




Algoritmos

Algoritmo de Dijkstra

Vinícius Luiz
Alexsandro Henrique



Contexto do problema

Em 1997, havia 332 aeroportos nos Estados Unidos, com isso, seria possível haver cerca de **54.946** rotas sem escala no território americano.

Observação: Uma rota de um aeroporto A para B é considerado a mesma rota do aeroporto B para A

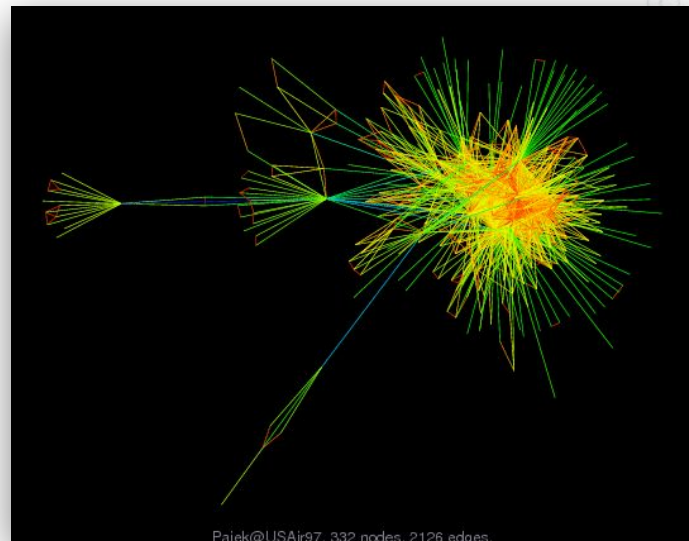
Das **54.946** rotas sem escalas possíveis, apenas **2126** são utilizadas, ou seja, **96,13%** das rotas possíveis não existem.

Base de Dados - Pajek network: US Air flights, 1997

A base de dados contém 2 arquivos importantes:

- “USAir97” contém:
 - Quantidade de vértices (aeroportos)
 - Quantidade de arestas (rotas sem escala)
 - Rotas juntamente com seus pesos (distância em milhas)
- “USAir97_nodename” contém:
 - Todos os nomes dos aeroportos em ordem de índice

O grafo é **ponderado** e **não-direcionado**



Pajek@USAir97, 332 nodes, 2126 edges

Desenho do grafo

Problema a ser resolvido

Encontrar o menor caminho entre dois aeroportos não diretamente conectados.

- Existem **52.820** rotas a serem encontradas.
- Devemos encontrar a menor rota **com escala** entre 2 aeroportos não conectados.

Etapas da resolução

ETAPA 1

Dado dois aeroportos A e B, verificamos se eles possuem uma rota sem escala.

ETAPA 2

Se não houver, o algoritmo de Dijkstra encontra a menor rota com escala entre A e B

Etapas da resolução

```
NOME DO AEROPORTO 1
```

```
> Mercedita
```

```
NOME DO AEROPORTO 2
```

```
> Los Angeles Intl
```

```
DISTÂNCIA DA ROTA SEM ESCALAS
```

```
>> INEXISTENTE
```

```
DISTÂNCIA DA ROTA MAIS CURTA COM ESCALAS
```

```
>> Mercedita > Miami Intl > Los Angeles Intl > MILHAS PERCORRIDAS: 3112.0
```

Tecnologias utilizadas

- A representação do grafo foi implementada por meio de uma matriz de adjacência
- Foi implementado também um dicionário que dado um vértice, visualizamos o nome dos seus adjacentes, assim, facilitando a busca.

```
g = Grafo_Matriz(5, ponderado = True, direcionado = False)
g.inserir_aresta('aeroporto1 aeroporto2', 10)
g.inserir_aresta('aeroporto3 aeroporto4', 5)
g.inserir_aresta('aeroporto5 aeroporto4', 2)
g.inserir_aresta('aeroporto2 aeroporto3', 1)
g.inserir_aresta('aeroporto5 aeroporto1', 3)
```

```
aeroporto1 [None, 10, None, None, 3]
aeroporto2 [None, 1, None, None]
aeroporto3 [None, 5, None]
aeroporto4 [None, 2]
aeroporto5 [None]
```

Resultados

Analisar 52.820 rotas inexistentes é muito custoso

Em uma análise **completa**, o algoritmo de Dijkstra consegue encontrar um menor caminho em **todas** elas!

```
[1] COMPARAR ROTAS COM E SEM ESCALA ENTRE 2 AEROPORTOS
[2] VER NOMES DOS AEROPORTOS
[3] VER QUANTIDADE DE ROTAS INEXISTENTES
[4] VER QUANTIDADE DE AEROPORTOS E ROTAS
[5] SAIR
> 3
FORAM ENCONTRADAS 52820 ROTAS SEM ESCALAS INEXISTENTES ENTRE DOIS AEROPORTOS
FORAM ENCONTRADAS 52820 ROTAS COM ESCALA PARA SUPRIR AS FALTANTES
NÃO SERÁ POSSÍVEL REALIZAR 0 VOOS
```


Resultados

Analisar 52.820 rotas inexistentes é muito custoso

Em uma análise **manual**, foi encontrado voos longos com até 5 escalas, sendo esse o menor caminho entre a origem e o destino.

```
NOME DO AEROPORTO 1  
> Guam Intl  
NOME DO AEROPORTO 2  
> Eek
```

```
DISTÂNCIA DA ROTA SEM ESCALAS  
>> INEXISTENTE
```

```
DISTÂNCIA DA ROTA MAIS CURTA COM ESCALAS  
>> Guam Intl > Honolulu Intl > Anchorage Intl > Bethel > Eek >
```

MILHAS PERCORRIDAS: 7499.0

Conclusão

1. O algoritmo de Dijkstra mostra grande eficiência quando se trata de rotas aéreas, visto que um país – como o EUA – possui aeroportos bem distribuídos pelo seu território.
2. Desde que todas os vértices (aeroportos) estejam bem distribuídos, conseguimos definir caminhos para suprir as rotas faltantes.
3. Em relação aos voos com muitas escalas, dependendo da demanda dessa rota, é viável criar novas rotas para diminuir as escalas desse voo.

A decorative network graph pattern in the top-left corner, featuring a complex web of interconnected nodes and edges. Some nodes are highlighted with blue circles, and others with blue dots.

Fim

Algoritmo de Dijkstra

A decorative network graph pattern in the bottom-right corner, featuring a complex web of interconnected nodes and edges. Some nodes are highlighted with blue circles, and others with blue dots.