Pesquisa Operacional II

Minimização de Redes: Algoritmo de Kruskal

Prof. Dr. Daniel Caetano

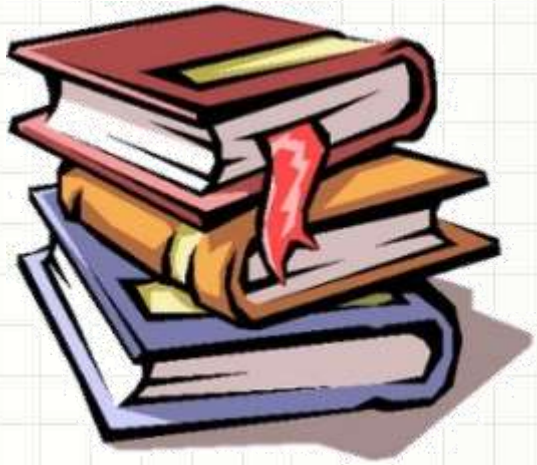
2019 - 1

Objetivos

- Compreender em que situações o algoritmo de Prim não é o mais adequado
- Aprender a determinar árvores geradoras mínimas para esses casos!

- **Atividade Aula 3 – SAVA!**

Material de Estudo



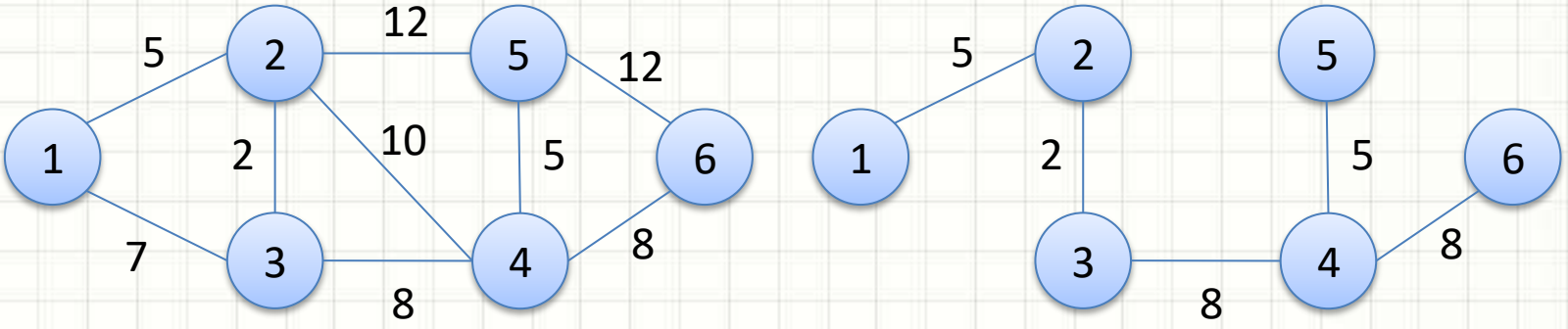
Material	Acesso ao Material
Apresentação	http://www.caetano.eng.br/ (Pesquisa Operacional II – Aula 3)
Minha Biblioteca	Introdução à Pesquisa Operacional (Hillier/Lieberman), Cap. 9, Seção 9.4
Recursos na Web	https://www.ime.usp.br/~pf/algoritmos_para_grafos /index.html#mst

Retomando:

Árvores Geradoras de Custo Mínimo

O que é Árvore Geradora Mínima?

- É uma árvore (sub-grafo onde não há ciclos)
- Que preserva todos os nós originais
- Cuja soma dos arcos tem valor mínimo



Algoritmo de Prim “Simplificado”

1. Escolher o nó ligado ao menor arco e colocar na árvore destino
2. A partir de um dos nós presentes na árvore destino, escolher o nó mais próximo no grafo origem – e que não esteja no destino – e colocar ele e o arco que ligam os nós na árvore destino
3. Repita até que todos os nós estejam na árvore destino

- “Escolhe um nó inicial e vai crescendo a árvore geradora pelas distâncias mais curtas, até que todos os nós estejam na árvore geradora”

Limitações do Algoritmo de Prim

Limitação Computacional (Prim)

- Complexidade Computacional
  - O quanto o tempo de processamento cresce em função do crescimento do "tamanho" do problema
- Complexidade do algoritmo de Prim

$$C = O(v^2)$$

Não fala nada dos arcos!

Vértices	Propoção de Tempo
10	100 milissegundos
100	10 segundos
1.000	17 minutos
10.000	28 horas
100.000	116 dias

Limitação Prática (Prim)

- Prim só funciona em grafos conexos
  - Todos os pontos precisam estar interligados no grafo original
- Embora não seja nosso foco, em alguns casos pode-se querer encontrar várias árvores mínimas simultaneamente
- Em ambos os casos, a solução pode ser o algoritmo de Kruskal

O Algoritmo de Kruskal

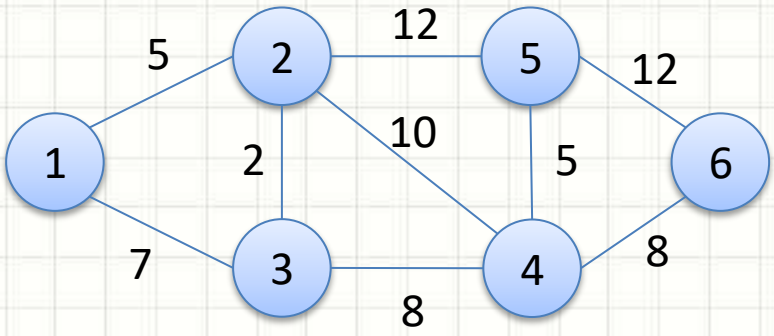
Algoritmo de Kuskal

- O algoritmo de Kruskal é como se segue:
  1. No grafo original **O**, escolha o arco **L** de menor custo que ainda não esteja na árvore destino **D** e nem feche um ciclo na mesma
  2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**
  3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O

