# PROJETO PRÁTICO – Laboratório de Programação Competitiva

Momento N3 – 2024.2 - Profa. Silvia Brandão

Data de entrega: 27.11 Apresentação28.11 (MostraTec)

**Formação dos grupos:**

Deve ser realizado em grupo de no máximo 4 alunos.

**Sistema de Avaliação:**

20 pontos distribuídos na avaliação do Momento 3.

**O que será o projeto?**

O uso de grafos deve andar lado a lado com a parte de produção tecnológica. Contudo, devemos nos manter fiel à característica fundamental de manipulação da estrutura de dados escolhida para representação e manipulação dele.

O aluno deverá realizar a documentação escrita do projeto com ilustrações, citando os passos necessários para a execução dele, assim como a implementação em Python.

## Sugestões de casos e propostas de soluções:

Aqui estão sete projetos diferentes de desenvolvimento em Python, que aplicam teorias de grafos e geometria computacional em contextos reais, como malhas de transporte, redes de serviços e otimização de recursos, dentre outros:

**1. Malha de Transporte Aéreo: Otimização de Rotas Aéreas**

Descrição:

Este projeto visa modelar a malha de transporte aéreo de um país ou região, utilizando grafos para representar aeroportos como nós e as rotas aéreas como arestas. O peso das arestas pode ser definido pela distância entre os aeroportos ou, alternativamente, pelo tempo de voo, custo ou consumo de combustível.

Objetivos:

- Modelar os aeroportos e rotas aéreas como um grafo ponderado.
- Implementar um algoritmo de busca para encontrar a rota mais curta (ex: Dijkstra) ou a rota com o menor custo (ex: A*).
- Considerar escalabilidade do problema, por exemplo, a partir de um grande conjunto de aeroportos e rotas.

Aplicações:

- Planejamento de rotas aéreas comerciais e de carga.
- Identificação de rotas mais econômicas e eficientes.
- Gestão de atrasos e redirecionamento de voos.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Geopy para cálculo de distâncias geográficas, Matplotlib para visualização).

**2. Instalação de Rede de Internet Fibra Ótica**

Descrição:

Este projeto visa otimizar a instalação de uma rede de internet fibra ótica em uma cidade ou região, considerando a minimização dos custos de infraestrutura, por exemplo, com o uso de grafos para modelar as ruas como arestas e as interseções ou prédios como nós.

Objetivos:

- Criar um grafo representando a cidade, onde cada nó é um ponto de conexão (interseções de ruas ou prédios) e as arestas são as ruas (com custos proporcionais ao comprimento da rua).
- Usar algoritmos de árvore geradora mínima (Kruskal ou Prim) para minimizar o custo de instalação da fibra ótica.
- Implementar um modelo para determinar quais bairros ou regiões da cidade devem ser priorizados para otimização.

Aplicações:

- Planejamento de redes de telecomunicações em cidades inteligentes.
- Redução de custos e melhoria na distribuição de sinal de internet.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Shapely e Geopandas para manipulação de dados geoespaciais, Matplotlib para visualização).

**3. Coleta de Lixo: Roteamento de Caminhões de Coleta**

Descrição:

O objetivo deste projeto é otimizar o roteamento de caminhões de coleta de lixo em uma cidade. A cidade é representada por um grafo, onde os nós representam os pontos de coleta e as arestas representam as ruas que conectam esses pontos.

Objetivos:

- Representar a cidade como um grafo ponderado, com distâncias ou tempo de viagem nas arestas.
- Implementar algoritmos de otimização, como o Problema do Caixeiro Viajante (TSP), para minimizar a distância percorrida pelos caminhões de coleta.
- Considerar várias restrições, como horários de pico, capacidade dos caminhões, e a quantidade de lixo em cada ponto de coleta.

Aplicações:

- Otimização dos custos operacionais e eficiência da coleta de lixo.
- Planejamento logístico de frotas de caminhões.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Google OR-Tools para otimização, Matplotlib para visualização).

**4. Rede de Abastecimento de Água: Otimização da Distribuição**

Descrição:

Este projeto se concentra na otimização de uma rede de distribuição de água em uma cidade. Os nós representam reservatórios, estações de bombeamento, e as arestas representam os canos conectando os diferentes pontos. O peso das arestas pode ser determinado pela resistência dos tubos e pela quantidade de água transportada.

