UNIVERSIDADE FEDERAL DO AMAZONAS
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO DE APOIO À PESQUISA
PROGRAMA DE INICIAÇÃO CIENTÍFICA

INTRODUÇÃO À PROGRAMAÇÃO LINEAR INTEIRA MULTIOBJETIVO

Bolsista: Hewerton Umbelito da Silva Cruz, CNPq

Manaus - Amazonas
2015

UNIVERSIDADE FEDERAL DO AMAZONAS
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO DE APOIO À PESQUISA
PROGRAMA DE INICIAÇÃO CIENTÍFICA

RELATÓRIO FINAL
PIB - E / 0114 / 2014-2015
INTRODUÇÃO À PROGRAMAÇÃO LINEAR INTEIRA MULTIOBJETIVO

Bolsista: Hewerton Umbelito da Silva Cruz, CNPq
Orientador: Prof^{o} Dr. Mário Salvatierra Júnior

Manaus - Amazonas
2015

Pois a sabedoria entrará em seu coração, e o conhecimento será agradável à sua alma
Provérbios(Biblía)

Resumo

Neste relatório são apresentados alguns resultados de programação linear multiobjetivo. Mas para isto, mostraremos alguns tópicos relacionados a esse assunto, como por exemplo, Programação Linear, Método do Simplex e Algoritmo Branch and Bound.

Os principais Teoremas e Proposições com respeito aos assuntos relacionados acima estão devidamente demonstrados, e as definições necessárias para a compreensão dos mesmos encontram-se neste relatório.

No capítulo de Programação Linear apresentamos as definições e proposições necessários para compreender um problema de programação linear. Um método de resolução de um problema de programação linear é apresentado no capítulo 2, que é chamado de Método do Simplex. No capítulo 3, redirecionamos o enfoque para a programação linear inteira mono-objetivo apresentando o "Algoritmo Branch and Bound"como um método para achar soluções ótimas inteiras para dado o problema. No capítulo 4, temos a introdução de programação linear multiobjetivo com dois métodos de resoluções de problemas desse tipo, chamado Método dos Pesos e Método da ϵ-Restrição. E no último capítulo é apresentado um algoritmo para resoluções de problemas de programação linear inteira bi-critério.

Abstract

In this report we present some results of entire multi-objective linear programming continuously. But for this, we will show some topics related to this subject, such as linear programming, the simplex method and Branch and Bound Algorithm.

The main theorems and propositions with respect to the matters listed above are properly stated, and the settings required for understanding thereof are included in this report.

In Linear Programming chapter presented the definitions and propositions necessary to understand a linear programming problem. A method of solving a linear programming problem is presented in Chapter 2, which is called the Simplex method. In Chapter 3, we redirect the focus to the integer linear programming mono-goal showing the algorithm "Branch and Bound"as a method to find optimal solutions for whole given problem. In chapter 4, we have the introduction of multi-objective linear programming with two methods of resolutions of such problems, called the Weight Method and Method of ϵ-Constraint. And in the last chapter is presented an algorithm for resolution of integer linear programming bi-criteria problems.

Conteúdo

Introdução

Este relatório trata-se de uma Introdução à Programação Linear Inteira Multiobjetivo. O projeto tem como objetivo estudar a programação linear inteira multiobjetivo, fazendo o levantamento de alguns métodos existentes e compreender algoritmos eficientes da área. Serão apresentados definições, proposições e teoremas relacionados a programação linear inteira mono-objetivo com intuito de sermos aptos a tratar problemas com m funções objetivos, isto é, problemas de programação linear multiobjetivo.

O projeto foi dividido em duas etapas: A primeira etapa consistiu em revisar o método do simplex de programação linear, entender o funcionamento do algoritmo "Branch and Bound"e também foi visto uma introdução à programação linear multiobjetivo focando em dois métodos de resoluções de problemas de programação linear multiobjetivo: Método dos Pesos e o Método da ϵ-Restrição. A segunda etapa consistiu em estudar com mais detalhes os dois métodos apresentados na primeira etapa e por fim finalizando com um estudo de um artigo da revista *European Journal of Operational Research* intitulado *An algorothm for the bi-criterion integer programming problem* [5].

Neste relatório encontra-se o conhecimento teórico obtido ao longo desse período com todos os objetivos alcançados com sucesso.

Capítulo 1

Programação Linear

1.1 Introdução

Neste capítulo iremos definir as principais definições e propriedades a respeito de programação linear [3].

1.2 Definições e Propriedades dos Problemas de Programação Linear

Um problema de pogramação linear pode ser definido sob a seguinte forma:

$$\text{maximizar } z = \sum_{j=1}^{p} c_j x_j \tag{1.1}$$

sujeito a:

$$\sum_{j=1}^{p} a_{ij} x_j \leq b_i, i = 1, 2, ..., q \tag{1.2}$$

$$x_j \geq 0, j = 1, 2, ..., p, \tag{1.3}$$

onde c_j, a_{ij} e b_i são dados (números reais) e x_j representa para $j = 1, 2, ..., p$, as variáveis de decisão. A função linear a ser maximizada em (1.1) é denominada função objetivo, função econômica ou função critério. As restrições de não negatividade (1.3) são conhecidas como triviais. Cada restrição i de (1.2) pode ser substituída com o acréscimo de uma variável $x_{p+i} \geq 0$, denominada variável de folga, por uma restrição de igualdade e uma restrição trivial:

$$\textstyle\sum_{j=1}^{p} a_{ij} x_j \leq b_j \Leftrightarrow \sum_{j=1}^{p} a_{ij} x_j + x_{p+i} = b_i, \; x_{p+i} \geq 0.$$

ou

$$\textstyle\sum_{j=1}^{p} a_{ij} x_j \geq \Leftrightarrow \sum_{j=1}^{p} a_{ij} x_j - x_{p+i} = b_i, \; x_{p+i} \geq 0.$$

Uma restrição de igualdade poderá também ser substituída por duas desigualdades:

$\sum_{j=1}^{p} a_{ij}x_j = b_i \Leftrightarrow \sum_{j=1}^{p} a_{ij}x_j \leq b_i$ e $\sum_{j=1}^{p} a_{ij}x_j \geq b_i$,

Sendo dado um problema de programação linear com restrições de igualdades e desigualdades, poderemos acrescentar variáveis de folga às desigualdades não triviais, passando dessa maneira a trabalharmos com restrições de igualdades e desigualdades triviais. Assim sendo, um problema de programação linear poderá sempre ser escrito da seguinte forma:

(PPL): maximizar $z = \sum_{j=1}^{n} c_j x_j$

sujeito a:

$\sum_{j=1}^{n} a_{ij}x_j = b_i, i = 1, 2, ..., m \; x_j \geq 0, j = 1, 2, ..., n,$

que poderá ser ainda apresentado sob a forma abaixo:

$$\text{(PPL): maximizar } z = cx \tag{1.4}$$

sujeito a:

$$Ax = b \tag{1.5}$$

$$x \geq 0, \tag{1.6}$$

onde $c = (c_1, c_2, ..., c_n)$, $x^T = (x_1, x_2, ..., x_n)$, $b^T = (b_1, b_2, ..., b_m)$, $A = (a_1, a_2, ..., a_n)$ e $a_j^T = (a_{1j}, a_{2j}, ..., a_{mj})$, isto é, $c^T \in \mathbb{R}^n$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$ e $a_j \in \mathbb{R}^m$. A desigualdade (1.6) indica que cada componente do vetor x é não negativo. Indicaremos, portanto, que um dado vetor x tem pelo menos uma componente negativa através da notação $x \not\geq 0$.

Definição 1.2.1 *Seja $X = \{x \in \mathbb{R}^n | Ax = b, x \geq 0\}$. O conjunto X é denominado conjunto ou região viável do (PPL) e se $x \in X$, então x é uma solução viável do mesmo problema. Dado $x^* \in X$, x^* é denominado uma solução ótima do (PPL) se $cx^* \geq cx$, para todo $x \in X$.*

Suporemos, sem perda de generalidade, que a matriz A tenha posto igual a m, isto é, existem m colunas de A linearmente independentes. Como observação podemos dizer que a presença de uma variável x_j irrestrita em sinal será expressa: $x_j^+ - x_j^-$, $x_j^+ \geq 0$ e $x_j^- \geq 0$, deixando sempre o problema na forma (PPL).