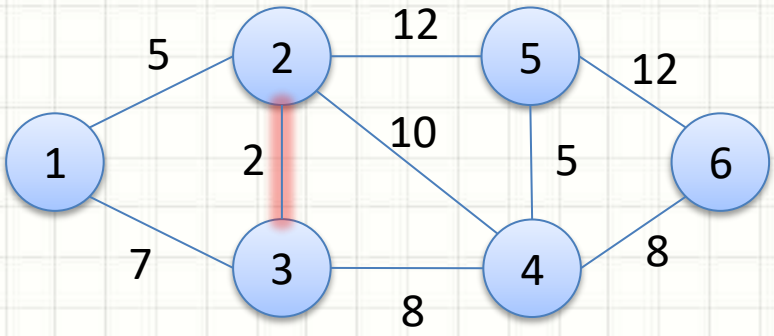


D

Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O

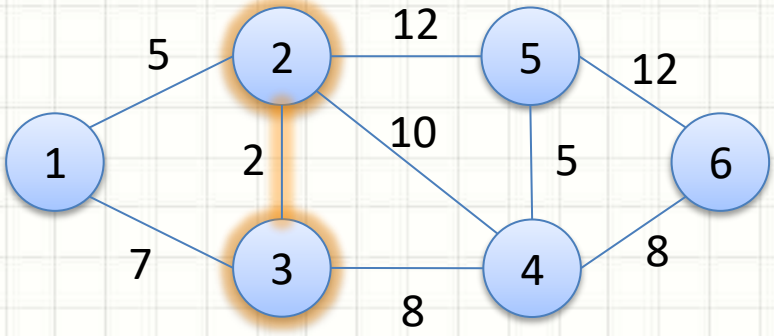


D

Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O

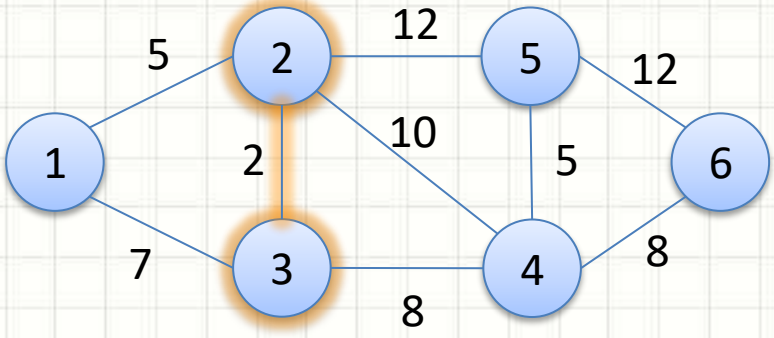


D

Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O

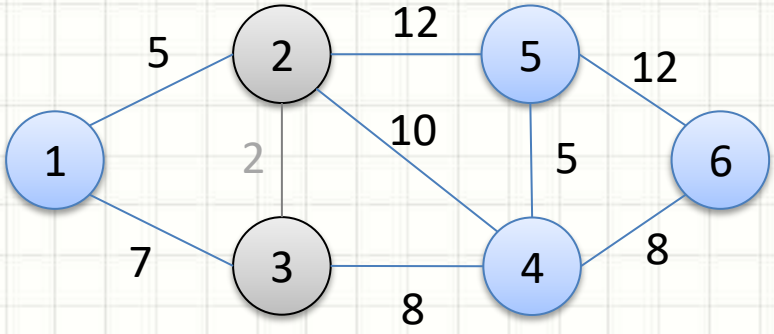


D

Aplicando o Algoritmo de Kuskal

3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

O

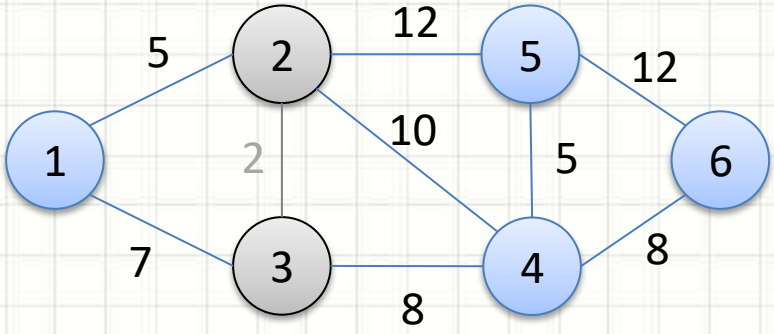


D

Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O

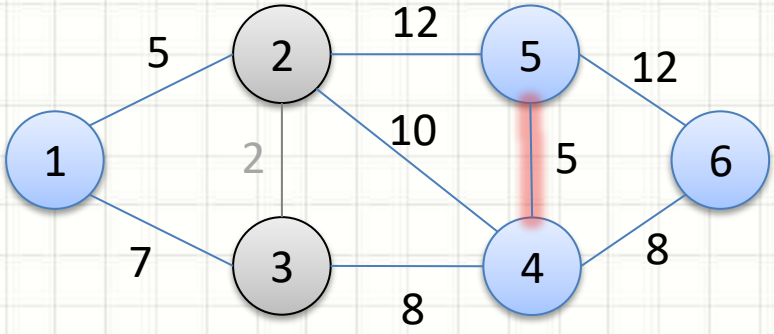


D

Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O

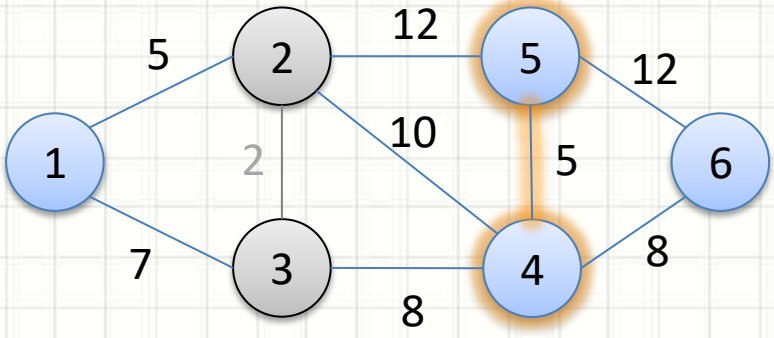


D

Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O

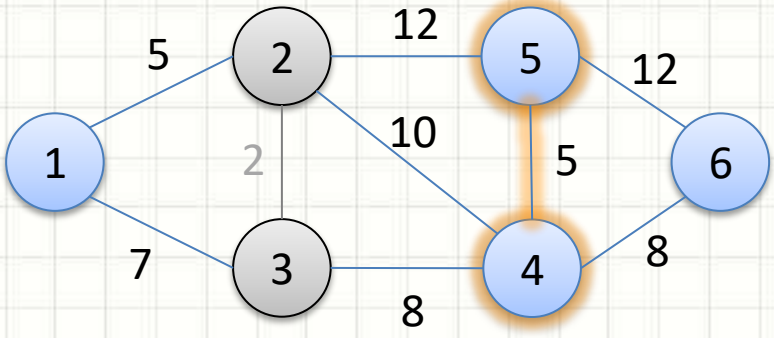


D

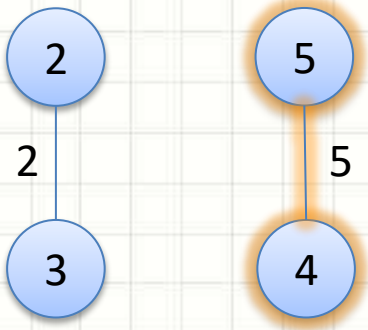
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



