Árvore Geradora Mínima
(*Minimum Spanning Tree*)

Árvore Geradora Mínima

- Problema (objetivo):
  - a partir de um grafo de elementos conectados, identificar a cobertura que permite ligar todos os elementos com menor custo.

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Aplicações:
  - projeto de circuitos eletrônicos → com a necessidade de conectar pinos de diversos componentes, é importante identificar uma configuração que minimiza o arranjo de fios.
  - projeto de canalização de água → com a ligação de diversos pontos, é necessários minimizar a distância entre eles.

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Problema (continuação):
  - grafo não dirigido;
  - G = (V, A) → onde V representa o conjunto de vértices, e A o conjunto de arestas;
  - para cada aresta entre dois vértices “u” e “v” (A(u, v)), existe um peso w(u, v) que pode ser distância, custo, ou outro fator relevante.
  - solução: conjunto $S \subseteq A$, que conecte todos os vértices com peso total minimizado e calculado como:

$$w(S) = \sum_{w(u,v) \in S} w(u,v)$$

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Exemplos de uso em problemas:
  - Projeto de redes
    - redes telefônicas, redes elétricas, redes hidráulicas, estradas, redes de TV a cabo, etc.

    - Redes telefônicas:
      - uma multinacional, com diversos escritórios em diferentes locais, precisa que todas as suas linhas telefônicas sejam conectadas; os custos de conexão são diferentes entre os diversos locais. É desejável então estabelecer a rede com o menor custo possível.

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Exemplos de uso em problemas (continuação):
  - Projeto de redes
    - Redes hidráulicas:
      - existem diversos locais em uma cidade para conectar por meio de uma rede hidráulica; a rede deve conectar todos os pontos. Deve-se garantir que as conexões são as menores possíveis.

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Exemplos de uso em problemas (continuação):
  - Outros exemplos conhecidos:
    - processamento de imagens para detecção de faces;
    - sequenciamento de proteínas;
    - simulação de partículas em fluxo de fluídos;
    - simulação de protocolos de Internet.

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- Abordagens:
  - Força bruta
    - estimativa para *n* vértices: exponencial ou equivalente.

O que fazer então?

Fonte: material prof. Sandro Rigo

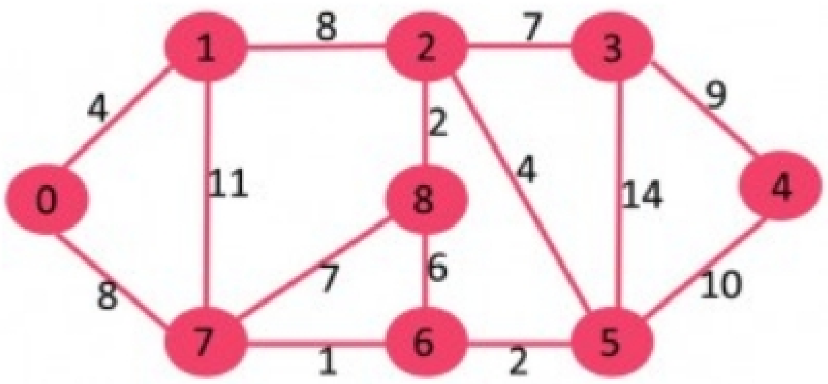
Árvore Geradora Mínima

- Abordagens:
  - Algoritmos gulosos
    - permite levar a complexidade para algo na escala de O(A lg V) para um grafo G = (A, V).
    - principais algoritmos
      - Kruskal
      - Prim

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

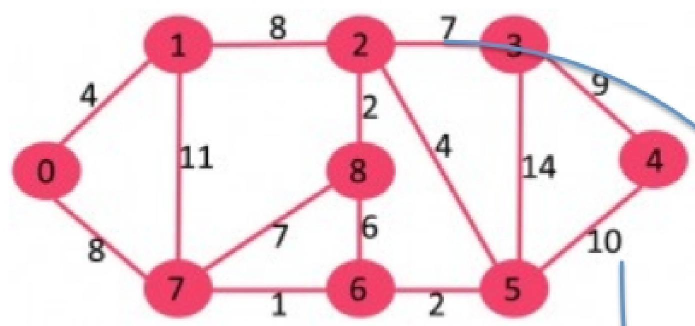
- **Kruskal –** visão geral
  - Etapa inicial:
    - dado G = (A, V), ordenar A pelo peso, em ordem crescente.
    - resultado: tabela com peso, origem e destino.
    - exemplo:



Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo



Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

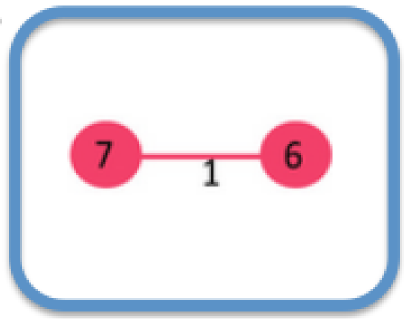
Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

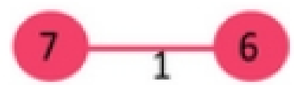


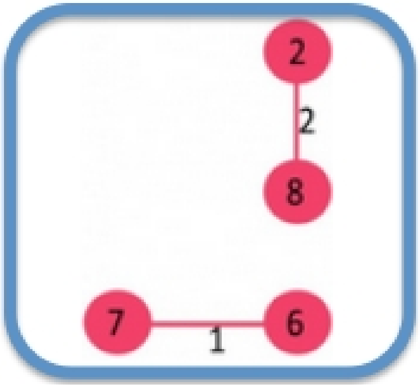
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