Particionaremos a matriz A da seguinte maneira: $A = (B \; N)$, onde B é uma matriz quadrada $m \times m$ e inversível. Analogamente particionaremos os vetores x e c: $x^T = (x_B^T \; x_N^T)$, $c = (c_B \; c_N)$, x_B e c_B possuirão m componentes associadas à matriz B. Dessa maneira o (PPL) poderá ser escrito:

$$\text{(PPL): maximizar } z = c_B x_B + c_N x_N \tag{1.7}$$

sujeito a:

$$Bx_B + Nx_N = b \tag{1.8}$$

$$x_B \geq 0,\ x_N \geq 0. \tag{1.9}$$

Explicitaremos x_b em função de x_N em (1.8):

$$x_B = B^{-1}b - B^{-1}Nx_N. \tag{1.10}$$

Façamos $x_N = 0$ e $\hat{x}_B = B_{-1}b$.

Definição 1.2.2 *$\hat{x}$ é uma solução básica de* (1.5) *se $\hat{x}^T = (\hat{x}_B^T\ 0)$. As variáveis associadas às componentes de $\hat{x}_B$ são denominadas básicas e as demais não básicas. Quando $\hat{x}_B$ possuir ao menos uma componente nula diremos que $\hat{x}$ é uma solução básica degenerada.*

No caso em que $\hat{x}_B$ for não negativo, isto é, $\hat{x}_B \geq 0$, então $\hat{x}$ satisfará à restrição (1.6). Por força do hábito, diremos que esta solução $\hat{x}$ é uma solução básica primal viável. Sejam I_B o conjunto dos índices das colunas de A pertencendo à matriz B e I_N o conjunto dos demais índices de A. Lembremos que $I_B \cap I_N = \emptyset$ e $I_B \cup I_N = \{1, 2, ..., n\}$.

Levando a expressão de x_B em (1.10) na função objetivo (1.7) teremos outra forma do (PPL):

$$\text{(PPL): maximizar } z = c_B B^{-1}b - (c_B B^{-1}N - c_N)x_N \tag{1.11}$$

sujeito a:

$$x_B = B^{-1}b - B^{-1}Nx_N \tag{1.12}$$

$$x_B \geq 0,\ x_N \geq 0. \tag{1.13}$$

Por comodidade, definiremos seguindo alguns autores clássicos dos textos de programação linear, por exemplo, Dantzig [Dan 63] e Simonnard [Si 72], novos parâmetros para o último (PPL):

$$u = c_B B^{-1},\ u_T \in \mathbb{R}^m,$$

$$\hat{x}_B = B^{-1}b,\ \hat{x}_B \in \mathbb{R}^m,$$

$$z_j = ua_j\ (j \in I_B \cup I_N),\ z_j \in \mathbb{R},$$

$$y_j = B^{-1}a_j\ (j \in I_B \cup I_N),\ y_j \in \mathbb{R}^m,$$

$$\hat{z} = c_B B^{-1}b = ub = c_B\hat{x}_B.$$

Assim poderemos escrever $c_B B^{-1}N - c_N)x_N = \sum_{j \in I_N}(z_j - c_j)x_j$ e o (PPL) se tornará:

$$\text{(PPL): maximizar } z = \hat{z} - \sum_{j \in I_N} (z_j - c_j) x_j \tag{1.14}$$

sujeito a:

$$x_B = \hat{x}_B - \sum_{j \in I_N} y_j x_j \tag{1.15}$$

$$x_B \geq 0, \ x_j \geq 0, \ j \in I_N. \tag{1.16}$$

Definindo $y_j^T = (y_{1j}, y_{2j}, ..., y_{mj})$, $x_B^T = (x_{B(1)}, x_{B(2)}, ..., x_{B(m)})$ e $\hat{x}_B^T = (\hat{x}_{B(1)}, \hat{x}_{B(2)}, ..., \hat{x}_{B(m)})$ então (1.15) poderá ainda ser escrito como:

$$x_{B(i)} = \hat{x}_{B(i)} - \sum_{j \in I_N} y_{ij} x_j, \ i = 1, 2, ..., m. \tag{1.17}$$

Proposição 1.2.1 *Se $\hat{x}_B \geq 0$ e $z_j - c_j \geq 0$, para todo $j \in I_N$, então o vetor $x^* \in \mathbb{R}^n$, onde $x^*_{B(i)} = \hat{x}_{B(i)}$, $i = 1, 2, ..., m$ e $x^*_j = 0$, $j \in I_N$, será uma solução ótima do (PPL).*

Demonstração: Como $z_j - c_j \geq 0$ e $x_j \geq 0$, $\forall j \in I_N$, então de (1.14) temos $z \leq \hat{z} = cx^*$. O máximo de z não ultrapassará $\hat{z} = cx^*$, mas $x*$ é uma solução viável do (PPL), logo x^* é uma solução ótima do (PPL), como queríamos demonstrar.

$\blacksquare$

No caso da propriedade 1.1, x^* é uma solução básica de (1.5).

Suponhamos agora que $x' \in \mathbb{R}^n$ seja uma solução viável de (1.5) e (1.6). logo será também de (1.15) e (1.16), isto é,

$$x'_{B(i)} = \hat{x}_{B(i)} - \sum_{j \in I_N} y_{ij} x'_j, \ i = 1, 2, ..., m \tag{1.18}$$

e $x' \geq 0$, $j \in I_N \cup I_N$, fornecendo um valor z' à função objetivo:

$$z' = \hat{z} - \textstyle\sum_{j \in I_N} (z_j - c_j) x'_j = cx'$$

Suporemos também que $x' \in \mathbb{R}^n$ não seja uma solução básica de (1.5), isto quer dizer que haverá ao menos uma componente $x'_j > 0$, $j \in I_N$.

Variando o valor de uma variável x_k, $k \in I_N$ enquanto que o valor das outras variáveis cujos índices pertencem a I_N não se modificam, isto é, $x_j = x'_j$ para $j \in I_N - \{k\}$. De (1.18):

$$x_{B(i)} = \hat{x}_{B(i)} - \sum_{j \in I_N - \{k\}} y_{ik} x_k, \ i = 1, 2, ..., m \tag{1.19}$$

onde x_k poderá variar (aumentar ou diminuir).

Sabemos que $x_k \geq 0$, $x_{b(i)} \geq 0$, $i = 1, 2, ..., m$, e que os outros valores associados a x_j, $j \in I_N - \{k\}$, não serão modificados. Assim sendo: $x_{B(i)} \geq 0$ implica que

$$\hat{x}_{B(i)} - \sum_{j \in I_N - \{k\}} y_{ij} x'_j \geq 0, \; i = 1, 2, ..., m. \tag{1.20}$$

Consideremos L_0, L_1, L_2 uma partição de $\{1, 2, ..., m\}$, tal que

$$L_0 = \{i | y_{ik} = 0\}, \; L_1 = \{i | y_{ik} > 0\}, \; L_2 = \{i | y_{ik} < 0\}.$$

Busquemos os limites de variação para x_k pois sabemos que de (1.20):

$$y_{ik} x_k \leq \hat{x}_{B(i)} - \sum_{j \in I_N - \{k\}} y_{ij} x'_j, \; i = 1, 2, ..., m. \tag{1.21}$$

Para $i \in L_0$ basta que o valor de x_k seja não negativo. Para $i \in L_1$:

$$x_k \leq \frac{1}{y_{ik}} \left(\hat{x}_{B(i)} - \textstyle\sum_{j \in I_N - \{k\}} y_{ij} x'_j \right).$$

Para $i \in L_2$:

$$x_k \geq \frac{1}{y_{ik}} \left(\hat{x}_{B(i)} - \textstyle\sum_{j \in I_N - \{k\}} y_{ij} x'_j \right).$$

Sejam

$$\alpha_k = \frac{1}{y_{sk}} \left(\hat{x}_{B(s)} - \textstyle\sum_{j \in I_N - \{k\}} y_{sj} x'_j \right) = \min_{i \in L_1} \left\{ \frac{1}{y_{ik}} \left(\hat{x}_{B(i)} - \textstyle\sum_{j \in I_N - \{k\}} y_{ij} x'_j \right) \right\},$$

$$\beta_k = \frac{1}{y_{lk}} \left(\hat{x}_{B(l)} - \textstyle\sum_{j \in I_N - \{k\}} y_{lj} x'_j \right) = \max_{i \in L_2} \left\{ \frac{1}{y_{ik}} \left(\hat{x}_{B(i)} - \textstyle\sum_{j \in I_N - \{k\}} y_{ij} x'_j \right) \right\},$$

e $\gamma_k = \max\{0, \beta_k\}$.

