteiro : $y0 = (1 , 0 , 0 , 0 , 0)$

Problema teste 3 :

minimizar $z = - 9y1^2 + 10y1y2 - 50y1 + 8y2 + 400$

sujeito a :

$$y1 - (0.2768y2^2 - 0.235y2 + 3.718) \leq 0$$

$$y1 - (- 0.019y2^3 + 0.446y2^2 - 3.98y2 + 15.854) \leq 0$$

$y1 , y2$ inteiros

ponto inicial inteiro : $y0 = (7 , 5)$

Os índices de todos os problemas testes foram rearrumados de uma maneira tal que facilite a vizualização e o manuseio dos problemas no decorrer da implementação dos algoritmos .

Notemos também que nesses exemplos as variáveis inteiras são de espaçamento uniforme , ou seja , a distância entre duas variáveis inteiras vizinhas são sempre iguais , portanto nos problemas testes não existem pontos isolados .

Os pontos iniciais inteiros dados pelos problemas não são garantidos como sendo pontos viáveis . No próximo subcapítulo será explicado o procedimento feito para achar o primeiro ponto viável inicial .

Observemos que nos exemplos 1 e 2 as variáveis inteiras envolvidas são binárias , enquanto no problema 3 não existem variáveis contínuas .

3.2 Implementação : Considerações

Na implementação dos três algoritmos analisados foram utilizados dois pacotes computacionais fechados matemáticos existentes no mercado , onde cada um deles é responsável por uma tarefa importante para o sucesso dos algoritmos, são eles :

1) **GINO** (General, Interactive Optimizer) é um programa para resolver problemas de otimização não lineares , onde as variáveis assumem valores contínuos [10] .

2) **LINDO** (Linear , Interactive And Discrete Optimizer) é um sistema de programação mista discreta inteira , linear interativa e quadrática . Podemos notar que este pacote é destinado para uma grande faixa de usuários . Neste trabalho , apenas aproveitamos a parte da resolução de problema lineares envolvendo variáveis mistas-discretas [11] .

A interface entre estes dois pacotes fechados é feito através de arquivos , ou seja ,montamos o modelo matemático de programação linear ou não linear que desejamos solucionar em um arquivo de entrada , uma vez entrado no pacote , chamamos o arquivo e mandamos resolvê-lo . Tudo isso é feito através de comandos préviamente preparados num arquivo.bat . O Gino não teve nenhum problema em relação a este tipo de manipulação , pois não apresentou nenhuma característica particular à interface . Infelizmente , não podemos dizer o mesmo sobre o Lindo , neste caso , não bastou armazenar o problema num arquivo de entrada qualquer . Para que o Lindo consiga manipular e executar os comandos programados , foi necessário antes de tudo , criar um arquivo de entrada salvo por ele mesmo , pois o Lindo só consegue executar comandos sobre os arquivos que são salvos por ele mesmo . Isto diretamente interfere no rendimento e na

velocidade dos algoritmos , uma vez que o número de iteração é o próprio número de chamadas de Lindo . Dependendo do algoritmo , este problema pode piorar mais ou menos (depende do modelo de linearização) a rapidez da resolução .

No primeiro algoritmo de Duran e Grossmann , antes de começarmos a primeira iteração , foi necessário chamar o Lindo para criar o arquivo de entrada de dados salvo por ele , mas nas iterações subsequentes , apenas acrescentamos as restrições novas calculados nos novos pontos canditados à ótimo , o modelo inicial básico não sofreu qualquer modificação . O que não acontece com o segundo e o terceiro algoritmo , pois nesses métodos , a entrada do problema preparada para cada iteração subsequente pode não ser acumulativa . Fazemos novas linearizações das restrições não lineares e da função objetivo (sem considerar às feitas em iterações anteriores) cada vez que achamos um novo ponto da sequência minimizante . Como o Lindo só reconhece os arquivos salvos por ele , acarretando assim a necessidade de fazer duas chamadas de Lindo . Como o Lindo é considerado um pacote razoavelmente grande , isto pode custar no tempo de execução , apesar do número de passos sérios podem ser menores .

Foram feitos para cada algoritmo , três programas computacionais diferentes , um para cada exemplo teste correspondente . O único que é um pouco peculiar é o terceiro problema teste , pois envolve apenas variáveis inteiras . Neste caso , foram feitas algumas adaptações computacionais em relação ao primeiro e segundo problemas testes , eliminando na prática os procedimentos associados às variáveis contínuas .

Foi feito um procedimento separado no programa principal de todas as implementações , com exceção para os algoritmos do terceiro problema teste , para achar o primeiro ponto inteiro viável , pois os pontos fornecidos pelos problemas testes não garantem a viabilidade dos pontos dados . Foi feito neste procedimento o teste de viabilidade do ponto inicial : substituindo esses valores inteiros no problema original tornando-os um problema projetado não linear com variáveis contínuas , chamamos o Gino para resolver . Se o ponto é um ponto viável , o problema projetado tem solução . Caso contrário , armazenamos este inteiro num conjunto e geramos um outro ponto aleatório pelo computador se y não for limitado , uma vez gerado este ponto aleatório , comparamos com os elementos existentes neste conjunto , se há coincidência , será achado uma outra

combinação até que o problema projetado substituído por este novo ponto seja um problema viável .

A idéia de introduzir o corte inteiro feito por Duran e Grossmann ,tem o objetivo de assegurar no decorrer da execução do programa ,que as combinações inteiras consideradas nas iterações anteriores não sejam analisadas novamente. Isto ajuda a reduzir o tempo gasto em procedimentos computacionais nas enumerações implícitas , quando resolvemos os problemas principais pelo Lindo nas iterações subsequêntes.

Para o caso de a combinação inteira ser um elemento de um hipercubo unitário , o sequinte lema apresenta um corte inteiro que tem uma boa performance:

LEMA : Dado uma combinaçõa inteira $y^i = \{ y^i_j : j = 1 , ..., m \} \in \{ 1 , 0 \}^m$ onde o conjunto de índices $B^i = \{ j : y^i_j = 1 \}$, $NB^i = \{ y^i_j = 0 \}$, tais que $| B^i | + | NB^i | = m$, a restrição inteira

$$\sum_{j \in B^i} y_j - \sum_{j \in NB^i} y_j \leq | B^i | - 1$$

será violada somente pelo y^i e não mais nenhum $y^k \neq y^i$.

Este corte inteiro é fácil de ser implementado . Para mais detalhes , consulte referência bibliográfica [1] .

Na implementação do Algoritmo de Linearização Sequencial foi feito uma grande modificação , pois acrescentamos também este conceito de corte inteiro. Se não eliminamos este ponto inteiro toda vez depois da resolução do problema principal ele poderá ser considerado novamente e satisfazendo assim o teste de parada sem ao menos de ter achado um outro ponto solução , canditado ao mínimo.

Alguns parâmetros usados no segundo algoritmo foram aqueles sugeridos pelo Loh e Papalambros :

$\varepsilon_0 = 1$;

$\varepsilon_f = 0.001$;

$r_\varepsilon = 2$;

$\delta = 0.001$;

O único parâmetro que sofreu modificação é o t_0 , que corresponde ao tamanho de região de confiança inicial . Em vez de valor 4 como estava no exemplo , foi substituído por valor 5 , para ter coerência com o desenvolvimento teórico feito em Loh e Papalambros .

Os programas foram feitos em Turbo Pascal . Eles estão anexados nos apêndices para os que tiverem um interesse maior nos detalhes de implementação . Como o primeiro e o segundo problemas testes tem as mesmas características , mas com número diferentes de variáveis , serão apresentados apenas os problemass testes 1 e 3 para todos algoritmos . Assim totalizando um número de seis programas .

3.3 Resultado dos testes

Algoritmo de aproximação externa aplicado ao problema teste 1(PP1.pas)

O valor da função ótima : 7.165374

As variáveis inteiras ótimas são :
y1 : 1
y2 : 0
y3 : 0

As variáveis contínuas ótimas são :
x1 : 1.312195
x2 : 1.312095
x3 : 0.838269

Número de iterações : 7
Tempo gasto para execução do problema (hh:min:seg.seg100):0:1:5.03

Algoritmo de aproximação externa aplicado ao problema teste 2(PP2.pas)

O valor da função ótima : 73.035316

As variáveis inteiras ótimas são :

y1 : 0

y2 : 1

y3 : 1

y4 : 1

y5 : 0

As variáveis contínuas ótimas são :

x1 : 0.000000

x2 : 2.000000

x3 : 0.652015

x4 : 0.326007

x5 : 1.078388

x6 : 1.078388

Número de iterações : 10

Tempo gasto para execução do problema (hh:min:seg.seg100):0:1:52.59

Algoritmo de aproximação externa aplicado ao problema teste 3(PP3.pas)

O valor da função ótima : 59.000000

As variáveis inteiras ótimas são :

y1 : 7

y2 : 5

Número de iterações : 2

Tempo gasto para execução do problema (hh:min:seg.seg100):0:0:11.31

Observação : O ponto obtido não é viável .

Algoritmo de linearização sequencial aplicado ao problema teste 1(Q1.pas)

O valor da função ótima : 7.1654

As variáveis inteiras ótimas são :

y1 : 1

y2 : 0

y3 : 0

As variáveis contínuas ótimas são :

x1 : 1.3 E +00

x2 : 1.3 E +00

x3 : 8.4 E -01

Número de iterações : 14

Tempo gasto para execução do problema (hh:min:seg.seg100):0:1:47.15

Algoritmo de linearização sequencial aplicado ao problema teste 2(Q2.pas)

O valor da função ótima : 134.713505

As variáveis inteiras ótimas são :

y1 : 1

y2 : 0

y3 : 0

y4 : 0

y5 : 0

As variáveis contínuas ótimas são :

x1 : 1.6E +00

x2 : 4.7E -05

x3 : -3.4E -05

x4 : -7.0E -06

x5 : 0.0E +00

x6 : 1.7E-05

Número de iterações : 2

Tempo gasto para execução do problema (hh:min:seg.seg100):0:0:19.38

Algoritmo de linearização sequencial aplicado ao problema teste 3(Q3.pas)

O valor da função ótima : 159.0000

As variáveis inteiras ótimas são :

y1 : 5

y2 : 3

Número de iterações : 6

Tempo gasto para execução do problema (hh:min:seg.seg100):0:0:31.80

Algoritmo de aproximação poliédrica com região de confiança aplicado ao problema teste 1(R1.pas)

O valor da função ótima : 7.165374

As variáveis inteiras ótimas são :

y1 : 1

y2 : 0

y3 : 0

As variáveis contínuas ótimas são :

x1 : 1.312195

x2 : 1.312095

x3 : 0.838269

O tamanho da região de confiança é : 100.000000

Número de iterações : 10

Tempo gasto para execução do problema (hh:min:seg.seg100):0:1:12.17

Algoritmo de aproximação poliédrica com região de confiança aplicado ao problema teste 2(R2.pas)

O valor da função ótima : 74.294030

As variáveis inteiras ótimas são :

y1 : 0

y2 : 1

y3 : 1

y4 : 0

y5 : 0

As variáveis contínuas ótimas são :

x1 : 0.000000

x2 : 2.000000

x3 : -0.000023

x4 : 0.000005

x5 : 1.600044

x6 : 1.600028

O tamanho da região de confiança é : 100.000000

Número de iterações : 4

Tempo gasto para execução do problema (hh:min:seg.seg100):0:0:17.24

Algoritmo de aproximação poliédrica com região de confiança aplicado ao problema teste 3(R3.pas)

O valor da função ótima : 159.0000

As variáveis inteiras ótimas são :

y1 : 5

y2 : 3

O tamanho da região de confiança é : 100.000000

Número de iterações : 4

Tempo gasto para execução do problema (hh:min:seg.seg100):0:0:12.63

i) Algoritmo de Aproximação Externa

Pelos resultados obtidos podemos analisar caso por caso :

Problema teste 1 . (PP1.pas)

Este foi o primeiro exemplo implementado , o resultado em princípio nào foi satisfatório , pois no trabalho de Duran e Grossmann os valores ótimos achados por eles são :

função objetivo = 6.00972

variáveis inteiras = (0 , 1 , 0)

variáveis contínuas = (1.30097 , 0 , 1)

Podemos ver que há uma diferença apesar de ser pequena mas considerável entre os resultados . Foram feitos vários testes para descobrir o porquê da diferença . O programa foi analisado iteração por iteração para ver os resultados que entravam e saíam dos pacotes Gino e Lindo . E finalmente , foi descoberto que mesmo quando consideramos as variáveis inteiras ótimas dadas por Duran e Grossmann e as substituimos no problema original , o problema projetado associado a este ponto inteiro resolvido por Gino , não tem os mesmos resultado ao do trabalho de Duran e Grossmann , pois a solução apresentado foi :

função objetivo = 7.906643

variáveis inteiras = (0 , 1 , 0)

variáveis contínuas = (2 , 0 , 1)

Consequentemente , em uma das iterações onde (0 , 1 , 0) foi novo canditado inteiro a ser ótimo , foi eleiminado depois de ter sido calculado a projeção pelo Gino . Esta conclusão foi confirmada ainda mais pelos resultados apresentados pelos outros dois algoritmos em relação ao mesmo problema teste .

Vale a pena ressaltar que houve 7 chamadas de Lindo . Pelo que foi explicado anteriormente no início deste capítulo , a primeira chamada do Lindo foi para criar entrada de arquivo salvo por ele , entào na verdade , foram achados em cada iteração subsequente 6 soluções inteiras canditadas a ser o ótimo . Sendo que o problema envolve com três variáveis inteiras , há uma possibilidade de no total 2^3 candidatos a ótimo .

Problema teste 2 . (PP2.pas)

O resultado encontrado neste teste , com a diferença do caso anterior , foi coerente com a solução encontrada no trabalho de Duran e Grossman :

função objetivo = 73.0353

variáveis inteiras = (0 , 1 , 1 , 1 , 0)

variáveis contínuas = (0 , 2 , 0.65201 , 0.32601 ,1.07839 , 1.07839)

Os valores que nós achamos correspondem ao mesmo ponto mas com uma casa decimal a mais de precisão .

Notemos que foram feitas 10 iterações para chegar até o ótimo , e foram cosiderados 9 combinações dentre dos 2^5 possibilidades .

Observemos também que a cada iteração acrescentamos 5 restrições , pois existem 4 funções não lineraes involucradas no problema original e mais uma restrição devido ao corte inteiro . Ao final da décima iteração , acumulamos mais ou menos umas 60 restrições . O que não é dificil de imaginar quando num problema grande onde as restrições não lineares são muito frequentes , o tamanho astronômico que pode resultar no modelo da programação linear mista discreta , acarretando por consequente problemas de estabilidade numérica .

Problema teste 3. (PP3.pas)

Como este problema envolve apenas variáveis inteiras , foi explicado na seção anterior que o procedimento para verificar o ponto inicial não se aplica neste caso .

A solução ótima fornecida pelo trabalho de Loh e Papalambros é a seguinte :

função objetivo = 159.0

variáveis inteiras ótimas = (5 , 3)

O algoritmo parou no ponto inicial pois verificou o teste de parada , mas não é a solução ótima , pois o ponto inicial (7 , 5) não é viável apesar de f (7 , 5) < f (5 , 3) .O Algorítmo de Aproximação Externa falha neste caso porque a função não é convexa . Logo na segunda iteração , o teste de parada já foi conferido , ou seja , inviabilidade do problema principal associado a (7 , 5) .

Foi feito um outro teste para confirmar que este algorítmo não trabalha bem nos casos não convexos . Em vez de pegar o ponto (7 , 5) não viável como sendo o inicial , foi substituído por um outro ponto (4 , 3) viável e observamos o seguinte : mesmo partindo com o ponto inicial (4 , 3) viável , na segunda iteração

a solução encontrada pelo Lindo candidato a ótimo foi o (0 , 0) , com mensagem impressa indicando que este ótimo não é viável , o que acarretou a satisfação do teste de parada , fazendo com que o algoritmo convirja erroneamente .

Há uma outra observação importante verificada : como este não é um problema convexo , a restrição que limitava superiormente e inferiormente o valor da função objetivo ótimo a ser encontrado no modelo de entrada do Lindo (PLMD) $Z^i \leq C^t y + \mu \leq$ Z ótimo foi retirada , pois os Z^i 's que representavam a solução da função objetivo ótimo solucionado pelo Lindo em cada iteração são maiores do que Z ótimo ,fazendo com que esta restrição nunca será satisfeita .

Uma curiosidade : se apenas retirarmos o lado esquerdo da restrição acima $Z^i \leq C^t y + \mu$, o algoritmo entra em um loop e a solução do Lindo é sempre o ponto (0 , 0) mas nunca confere o teste de parada .

ii) Algoritmo de Linearização Sequencial

Durante a implementação deste algoritmo , foi verificado que devido ao forma de linearização proposto pelo algoritmo , as saídas do Lindo (PLMD) apresentam formas não muitos satisfatórias , pois o Lindo mesmo sem conseguir achar soluções viáveis inteiras , ele mostra na saída uma mensagem de erro mas solucionando o problema com os valores reais para as variáveis inteiras .

Para contornar este tipo de obstáculo , foi acrescido no procedimento da leitura do arquivo de saída do Lindo , um comando que arredonda as variáveis inteiras para o valor inteiro mais próximo .

Esta modificação junto com a do corte inteiro , fez com que o algoritmo convirja para o valor ótimo da função objetivo no primeiro problema teste . Uma análise mais detalhada será apresentada a seguir .

Problema teste 1 . (Q1.pas)

Na análise passo a passo deste problema durante a execução do algoritmo foi verificado que a sequência dos pontos inteiros é :

iteração	ponto inteiro
0	(1 , 0 , 1)
2	(0 , 1 , 0)
4	(1 , 0 , 0)
6	(0 , 1 , 1)
8	(0 , 0 , 0)
10	(0 , 0 , 0)
12	(0 , 0 , 1)
14	(1 , 0 , 0)

(tabela 1)

Depois da décima quarta iteração , o algoritmo convergiu , pois satisfez o teste de parada . Repare que nas iterações onde as soluções inteiras aparecem idênticas não significam que os valores que saem do Lindo nessas iterações são exatamente iguais , pois podem ser números reais aproximados aos inteiro mais próximo . Exemplo : na iteração 8 , o valor exato fornecido pela saída do Lindo é (0.03148 , 0.03 , 0) e na iteração 10 é (0.375 , 0 , 0.625) .

O valor da função objetivo ótimo achado é o mesmo valor do primeiro algoritmo : 7.1654 . E as variáveis inteiras e contínuas também . A única diferença é o número de iterações e o tempo gasto na execução do algoritmo . Já foi explicado anteriormente que neste algorítmo para que um novo ponto , canditado a ótimo seja achado são necessários duas chamadas de Lindo .

