Pesquisa Operacional Meruru

by RTR

Modelo matemático de logística no registro de cartas da Meruru

São Paulo

2025

César Martins 03231029

**Introdução à pesquisa operacional**

Uma das técnicas mais utilizadas na abordagem de problemas em P.O. (pesquisa operacional) é a programação linear. As aplicações mais conhecidas são feitas em sistemas estruturados, como os de produção, finanças, controle de estoques etc. O modelo matemático de programação linear é composto de uma função objetivo linear, e de restrições técnicas representadas por um grupo de inequações também lineares.

**Desafio do cliente Meruru**

Meruru e uma loja dentre muitas no mercado de TCG (Trading Card Game), consiste na venda de cartas colecionáveis e derivados produtos. Na logística desse negócio existe o processo do cadastro de cartas no estoque, algo que exige conhecimento da área, e algo muito valioso, tempo. Esse é o principal gargalo na operação, quando é adquirido um carregamento do grande de cartas...Às vezes o registro de uma carta manualmente pode levar até 5 minutos!

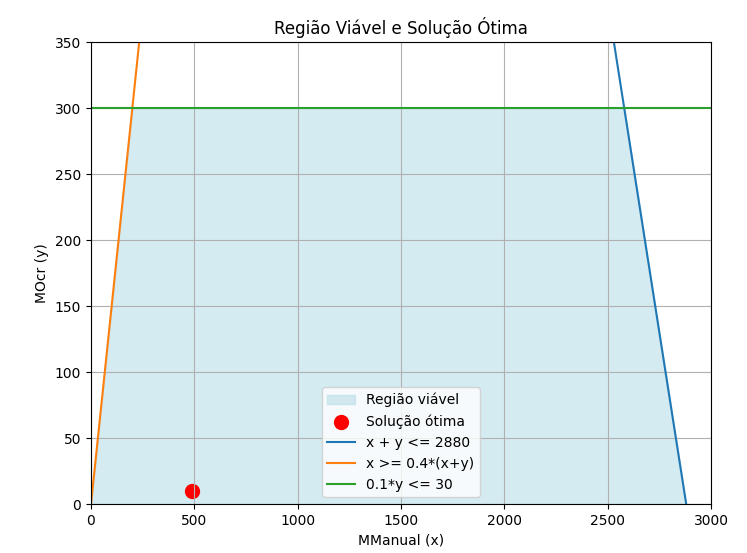
**Solução RTR**

Com olhar de negócio analítico, a RTR traz uma solução direta, que através de tecnologia, os operadores conseguirão reduzir o tempo desse processo para segundos (10 segundos em média). Por questões lógicas o método de OCR (Optical Character Reading) da RTR é o ideal para o objetivo de minimização do uso de tempo nos cadastros. Porém, qual tem o melhor custo-benefício? Aqui será uma maneira típica de aplicar o uso da programação linear, encontrar a minimização dos custos ao usar os dois métodos mutualmente!

**Modelo RTR em programação linear**

* Variáveis de decisão:
  + X1 = carta registrada método manual
  + X2 = carta registrada método OCR
* Objetivo:
  + Minimizar o custo, que pode ser calculado:
    - Custo devido manual = 0,54 \*x1 (5 minutos de salário-mínimo CLT\*carta registrada manualmente)
    - Custo devido OCR = 0,10\*x2 (custo por registro OCR com infra geral\*carta registrada com OCR)
    - Custo total = C = 0,54\*x1+0,10\*x2
    - Objetivo = Min C = 0,54\*x1+0,10\*x2
* Restrições:
  + Média de cartas para registro no dia 500! X1+X2 >= 500
  + Em 8 horas de expediente o máximo de registros possíveis é de 2.880 cartas! X1 + X2 <= 2880
  + 40% das cartas são holográficas e especiais que o OCR não lê com facilidade, então o método manual é necessário! X1 >= 0,4\*(X1+X2)
  + Há um orçamento de 30 reais para serem gastos com AWS Services por dia! 0,10\*X2 <= 30
  + Não negativação! X1>0, X2>=0

Solução Gráfica:



Fica claro que o uso manual é mais barato e compensa em questões economicas, no entanto, tempo muitas vezes é mais valioso que dinheiro!

Script Python para a solução do problema com exibição gráfica no final.

# Instalar Pulp, PuLP é um framework para modelagem e resolução de problemas

import subprocess

import sys

subprocess.check\_call([sys.executable, "-m", "pip", "install", "pulp"])

from pulp import \*

# Importar gráfico

subprocess.check\_call([sys.executable, "-m", "pip", "install", "matplotlib"])

subprocess.check\_call([sys.executable, "-m", "pip", "install", "numpy"])

import matplotlib.pyplot as plt

import numpy as np

# Variáveis

x = LpVariable("MManual", cat="Continuous", lowBound=1, upBound=10)

y = LpVariable("MOcr", cat="Continuous", lowBound=1, upBound=10)

# Problemas Lineares

prob = LpProblem("Minimização de custos")

# Objetivo e restrições

prob += 0.54\*x + 0.10\*y # função objetivo

prob += x+y >= 500

prob += x+y <= 2880

prob += x >= 0.4\*(x+y)

prob += 0.1\*y <= 30

# Resolver o modelo utilizando o método solve()

status = prob.solve()

# Mostrar o status da solução

LpStatus[status]

# Mostrar as variáveis e seus valores no meio do Prob

for var in prob.variables():

print("{} = {}".format(var.name, var.varValue))

print("custo min em BRL = ", pulp.value(prob.objective))

# --------- Criar gráfico ---------

x\_val = x.varValue

y\_val = y.varValue

custo = value(prob.objective)

# --- Plotando a região viável ---

x\_range = np.linspace(0, 3000, 500)

# Restrições

y1 = 2880 - x\_range # x + y <= 2880

y2 = (3/2)\*x\_range # x >= 0.4(x+y) -> y <= 1.5x

y3 = 300 \* np.ones\_like(x\_range) # 0.1\*y <= 30 -> y <= 300

plt.figure(figsize=(8,6))

# Região viável (interseção das restrições)

plt.fill\_between(x\_range, 0, np.minimum(np.minimum(y1, y2), y3), color='lightblue', alpha=0.5, label='Região viável')

# Ponto ótimo

plt.scatter(x\_val, y\_val, color='red', s=100, label='Solução ótima')

# Linhas das restrições

plt.plot(x\_range, y1, label='x + y <= 2880')

plt.plot(x\_range, y2, label='x >= 0.4\*(x+y)')

plt.plot(x\_range, y3, label='0.1\*y <= 30')

plt.xlim(0, 3000)

plt.ylim(0, 350)

plt.xlabel('MManual (x)')

plt.ylabel('MOcr (y)')

plt.title('Região Viável e Solução Ótima')

plt.legend()

plt.grid(True)

plt.show()