Logo $\gamma_k \leq x_k \leq \alpha_k$.

Quando $L_1 = \emptyset \Rightarrow \alpha_k = \infty$ e quando $L_2 = \emptyset \Rightarrow \beta_k = -\infty$.

A matriz $B = (a_{B(1)}, a_{B(2)}, ..., a_{B(m)})$ extraída de A foi utilizada para chegarmos ao sitema (2.15) a partir de (1.5). Os vetores $a_{B(i)}$, $i = 1, 2, ..., m$, formam uma base do $\mathbb{R}^m$, logo existem $\lambda_i \in \mathbb{R}$, $i = 1, 2, ..., m$, para os quais $a_k = \sum_{i=1}^{m} \lambda_i a_{B(i)}$. Seja $s \in \{1, 2, ..., m\} = M$ tal que $\lambda_s \neq 0$, então $a_{b(s)} = \frac{1}{\lambda_s} \left(a_k - \sum_{i \in M - \{s\}} \lambda_i a_{B(i)} \right)$, como $a_{B(s)} \neq 0$, a coluna a_k não pode ser escrita como uma combinação linear das colunas $a_{B(i)}$, $i \in M - \{M\}$; isto quer dizer que $a_{B(1)}, a_{B(2)}, ..., a_{B(s-1)}, a_k, a_{B(s+1)}, ..., a_{B(m)}$ formam também uma base do $\mathbb{R}^m$.

Seja $v^T = (\lambda_1, \lambda_2, ..., \lambda_m)$, assim podemos escrever $a_k = Bv$, logo $v = B^{-1} a_k$, isto é, $v = y_k$.

Basta que $y_{sk} \neq 0$ para que possamos substituir a base formada pelas colunas de B por outra base em que o vetor $a_{B(s)}$ é substituído por a_k.

Procedimento 1

Tomemos x_k tal que $x_k = x'_k > 0$ e $k \in I_N$.

1º caso: $z_k - c_k > 0$, descrevemos o valor de x_k até alcançar γ_k;

se $\gamma_k = 0$, faremos $x_k = 0$ e utilizaremos (1.19) para atualizar os valores de $x_{B(i)}$, $i = 1, 2, ..., m$;

se $\gamma_k = \beta_k$, faremos $x_k = \beta_k$ que ocasionará $x_{b(l)} = 0$ em (1.19), como $y_{lk} \neq 0$ então poderemos fazer

$$I_B := (I_B - \{B(l)\}) \cup \{k\},$$

$$I_N := (I_N - \{k\}) \cup \{B(l)\},$$

isto é, teremos uma nova matriz B inversível, extraída de A, onde a coluna $a_{B(l)}$ será substituída por a_k;

2º caso: $z_k - c_k < 0$, aumentaremos o valor de x_k até alcançar α_k;

se $\alpha_k = +\infty$, a solução do (PPL) será ilimitada, pois $x_k \to +\infty$ implica $z \to +\infty$;

se $\alpha_k < \infty$, faremos $x_k = \alpha_k$ que ocasionará $x_{B(s)} = 0$ em (1.19), como $y_{sk} \neq 0$ então poderemos fazer

$$I_B := (I_B - \{B(s)\}) \cup \{k\},$$

$$I_N := (I_N - \{k\}) \cup \{B(s)\},$$

isto é, teremos uma nova matriz B inversível, extraída de A, onde a coluna $a_{B(s)}$ será substituída por a_k;

3º caso: $z_k - c_k = 0$, aplicaremos o que foi realizado no 1º caso.

Fim do procedimento 1

Para cada $j \in I_N$ tal que $x'_j > 0$, o procedimento 1 feito para o índice k será repitido até que os valores atribuídos às varáveis x_j, $j \in I_N$, sejam nulos, ou que a solução máxima (ótima) do (PPL) seja ilimitada (2º caso, $\alpha_k = +\infty$).

o procedimento 1 será aplicado r vezes onde $r = |\left\{j \in I_N | x'_j < 0\right\}|$.

Com as explicações anteriores poderemos enunciar mais duas propriedades a seguir.

Proposição 1.2.2 *Se* (1.5), (1.6) *admitirem uma solução viável, então haverá ao menos uma solução básica de* (1.5) *satisfazendo* (1.6).

Proposição 1.2.3 *Se o (PPL) posuir ótimo finito ao menos uma solução ótimaserá viável.*

A propriedade 1.1.2 poderá ser demonstrada ultilizando o 1º caso do procedimento 1 para qualquer $z_k - c_k \in \mathbb{R}$, $k \in I_N$, $x'_k > 0$.

Aplicaremos r vezes, onde $r = |\left\{j \in I_N | x'_j > 0\right\}|$, o procedimento 1 para demonstrar a propriedade 1.1.3.

Capítulo 2

Método do Simplex

2.1 Introdução

Dantzig, em 1947, introduziu o método do simplex [3] para resolver um problema de programação linear (PPL).

A idéia do método a partir de uma solução básica de (1.5) satisfazendo (1.6), isto é, uma solução básica primal viável, passar para outra solução básica primal viável sem que o valor da função objetivo diminua (no caso de maximização). Como o número de soluções básicas é finito, o algoritmo, sob algumas condições, convergirá.

Dada a matriz B quadrada e inversível, extraída de A, tal que $\hat{x}_B \geq 0$, colocaremos o problema de programação linear (1.4), (1.5) e (1.6) sob a forma (1.14), (1.15) e (1.16). Utilizando a propriedade 1.1.1, testaremos se esta solução é ótima, caso não o seja tentaremos aumentar o valor de uma variável x_k, $k \in I_N$, tal que $z_k - c_k < 0$, como já explicado no 2º caso do procedimento 1. Se $\alpha_k == \infty$ então não haverá ótimo finito.

No caso em que só iremos trabalhar com soluções básicas viáveis, o cálculo de α_k é mais simplicado:

$$\alpha_k = \frac{\hat{x}_{B(s)}}{y_{sk}} = \min_{i \in L_1} \left\{ \frac{\hat{x}_{B(i)}}{y_{ik}} \right\}$$

e caso $L_1 = \emptyset$ faremos $\alpha_k = +\infty$.

a seguir descreveremos um procedimento que resume o método do simplex.

2.2 Algoritmo do Simplex

Procedimento 2

Dada uma solução básica primal viável para o (PPL).

Se $z_j - c_j \geq 0$, $\forall j \in I_N$, a solução dada é uma solução ótima.PARE.

Caso contrário, escolhe-se um $k \in I_N$ para o qual $z_k - c_k < 0$;

se $\alpha_k = +\infty$, a solução do (PPL) é limitada.PARE.

se $\alpha_k < +\infty$, faremos $x_k = \alpha_k$, acarretando $x_{B(s)} = 0$, a coluna a_k ocupará o lugar da coluna $a_{B(s)}$ em B (Mudança de base).

Fim do procedimento 2

Para cada nova base o procedimento 2 é repetido até que uma regra de parada seja verificada. Este procedimento possui duas saídas: solução ótima encontrada ou solução ilimitada.

