UNIVERSIDADE FEDERAL DE SÃO PAULO
INSTITUTO DE CIÊNCIA E TECNOLOGIA

DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA

# Otimização Linear aplicada a Problemas de Escala de Equipes de Enfermagem

Magno Gonçalves Fonseca

Orientador: Prof. Dr. Thadeu Alves Senne

São José dos Campos, SP

2023

# Sumário

# 1 Resumo

Enfermeiras e enfermeiros desempenham um papel vital no funcionamento eficiente de um hospital, oferecendo cuidados diretos aos pacientes e desempenhando diversas tarefas essenciais. A gestão eficiente do escalonamento de enfermeiras é fundamental para garantir a qualidade do atendimento, otimizar os recursos humanos e reduzir os custos operacionais. Porém, quando essa gestão é feita de forma manual, podemos ter problemas de eficiência, confiabilidade e custo. O presente projeto visa o estudo do uso de Programação Linear para obter uma solução ótima de um problema de gestão de turnos, com o objetivo de minimizar o custo da operação.

Com a formulação adequada e o uso de algoritmos de otimização, é possível obter soluções que atendam às restrições estabelecidas, maximizando a eficiência do escalonamento e contribuindo para a melhoria do atendimento aos pacientes e a redução de custos operacionais.

# 2 Introdução e Justificativa

O trabalho das enfermeiras em um hospital é de extrema importância para o funcionamento eficiente e eficaz da instituição de saúde. Os enfermeiros desempenham um papel vital na prestação de cuidados diretos aos pacientes, monitorando sua condição, administrando medicamentos, realizando procedimentos e oferecendo suporte emocional. A gestão eficiente do escalonamento de enfermeiras em um hospital é fundamental para garantir a qualidade do atendimento aos pacientes, otimizar a alocação de recursos humanos e minimizar custos operacionais.[1] Tendo isso em mente, este projeto busca a solução computacional de um problema que envolve a elaboração de uma escala de trabalho para cada enfermeira, considerando turnos de serviço e dias de folga em um período de planejamento de curto prazo.

Problemas de escalonamento de enfermeiras têm sido amplamente investigados nas últimas décadas. Devido à crescente demanda por recursos de saúde, a Organização Mundial da Saúde (OMS) enfatiza que a gestão eficiente e utilização dos recursos humanos serão uma das principais prioridades para a indústria da saúde nas próximas décadas. Além disso, os hospitais estão sob pressão crescente para reduzir os custos com enfermagem, uma vez que essas despesas representam mais de $40\%$ do seu orçamento total.[2] Nesse sentido, fornecer um escalonamento eficiente e personalizado para as enfermeiras pode ajudar a mitigar custos desnecessários.

Nos últimos anos, foram propostos e aplicados diferentes métodos e metodologias para lidar com os Problemas de Escala de Enfermeiros (PEE). A Programação Matemática (PM), métodos heurísticos e a Inteligência Artificial (IA) são três abordagens comuns nessa área.[4] A MP é uma abordagem tradicional que busca soluções ótimas para os problemas de escalas de enfermeiros (PEE). Venkataraman e Brusco [3] abordaram o problema de dimensionamento de equipes de enfermagem por meio de um modelo de Programação Linear Inteira Mista (PLIM), visando determinar as necessidades globais de mão de obra para um período de planejamento de seis meses e dividir a escala em períodos de duas semanas. Os autores também introduziram um sistema integrado de escalas e dimensionamento de enfermeiros para avaliar as políticas de gestão da força de trabalho de enfermagem. Através desse sistema, eles analisaram os impactos das estratégias de escalas e dimensionamento nos custos de enfermagem, e os resultados obtidos revelaram relações significativas entre essas estratégias.

Em outro estudo semelhante, Mischek e Musliu [5] desenvolveram e avaliaram várias extensões de formulações padrão de Programação Inteira para problemas de escalas de enfermeiros, lidando com configurações de múltiplas etapas. As restrições flexíveis consideradas em seu estudo incluem falta de pessoal para cobertura ideal, número de turnos consecutivos e dias de folga, número de fins de semana de trabalho possíveis, entre outros.