Objetivos:

- Criar um modelo de grafo ponderado para a rede de abastecimento de água.
- Utilizar algoritmos de fluxo máximo (como o algoritmo de Edmonds-Karp) para otimizar o fluxo de água entre diferentes pontos da rede.
- Simular falhas em determinadas áreas da rede e encontrar soluções alternativas.

Aplicações:

- Planejamento e otimização da infraestrutura de abastecimento de água em grandes cidades.
- Análise de desempenho e de falhas em sistemas de distribuição.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Scipy para algoritmos de fluxo máximo, Matplotlib para visualização).

**5. Rede Elétrica do Brasil: Análise e Otimização do Sistema**

Descrição:

Este projeto envolve o uso de grafos para modelar a rede elétrica do Brasil, onde os nós representam usinas de geração de energia e subestações, enquanto as arestas representam as linhas de transmissão. O peso das arestas pode ser relacionado ao custo de transmissão ou à perda de energia ao longo do percurso.

Objetivos:

- Modelar a rede elétrica como um grafo ponderado, onde as arestas representam linhas de transmissão e os nós, usinas e subestações.
- Aplicar algoritmos de fluxo de potência para garantir uma distribuição eficiente e confiável de energia.
- Analisar vulnerabilidades e possíveis falhas na rede elétrica, como a sobrecarga de determinadas linhas.

Aplicações:

- Planejamento e expansão da infraestrutura elétrica de grandes áreas.
- Identificação de gargalos na transmissão de energia.
- Redução de perdas de energia e aumento da eficiência operacional.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, PyPSA ou pandapower para análise de redes elétricas, Matplotlib para visualização).

**6. Roteamento e Distribuição de Entregas: Sistema de Logística de Cargas**

Descrição:

Este projeto trata de otimizar o processo de distribuição de entregas, onde empresas de logística precisam fazer entregas de pacotes para vários destinos. As cidades ou pontos de entrega são representados como nós, e as estradas ou rotas de transporte como arestas, com pesos relacionados ao tempo de viagem, custo ou distância.

Objetivos:

- Modelar uma malha de transporte com grafos ponderados, onde os nós representam locais de entrega e as arestas representam as estradas.
- Implementar um Algoritmo de Roteamento de Veículos (VRP), buscando otimizar a distribuição dos pacotes com o mínimo de custo, considerando a capacidade das frotas.
- Permitir a visualização da rota de entrega no mapa, levando em conta restrições como prazos, custos de combustível e limites de capacidade dos veículos.

Aplicações:

- Otimização logística em empresas de transporte de carga e entrega de encomendas.
- Planejamento de rotas para reduzir os custos e o tempo de entrega.
- Distribuição de produtos em grandes centros urbanos ou áreas industriais.

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Google OR-Tools para resolver problemas de otimização de rotas de veículos, Matplotlib e Folium para visualização das rotas em mapas).

**7. Planejamento Urbano: Análise e Otimização de Malhas de Trânsito**

Descrição:

Neste projeto, você pode aplicar grafos para modelar uma malha de tráfego de uma cidade, onde os nós representam interseções e as arestas representam ruas, avenidas ou rodovias. A ideia é otimizar o tráfego e sugerir melhorias na malha viária com base em análises de fluxo e congestionamento.

Objetivos:

- Modelar o sistema de tráfego como um grafo ponderado, onde os pesos das arestas podem ser baseados no tempo médio de deslocamento ou no volume de tráfego.
- Usar algoritmos de Fluxo Máximo e Caminhos Mínimos para encontrar as rotas mais congestionadas e sugerir alternativas.
- Analisar como mudanças em uma interseção ou construção de novas ruas podem impactar a eficiência do sistema de tráfego.

Aplicações:

- Melhoria da mobilidade urbana em cidades, identificando gargalos e propondo soluções para otimizar o tráfego.
- Planejamento de novas infraestruturas viárias e ajustes em sinais de trânsito.
- Redução de congestionamentos e melhor distribuição do tráfego.

Tecnologias:

- Python (Bibliotecas: NetworkX para modelar o grafo do sistema de tráfego, Matplotlib para visualização do grafo e fluxos de tráfego e Geopandas para integração de dados espaciais e análise geoespacial).