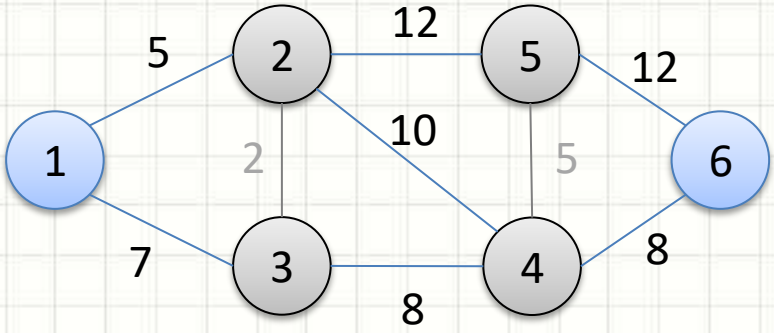
D



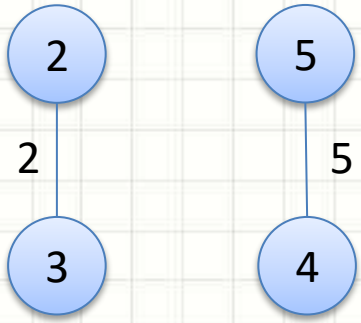
Aplicando o Algoritmo de Kuskal

3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

O



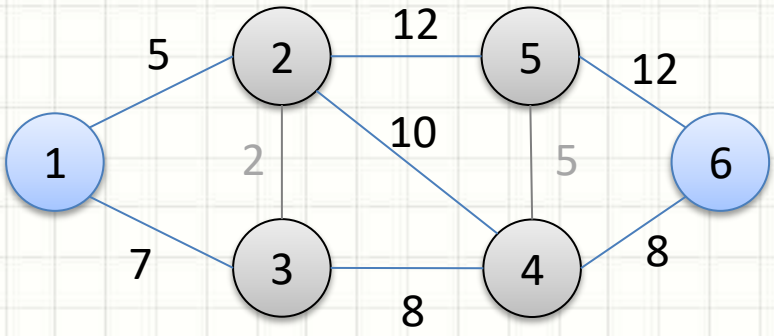
D



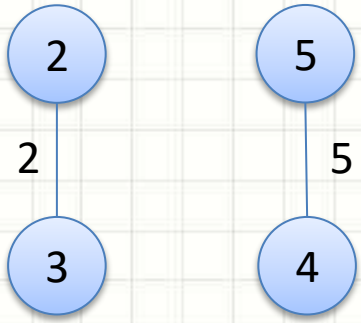
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



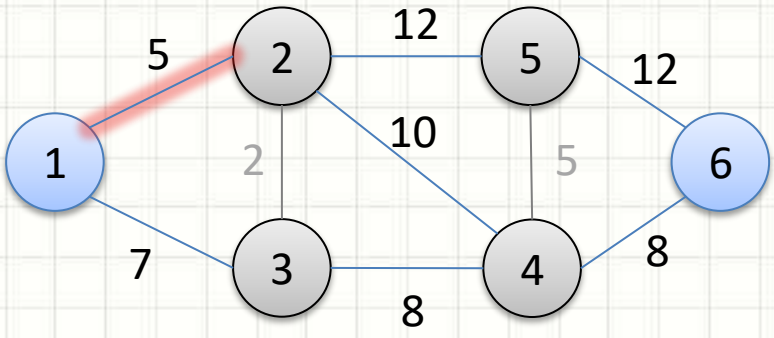
D



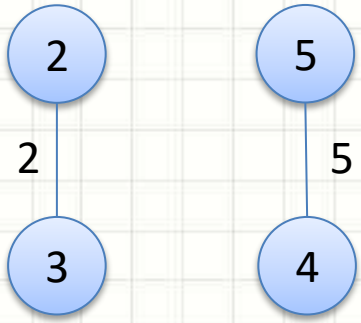
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



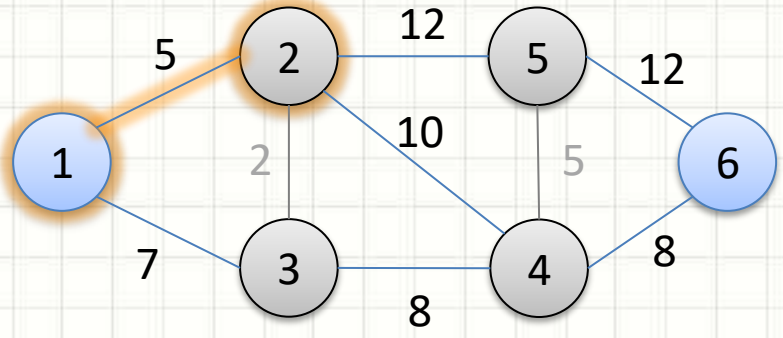
D



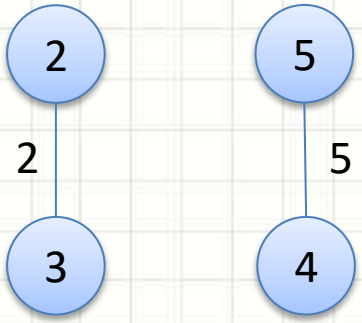
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