Foi feito também para este problema teste o Algoritmo de Linearização Sequencial sem modificação de arredondamento e nem corte inteiro , ou seja , exatamente como estava no trabalho de Loh e Papalambros . Pelo que nós vemos na tabela de soluções , o valor encontrado da função objetivo não foi muito

satisfatória apesar do número de iteração e o tempo de execução tem diminuído sensívelmente .

Analisamos passo a passo pela seguinte tabela :

iteração	função objetivo	variáveis inteiras	variáveis contínuas
0	15.165374377	(1 , 0 , 1)	(1.312195,1.312195, 0.838269)
2	9.9986364847	(0 , 0 , 0)	(0.0001, 9.7×10^{-5}, 8.0×10^{-5})
4	9.9986364847	(0 , 0 , 0)	(0.0001, 9.7×10^{-5}, 8.0×10^{-5})

(tabela 2)

Depois da quarta iteração , o algorítmo convergiu , mas não com soluções satisfatórias .Sem as modificações , o algoritmo achou um novo canditado ao ponto inteiro ótimo diferente daquele que foi selecionado com as modificações .

Problema teste 2 . (Q2.pas)

O resultado apresentado pelos algoritmos houve uma convergência , mas para um ponto muito longe de ser considerado ótimo .E logo na segunda iteração , o teste de parada já foi conferido . Isto significa que uma vez o Algoritmo de Linearização Sequencial chega a um ponto mínimo , ele não consegue sair dessa região à procura de um outro ponto de mínimo local canditado ao ótimo . Uma vez que ele chega neste ponto , o que ele faz é diminuir a região de confiança e resolver o problema associado novamente . Se o ponto é mínimo local , o mesmo vai continuar sendo a solução mínima para regiões menores , satisfazendo assim o teste de parada .

Pelo o que observamos deste algoritmo nos dois problemas testes 1 e 2 , concluimos que a escolha do ponto inicial é fundamental para o sucesso deste algoritmo .

Também foi feito para este exemplo o Algoritmo de Linearização Sequencial sem arredondamento e sem corte . Na verdade o algoritmo não convergiu neste problema apesar de apresentar uma solução , pois o ponto inicial estava muito próximo de um mínimo local e foi tentando achar um outro mínimo local diminuindo a região de confiança até exceder o limite máximo permitido .

Problema teste 3 . (Q3.pas)

Para este problema , o algoritmo trabalhou conforme como estava no trabalho de Loh e Papalambros . Foram necessários seis iterações para chegar ao ótimo , ou seja , foram achados três canditados à mínimo antes de conferir o teste de parada . O tempo gasto para execução é considerado peqùeno .

iii) Algoritmo de aproximação poliédrica com região de confiança

Problema teste 1 . (R1.pas)

O algoritmo implementado para este problema teste funcionou perfeitamente . A primeira impressão quando olhamos para o resultado do teste , 10 iterações foram necessários para chegarmos ao resultado ótimo . Mas acompanhando o algoritmo passo a passo , apenas cinco combinações inteiras foram selecionadas e analisadas .

Olhe a seguinte tabela para saber a sequência dos pontos inteiros selecionados .

iteração	ponto inteiro
0	(1 , 0 , 1)
2	(0 , 1 , 0)
4	(0 , 0 , 0)
6	(1 , 0 , 0)
8	(0 , 1 , 1)
10	(0 , 0 , 1)

(tabela 3)

Na quarta e na oitava iteração houve uma extensão das restrições do problema master a serem resolvidos , pois nessas duas iterações , a combinação inteira selecionada não conseguiu fornecer um valor da função objetiva melhor do que já existente (passo sério = false) . Neste caso , houve uma tentativa na diminuição da região de confiança ,mas como $d := ||u^k - u^k_{i+1}|| < \rho_L$ ($\rho_L = 2.0$) o que implica no aumento das restrições para melhorar a aproximação .

Teoricamente , o teste de parada 2 devia ser conferido na iteração oito , mas na prática isto não aconteceu , pois o resuldado fornecido pelo Gino não foi suficientemente bom . O cálculo de teste de parada 2 foi -3.0632578 que é um valor que aproxima de zero . A tolerância utilizada na implementação para este cálculo foi de -0.0001 e não foi suficiente para cobrir o erro .

O teste somente foi conferido no passo seguinte .

O tamanho da região de confiança 100.00000 indica que o algoritmo convergiu para um mínimo global .

Notemos que apesar de o método apresentar neste problema teste um número de iterações maior do que o método de Aproximação Externa , o tempo gasto para resolução foi apenas um pouco maior . Portanto levando o comentário feito anteriormente que por problemas na implementação do método com o Lindo (para cada iteração do algoritmo , é necessário fazer duas chamadas do Lindo devido a que este só reconhece arquivos salvos por ele) , o seu comportamento resulta satisfatório .

Problema teste 2 . (R2.pas)

Para resolver este problema , o algoritmo gastou apenas 17,24 segundos em quatro iterações . Apesar de a resposta ótima final não é o esperado , ela está razoavelmente perto da solução real . O motivo do erro foi justamente o mesmo do problema teste anterior , devido à solução do Gino não apresentar exatidão . Só que o efeito disso neste problema foi o contrário . Pois o ponto supostamente ser o ótimo foi atingido logo na segunda iteração , e como este é considerado perto do ótimo real , o fornecimento da solução inexato pelo Gino fez com que o algoritmo convirja . A tabela a seguir mostra a sequência da resolução :

iteração	função objetiva	variáveis inteiras	variáveis contínuas
0	134.71310524	(1 , 0 , 0 , 0 , 0)	(1.648829,4.7×10^{-5}, -3.4×10^{-5},-7×10^{-6},0, 1.7×10^{-5})
2	74.29403006	(0 , 1 , 1 , 0 , 0)	(0,2, -2.3×10^{-5}, 5×10^{-6},1.600044, 1.600028)
4	124.29468505	(0 , 1 , 0 , 0 , 0)	(0,2, -7×10^{-6}, 8×10^{-6},0, -1.6×10^{-5})

(tabela 4)

O valor do cálculo do teste de parada 2 foi de 50.00065498 . Note que a tolerância de -0.0001 também não foi suficiente para cobrir o erro .

O tamanho da região de confiança neste caso é 100 , que é o valor default do programa que indica o infinito , o que significa que para este problema , o algoritmo convergiu para um mínimo global .

Se modificarmos o teste da parada 2 para uma tolerânica de 51 o resultado ótimo será o seguinte :

função objetivo : 73.035316

variável inteira : (0,1,1,1,0)

variável contínua : (0,2,0.652015,0.326007,1.078388,1.078388)

região de confiança : 100.0000

interação : 20

tempo de execução : 2min 37seg 47

A solução ótima foi encontrada na 16^{0} iteração

Problema teste 3 . (R3.pas)

A patir do ponto inicial não viável (7 , 5) , fornecido pelo problema , o algoritmo não apresentou uma solução satisfatória , pois na segunda iteração , o novo canditado fornecido pelo Lindo foi o ponto inteiro viável (4 , 3) , o teste de parada 2 foi conferido logo nessa iteração. Onde f (7 , 5) = 159 e f (4 , 3) = 200.

Foi feito um teste para este programa substituindo o ponto inicial não viável por (4 , 3) para observar o comportamento do algoritmo em relação aos problemas não convexos mas com uma boa escolha do ponto inicial . O resultado pelo menos para este caso tem sido bastante satisfatório , pois o algoritmo convergiu na quarta iteração com tempo de execução de 12,63 segundos e para um ótimo global real . Olhe a seguinte tabela para a sequência da resolução :

iteração	ponto inteiro
0	(4 , 3)
2	(5 , 3)
4	(5 , 3)

(tabela 5)

Com estes resultados , é óbvio que na quarta iteração o teste de parada 2 foi satisfatório .

O tamanho da região da confiança para o caso com ponto inicial (4 , 3) é 100 , mas como neste caso a função é f não é convexa, não podemos garantir que este ponto seja um mínimo global .

Os resultados dos problemas testes poderão mais fácil de serem visualizados e comparados a partir das tabelas :

Exemplo 1 :

	primeiro método	segundo método	terceiro método	Duran Grossmann
função objetivo	7.165374	7.1654	7.165374	6.00972
solução de variáveis inteira	(1 , 0 , 0)	(1 , 0 , 0)	(1 , 0 , 0)	(0 , 1 , 0)
solução de variáveis contínuas	(1.312195 , 1.312095 , 0.838269)	(1.3 E +∞ , 1.3 E +∞ , 8.4 E +∞)	(1.312195 , 1.312095 , 0.838269)	(1.30097 , 0 , 1)
tempo de execução	1m5s03	1m47s15	1m12s17	_____
número de iterações	7	14	10	_____

(tabela 6)

Exemplo 2 :

	primeiro método	segundo método	terceiro método	Duran Grossmann
função objetivo	73.035316	134.713505	74.294030	73.0353
solução de variáveis inteira	(0,1,1,1,0)	(1,0,0,0,0)	(0,1,1,0,0)	(0,1,1,1,0)
solução de variáveis contínuas	(0 , 2 , 0.652015 , 0.326007 , 1.078388 , 1.078388)	(1.6 E +∞ , 14.7 E -05 , -3.4 E -05 , -7.0 E -06 , 0 , 1.7 E -05)	(0 , 2 , -0.000023 , 0.000005 , 1.600044 , 1.600028)	(0 , 2 , 0.65201 , 0.32601 , 1.07839 , 1.07839)
tempo de execução	1m52s59	19s38	17s24	_____
número de iterações	10	2	4	_____

(tabela 7)

Exemplo 3 :

	primeiro método	segundo método	terceiro método	Loh Papalambros
função objetivo	59	159	159	159
solução de variáveis inteira	(não viável) (7 , 5)	(5 , 3)	(5 , 3)	(5 , 3)
solução de variáveis contínuas	_____	_____	_____	_____
tempo de execução	11s31	31s80	12s63	_____
número de iterações	2	5	4	_____

(tabela 8)

Conclusão

Pelos testes computacionais apresentados no capítulo anterior , podemos nitidamente perceber as vantagens e desavantagens que cada algoritmo apresenta . Durante a implementação e execução dos testes , houve uma observação interessante . O modelo de programação linear mista discreta , o que chamamos de problema principal ou " master " proposto pelo terceiro algoritmo (Algoritmo de Aproximação Poliédrica com Região de Confiança) apresentou na resolução do Lindo menos soluções inviáveis do que o Algoritmo de Linearização Sequencial , pois neste método teve iterações onde o Lindo não consequiu resolver os problemas lineares mistos , fornecendo as soluções das variáveis inteiras com valores reais . O algoritmo de Aproximação Externa teve também um ótimo desempenho nesse aspecto , tanto que o teste de parada deste algoritmo é justamente quando a solução do problema " master " for inviável .

Apesar do terceiro algoritmo exigir duas chamadas de Lindo para encontrar uma nova solução mista inteira , por problemas do próprio pacote como já foi comentado anteriormente , este algoritmo consegue convergir ou terminar o algoritmo em menos passos do que o Algoritmo de Aproximação Externa , pois este algoritmo precisou enumerar seis dos oito combinações possíveis em um conjunto de três variáveis inteiras assumindo valores 0 e 1 do primeiro problema teste e nove das trinta e duas combinações possíveis no segundo problema teste . Enquanto o Algoritmo de Aproximação Poliédrica com Região de Confiança necessitou de apenas cinco e dois para resolver o primeiro e segundo problema respectivamente . Olhando para a tabela comparativo do problema teste 1 , podemos perceber apesar do terceiro algoritmo apresentar um número de iteração maior , o tempo gasto não é muito maior em relação ao primeiro método , pois várias chamadas de Lindo foram feitas para efetuar tarefas fácieis como criação de arquivos e extensão das restrições . Essa diferença de gasto de tempo poderá diminuir ainda se o tamanho do problema não linear mista discreta original for muito mais extenso , com uma grande quantidade de restrições não lineares e

variáveis inteiras . O que torna o Algoritmo de Aproximação Externa impraticável mesmo tendo uma boa convergência .

O Algoritmo de Linearização Sequencial apesar de ter convergido no primeiro problema teste num gasto de tempo e número de passos considerados razoáveis , o seu desempenho em geral não é satisfatório , pois logo no seguinte teste , o algoritmo já convergiu para um ponto que não é solução do problema original . Houve falhas no algoritmo que foram corrigidas durante a implementação do mesmo , assim como o corte inteiro , o arredondamento das variáveis inteiras e o procedimento para encontrar o primeiro ponto inicial viável . Tudo isso já foi dito nos capítulos anteriores.

Foi detectado também que o sucesso do Algoritmo de Linearização Sequencial dependia muito do ponto inteiro inicial dado . Uma vez que o algoritmo encontrou um ponto mínimo , as iterações seguintes apenas diminuem o tamanho da região de confiança e isto fez com que o algoritmo encontrou a mesma solução , satisfazendo assim o teste de parada . Este algoritmo converge para o mínimo mais próximo de onde estiver localizado o ponto inicial .

Já estava prevista desde o início que o Algoritmo de Aproximação Externa não convirja para os casos onde as funções que envolvem o problema são não convexas . Ele atende apenas uma classe de funções com características mais particulares , pois vai convexificando cada vez mais o problema , então se o ponto ótimo não ficar numa região viável do problema , ela não será mais levada em conta . Foi o que aconteceu com o terceiro problema teste , logo na segunda iteração encontrou um problema linear mista discreta inviável , conferindo o teste de parada . E o Algoritmo de Aproximação Poliedral com Região de Confiança apesar de não ter sido analisado para os casos não convexos teoricamente , mas convergiu para o terceiro problema teste , porém não com o ponto inicial fornecido pelo problema , e sim com um ponto viável pego aleatóriamente , e o tempo gasto foi mínimo . Este se deve a que um ponto não considerado em uma iteração pode ser considerado em iterações posteriores , neste sentido é mais flexível e portanto espera-se um desempenho melhor que o anterior para os casos não convexos . Ao passo que o Algoritmo de Linearização Sequencial precisou fazer duas iterações a mais e o tempo de execução quase o tríplo .

Independentemente das soluções encontradas pelo terceiro algoritmo , notemos que em geral , há um desempenho melhor em termos de números de passos necessários e tempo de execução . Na análise teórico o Algoritmo de Aproximação Poliedral com Região de Confiança é supostamene melhor do que os outros dois algoritmos apresentados , pois foi construído tirando o que tem de vantagens dos mesmos . Só que na prática , não conseguimos provar isso efetivamente , com a utilização do pacote fechado Gino . Devido a sua inexatidão na hora de solucionar o problema não lineares projetados , o teste de parada 2 do terceiro algoritmo torna-se sensível para pontos próximos da solução ótima. Isto fez com que o algoritmo conferisse o teste de parada 2 sem mesmo ter chegado ao ótimo (problema teste 2) ou em situaçõa contrária , onde o teste não foi satisfeito quando se chega ao ótimo global (problema teste 1) .

No terceiro algoritmo , foi feito também um teste modificando a convergência da região de confiança usando a norma um ao invés da norma infinito. Verificamos no problema teste 1 , com a norma um , não houve a necessidade de extender as restrições . A solução ótima foi resolvida em 10 iterações e o tempo gasto foi um pouco a mais do que a metade gasto com a norma infinito . Entretanto , o tamanho da região de confiança na solução ótima foi de 1.99995 , o que garante apenas que seja um mínimo local . Com a norma infinito , na primeira tentativa de diminuir a região de confiança , houve a necessidade de extensão das restrições pois a convergência foi muito rápida , logo na primeira tentativa já conferiu o teste de limite inferior da região de confiança . A vantagem é que quando extendemos as restrições , o tamanho da região de confiança volta ao infinito (no programa representado por valor 100) , o que garante que a solução ótima encontrada é um mínimo global . Nos problemas testes 2 e 3 não houve mudanças quando modificamos a norma .

Por falta de tempo , não foi possível adaptar os algoritmos ao pacote aberto Minus implementado em Fortran destinado para resolver problemas não lineares , semelhante ao Gino . Este trabalho de adaptação já está em andamento , e pretende-se apresentar os trabalhos num futuro próximo .

Também fica em aberto uma análise mais profunda sobre a sensibilidade do teste de parada 2 considerado no terceiro método e a possibilidade de incorporar outros testes .

Recentemente foi lançado um estudo baseado no trabalho de Duran e Grossmann , desenvolvido por R.Fletcher e S.Leyffer [14] , onde utilizam o método de Aproximação Externa para resolver os problemas não lineares misto discretos , mas sem as variáveis inteiras assumindo a linearidade . Possibilitando assim a resolução de uma classe muito maior de problemas não lineares misto discretos.