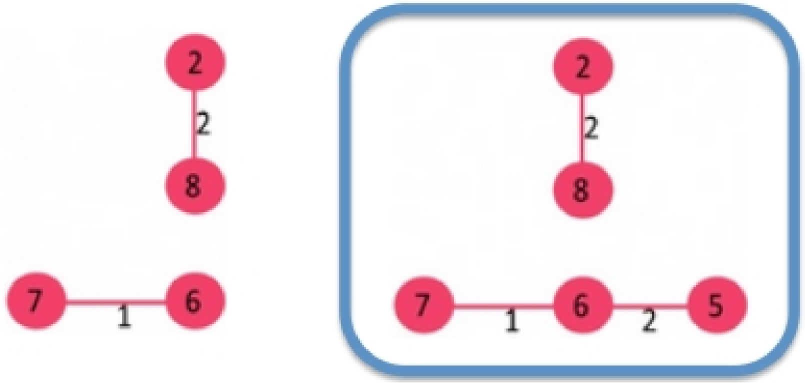


Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

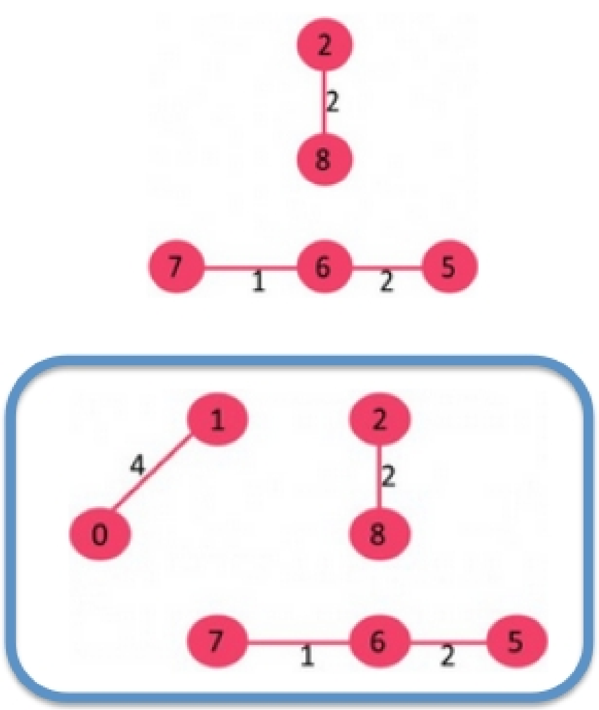


Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

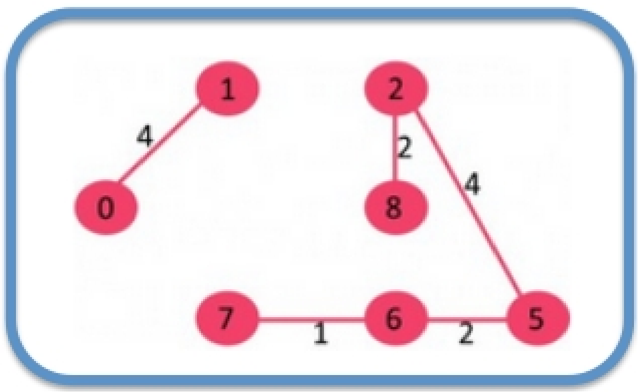


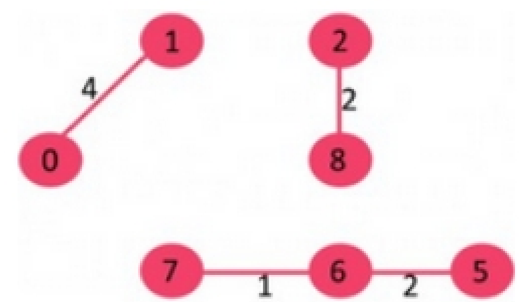
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



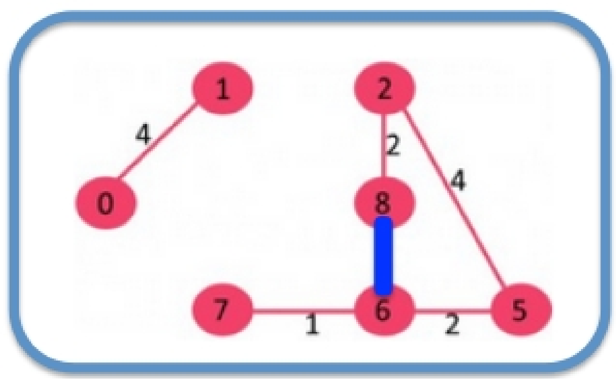


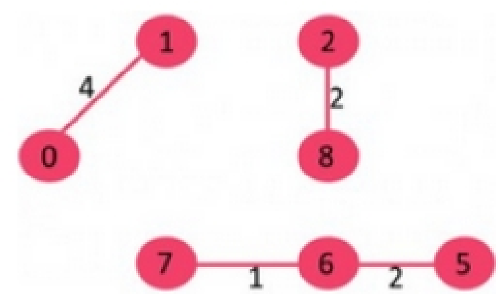
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5





