ÁRVORE GERADORA MÍNIMA

ALGORITMO PRIM

ALGORITMOS E ESTRUTURAS DE DADOS II - DCA0209

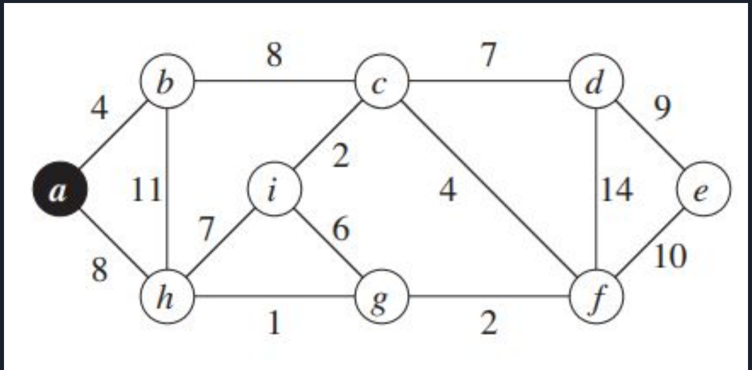
Jefferson Estevo Feitosa

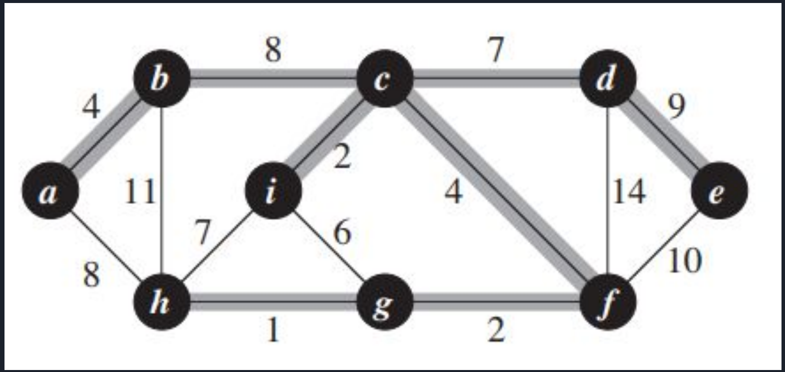
ALGORITMO PRIM

Problema:

Encontrar uma MST (árvore geradora de custo mínimo) de um grafo não-direcionado com custos nas arestas.

Os custos das arestas são números inteiros arbitrários (positivos ou negativos). O problema tem solução se e somente se o grafo é conexo.





CARACTERÍSTICAS ALGORITMO PRIM

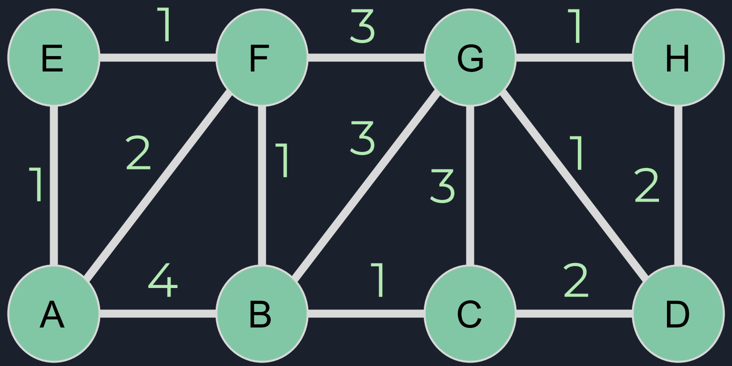
- ALGORITMO GULOSO
- GERA UM SUB-GRAFO (ÁRVORE) DE MENOR CUSTO
- PODE TER ATUALIZAÇÃO DE ARESTAS EXISTENTES
- A ÁRVORE É GERADA DE FORMA INCREMENTAL
- PARA CADA ITERAÇÃO TEMOS UMA SOLUÇÃO PARCIAL
- NÃO FORMA CICLOS FECHADOS
- CONECTA TODOS OS VÉRTICES DE FORMA EFICIENTE