Apêndice A

```
{*************************************************************************
 *                                                                       *
 *   Este programa foi baseado no algoritmo de Aproximacao Externa,      *
 *   elaborado por Duran & Grossmann, que tem por objetivo de resolver   *
 *   problemas nao lineares com variaveis mista-inteiras.                *
 *                                                                       *
 *   Exemplo 1, tirado do "paper" de Duran & Grossmann                   *
 *                                                                       *
 *   Autora : Hsing Pei Chin ( maio de 1994 )                            *
 *                                                                       *
 *************************************************************************}
{$m $8000,0,$4000}
uses
  Dos;
const
  Ydimensao = 3;
var
  X,
  X_otimo    : array[1..3] of real ;

  YCoef,
  Y_otimo,
  Y          : array[1..Ydimensao] of longint ;
  Dominio_Y  : array[1..7] of array[1..Ydimensao] of longint ;

  Z_otimo,
  Z_minimo   : real;
  ZConst     : longint;

  PPviavel,
  PLMDviavel : boolean;

  k,
  iteracao,            { numero de chamadas de LINDO }
  cont,                { numero de Y nao viaveis }
  conjunto   : longint;

  corte      : string;
{-------------------------------------------------------------------------}

procedure Inicializa_Variaveis;
begin
  YCoef[1]:= 5;
  YCoef[2]:= 6;
  YCoef[3]:= 8;
  Y[1]:= 0;
  Y[2]:= 1;
  Y[3]:= 0;
  ZConst:= 10;
  Z_otimo:= 100.0;
```

```
  Z_minimo:= -100.0;
  cont:= 0;
  iteracao:= 0;
  PPviavel:= true;
  PLMDviavel:= true;
  settime( 0, 0, 0, 0);
end;

{-------------------------------------------------------------------------}

procedure Cria_Arquivo_Bat_Gino;
var
  arq : text;

begin
  assign(arq,'GINO1.BAT');
  rewrite(arq);
  writeln(arq,'RETR GINOENT1.TXT');
  writeln(arq,'DIVERT GINOOUT1.TXT');
  writeln(arq,'GO');
  writeln(arq,'QUIT');
  close(arq);
end;

{-------------------------------------------------------------------------}

procedure Prepara_Modelo;
var
  arq   : text;
  soma,
  i     : longint;

begin
  soma:= 0;
  assign( arq, 'GINOENT1.TXT' );
  rewrite(arq);
  writeln(arq, 'MODEL');
  for i:=1 to Ydimensao do
    soma:= soma + Y[i] * Ycoef[i];
  soma:= soma + ZConst;
  writeln(arq, 'MIN = 10 * X1 - 18 * LOG(X2 + 1) - 19.2 * LOG(X1 - X2 + 1) - 7 * X3 +
', soma,';');
  writeln(arq, '- .8 * LOG(X2 + 1) - .96 * LOG(X1 - X2 + 1) + .8 * X3 < 0;');
  writeln(arq, '- X1 + X2 < 0;');
  writeln(arq, 'X2 + ', -2*Y[1], ' < 0;');
  writeln(arq, 'X1 - X2 + ', -2*Y[2], ' < 0;');
  writeln(arq, '-LOG(X2 + 1) - 1.2 * LOG(X1 - X2 + 1) + X3 - 1 < 0;');
  writeln(arq,'X1 > 0;');
  writeln(arq,'X1 < 2;');
  writeln(arq,'X2 > 0;');
  writeln(arq,'X2 < 2;');
  writeln(arq,'X3 > 0;');
  writeln(arq,'X3 < 1;');
```

```
  writeln(arq,'END');
  close (arq);
end;

{-------------------------------------------------------------------------}

procedure Verifica_Se_PPviavel;
var
  arq   : text;
  linha : string;

begin
  assign(arq, 'GINOOUT1.TXT');
  reset (arq);
  readln(arq, linha );
  if copy(linha, 20, 10) = 'INFEASIBLE 'then
    PPviavel:= false
  else
    PPviavel:= true;
  close(arq);
end;

{-------------------------------------------------------------------------}

procedure Gera_Novo_Y ;
var
  identico : boolean;
  i,j      : longint;

begin
  repeat
    randomize;
    for i:= 1 to Ydimensao do
      Y[i]:= random(2);
    for i:= 1 to cont do
      begin
        j:= 1;
        repeat
          identico:= true;
          if Dominio_Y[i,j] = Y[j] then
            inc(j)
          else
            identico:= false;
        until (j=Ydimensao) or (identico=false);
        if (j=Ydimensao) and (identico=true) then
          i:= cont+1;
      end;
  until identico=false;
end;

{-------------------------------------------------------------------------}

Procedure Modelo_Linear_Inicial;
```

```
 var
   arq   : text;

 begin
   assign(arq, 'LIN_ENT1.TXT');
   rewrite(arq);
   writeln(arq,'MIN 5y1 + 6y2 + 8y3 -r + int');
   writeln(arq,'ST');
   writeln(arq,'int = 10');
   writeln(arq,'x1 - x2 - 2y2 < 0');
   writeln(arq,'x2 - x1 < 0');
   writeln(arq,'x2 - 2y1 < 0');
   writeln(arq,'y1 + y2 < 1');
   writeln(arq,'x1 > 0');
   writeln(arq,'x1 < 2');
   writeln(arq,'x2 > 0');
   writeln(arq,'x2 < 2');
   writeln(arq,'x3 > 0');
   writeln(arq,'x3 < 1');
   writeln(arq,'END');
   writeln(arq,'INTEGER y1');
   writeln(arq,'INTEGER y2');
   writeln(arq,'INTEGER y3');
   writeln(arq,'LEAVE');
   close(arq);

   assign(arq, 'LIN_INI1.BAT');
   rewrite(arq);
   writeln(arq, 'TAKE LIN_ENT1.TXT');
   writeln(arq, 'SAVE LIN_ENT1.SAV');
   writeln(arq, 'QUIT');
   close(arq);
   exec('\dos\command.com','/C lindo < LIN_INI1.BAT');  { chamada do lindo }
   inc(iteracao);
 end;

 {-------------------------------------------------------------------------}

 Procedure Obter_X_na_Saida_do_Gino;
 var
   i, j, w,
   erro    : integer;
   arq     : text;
   linha   : string;
   letra   : char;

 begin
   assign(arq, 'GINOOUT1.TXT');
   reset(arq);

   for i:= 1 to 5 do
     readln(arq, linha);
```

```
  j:= ord(linha[0]);

  for i:= 1 to 3 do
    readln(arq, linha);

  for w:= 1 to 3 do
    begin
      i:=j;
      letra:= linha[i];
      while letra <> ' ' do
        begin
          dec(i);
          letra:=linha[i];
        end;
      val(copy(linha, i+1, j-i ), X[w], erro);
      readln(arq, linha);
    end;
  close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Atualizar_Otimos;
var
  erro,
  i,j   : integer;
  arq   : text;
  linha : string;
  letra : char;
  Zaux  : real;

begin
  assign(arq, 'GINOOUT1.TXT');
  reset(arq);  { arq representa GINOOUT1.TXT }
  for i:=1 to 5 do     { na 5a linha contem o Z_otimo }
    readln(arq, linha);
  j:= ord(linha[0]);
  i:=j;
  repeat
    letra:=linha[i];
    dec(i);
  until letra = ' ';
  val(copy(linha, i+1, j-i ), Zaux, erro);

{ substituicao de valores otimos }
  if Zaux < Z_otimo then
    begin
      Z_otimo:= Zaux;
      for i:= 1 to 3 do
        begin
          X_otimo[i]:= X[i];
          Y_otimo[i]:= Y[i];
        end;
```

```
   end;
  close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Cria_Arquivo_Bat_Lindo;
const
  strX : array[1..3] of string = ('x1', 'x2', 'x3');

var
  fx, g1, g2 , valor  : real;
  df,dg1,dg2,n        : array [1..3] of real;
  restricao           : array [1..3] of string;
  sn, sdf, sdg1, sdg2 : array [1..3] of string[20];
  i, erro             : integer;
  arq                 : text;

begin
  for i:= 1 to 3 do
    begin
      sn[i]:='';
      sdf[i]:='';
      sdg1[i]:='';
      sdg2[i]:='';
      restricao[i]:='';
    end;

  df[1]:= 10 - 19.2/(x[1]-x[2]+1);

  dg1[1]:= -0.96/(x[1]-x[2]+1);

  dg2[1]:= -1.2/(x[1]-x[2]+1);

  df[2]:= -18/(x[2]+1) + 19.2/(x[1]-x[2]+1);

  dg1[2]:= -0.8/(x[2]+1) + 0.96/(x[1]-x[2]+1);

  dg2[2]:= -1/(x[2]+1) +1.2/(x[1]-x[2]+1);

  df[3]:= -7;

  dg1[3]:= 0.8;

  dg2[3]:= 1;

  if df[1] <> 0 then
    begin
      str( df[1]:8:6, sdf[1] );
      restricao[1]:= sdf[1] + strX[1];
    end;

  if dg1[1] <> 0 then
```

```
  begin
    str( dg1[1]:8:6, sdg1[1] );
    restricao[2]:= sdg1[1] + strX[1];
  end;

if dg2[1] <> 0 then
  begin
    str( dg2[1]:8:6, sdg2[1] );
    restricao[3]:= sdg2[1] + strX[1];
  end;

for i:=2 to 3 do
  begin
    if df[i] <> 0 then
      begin
        str( df[i]:8:6, sdf[i] );
        if df[i] > 0  then
          if restricao[1] <> '' then
            restricao[1]:= restricao[1] + '+' + sdf[i] + strX[i]
          else
            restricao[1]:= restricao[1] + sdf[i] + strX[i]
        else
          restricao[1]:= restricao[1] + sdf[i] + strX[i];
      end;

    if dg1[i] <> 0 then
      begin
        str( dg1[i]:8:6, sdg1[i] );
        if dg1[i] > 0  then
          if restricao[2] <> '' then
            restricao[2]:= restricao[2] + '+' + sdg1[i] + strX[i]
          else
            restricao[2]:= restricao[2] + sdg1[i] + strX[i]
        else
          restricao[2]:= restricao[2] + sdg1[i] + strX[i];
      end;

    if dg2[i] <> 0 then
      begin
        str( dg2[i]:8:6, sdg2[i] );
        if dg2[i] > 0  then
          if restricao[3] <> '' then
            restricao[3]:= restricao[3] + '+' + sdg2[i] + strX[i]
          else
            restricao[3]:= restricao[3] + sdg2[i] + strX[i]
        else
          restricao[3]:= restricao[3] + sdg2[i] + strX[i];
      end;
  end;


fx:= 10*x[1] - 7*x[3] - 18*ln(x[2]+1) - 19.2*ln(x[1]-x[2]+1);
```

```
  g1:= -0.8*ln(x[2]+1) - 0.96*ln(x[1]-x[2]+1) - 0.8*x[3];

  g2:= -ln(x[2]+1) - 1.2*ln(x[1]-x[2]+1) + x[3];

  n[1]:= fx-(df[1]*x[1]+df[2]*x[2]+df[3]*x[3]);

  n[2]:= g1-(dg1[1]*x[1]+dg1[2]*x[2]+dg1[3]*x[3]);

  n[3]:= g2-(dg2[1]*x[1]+dg2[2]*x[2]+dg2[3]*x[3])-2;


  for i:=1 to 3 do
   begin
    valor:=0;
    if n[i] <> 0 then
     begin
      if n[i] > 0 then
       begin
        str( n[i]:8:6, sn[i]);
        sn[i]:= '-' + sn[i];
       end
      else
       begin
        str( n[i]:8:6, sn[i]);
        val( sn[i],valor,erro);
        valor:=(-1)*valor;
        str( valor:8:6, sn[i] );
       end
     end
    else
     sn[i]:= ' 0 ';
   end;


  assign(arq,'LINDO1.BAT');
  rewrite(arq);
  writeln(arq,'RETRIVE LIN_ENT1.SAV');
  writeln(arq,'EXTEND');
  writeln(arq,'5y1 + 6y2 + 8y3 - r < ',Z_otimo:8:4);
  writeln(arq,'5y1 + 6y2 + 8y3 - r > ',Z_minimo:8:4);
  writeln(arq, corte + ' < ',conjunto);
  writeln(arq, restricao[1] + ' - r < ',sn[1]);
  writeln(arq, restricao[2] + '< ',sn[2]);
  writeln(arq, restricao[3] + ' + 2y3 < ',sn[3]);
  writeln(arq,'END');
  writeln(arq,'SAVE LIN_ENT1.SAV');
  writeln(arq,'PAGE 0');
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT1.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);
end;
```

```
{-------------------------------------------------------------------------}

Procedure Verifica_PLMD;
var
  arq    : text;
  linha  : string;
  letra  : char;
  i,j,w,
  erro   : integer;
  aux    : real;

begin
  assign(arq, 'LIN_OUT1.TXT');
  reset(arq);
  repeat
    readln(arq, linha);
  until linha <> '';
  i:= 1;
  repeat
    letra:= linha[i];
    inc(i);
  until letra <> ' ';
  if letra = 'W' then
    begin
      PLMDviavel:= false;
      close(arq);
    end
  else
    begin
      repeat
        readln(arq, linha);
      until linha <> '';


      j:= ord(linha[0]) - 4 ;
      i:= j;
      repeat
        letra:= linha[i];
        dec(i);
      until letra = ' ';
      val(copy(linha, i+1, j-i ), Z_minimo, erro);

      w:= 3;
      repeat
        readln(arq, linha);
        dec(w);
      until w = 0;

      for w:= 1 to 3 do  { sao 3 variaveis inteiras }
        begin
          i:= j + 3;
          repeat
```

```
          letra:= linha[i];
          dec(i);
        until letra = ' ';
        val(copy(linha, i+1, j-i ), aux, erro);
        Y[w]:= round(aux);
        readln(arq, linha);
      end;
    close (arq);
  end;
end;

{-------------------------------------------------------------------------}

Procedure Int_Cut;
Const
 strY      : array [1..3] of string = ('Y1', 'Y2', 'Y3');

Var
 i        : longint;

begin
 corte:='';
 conjunto:=0;
 for i:=1 to 3 do
   begin
    if Y[i] = 1 then
      begin
       if corte = '' then
           corte:= corte + strY[i]
          else
           corte:= corte + '+' + strY[i];
       conjunto:=conjunto + 1;
      end
    else
      corte:= corte + '-' + strY[i];
  end;
 conjunto:=conjunto-1;
end;

{-------------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
 hora,
 minuto,
 segundo,
 seg100   : word;

begin
 gettime(hora, minuto, segundo, seg100);
 writeln('O Valor da Funcao Otima : ', Z_otimo:8:6);
 writeln;
 writeln;
```

```
  writeln('As variaveis inteiras otimas sao : ');
  writeln;
  writeln('y1     : ', Y_otimo[1]:3);
  writeln('y2     : ', Y_otimo[2]:3);
  writeln('y3     : ', Y_otimo[3]:3);
  writeln;
  writeln;
  writeln('As variaveis continuas otimas sao :');
  writeln;
  writeln('x1     : ', X_otimo[1]:10:6);
  writeln('x2     : ', X_otimo[2]:10:6);
  writeln('x3     : ', X_otimo[3]:10:6);
  writeln;
  writeln;
  writeln('Numero de iteracoes : ', iteracao :3);
  writeln;
  writeln('Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
          hora,':', minuto,':', segundo,'.', seg100);
end;

(*-------------------------------------------------------------------------
                    PROGRAMA PRINCIPAL
-------------------------------------------------------------------------*)
{ 1a PARTE DO PROGRAMA: VERIFICA A VIABILIDADE DA VARIAVEL
INTEIRA Y INICIAL, CASO NAO FOR VIAVEL, TENTA ACHAR UMA OUTRA
COMBINACAO DE Y TAL QUE SATISFACA A CONDICAO E ELIMINA A Y
INVIAVEL DO DOMINIO.                                    }

begin
  Inicializa_Variaveis;
  Cria_Arquivo_Bat_Gino;
  repeat
    Prepara_Modelo;
    exec('\dos\command.com','/C gino < GINO1.BAT');  { chamada do gino }
    Verifica_Se_PPviavel;
    if PPviavel = false then
      begin
        inc(cont);
        for k:= 1 to Ydimensao do
          Dominio_Y[cont,k]:= Y[k];
        Gera_Novo_Y;
      end
  until PPviavel=true;

{ 2a PARTE DO PROGRAMA: RESOLVER O PROBLEMA MASTER LINEAR
MISTA PELO LINDO   }

  Modelo_Linear_Inicial;

  repeat


    Obter_X_na_Saida_do_Gino;
```

```
    Int_Cut;
    if PPviavel=true then
      Atualizar_Otimos;
    Cria_Arquivo_Bat_Lindo;
    exec('\dos\command.com','/C lindo < LINDO1.BAT');  { chamada do lindo }
    inc(iteracao);
    Verifica_PLMD;
    if PLMDviavel=true then
      begin
        Prepara_Modelo;
        exec('\dos\command.com','/C gino < GINO1.BAT');  { chamada do gino }
        Verifica_Se_PPviavel;
      end;
  until PLMDviavel=false;
  if PLMDviavel=false then
    Mostra_Resultado_Final;
end.
```

```
{*************************************************************
 *                                                           *
 *   Este programa foi baseado no algoritmo de Aproximacao Externa,  *
 *   elaborado por Duran & Grossmann, que tem por objetivo de resolver  *
 *   problemas nao lineares com variaveis mista-inteiras.    *
 *   Exemplo 3, tirado do "paper" de Loh & Papalambros       *
 *                                                           *
 *   Autora : Hsing Pei Chin ( setembro de 1994 )            *
 *                                                           *
*************************************************************}
{$m $8000,0,$4000}
uses
  Dos;
type
  Vetor  = array[1..2] of integer;
var

  Y_otimo,
  Y         : array[1..2] of integer ;

  Z_aux,
  Z_otimo,
  Z_minimo   : real;

  PLMDviavel : boolean;

  k,
  iteracao   : integer;          { numero de chamadas de LINDO }

{-------------------------------------------------------------------------}

{Function fx(vi:Vetor):real;
var
  soma   : real;
begin
  soma:=0;
  soma:= -9*vi[1]*vi[1]+10*vi[1]*vi[2]-50*vi[1]+8*vi[2]+460;
  fx:= soma;
end;}

{-------------------------------------------------------------------------}

Procedure Atualiza_Otimos;
Var
  i     : integer;
begin
  if Z_aux < Z_otimo then
    begin
      Z_otimo:=Z_aux;
      for i:=1 to 2 do
```

```
        Y_otimo[i]:=Y[i];
    end;
end;

{-------------------------------------------------------------------------}

Procedure Inicializa_Variaveis;
Var
  i   : integer;
begin
  Y[1]:= 4;
  Y[2]:= 3;
  for i:=1 to 2 do
    Y_otimo[i]:= Y[i];
  Z_otimo:=1000;
  Z_aux:=0;
  Z_minimo:= -100.0;
  iteracao:= 0;
  PLMDviavel:= true;
  settime( 0, 0, 0, 0);
end;

{-------------------------------------------------------------------------}


Procedure Modelo_Linear_Inicial;
var

  arq   : text;
begin
  assign(arq, 'LIN_ENT3.TXT');
  rewrite(arq);
  writeln(arq,'MIN -R + int');
  writeln(arq,'ST');
  writeln(arq,'int = 460');
  writeln(arq,'END');
  writeln(arq,'LEAVE');
  close(arq);

  assign(arq, 'LIN_INI3.BAT');
  rewrite(arq);
  writeln(arq, 'TAKE LIN_ENT3.TXT');
  writeln(arq, 'SAVE LIN_ENT3.SAV');
  writeln(arq, 'QUIT');
  close(arq);
  exec('\dos\command.com','/C lindo < LIN_INI3.BAT');  { chamada do lindo }
  inc(iteracao);
end;
```

```
{-------------------------------------------------------------------------}

Procedure Cria_Arquivo_Bat_Lindo;
const
  strX : array[1..2] of string = ('y1', 'y2');

var
  f, g1, g2 , valor  : real;
  df,dg1,dg2         : array [1..2] of real;
  restricao          : array [1..3] of string;
  sdf, sdg1, sdg2    : array [1..2] of string[20];
  n                  : array [1..3] of real;
  sn                 : array [1..3] of string[20];
  i, erro            : integer;
  arq                : text;

begin
  for i:= 1 to 3 do
    begin
      sn[i]:='';
      restricao[i]:='';
    end;
  for i:= 1 to 2 do
    begin
      sdf[i]:='';
      sdg1[i]:='';
      sdg2[i]:='';
    end;

  df[1]:= - 18*y[1]+10*y[2]-50;

  df[2]:= 10*y[1]+8;

  dg1[1]:= 1;

  dg1[2]:= -2*0.2768*y[2]-0.235;

  dg2[1]:= 1;

  dg2[2]:= -3*(-0.019*y[2]*y[2]+2*0.446*y[2]-3.98);

  if df[1] <> 0 then
    begin
      str( df[1]:8:6, sdf[1] );
      restricao[1]:= sdf[1] + strX[1];
    end;

  if dg1[1] <> 0 then
    begin
```

```
    str( dg1[1]:8:6, sdg1[1] );
    restricao[2]:= sdg1[1] + strX[1];
   end;

 if dg2[1] <> 0 then
  begin
    str( dg2[1]:8:6, sdg2[1] );
    restricao[3]:= sdg2[1] + strX[1];
  end;

 for i:=2 to 2 do
  begin
    if df[i] <> 0 then
      begin
        str( df[i]:8:6, sdf[i] );
        if df[i] > 0  then
          if restricao[1] <> '' then
            restricao[1]:= restricao[1] + '+' + sdf[i] + strX[i]
          else
            restricao[1]:= restricao[1] + sdf[i] + strX[i]
        else
          restricao[1]:= restricao[1] + sdf[i] + strX[i];
      end;

    if dg1[i] <> 0 then
      begin
        str( dg1[i]:8:6, sdg1[i] );
        if dg1[i] > 0  then
          if restricao[2] <> '' then
            restricao[2]:= restricao[2] + '+' + sdg1[i] + strX[i]
          else
            restricao[2]:= restricao[2] + sdg1[i] + strX[i]
        else
          restricao[2]:= restricao[2] + sdg1[i] + strX[i];
      end;

    if dg2[i] <> 0 then
      begin
        str( dg2[i]:8:6, sdg2[i] );
        if dg2[i] > 0  then
          if restricao[3] <> '' then
            restricao[3]:= restricao[3] + '+' + sdg2[i] + strX[i]
          else
            restricao[3]:= restricao[3] + sdg2[i] + strX[i]
        else
          restricao[3]:= restricao[3] + sdg2[i] + strX[i];
      end;
  end;

f:= -9*y[1]*y[1]+10*y[1]*y[2]-50*y[1]+8*y[2];
```

```
g1:= y[1]-(0.2768*y[2]*y[2]-0.235*y[2]+3.718);

g2:= y[1]-(-0.019*y[2]*y[2]*y[2]+0.446*y[2]*y[2]-3.98*y[2]+15.854);


n[1]:= f-(df[1]*y[1]+df[2]*y[2]);

n[2]:= g1-(dg1[1]*y[1]+dg1[2]*y[2]);

n[3]:= g2-(dg2[1]*y[1]+dg2[2]*y[2]);

for i:=1 to 3 do
 begin
  valor:=0;
  if n[i] <> 0 then
   begin
    if n[i] > 0 then
     begin
      str( n[i]:8:6, sn[i]);
      sn[i]:= '-' + sn[i];
     end
    else
     str( n[i]:8:6, sn[i]);

    val( sn[i],valor,erro);
    valor:=(-1)*valor;
    str( valor:8:6, sn[i] );
   end
  else
   sn[i]:= ' 0 ';
 end;



assign(arq,'LINDO3.BAT');
rewrite(arq);
writeln(arq,'RETRIVE LIN_ENT3.SAV');
writeln(arq,'EXTEND');
{  writeln(arq,' R < ',Z_otimo:8:4);
writeln(arq,' R > ',Z_minimo:8:4); }
writeln(arq, restricao[1] + ' - R < ',sn[1]);
writeln(arq, restricao[2] + ' < ',sn[2]);
writeln(arq, restricao[3] + ' < ',sn[3]);
writeln(arq,'END');
writeln(arq,'GIN Y1');
writeln(arq,'GIN Y2');
writeln(arq,'SAVE LIN_ENT3.SAV');
writeln(arq,'PAGE 0');
```

```
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT3.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Verifica_PLMD;
var
  arq    : text;
  linha  : string;
  letra  : char;
  i,j,w,
  erro   : integer;
  aux    : real;

begin
 assign(arq, 'LIN_OUT3.TXT');
 reset(arq);
 repeat
   readln(arq, linha);
 until linha <> '';
 i:= 1;
 repeat
   letra:= linha[i];
   inc(i);
 until letra <> ' ';
 if letra = 'W' then
   begin
    PLMDviavel:= false;
    close(arq);
   end
 else
   begin
    repeat
      readln(arq, linha);
    until linha <> '';


    j:= ord(linha[0]) - 4 ;
    i:= j;
    repeat
      letra:= linha[i];
      dec(i);
    until letra = ' ';
    val(copy(linha, i+1, j-i ), Z_minimo, erro);

    w:= 3;
```

```
    repeat
      readln(arq, linha);
      dec(w);
    until w = 0;

    for w:= 1 to 2 do  { sao 2 variaveis inteiras }
      begin
        i:= j + 3;
        repeat
          letra:= linha[i];
          dec(i);
        until letra = ' ';
        val(copy(linha, i+1, j-i ), aux, erro);
        Y[w]:= round(aux);
        readln(arq, linha);
      end;
    close (arq);
  end;
end;

{-------------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
 hora,
 minuto,
 segundo,
 seg100   : word;

begin
 gettime(hora, minuto, segundo, seg100);
 writeln('O Valor da Funcao Otima : ', Z_otimo:8:6);
 writeln;
 writeln;
 writeln('As variaveis inteiras otimas sao : ');
 writeln;
 writeln('y1      : ', Y_otimo[1]:3);
 writeln('y2      : ', Y_otimo[2]:3);
 writeln;
 writeln;
 writeln;
 writeln('Numero de iteracoes : ', iteracao :3);
 writeln;
 writeln('Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
        hora,':', minuto,':', segundo,'.', seg100);
end;

(*-------------------------------------------------------------------------
                    PROGRAMA PRINCIPAL
-------------------------------------------------------------------------*)
```

```
begin
  Inicializa_Variaveis;

  Modelo_Linear_Inicial;

  repeat

    Z_aux:= -9*y[1]*y[1]+10*y[1]*y[2]-50*y[1]+8*y[2]+460;
    Atualiza_Otimos;
    Cria_Arquivo_Bat_Lindo;
    exec('\dos\command.com','/C lindo < LINDO3.BAT');  { chamada do lindo }
    inc(iteracao);
    Verifica_PLMD;
  until PLMDviavel=false;
  if PLMDviavel=false then
    Mostra_Resultado_Final;
end.
```