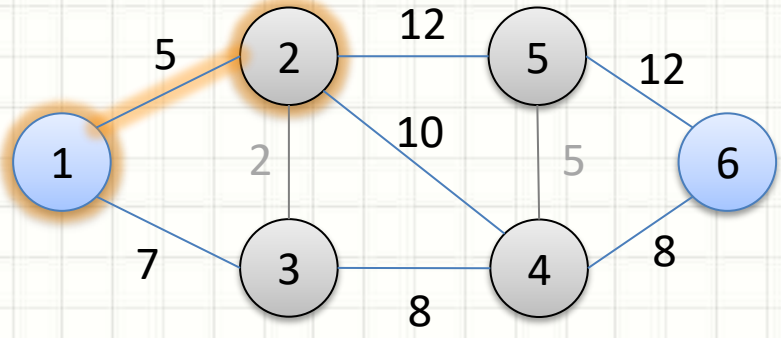
D



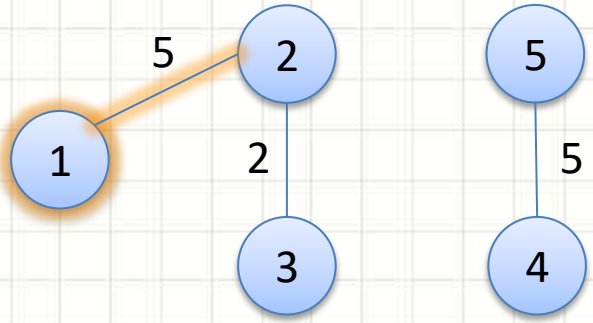
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



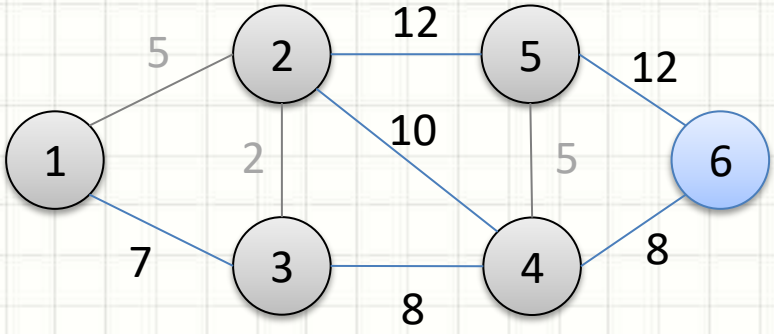
D



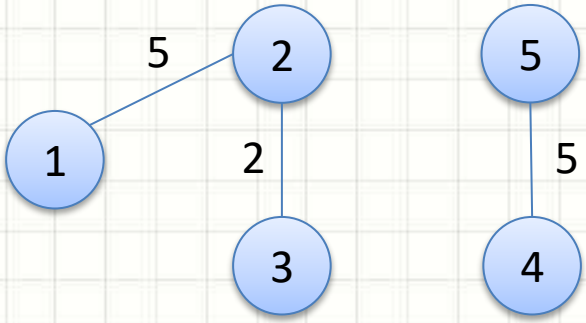
Aplicando o Algoritmo de Kuskal

3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

O



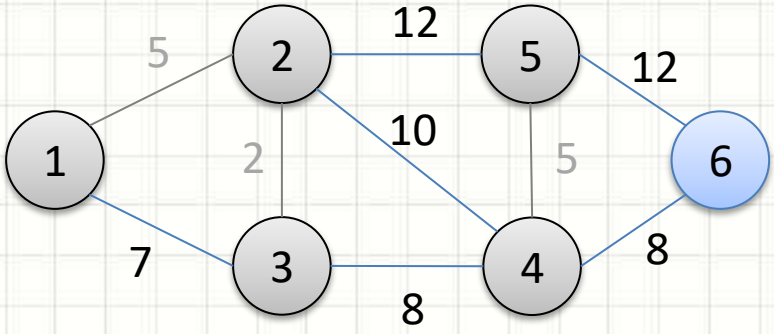
D



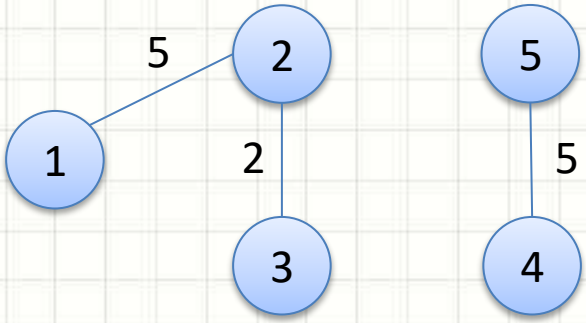
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



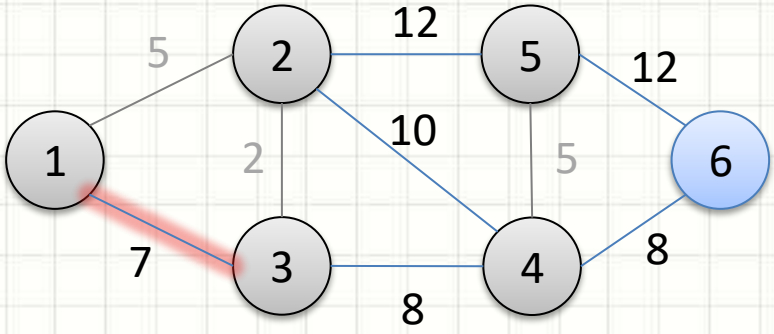
D



Aplicando o Algoritmo de Kuskal

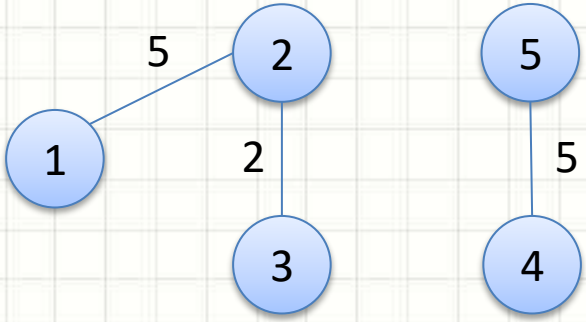
1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



Não é seguro!

