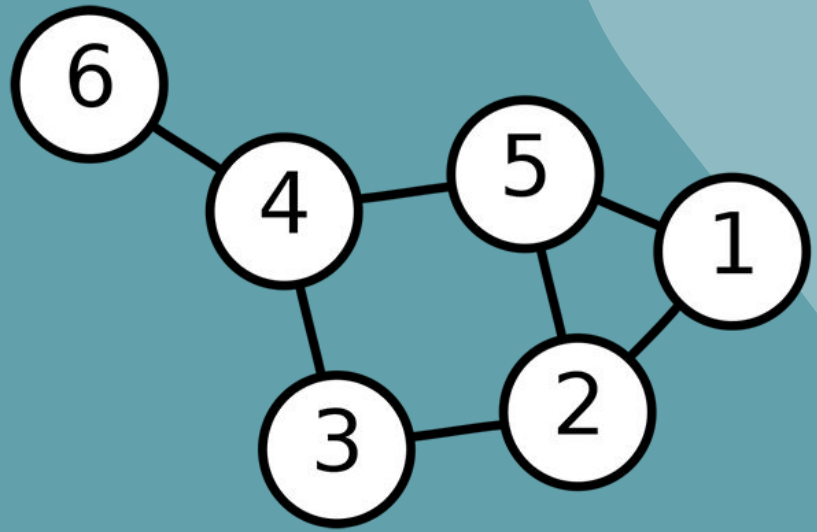
GRAFOS:
Árvores Espalhadas Mínimas



Árvores Espalhadas Mínimas

Definição:

- Subgrafo conectado acíclico que liga todos os vértices do grafo

Nome em inglês: **Spanning Tree**

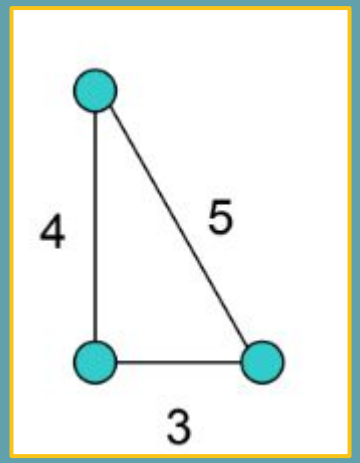
Outras traduções

- Uma tradução comum é “Árvores Geradoras” ou “Árvores de Amplitude”

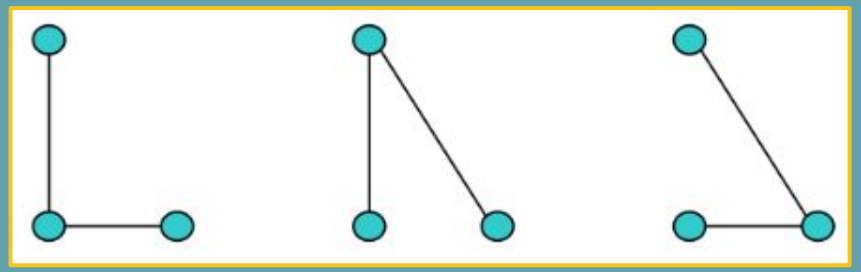
Árvores Espalhadas Mínimas

Um grafo pode ter várias Árvores Espalhadas

Grafo



Árvores Espalhadas:



Árvores Espalhadas Mínimas

Em um grafo valorado, é a árvore espalhada que possui menor custo/peso dentre todas

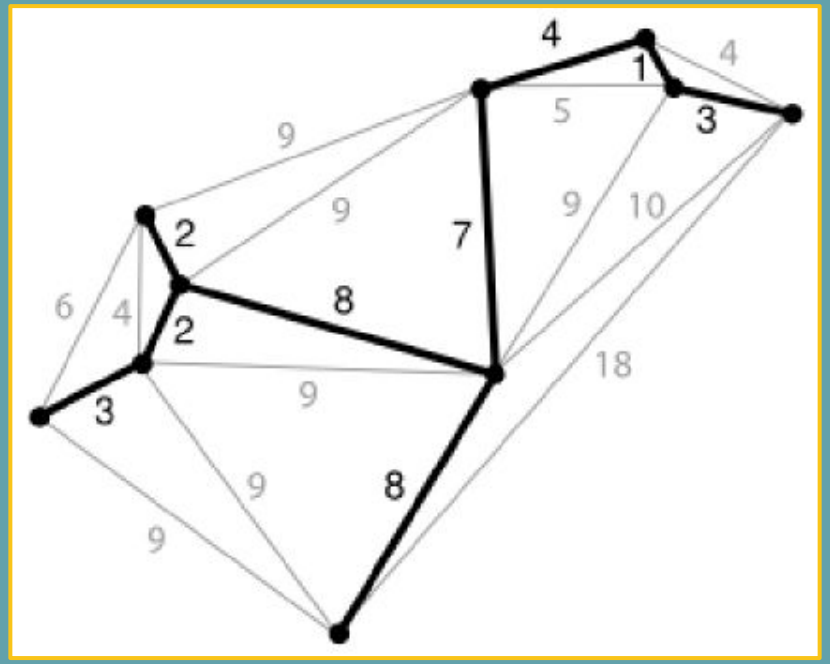
Por que “espalhada”?