Programação Linear permite modelar as restrições e objetivos do problema em um sistema de equações e desigualdades lineares, buscando encontrar a melhor alocação de enfermeiras para os diferentes turnos de trabalho. Por meio de uma formulação matemática e do uso de algoritmos de otimização, é possível encontrar soluções eficientes que atendam aos requisitos do problema, as restrições de turnos e ao objetivo de minimizar custos operacionais desnecessários.[6]

Além do estudo de soluções para o problema proposto, este projeto também inclui a construção de ferramentas visuais para que usuários comuns possam manipular dados de entrada do problema e navegar em uma ferramenta de otimização de turnos acessível e intuitiva. Para isso, será construída uma aplicação web escrita em C# usando o ASP.NET. Levando em consideração os requisitos de negócio envolvidos, um banco de dados Microsoft SQL server será modelado e os dados de entrada armazenados nele.

Por fim, a ideia é que a solução do problema seja entregue em uma planilha para que o gestor da equipe de enfermagem avalie a solução e faça os ajustes que achar conveniente.

# 3 Descrição do problema e formulação

## 3.1 Pressupostos

O modelo proposto aborda a alocação de enfermeiras nos turnos com os quais os hospitais disponibiliza, levando em consideração as políticas específicas que restringem essa atribuição. Cada hospital possui suas próprias regras em relação ao número máximo de turnos consecutivos permitidos, distribuição de turnos noturnos e concessão de férias aos funcionários. Essas políticas variam de um hospital para outro. No modelo matemático proposto, são definidos parâmetros e restrições que levam em conta essas políticas, garantindo a conformidade com as regras estabelecidas por cada instituição hospitalar.

Os pressupostos considerados neste estudo são listados abaixo:

- O horizonte de escalonamento abrange uma semana, de domingo a sábado.
- De acordo com as regulamentações do hospital, uma enfermeira não pode estar de plantão por mais de um turno em um único dia.
- Cada dia é dividido em três turnos de 8 horas: das 7h às 15h (turno da manhã), das 15h às 23h (turno da tarde) e das 23h às 7h (turno da noite).
- Cada enfermeira possui dois dias de folga, de acordo com uma distribuição que não desfalque o hospital.

## 3.2 Modelagem de restrições

Buscamos a solução do problema de gestão de turnos semanalmente, sugerindo uma agenda das escalas. Cada turno possui seus atributos, como demonstrado na tabela abaixo, onde temos exemplos de turnos segunda e terça feira:

Para saber se o funcionário $e$ estará escalado no turno $t$, definimos uma matriz $R$ de dimensões [$t$ x $m$] que assume valores binários para representar esses estados: $1$ para escalado e $0$ para não escalado. Observe o exemplo abaixo, onde vemos que os enfermeiros $e_1$ e $e_2$ e $e_m$ foram convocados a compor o turno $1$, $2$ e $2$ respectivamente:

| Id Turno | Departamento | Dia da Semana | Horario Inicio | Horario Fim | Minimo Pessoas | Maximo Pessoas | Duracao Turno |
|---|---|---|---|---|---|---|---|
| 1 | Emergencia | Segunda | 7 | 15 | 3 | 5 | 8 |
| 2 | Emergencia | Segunda | 15 | 23 | 4 | 7 | 8 |
| 3 | Emergencia | Segunda | 23 | 7 | 2 | 5 | 8 |
| 4 | Consultas | Segunda | 7 | 15 | 10 | 13 | 8 |
| 5 | Consultas | Segunda | 15 | 23 | 8 | 12 | 8 |
| 6 | Cardiologia | Segunda | 7 | 15 | 10 | 13 | 8 |
| 7 | Cardiologia | Segunda | 15 | 23 | 8 | 12 | 8 |
| 8 | Pediatria | Segunda | 7 | 15 | 2 | 4 | 8 |
| 9 | Emergencia | Terça | 7 | 15 | 4 | 7 | 8 |
| 10 | Emergencia | Terça | 15 | 23 | 2 | 5 | 8 |
| 11 | Emergencia | Terça | 23 | 7 | 3 | 7 | 8 |
| 12 | Consultas | Terça | 7 | 15 | 10 | 13 | 8 |
| 13 | Consultas | Terça | 15 | 23 | 8 | 12 | 8 |
| 14 | Cardiologia | Terça | 7 | 15 | 4 | 7 | 8 |
| 15 | Cardiologia | Terça | 15 | 23 | 2 | 5 | 8 |
| 16 | Cardiac Care | Terça | 18 | 2 | 3 | 7 | 8 |
| 17 | Pediatria | Terça | 15 | 23 | 2 | 4 | 8 |