ALGORITMO DE PRIM

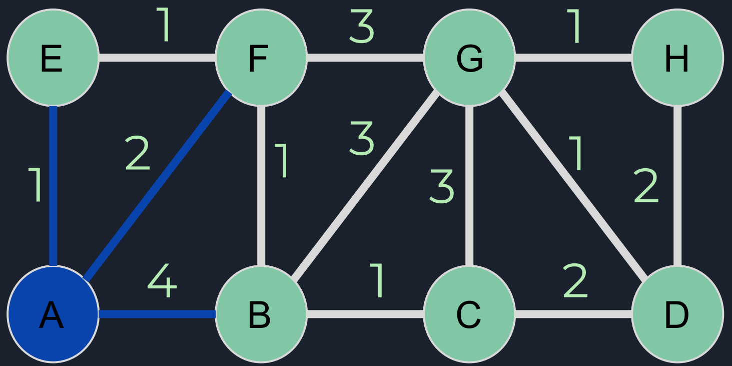
MST-Prim(G, w, r)

```
for cada vértice u ∈ V(G)
        chave[u] = ∞
        π[u] = 0
chave[r] = 0
Q = V(G)
while Q ≠ Ø
        u = Extract-Min(Q)  // Menor chave[u]
        for cada  v ∈ Adj[u]
              if (v ∈ Q ) && (w(u, v) < chave[v])
                    π[v] = u
                    chave[v] = w(u, v)
```

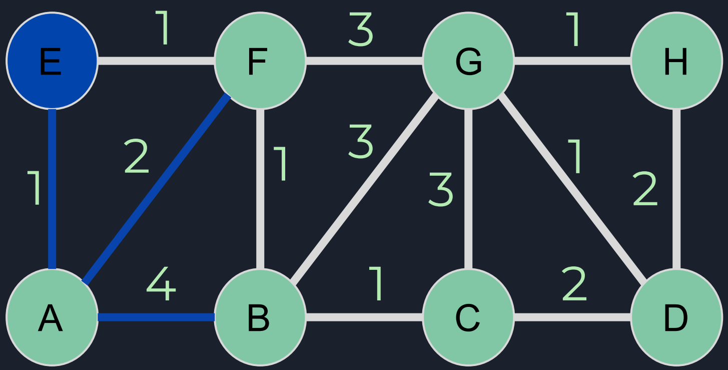
EXEMPLO 1



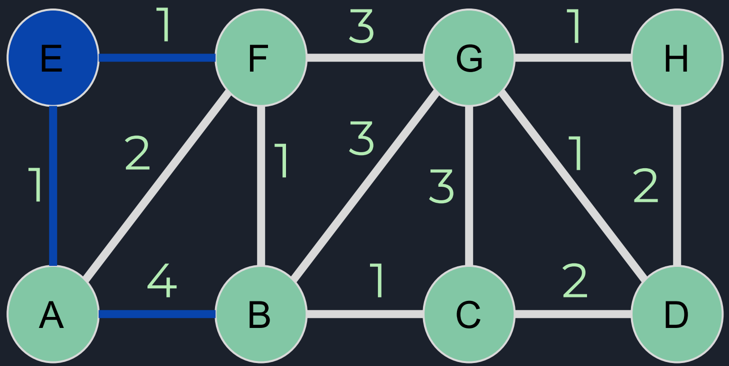
Q = {(A,0,*), (B,∞,*), (C,∞,*), (D,∞,*), (E,∞,*), (F,∞,*), (G,∞,*), (H,∞,*)}