- Porque é um subgrafo que tem o mesmo conjunto de vértices do grafo original

Por que “mínima”?

- Porque possui menor custo/peso dentre todas

Árvores Espalhadas Mínimas



Custo: 1 + 2 + 2 + 3 + 3 + 7 + 8 + 8 = **34**

Algoritmo de Kruskal

Algoritmo de Kruskal

MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
    MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
    if (FIND-SET(u) ≠ FIND-SET(v))
        adiciona (u,v) a A
        UNION(u,v)

Algoritmo de Kruskal

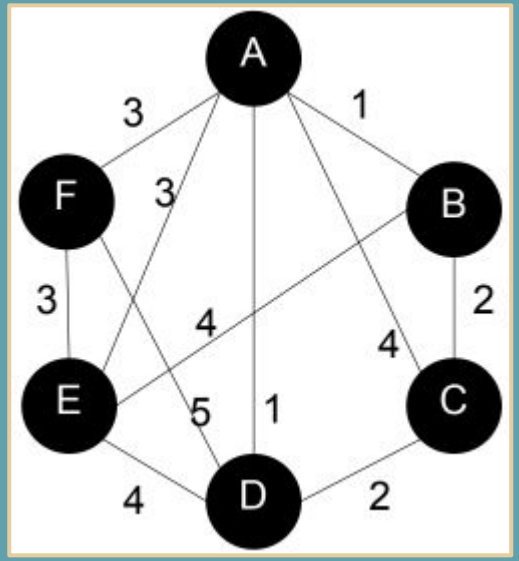
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
if (FIND-SET(u) ≠ FIND-SET(v))
adiciona (u,v) a A
UNION(u,v)



{{a}, {b}, {c}, {d}, {e}, {f}}

Algoritmo de Kruskal

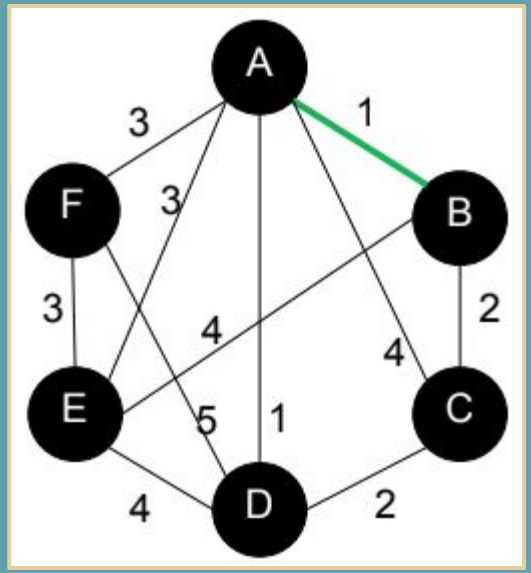
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
if (FIND-SET(u) ≠ FIND-SET(v))
adiciona (u,v) a A
UNION(u,v)



{{a, b}, {c}, {d}, {e}, {f}}

Algoritmo de Kruskal

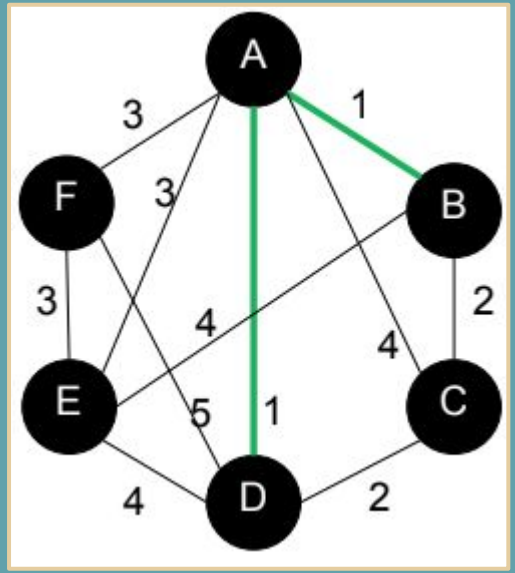
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
if (FIND-SET(u) ≠ FIND-SET(v))
adiciona (u,v) a A
UNION(u,v)



