

# GRAFOS

## 06/10

1. Seja  $G = (V, E)$  um grafo não-direcionado e conexo. **Seja  $(G, w)$  um grafo ponderado em que  $w: E \rightarrow R^+$ .** Projete uma solução para encontrar um sub-grafo conexo de forma que a soma dos pesos das arestas seja o maior possível.

R: Escolhemos qualquer vértice, pois qualquer um deles será um sub-grafo com o menor peso possível na soma de suas arestas.

2. Seja  $G = (V, E)$  um grafo não-direcionado e conexo. **Seja  $(G, w)$  um grafo ponderado em que  $w: E \rightarrow R^+$ .** Projete uma solução para encontrar um sub-grafo  $T = (V, E') \leq G$  conexo de forma que a soma dos pesos das arestas seja o maior possível.

R: PRIM DJKISTRA | KRUSKAL | REMOÇÃO REVERSA

## 08/10

1. Seja  $G = (V, E)$  um grafo não-direcionado e conexo, e  $(G, w)$  um grafo ponderado em que  $w: E \rightarrow R$ . Projete uma solução para encontrar uma árvore geradora mínima a partir de  $(G, w)$ .

R:

2. Seja  $G = (V, E)$  um grafo não-direcionado e conexo, e  $(G, w)$  um grafo ponderado em que  $w: E \rightarrow R$ . Projete uma solução para encontrar uma árvore geradora máxima a partir de  $(G, w)$ .

R:

3. É possível DIJKSTRA e KRUSKAL gerarem a "mesma árvore"?

Union-Find

## 15/10

1. Seja  $G = (V, E)$  um grafo não-direcionado e simples. Seja  $(G, w)$  um grafo ponderado em que  $w: E \rightarrow R^+$ 
  - a. Como calcular uma árvore geradora mínima em  $(G, w)$ ?

R: PRIM, KRUSKAL, REMOÇÃO REVERSA.

- b. Como aplicar Dijkstra em  $(G, w)$  a partir de um vértice dado?

R: Pode funcionar, mas nem sempre.

- c. Projete um algoritmo entre uma origem e um destino que tenha números positivos e negativos.

R:

Quando meu grafo é uma árvore, o Djikstra encontra a mesma árvore geradora.

Árvore  $T = (V, E)$

$u, v$  pertencem a  $V$

- Quantos caminhos simples existem entre  $u$  e  $v$ ?

R: Só existe um caminho, pois em árvores só possuem um caminho.

Conceitos:

- Árvore geradora de um grafo conexo → É um **SUBGRAFO** conexo desse grafo com todos os vértices do primeiro.
- Árvore geradora **MÍNIMA** de um grafo conexo → É um **SUBGRAFO** conexo desse grafo com todos os vértices do primeiro, de forma que tenha a menor quantidade de arestas.
  - Se for um grafo ponderado, tratamos o **SUBGRAFO** conexo observando a menor soma de pesos possível.

Como garantimos que elementos terminais se mantenham conexos.

Árvore de Steiner é um subgrafo, que minimize a soma de algum peso que a gente queira da minha árvore, mantendo os terminais conexos.

- Ela n precisa conter todos os elementos.
- Os pontos de Steiner são os pontos que não são terminais, mas que fazem parte da árvore de Steiner.

Quando a árvore de Steiner é resolvida de forma fácil?

- Quando o número de vértices é igual ao número de terminais.
- Pq é uma árvore geradora mínima para esse caso.

Árvore degenerada → Uma linha.

Estudar o que é uma **HEAP** mínima e máxima, principalmente por vetor.

PRIM, KRUSKAL E REMOÇÃO REVERSA geram a **MESMA** árvore se tiverem todos os pesos diferentes.

Árvore geradora máxima só funciona para pesos positivos para os algoritmos acima.

### Shortest Path

1. Uma origem e um destino → Djkistra funciona aqui.
2. Uma origem e vários destinos → Djkistra funciona aqui.
3. Várias origens e vários destinos → Djkistra não funciona de forma adequada aqui.

## 22/10

$$1. G = (V, E) \text{ em que } V = \{1, 2, 3, 4\} \mid E = \{\{1, 2\}, \{3, 4\}\}$$

$$H = (V', E') \text{ em que } V' = \{a, b, c, d\} \mid E' = \{\{a, d\}, \{c, b\}\}$$

$G = H$ ? Justifique sua resposta.

R: Dois grafos são iguais se e somente se  $V = V'$  e  $E = E' \rightarrow$  Mesmo conjunto de vértices e mesmo conjunto de arestas. Nesse caso, não são iguais.

$$V \subseteq V' \wedge V' \subseteq V$$

$$E \subseteq E' \wedge E' \subseteq E$$

Agora, caso seja possível mapear cada vértice de um grafo em um outro grafo, de forma que os dois terão vértices e arestas com as mesmas características, eles são **ISOMORFOS**.