**8. Malha de transportes sobre trilhos: Otimização de Rotas Ferroviárias**

Descrição:

Este projeto tem como objetivo modelar e otimizar a malha de transporte ferroviário de uma região, utilizando grafos para representar as estações ferroviárias como nós e as linhas ferroviárias como arestas. O peso das arestas pode ser baseado em vários fatores, como distância, tempo de viagem, custo operacional ou capacidade de carga.

Objetivos:

- Modelar as estações e linhas ferroviárias como um grafo ponderado.
- Implementar algoritmos de busca para encontrar as rotas mais curtas ou mais econômicas.
- Levar em consideração a **escalabilidade** para grandes redes ferroviárias, como aquelas de países ou continentes inteiros.

Aplicações:

- Planejamento de rotas ferroviárias comerciais.
- Gerenciamento de Tráfego Ferroviário.
- Identificação de Rotas Econômicas.
- Respostas a Emergências ou Atrasos (com sugestão de rotas alternativas para minimizar os impactos no sistema ferroviário).

Tecnologias:

- Python (Bibliotecas: NetworkX para grafos, Geopy para cálculo de distâncias geográficas, Matplotlib para visualização, Pandas para manipulação de dados).

**Bibliotecas Python para Interface Gráfica**

Para trabalhar com a interface gráfica dos projetos, existem várias bibliotecas em Python que podem ser utilizadas para criar interfaces interativas, tanto para visualização dos grafos quanto para interação com o usuário. Aqui estão algumas sugestões:

**1. Tkinter:**

- Descrição: Tkinter é a biblioteca padrão de Python para construção de interfaces gráficas. É uma boa opção se você precisar de uma interface simples para interagir com o usuário.
- Uso: Ideal para aplicações desktop mais simples, onde você quer permitir a entrada de dados, gerar gráficos ou exibir informações ao usuário.
- Integração com Projetos: Você pode usar o Tkinter para criar formulários para inserção de dados (ex: entrada de coordenadas, escolha de cidades para rotas, etc.) e para visualização gráfica com o `Matplotlib`.

**2. PyQt / PySide:**

- Descrição: PyQt e PySide são bibliotecas que permitem criar interfaces gráficas mais sofisticadas, com suporte para widgets modernos, gráficos interativos e até integração com recursos da web.
- Uso: É ideal para criar aplicativos mais robustos e com maior interatividade.
- Integração com Projetos: PyQt pode ser usado para construir painéis de controle interativos para visualizar e manipular as rotas, redes e fluxos de diferentes sistemas (como redes de transporte e energia).

**3. Dash (para aplicações web):**

- Descrição: Dash é uma biblioteca que facilita a criação de aplicativos web interativos com visualizações, gráficos e componentes de entrada.
- Uso: Se você quiser criar uma interface de visualização para o seu projeto de otimização de redes (por exemplo, mapas interativos de rotas de transporte ou redes elétricas), Dash permite criar interfaces dinâmicas facilmente integradas com bibliotecas como `Plotly` para gráficos.
- Integração com Projetos: Você pode usar Dash para criar dashboards interativos, onde o usuário pode inserir parâmetros e ver os resultados em tempo real (como o tempo de rota, capacidade de rede, etc.).

**4. Streamlit (para protótipos rápidos e visualização):**

- Descrição: Streamlit é uma biblioteca muito fácil de usar para a criação de aplicativos web interativos. É mais voltada para protótipos rápidos e visualização de dados.
- Uso: Ideal para construir uma interface de visualização de dados que permita ao usuário interagir com o modelo de otimização (ex: mudar parâmetros de uma rota de entrega e ver os resultados imediatos).
- Integração com Projetos: Streamlit pode ser usado para criar interfaces simples e rápidas para monitoramento de sistemas de tráfego, otimização de rotas ou análise de redes de fornecimento.

**Exemplo de Integração de `Tkinter` com `Matplotlib`**

Se você quiser criar uma interface gráfica simples para visualizar, por exemplo, uma rota de transporte, pode usar `Tkinter` para a interface e `Matplotlib` para a visualização do grafo. Aqui está um exemplo básico de como isso poderia ser feito:

```
import tkinter as tk
import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

# Criar o grafo
G = nx.erdos_renyi_graph(10, 0.5)
pos = nx.spring_layout(G)

# Criar a interface
root = tk.Tk()
root.title("Visualização de Rota de Transporte")

# Função para desenhar o grafo
def desenhar_grafo():
```

```
    fig, ax = plt.subplots(figsize=(5, 5))
    nx.draw(G, pos, with_labels=True, ax=ax, node_color="skyblue",
font_weight="bold")

    # Integrar o Matplotlib com Tkinter
    canvas = FigureCanvasTkAgg(fig, master=root)
    canvas.get_tk_widget().pack()
    canvas.draw()

# Botão para desenhar o grafo
btn = tk.Button(root, text="Desenhar Grafo", command=desenhar_grafo)
btn.pack()

root.mainloop()
```

Este código cria uma janela com um botão. Quando o botão é clicado, ele desenha um grafo gerado aleatoriamente usando `NetworkX` e exibe o gráfico dentro da janela Tkinter.

