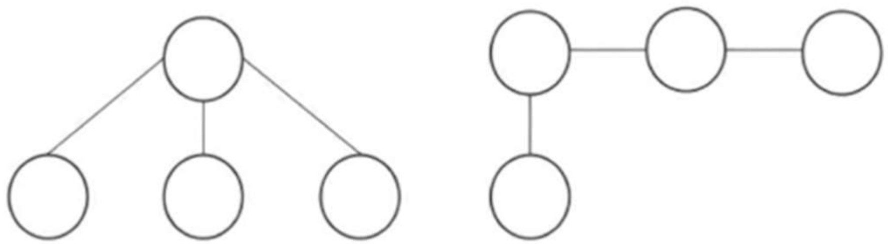
Teoria dos GRAFOS

Aula 06: Árvores

Prof. José Alberto S. Torres

O que são **árvores?**

- No estudo de grafos, uma árvore é um tipo especial de grafo que possui duas características principais:
  - **É um grafo conectado**: é possível ir de qualquer vértice para qualquer outro. Não existem partes isoladas.
  - **Não contém ciclos**: não há caminhos fechados. Se você começar a percorrer o grafo a partir de um vértice, nunca voltará ao ponto de partida sem ter que refazer seus passos.



O que são **árvores?**

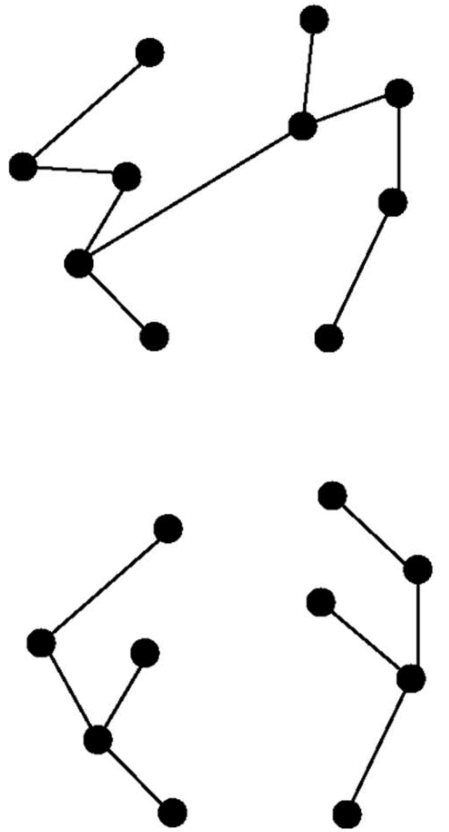
- A partir dessas duas propriedades, podemos inferir outras características importantes das árvores:
    - **Arestas e Vértices**: Uma árvore com n vértices sempre terá n−1 arestas. Por exemplo, se uma árvore tem 5 vértices, ela terá exatamente 4 arestas.
    - **Aresta de Corte (ponte):** Cada aresta em uma árvore é uma "aresta de corte". Isso significa que se você remover qualquer aresta, o grafo se tornará desconectado.
    - **Caminho Único:** Existe um e somente um caminho simples entre quaisquer dois vértices em uma árvore.

Floresta

- Em teoria dos grafos, uma **floresta** (*forest*) é uma **coleção de uma ou mais árvore**
- Formalmente, uma floresta é um **grafo não-direcionado acíclico**.
- A diferença fundamental em relação a uma árvore é que uma floresta **não precisa ser conectada**.

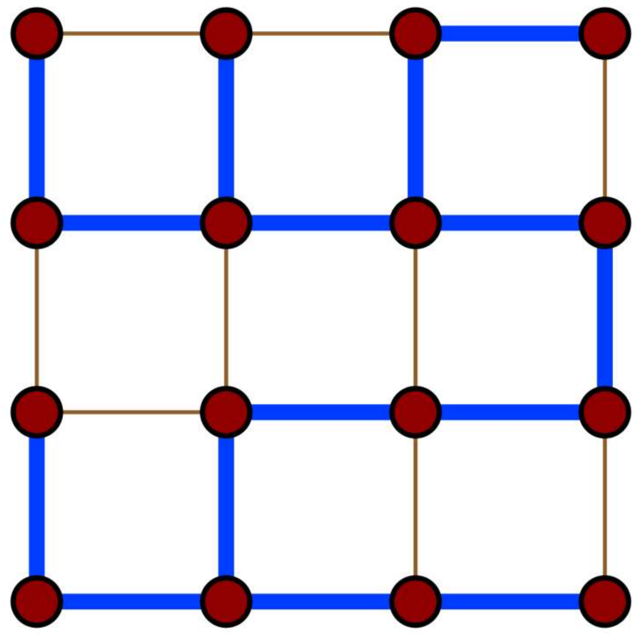
Árvore X Floresta

- Uma **Árvore** é um grafo **acíclico E conectado**.
- Uma **Floresta** é um grafo **acíclico** (que pode ser conectado ou desconectado).
- Toda árvore é, tecnicamente, uma floresta (uma floresta com apenas um componente conectado).
- Uma floresta é um grafo cujos componentes conectados são, individualmente, árvores.