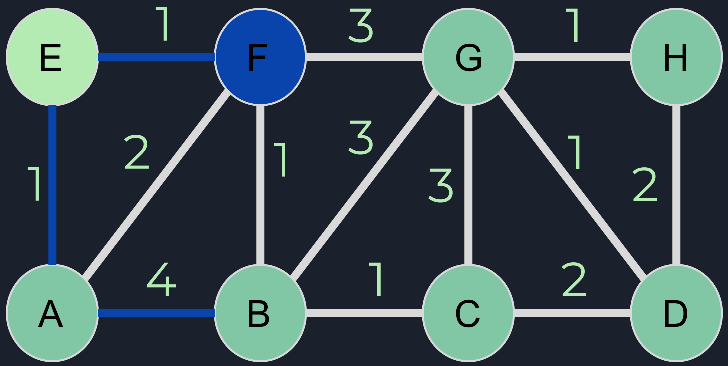
Q = {(E,1,A), (F,2,A), (B,4,A), (C,∞,*), (D,∞,*), (G,∞,*), (H,∞,*)}



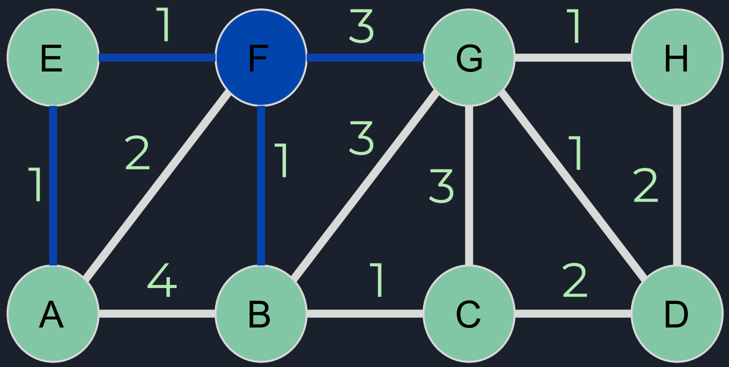
Q = {(F,2,A), (B,4,A), (C,∞,*), (D,∞,*), (G,∞,*), (H,∞,*)}



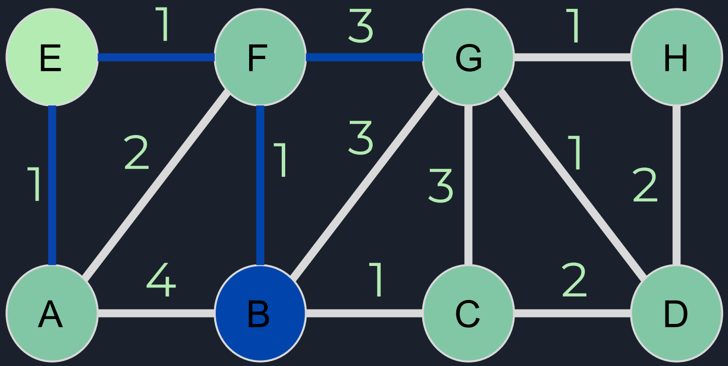
Q = {(F,1,E), (B,4,A), (C,∞,*), (D,∞,*), (G,∞,*), (H,∞,*)}



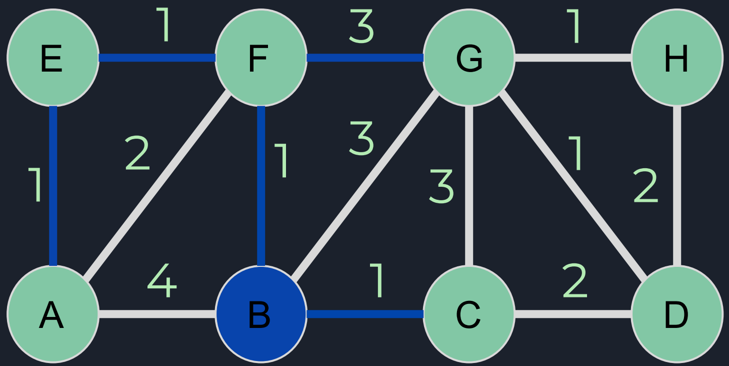
Q = {(B,4,A), (C,∞,*), (D,∞,*), (G,∞,*), (H,∞,*)}