**Conclusão**

Cada um desses projetos aplica teoria de grafos e geometria computacional para resolver problemas do mundo real, como otimização de redes de transporte, infraestrutura e serviços públicos. A escolha de ferramentas como NetworkX, Google OR-Tools e bibliotecas geoespaciais permite modelar grafos, otimizar rotas e analisar redes complexas de maneira eficiente. Além disso, você pode escolher uma biblioteca Python adequada para criar interfaces gráficas de acordo com a complexidade e as necessidades do seu projeto, seja usando Tkinter, PyQt, Dash, ou Streamlit.

**Exemplos de grafos quanto as malhas ferroviárias e aéreas:**

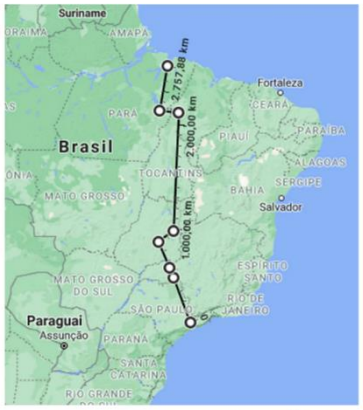



Figura – Rota em linha operada pelo voo 254 da VARIG


Figura – Rota circular hipotética

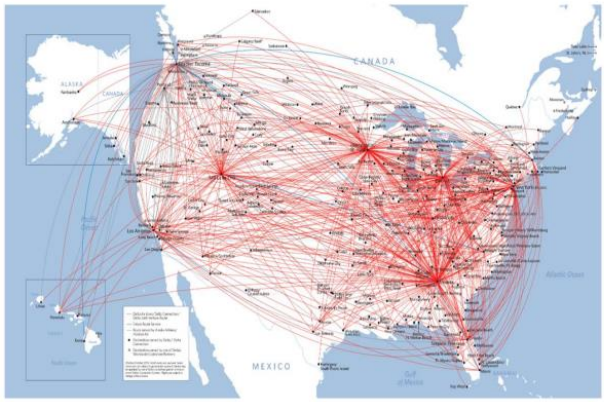



Figura – Malha aérea da Delta Air Lines com múltiplos hubs

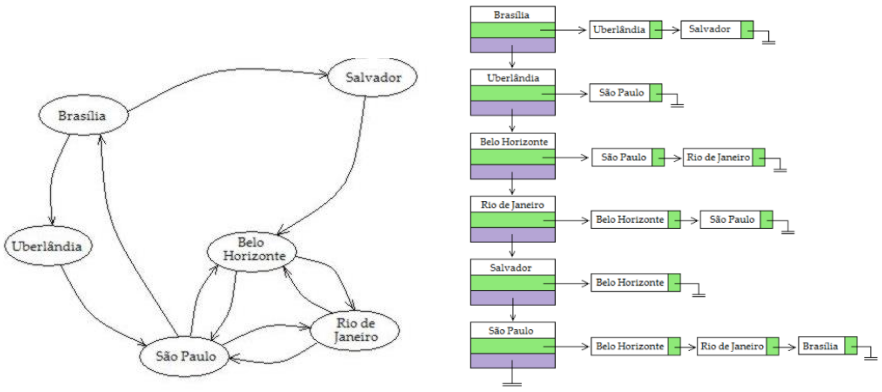
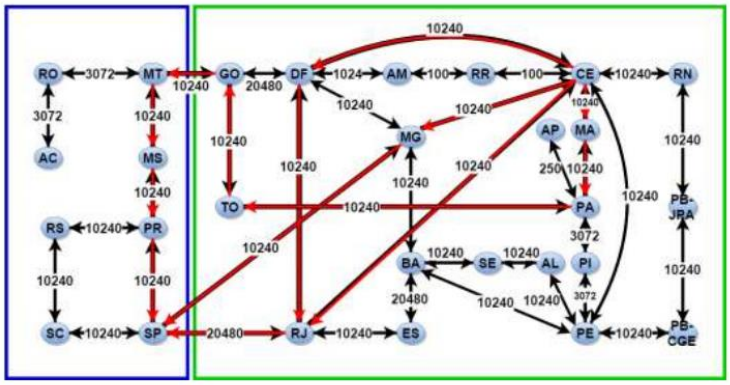