Apêndice B

```
{*************************************************************************
*                                                                        *
*   Este Programa foi baseado no algoritmo de Aproximacao  Sequencial    *
*   Linear elaborado por Loh & Papalambros, que tem por objetivo  de     *
*   resolver problemas nao lineares com variaveis mista inteiras.        *
*                                                                        *
*   Exemplo 1, tirado do "paper" de Duran & Grossmann                    *
*                                                                        *
*   Autora : Hsing Pei Chin ( Junho de 1994 )                            *
*                                                                        *
*                                                                        *
*************************************************************************}


{$m $4000,0,$4000}
Uses
  Dos;
Const
  Ydimensao = 3;
Type
  Vetor1 = array[1..Ydimensao] of integer;
  Vetor2 = array[1..3] of real;
Var
  X,
  X_otimo    : Vetor2;

  YCoef,
  Y,
  Y_ant,
  Y_otimo    : Vetor1;

  Dominio_Y  : array[1..7] of Vetor1;

  Z,
  Z_otimo    : real;
  Zconst     : integer;

  num_cut,
  k,
  cont,
  iteracao   : integer;     { numero de chamadas de Lindo }

  PPviavel,
  soma_maior,
  pare,
  teste1,
  teste2     :boolean;

  epslon,
```

```
  epslon_i,
  epslon_f,
  t,
  t_ini,
  delta      : real;

  rt,
  re,
  indice     : integer;

  corte      : array[1..8] of string;
  conjunto   : array[1..8] of integer;

{-------------------------------------------------------------------}

Function suminf(vi:Vetor1;vc:Vetor2):real;
var
  g     : array[1..12] of real;
  i     : integer;
  sum   : real;

begin

 sum:=0;

 g[1]:= -0.8*ln(vc[2]+1)-0.96*ln(vc[1]-vc[2]+1)+0.8*vc[3];
 g[2]:= vc[2]-vc[1];
 g[3]:= vc[2]-2*vi[1];
 g[4]:= vc[1]-vc[2]-2*vi[2];
 g[5]:= -ln(vc[2]+1)-1.2*ln(vc[1]-vc[2]+1)+vc[3]+2*vi[3]-2;
 g[6]:= vi[1]+vi[2]-1;
 g[7]:= vc[1]-2;
 g[8]:= -vc[1];
 g[9]:= vc[2]-2;
 g[10]:= -vc[2];
 g[11]:= vc[3]-1;
 g[12]:= -vc[3];


 for i:=1 to 12 do
   begin
   if g[i] < 0 then
      g[i]:=0;
   sum:= sum+g[i];
   end;
 suminf:=sum;

end;

{-------------------------------------------------------------------}

Function fx(vi:Vetor1;vc:Vetor2):real;
var
```

```
  temp1,
  temp2    :real;
begin
 temp1:=5*vi[1]+6*vi[2]+8*vi[3];
 temp2:=10*vc[1]-7*vc[3]-18*ln(vc[2]+1)-19.2*ln(vc[1]-vc[2]+1)+10;
 fx:= temp1+temp2;
end;

{-------------------------------------------------------------------------}

Procedure Inicializacao;

begin
 YCoef[1]:= 5;
 YCoef[2]:= 6;
 YCoef[3]:= 8;
 Y[1]:= 1;
 Y[2]:= 0;
 Y[3]:= 1;
 Zconst:= 10;
 iteracao:= 0;
 num_cut:= 0;
 cont:= 0;
 Soma_maior:= true;
 pare:=false;
 teste1:= true;
 teste2:= true;
 PPviavel:= true;
 epslon_i:=1;
 epslon_f:=0.001;
 epslon:= epslon_i;
 re:=2;
 t_ini:= 5.0;
 t:= t_ini;
 rt:= 2;
 delta:= 0.001;
 settime( 0, 0, 0, 0);
end;

{-------------------------------------------------------------------------}

Procedure Prepare_Modelo;
var
  arq   : text;
  soma,
  i     : integer;

begin
 soma:= 0;
 assign(arq, 'GINOENT1.TXT' );
 rewrite(arq);
 writeln(arq, 'MODEL');
 for i:=1 to Ydimensao do
```

```
    soma:= soma + Y[i] * YCoef[i];
   soma:= soma + Zconst;
   writeln(arq, 'MIN=10*X1-18*LOG(X2+1)-19.2*LOG(X1-X2+1)-7*X3+',soma,';');
   writeln(arq, '-0.8*LOG(X2+1)-0.96*LOG(X1-X2+1)+0.8*X3<0;');
   writeln(arq, '-X1+X2<0;');
   writeln(arq, 'X2+',-2*Y[1],'<0;');
   writeln(arq, 'X1-X2+',-2*Y[2], '<0;');
   writeln(arq, '-LOG(X2+1)-1.2*log(X1-X2+1)+X3-1<0;');
   writeln(arq, 'X1>0;');
   writeln(arq, 'X1<2;');
   writeln(arq, 'X2>0;');
   writeln(arq, 'X2<2;');
   writeln(arq, 'X3>0;');
   writeln(arq, 'X3<1;');
   writeln(arq, 'END');
   close(arq);
 end;

 {-------------------------------------------------------------------------}

 Procedure Verifica_Se_PPviavel;
 var
    arq    : text;
    linha  : string;

 begin
   assign(arq,  'GINOOUT1.TXT');
   reset(arq);
   readln(arq, linha );
   if copy(linha, 20, 10) = 'INFEASIBLE ' then
     PPviavel := false
   else
     PPviavel := true;
   close(arq);
 end;

 {-------------------------------------------------------------------------}

 Procedure Obter_X_na_Saida_do_Gino;
 var
   i,j,w,
   erro      : integer;
   arq       : text;
   linha     : string;
   letra     : char;

 begin
   assign(arq, 'GINOOUT1.TXT');
   reset(arq);
   for i:=1 to 5 do
     readln(arq, linha);

   j:= ord(linha[0]);
```

```
  for i:=1 to 3 do
    readln(arq, linha);

  for w:=1 to 3 do
    begin
      i:=j;
      letra:= linha[i];
      while letra <> ' ' do
        begin
          dec(i);
          letra:=linha[i];
        end;
      val(copy(linha, i+1, j-i ), X[w], erro);
      readln(arq, linha);
    end;

  close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Gera_Novo_Y;
var
  identico : boolean;
  i,j      : integer;

begin

  repeat
    randomize;
    for i:= 1 to Ydimensao do
    Y[i]:= random(2);
    for i:= 1 to cont do
      begin
        j:=1;
        repeat
          identico:= true;
          if Dominio_Y[i,j] = Y[j] then
            inc(j)
          else
            identico:= false;
        until (j=Ydimensao) or (identico=false);
        if (j=Ydimensao) and (identico=true) then
          i:= cont+1;
      end;

  until identico = false;

end;

{-------------------------------------------------------------------------}
```

```
Procedure Cria_Arquivo_Bat_Gino;
var
  arq : text;

begin
 assign(arq,'Gino1.bat');
 rewrite(arq);
 writeln(arq,'RETR GINOENT1.TXT');
 writeln(arq,'DIVERT GINOOUT1.TXT');
 writeln(arq,'GO');
 writeln(arq,'QUIT');
 close(arq);
end;

{-------------------------------------------------------------------}

Procedure Cria_Arquivo_Bat_Lindo;
var
  arq : text;

begin

 assign(arq,'Lindo1.bat');
 rewrite(arq);
 writeln(arq,'RETR LIN_ENT1.SAV');
 writeln(arq,'PAGE 0');
 writeln(arq,'GO');
 writeln(arq,'DIVERT LIN_OUT1.TXT');
 writeln(arq,'SOLUTION');
 writeln(arq,'QUIT');
 close(arq);
end;

{-------------------------------------------------------------------}

Procedure Prepare_Modelo_Linear;
const
 strYX : array[1..6] of string = ('Y1','Y2','Y3','X1','X2','X3');

var
  arq      : text;

  df,
  dg1,
  dg2      : array[1..6] of real;

  n        : array[1..3] of real;

  valor,
  g1,
  g2       : real;

  erro,
```

```
  i       : integer;

  sdf,
  sdg1,
  sdg2     : array [1..6] of string;

  sn       : array[1..3] of string;

  restricao : array[1..3] of string;

begin
 for i:= 1 to 3 do
  begin
   sn[i]:='';
   restricao[i]:='';
  end;

 for i:= 1 to 6 do
  begin
   sdf[i]:='';
   sdg1[i]:='';
   sdg2[i]:='';
  end;

 df[1]:= 5;

 df[2]:= 6;

 df[3]:= 8;

 df[4]:=10-19.2/(x_otimo[1]-x_otimo[2]+1);

 df[5]:= -18/(x_otimo[2]+1)+19.2/(x_otimo[1]-x_otimo[2]+1);

 df[6]:= -7;

 dg1[1]:= 0;

 dg1[2]:= 0;

 dg1[3]:= 0;

 dg1[4]:= -0.96/(x_otimo[1]-x_otimo[2]+1);

 dg1[5]:= -0.8/(x_otimo[2]+1)+0.96/(x_otimo[1]-x_otimo[2]+1);

 dg1[6]:= 0.8;

 dg2[1]:= 0;

 dg2[2]:= 0;

 dg2[3]:= 2;
```

```
dg2[4]:= -1.2/(x_otimo[1]-x_otimo[2]+1);

dg2[5]:= -1/(x_otimo[2]+1)+1.2/(x_otimo[1]-x_otimo[2]+1);

dg2[6]:= 1;

g1:= -0.8*ln(x_otimo[2]+1)-0.96*ln(x_otimo[1]-x_otimo[2]+1)+0.8*x_otimo[3];

g2:= -ln(x_otimo[2]+1)-1.2*ln(x_otimo[1]-x_otimo[2]+1)+x_otimo[3]
    +2*y_otimo[3]-2;

n[1]:= -(df[1]*y_otimo[1]+df[2]*y_otimo[2]+df[3]*y_otimo[3]
     +df[4]*x_otimo[1]+df[5]*x_otimo[2]+df[6]*x_otimo[3]);

n[2]:= g1-(dg1[1]*y_otimo[1]+dg1[2]*y_otimo[2]+dg1[3]*y_otimo[3]
     +dg1[4]*x_otimo[1]+dg1[5]*x_otimo[2]+dg1[6]*x_otimo[3]);

n[3]:= g2-(dg2[1]*y_otimo[1]+dg2[2]*y_otimo[2]+dg2[3]*y_otimo[3]
     +dg2[4]*x_otimo[1]+dg2[5]*x_otimo[2]+dg2[6]*x_otimo[3]);

if df[1] <> 0 then
 begin
  str( df[1]:8:6 , sdf[1] );
  restricao[1]:= sdf[1] + strYX[1];
 end;

if dg1[1] <> 0 then
 begin
  str( dg1[1]:8:6 , sdg1[1] );
  restricao[2]:= sdg1[1] + strYX[1];
 end;

if dg2[1] <> 0 then
 begin
  str( dg2[1]:8:6 , sdg2[1] );
  restricao[3]:= sdg2[1] + strYX[1];
 end;


for i:=2 to 6 do
 begin
  if df[i] <> 0 then
   begin
    str( df[i]:8:6, sdf[i] );
    if df[i] > 0 then
     if restricao[1] <> '' then
      restricao[1]:= restricao[1] + '+' + sdf[i] + strYX[i]
     else
      restricao[1]:= restricao[1] + sdf[i] + strYX[i]
    else
     restricao[1]:= restricao[1] + sdf[i] + strYX[i]
   end;
```

```
    if dg1[i] <> 0 then
     begin
      str( dg1[i]:8:6, sdg1[i] );
      if dg1[i] > 0 then
       if restricao[2] <> '' then
        restricao[2]:= restricao[2] + '+' + sdg1[i] + strYX[i]
       else
        restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
      else
       restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
     end;

    if dg2[i] <> 0 then
     begin
      str( dg2[i]:8:6, sdg2[i] );
      if dg2[i] > 0 then
       if restricao[3] <> '' then
        restricao[3]:= restricao[3] + '+' + sdg2[i] + strYX[i]
       else
        restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
      else
       restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
     end;
   end;


for i:= 1 to 3 do
  begin
    valor:=0;
    if n[i] <> 0 then
      begin
        if n[i] > 0 then
          begin
            str(n[i]:8:6, sn[i]);
            sn[i]:= '-' + sn[i];
          end
        else
          begin
           str( n[i]:8:6, sn[i]);
           val(sn[i],valor,erro);
           valor:=(-1)*valor;
           str(valor:8:6,sn[i]);
          end;

      end
    else
     sn[i]:= ' 0 ';
  end;

assign(arq, 'LIN_ENT1.TXT');
rewrite(arq);
writeln(arq, 'MIN' + restricao[1] );
```

```
  writeln(arq, 'ST');
{  writeln(arq, 'R =',sn[1]); }
  writeln(arq, restricao[2] + ' < ', sn[2]);
  writeln(arq, restricao[3] + ' <  ',sn[3]);
  for i:=1 to num_cut do
    writeln(arq, corte[i] + ' < ',conjunto[i]);
  writeln(arq, 'X2 - X1 < 0');
  writeln(arq, 'X2 - 2Y1 < 0');
  writeln(arq, 'X1 - X2 - 2Y2 < 0');
  writeln(arq, 'Y1 + Y2 < 1');
  writeln(arq, 'X1 > 0');
  writeln(arq, 'X1 < 2');
  writeln(arq, 'X2 > 0');
  writeln(arq, 'X2 < 2');
  writeln(arq, 'X3 > 0');
  writeln(arq, 'X3 < 1');
  writeln(arq, 'Y1 > ',(-t+Y_otimo[1]):8:6);
  writeln(arq, 'Y1 < ',(t+Y_otimo[1]):8:6);
  writeln(arq, 'Y2 > ',(-t+Y_otimo[2]):8:6);
  writeln(arq, 'Y2 < ',(t+Y_otimo[2]):8:6);
  writeln(arq, 'Y3 > ',(-t+Y_otimo[3]):8:6);
  writeln(arq, 'Y3 < ',(t+Y_otimo[3]):8:6);
  writeln(arq, 'X1 > ',(-t+X_otimo[1]):8:6);
  writeln(arq, 'X1 < ',(t+X_otimo[1]):8:6);
  writeln(arq, 'X2 > ',(-t+X_otimo[2]):8:6);
  writeln(arq, 'X2 < ',(t+X_otimo[2]):8:6);
  writeln(arq, 'X3 > ',(-t+X_otimo[3]):8:6);
  writeln(arq, 'X3 < ',(t+X_otimo[3]):8:6);
  writeln(arq, 'END');
  writeln(arq, 'INTEGER Y1');
  writeln(arq, 'INTEGER Y2');
  writeln(arq, 'INTEGER Y3');
  writeln(arq, 'LEAVE');
  close(arq);

  assign(arq, 'LIN_INI1.BAT');
  rewrite(arq);
  writeln(arq, 'TAKE LIN_ENT1.TXT');
  writeln(arq, 'SAVE LIN_ENT1.SAV');
  writeln(arq, 'QUIT');
  close(arq);
  exec('\dos\command.com','/C lindo < LIN_INI1.BAT');  { chamada do lindo }
  inc(iteracao);

end;

{-------------------------------------------------------------------}

Procedure Ler_Saida_Lindo;
var
   arq      : text;
   linha    : string;
   letra    : char;
```

```
  i,
  j,
  w,
  erro    : integer;
  aux     : real;

begin
 assign(arq, 'LIN_OUT1.TXT');
 reset(arq);
 repeat
   readln(arq, linha);
 until linha <> '';      {Procurar a primeira linha da saida que nao seja
                         solucao}
 i:=1;
 repeat
   letra:= linha[i];     {ler a primeira letra da primeira linha}
   inc(i);
 until letra <> ' ';

 if letra = 'W' then
   begin
     repeat
       readln(arq, linha);
     until linha <> '';
   end;

 repeat
   readln(arq, linha);
 until linha <> '';      {Procurar o valor otima da funcao objetivo}

 j:= ord(linha[0])-4;
 i:=j;
 repeat
   letra:= linha[i];
   dec(i);
 until letra = ' ';
 val(copy(linha, i+1, j-i), Z, erro);

 w:=3;
 repeat                  {Procurar as solucoes das variaveis inteiras}
    readln(arq,linha);
    dec(w);
 until w = 0;

 for w:=1 to 3  do
    begin
      i:=j+3;
      repeat
        letra:= linha[i];
        dec(i);
      until letra = ' ';
      val(copy(linha, i+1, j-i), aux , erro);
      Y[w]:= round(aux);
```

```
      readln(arq, linha);
    end;
  close(arq);
end;

{-------------------------------------------------------------------}

Procedure Int_Cut;
Const
  strY      : array [1..3] of string = ('Y1', 'Y2', 'Y3');

Var
  i         : integer;

begin
  inc(num_cut);
  corte[num_cut]:='';
  conjunto[num_cut]:=0;
  for i:=1 to 3 do
    begin
     if Y[i] = 1 then
       begin
        if corte[num_cut] = '' then
            corte[num_cut]:= corte[num_cut] + strY[i]
           else
            corte[num_cut]:= corte[num_cut] + '+' + strY[i];
        conjunto[num_cut]:=conjunto[num_cut] + 1;
       end
     else
       corte[num_cut]:= corte[num_cut] + '-' + strY[i];
    end;
  conjunto[num_cut]:=conjunto[num_cut]-1;
end;

{-------------------------------------------------------------------}

Procedure Atualiza_Epslon;
begin
  if suminf(y,x)/re > epslon_f then
     epslon:= suminf(y,x)/re
  else
    epslon:= epslon_f;

end;

{-------------------------------------------------------------------}

Procedure Teste_de_Parada;
var
  dif       : array[1..6] of real;
  i         : integer;
  menor     : boolean;
begin
```

```
  for i:=1 to 3 do
    begin
      dif[i]:=Y[i]-Y_otimo[i];
    end;

  for i:=4 to 6 do
    begin
      dif[i]:=X[i-3]-X_otimo[i-3];
    end;

  i:=1;
  menor:= true;
  repeat
    if abs(dif[i]) < delta then
       begin
         menor:= true;
         inc(i);
       end
    else
      menor:= false;
  until (i = 7) or ( menor = false );

  if (i = 7) and ( menor = true) then
     begin
       teste1:=false;
       teste2:=false;
       pare:=true;
     end
   else pare:= false;
end;

{-------------------------------------------------------------------------}

Procedure Teste_t;
begin
  if abs(t) <  delta then
    begin
      writeln('Erro : Este problema nao esta convergindo.');
      writeln('E a solucao aproximada ate o presente momento e:',Z_otimo:8:6);
      teste1:= false;
      teste2:= false;
    end;
end;

{-------------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
  hora,
  minuto,
  segundo,
  seg100    : word;
```

```
begin
  gettime(hora, minuto, segundo, seg100);
  writeln( 'O Valor da Funcao Otima : ', Z_otimo:8:4);
  writeln;
  writeln;
  writeln( 'As variaveis inteiras otimas sao: ');
  writeln;
  writeln( 'y1      : ', Y_otimo[1]:3);