- Mesma quantidade de vértices.
- Mesma quantidade de arestas.

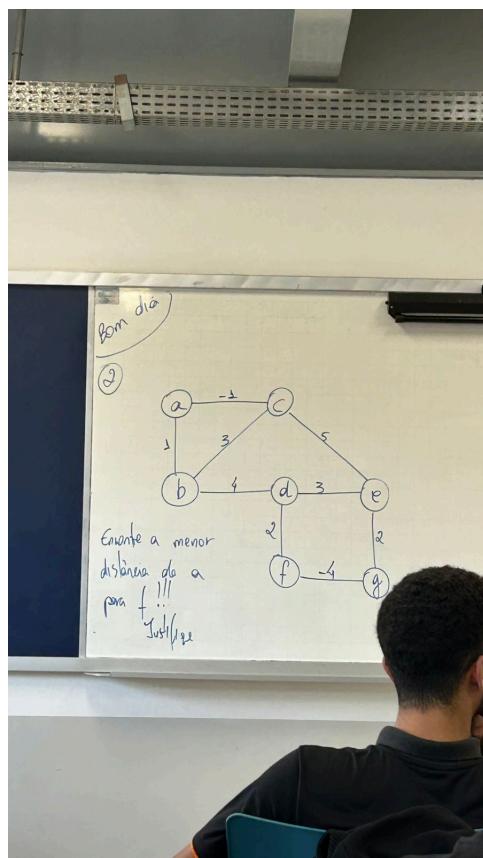
- Mesma sequência de graus.
- Vizinhos de um vértice com as mesmas características; Vizinhos dos vizinhos com as mesmas características (Mesmo mapeamento).
- Mesma quantidade de componentes.

Grafo associado → Grafo direcionado que tiramos a direção

- Adaptações ao processo de isomorfismo

Estrutura topológica → Dois grafos possuem a mesma estrutura, logo são **ISOMORFOS**.

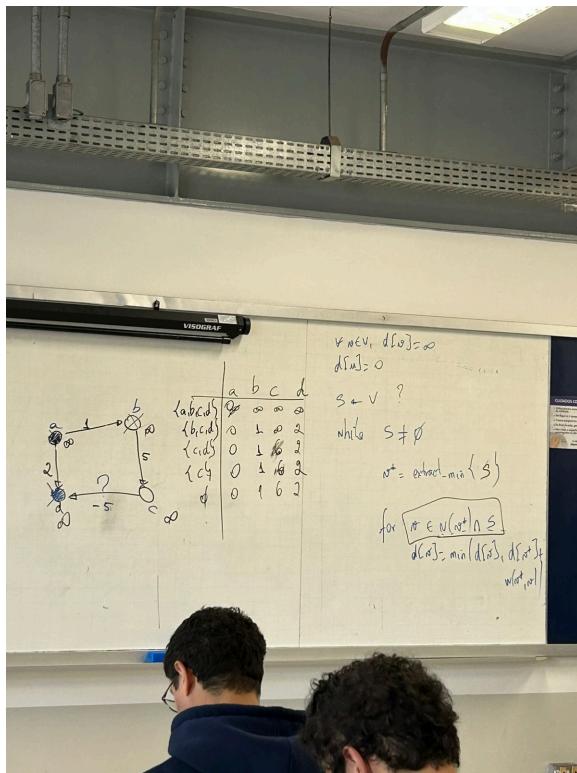
2. Questão 2:



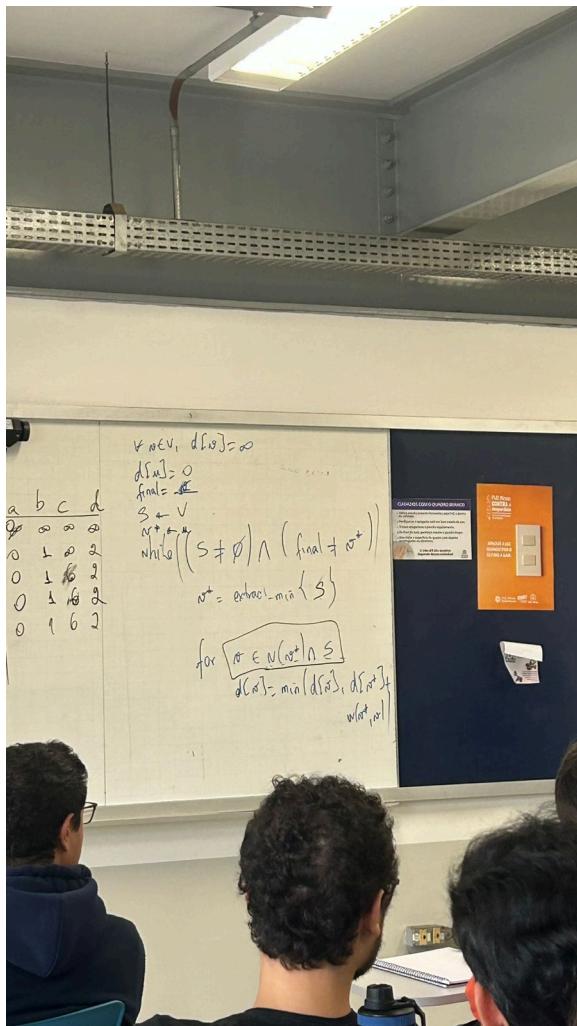
Bellman Ford não funciona para grafos não-direcionados.