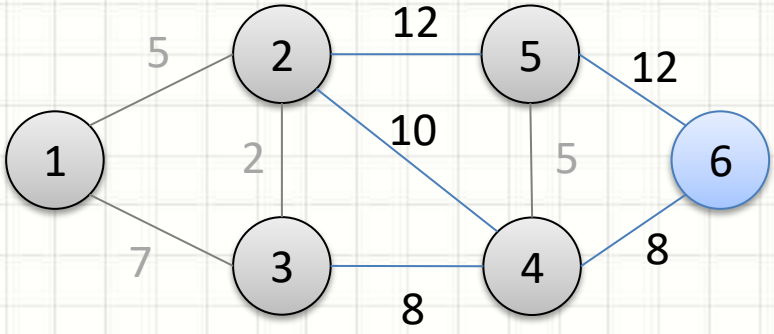
D



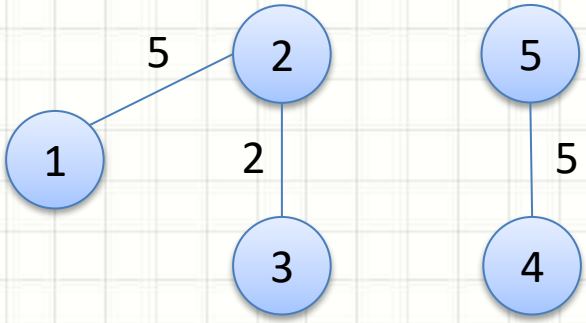
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



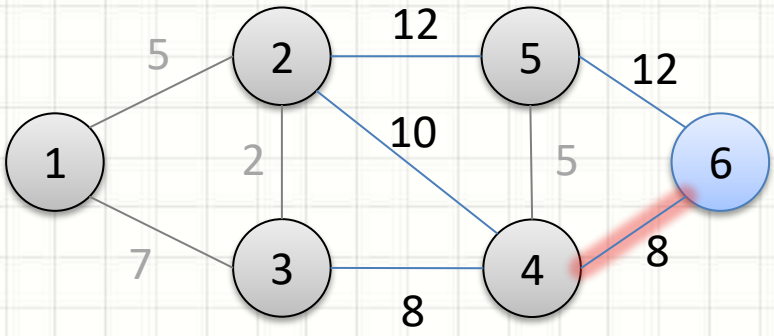
D



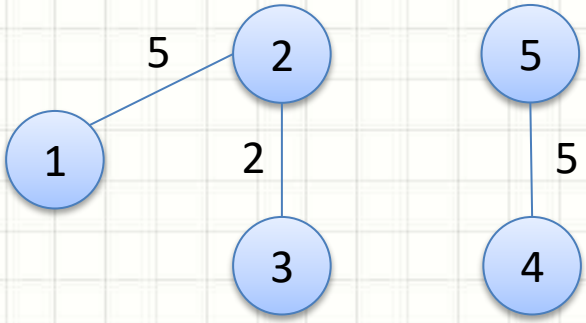
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



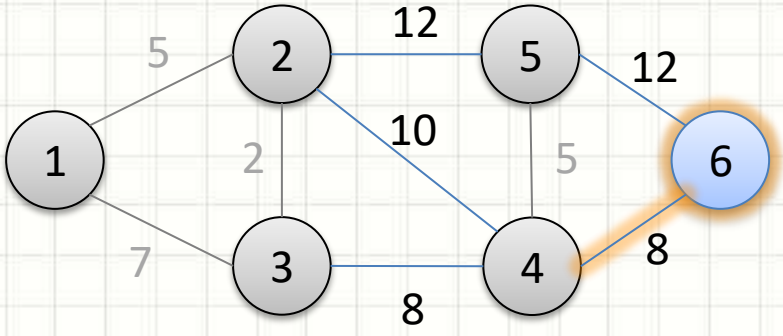
D



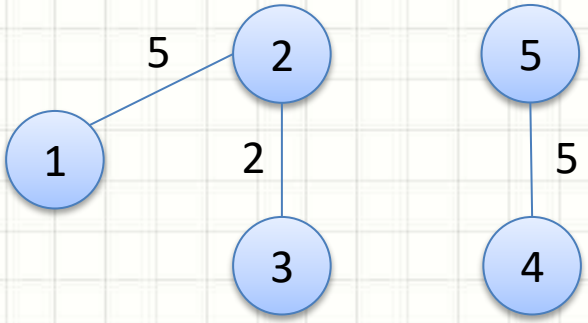
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



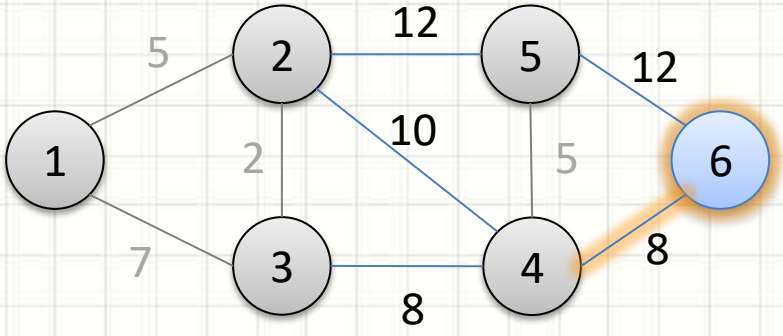
D



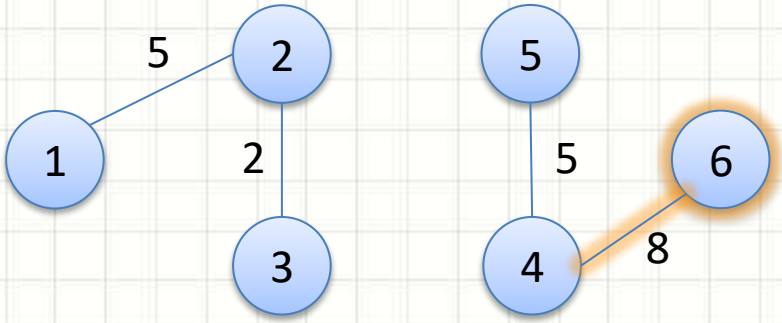
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