2.3 Busca de uma Solução Básica Viável

Quando não tivermos uma solução básica viável para o (PPL), poderemos proceder da seguinte maneira.

Acrescentaremos uma variável artificial $g_i \geq 0$ à esquerda de cada restrição $\sum_{j=1}^{n} a_{ij}x_j = b_i$, $i = 1, 2, ..., m$. Suporemos $b_i \geq 0$, $i = 1, 2, ..., m$. Teríamos o seguinte conjunto de restrições:

$$\sum_{j=1}^{n} a_{ij}x_j + g_i = b_i, i = 1, 2, ..., m \tag{2.1}$$

$$x_j \geq 0, j = 1, 2, ..., n, \tag{2.2}$$

$$g_i \geq 0, i = 1, 2, ..., m \tag{2.3}$$

Construiremos um outro problema de programação linear

$$\text{(PA) : minimizar} \textstyle\sum_{i=1}^{n} g_i$$

sujeito a

$$\textstyle\sum_{j=1}^{n} a_{ij}x_j + g_i = b_i, i = 1, 2, ..., m$$

$$x_j \geq 0, j = 1, 2, ..., n,$$

$$g_i \geq 0, i = 1, 2, ..., m.$$

Utilizaremos $v(\cdot)$ para representar o valor ótimo da função objetivo do problema de programação linear $(\cdot)$.

É fácil verificar que as variáveis $x_j = 0, j = 1, 2, ..., n$ e $g_i = b_i \geq 0$ estão associadas a um solução básica de (2.1) satisfazendo (2.2) e (2.3). Esta solução básica será tomada como solução inicial para a solução de (PA) utilizando o método do simplex (procedimento 2) para o caso de minimização, lembrando que $\min z = -\max(-z)$.

Se $v(PA) > 0$ o conjunto de restrições do (PPL) é vazio.

Se $v(PA) = 0$ a solução ótima obtida pra (PA) terá $g_i = 0, i = 1, 2, ..., m$ e $x_j = \hat{x}_j \geq 0$ para $j = 1, 2, ..., n$ satisfazendo a $\sum_{j=1}^{n} a_{ij}\hat{x}_j = b_i, i = 1, 2, ..., m$. Se a base final da solução ótima de (PA) não contiver nenhuma coluna associada às variáveis artificiais $g - i, i = 1, 2, ..., m$ esta será também uma base primal viável do (PPL) original. Caso a solução básica ótima encontrada para o (PA) seja degenerada, isto é, há pelo menos uma coluna associada a g_i na base ótima e $g_i = 0$, poderemos iniciar o método do simplex para o (PPL) com esta base, não permitindo que as variáveis g_i associadas à base tenham valores diferentes de zero.

A solução básica do (PA) é conhecida como sendo a primeira fase do método do simplex.

2.4 Cálculo de B^{-1}

Lembremos que em cada etapa do método do simplex necessitamos determinar:

$$u = c_B B^{-1},\ \hat{x}_B B^{-1} b \text{ e } y_k = B^{-1} a_k,$$

onde a_k é a coluna escolhida para entrar na base.

Na realidade temos três sistemas de equações lineares simultâneas para serem resolvidos:

$$uB = c_B,\ B\hat{x}_B = b \text{ e } By_k = a_k,$$

que determinam respectivamente os vetores u, $\hat{x}_b$ e y_k.

caso só tivéssemos de resolver uma vez esses três sistemas, poderíamos utilizar os métodos numéricos de resolução de sistemas de equações lineares simultâneas sem a inversão explícita de B, no entanto, em cada interação do simplex buscamos a solução dos três sistemas para os quais B, c_B e a_k variam de interação em interação.

Geralmente, a primeira base B associada a uma solução básica viável, no método simplex, é a matriz unitária I. por outro lado, já verificamos que de uma interação à seguinte a matriz B se transforma em outra martiz B' trocando-se somente uma coluna de B.

Seja $B_r = (a_1, a_2, ..., a_{r-1}, a_r, a_{r+1}, ..., a_m)$ uma matriz quadrada inversível, $m \times m$. Suponhamos conhecida B_r^{-1}.

Consideremos,

$$B_r^{-1} B_p = (B_r^{-1} a_1,\ B_r^{-1} a_2,\ ..., B_r^{-1} a_{r-1},\ B_r^{-1} a_p,\ B_r^{-1} a_{r+1},\ ..., B_r^{-1} a_m),$$

sabemos que $B_r^{-1} a_j = e_j, j \neq p$, onde e_j é um vetor com todas as componentes nulas exceto a j-ésima componente qu é igual a um.

Seja $v = B_r^{-1} a_p = (v_1, v_2, ..., v_{r-1}, v_r, v_{r+1}, ..., v_m)^T$, então

$$B_r^{-1} B_p = (e_1, e_2, ..., e_{r-1}, v, e_{r+1}, ..., e_m) = E_r, \tag{2.4}$$

ou ainda

$$E_r = \begin{bmatrix} 1 & 0 & 0 & 0 & v_1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & v_2 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 1 & v_{r-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_r & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_{r+1} & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & v_{m-1} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & v_m & 0 & 0 & 0 & 1 \end{bmatrix}$$

De (3.4) teremos que $B_p = B_r E_r$. Verificamos que existirá B_p^{-1} se e somente se existir E_r^{-1}. Notemos que o determinante de E_r é igual a v_r, basta que $v_r \neq 0$ para termos E_r^{-1}, assim sendo:

$$B_p^{-1} = (B_r E_r)^{-1} = E_r^{-1} B_r^{-1} \tag{2.5}$$

Proposição 2.4.1 *Se $v_r \neq 0$, então a inversa de E_r será*

$$E_r^{-1} = (e_1, e_2, ..., e_{r-1}, \overline{v}, e_{r+1}, ..., e_m),$$

onde

$$\overline{v} = (-\tfrac{v_1}{v_r}, -\tfrac{v_2}{v_r}, ..., -\tfrac{v_{r-1}}{v_r}, \tfrac{1}{v_r}, -\tfrac{v_{r+1}}{v_r}, ..., -\tfrac{v_{m-1}}{v_r}, -\tfrac{v_m}{v_r})^T.$$

Ou ainda

$$E_r^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & -\frac{v_1}{v_r} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -\frac{v_2}{v_r} & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 1 & -\frac{v_{r-1}}{v_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{v_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{v_{r+1}}{v_r} & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & -\frac{v_{m-1}}{v_r} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -\frac{v_m}{v_r} & 0 & 0 & 0 & 1 \end{bmatrix}$$

Demonstração

Basta realizarmos $E_r E_r^{-1}$ e encontraremos I.

■

Suponhamos agora que $B = (a_1, a_2, ..., a_m)$ é uma matriz quadrada $m \times m$ e desejamos encontrar sua inversa, caso exista.

Partiremos da matriz unitária $I = (e_1, e_2, ..., e_m)$, cuja inversa é a própria matriz I. Iniciaremos calculando a inversa de $B_1 = (a_1, e_2, ..., e_m)$ a partir de I tal como foi feito neste capítulo, isto é, $E_1 = B_1$, então $B_1^{-1} = E_1^{-1}$. A partir de B_1 da qual já conhecemos a inversa, passaremos a calcular a inversa de $B_2 = (a_1, a_2, e_3, ..., e_m)$. Sabemos que $B_1^{-1}B_2 = (e_1, B_1^{-1}a_2, e_3, ..., e_m) = E_2$, a inversa de E_2 é obtida pelo método já visto. Assim sendo $B_2 = B_1E_2 \Rightarrow B_2^{-1} = E_2^{-1}B_1^{-1} = E_2^{-1}E_1^{-1}$. Seja $B_3 = (a_1, a_2, a_3, e_4, ..., e_m)$ cuja inversa desejamos obter, logo poderemo escrever $B_2^{-1}B_3 = (e_1, e_2, B_2^{-1}a_3, e_4, ..., e_m) = E_3 \Rightarrow B_3^{-1} = E_3^{-1}B_2^{-1} = E_3^{-1}E_2^{-1}E_1^{-1}$. Continuando com esse raciocínio obteremos $B^{-1} = E_m^{-1}E_{m-1}^{-1}...E_3^{-1}E_2^{-1}E_1^{-1}$.