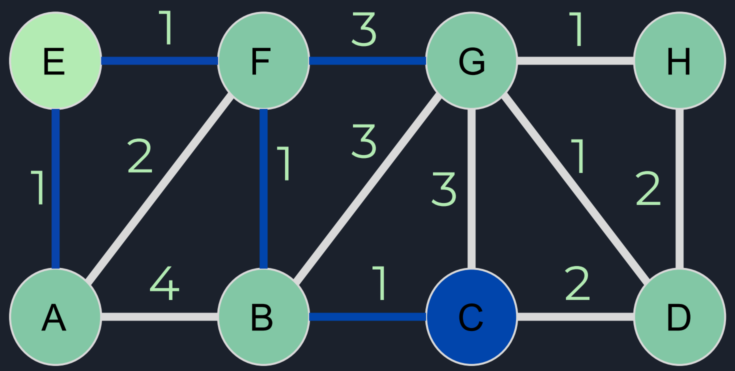
Q = {(B,1,F), (G,3,F), (C,∞,*), (D,∞,*), (H,∞,*)}



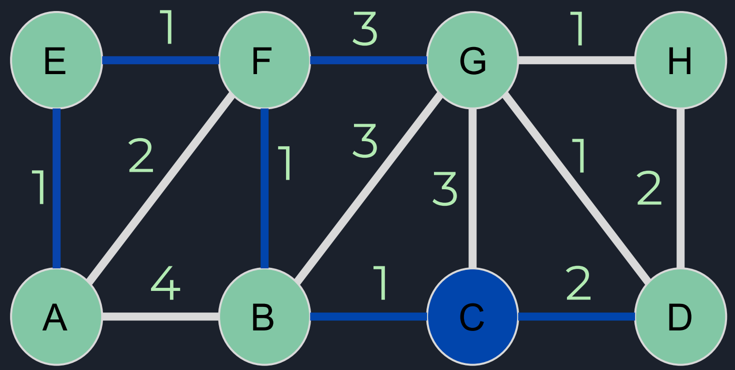
$Q = \{(G,3,F), (C,\infty,*), (G,\infty,*), (H,\infty,*)\}$



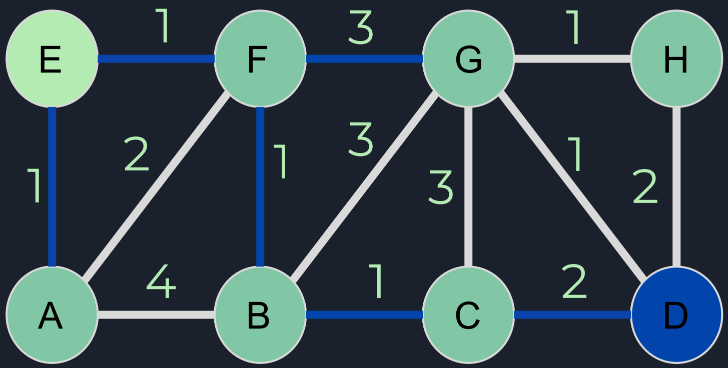
$Q = \{(C,1,B), (G,3,F), (D,\infty,*), (H,\infty,*)\}$