  writeln( 'y2      : ', Y_otimo[2]:3);
  writeln( 'y3      : ', Y_otimo[3]:3);
  writeln;
  writeln;
  writeln( 'As variaveis continuas otimas sao: ');
  writeln;
  writeln( 'x1      : ', X_otimo[1]:3);
  writeln( 'x2      : ', X_otimo[2]:3);
  writeln( 'x3      : ', X_otimo[3]:3);
  writeln;
  writeln;
  writeln( 'Numero de iteracoes : ', iteracao :3);
  writeln;
  writeln( 'Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
           hora,':', minuto,':', segundo,'.', seg100);

end;

{------------------------------------------------------------------------}

{*-------------------------------------------------------------------------
                    Programa Principal
-------------------------------------------------------------------------*}

begin
  Inicializacao;
  Cria_Arquivo_Bat_Gino;
  Cria_Arquivo_Bat_Lindo;

{ 1a PARTE DO PROGRAMA: Nesta parte do programa tenta em primeiro lugar
  a partir de uma variavel inteira dada inicial, obter uma variavel con_
  tinua que servira para inicializar o resolucao do problema.          }


  repeat
    Prepare_Modelo;
    exec ('\dos\command.com','/C gino < Gino1.bat');  { chamada de gino }
    Verifica_Se_PPviavel;
    if PPviavel = false then
      begin
        inc(cont);
        for k:= 1 to Ydimensao do
          Dominio_Y[cont,k]:= Y[k];
          Gera_Novo_Y;
      end;
```

```
until ppviavel=true;


Int_Cut;
Obter_X_na_Saida_do_Gino;

for k:=1 to 3 do
 begin
   X_otimo[k]:= X[k];
   Y_otimo[k]:= Y[k];              {Variaveis otimas iniciais}
 end;

Z_otimo:=fx(Y_otimo,X_otimo);      {Valor da Z_otima inicial}

repeat
 if suminf(Y_otimo,X_otimo) > epslon then
    soma_maior:= true
  else
    soma_maior:= false;

 repeat

   Prepare_Modelo_Linear;          { RMDLP }
   exec( '\DOS\COMMAND.COM','/C lindo < Lindo1.bat'); { chamada do lindo }
   inc(iteracao);
   Ler_Saida_Lindo;
   Int_Cut;
   Prepare_Modelo;
   exec( '\dos\command.com','/C gino < Gino1.bat'); { chamada de gino }
   Obter_X_na_Saida_do_Gino;
   Teste_De_Parada;

   while not pare do
    begin
     if soma_maior=true then
      begin
       if suminf(y,x) < suminf(y_otimo,x_otimo) then
          begin
           Z_otimo:=fx(y,x);
           for k:=1 to 3 do
            begin
             Y_otimo[k]:=Y[k];
             X_otimo[k]:=X[k];
            end;
           Atualiza_Epslon;
           teste1:=true;
           teste2:=false;
          end
         else
          begin
           t:=t/rt;
           teste2:=true;
           Teste_t;
```

```
            end;
          pare:=true;
        end;
      if soma_maior=false then
        begin
          if suminf(y,x) > epslon then
            begin
              t:=t/rt;
              teste2:=true;
              Teste_t;
            end
          else
            begin
              if fx(y,x) >= fx(y_otimo,x_otimo) then
                begin
                  t:=t/rt;
                  teste2:=true;
                  Teste_t;
                end
              else
                begin
                  Z_otimo:=fx(y,x);
                  for k:=1 to 3 do
                   begin
                     Y_otimo[k]:=Y[k];
                     X_otimo[k]:=X[k];
                   end;
                  Atualiza_Epslon;
                  teste2:=false;
                  teste1:=true;
                end;
            end;
          pare:= true;
        end;
    end;
  until teste2 = false;
until teste1 = false;

Mostra_Resultado_Final;
end.
```

```
{*************************************************************************
*                                                                        *
*   Este Programa foi baseado no algoritmo de Aproximacao  Sequencial    *
*   Linear elaborado por Loh & Papalambros, que tem por objetivo  de     *
*   resolver problemas nao lineares com variaveis mista inteiras.        *
*                                                                        *
*   Exemplo 3, tirado do "paper" de Loh & Papalambros                    *
*                                                                        *
*   Autora : Hsing Pei Chin ( Setembro de 1994 )                         *
*                                                                        *
*                                                                        *
*************************************************************************}

{$m $4000,0,$4000}
Uses
  Dos;
Const
  Ydimensao = 2;
Type
  Vetor1 = array[1..Ydimensao] of integer;
Var

  Y,
  Y_otimo    : Vetor1;

  Z,
  Z_otimo    : real;
  Zconst     : integer;

  k,
  iteracao   : integer;     { numero de chamadas de Lindo }

  soma_maior,
  pare,
  teste1,
  teste2     :boolean;

  epslon,
  epslon_i,
  epslon_f,
  t,
  t_ini,
  delta      : real;

  rt,
  re         : integer;


{-------------------------------------------------------------------------}

Function suminf(vi:Vetor1):real;
var
  g     : array[1..2] of real;
```

```
   i     : integer;
   sum   : real;

 begin

  sum:=0;

  g[1]:= vi[1]-(0.2768*vi[2]*vi[2]-0.235*vi[2]+3.718);
  g[2]:= vi[1]-(-0.019*vi[2]*vi[2]*vi[2]+0.446*vi[2]*vi[2]-3.98*vi[2]+15.854);

  for i:=1 to 2 do
    begin
    if g[i] < 0 then
        g[i]:=0;
    sum:= sum+g[i];
    end;
  suminf:=sum;

end;

{------------------------------------------------------------------------}

Function fx(vi:Vetor1):real;
var
  temp1     : real;
begin
 temp1:=-9*vi[1]*vi[1]+10*vi[1]*vi[2]-50*vi[1]+8*vi[2]+460;
 fx:= temp1;
end;

{------------------------------------------------------------------------}

Procedure Inicializacao;

begin
 Y[1]:= 7;
 Y[2]:= 5;
 iteracao:= 0;
 Soma_maior:= true;
 pare:= false;
 teste1:= true;
 teste2:= true;
 epslon_i:=1;
 epslon_f:=0.01;
 epslon:= epslon_i;
 re:=2;
 t_ini:= 4.0;
 t:= t_ini;
 rt:= 2;
 delta:= 0.001;
 settime( 0, 0, 0, 0);
end;
```

```
{-------------------------------------------------------------------------}


Procedure Cria_Arquivo_Bat_Lindo;
var
  arq : text;

begin

 assign(arq,'Lindo3.bat');
 rewrite(arq);
 writeln(arq,'RETR LIN_ENT3.SAV');
 writeln(arq,'PAGE 0');
 writeln(arq,'GO');
 writeln(arq,'DIVERT LIN_OUT3.TXT');
 writeln(arq,'SOLUTION');
 writeln(arq,'QUIT');
 close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Prepare_Modelo_Linear;
Const
   strYX : array[1..2] of string = ('Y1','Y2');

var
  arq      : text;

  df,
  dg1,
  dg2      : array[1..2] of real;

  n        : array[1..3] of real;

  valor,
  g1,
  g2       : real;

  erro,
  i        : integer;

  sdf,
  sdg1,
  sdg2     : array [1..2] of string[9];


  sn       : array[1..3] of string[9];

  restricao : array[1..3] of string[50];

begin
 sn[1]:='';
```

```
sn[2]:='';
sn[3]:='';

for i:= 1 to 2 do
 begin
  sdf[i]:='';
  sdg1[i]:='';
  sdg2[i]:='';
 end;

df[1]:=-18*y_otimo[1]+10*y_otimo[2]-50;

df[2]:= 10*y_otimo[1]+8;

dg1[1]:= 1;

dg1[2]:= -(2*0.2768*y_otimo[2]-0.235);

dg2[1]:= 1;

dg2[2]:= -(-3*0.019*y_otimo[2]*y_otimo[2]+2*0.446*y_otimo[2]-3.98);

g1:= y_otimo[1]-(0.2768*y_otimo[2]*y_otimo[2]-0.235*y_otimo[2]+3.718);

g2:= y_otimo[1]-(-0.019*y_otimo[2]*y_otimo[2]*y_otimo[2]+0.446*y_otimo[2]
    *y_otimo[2]-3.98*y_otimo[2]+15.854);


n[1]:= -(df[1]*y_otimo[1]+df[2]*y_otimo[2]);

n[2]:= g1-(dg1[1]*y_otimo[1]+dg1[2]*y_otimo[2]);

n[3]:= g2-(dg2[1]*y_otimo[1]+dg2[2]*y_otimo[2]);


 if df[1] <> 0 then
 begin
  str( df[1]:8:6 , sdf[1] );
  restricao[1]:= sdf[1] + strYX[1];
 end;

 if dg1[1] <> 0 then
 begin
  str( dg1[1]:8:6 , sdg1[1] );
  restricao[2]:= sdg1[1] + strYX[1];
 end;

if dg2[1] <> 0 then
 begin
  str( dg2[1]:8:6 , sdg2[1] );
  restricao[3]:= sdg2[1] + strYX[1];
 end;
```

```
  if df[2] <> 0 then
   begin
    str( df[2]:8:6, sdf[2] );
    if df[2] > 0 then
     if restricao[1] <> '' then
      restricao[1]:= restricao[1] + '+' + sdf[2] + strYX[2]
     else
      restricao[1]:= restricao[1] + sdf[2] + strYX[2]
    else
     restricao[1]:= restricao[1] + sdf[2] + strYX[2]
   end;

  if dg1[2] <> 0 then
   begin
    str( dg1[2]:8:6, sdg1[2] );
    if dg1[2] > 0 then
     if restricao[2] <> '' then
      restricao[2]:= restricao[2] + '+' + sdg1[2] + strYX[2]
     else
      restricao[2]:= restricao[2] + sdg1[2] + strYX[2]
    else
     restricao[2]:= restricao[2] + sdg1[2] + strYX[2]
   end;

  if dg2[2] <> 0 then
   begin
    str( dg2[2]:8:6, sdg2[2] );
    if dg2[2] > 0 then
     if restricao[3] <> '' then
      restricao[3]:= restricao[3] + '+' + sdg2[2] + strYX[2]
     else
      restricao[3]:= restricao[3] + sdg2[i] + strYX[2]
    else
     restricao[3]:= restricao[3] + sdg2[i] + strYX[2]
   end;


for i:= 1 to 3 do
 begin
  valor:=0;
  if n[i] <> 0 then
   begin
    if n[i] > 0 then
     begin
      str(n[i]:8:4, sn[i]);
      sn[i]:= '-' + sn[i];
     end
    else
     begin
      str( n[i]:8:4, sn[i]);
      val(sn[i],valor,erro);
      valor:=(-1)*valor;
      str(valor:8:4,sn[i]);
```

```
        end;
      end
    else
     sn[i]:= ' 0 ';
  end;

  assign(arq, 'LIN_ENT3.TXT');
  rewrite(arq);
  writeln(arq, 'MIN'+ restricao[1] );
  writeln(arq, 'ST');
  writeln(arq, restricao[2] + ' < ',sn[2]);
  writeln(arq, restricao[3] + ' < ',sn[3]);
  writeln(arq, 'Y1<',(Y_otimo[1]+t):8:6);
  writeln(arq, 'Y1>',(Y_otimo[1]-t):8:6);
  writeln(arq, 'Y2<',(Y_otimo[2]+t):8:6);
  writeln(arq, 'Y2>',(Y_otimo[2]-t):8:6);
  writeln(arq, 'END');
  writeln(arq, 'GIN 2');
  writeln(arq, 'LEAVE');
  close(arq);

  assign(arq, 'LIN_INI3.BAT');
  rewrite(arq);
  writeln(arq, 'TAKE LIN_ENT3.TXT');
  writeln(arq, 'SAVE LIN_ENT3.SAV');
  writeln(arq, 'QUIT');
  close(arq);
  exec('\dos\command.com','/C lindo < LIN_INI3.BAT');  { chamada do lindo }
  inc(iteracao);


end;

{-------------------------------------------------------------------}

Procedure Ler_Saida_Lindo;
var
   arq      : text;
   linha    : string;
   letra    : char;
   i,
   j,
   w,
   erro     : integer;
   aux      : real;

begin
  assign(arq, 'LIN_OUT3.TXT');
  reset(arq);
  repeat
    readln(arq, linha);
  until linha <> '';      {Procurar a primeira linha da saida que nao seja
                           solucao}
```

```
  i:=1;
  repeat
    letra:= linha[i];      {ler a primeira letra da primeira linha}
    inc(i);
  until letra <> ' ';

  if letra = 'W' then
    begin
      repeat
        readln(arq, linha);
      until linha <> '';
    end;

  repeat
    readln(arq, linha);
  until linha <> '';       {Procurar o valor otima da funcao objetivo}

  j:= ord(linha[0])-4;
  i:=j;
  repeat
    letra:= linha[i];
    dec(i);
  until letra = ' ';
  val(copy(linha, i+1, j-i), Z, erro);

  w:=3;
  repeat                   {Procurar as solucoes das variaveis inteiras}
    readln(arq,linha);
    dec(w);
  until w = 0;

  for w:=1 to 2  do
    begin
      i:=j+3;
      repeat
        letra:= linha[i];
        dec(i);
      until letra = ' ';
      val(copy(linha, i+1, j-i), aux , erro);
      Y[w]:= round(aux);
      readln(arq, linha);
    end;
 close(arq);
end;

{-----------------------------------------------------------------}

Procedure Atualiza_Epslon;
var
  temp     : real;
begin
  temp:= suminf(y);
  if temp > epslon_f then
```

```
    epslon:= temp
  else
    epslon:= epslon_f;

end;

{-----------------------------------------------------------------}

Procedure Teste_de_Parada;
var
  dif      : array[1..2] of real;
  i        : integer;
  menor    : boolean;
begin

 for i:=1 to 2 do
   begin
     dif[i]:=Y[i]-Y_otimo[i];
   end;


 i:=1;
 menor:= true;
 repeat
   if abs(dif[i]) < delta then
     begin
       menor:= true;
       inc(i);
     end
   else
     menor:= false;
 until (i = 3) or ( menor = false );

 if (i = 3) and ( menor = true) then
    begin
      teste1:=false;
      teste2:=false;
      pare:= true;
    end
 else
  pare:= false;
end;

{-----------------------------------------------------------------}

Procedure Teste_t;
begin
 if abs(t) <  delta then
   begin
     writeln('Erro : Este problema nao esta convergindo.');
     writeln('E a solucao aproximada ate o presente momento e:');
     teste1:= false;
     teste2:= false;
```

```
   end;
end;

{-------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
  hora,
  minuto,
  segundo,
  seg100    : word;

begin
  gettime(hora, minuto, segundo, seg100);
  writeln( 'O Valor da Funcao Otima : ', Z_otimo:8:4);
  writeln;
  writeln;
  writeln( 'As variaveis inteiras otimas sao: ');
  writeln;
  writeln( 'y1      : ', Y_otimo[1]:3);
  writeln( 'y2      : ', Y_otimo[2]:3);
  writeln;
  writeln;
  writeln;
  writeln( 'Numero de iteracoes : ', iteracao :3);
  writeln;
  writeln( 'Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
           hora,':', minuto,':', segundo,'.', seg100);

end;

{-------------------------------------------------------------------}

{*-------------------------------------------------------------------
                    Programa Principal
-------------------------------------------------------------------*}

begin
  Inicializacao;
  Cria_Arquivo_Bat_Lindo;

  for k:=1 to 2 do
    Y_otimo[k]:= Y[k];              {Variaveis otimas iniciais}


  Z_otimo:=fx(Y_otimo);      {Valor da Z_otima inicial}

  repeat

    if suminf(Y_otimo) > epslon then
       soma_maior:= true
    else
       soma_maior:= false;
```

```
repeat

  Prepare_Modelo_Linear;          { RMDLP }
  exec( '\DOS\COMMAND.COM','/C lindo < Lindo3.bat'); { chamada do lindo }
  inc(iteracao);
  Ler_Saida_Lindo;
  Teste_De_Parada;

  while not pare do
   begin
    if soma_maior=true then
     begin
      if suminf(y) < suminf(y_otimo) then
         begin
          Z_otimo:=fx(y);
          for k:=1 to 2 do
           Y_otimo[k]:=Y[k];
          Atualiza_Epslon;
          teste1:=true;
          teste2:=false;
         end
        else
         begin
          t:=t/rt;
          teste2:=true;
          Teste_t;
         end;
     end;
    if soma_maior=false then
     begin
      if suminf(y) > epslon then
         begin
             t:=t/rt;
          teste2:=true;
          Teste_t;
         end
        else
         begin
          if fx(y) >= fx(y_otimo) then
             begin
              t:=t/rt;
              teste2:=true;
              Teste_t;
             end
           else
            begin
             Z_otimo:=fx(y);
             for k:= 1 to 2 do
              y_otimo[k]:=y[k];
             Atualiza_Epslon;
             teste2:=false;
             teste1:=true;
```

```
                end;
              end;

          end;
        pare:= true;
      end;

  until teste2 = false;

until teste1 = false;

Mostra_Resultado_Final;
end.
```