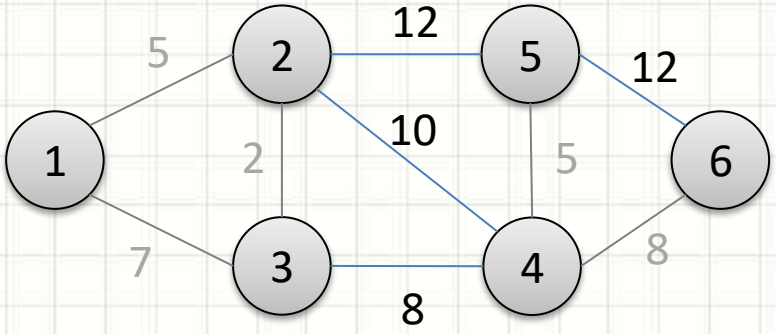
D



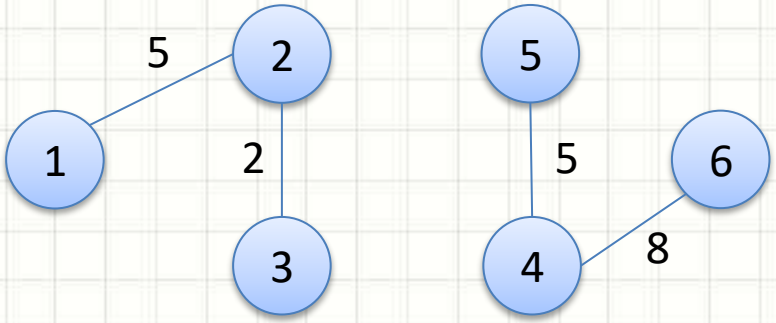
Aplicando o Algoritmo de Kuskal

3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

O



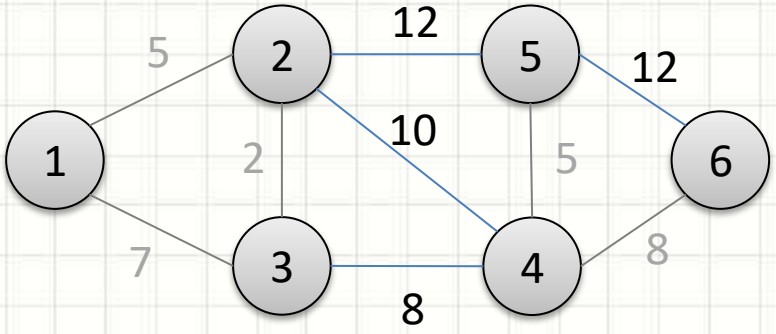
D



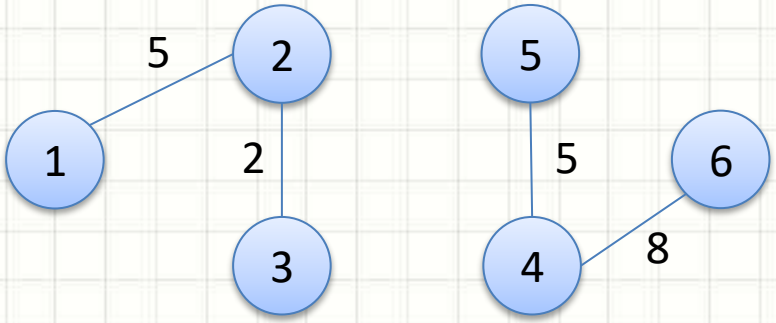
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



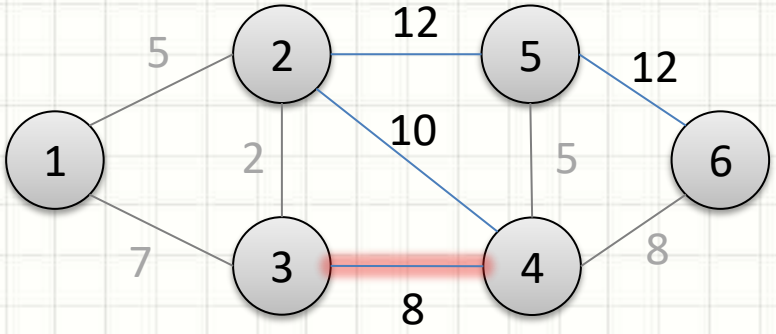
D



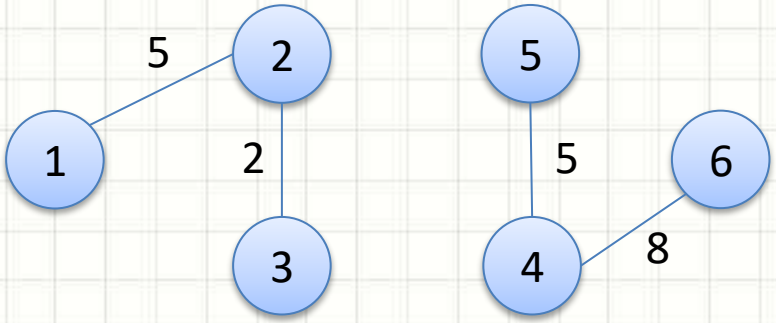
Aplicando o Algoritmo de Kuskal

1. No grafo original **O**, escolha o arco **L** de menor custo “seguro”

O



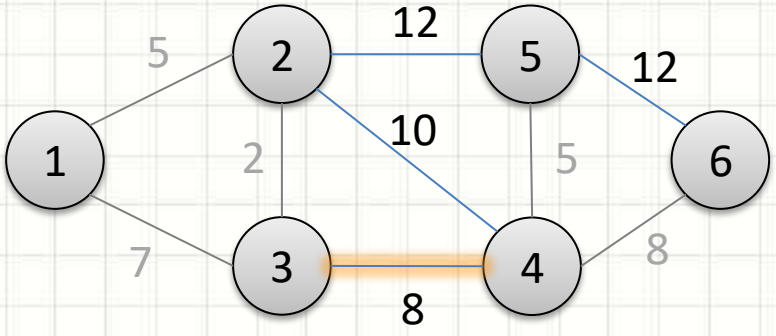
D



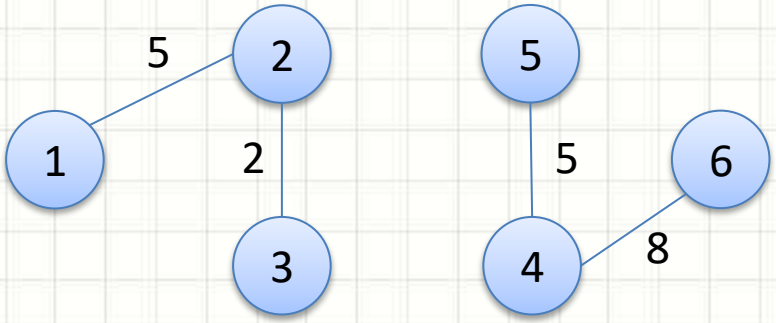
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