Capítulo 3

Programação Linear Inteira

3.1 Introdução

Neste capítulo introduzimos a ideia de programação inteira juntamente com o algoritmo "Branch and Bound"[1].

Quando nos problemas de programação linear obrigarmos algumas ou todas as variáveis de decisão a só admitirem valores inteiros, estaremos diante de um problema de programação linear inteira. Por conseguinte, uma primeira aproximação para a solução de qualquer problema de programação linear inteira é ignorar a restrição das variáveis inteiras e resolver por algum método, por exemplo, método do simplex. Se a solução ótima para o problema de programação linear for inteiro, em seguida, esta solução também é uma solução ótima para o problema original. Caso contrário (e esta é uma situação normal), pode-se arredondar os componentes com os números inteiros mais próximos viáveis e obter uma segunda aproximação. Este procedimento é muitas vezes levado em conta, especialmente quando a primeira aproximação envolve grandes números mas pode ser impreciso quando os números são pequenos.

3.2 Algoritmo Branch and Bound

O método baseia-se na idéia de desenvolver uma enumeração inteligente dos pontos candidatos à solução ótima inteira de um problema. O termo *branch* refere-se ao fato de que o método efetua partições no espaço das soluções. O termo *bound* ressalta que a prova da otimalidade da solução utiliza-se de limites calculados ao longo da enumeração. Definindo:

$$\text{(P)} = \text{mazimizar } cx$$

$$\text{sujeito a:}$$

$$Ax = b$$

$$x \geq 0$$

$$x \in \mathbb{Z}^+$$

e

$$(\overline{P}) \text{ maximizar } cx$$

$$\text{sujeito a:}$$

$$Ax = b$$

$$x \geq 0$$

$$x \in \mathbb{R}^+$$

Definindo ainda $V*(P)$ e $V*(\overline{P})$ os valores das funções objetivo no ótimo de (P) e $(\overline{P})$ respectivamente temos que:

$$V*(P) \leq V*(\overline{P})$$

Considerando ainda qualquer solução viável $\hat{x}$ de (P) e chamado de $V(\hat{x})$ o valor da função objetivo no ponto $\hat{x}$, então:

$$V(\hat{x}) \leq V*(P)$$

e dessa forma $V*(\overline{P})$ é um limite superior de (P) e qualquer de suas soluções viáveis. Se $\overline{x}$ é a solução ótima de $(\overline{P})$ tal que $\overline{x}_j$ é não inteiro, teremos:

$$x_j \geq \overline{x}_j + 1 \text{ ou } x_j \leq \overline{x}_j$$

em toda solução viável de (P). Dessa forma, o problema (P) pode ser dividid em dois novo problemas (P_1) e (P_2) em que a envoltória convexa $C \in (P_1) \cup (P_2)$ está estritamente contida na envoltória de (P).

Um dos pontos fundamentais para o sucesso do *Branch and Bound* é qualidade do limite gerado pela solução inteira. Em várias situações, esses limites podem ser alcançados através de procedimentos heurísticos. A qualidade do limite alcançado normalmente depende, para cada problema, da estratégia de desdobramento da árvore de busca.

Capítulo 4

Programação Linear Multiobjetivo

4.1 Introdução

Neste capítulo entramos no assunto do projeto, na qual pegamos as principais definições e propriedades aqui aprensentadas [4].

Até agora, tratamos somente com problemas de programação linear mono-objetivo, isto é, contendo apenas uma função objetivo. O Conceito de programação linear multiobjetivo trata-se de trabalhar com p funções objetivos em um único problema. Consideremos o seguinte problema:

$$\begin{gathered} \min(f_1(x), ..., f_p(x)) \\ \text{sujeito a } x \in X \end{gathered} \tag{4.1}$$

A partir daí, veremos a seguir as definições e propriedades de um problema de programação linear multiobjetivo.

4.2 Soluções Eficientes e Pontos Não-Dominados

A imagem do conjunto factível X sob a função objetivo mapeando f é denotada por $\mathbb{Y} := f(X)$. Vamos definir formalmente as definições de soluções eficientes e pontos não-dominados.

Definição 4.2.1 *O conjunto factível $\hat{x} \in X$ é chamado eficiente ou Pareto Ótimo, se e somente se, não existir $x \in X$ tal que $f(x) \leq f(\hat{x})$. Se $\hat{x}$ é eficiente, $f(\hat{x})$ é chamado de ponto não-dominado. Se $x^1, x^2 \in X$ e $f(x^1) \leq f(x^2)$ dizemos que x^1 domina x^2 e $f(x^1)$ domina $f(x^2)$. O conjunto de todas as soluções eficientes $\hat{x} \in X$ é detonado por X_E e chamado de Conjunto Eficiente. O conjunto de todos os pontos não-dominados $\hat{y} = f(\hat{x}) \in \mathbb{Y}$, onde $\hat{x} \in X_E$, é denotado por $\mathbb{Y}_N$ e chamado de Cojunto Não-Dominante.*

Se $\hat{x}$ é eficiente, então vale as seguintes propriedades:

(i) Não existe $x \in X$ tal que $f_i(x) \leq f_i(\hat{x}), i = 1, 2, ..., m$ e $f_j(x) < f_j(\hat{x})$ para algum $j = 1, ..., i$;

(ii) Não existe $x \in X$ tal que $f(x) - f(\hat{x}) \in -(\mathbb{R}^m_{\geq} + \{0\})$;

(iii) $f(x) - f(\hat{x}) \in \mathbb{R}^m - (\mathbb{R}^m_{\geq} + \{0\}) \forall x \in X$;

(iv) $f(X) \cap (f(\hat{x}) - \mathbb{R}^m_{\geq}) = \{f(\hat{x})\}$;

(v) Não existe $f(x) \in f(X) - \{f(\hat{x})\}$ com $f(x) \in f(\hat{x}) - \mathbb{R}^m_{\geq}$;

(vi) $f(x) \leq f(\hat{x})$ para algum $x \in X \Rightarrow f(x) = f(\hat{x})$.

Proposição 4.2.1 $\mathbb{Y}_N = (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$.

Demonstração

O resultado é trivial se $\mathbb{Y} = \emptyset$ por que $\mathbb{Y} + \mathbb{R}^m_{\geq} = \emptyset$ e o conjunto não-dominante é vazio também.

Seja $\mathbb{Y} \neq \emptyset$. Primeiro, assumimos $y \in (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$, mas $y \notin \mathbb{Y}_N$. Onde há duas possibilidades. Se $y \in \mathbb{Y}$ onde $y' \in \mathbb{Y}$ e $0 \neq d \in \mathbb{R}^m_{\geq}$ tal que $y = y' + d$, desde $y' = y' + 0 \in \mathbb{Y} + \mathbb{R}^m_{\geq}$, obtemos $y \in (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$, contradição.

Se $y \in \mathbb{Y}$ onde $y' \in \mathbb{Y}$ tal que $y' \leq y$. Seja $d = y - y' \in \mathbb{R}^m_{\geq} - \{0\}$. Portanto $y = y' + d$ e $y \notin (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$, daí novamente contradizendo a suposição, daí em ambos os casos $y \in \mathbb{Y}_N$.

Segundo, assumimos $y \in \mathbb{Y}_N$ mas $y \notin (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$. Em seguida, $y' \in \mathbb{Y} + \mathbb{R}^m_{\geq}$ com $y - y' = d' \in \mathbb{R}^m_{\geq} - \{0\}$. Fazendo, $y' = y'' + d''$ com $y'' \in \mathbb{Y}$, $d'' \in \mathbb{R}^m_{\geq} - \{0\}$ e portanto $y = y' + d' = y'' + (d' + d'') = y'' + d$ com $d = d' + d'' \in \mathbb{R}^m_{\geq} - \{0\}$. Implica $y \notin \mathbb{Y}_N$, contradizendo a suposição.