Apêndice C

```
{*************************************************************************
*                                                                        *
*  ESTE PROGRAMA  FOI BASEADO NO TERCEIRO ALGORITMO, ONDE                *
*  HOUVE  UMA  MODIFACACAO COMBINANDO O METODO DE                        *
*  APROXIMACAO EXTERNA  DE DURAN & GROSSMAN E APROXIMACAO                *
*  SEQUENCIAL LINEAR  DE LOH &  PAPALAMBROS,COM  OBJETIVO DE             *
*  TENTEAR CORRIGIR OS DEFEITOS QUE CADA UM ANTERIORMENTE                *
*   APRESENTA.O OBJETIVO PRINCIPAL E RESOLVER OS PROBLEMAS NAO           *
*   LINEARES COM VARIAVEIS MISTA INTEIRAS.                               *
*                                                                        *
*   EXEMPLO 1, TIRADO DO "PAPER"  DE DURAN & GROSSMAN                    *
*                                                                        *
*   AUTORA : Hsing Pei Chin ( Setembro de 1994 )                         *
*                                                                        *
*************************************************************************}
{$M $4000,0,$4000}

Uses
  Dos;
Const
  Ydimensao = 3;

Type
  Vetor1 = array[1..Ydimensao] of integer;
  Vetor2 = array[1..3] of real;

var
 X,
 X_otimo    : vetor2;

 YCoef,
 Y_otimo,
 Y          : vetor1;
 Dominio_Y  : array[1..7] of array[1..Ydimensao] of integer ;

 Z_otimo,
 t_otimo,
 t          : real;


 ZConst     : integer;

 PPviavel,
 PLMDviavel,
 teste1,
 teste2,
 passo_serio: boolean;


 ext,                    {No de incrementacoes de restricoes feitas
                          dentro de uma iteracao para atualizar o otimo}
```

```
  k,
  num_cut,                    { numero de cortes feitas}
  iteracao,                   { numero de atualizacoes}
  cont      : integer;        { numero de Y nao viaveis }

  corte     : array [1..8] of string;

  restricao,
  extensao  : array [1..3] of string;

  conjunto  : array [1..8] of integer;

{-------------------------------------------------------------------------}

Function fx(vi:Vetor1;vc:Vetor2):real;
var
  temp1,
  temp2    :real;
begin
 temp1:=5*vi[1]+6*vi[2]+8*vi[3];
 temp2:=10*vc[1]-7*vc[3]-18*ln(vc[2]+1)-19.2*ln(vc[1]-vc[2]+1)+10;
 fx:= temp1+temp2;
end;

{-------------------------------------------------------------------------}


Procedure Inicializacao;
Var
   i  : integer;
begin
 YCoef[1]:= 5;
 YCoef[2]:= 6;
 YCoef[3]:= 8;
 Y[1]:= 1;
 Y[2]:= 0;
 Y[3]:= 1;
 ZConst:= 10;
 Z_otimo:= 100.0;
 t:=100;
 cont:= 0;
 ext:=0;
 iteracao:= 0;
 num_cut:= 0;
 PPviavel:= true;
 PLMDviavel:= true;
 teste1:= true;
 teste2:= true;
 passo_serio:= true;
 for i:= 1 to 8 do
  begin
   corte[i]:='';
   conjunto[i]:=0;
```

```
    end;
  for i:= 1 to 3 do
    begin
     restricao[i]:='';
     extensao[i]:='';
    end;
  settime( 0, 0, 0, 0);
end;

{-------------------------------------------------------------------------}

Procedure Cria_Arquivo_Bat_Gino;
var
  arq : text;

begin
  assign(arq,'GINO1.BAT');
  rewrite(arq);
  writeln(arq,'RETR GINOENT1.TXT');
  writeln(arq,'DIVERT GINOOUT1.TXT');
  writeln(arq,'GO');
  writeln(arq,'QUIT');
  close(arq);

end;

{-------------------------------------------------------------------------}

Procedure Cria_Arquivos_Bat_Lindo;
Var
  arq : text;
begin

  assign(arq,'LIN_INI1.BAT');
  rewrite(arq);
  writeln(arq,'TAKE LIN_ENT1.TXT');
  writeln(arq,'SAVE LIN_ENT1.SAV');
  writeln(arq,'QUIT');
  close(arq);



  assign(arq,'LINDO11.BAT');
  rewrite(arq);
  writeln(arq,'RETR LIN_ENT1.EXT');
  writeln(arq,'PAGE 0');
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT1.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);

  assign(arq,'LINDO12.BAT');
```

```
  rewrite(arq);
  writeln(arq,'RETR LIN_ENT1.SAV');
  writeln(arq,'PAGE 0');
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT1.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);

end;

{-------------------------------------------------------------------------}

Procedure Prepara_Modelo;
var
  arq   : text;
  soma,
  i     : integer;

begin
  soma:= 0;
  assign( arq, 'GINOENT1.TXT' );
  rewrite(arq);
  writeln(arq, 'MODEL');
  for i:=1 to Ydimensao do
    soma:= soma + Y[i] * Ycoef[i];
  soma:= soma + ZConst;
  writeln(arq, 'MIN = 10 * X1 - 18 * LOG(X2 + 1) - 19.2 * LOG(X1 - X2 + 1) - 7 * X3 +
', soma,';');
  writeln(arq, '- .8 * LOG(X2 + 1) - .96 * LOG(X1 - X2 + 1) + .8 * X3 < 0;');
  writeln(arq, '- X1 + X2 < 0;');
  writeln(arq, 'X2 + ', -2*Y[1], ' < 0;');
  writeln(arq, 'X1 - X2 + ', -2*Y[2], ' < 0;');
  writeln(arq, '-LOG(X2 + 1) - 1.2 * LOG(X1 - X2 + 1) + X3 - 1 < 0;');
  writeln(arq,'X1 > 0;');
  writeln(arq,'X1 < 2;');
  writeln(arq,'X2 > 0;');
  writeln(arq,'X2 < 2;');
  writeln(arq,'X3 > 0;');
  writeln(arq,'X3 < 1;');
  writeln(arq,'END');
  close (arq);
end;

{-------------------------------------------------------------------------}

procedure Verifica_Se_PPviavel;
var
  arq   : text;
  linha : string;

begin
  assign(arq, 'GINOOUT1.TXT');
```

```
  reset (arq);
  readln(arq, linha );
  if copy(linha, 20, 10) = 'INFEASIBLE 'then
    PPviavel:= false
  else
    PPviavel:= true;
  close(arq);
end;

{-------------------------------------------------------------------------}

procedure Gera_Novo_Y ;
var
  identico : boolean;
  i,j      : integer;

begin
  repeat
    randomize;
    for i:= 1 to Ydimensao do
      Y[i]:= random(2);
    for i:= 1 to cont do
      begin
        j:= 1;
        repeat
          identico:= true;
          if Dominio_Y[i,j] = Y[j] then
            inc(j)
          else
            identico:= false;
        until (j=Ydimensao) or (identico=false);
        if (j=Ydimensao) and (identico=true) then
          i:= cont+1;
      end;
  until identico=false;
end;

{-------------------------------------------------------------------------}

Procedure Obter_X_na_Saida_do_Gino;
var
  i,j,w,
  erro      : integer;
  arq       : text;
  linha     : string;
  letra     : char;

begin
  assign(arq, 'GINOOUT1.TXT');
  reset(arq);
  for i:=1 to 5 do
    readln(arq, linha);
```

```
  j:= ord(linha[0]);

  for i:=1 to 3 do
    readln(arq, linha);

  for w:=1 to 3 do
    begin
      i:=j;
      letra:= linha[i];
      while letra <> ' ' do
        begin
          dec(i);
          letra:=linha[i];
        end;
      val(copy(linha, i+1, j-i ), X[w], erro);
      readln(arq, linha);
    end;

  close(arq);
end;

{-------------------------------------------------------------------------}

Procedure Int_Cut;
Const
  strY      : array [1..3] of string = ('Y1', 'Y2', 'Y3');

Var
  i         : integer;

begin
  inc(num_cut);
  for i:=1 to 3 do
    begin
      if Y[i] = 1 then
        begin
          if corte[num_cut] = '' then
              corte[num_cut]:= corte[num_cut] + strY[i]
            else
              corte[num_cut]:= corte[num_cut] + '+' + strY[i];
          conjunto[num_cut]:=conjunto[num_cut] + 1;
        end
      else
        corte[num_cut]:= corte[num_cut] + '-' + strY[i];
    end;
  conjunto[num_cut]:=conjunto[num_cut]-1;

end;

{-------------------------------------------------------------------------}

Procedure Teste_de_Parada1;
Var
```

```
  i        : integer;
  gap      : array[1..6] of real;

  soma     : real;

begin
  if (num_cut=8) then
   begin
    teste1:= false;
    teste2:= false;
   end
  else
   if (iteracao <> 0) and (passo_serio = false) then
    begin
     soma:=0;
     for i:=1 to 3 do
        gap[i]:=abs(y[i]-y_otimo[i]);
     for i:=4 to 6 do
        gap[i]:=abs(x[i-3]-x_otimo[i-3]);
     soma:=gap[1];
      for i:= 2 to 6 do
       begin
          if soma < gap[ i ] then
            soma:= gap[ i ] ;
       end;
        if soma < 2.0 then
         begin
          t:=100;
          inc(ext);
         end
        else
         begin
          t:=soma/2;
          passo_serio:= true;
         end;
    end;
end;

{-------------------------------------------------------------------------}

Procedure Teste_de_Parada2(vi:Vetor1;vc:Vetor2);
var
  df       : array[1..6] of real;
  soma     : real;
begin
  soma:=0;

  df[1]:= 5;
  df[2]:= 6;
  df[3]:= 8;
  df[4]:= 10-19.2/(x_otimo[1]-x_otimo[2]+1);
  df[5]:= -18/(x_otimo[2]+1)+19.2/(x_otimo[1]-x_otimo[2]+1);
  df[6]:= -7;
```

```
    soma:= df[1]*(vi[1]-y_otimo[1])+df[2]*(vi[2]-y_otimo[2])+
           df[3]*(vi[3]-y_otimo[3])+df[4]*(vc[1]-x_otimo[1])+
           df[5]*(vc[2]-x_otimo[2])+df[6]*(vc[3]-x_otimo[3]);
    if soma >= -0.0001 then
      begin
        teste1:=false;
        teste2:=false;
      end;

end;

{-------------------------------------------------------------------------}

Procedure Prepare_Modelo_Linear;
const
  strYX : array[1..6] of string = ('Y1','Y2','Y3','X1','X2','X3');

var
   arq      : text;

   df,
   dg1,
   dg2      : array[1..6] of real;

   n        : array[1..3] of real;

   valor,
   f,
   g1,
   g2       : real;

   erro,
   i        : integer;

   sdf,
   sdg1,
   sdg2     : array [1..6] of string;

   sn       : array[1..3] of string;

begin

  for i:= 1 to 3 do
    begin
      sn[i]:='';
      restricao[i]:='';
    end;

  for i:= 1 to 6 do
    begin
      sdf[i]:='';
      sdg1[i]:='';
```

```
    sdg2[i]:='';
  end;

 df[1]:= 5;

 df[2]:= 6;

 df[3]:= 8;

 df[4]:=10-19.2/(x_otimo[1]-x_otimo[2]+1);

 df[5]:= -18/(x_otimo[2]+1)+19.2/(x_otimo[1]-x_otimo[2]+1);

 df[6]:= -7;

 dg1[1]:= 0;

 dg1[2]:= 0;

 dg1[3]:= 0;

 dg1[4]:= -0.96/(x_otimo[1]-x_otimo[2]+1);

 dg1[5]:= -0.8/(x_otimo[2]+1)+0.96/(x_otimo[1]-x_otimo[2]+1);

 dg1[6]:= 0.8;

 dg2[1]:= 0;

 dg2[2]:= 0;

 dg2[3]:= 2;

 dg2[4]:= -1.2/(x_otimo[1]-x_otimo[2]+1);

 dg2[5]:= -1/(x_otimo[2]+1)+1.2/(x_otimo[1]-x_otimo[2]+1);

 dg2[6]:= 1;

 f:=5*y_otimo[1]+6*y_otimo[2]+8*y_otimo[3]+10*x_otimo[1]-7*x_otimo[3]+
   -18*ln(x_otimo[2]+1)-19.2*ln(x_otimo[1]-x_otimo[2]+1)+10;

 g1:= -0.8*ln(x_otimo[2]+1)-0.96*ln(x_otimo[1]-x_otimo[2]+1)+0.8*x_otimo[3];

 g2:= -ln(x_otimo[2]+1)-1.2*ln(x_otimo[1]-x_otimo[2]+1)+x_otimo[3]
     +2*y_otimo[3]-2;

 n[1]:= f-(df[1]*y_otimo[1]+df[2]*y_otimo[2]+df[3]*y_otimo[3]
       +df[4]*x_otimo[1]+df[5]*x_otimo[2]+df[6]*x_otimo[3]);

 n[2]:= g1-(dg1[1]*y_otimo[1]+dg1[2]*y_otimo[2]+dg1[3]*y_otimo[3]
       +dg1[4]*x_otimo[1]+dg1[5]*x_otimo[2]+dg1[6]*x_otimo[3]);
```

```
n[3]:= g2-(dg2[1]*y_otimo[1]+dg2[2]*y_otimo[2]+dg2[3]*y_otimo[3]
     +dg2[4]*x_otimo[1]+dg2[5]*x_otimo[2]+dg2[6]*x_otimo[3]);

if df[1] <> 0 then
 begin
  str( df[1]:8:6 , sdf[1] );
  restricao[1]:= sdf[1] + strYX[1];
 end;

if dg1[1] <> 0 then
 begin
  str( dg1[1]:8:6 , sdg1[1] );
  restricao[2]:= sdg1[1] + strYX[1];
 end;

if dg2[1] <> 0 then
 begin
  str( dg2[1]:8:6 , sdg2[1] );
  restricao[3]:= sdg2[1] + strYX[1];
 end;


for i:=2 to 6 do
 begin
  if df[i] <> 0 then
   begin
    str( df[i]:8:6, sdf[i] );
    if df[i] > 0 then
     if restricao[1] <> '' then
      restricao[1]:= restricao[1] + '+' + sdf[i] + strYX[i]
     else
      restricao[1]:= restricao[1] + sdf[i] + strYX[i]
    else
     restricao[1]:= restricao[1] + sdf[i] + strYX[i]
   end;

  if dg1[i] <> 0 then
   begin
    str( dg1[i]:8:6, sdg1[i] );
    if dg1[i] > 0 then
     if restricao[2] <> '' then
      restricao[2]:= restricao[2] + '+' + sdg1[i] + strYX[i]
     else
      restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
    else
     restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
   end;

  if dg2[i] <> 0 then
   begin
    str( dg2[i]:8:6, sdg2[i] );
    if dg2[i] > 0 then
     if restricao[3] <> '' then
```

```
          restricao[3]:= restricao[3] + '+' + sdg2[i] + strYX[i]
        else
          restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
      else
        restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
    end;
  end;

for i:= 1 to 3 do
  begin
    valor:=0;
    if n[i] <> 0 then
      begin
        if n[i] > 0 then
          begin
            str(n[i]:8:6, sn[i]);
            sn[i]:= '-' + sn[i];
          end
        else
         begin
           str( n[i]:8:6, sn[i]);
           val(sn[i],valor,erro);
           valor:=(-1)*valor;
           str(valor:8:6,sn[i]);
         end;
      end
    else
     sn[i]:= ' 0 ';
  end;

assign(arq, 'LIN_ENT1.TXT');
rewrite(arq);
writeln(arq, 'MIN  R ');
writeln(arq, 'ST');
writeln(arq, 'Y1 > ',(-t+Y_otimo[1]):8:6);
writeln(arq, 'Y1 < ',(t+Y_otimo[1]):8:6);
writeln(arq, 'Y2 > ',(-t+Y_otimo[2]):8:6);
writeln(arq, 'Y2 < ',(t+Y_otimo[2]):8:6);
writeln(arq, 'Y3 > ',(-t+Y_otimo[3]):8:6);
writeln(arq, 'Y3 < ',(t+Y_otimo[3]):8:6);
writeln(arq, 'X1 > ',(-t+X_otimo[1]):8:6);
writeln(arq, 'X1 < ',(t+X_otimo[1]):8:6);
writeln(arq, 'X2 > ',(-t+X_otimo[2]):8:6);
writeln(arq, 'X2 < ',(t+X_otimo[2]):8:6);
writeln(arq, 'X3 > ',(-t+X_otimo[3]):8:6);
writeln(arq, 'X3 < ',(t+X_otimo[3]):8:6);
writeln(arq, 'X2 - X1 < 0');
writeln(arq, 'X2 - 2Y1 < 0');
writeln(arq, 'X1 - X2 - 2Y2 < 0');
writeln(arq, 'Y1 + Y2 < 1');
writeln(arq, 'X1 > 0');
writeln(arq, 'X1 < 2');
writeln(arq, 'X2 > 0');
```

```
  writeln(arq, 'X2 < 2');
  writeln(arq, 'X3 > 0');
  writeln(arq, 'X3 < 1');
  writeln(arq,copy(restricao[1],1,70));
  writeln(arq, copy(restricao[1],71,256) +  ' -R < ',sn[1]);
  writeln(arq,copy(restricao[2],1,70));
  writeln(arq, copy(restricao[2],71,256) + ' < ', sn[2]);
  writeln(arq,copy(restricao[3],1,70));
  writeln(arq, copy(restricao[3],71,256) + ' <  ',sn[3]);
  for i:=1 to num_cut do
     writeln(arq, corte[i] + ' < ',conjunto[i]);
  writeln(arq, 'END');
  writeln(arq, 'INTEGER Y1');
  writeln(arq, 'INTEGER Y2');
  writeln(arq, 'INTEGER Y3');
  writeln(arq, 'LEAVE');
  close(arq);
  swapvectors;
  exec ('\dos\command.com','/C lindo < LIN_INI1.bat');
  swapvectors;
  inc(iteracao);

end;

{-------------------------------------------------------------------------}

Procedure Prepare_Extensao;
const
    strYX : array[1..6] of string = ('Y1','Y2','Y3','X1','X2','X3');
var
  f, g1, g2 , valor   : real;
  df,dg1,dg2          : array [1..6] of real;
  sdf, sdg1, sdg2     : array [1..6] of string[20];
  n                   : array [1..3] of real;
  sn,
  extensao            : array [1..3] of string;
  i, erro             : integer;
  arq                 : text;

begin
  for i:= 1 to 3 do
    begin
      sn[i]:='';
      extensao[i]:='';
    end;

  for i:= 1 to 6 do
    begin
      sdf[i]:='';
      sdg1[i]:='';
      sdg2[i]:='';
    end;
```

```
df[1]:= 5;

df[2]:= 6;

df[3]:= 8;

df[4]:=10-19.2/(x[1]-x[2]+1);

df[5]:= -18/(x[2]+1)+19.2/(x[1]-x[2]+1);

df[6]:= -7;

dg1[1]:= 0;

dg1[2]:= 0;

dg1[3]:= 0;

dg1[4]:= -0.96/(x[1]-x[2]+1);

dg1[5]:= -0.8/(x[2]+1)+0.96/(x[1]-x[2]+1);

dg1[6]:= 0.8;

dg2[1]:= 0;

dg2[2]:= 0;

dg2[3]:= 2;

dg2[4]:= -1.2/(x[1]-x[2]+1);

dg2[5]:= -1/(x[2]+1)+1.2/(x[1]-x[2]+1);

dg2[6]:= 1;

f:=5*y[1]+6*y[2]+8*y[3]+10*x[1]-7*x[3]-18*ln(x[2]+1)
  -19.2*ln(x[1]-x[2]+1)+10;

g1:= -0.8*ln(x[2]+1)-0.96*ln(x[1]-x[2]+1)+0.8*x[3];

g2:= -ln(x[2]+1)-1.2*ln(x[1]-x[2]+1)+x[3]
    +2*y[3]-2;

n[1]:= f-(df[1]*y[1]+df[2]*y[2]+df[3]*y[3]
      +df[4]*x[1]+df[5]*x[2]+df[6]*x[3]);

n[2]:= g1-(dg1[1]*y[1]+dg1[2]*y[2]+dg1[3]*y[3]
      +dg1[4]*x[1]+dg1[5]*x[2]+dg1[6]*x[3]);

n[3]:= g2-(dg2[1]*y[1]+dg2[2]*y[2]+dg2[3]*y[3]
      +dg2[4]*x[1]+dg2[5]*x[2]+dg2[6]*x[3]);
```

```
if df[1] <> 0 then
  begin
    str( df[1]:8:6 , sdf[1] );
    restricao[1]:= sdf[1] + strYX[1];
  end;

if dg1[1] <> 0 then
  begin
    str( dg1[1]:8:6 , sdg1[1] );
    restricao[2]:= sdg1[1] + strYX[1];
  end;

if dg2[1] <> 0 then
  begin
    str( dg2[1]:8:6 , sdg2[1] );
    restricao[3]:= sdg2[1] + strYX[1];
  end;


for i:=2 to 6 do
  begin
    if df[i] <> 0 then
      begin
        str( df[i]:8:6, sdf[i] );
        if df[i] > 0 then
          if extensao[1] <> '' then
            extensao[1]:= extensao[1] + '+' + sdf[i] + strYX[i]
          else
            extensao[1]:= extensao[1] + sdf[i] + strYX[i]
        else
          extensao[1]:= extensao[1] + sdf[i] + strYX[i]
      end;

    if dg1[i] <> 0 then
      begin
        str( dg1[i]:8:6, sdg1[i] );
        if dg1[i] > 0 then
          if extensao[2] <> '' then
            extensao[2]:= extensao[2] + '+' + sdg1[i] + strYX[i]
          else
            extensao[2]:= extensao[2] + sdg1[i] + strYX[i]
        else
          restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
      end;

    if dg2[i] <> 0 then
      begin
        str( dg2[i]:8:6, sdg2[i] );
        if dg2[i] > 0 then
          if extensao[3] <> '' then
            extensao[3]:= extensao[3] + '+' + sdg2[i] + strYX[i]
          else
```

```
          extensao[3]:= extensao[3] + sdg2[i] + strYX[i]
        else
          extensao[3]:= extensao[3] + sdg2[i] + strYX[i]
      end;
  end;
for i:= 1 to 3 do
  begin
    valor:=0;
    if n[i] <> 0 then
      begin
        if n[i] > 0 then
          begin
            str(n[i]:8:6, sn[i]);
            sn[i]:= '-' + sn[i];
          end
        else
         begin
           str( n[i]:8:6, sn[i]);
           val(sn[i],valor,erro);
           valor:=(-1)*valor;
           str(valor:8:6,sn[i]);
         end;
      end
    else
     sn[i]:= ' 0 ';
  end;

assign(arq,'LIN_EXT0.BAT');
rewrite(arq);
writeln(arq,'RETR LIN_ENT1.SAV');
writeln(arq,'EXTEND');
writeln(arq,copy(extensao[1],1,70));
writeln(arq, copy(extensao[1],71,70) + ' -R < ', sn[1]);
writeln(arq,copy(extensao[2],1,70));
writeln(arq, copy(extensao[2],71,256) + ' < ', sn[2]);
writeln(arq,copy(extensao[3],1,70));
writeln(arq, copy(extensao[3],71,256) + ' < ', sn[3]);
for i:= 1 to NUM_CUT do
    writeln(arq,corte[i] + '<', conjunto[i]);
writeln(arq,'END');
writeln(arq,'SAVE LIN_ENT1.EXT');
writeln(arq,'QUIT');
close(arq);

assign(arq,'LIN_EXTN.BAT');
rewrite(arq);
writeln(arq,'RETR LIN_ENT1.EXT');
writeln(arq,'EXTEND');
writeln(arq,copy(extensao[1],1,70));
writeln(arq, copy(extensao[1],71,70) + ' -R < ', sn[1]);
writeln(arq,copy(extensao[2],1,70));
writeln(arq, copy(extensao[2],71,256) + ' < ', sn[2]);
writeln(arq,copy(extensao[3],1,70));
```

```
    writeln(arq, copy(extensao[3],71,256) + ' < ', sn[3]);
    for i:= 1 to NUM_CUT do
          writeln(arq,corte[i] + '<', conjunto[i]);
    writeln(arq,'END');
    writeln(arq,'SAVE LIN_ENT1.EXT');
    writeln(arq,'QUIT');
    close(arq);

   if (ext = 1) then
     begin
       swapVectors;
       exec( '\dos\command.com','/C lindo < lin_ext0.bat');
       swapVectors;
       inc(iteracao);
     end
   else
     begin
       swapVectors;
       exec( '\dos\command.com','/C lindo < lin_extn.bat');
       swapVectors;
       inc(iteracao);
     end;
end;
{-------------------------------------------------------------------------}
Procedure Verifica_PLMD;
var
  arq    : text;
  linha  : string;
  letra  : char;
  i,j,w,
  erro   : integer;
  aux    : real;

begin
  PLMDviavel:= true;
  assign(arq, 'LIN_OUT1.TXT');
  reset(arq);
  repeat
    readln(arq, linha);
  until linha <> '';
  i:= 1;
  repeat
    letra:= linha[i];
    inc(i);
  until letra <> ' ';
  if letra = 'W' then
    begin
      PLMDviavel:= false;
      close(arq);
      teste1:=false;
      teste2:=false;
    end
  else
```

```
    begin
      repeat
        readln(arq, linha);
      until linha <> '';


      j:= ord(linha[0]) - 4 ;

      w:= 3;
      repeat
        readln(arq, linha);
        dec(w);
      until w = 0;

      for w:= 1 to 3 do  { sao 3 variaveis inteiras }
        begin
          i:= j + 3;
          repeat
            letra:= linha[i];
            dec(i);
          until letra = ' ';
          val(copy(linha, i+1, j-i ), aux, erro);
          Y[w]:= round(aux);
          readln(arq, linha);
        end;

      readln(arq, linha);

      for w:= 1 to 3 do  { sao 3 variaveis continuas }
        begin
          i:= j + 3;