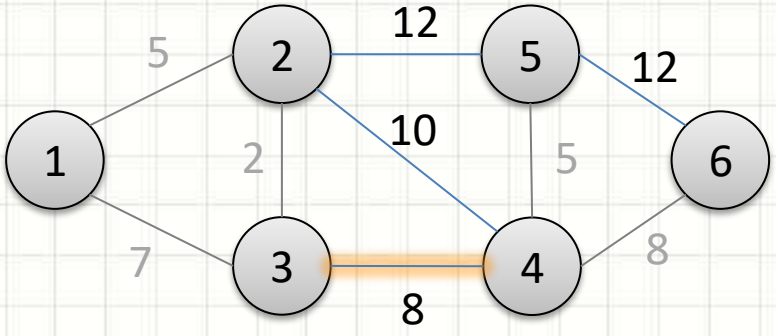
D



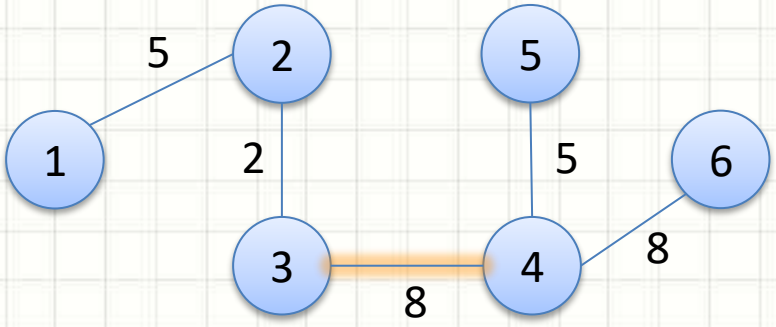
Aplicando o Algoritmo de Kuskal

2. Transfira o arco **L** e os vértices **A** e **B** de seus extremos para a árvore destino **D**

O



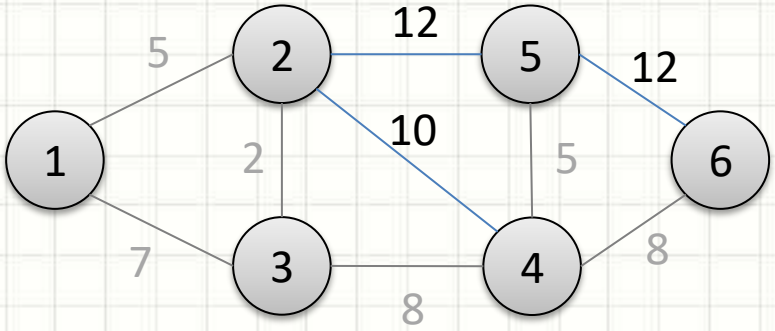
D



Aplicando o Algoritmo de Kuskal

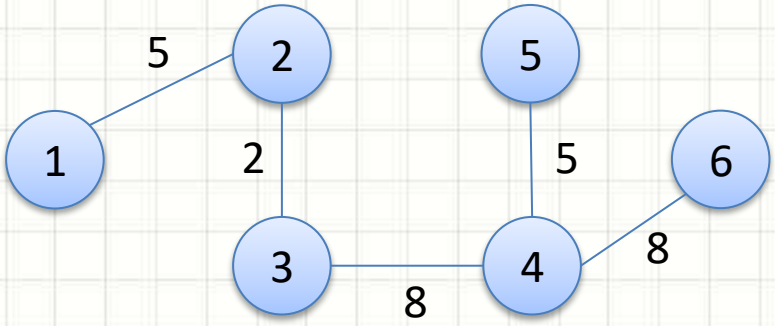
3. Se ainda há nós em **O** que não estão em **D**, ou há nós desconectados em **D**, volte ao passo 1.

O



Acabou!

D



Custo: 28

Solução idêntica à do Prim, mas nem sempre isso acontece!

Quando Usar Prim e Quando Usar Kruskal

Comparando Prim x Kruskal

- Complexidade Computacional de Prim

$$C = O(v^2)$$

Não fala nada dos arcos!

- Complexidade do algoritmo de Kruskal

$$C = O(a.\log v)$$

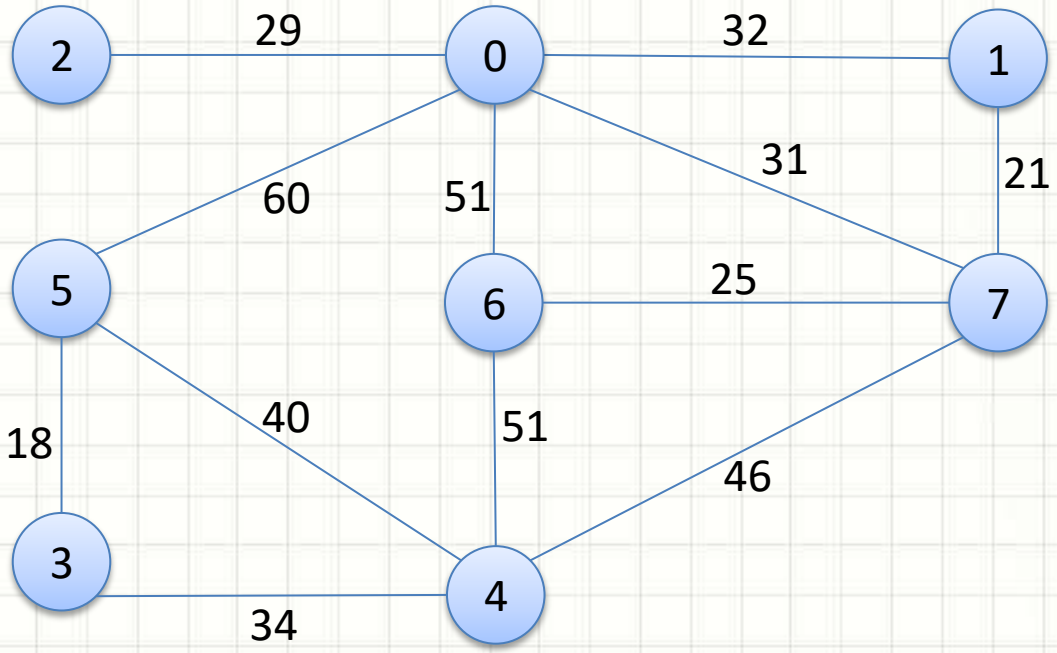
Arcos pesam mais que nós

- Usamos Prim
  - Quando o grafo é todo conexo
  - Quando a densidade é alta (muitos arcos por nó)
- Usamos Kruskal
  - Quando o grafo não é todo conexo
  - Quando a densidade é baixa (poucos arcos)