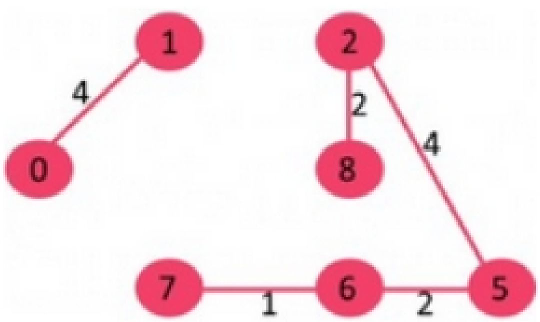
Fonte: material prof. Sandro Rigo

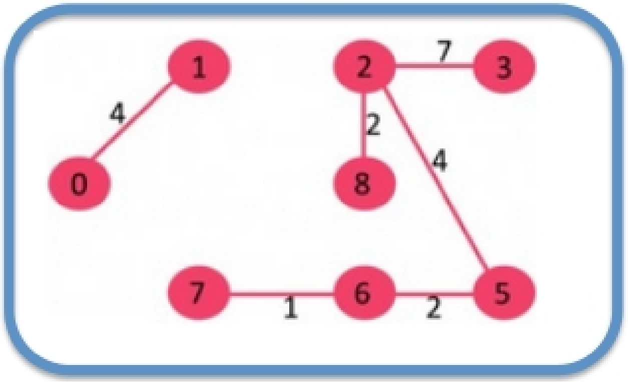
UNISINOS

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



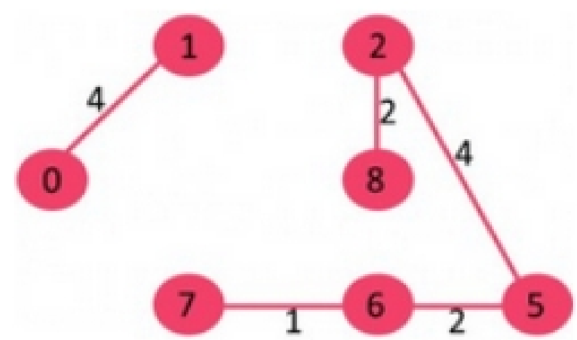


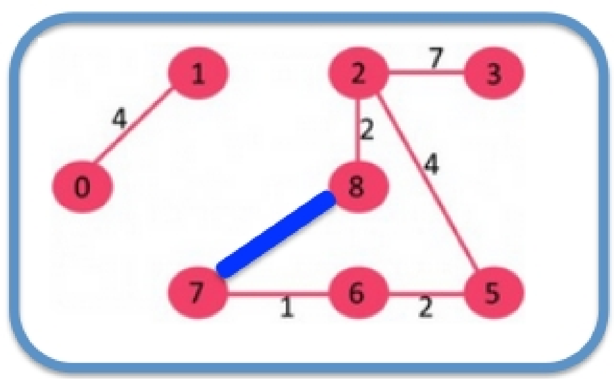
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



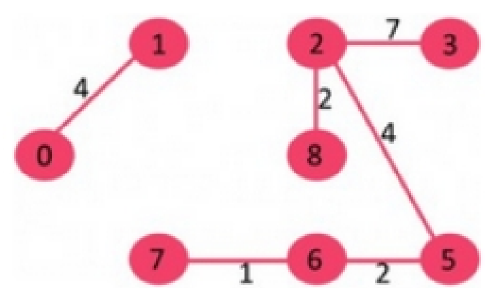


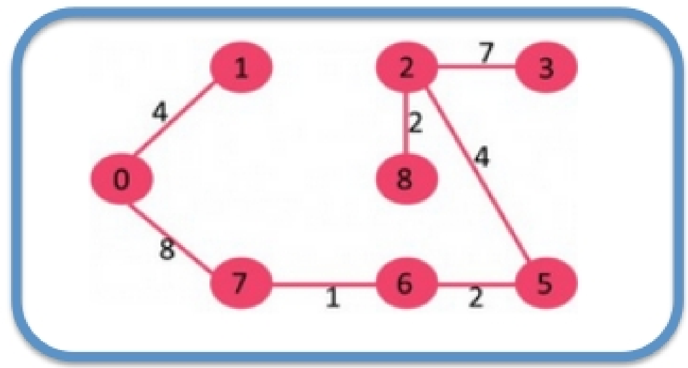
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



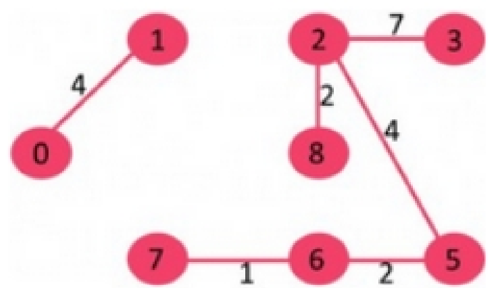


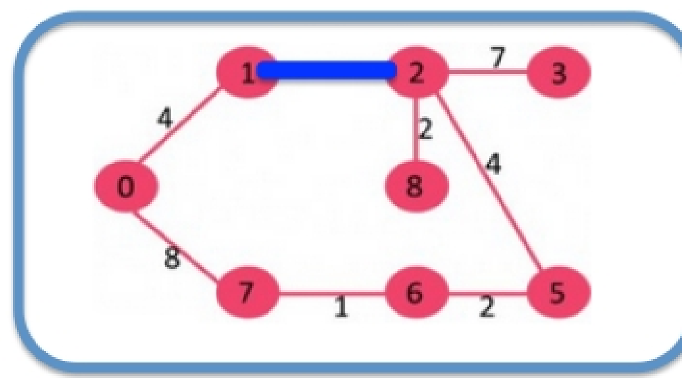
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