Logo, $y \in (\mathbb{Y} + \mathbb{R}^m_{\geq})_N$, como queríamos demonstrar.

■

"Os pontos não-dominados de $\mathbb{Y}_N$ e $(\mathbb{Y} + \mathbb{R}^m_{\geq})_N$ são os mesmos."

4.3 Método dos Pesos

Um dos métodos mais comuns para resolver um problema de programação linear multiobjetivo é o método dos pesos (em inglês, *the weighting method*), onde todas as funções objetivo são combinadas em uma única função objetivo usando um vetor de pesos $\lambda \leq 0$, com $\| \lambda \| = 1$. Dessa forma, o problema original transforma-se num problema de um único objetivo com as restrições originais. O método dos pesos serve para obter uma aproximação da fronteira eficiente e foi experimentado, primeiramente, em Zadeh [Zadeh, 1963]. A vantagem deste método é sua simplicidade.

Vamos buscar soluções eficientes para problemas de programação linear multiobjetivo da seguinte composição:

$$\min_{x \in X} \sum_{k=1}^{p} \lambda_k f_k(x) \tag{4.2}$$

Definição 4.3.1 *Uma solução ótima $\hat{x} \in X$ é chamada propriamente eficiente, se for eficiente e se existir um número real $M > 0$ tal que para todo i e $x \in X$ satisfazendo $f_i(x) < f_i(\hat{x})$ existe um índice j tal que $f_j(\hat{x}) < f_j(x)$ tal que*

$$\frac{f_i(\hat{x}) - f_i(x)}{f_j(x) - f_j(\hat{x})} \leq M. \tag{4.3}$$

O correspondente ponto $\hat{y} = f(\hat{x})$ é chamado de propriamente não-dominado.

Definição 4.3.2 *Seja $X \subset \mathbb{R}^n$ um conjunto convexo, sejam $h_k : \mathbb{R}^n \to \mathbb{R}$ funções convexas, $k = 1, ..., p$. Se o sistema $h_k(x) < 0, k = 1, ..., p$ não tem solução para $x \in X$, então existem $\lambda_k \geq 0, \sum_{k=1}^{p} \lambda_k = 1$ tal que para todo $x \in X$ satisfaça*

$$\sum_{k=1}^{p} \lambda_k h_k(x) \geq 0 \tag{4.4}$$

Teorema 4.3.1 *Seja $\lambda_k > 0, k = 1, ..., p$ com $\sum_{k=1}^{p} \lambda_k = 1$ de pesos positivos. Se $\hat{x}$ é uma solução ótima para $\min_{x \in X} \sum_{k=1}^{p} \lambda_k f_k(x)$ onde $\hat{x}$ é uma solução propriamente eficiente de $\min_{x \in X}(f_1(x), ..., f_p(x))$.*

Demonstração

Seja $\hat{x}$ uma solução ótima de $\min_{x \in X} \sum_{k=1}^{p} \lambda_k f_k(x)$. Para mostrar que $\hat{x}$ é eficiente vamos supor que exista algum $x' \in X$ com $f(x') \leq f(\hat{x})$. Pela positividade dos pesos λ_k e $f_i(x') < f_i(\hat{x})$ para algum $i \in \{1, ..., p\}$ implica na contradição

$$\textstyle\sum_{k=1}^{p} \lambda_k f_k(x') < \sum_{k=1}^{p} \lambda_k f_k(\hat{x}).$$

Para mostrar que $\hat{x}$ é propriamente eficiente, nós escolhemos um M adequadamente grande assumindo que há um trade-off maior que M produz uma conrtadição na otimalidade de $\hat{x}$ para o problema dos pesos. Seja

$$M := (p-1) \max_{i,j} \tfrac{\lambda_j}{\lambda_i}.$$

Supondo que $\hat{x}$ não é propriamente eficiente. Então existem $i \in \{1, ..., p\}$ e $x \in X$ tal que $f_i(x) < f_i(\hat{x})$ e $f_i(\hat{x}) - f_i(x) > M(f_j(x) - f_j(\hat{x})) \forall j \in \{1, ..., p\}$ tal que $f_j(\hat{x}) < f_j(x)$. Portanto,

$$f_i(\hat{x}) - f_i(x) > \tfrac{p-1}{\lambda_i} \lambda_j (f_j(x) - f_j(\hat{x}))$$

para todo $j \neq i$ pela escolha de M (note que a desigualdade é trivialmente verdadeira se $f_j(\hat{x}) > f_j(x)$). Multiplicando ambos os lados da desigualdade por $\frac{\lambda_i}{(p-1)}$ e somando os $j \neq i$

$$\lambda_i(f_i(\hat{x}) - f_i(x)) > \textstyle\sum_{j \neq i} \lambda_i (f_j(x) - f_j(\hat{x}))$$

$$\Rightarrow \lambda_i f_i(\hat{x}) - \lambda_i f_i(x) > \textstyle\sum_{j \neq i} \lambda_j f_j(x) - \sum_{j \neq i} \lambda_j f_j(\hat{x})$$

$$\Rightarrow \lambda_i f_i(\hat{x}) + \textstyle\sum_{j \neq i} \lambda_j f_j(\hat{x}) > \lambda_i f_i(x) + \sum_{j \neq i} \lambda_j f_j(x)$$

$$\textstyle\sum_{i=1}^{p} \lambda_i f_i(\hat{x}) > \sum_{i=1}^{p} \lambda_i f_i(x),$$

contradizendo a otimalidade de $\hat{x}$ para $\min_{x \in X} \sum_{k=1}^{p} \lambda_k f_k(x)$. Portanto, $\hat{x}$ é propriamente eficiente, como queríamos demonstrar.

$\blacksquare$

Teorema 4.3.2 *Seja $X \subset \mathbb{R}^n$ convexo e assumindo $f_k : X \to \mathbb{R}$ convexos para $k = 1, ..., p$. Então $\hat{x} \in X$ é propriamente eficiente se, e somente se $\hat{x}$ é uma solução ótima de $\min_{x \in X} \sum_{k=1}^{p} \lambda_k f_k(x)$, com os pesos estritamente positivos, $\lambda_k, k = 1, ..., p$.*

Demonstração

Devido ao Teorema 5.3.1 nós só temos que demonstrar a necessidade da condição. Seja $\hat{x} \in X$ propriamente eficiente. Então, por definição, existe um número $M > 0$ tal que para todo $i = 1, ..., p$ o sistema

$$f_i(x) < f_i(\hat{x})$$

$$f_i(x) + Mf_j(x) < f_i(\hat{x}) + Mf_j(\hat{x}) \ \forall \ j \neq i$$

não tem solução. Para ver basta reorganizar as desigualdades na Definição 5.3.1.

A propriedade de funções convexas, que afirma na Definição 5.3.2 implica que para o sistema existam $\lambda_k^i \geq 0, k = 1, ..., p$ com $\sum_{k=1}^p \lambda_k^i = 1$ tal que para todo $x \in X$ as seguintes desigualdades detém

$$\lambda_i^i f_i(x) + \sum_{k\neq i}(f_i(x) + Mf_k(x)) \geq \lambda_i^i f_i(\hat{x}) + \sum_{k\neq i} \lambda_k^i (f_i(\hat{x}) + Mf_k(\hat{x}))$$

$$\Leftrightarrow \lambda_i^i f_i(x) + \sum_{k\neq i} \lambda_k^i f_i(x) + M\sum_{j\neq i} \lambda_j^i f_k(x) \geq \lambda_i^i f_i(\hat{x}) + \sum_{k\neq i} \lambda_k^i f_i(\hat{x}) + M\sum_{k\neq i} \lambda_j^i f_k(\hat{x})$$

$$\Rightarrow \sum_{k=1}^p \lambda_k^i f_i(x) + M\sum_{k\neq i} \lambda_k^i f_k(x) \geq \sum_{k=1} \ p\lambda_k^i f_i(\hat{x}) + M\sum_{k\neq i} \lambda_k^i f_j(\hat{x})$$

$$\Leftrightarrow f_i(x) + M\sum_{k\neq i} \lambda_k^i f_k(x) \geq f_i(\hat{x}) + M\sum_{k\neq i} \lambda_k^i f_k(\hat{x})$$

Nós temos uma desigualdade para cada $i = 1, ..., p$ e agora simplesmente somando todos os i's obtemos

$$\sum_{i=1}^p f_i(x) + M\sum_{i=1}^p \sum_{k\neq i} \lambda_k^i f_k(x) \geq \sum_{i=1}^p f_i(\hat{x}) + M\sum_{k=1}^p \sum_{k\neq i} \lambda_j^i f_k(\hat{x})$$

$$\Rightarrow \sum_{k=1}^p (1 + M\sum_{i\neq k} \lambda_k^i) f_k(x) \geq \sum_{k=1}^p (1 + M\sum_{i\neq k} \lambda_k^i) f_k(\hat{x})$$

para todo $x \in X$.