Figura 1: Exemplo de turnos segunda e terça-feira

$$R = \begin{bmatrix} e_{1_1} & e_{1_2} & ... & e_{1_t} \\ e_{2_1} & e_{2_2} & ... & e_{2_t} \\ ... & ... & ... & ... \\ e_{m_1} & e_{m_2} & ... & e_{m_t} \end{bmatrix} = \begin{bmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & ... & 0 \\ ... & ... & ... & ... \\ 0 & 1 & ... & 0 \end{bmatrix}$$

Note pelo exemplo da Figura 1 que é impossível o mesmo enfermeiro ocupar os turnos $1$ e $4$, pois ocorrem ao mesmo tempo em departamentos diferentes. Portanto, temos que:

$$\begin{bmatrix} e_{1_1} + e_{1_4} \\ e_{2_1} + e_{3_4} \\ ... \\ e_{m_{t_a}} + e_{t_b} \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix}.$$

Essa condição deve ocorrer para todos os pares de turnos incompatíveis $t_a$ e $t_b$.

Em um hospital, temos turnos todos os dias da semana, inclusive sábados e domingos. Porém, cada enfermeira deve possuir 2 dias de folga.Matematicamente, temos:

$$R. \begin{bmatrix} 1 & 1 & ... & 1 \end{bmatrix}^T = \begin{bmatrix} e_{1_1} & e_{1_2} & ... & e_{1_t} \\ e_{2_1} & e_{2_2} & ... & e_{2_t} \\ ... & ... & ... & ... \\ e_{m_1} & e_{m_2} & ... & e_{m_t} \end{bmatrix} . \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ ... \\ 5 \end{bmatrix}.$$

Agora, para evitar a escalação de dois turnos no mesmo dia, mapeamos os turnos que ocorrem no

mesmo dia, ou seja, para o exemplo de segunda e terça, temos:

$$\begin{bmatrix} e_{1_1} & e_{1_2} & e_{1_4} & e_{1_5} & e_{1_6} & e_{1_7} & e_{1_8} & 0 & 0 \\ e_{2_1} & e_{2_2} & e_{2_4} & e_{2_5} & e_{2_6} & e_{2_7} & e_{2_8} & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & & \\ e_{1_9} & e_{1_{10}} & e_{1_{11}} & e_{1_{12}} & e_{1_{13}} & e_{1_{14}} & e_{1_{15}} & e_{1_{16}} & e_{1_{17}} \\ e_{2_9} & e_{2_{10}} & e_{2_{11}} & e_{2_{12}} & e_{2_{13}} & e_{2_{14}} & e_{2_{15}} & e_{2_{16}} & e_{2_{17}} \end{bmatrix} . \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \\ 1 \end{bmatrix},$$

havendo uma linha para representar cada enfermeiro nos turnos do dia.