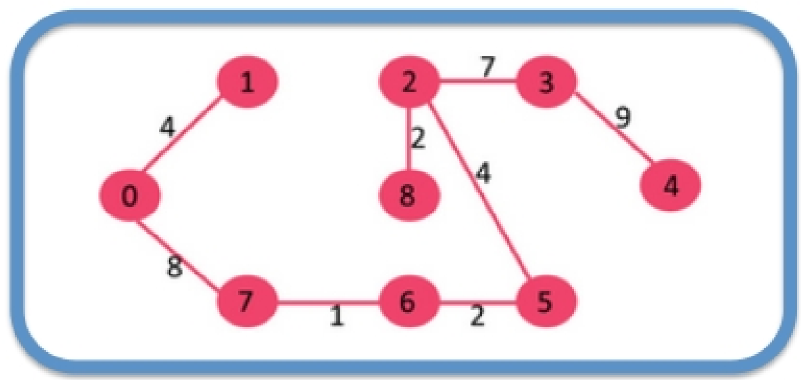


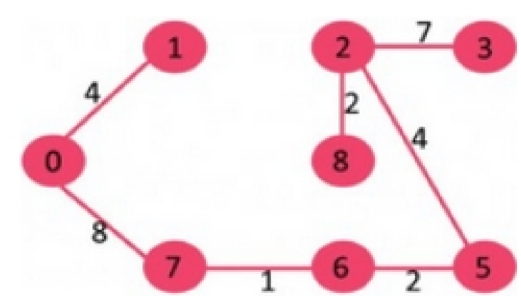
Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

Peso	Origem	Destino
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

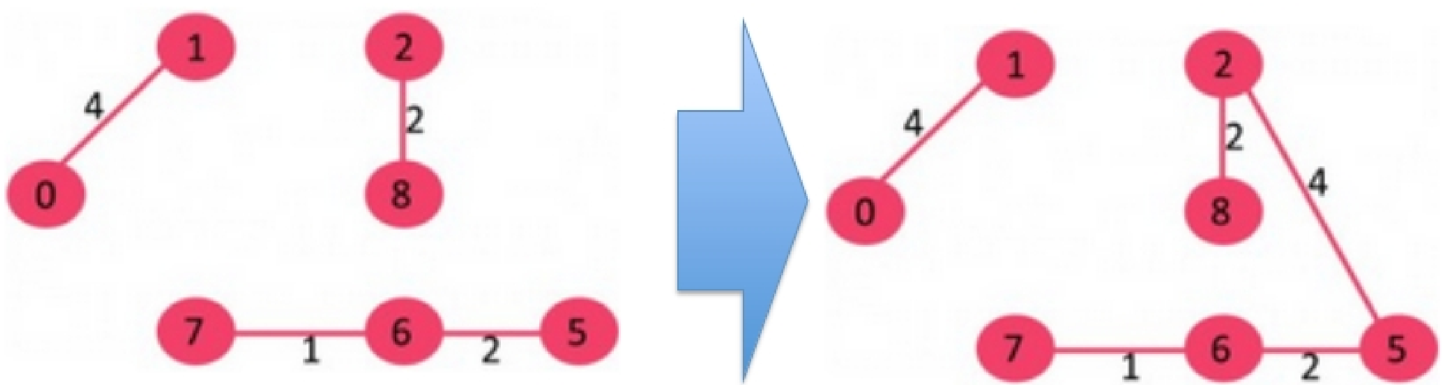




Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** elementos da estratégia gulosa
  - Representar a solução “S” como uma floresta com todos os vértices de G = (V, A).
  - A aresta segura é sempre a aresta de peso mínimo do grafo G, que conecte dois componentes distintos.



Fonte: material prof. Sandro Rigo

UNISINOS

Árvore Geradora Mínima

- **Kruskal –** algoritmo guloso

```
MST-KRUSKAL(G, w)
A = Ø
for each vertex v ∈ G.V
    MAKE-SET(v)
sort the edges of G.E into nondecreasing order by weight w
for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
    if FIND-SET(u) ≠ FIND-SET(v)
        A = A ∪ {(u, v)}
        UNION(u, v)
return A
```

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal –** visão geral → exemplo

```
MST-KRUSKAL(G, w)
  A = ∅
  for each vertex v ∈ G.V
      MAKE-SET(v)
  sort the edges of G.E into nondecreasing order by weight w
  for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
      if FIND-SET(u) ≠ FIND-SET(v)
          A = A ∪ {(u, v)}
          UNION(u, v)
  return A
```