          repeat
            letra:= linha[i];
            dec(i);
          until letra = ' ';
          val(copy(linha, i+1, j-i ), X[w], erro);
          readln(arq, linha);
        end;

      close (arq);
    end;
end;


{-------------------------------------------------------------------------}

Procedure Atualiza_Otimo;
Var
  i   : integer;
begin
  for i:=1 to 3 do
    begin
      x_otimo[i]:= x[i];
```

```
      y_otimo[i]:= y[i];
    end;
  z_otimo:= fx(y,x);
  t_otimo:= t;
  t:=100.0;
end;

{-------------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
  hora,
  minuto,
  segundo,
  seg100   : word;

begin
  gettime(hora, minuto, segundo, seg100);
  writeln('O Valor da Funcao Otima : ', Z_otimo:8:6);
  writeln;
  writeln('As variaveis inteiras otimas sao : ');
  writeln;
  writeln('y1      : ', Y_otimo[1]:3);
  writeln('y2      : ', Y_otimo[2]:3);
  writeln('y3      : ', Y_otimo[3]:3);
  writeln;
  writeln('As variaveis continuas otimas sao :');
  writeln;
  writeln('x1      : ', X_otimo[1]:10:6);
  writeln('x2      : ', X_otimo[2]:10:6);
  writeln('x3      : ', X_otimo[3]:10:6);
  writeln;
  writeln('O tamanho da regiao de confianca e:', t_otimo:8:6);
  writeln;
  writeln('Numero de iteracoes : ', iteracao :3);
  writeln;
  writeln('Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
          hora,':', minuto,':', segundo,'.', seg100);
end;


(*-------------------------------------------------------------------------
                  PROGRAMA PRINCIPAL
-------------------------------------------------------------------------*)

{ 1a PARTE DO PROGRAMA: VERIFICA A VIABILIDADE DA VARIAVEL
INTEIRA Y INICIAL, CASO NAO FOR VIAVEL,TENTA ACHAR UMA OUTRA
COMBINACAO DE Y TAL QUE SATISFACA A CONDICAO E ELIMINAR A Y
INVIAVEL DO DOMINIO.                                    }


begin
```

```
Inicializacao;
Cria_arquivo_bat_gino;
Cria_arquivos_bat_lindo;

repeat
  Prepara_Modelo;
  swapVectors;
  exec ('\dos\command.com','/C gino < Gino1.bat');   { chamada de gino}
  swapVectors;
  Verifica_Se_PPviavel;
  If PPviavel = false then
   begin
    inc(cont);
    for k:= 1 to Ydimensao do
      Dominio_Y[cont,k]:= Y[k];
      Gera_Novo_Y;
    end;
  until ppviavel=true;



Obter_X_na_Saida_do_Gino;
Atualiza_otimo;
Int_Cut;

{2a PARTE DO PROGRAMA: RESOLVER PROBLEMA MASTER LINEAR MISTA
PELO LINDO   }


repeat
  Teste_De_Parada1;

  if teste2 = true then
   begin
    if Passo_Serio = true then
      Prepara_Modelo_Linear;
    if Passo_Serio = false then
      begin
        Prepare_Extensao;
        swapVectors;
        exec( '\dos\command.com','/C lindo < lindo11.bat');
        swapVectors;
        inc(iteracao);
        Verifica_PLMD;
      end
    else
      begin
         swapVectors;
         exec( '\dos\command.com','/C lindo < lindo12.bat');
         swapVectors;
         inc(iteracao);
         Verifica_PLMD;
```

```
      end;
    Int_Cut;
    { Teste_de_parada2(y,x);}

    while PLMDviavel do
      begin
        Prepara_Modelo;
        swapVectors;
        exec( '\dos\command.com','/C gino < Gino1.bat');
        swapVectors;
        Verifica_Se_PPviavel;
        if PPviavel = true then
          begin
            Obter_X_na_Saida_do_Gino;
            Teste_de_Parada2(y,x);
            if fx(y,x) < fx(y_otimo,x_otimo) then
              begin
                Atualiza_otimo;
                passo_serio:= true;
                ext:=0;   {numero de vezes da extensao ja feita}
                PLMDviavel:=false;
              end
            else
              begin
                passo_serio:= false;
                PLMDviavel:= false;
              end;
          end
        else
          begin
            passo_serio:= true;
            PLMDviavel:= false;
          end;
      end;
  end;

until teste1 = false;

Mostra_Resultado_Final;
end.
```

```
{*************************************************************************
*                                                                        *
*   ESTE PROGRAMA  FOI BASEADO NO TERCEIRO ALGORITMO, ONDE               *
*   HOUVE  UMA  MODIFACACAO COMBINANDO O METODO DE                       *
*   APROXIMACAO EXTERNA DE DURAN & GROSSMAN E APROXIMACAO                *
*   SEQUENCIAL LINEAR  DE LOH & PAPALAMBROS,COM  OBJETIVO DE             *
*   TENTEAR CORRIGIR OS DEFEITOS QUE CADA UM ANTERIORMENTE               *
*   APRESENTA.O OBJETIVO PRINCIPAL E RESOLVER OS PROBLEMAS NAO           *
*   LINEARES COM VARIAVEIS MISTA INTEIRAS.                               *
*                                                                        *
*   EXEMPLO 3, TIRADO DO "PAPER"  DE LOH & PAPALAMBROS                   *
*                                                                        *
*   AUTORA : Hsing Pei Chin ( Outubro de 1994 )                          *
*                                                                        *
*                                                                        *
*************************************************************************}


{$M $4000,0,$4000}

Uses
  Dos;
Const
  Ydimensao = 2;

Type
  Vetor1 = array[1..Ydimensao] of integer;

var

 Y_otimo,
 Y        : vetor1;

 Z_otimo,
 t_otimo,
 t        : real;

 PLMDviavel,
 teste1,
 teste2,
 passo_serio: boolean;


 ext,                    {No de incrementacoes de restricoes feitas
                         dentro de uma iteracao para atualizar o otimo}
 k,
 iteracao  : integer;                  { numero de atualizacoes}


 restricao,
 extensao   : array [1..3] of string;
```

```
{-------------------------------------------------------------------------}

Function fx(vi:Vetor1):real;
var
  temp1    :real;
begin
 temp1:=-9*vi[1]*vi[1]+10*vi[1]*vi[2]-50*vi[1]+8*vi[2]+460;;
 fx:= temp1;
end;

{-------------------------------------------------------------------------}


Procedure Inicializacao;
Var
   i  : integer;
begin
 Y[1]:= 4;
 Y[2]:= 3;
 Y_otimo[1]:=Y[1];
 Y_otimo[2]:=Y[2];
 Z_otimo:= fx(y_otimo);
 t:=100;
 ext:=0;
 iteracao:= 0;
 PLMDviavel:= true;
 teste1:= true;
 teste2:= true;
 passo_serio:= true;
 for i:= 1 to 3 do
   begin
    restricao[i]:='';
    extensao[i]:='';
   end;
 settime( 0, 0, 0, 0);
end;

{-------------------------------------------------------------------------}

Procedure Cria_Arquivos_Bat_Lindo;
Var
 arq : text;
begin

 assign(arq,'LIN_INI3.BAT');
 rewrite(arq);
 writeln(arq,'TAKE LIN_ENT3.TXT');
 writeln(arq,'SAVE LIN_ENT3.SAV');
 writeln(arq,'QUIT');
 close(arq);
```

```
  assign(arq,'LINDO31.BAT');
  rewrite(arq);
  writeln(arq,'RETR LIN_ENT3.EXT');
  writeln(arq,'PAGE 0');
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT3.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);

  assign(arq,'LINDO32.BAT');
  rewrite(arq);
  writeln(arq,'RETR LIN_ENT3.SAV');
  writeln(arq,'PAGE 0');
  writeln(arq,'GO');
  writeln(arq,'DIVERT LIN_OUT3.TXT');
  writeln(arq,'SOLUTION');
  writeln(arq,'QUIT');
  close(arq);

end;

{-------------------------------------------------------------------------}

Procedure Teste_de_Parada1;
Var
  i        : integer;
  gap      : array[1..2] of real;

  soma     : real;

begin
   if (iteracao <> 0) and (passo_serio = false) then
     begin
       soma:=0;
       for i:=1 to 2 do
          gap[i]:=abs(y[i]-y_otimo[i]);
       if gap[1] < gap [2]  then
          soma:= gap [2]
        else soma:= gap [1];
          if soma < 2.0 then
            begin
              t:=100;
              inc(ext);
            end
          else
            begin
              t:=soma/2;
              passo_serio:= true;
            end;
     end;
end;
```

```
{-------------------------------------------------------------------------}

Procedure Teste_de_Parada2(vi:Vetor1);
var
  df       : array[1..2] of real;
  soma     : real;
begin
   soma:=0;

  df[1]:= -18*y_otimo[1]+10*y_otimo[2]-50;
  df[2]:= 10*y_otimo[1]+8;

   soma:= df[1]*(vi[1]-y_otimo[1])+df[2]*(vi[2]-y_otimo[2]);

   if soma >= -0.0001 then
    begin
      teste1:=false;
      teste2:=false;
    end;

end;

{-------------------------------------------------------------------------}

Procedure Prepare_Modelo_Linear;
const
 strYX : array[1..2] of string = ('Y1','Y2');

var
  arq      : text;

  df,
  dg1,
  dg2      : array[1..2] of real;

  n        : array[1..3] of real;

  valor,
  f,
  g1,
  g2       : real;

  erro,
  i        : integer;

  sdf,
  sdg1,
  sdg2     : array [1..2] of string;

  sn       : array[1..3] of string;

begin
```

```
for i:= 1 to 3 do
  begin
    sn[i]:='';
    restricao[i]:='';
  end;

for i:= 1 to 2 do
  begin
    sdf[i]:='';
    sdg1[i]:='';
    sdg2[i]:='';
  end;

df[1]:= -18*y_otimo[1]+10*y_otimo[2]-50;

df[2]:= 10*y_otimo[1]+8;

dg1[1]:= 1;

dg1[2]:= -(2*0.2768*y_otimo[2]-0.235);

dg2[1]:= 1;

dg2[2]:= -3*(-0.019*y_otimo[2]*y_otimo[2]+2*0.446*y_otimo[2]-3.98);

f:=-9*y_otimo[1]*y_otimo[1]+10*y_otimo[1]*y_otimo[2]-50*y_otimo[1]+
   8*y_otimo[2]+460;

g1:=y_otimo[1]-(0.2768*y_otimo[2]*y_otimo[2]-0.235*y_otimo[2]+3.718);

g2:=y_otimo[1]-(-0.019*y_otimo[2]*y_otimo[2]*y_otimo[2]+0.446*y_otimo[2]
    *y_otimo[2]-3.98*y_otimo[2]+15.854);

n[1]:= f-(df[1]*y_otimo[1]+df[2]*y_otimo[2]);

n[2]:= g1-(dg1[1]*y_otimo[1]+dg1[2]*y_otimo[2]);

n[3]:= g2-(dg2[1]*y_otimo[1]+dg2[2]*y_otimo[2]);

if df[1] <> 0 then
  begin
    str( df[1]:8:6 , sdf[1] );
    restricao[1]:= sdf[1] + strYX[1];
  end;

if dg1[1] <> 0 then
  begin
    str( dg1[1]:8:6 , sdg1[1] );
    restricao[2]:= sdg1[1] + strYX[1];
  end;

if dg2[1] <> 0 then
  begin
```

```
    str( dg2[1]:8:6 , sdg2[1] );
    restricao[3]:= sdg2[1] + strYX[1];
   end;


 for i:=2 to 2 do
   begin
    if df[i] <> 0 then
     begin
      str( df[i]:8:6, sdf[i] );
      if df[i] > 0 then
       if restricao[1] <> '' then
        restricao[1]:= restricao[1] + '+' + sdf[i] + strYX[i]
       else
        restricao[1]:= restricao[1] + sdf[i] + strYX[i]
      else
       restricao[1]:= restricao[1] + sdf[i] + strYX[i]
     end;

    if dg1[i] <> 0 then
     begin
      str( dg1[i]:8:6, sdg1[i] );
      if dg1[i] > 0 then
       if restricao[2] <> '' then
        restricao[2]:= restricao[2] + '+' + sdg1[i] + strYX[i]
       else
        restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
      else
       restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
     end;

    if dg2[i] <> 0 then
     begin
      str( dg2[i]:8:6, sdg2[i] );
      if dg2[i] > 0 then
       if restricao[3] <> '' then
        restricao[3]:= restricao[3] + '+' + sdg2[i] + strYX[i]
       else
        restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
      else
       restricao[3]:= restricao[3] + sdg2[i] + strYX[i]
     end;
   end;

 for i:= 1 to 3 do
   begin
    valor:=0;
    if n[i] <> 0 then
     begin
      if n[i] > 0 then
       begin
        str(n[i]:8:6, sn[i]);
        sn[i]:= '-' + sn[i];
```

```
          end
        else
         begin
          str( n[i]:8:6, sn[i]);
          val(sn[i],valor,erro);
          valor:=(-1)*valor;
          str(valor:8:6,sn[i]);
         end;
      end
    else
     sn[i]:= ' 0 ';
  end;

  assign(arq, 'LIN_ENT3.TXT');
  rewrite(arq);
  writeln(arq, 'MIN  R ');
  writeln(arq, 'ST');
  writeln(arq, 'Y1 > ',(-t+Y_otimo[1]):8:6);
  writeln(arq, 'Y1 < ',(t+Y_otimo[1]):8:6);
  writeln(arq, 'Y2 > ',(-t+Y_otimo[2]):8:6);
  writeln(arq, 'Y2 < ',(t+Y_otimo[2]):8:6);
  writeln(arq,copy(restricao[1],1,70));
  writeln(arq, copy(restricao[1],71,256) +  ' -R < ',sn[1]);
  writeln(arq,copy(restricao[2],1,70));
  writeln(arq, copy(restricao[2],71,256) + ' < ', sn[2]);
  writeln(arq,copy(restricao[3],1,70));
  writeln(arq, copy(restricao[3],71,256) + ' <  ',sn[3]);
  writeln(arq, 'END');
  writeln(arq, 'GIN Y1');
  writeln(arq, 'GIN Y2');
  writeln(arq, 'LEAVE');
  close(arq);
  swapvectors;
  exec ('\dos\command.com','/C lindo < LIN_INI3.bat');
  swapvectors;
  inc(iteracao);

end;

{-------------------------------------------------------------------------}

Procedure Prepare_Extensao;
const
    strYX : array[1..2] of string = ('Y1','Y2');
var
  f, g1, g2 , valor   : real;
  df,dg1,dg2          : array [1..2] of real;
  sdf, sdg1, sdg2     : array [1..2] of string[20];
  n                   : array [1..3] of real;
  sn,
  extensao            : array [1..3] of string;
  i, erro             : integer;
  arq                 : text;
```

```
begin
 for i:= 1 to 3 do
  begin
   sn[i]:='';
   extensao[i]:='';
  end;

 for i:= 1 to 2 do
  begin
   sdf[i]:='';
   sdg1[i]:='';
   sdg2[i]:='';
  end;

 df[1]:= -18*y[1]+10*y[2]-50;

 df[2]:= 10*y[1]+8;

 dg1[1]:= 1;

 dg1[2]:= -(2*0.2768*y[2]-0.235);

 dg2[1]:= 1;

 dg2[2]:= -3*(-0.019*y[2]*y[2]+2*0.446*y[2]-3.98);

 f:=-9*y[1]*y[1]+10*y[1]*y[2]-50*y[1]+8*y[2]+460;

 g1:=y[1]-(0.2768*y[2]*y[2]-0.235*y[2]+3.718);

 g2:=y[1]-(-0.019*y[2]*y[2]*y[2]+0.446*y[2]*y[2]-3.98*y[2]+15.854);

 n[1]:= f-(df[1]*y[1]+df[2]*y[2]);

 n[2]:= g1-(dg1[1]*y[1]+dg1[2]*y[2]);

 n[3]:= g2-(dg2[1]*y[1]+dg2[2]*y[2]);



 if df[1] <> 0 then
  begin
   str( df[1]:8:6 , sdf[1] );
   restricao[1]:= sdf[1] + strYX[1];
  end;

 if dg1[1] <> 0 then
  begin
   str( dg1[1]:8:6 , sdg1[1] );
   restricao[2]:= sdg1[1] + strYX[1];
  end;
```

```
if dg2[1] <> 0 then
  begin
    str( dg2[1]:8:6 , sdg2[1] );
    restricao[3]:= sdg2[1] + strYX[1];
  end;


for i:=2 to 6 do
  begin
    if df[i] <> 0 then
      begin
        str( df[i]:8:6, sdf[i] );
        if df[i] > 0 then
          if extensao[1] <> '' then
            extensao[1]:= extensao[1] + '+' + sdf[i] + strYX[i]
          else
            extensao[1]:= extensao[1] + sdf[i] + strYX[i]
        else
          extensao[1]:= extensao[1] + sdf[i] + strYX[i]
      end;

    if dg1[i] <> 0 then
      begin
        str( dg1[i]:8:6, sdg1[i] );
        if dg1[i] > 0 then
          if extensao[2] <> '' then
            extensao[2]:= extensao[2] + '+' + sdg1[i] + strYX[i]
          else
            extensao[2]:= extensao[2] + sdg1[i] + strYX[i]
        else
          restricao[2]:= restricao[2] + sdg1[i] + strYX[i]
      end;

    if dg2[i] <> 0 then
      begin
        str( dg2[i]:8:6, sdg2[i] );
        if dg2[i] > 0 then
          if extensao[3] <> '' then
            extensao[3]:= extensao[3] + '+' + sdg2[i] + strYX[i]
          else
            extensao[3]:= extensao[3] + sdg2[i] + strYX[i]
        else
          extensao[3]:= extensao[3] + sdg2[i] + strYX[i]
      end;
  end;

for i:= 1 to 3 do
  begin
    valor:=0;
    if n[i] <> 0 then
      begin
        if n[i] > 0 then
          begin
```

```
          str(n[i]:8:6, sn[i]);
          sn[i]:= '-' + sn[i];
         end
        else
         begin
          str( n[i]:8:6, sn[i]);
          val(sn[i],valor,erro);
          valor:=(-1)*valor;
          str(valor:8:6,sn[i]);
         end;
      end
    else
     sn[i]:= ' 0 ';
 end;
assign(arq,'LIN_EXT0.BAT');
rewrite(arq);
writeln(arq,'RETR LIN_ENT3.SAV');
writeln(arq,'EXTEND');
writeln(arq,copy(extensao[1],1,70));
writeln(arq, copy(extensao[1],71,70) + ' -R < ', sn[1]);
writeln(arq,copy(extensao[2],1,70));
writeln(arq, copy(extensao[2],71,256) + ' < ', sn[2]);
writeln(arq,copy(extensao[3],1,70));
writeln(arq, copy(extensao[3],71,256) + ' < ', sn[3]);
for i:= 1 to NUM_CUT do
    writeln(arq,corte[i] + '<', conjunto[i]);
writeln(arq,'END');
writeln(arq,'SAVE LIN_ENT3.EXT');
writeln(arq,'QUIT');
close(arq);
assign(arq,'LIN_EXTN.BAT');
rewrite(arq);
writeln(arq,'RETR LIN_ENT3.EXT');
writeln(arq,'EXTEND');
writeln(arq,copy(extensao[1],1,70));
writeln(arq, copy(extensao[1],71,70) + ' -R < ', sn[1]);
writeln(arq,copy(extensao[2],1,70));
writeln(arq, copy(extensao[2],71,256) + ' < ', sn[2]);
writeln(arq,copy(extensao[3],1,70));
writeln(arq, copy(extensao[3],71,256) + ' < ', sn[3]);
for i:= 1 to NUM_CUT do
    writeln(arq,corte[i] + '<', conjunto[i]);
writeln(arq,'END');
writeln(arq,'SAVE LIN_ENT3.EXT');
writeln(arq,'QUIT');
close(arq);
if (ext = 1) then
 begin
  swapVectors;
  exec( '\dos\command.com','/C lindo < lin_ext0.bat');
  swapVectors;
  inc(iteracao);
 end
```

```
  else
    begin
      swapVectors;
      exec( '\dos\command.com','/C lindo < lin_extn.bat');
      swapVectors;
      inc(iteracao);
    end;
end;

{-------------------------------------------------------------------------}

Procedure Verifica_PLMD;
var
  arq    : text;
  linha  : string;
  letra  : char;
  i,j,w,
  erro   : integer;
  aux    : real;

begin
  PLMDviavel:= true;
  assign(arq, 'LIN_OUT3.TXT');
  reset(arq);
  repeat
    readln(arq, linha);
  until linha <> '';
  i:= 1;
  repeat
    letra:= linha[i];
    inc(i);
  until letra <> ' ';
  if letra = 'W' then
    begin
      PLMDviavel:= false;
      close(arq);
      teste1:=false;
      teste2:=false;
    end
  else
    begin
      repeat
        readln(arq, linha);
      until linha <> '';


      j:= ord(linha[0]) - 4 ;

      w:= 3;
      repeat
        readln(arq, linha);
        dec(w);
      until w = 0;
```

```
    for w:= 1 to 2 do  { sao 2 variaveis inteiras }
      begin
        i:= j + 3;
        repeat
          letra:= linha[i];
          dec(i);
        until letra = ' ';
        val(copy(linha, i+1, j-i ), aux, erro);
        Y[w]:= round(aux);
        readln(arq, linha);
      end;

    close (arq);
  end;
end;


{-------------------------------------------------------------------------}

Procedure Atualiza_Otimo;
Var
  i  : integer;
begin
 for i:=1 to 2 do
   begin
     y_otimo[i]:= y[i];
   end;
 z_otimo:= fx(y);
 t_otimo:= t;
 t:=100.0;
end;

{-------------------------------------------------------------------------}

Procedure Mostra_Resultado_Final;
var
 hora,
 minuto,
 segundo,
 seg100   : word;

begin
 gettime(hora, minuto, segundo, seg100);
 writeln('O Valor da Funcao Otima : ', Z_otimo:8:6);
 writeln;
 writeln('As variaveis inteiras otimas sao : ');
 writeln;
 writeln('y1     : ', Y_otimo[1]:3);
 writeln('y2     : ', Y_otimo[2]:3);
 writeln;
 writeln;
 writeln('O tamanho da regiao de confianca e:', t_otimo:8:6);
```

```
  writeln;
  writeln('Numero de iteracoes : ', iteracao :3);
  writeln;
  writeln('Tempo gasto para execucao do problema (hh:min:seg.seg100) : ',
          hora,':', minuto,':', segundo,'.', seg100);
end;


(*-------------------------------------------------------------------------
                    PROGRAMA PRINCIPAL
-------------------------------------------------------------------------*)

begin

  Inicializacao;

  Cria_Arquivos_bat_lindo;

  repeat
    Teste_De_Parada1;

    if teste2 = true then
      begin
        if Passo_Serio = true then
           Prepare_Modelo_Linear;
        if Passo_Serio = false then
          begin
            Prepare_Extensao;
            swapVectors;
            exec( '\dos\command.com','/C lindo < lindo31.bat');
            swapVectors;
            inc(iteracao);
            Verifica_PLMD;
          end
        else
          begin
            swapVectors;
            exec( '\dos\command.com','/C lindo < lindo32.bat');
            swapVectors;
            inc(iteracao);
            Verifica_PLMD;
          end;
       { Teste_de_parada2(y,x);}

        while PLMDviavel do
          begin
             Teste_de_parada2(y);
             if fx(y) < fx(y_otimo) then
                  begin
                     Atualiza_otimo;
                     passo_serio:= true;
                     ext:=0;   {numero de vezes da extensao ja feita}
                     PLMDviavel:=false;
```

```
                end
              else
                begin
                  passo_serio:= false;
                  PLMDviavel:= false;
                end
        end;
    end;

  until teste1 = false;

  Mostra_Resultado_Final;
end.
```

