# Árvore Geradora Mínima

Thierson Couto Rosa

thierson@inf.ufg.br

Humberto José Longo

longo@inf.ufg.br

Instituto de Informática - INF/UFG

# Árvore Geradora Mínima

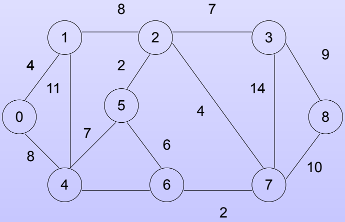
Seja $G = (V, A)$ um grafo não direcionado, valorado, onde $V$ é o conjunto de vértices de $G$ e $A$ é o conjunto de arestas, $A = \{(a, b) \,|a, b \in V\}$. Seja *peso* : $E \rightarrow \mathbb{R}$ uma função que corresponde ao peso de uma dada aresta $(a, b)$ de $A$.
Uma *árvore geradora mínima* $T = (V, A_T)$, $A_T \subseteq A$ é um subgrafo acíclico conectado de $G$, tal que :

$$\sum_{i=1}^{|A_T|} peso(e_i)$$

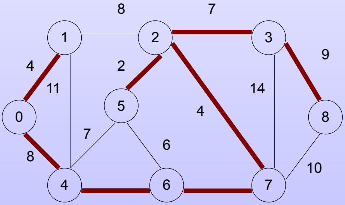
é a menor possível.

# Exemplo

Seja $G$ o seguinte grafo




Uma árove geradora mínima correspondente a G:



# Algoritmo Genérico Guloso

- O algoritmo aumenta a árvore geradora, uma aresta por vez.
- O algoritmo mantém um conjunto de aresta $A_T$ e mantém o seguinte laço invariante:

  Antes de cada iteração $A_T$ é um subconjunto de alguma árvore geradora mínima.

- Em cada etapa escolhe-se uma aresta $(a, b)$, tal que $A_T = A_T \cup \{(a, b)\}$ continue sendo um subconjunto de uma árvore geradora mínima para $G$.
- A aresta $(a, b)$ é denominada *aresta segura*.

# Pseudo-código do Algoritmo Genérico

$AGM - genérico$ (G, peso)

1 $A_T \leftarrow \emptyset$;

2 **enquanto** $A_T$ *não formar uma árvore geradora* **faça**

3 encontrar uma aresta $(a, b) \in A$ em $G$ que seja segura para $A_T$
$A_T \leftarrow A_T \cup \{a, b\}$

4 **fim**

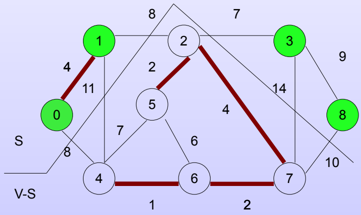
5 retone $A_T$;

# Corte de um Grafo

Dado um grafo não direcionado $G = (V, A)$, um *corte*$(S, V - S)$ é uma partição de $V$. Definições Relacionadas:

- Uma aresta $(a, b) \in A$ *cruza* o *corte*$(S, V - S)$, se: $(a \in S \wedge b \in V - S) \vee (b \in S \wedge a \in V - S)$
- Um *corte*$(S, V - S)$ *respeita* um conjunto $A_T$ de arestas, se nenhuma aresta $a \in A_T$ cruza o corte.
- Uma aresta $(a, b) \in A$ que cruza um *corte*$(S, V - S)$ é dita *leve* se *peso*$((a, b))$ é o menor peso entre todas as arestas que cruzam *corte*$(S, V - S)$.

# Exemplo

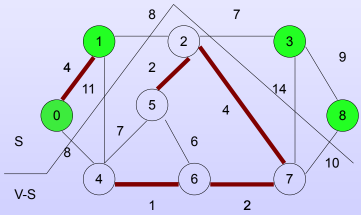



- O Corte respeita $A_T = \{(0.1), (2,5), (2,7), (4,6), (6,7)\}$
- A aresta $(2,3)$ é a única aresta leve que cruza o corte.

## Como Obter uma Aresta Segura

*Teorema:* Seja um grafo não direcionado $G(V, A)$ e uma função *peso* que associa cada aresta $a \in A$ a um número real e seja $A_T$ um subconjunto de uma árvore geradora mínima de $G$. Seja $corte(S, S - V)$ um corte de $G$ que respeita $A_T$. Uma aresta segura para $A_T$ corresponde a uma aresta leve que cruza $corte(S, S - V)$.

# Exemplo




- A aresta $(2,3$ é uma aresta segura para $A_T = \{(0.1),(2,5),(2,7),(4,6),(6,7)\}$
- A aresta $(2,3)$ é a única aresta leve que cruza o corte.

# Algoritmos que se baseiam no Algoritmo Genérico

Dois algoritmos para se obter uma árvore geradora mínima utilizam modo distintos para obter arestas seguras:

- Algoritmo de Kruskal.
- Algoritmo de Prim.

# Algoritmo de Kruskal

- O conjunto $E_T$ corresponde a um floresta.
- Em cada iteração, uma aresta segura é a aresta de menor peso que liga dois componentes distintos (duas árvores), formando uma nova subárvore.
- O algoritmo utiliza uma estrutura de dados para armazenar conjunto de conjuntos disjuntos (*union-find*), onde cada conjunto corresponde a uma árvore da floresta.
- Quando uma aresta leve é encontrada, o algoritmo efetua a união das duas árvores ligadas pela aresta leve.
- Ao final, se o grafo original for conectado, haverá um conjunto único de arestas que corresponde à uma árvore geradora mínima para o grafo.

# Algoritmo de Kruskal

Kruskal $(G(V, A), peso)$

$A_T \leftarrow \emptyset$;<br>
**para** $v \in V$ **faça**<br>
    crie um conjunto unitário contendo $x$<br>
**fim**<br>
Ordene as arestas de $A$ em ordem crescente de peso;<br>
**para** $(u, v) \in A$, *em ordem crescente de peso* **faça**<br>
    **se** $findSet(u) \neq findSet(v)$ **então**<br>
        $A_T \leftarrow A \cup \{(u, v)\}$ Union(u,v)<br>
    **fim**<br>
**fim**<br>
retorne $A_T$