Preparação dos dados

Percorre todo o conjunto de arestas

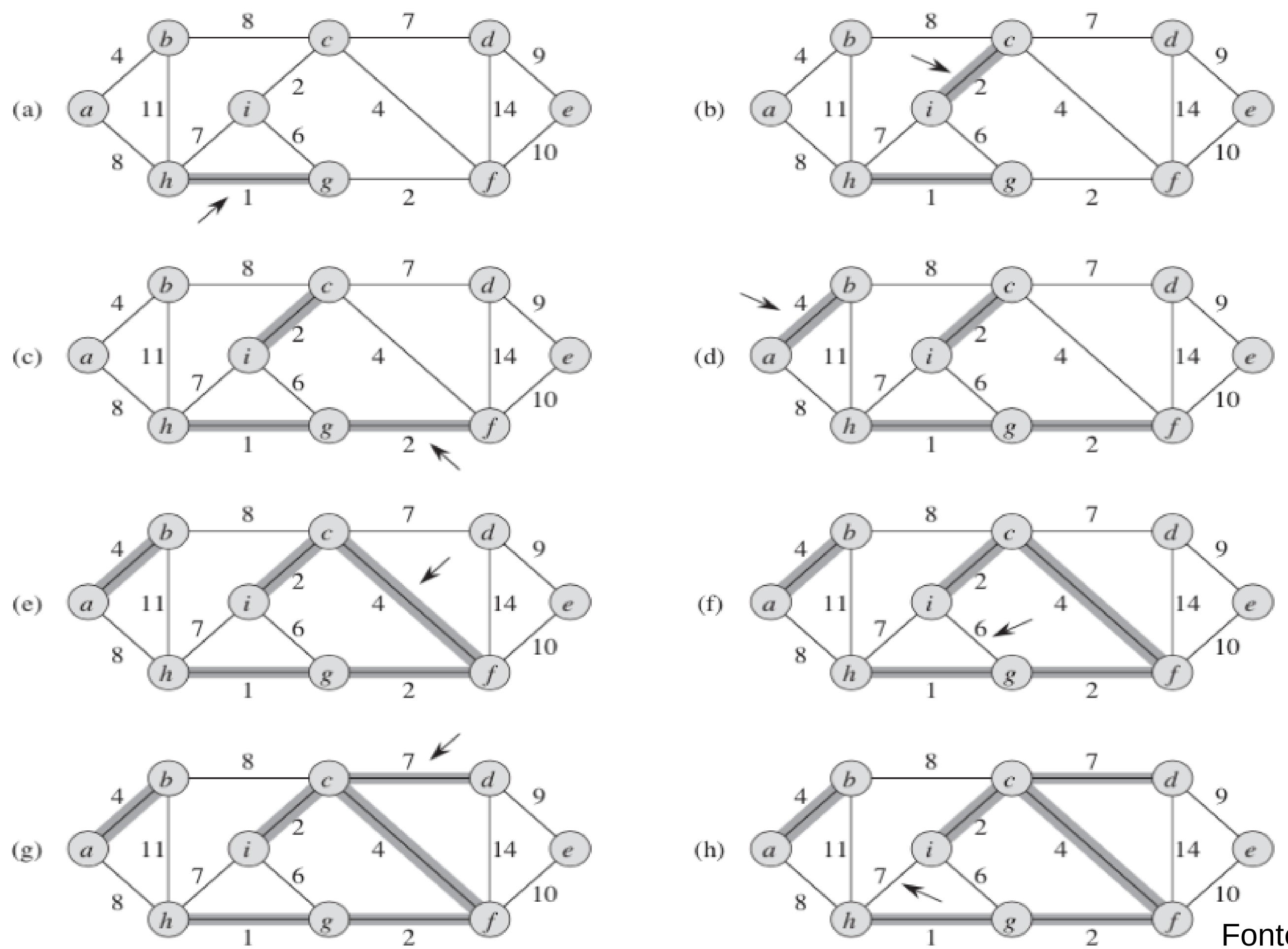
Executa uma escolha gulosa

Atualiza o conjunto de soluções

Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

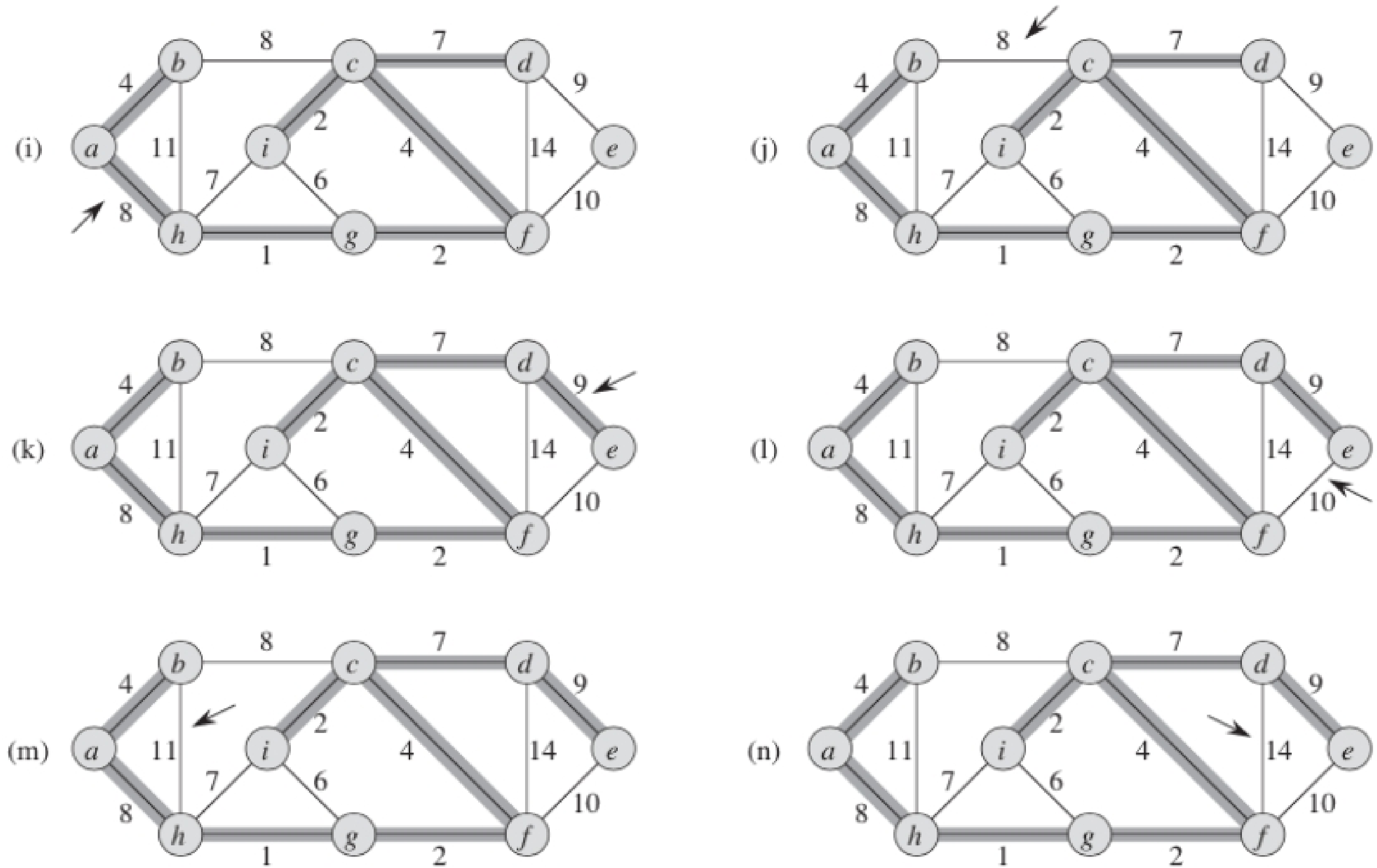
- **Kruskal**



Fonte: material prof. Sandro Rigo

Árvore Geradora Mínima

- **Kruskal**



Fonte: material prof. Sandro Rigo

Caminho mais Curto em Grafo
(*Shortest-Path Problem*)

Caminho mais curto em grafo

- Problema (objetivo):
  - Encontrar o menor caminho entre um ponto e outro em um grafo de elementos conectados.

Fonte: Cormen *et al.*

Caminho mais curto em grafo

- Aplicações:
  - envio de pacotes de um ponto a outro pela rede.
  - roteamento de veículos em um sistema de transporte.
  - planejamento de trajetórias de veículos aéreos não tripulados.
  - ...

Fonte: Cormen *et al.*

Caminho mais curto em grafo

- Problema (continuação):
  - grafo dirigido;
  - G = (V, A) → onde V representa o conjunto de vértices, e A o conjunto de arestas;
  - para cada aresta entre dois vértices “u” e “v” (A(u, v)), existe um peso w(u, v) (não negativo) que pode ser distância, custo, ou outro fator relevante.
  - solução: conjunto $S \subseteq A$, que conecte todos os vértices do grafo resultante com peso total minimizado (menor caminho).

Fonte: Cormen *et al.*

Caminho mais curto em grafo

- **Dijkstra** – problema:
  - Encontrar o menor caminho entre um ponto e qualquer outro em um grafo de elementos conectados.
  - Resolve o problema *single-source shortest paths.*

Fonte: Cormen *et al.*

Caminho mais curto em grafo

- **Dijkstra** – visão geral
  - Mantém um conjunto *S* de vértices, com o menor caminho a partir de um vértice inicial *s*.
  - Repetidamente seleciona um vértice $u \in V - S$ com a estimativa do menor caminho, adiciona *u* a *S*, e **relaxa** todas as arestas que partem de *u*.
    - relaxamento: processo que consiste em testar se é possível melhorar o menor caminho de *u* até *v*.
      - inicializa o vértice inicial com peso 0;
      - inicializa os demais vértices com peso infinito (no início do algoritmo, não sabemos a distância entre o vértice inicial e os demais vértices do grafo);
      - nas demais iterações, atualiza os vértices com o valor correspondente ao menor caminho.