## 23/10

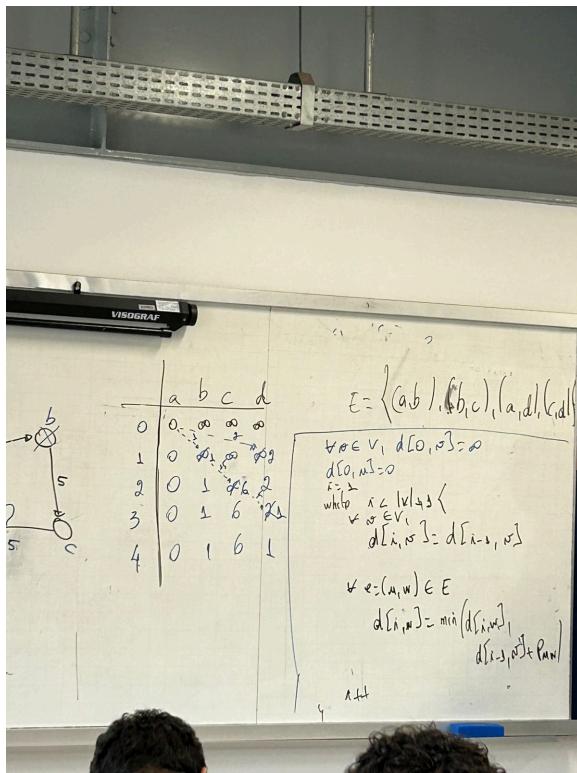
Dijkstra x Peso negativo → Um vértice já visitado pode não ser atualizado novamente, mesmo que seja necessário para encontrar o menor caminho, caso tenham números negativos e positivos.



Dijkstra com início e sem fim, com pesos positivos e negativos.



Dijkstra com início e fim, com pesos positivos e negativos.



Bellman Ford (O segundo "u" não se refere ao primeiro).

Grafos Eulerianos:

- Ciclo.
- Passar em todas as arestas uma única vez.
- Voltar para origem.

## 27/10 - Exercício 4, 12, 14

Exercício 4:

Em um rede de comunicação direcionada, uma mensagem é enviada de um nó  $a$  para outro nó  $b$ . Em cada arco  $(i, j)$  da rede, existe uma probabilidade  $p_{i,j}$  de que a mensagem se perca neste arco. Essas probabilidades são fixas e independentes. O gerente da rede deseja escolher um caminho de  $a$  até  $b$ , tal que a probabilidade de a mensagem se perder nesse caminho seja mínima. Traduza este problema em um problema de menor caminho. Este problema pode ser resolvido com o algoritmo de Dijkstra?

1. Produto das probabilidades que estão em cada aresta.
2.  $\log b(a.c) = \log b(a) + \log b(c) \rightarrow$  Temos que fazer essa transformação, pois o Dijkstra não funciona fazendo o produto do peso das arestas.

3. Aplicamos  $\text{logb}(w(e)) \rightarrow$  Aplicamos o log em cada probabilidade (peso) das arestas.
4.  $0 < w(e) < 1 \rightarrow$  resultado negativo.
5.  $w'(e) = -\text{logb}(w(e)) \rightarrow$  Negamos o resultado do log para que ele fique positivo.
6. Aplicamos o Dijkstra.

### Exercício 12:

Seja um grafo conexo  $G$  com pesos positivos e uma árvore geradora mínima de  $G$ . Assinale a(s) afirmativa(s) correta(s) para manter a mesma árvore geradora mínima.