{{a, b, d}, {c}, {d}, {e}, {f}}

Algoritmo de Kruskal

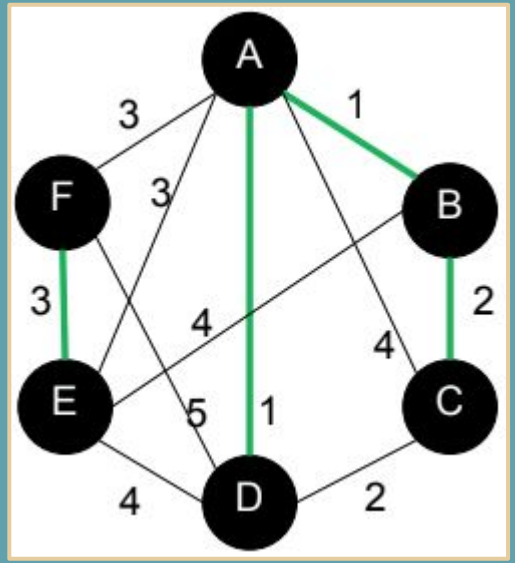
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
    MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
    if (FIND-SET(u) ≠ FIND-SET(v))
        adiciona (u,v) a A
        UNION(u,v)



{{a, b, d, c}, {e}, {f}}

Algoritmo de Kruskal

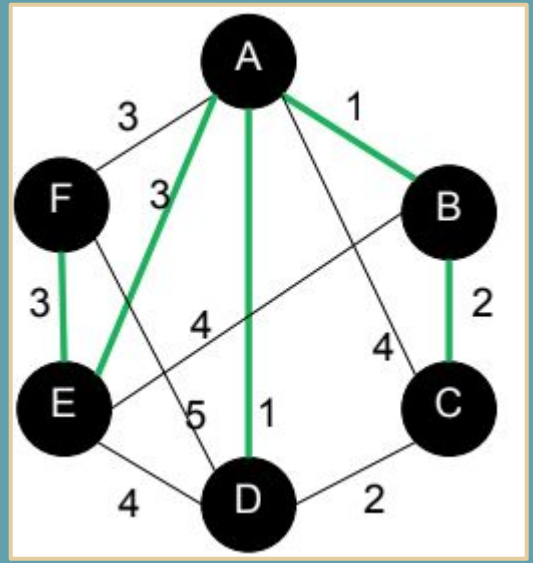
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
if (FIND-SET(u) ≠ FIND-SET(v))
adiciona (u,v) a A
UNION(u,v)



{{a, b, d, c, e, f}}

Algoritmo de Kruskal

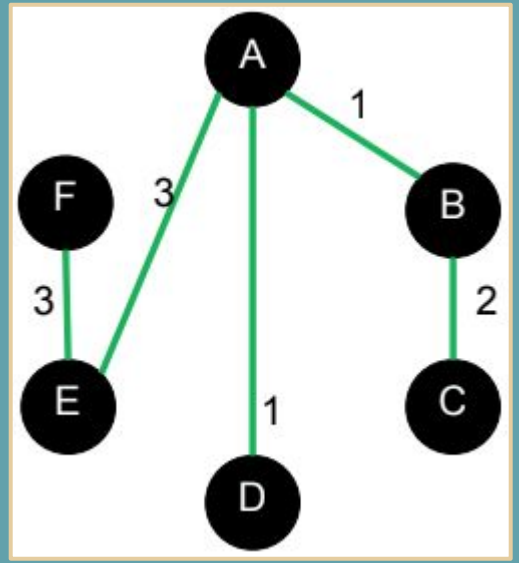
MST-KRUSKAL (G)

A = VAZIO;

para cada vértice v
MAKE-SET (v);

ordena as arestas;

para cada aresta (u,v), em ordem crescente
if (FIND-SET(u) ≠ FIND-SET(v))
adiciona (u,v) a A
UNION(u,v)



{{a, b, d, c, e, f}}

Algoritmo de Prim

Algoritmo de Prim

MST-PRIM(G, r)

apenas o vértice r é considerado conectado a A;

enquanto não conectar todo vértice

encontra o vértice v mais barato de ser ligado a A;

adiciona em A, a aresta que liga v;

atualiza custo de ligar A a outros vértices;

retorna A;

OBRIGADO!

DÚVIDAS?

marcos.azevedo@unicap.br