Fonte: Cormen *et al.*

Caminho mais curto em grafo

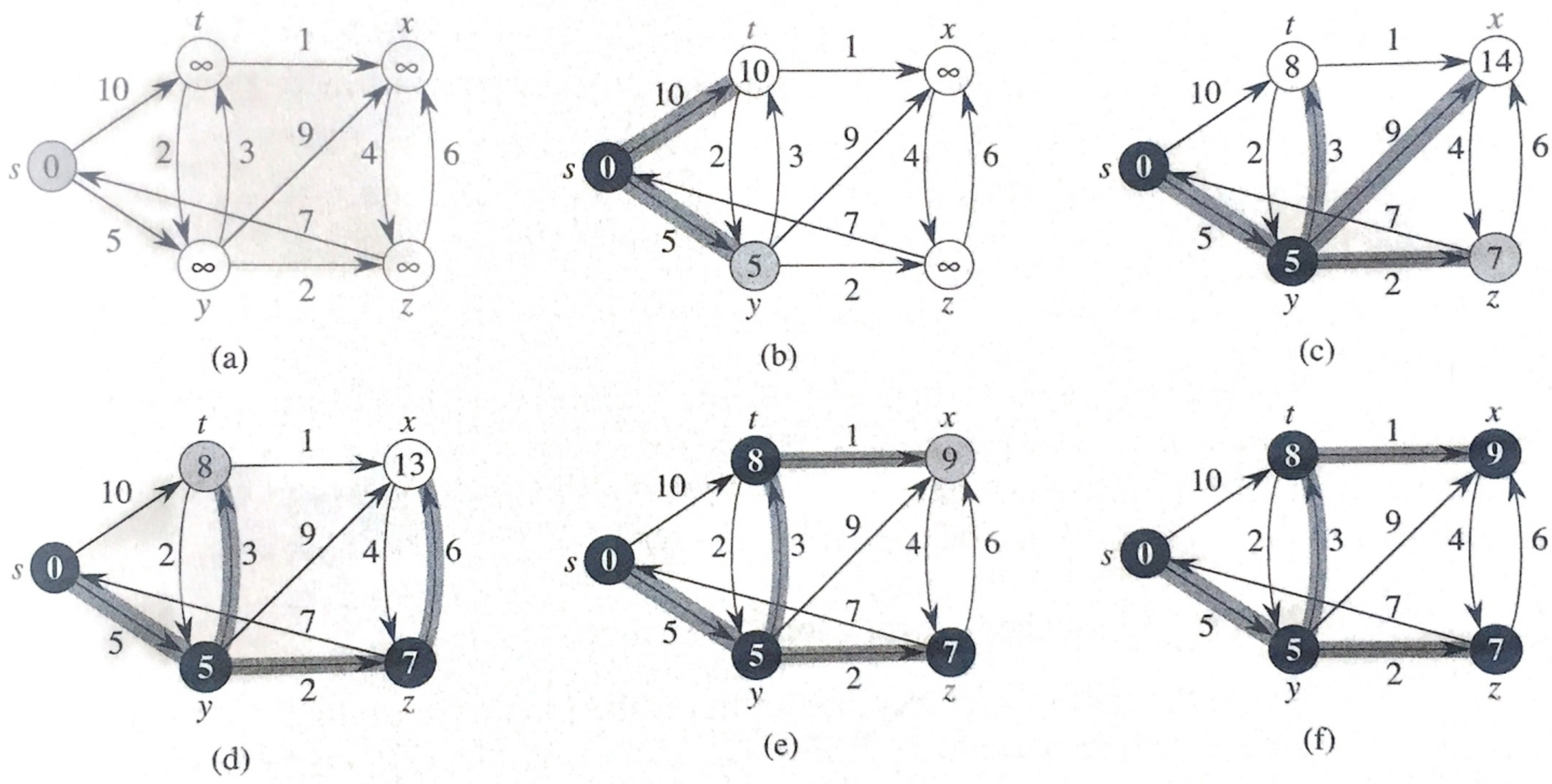
- **Dijkstra** – algoritmo guloso

```
DIJKSTRA(G, w, s)
    INITIALIZE-SINGLE-SOURCE(G, s)
    S = ∅
    Q = G.V
    while Q ≠ ∅
        u = EXTRACT-MIN(Q)
        S = S ∪ {u}
        for each vertex v ∈ G.Adj[u]
            RELAX(u, v, w)
```

Fonte: Cormen *et al.*

Caminho mais curto em grafo

- **Dijkstra**



Fonte: Cormen *et al.*

Caminho mais curto em grafo

- **Dijkstra**
  - Complexidade (continuação):
    - O tempo de execução do algoritmo vai depender de como a fila de prioridades será implementada:
      - *array* ordenado pelos vértices: $O(V^2 + A) \rightarrow O(V^2)$
      - *binary min-heap*: $O((V + A)\lg V) \rightarrow O(A \lg V)$
      - *fibonacci* heap: $O(V \lg V + A)$

Fonte: Cormen *et al.*

Mais exemplos...

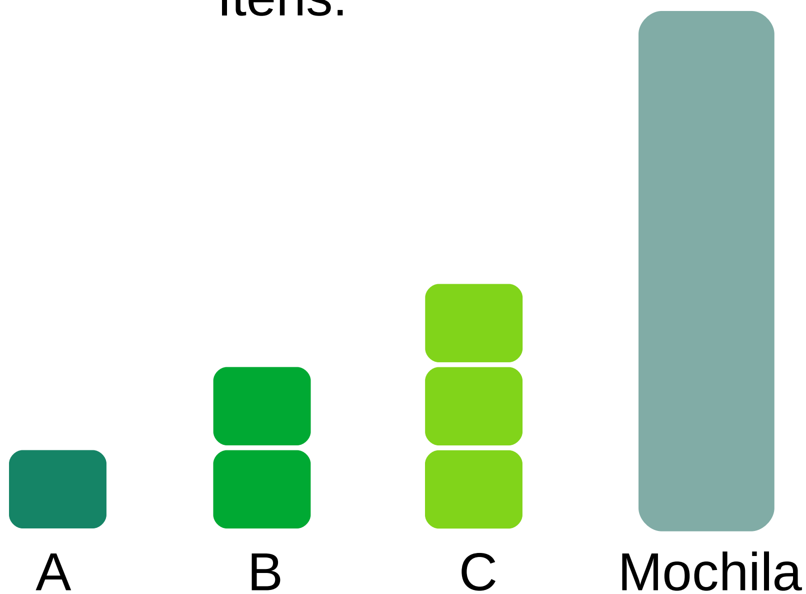
Mais exemplos...

- O problema da mochila
  - Carregar *n* itens, cada um com peso *p* e valor *v*, de modo a carregar a carga mais valiosa em uma mochila de capacidade de carga limitada em C. O item deve ser carregado inteiro, sem fracionar.