- a) Altere o peso de cada aresta de  $w(e)$  para  $w(e) + 15$ .
- b) Altere o peso de cada aresta de  $w(e)$  para  $w(e) \times 15$
- c) Altere o peso de cada aresta de  $w(e)$  para  $w(e) \times w(e)$

### AGM → Usando Kruskal:

- Grafo conexo.
- Ordena “Não-Decrescente”.
- Análise de arestas que estão gerando ciclos  $\rightarrow$  A árvore geradora mínima contém a menor quantidade de arestas, desde que não tenha ciclos.
- $T = (V, E') \subseteq G$   
 $T' = (V, E'') \subseteq G$   
onde  $E' = E''$
- Para essa questão, devemos preservar a ordem da análise das arestas.
  - Isso nos leva a aplicar a mesma função tanto no grafo  $G$ , quanto na AGM.
  - $(T, w)$  por exemplo.
  - Como nessa questão o peso é **SEMPRE** positivo, está tudo bem e podemos afirmar **TODAS** as afirmativas.
  - A atenção está para quando existem pesos negativos, pois a ordem de cada aresta pode variar.
    - $w: E \rightarrow N \mid$  Funciona

- $w: E \rightarrow Z^+ \mid$  Funciona
- $w: E \rightarrow R^+ \mid$  Funciona
- Não funciona com certeza quando tivermos pesos negativos, ou quando a constante que multiplicarmos for negativa.

#### Exercício 14:

Projete um algoritmo para determinar a menor mudança no custo da aresta que produziria uma mudança na árvore geradora mínima.

$$G = (V, E)$$

$$T = (V, E') \text{ em que } E' \subseteq E$$

$$T' = (V, E'') \text{ em que } E'' \subseteq E$$

$T$  é uma AGM se  $\nexists T'$  tal que a soma de pesos das arestas de  $T'$  seja menor que a soma de pesos das arestas de  $T$ .

Passo a passo de para montar uma AGM com o Kruskal:

1. Ordena de forma “não decrescente” os pesos das arestas.
2. Começa a unir as arestas, analisando sempre se a inclusão de uma aresta irá criar um ciclo. Se criar, não é incluída.

Para essa questão, devemos fazer um algoritmo que acha ciclos no grafo original. No ciclo vamos verificar a menor ...

## 28/10

1. Seja  $G = (V, E)$  um grafo não-direcionado. Analise e responda:

a. Se  $G$  é euleriano, então  $G$  é hamiltoniano?

Podemos dizer que é possível ter um grafo euleriano e hamiltoniano, mas não afirmar que sendo euleriano ele é hamiltoniano sempre. O motivo disso é: para ser euleriano ele deve passar por todas as arestas, mas sem repetí-las. Passar por todas arestas implica visitar todos os vértices, mas não podemos garantir que os vértices não serão repetidos. Caso tenham vértices repetidos, não podemos dizer que ele é hamiltoniano. Um ciclo em forma de um triângulo, por exemplo, é euleriano e hamiltoniano, pois ele consegue passar por todas as arestas

sem repetição das mesmas e em todos os vértices sem repetição dos mesmos, exceto do inicial que também é o ponto de retorno. Agora, um pentágono com um vértice diagonal em seu interior não pode ser euleriano e hamiltoniano, apenas euleriano. Isso acontece justamente porque, nesse caso, é possível visitar todas as arestas sem repetí-las, mas impossível não repetir um vértice. Ou seja, quando temos dois ciclos que compartilham um vértice entre si, não é possível que o grafo tenha ambas classificações.

b. Se  $G$  é hamiltoniano, então  $G$  é euleriano?

A resposta é não, pois temos um quadrado, por exemplo, em que é possível passar em todos os vértices, mas não é possível passar em todas as arestas. (Quando a resposta for não, podemos apenas mostrar um contra exemplo)

2. Seja  $G = (V, E)$  um grafo direcionado, projete um algoritmo para encontrar o maior caminho de  $G$ , caso possível. Façam as considerações que julgarem necessárias.

Temos três opções:

Primeiramente devemos considerar que temos um grafo **ACÍCLICO**.

- Partimos do pressuposto que fizemos uma ordenação topológica (colocar em uma ordem de forma que todas as arestas saiam da esquerda para direita - utilizamos o algoritmo de remoção de bases para isso). Essa ordenação define a ordem de análise dos vértices, garantindo que todo o fecho transitivo inverso de um  $V$  já tenha sido analisado quando chega a vez dele.
- Utilizamos o algoritmo de remoção de bases atualizando a distância e os antecessores de cada vértice, sempre atualizando o antecessor pelo último encontrado.
- Utilizamos o Dijkstra nas bases do grafo e, antes disso, colocamos o peso das arestas como -1.

3. Seja  $G = (V, E)$  um grafo não-direcionado e  $(G, w)$  um grafo ponderado.

Seja  $T = (V, E')$  uma árvore obtida pelo Dijkstra para achar o maior caminho e  $T' = (V, E'')$  uma árvore geradora máxima. Podemos afirmar que a aresta de maior peso está em ambas as árvores?

Dijkstra só funciona para acíclicos e com pesos todos positivos ou todos negativos.

Em grafos acíclicos a resposta é verdadeiro.

Em grafos cíclicos a resposta é falsa.