Podemos agora normalizar os valores $(1 + M\sum_{k\neq i} \lambda_k^i)$, tal que a soma é igual a um até obter $\lambda_i, i = 1, ..., p$ positivo para o qual $\hat{x}$ é ótimo para $\min_{x\in X} \sum_{k=1}^p \lambda_k f_k(x)$, como queríamos demonstrar.

■

4.4 Método da ϵ-Restrição

O método da ϵ-Restrição é bastante conhecido. Ele consiste em minimizar um único objetivo enquanto os outros são incorporados ao conjunto de restrições do problema, sendo restringidos pelos valores das componentes de um vetor $\epsilon \in \mathbb{R}^{p-1}$, fornecido a priori. Esta abordagem de otimizar uma função objetivo enquanto outras são incorporadas às restrições parece ter sido, inicialmente, sugerida em [Marglin, 1967]. Posteriormente, Haimes *et at.* apresentaram uma nova formulação para um problema envolvendo duas funções objetivo [Haimes *et al.*, 1971]. Esta formulação, intitulada por eles de formulação da ϵ-restrição, é a origem do método que detém o mesmo nome.

4.4.1 Descrição do método

O problema que iremos trabalhar é o seguinte:

$$\begin{gathered} P_k(\epsilon) : min \ f_k(x) \\ s.a : f_j(x) \leq \epsilon_j, j = 1, ..., p, j \neq k, \\ g_i(x) \leq 0, i = 1, ..., m. \end{gathered}$$

A função objetivo que será minimizada é escolhida a priori, assim o vetor $\epsilon \in \mathbb{R}^{p-1}$, o qual é considerado como parâmetro do método. Satisfazendo algumas condições, uma solução ótima do problema pode ser uma solução eficiente para o problema original. Uma das condições necessárias para que isso ocorra está relacionada, antes de tudo, à viabilidade do problema. Para que soluções viáveis existam, é fundamental que o vetor $\epsilon \in \mathbb{R}^{p-1}$ seja escolhida adequadamente, de forma que exista pelo menos um ponto x satisfazendo as restrições do problema.

Neste método, o que fazemos é escolher alguma função objetivo f_k para ser minimizada e tentar obter uma aproximação da fronteira eficiente a partir das soluções ótimasdo problema para diferentes parâmetro de entrada ϵ.

Proposição 4.4.1 *x^* é uma solução eficiente para o problema multiobjetivo se, e somente se, x^* resolve $P_k(x^*)$, com $\epsilon_j^* = f_j(x^*), j \neq k$, para todo $k = 1, ..., p$.*

Demonstração

($\Rightarrow$) Seja x^* uma solução eficiente. Por definição, não existe $x \in X$ tal que $f(x) \leq f(x^*)$ e $f(x) \neq f(x^*)$. Logo, x^* resolve $P_k(\epsilon^*)$, com $\epsilon_j^* = f_j(x^*), j \neq k$, para todo $k = 1, ..., p$.

($\Leftarrow$) Suponha que x^* resolve $P_k(\epsilon^*)$, onde $\epsilon_j^* = f_j(x^*), j \neq k$, para todo $k = 1, ..., p$. Por contradição, suponha que x^* é inferior. Logo, deve existir algum ponto $x \in X$ tal que $f(x) \leq f(x^*)$ e, para algum k, $f_k(x) < f_k(x^*)$, concluímos que x é solução viável para $P_k(\epsilon^*)$. Como $f_k(x) < f_k(x^*)$, x^*) não resolve $P_k(\epsilon^*)$. Absurdo.

■

Definição 4.4.1 ***[Condições de KKT para** $P_k(\epsilon)$ (KKT_ϵ)**]:*** *Seja $f_j(x), j = 1, ..., p$, e $g_i(x), i = 1, ..., m$, funções continuamente diferenciáveis. Se $x \in X \cap x \mid f_j(x) \leq \epsilon_j, j = 1, ..., p, j \neq k$ é um mínimo local de $P_k(\epsilon)$ para um determinado k e algum $\epsilon \in \mathbb{R}^{p-1}$ e vale alguma condição de qualificação em x, então existem multiplicadores $\lambda_j \geq 0, j = 1, ..., p, j \neq k$, e $\mu_i \geq 0, i = 1, ..., m$ tais que*

$$f_k(x) + \sum_{j \neq k} \lambda_j \nabla f_j(x) + \sum_{i=1}^{m} \mu_i \nabla g_i(x) = 0, \tag{4.5}$$

$$\lambda_j (f_j(x) - \epsilon_j) = 0, j = 1, ..., p, j \neq k, \tag{4.6}$$

$$\mu_i g_i(x) = 0, i = 1, ..., m. \tag{4.7}$$

Proposição 4.4.2 *Suponha que x^* é uma solução propriamente eficiente do problema de otimização multiobjetivo original. Seja $k \in \{1, ..., p\}$. Suponha que os gradientes das restrições de $P_k(\epsilon^*)$, com $\epsilon_j^* = f_j(x^*), j \neq k$, ativas em x^* são linearmente independentes e que todas as funções f_i e g_i são continuamente diferenciáveis. Então, x^* resolve $P_k(\epsilon^*)$ e satisfaz as condições de KKT_ϵ com $(\lambda, \mu) \geq 0$ para $P_k(\epsilon^*)$. Além disso, todos os multiplicadores de Lagrange associados às restrições $f_j(x) \leq \epsilon_j, j \neq k$, de $P_k(\epsilon^*)$ são estritamente positivos.*

Demonstração

Suponha que x^* é uma solução propriamente eficiente. Como x^* é eficiente, pela Proposição 5.4.1, x^* resolve $P_k(\epsilon^*)$, com $\epsilon_j^* = f)j(x^*), j \neq k$, para todo $k = 1, ..., p$. Seja $k \in \{1, ..., p\}$. Supondo que os gradientes das restrições de $P_k(\epsilon^*)$ ativas em x^* são linearmente independentes e que todas as funções f_i e g_i são continuamente diferenciáveis, garante-se a existência de um conjunto de multiplicadores de Lagrange $\lambda_{kj}^* \geq 0, j \neq k$, e $\mu_i^* \geq 0$, para todo $i = 1, ..., m$, tal que as condições KKT_ϵ para $P_k(\epsilon^*)$ sejam satisfeitas. Devemos mostrar que λ_{kj}^* deve ser estritamente positivo para todo $j \neq k$. Utilizando o resultado de um teorea em [Geoffrion, 1968](*Comprehensive Theorem)*, temos que, pela eficiência própria e propriedade de regularidade de x^*, existem $\lambda_j^0 > 0, j = 1, ..., p$ e $\mu_i^0 \geq 0, i = 1, ..., m$, tal que

$$\textstyle\sum_{j=1}^{p} \lambda_j^0 \nabla f_j(x^*) + \sum_{i=1}^{m} \mu_i^0 \nabla g_i(x^*) = 0$$

$$\mu_i^0 g_i(x^*) = 0, i = 1, ..., m,$$

ou, equivalente,

$$\nabla f_k(x^*) + \sum_{j \neq k} (\frac{\lambda_j^0}{\lambda_k^0}) \nabla f_j(x^*) + \sum_{i=1}^{m} (\frac{\mu_i^0}{\mu_k^0}) \nabla g_i(x^*) = 0, \tag{4.8}$$

$$(\frac{\mu_i^0}{\lambda_k^0}) g_i(x^*) = 0, i = 1, ..., m.$$

Como $\nabla f_j(x^*), j \neq k$, e $\nabla g_i(x^*) = 0$, são linearmente independentes, temos que (5.8) possuirá solução única em termos de $\frac{\lambda_j^0}{\lambda_k^0}, j \neq k$, e $\frac{\mu_i^0}{\mu_k^0}, i = 1, ..., m$. Combinando com isso com o fato de que (5.8) é nada mais que uma das condições de KKT_ϵ para $P_k(\epsilon^*)$, a qual é satisfeita por $\lambda_{kj}^*, j \neq k$, e $\mu_i^*, i = 1, ..., m$, temos que $\lambda_{kj}^* = \frac{\lambda_j^0}{\lambda_k^0} > 0$ para todo $j \neq k$.