**Figura.** Malha aérea usando lista de listas

**Figura** . Fluxo na rede Ipê.




fonte destino do fluxo

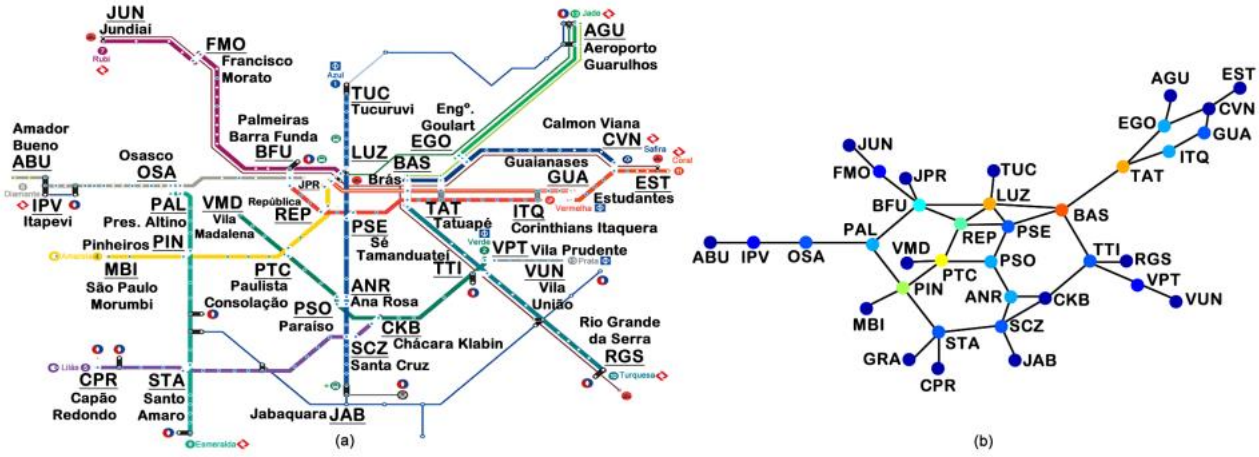



**Figura.** Malha de transportes sobre trilhos

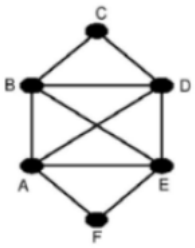



**Figura.** Representação geométrica do problema da coleta de lixo.


**Figura.** Grafo representativo da rede e água de uma cidade.

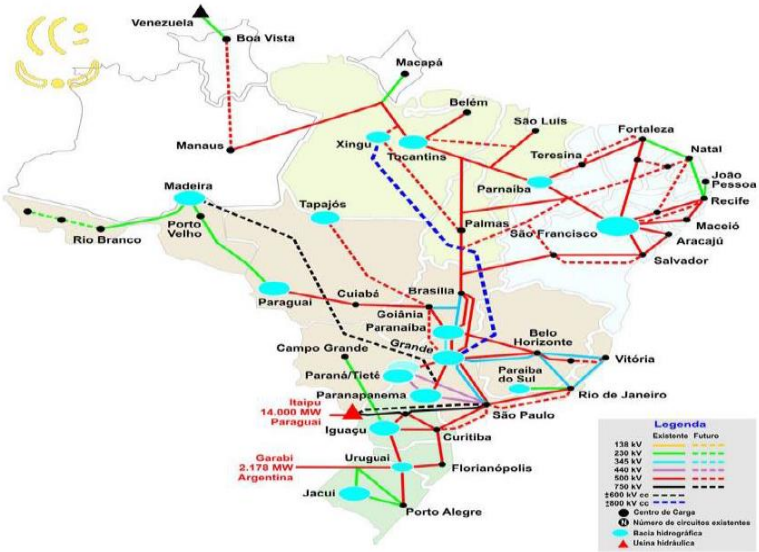



**Figura.** Mapa da rede elétrica do Brasil.