Fonte: material prof. Sandro Rigo

O problema da mochila

Itens:



A → 10kg, R$ 60,00 (R$ 6,00 por kg)

B → 20kg, R$ 100,00 (R$ 5,00 por kg)

C → 30kg, R$ 120,00 (R$ 4,00 por kg)

Mochila → 50kg

Fonte: material prof. Sandro Rigo

O problema da mochila

Itens:

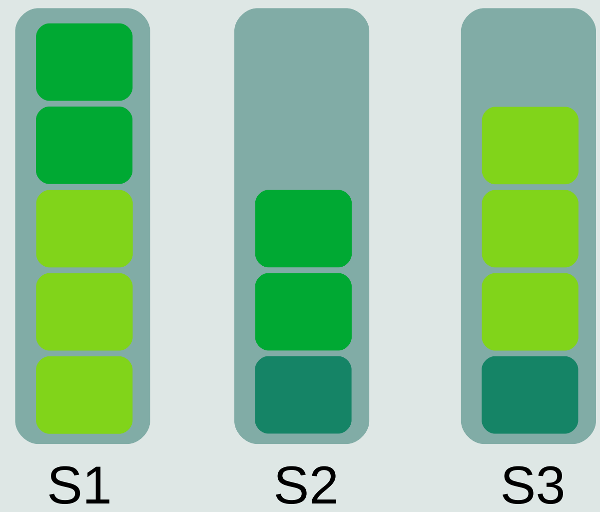
A → 10kg, R$ 60,00 (R$ 6,00 por kg)

B → 20kg, R$ 100,00 (R$ 5,00 por kg)

C → 30kg, R$ 120,00 (R$ 4,00 por kg)

Mochila → 50kg

Combinações



S1 = B + C → R$ 100,00 + R$ 120,00 = R$ 220,00

S2 = A + B → R$ 60,00 + R$ 100,00 = R$ 160,00

S3 = A + C → R$ 60,00 + 120,00 = R$ 180,00

Fonte: material prof. Sandro Rigo

O problema da mochila

Itens:

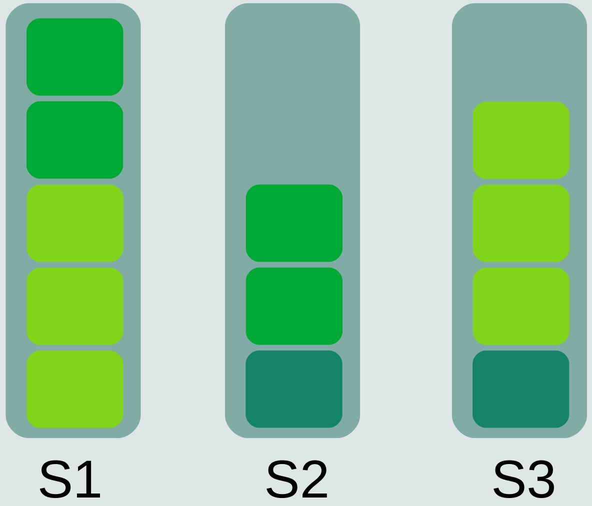
A → 10kg, R$ 60,00 (R$ 6,00 por kg)

B → 20kg, R$ 100,00 (R$ 5,00 por kg)

C → 30kg, R$ 120,00 (R$ 4,00 por kg)

Mochila → 50kg

Combinações



S1 = B + C → R$ 100,00 + R$ 120,00 = R$ 220,00

S2 = A + B → R$ 60,00 + R$ 100,00 = R$ 160,00

S3 = A + C → R$ 60,00 + 120,00 = R$ 180,00

Fonte: material prof. Sandro Rigo

O problema da mochila

- Estratégia gulosa possível (resumida):
  - Levar o item de maior valor por peso.
  - Repetir o processo se houver espaço.

Fonte: material prof. Sandro Rigo

O problema da mochila

Itens:

A → 10kg, R$ 60,00 (R$ 6,00 por kg)

B → 20kg, R$ 100,00 (R$ 5,00 por kg)

C → 30kg, R$ 120,00 (R$ 4,00 por kg)

Mochila → 50kg

Combinações

S1 = B + C → R$ 100,00 + R$ 120,00 = R$ 220,00

S2 = A + B → R$ 60,00 + R$ 100,00 = R$ 160,00

S3 = A + C → R$ 60,00 + 120,00 = R$ 180,00

Fonte: material prof. Sandro Rigo

O problema da mochila

Itens:

- A → 10kg, R$ 60,00 (R$ 6,00 por kg)
- B → 20kg, R$ 100,00 (R$ 5,00 por kg)
- C → 30kg, R$ 120,00 (R$ 4,00 por kg)
- Mochila → 50kg

Combinações

S1 = B + C → R$ 100,00 + R$ 120,00 = R$ 220,00

S2 = A + B → R$ 60,00 + R$ 100,00 = R$ 160,00

S3 = A + C → R$ 60,00 + 120,00 = R$ 180,00

Fonte: material prof. Sandro Rigo

O problema da mochila

- Com o algoritmo guloso, a melhor escolha não pode ser obtida. Nesse caso, é preciso avaliar todas as opções antes.

- Mas então, como resolver esse problema?

Leitura Complementar

- Livro *Introduction to Algorithms* (Cormen *et al.*)
  - Capítulo 16 – Greedy Algorithms
  - Capítulo 23 – Minimum Spanning Trees
    - 23.2 – The algorithms of Kruskal and Prim
  - Capítulo 24 – Single Source Shortest Paths
    - 24.3 – Dijkstra's algorithm

Referências Bibliográficas

- Livro *Introduction to Algorithms* (Cormen *et al.*)
  - Capítulo 16 – Greedy Algorithms
  - Capítulo 23 – Minimum Spanning Trees
    - 23.2 – The algorithms of Kruskal and Prim
  - Capítulo 24 – Single Source Shortest Paths
    - 24.3 – Dijkstra's algorithm

- Material desenvolvido com base nos originais do professor Sandro Rigo.