Note que cada turno possui uma demanda mínima e máxima de funcionários, que são armazenadas nos vetores $P_{min}$ e $P_{max}$ com a mesma dimensão de $t$, que contém o identificador do turno. Dessa forma, nota-se que:

$$P_{min} \leq r^T . \begin{bmatrix} 1 & 1 & ... & 1 \end{bmatrix}^T \leq P_{max}$$

$$\begin{bmatrix} 3 \\ 4 \\ ... \\ n_{P_{min}} \end{bmatrix} \leq \begin{bmatrix} e_{1_1} & e_{2_1} & ... & e_{m_1} \\ e_{1_2} & e_{2_2} & ... & e_{m_2} \\ ... & ... & ... & ... \\ e_{1_t} & e_{2_t} & ... & e_{m_t} \end{bmatrix} . \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 7 \\ ... \\ n_{P_{max}} \end{bmatrix}.$$

Semanalmente, um funcionário deve trabalhar no máximo 40h. Portanto:

$$(R.D_{Turno})^T j = \left( \begin{bmatrix} e_{1_1} & e_{1_2} & ... & e_{1_t} \\ e_{2_1} & e_{2_2} & ... & e_{2_t} \\ ... & ... & ... & ... \\ e_{m_1} & e_{m_2} & ... & e_{m_t} \end{bmatrix} . \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix} \right)^T \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix} \leq \begin{bmatrix} 40 \\ 40 \\ ... \\ 40 \end{bmatrix}$$

sendo $j$ uma matriz coluna preenchida com 1 da mesma dimensão de linhas que a matriz $r$ e $D_{Turno}$ a matriz que armazena a duração dos turnos em sua diagonal principal.

## 3.3 Função Objetivo

Este projeto pretende estudar a minimização de custos, supondo que os funcionários recebem por hora trabalhada (Cada um de acordo com seu nível de senioridade). Para calcular o salário semanal, precisamos obter a duração de cada turno $D_{Turno}$ dos funcionários e multiplicar pelo seu salário por hora $S_e$ e, por fim, devemos somar os salários para obter o custo total da operação. A expressão

matricial abaixo demonstra esse cálculo:

$$(R.D_{Turno})^T S_e = \left( \begin{bmatrix} e_{1_1} & e_{1_2} & ... & e_{1_t} \\ e_{2_1} & e_{2_2} & ... & e_{2_t} \\ ... & ... & ... & ... \\ e_{m_1} & e_{m_2} & ... & e_{m_t} \end{bmatrix} . \begin{bmatrix} 6 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & 0 & n \end{bmatrix} \right)^T \begin{bmatrix} 34 & 0 & 0 & 0 \\ 0 & 32 & 0 & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & 0 & n \end{bmatrix} = C$$

Afim de somar todos os items da matriz de custos $C$, faremos:

$$\left( C. \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix} \right)^T . \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix} = T$$

Sendo $T$ o custo total da operação. Por fim, ficamos com a expressão:

$$Minimize$$
$$[((r.D_{Turno})^T S_e).k]^T k = T$$

sendo $k$ uma matriz coluna preechida com $1$ da mesma dimensão de linhas que a Matriz de Custo

# 4 Metodologia

Toda a solução computacional do problema será construída em Python, nele, importaremos uma biblioteca chamada Pulp, que carrega rotinas de Programação Linear. A Figura 2 mostra um pequeno exemplo de como usar essa ferramenta:

```python
from pulp import *

# Criação do problema
prob = LpProblem("Exemplo de Programação Linear", LpMinimize)

# Variáveis de decisão
x = LpVariable("x", lowBound=0)
y = LpVariable("y", lowBound=0)

# Função objetivo
prob += 3 * x + 5 * y

# Restrições
prob += x + 2 * y >= 10
prob += 4 * x + 3 * y >= 20

# Resolução do problema
prob.solve()

# Status da solução
print("Status:", LpStatus[prob.status])

# Valor ótimo das variáveis de decisão
print("Valor ótimo de x:", value(x))
print("Valor ótimo de y:", value(y))

# Valor ótimo da função objetivo
print("Valor ótimo da função objetivo:", value(prob.objective))
```

Figura 2: Exemplo de Programação Linear com Pulp

A função *solve()* da biblioteca PuLP utiliza diferentes métodos matemáticos dependendo do tipo de problema de otimização sendo resolvido. A escolha do método é automatizada e feita internamente pela biblioteca com base nas características do problema. Para problemas de Programação Linear Inteira (ILP), a biblioteca PuLP utiliza dois principais solvers externos, como o CBC (Coin-or branch and cut) e o GLPK (GNU Linear Programming Kit), que implementam uma variedade de algoritmos para resolver problemas de programação linear inteira, incluindo o método de Ramificação e Poda (Branch and Bound). :

