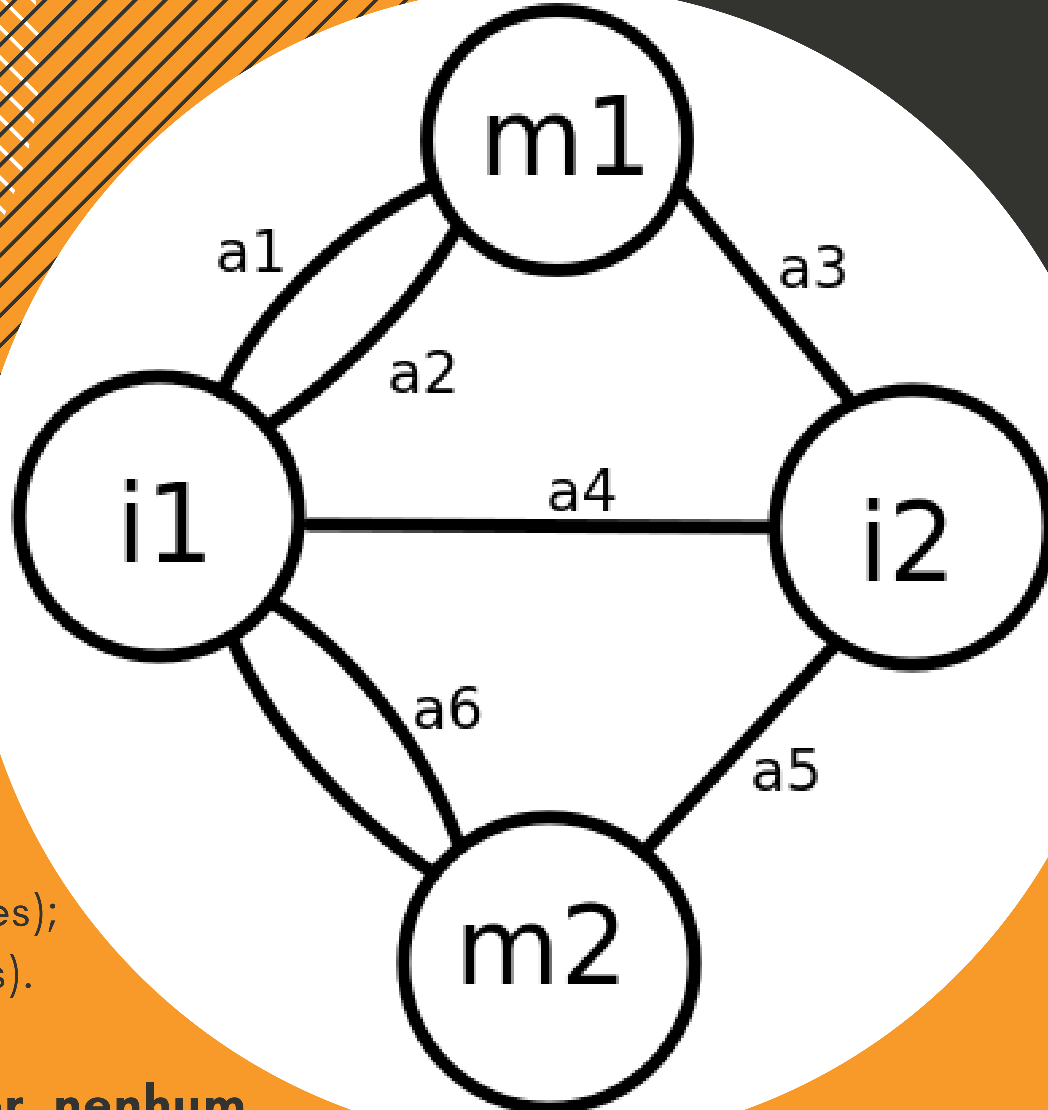
PROF. MATHEUS
LEANDRO FERREIRA

Teoria de Grafos

JHONNY MEZZARI, JOÃO VITOR ALVES,
LEONARDO SPILERE, NATANAEL ALVES

Pontes de Königsberg



- i1 e i2 representam as ilhas (vértices);
- m1 e m2 representam as margens (vértices);
- a1 até a6 representam as pontes (arestas).

- **De acordo com o Teorema de Euler, nenhum dos nós tem um grau par, isso significa que não é possível criar um caminho que cruze todas as pontes uma única vez e retorne à márgem inicial, sendo assim o problema das pontes de Königsberg não tem solução, pois não é possível criar um caminho que satisfaça todas as condições necessárias.**

Teorema Euleriano

Considere um conjunto de cidades com estradas ponderadas entre elas:

- Cidade A está conectada a B, C e D.
- Cidade B está conectada a A, C e D.
- Cidade C está conectada a A, B e D.
- Cidade D está conectada a A, B e C.