Referências Bibliográficas

[1] -Marco A . Duran And Ignacio E . Grossmann .
An Outer Approximation Algorithm For A Class Of Mixed Integer Nonlinear Programs .
Mathmatical Programming 36 (1986) 307-339.

[2] -A . M . Geoffrion
Generalized Benders Decomposition .
Journal Of Optimization Theory And Application : Vol . 10 , N° 4 , 1992 .

[3] -Arthur M . Geoffrion
Elements Of Large Scale Mathmatical Programming Part I , Part II .
Management Science Vol . 16 , Vol . 17 , July 1970 .

[4] -Han Tong Loh And Panos Y . Papalambros .
A Sequencial Linearization Approach For Solving Mixed Discrete Nonlinear Design Optimization Problems .
Technical Report Um-Meam - 89 - 08 June 1989

[5] -Han Tong Loh And Panos Y . Papalambros .
Computational Implementation And Tests Of A Sequential Linearization Algorithm For Mixed Discrete Nonlinear Design Optimization Problems .
Technical Report Um-Meam - 89 - 09

[6] -David G . Luenberger .
Linear And Nonlinear Programming .

[7] -George L . Nemhauser And laurence A . Wolsey .
Integer And Combinatorial Optimization .

[8] -Josef Stder And Christoph Witzgall .
Convexity And Optimization In finite Dimensions I .

[9] -Handy A . Taha .
Integer Programming Theory , Applications And Comtations .

[10] -Judith Liebman , Linus Schrage , Leon Lasdon , Allan Waren .
Applications Of Modeling And Optimization With Gino (General interactive Optimizer) .

[11] - Linus Schrage .
User's Manual For Linear , Integer And Quadratic Programming With Lindo .

[12] - R . Tynell Rockfeller.
Convex Analysis ,
Princetown University Press , Princetown , USA , 1972 .

[13] - E . Balas and R . Jeroslow .
Canonical Cuts On The Unit Hypercube ,
Siam Journal Of Applied Mathematics 23 . (1972) 61-79.

[14] - R . Fletcher , S.Leyffer .
Solving Mixed Integer Nonlinear Programs By Outer Approximation ,
Mathematical Programming 66 (1994) 327-349 .