EXERCÍCIOS

Exercício

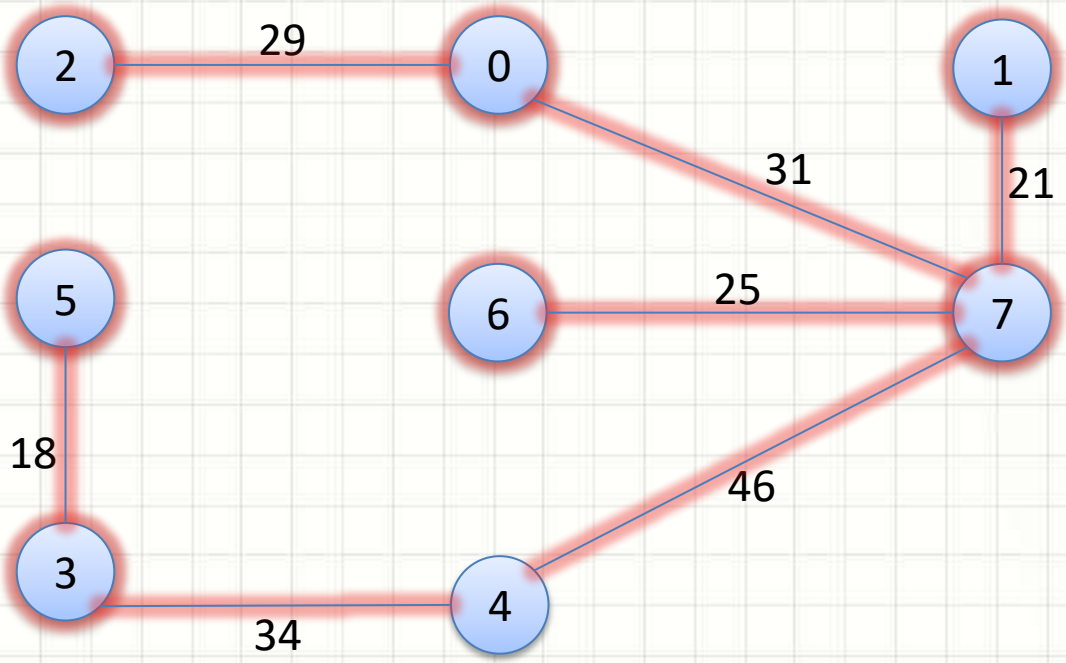
1. Aplique o Algoritmo de Kruskal



Fonte: https://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/prim.html

Exercício

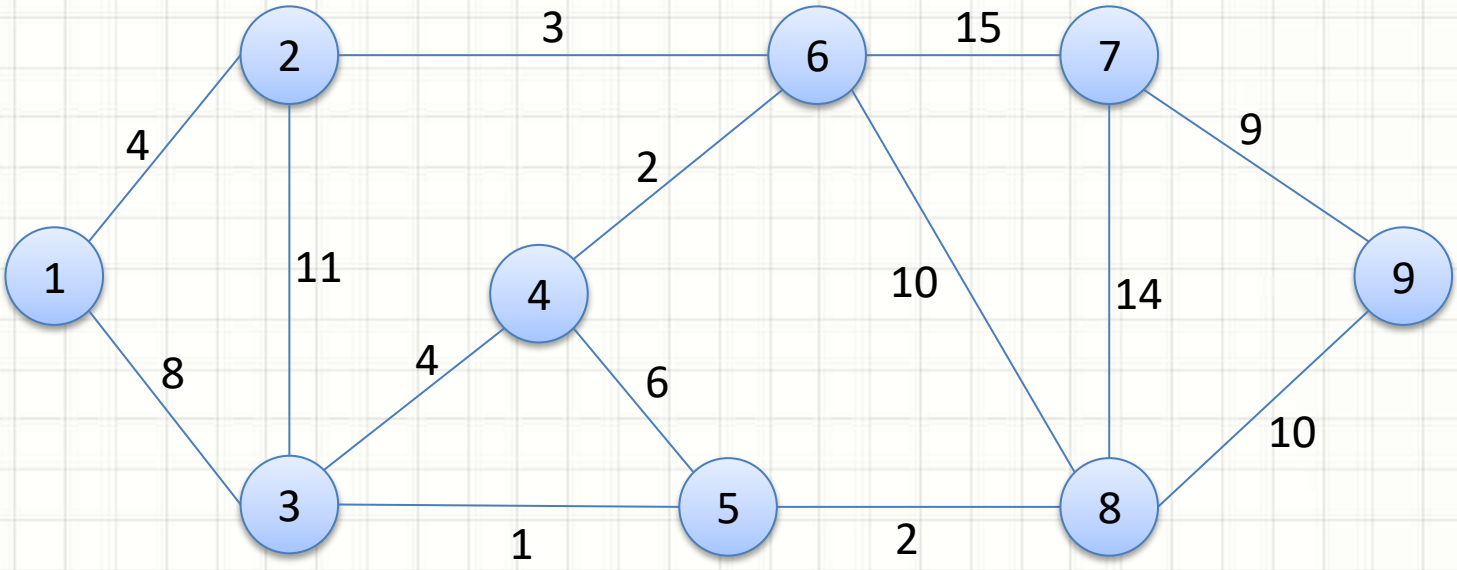
1. Aplique o Algoritmo de Kruskal



Fonte: https://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/prim.html

Exercício (para entrega!)

2. Aplique o Algoritmo de Kruskal



Fonte: http://www.ic.unicamp.br/~zanoni/mc448/2003-2s/listas/lista23.pdf

Conclusões

Resumo

- Algoritmo de Prim: tem limites
- Algoritmo de Kruscal: também tem!
- Cada um se dá melhor em casos diferentes
- **TAREFA**: Exercícios Aula 3

- Como encontrar o caminho mínimo?
  - Entre dois pontos?
  - Entre quaisquer pontos?

PERGUNTAS?