E uma matriz de adjacência que representa os custos de viagem entre as cidades:

	A	B	C	D
A	0	5	8	6
B	5	0	7	9
C	8	7	0	4
D	6	9	4	0

Uma possível solução seria:
A -> B -> C -> D -> A

Circuito

Teorema Euleriano

Vamos considerar o seguinte exemplo de grafo:

```
A -- B -- C
|    |    |
D -- E -- F
```

Neste grafo, os vértices têm os seguintes graus:

- A: grau 2
- B: grau 3
- C: grau 2
- D: grau 2
- E: grau 4
- F: grau 2

Uma possível solução seria:
A -> B -> C -> D -> A

Caminho

1 Método de Dijkstra

- **Passo 1:**

Inicialização. Nós não visitados: A, B, C, D, E, F.
Tabela de distâncias mínimas: A = 0 (A é o nó de origem)

- **Passo 2:**

Escolher o nó mais próximo não visitado.
O nó A é o nó mais próximo não visitado, com uma distância de 0.

- **Passo 3:**

Atualização das distâncias mínimas.
A -> B: 8 (atualização da distância de B)
A -> C: 2 (atualização da distância de C)

Tabela de distâncias mínimas atualizada:
A: 0   B: 8   C: 2

1 Método de Dijkstra

- **Passo 4:**

Escolha o próximo nó mais próximo não visitado.
O nó C é o nó mais próximo não visitado, com uma distância de 2.

- **Passo 5:**

Atualização das distâncias mínimas.
C -> D: 2 (atualização da distância de D)
C -> E:5 (atualização da distância de E)

Tabela de distâncias mínimas atualizada:
A: 0 B: 3 C: 2 D: 10 E: 12 F: nulo

1 Método de Dijkstra

- **Passo 6:**

Continue escolhendo e atualizando os nós mais próximos não visitados até que todos os nós tenham sido visitados.

Próximo nó mais próximo: B (com distância 3)
Atualização de D: 8 (B -> D)

Próximo nó mais próximo: D (com distância 8)
Atualização de E: 10 (D -> E)
Atualização de F: 14 (D -> F)

Próximo nó mais próximo: E (com distância 10)

Próximo nó mais próximo: F (com distância 12)
A tabela final de distâncias mínimas será:
A: 0

2 Método de Dijkstra

- **Passo 1:**

Inicialização. Nós não visitados: A, B, C, D, E, F, G, H

Tabela de distâncias mínimas:

A: 0 (A é o nó de origem)
B: nulo
C: nulo
D: nulo
E: nulo
F: nulo
G: nulo
H: nulo

- **Passo 2:**

O nó A é o nó mais próximo não visitado, com uma distância de 0.

2 Método de Dijkstra

- **Passo 3:**

Atualização das distâncias mínimas dos nós vizinhos de A e sua sequência:

A -> C: 2 (atualização da distância de C)
C -> D: 2 (atualização da distância de D)
D -> E: 1 (atualização da distância de E)
E -> G: 1 (atualização da distância de G)
G -> F: 0 (atualização da distância de F)
F -> H: 3 (atualização da distância de H)

Tabela de distâncias mínimas atualizada:

A: 0 B: nulo C: 2 D: 4 E: 5 F: 6 G: 6 H: 9

Portanto, o caminho mais curto de A para H é:

A -> C -> D -> E -> G -> F -> H

com uma distância total de:

2 + 2 + 1 + 1 + 0 + 3 = 9 unidades

de acordo com os pesos das arestas no grafo.

Método Bellman-Ford

Inicializamos a distância de todos os vértices como infinito exceto o vértice de origem que será 0;

dist[A] = 0
dist[B] = ∞(infinito),
dist[C] = ∞(infinito),
dist[D] = ∞(infinito),
dist[E] = ∞(infinito),

Fazemos N-1 iterações onde N é o número total de vértices no grafo
N-1 = 5-1 = 4

Método Bellman-Ford

- **Primeira iteração:**

O algoritmo irá pegar a atual distância do primeiro vértice= [B],
Sua distância atual = ∞(infinito),
Será maior = >,
Que a distância de A = 0,
E a distância de A para B = -1,
Somados = 0 - 1 = -1,
Que resultará em = ∞(infinito) > -1 (infinito maios que -1);

Aplicando a regra obtemos os valores:
dist[B] = min(dist[B] > dist[A] + peso(A -> B)) = min(∞, 0 - 1) = -1
dist[C] = min(dist[C] > dist[A] + peso(A -> C)) = min(∞, 0 + 4) = 4

Método Bellman-Ford

Atualizando a tabela temos:

dist[A] = 0,
dist[B] = -1,
dist[C] = 4,
dist[D] = ∞(infinito),
dist[E] = ∞(infinito),

- **Segunda Iteração:**

dist[C] = min(dist[C], dist[B] + peso(B -> C)) = min(4, -1 + 3) = 2
dist[D] = min(dist[D], dist[B] + peso(B -> D)) = min(∞, -1 + 2) = 1
dist[E] = min(dist[E], dist[B] + peso(B -> E)) = min(∞, -1 + 2) = 1