Árvore Geradora

- Uma Árvore Geradora de um grafo G conectado e não-direcionado é um subgrafo de G que satisfaz três condições:
  - Inclui todos os vértices do grafo original G. (Esta é a parte "geradora" ou "spanning").
  - É acíclica (não contém ciclos) e é conectada.
  - É formada por um subconjunto das arestas do grafo original G.



Algoritmo de **Kruskal**

- Mais interessante do que simplesmente notar que um grafo conectado G possui uma árvore geradora, é **encontrar uma árvore geradora mínima em um grafo conectado ponderado G**.
- Em outras palavras, nosso objetivo é encontrar uma árvore geradora T com peso mínimo entre todas as árvores geradoras de G.
- Um algoritmo que constrói eficientemente tal árvore foi projetado por Kruskal.

Algoritmo de **Kruskal**

- O Algoritmo de Kruskal é um exemplo clássico de algoritmo guloso (greedy algorithm).
- A estratégia gulosa aqui é: **Sempre escolha a opção que parece ser a melhor no momento**.
- Para Kruskal, a "**melhor opção**" é sempre **a aresta de menor peso disponível que ainda não foi escolhida**.
- Ele constrói a árvore aresta por aresta, **começando pelas mais "baratas".**
- A única regra é: ao adicionar uma nova aresta, você **não pode criar um ciclo** com as arestas que já foram adicionadas.

Kruskal passo a passo

1. **Ordenar as Arestas:** Crie uma lista de todas as arestas do grafo e ordene-a em ordem crescente de peso (da mais leve para a mais pesada).
2. **Inicializar a Floresta:** Crie uma "floresta" onde cada vértice do grafo é sua própria árvore individual. Em outras palavras, no início, nenhum vértice está conectado a outro.

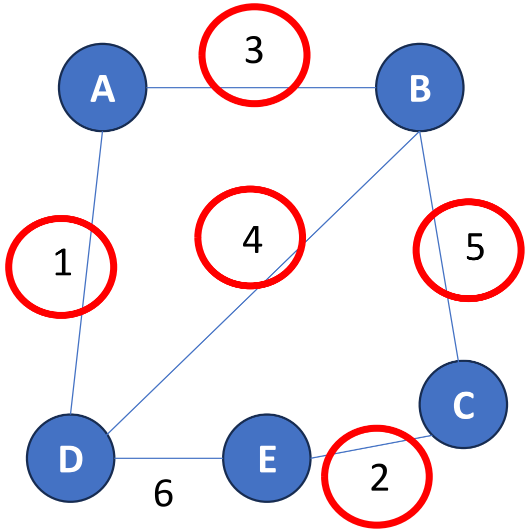
Kruskal passo a passo

3. **Iterar e Construir a Árvore:** Percorra a lista de arestas ordenadas (da mais barata para a mais cara):
   - Para cada aresta (u, v):
   - Verifique se os vértices u e v já pertencem à mesma árvore (ou ao mesmo componente conectado).
     - **Se NÃO pertencem à mesma árvore:** A aresta (u, v) é segura. Adicioná-la não criará um ciclo. Então:
       - Adicione a aresta (u, v) à sua Árvore Geradora Mínima.
       - Junte (una) as duas árvores que continham u e v em uma única árvore.
     - **Se JÁ pertencem à mesma árvore:** Descarte a aresta (u, v). Adicioná-la criaria um ciclo.

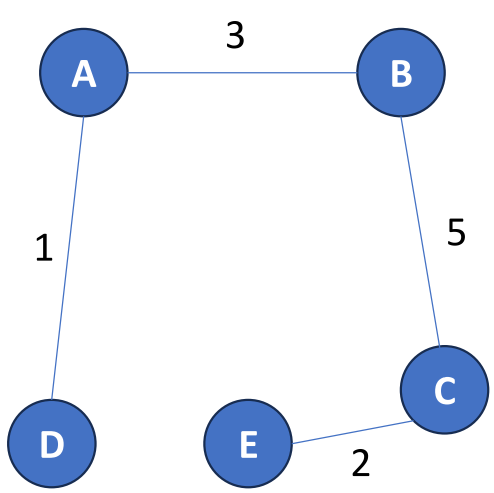
Kruskal passo a passo

4. **Condição de Parada:** O algoritmo termina quando a Árvore Geradora Mínima tiver **n - 1** arestas, onde n é o número de vértices. Nesse ponto, todos os vértices estarão conectados em uma única árvore.

Exemplo Prático



AGM



Componentes = {A}, {B}, {C}, {D}, {E}

A e D estão em componentes diferentes.

Componentes = {A, D}, {B}, {C}, {E}

C e E estão em componentes diferentes.

Componentes = {A, D}, {B}, {C,E}

A e B estão em componentes diferentes.

Componentes = {A, D, B}, {C,E}

B e D estão no mesmo componente

B e C estão em componentes diferentes. Unir {A,B,D} e {C, E}

Componentes = {A, D, B, C, E}

Todos os vértices conectados! Fim!

Atividade 1

- Crie o algoritmo para implementar o código de Kruskal.
- Adaptação do Grafo Esparso para conter pesos