- **CBC**: Utiliza o método *Branch and Bound*, que é uma técnica utilizada para resolver problemas de Programação Linear Inteira. Ele consiste em dividir o problema em subproblemas menores e mais fáceis de resolver, por meio da ramificação. Inicialmente, o problema é relaxado para uma forma de Programação Linear, e uma solução ótima é encontrada. Em seguida, são criadas ramificações do problema original, adicionando restrições inteiras, e os subproblemas são resolvidos. Um critério de poda é utilizado para eliminar subproblemas não promissores. O método

continua ramificando e podando até encontrar a solução ótima ou explorar todas as soluções viáveis.[8] [7]

- **GNU Linear Programming Kit**: Um solver também *open-source* que utiliza o método Simplex, que é um algoritmo clássico para resolver problemas de Programação Linear. Ele busca iterativamente uma solução ótima movendo-se de um vértice a outro na região viável do problema, até encontrar a solução ótima ou determinar que o problema é ilimitado ou inviável, além de também usar o método de *Branch and Bound* como no solver CBC. [9]

## 4.1 Forma de análise de resultados

Ao implementar o PEE utilizando o Python, o primeiro passo é fazer valer a solução do problema apenas com as restrições horárias, como mostra a formulação demonstrada na seção 3. Para isso, é preciso criar cenários fictícios para validar o modelo simplificado, ou seja, listar uma quantidade de enfermeiras, turnos, salários e etc.

Após o modelo simplificado ser validado, serão acrescentadas novas regras na modelagem. Regras que classificam as enfermeiras por habilidades e senioridade, fazendo com que apenas pessoas com certos talentos ou experiências sejam escaladas para turnos com a devida classificação. Isso traz mais critérios que tornam o PEE mais completo e assertivo.

Para as avaliações citadas, os resultados obtidos serão comparados com os resultados encontrados na literatura ou obtidos em publicações anteriores sobre o tema.

# 5 Objetivos

O objetivo principal deste trabalho é compreender a modelagem matemática dos problemas de escalonamento de enfermeiros, bem como realizar a sua implementação computacional, analisar as escalas obtidas por cada modelo, o simplificado (apenas restrições horárias) e o completo (adicionando restrições de habilidade e experiência), e identificar, se possível, as vantagens e desvantagens do uso de cada uma das formulações.

# Referências


[1] Xiao, Y., et al. (2020). The impact of nurse scheduling on patient safety, nurse outcomes, and patient outcomes: A systematic review. Journal of Advanced Nursing, 76(5), 1051-1063.

[2] Zurn P, Dolea C and Stilwell B. Nurse retention and recruitment: developing a motivated workforce (The Global Nursing Review Initiative, Issue paper no. 4). Geneva: WHO, 2005.

[3] Venkataraman R and Brusco M. An integrated analysis of nurse staffing and scheduling policies. Omega 1996; 24: 57–71.

[4] M Hamid, R.T Moghaddam, , FGolpaygani1 and B.V Nouri. A multi-objective model for a nurse scheduling problem by emphasizing human factors. Jounal of Engineering in Medicine.

[5] Mischek F and Musliu N. Integer programming model extensions for a multi-stage nurse rostering problem. Ann Oper Res 2019; 275: 123–143

[6] C.E. Savillea, P. Griffithsb, J.E. Ballc, T. Monksc.How many nurses do we need? A review and discussion of operational research techniques applied to nurse staffing. International Journal of Nursing Studies

[7] Wolsey, L. A. (1998). Integer programming. John Wiley & Sons.

[8] J. Forrest. The COIN-OR Branch and Cut solver (CBC) Documentation, disponible in https://coin-or.github.io/Cbc/intro.html.

[9] GNU Linear Programming Kit | Modeling Language GNU MathProg DOcumentation, disponible in https://www.ime.unicamp.br/ moretti/ms428/1sem2006/lang.pdf.