Método Bellman-Ford

Atualizando a tabela temos:

dist[A] = 0 dist[B] = -1 dist[C] = 2,
dist[D] = 1 dist[E] = 1,

- **Terceira Iteração:**

dist[D] = min(dist[D], dist[E] + dist[E -> D]) = min(1, 1 + (-3)) = -2

Atualizando a tabela temos:

dist[A] = 0 dist[B] = -1 dist[C] = 2,
dist[D] = -2 dist[E] = 1

Método Bellman-Ford

- **Quarta Iteração:**

dist[B]= min(-1, -2 + 1)= -1
dist[C]= min(2, -2 + 5)= 3

Atualizando a tabela temos:

dist[A] = 0 dist[B] = -1 dist[C] = 2,
dist[D] = -2 dist[E] = 1,

Resumindo o menos caminho de A para os vértices serão:

A -> B = -1 (A,B);
A -> C = 2 (A,B,C);
A -> D = -2 (A,B,E,D);
A -> E = 1 (A,B,E);

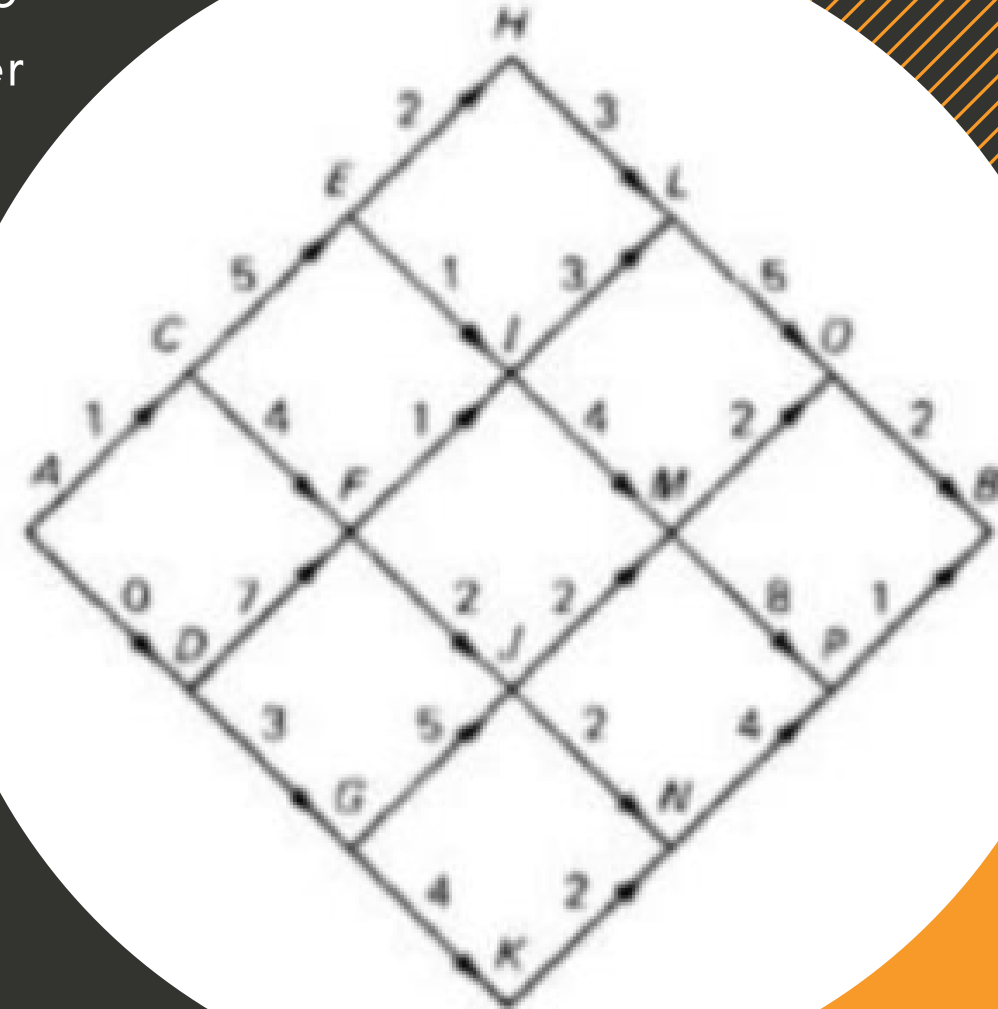
Ponto Extra

Para encotrar o menor caminho do grafo, o melhor algorítimo para ser utilizado será o **Bellman-Ford**.

O caminho será:

A -> C = 1
C -> F = 4
F -> J = 2
J -> M = 2
M -> O = 2
O -> B = 2

Custo Total = 13



PROF. MATHEUS
LEANDRO FERREIRA

Teoria de Grafos

JHONNY MEZZARI, JOÃO VITOR ALVES,
LEONARDO SPILERE, NATANAEL ALVES