■

Capítulo 5

Um algoritmo para problemas de programação linear inteira multiobjetivo

5.1 Introdução

Dado uma função vetorial $z(x) = (z_1(x), z_2(x), ..., z_p(x))^T$, onde $z_i(x)$ é uma função escalar e o conjunto de pontos factível $X \subset \mathbb{R}$, o problema multiobjetivo:

$$(G)\ max\ z(x)$$

$$s.a.\ x \in X,$$

o problema é achar todos os pontos não-dominados (eficientes). O ponto x é dito não-dominado com respeito a z e X se $x \in X$ e não existe nenhum $y \in X$ tal que $z(y) \geq z(x)$ com $z(y) \neq z(x)$. Analogamente, um ponto $z(x)$ no espaço objetivo é dito não-dominado com respeito a X se o ponto x é não-dominado. Quando X é uma coleção restrita de pontos inteiros por inequações lineares e $z_1(x), z_1(x), ..., z_p(x)$ são funções lineares, o problema de achar todos os pontos pontos não-dominados é formulado da seguinte maneira:

$$(P)\ max\ z(x) = Cx$$

$$s.a.\ Ax \leq b$$

$$x \geq 0,\ inteiro$$

onde $C = (c^1, c^2, ..., c^p)^T$ é uma matriz de coeficientes da função objetivo.

5.2 O algoritmo

O problema que abordaremos é,

$$max(z_1(x), z_2(x))$$

$$s.a.\ x \in X$$

Onde contém a estipulação que x seja inteiro e $z_1(x)$ e $z_2(x)$ assumam apenas valores inteiros para $x \in X$.

No que se segue, os subscritos indicam componentes de um vetor, sobrescrito denotam vetores, enquanto sobrescrito entre parênteses indica o contador de interação. O sinal

":="denota *torna-se*. O algoritmo cria quatro conjuntos à medida que progride: E contém pares adjacentes de pontos não-dominados no espaço objetivo; N contém os pontos não-dominados no espaço de decisão; Z contém os pontos não-dominados no espaço objetivo; e Q contém candidatos a pares adjacentes de pontos não-dominados no espaço objetivo.

Passo 0.

Deixe $E = \emptyset$.

Deixe $z_2^{(1)} = max\{z_2(x)|x \in X\}$. Se o problema é inviável, pare; o problema bi-critério é inviável. Caso contrário, deixe $z_1^{(1)} = max\{z_1(x)|x \in X, z_2(x) \geq z_2^{(1)}\}$ com solução $x^{(1)}$. Deixe $N = \{x^{(1)}\}$ e $Z = \{z^{(1)}\}$.

Deixe $z_1^{(2)} = max\{z_1(x)|x \in X\}$. Se $z_1^{(1)} = z_1^{(2)}$, pare; um ponto ideal viável, $x^{(1)}$, foi encontrado. Caso contrário, deixe $z_2^{(2)} = max\{z_2(x)|x \in X, z_1(x) \geq z_1^{(2)}\}$ com solução $x^{(2)}$. Deixe $N := N \cup \{x^{(2)}\}$, $Z := Z \cup \{z^{(2)}\}$ e $Q = \{(z^{(1)}, z^{(2)})\}$. Deixe $t = 3$.

Passo 1.

Se $Q = \emptyset$, pare; todos os pontos não-dominados no espaço objetivo foram encontrados. Caso contrário, selecionar $p = (z^{(r)}, z^{(s)}) \in Q$ e definir $Q := Q - \{p\}$. Escolher $\lambda_1^{(t)} > 0$ e $\lambda_2^{(t)} > 0$. Deixe $\hat{z}_1 = min\{z_1^{(r)}, z_1^{(s)}\}$ e $\hat{z}_2 = min\{z_2^{(r)}, z_2^{(s)}\}$.

Passo 2.

Resolver subproblema

$$(P_t) max \lambda_1^{(t)} z_1(x) + \lambda_2^{(t)} z_2(x),$$

$$s.a.\ \hat{z}_i \geq \hat{z}_1 + 1,\ i = 1, 2.$$

$$x \in X$$

Se (P_t) é inviável, deixe $E := E \cup \{p\}$ e ir para ***Passo*** 1. Caso contrário, deixe $x^{(t)}$ ser uma solução, e deixe $z_1^{(t)} = z_1(x^{(t)})$ e $z_2^{(t)} = z_2(x^{(t)})$. Deixe $N := N \cup \{x^{(t)}\}$, $Z := Z \cup \{z^{(t)}\}$, $Q := Q \cup \{(z^{(t)}, z^{(s)})\} \cup \{(z^{(r)}, z^{(t)}\}$. Deixe $t := t + 1$, e ir para ***Passo*** 1.

Conclusão

Através dos estudos desenvolvidos, as metas previstas para o projeto foram todas alcançadas. Inicialmente foi feito o desenvolvimento teórico para problemas de Otimização Matemática.

Mais adiante ainda na primeira etapa do projeto foram exploradas as principais propriedades relacionadas a Programação Linear Inteira. E por fim a segunda etapa, na qual foi levantado dados a respeito dos dois métodos apresentados no capítulo 5 a respeito de programação linear multiobjetivo, afim da melhor compreensão relacionada ao assunto. Daí, o foco final foi o artigo apresentado no último capítulo como um método de programação linear inteira multiobjetivo para casos específicos.

Na programação linear inteira mono-objetivo ficou claro o comportamento da função objetivo com as restrições inteiras. O método dos pesos no capítulo foi essencial para a compreensão mais madura de problemas de programação linear multiobjetivo. O método da ϵ-Restrição deixou mais claro a questão de como tratar cada problema e como caracterizar.

Com o artigo foi interessante o tratamento de dados apresentados nele e a ideia que os autores tiveram para desenvolver o algoritmo forma tão simples e eficiente, para os casos apresentados.

Bibliografia

[1] GOLDBARG, Marco Cesar; LUNA, Henrique Pacca Loureiro. Otimização Combinatória e Programação Linear: Modelos e Algoritmos. 5ª ed. Rio de Janeiro: Editora Elsevier, 2005.

[2] BRONSON, Richard; NAADIMUTHU, Govindasami. Operations Research. 2ªed. United States of America: Editora McGraw-Hill, 1997.

[3] MACULAN, Nelson; FAMPA, Marcia H. Costa. Otimização Linear. 1ªed. Rio de Janeiro: Editora UFRJ, 2004.

[4] EHRGOTT, Matthias. Multicriteria Optimization. 2ªed. Germany: Editora Heidelberg, 2005.

[5] CHALMET, L.G.; LEMONIDIS, L.; ELZINGA, D.J.. An algorithm for the bi-criterion integer programming problem. North-Holland: European Journal of Operational Research, 1986.

[6] GEOFFRION, Arthur M.. Proper Efficiency and the Theory of Vector Maximization. Belgium: Journal of Mathematical Analysis and applications, 1968.

[7] SAMPAIO, Phillipe Rodrigues. Teoria, métodos e aplicações de otimização multiobjetivo. São Paulo: Dissertação de mestrado - USP, 2011.