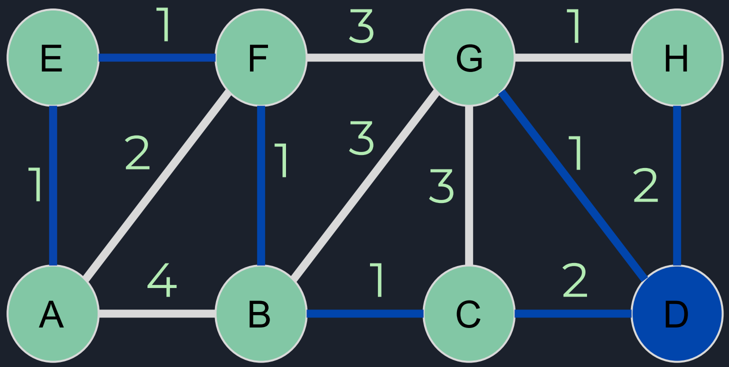
Q = {(G,3,F), (D,∞,*), (H,∞,*)}



Q = {(D,2,C), (G,3,F), (H,∞,*)}

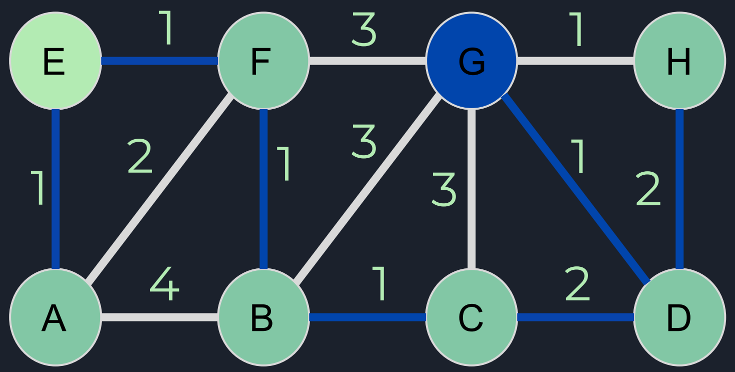


Q = {(G,3,F), (H,∞,*)}

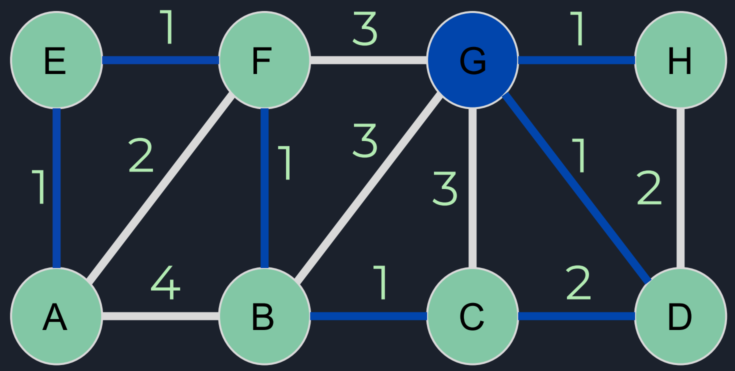


Q = {(G,1,D), (H,2,D)}

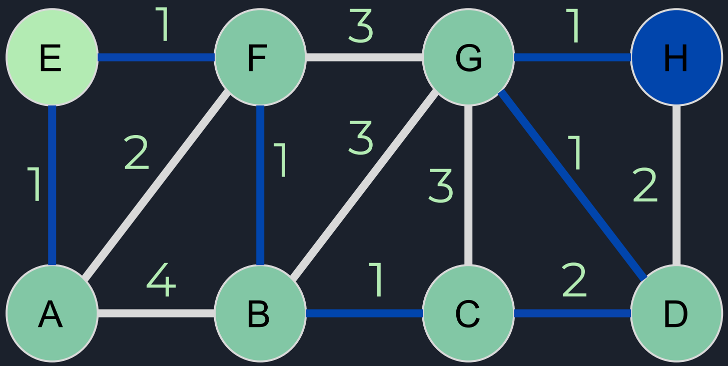
Nesta iteração temos uma Árvore Geradora. Porém, ela ainda não é mínima, pois a fila Q ainda não está vazia



Q = {(H,2,D)}

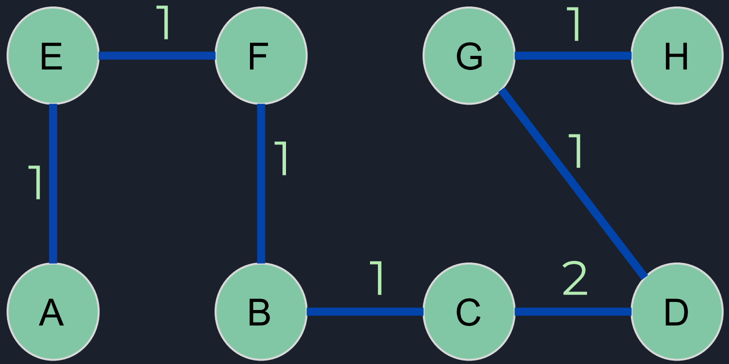


Q = {(H,1,G)}

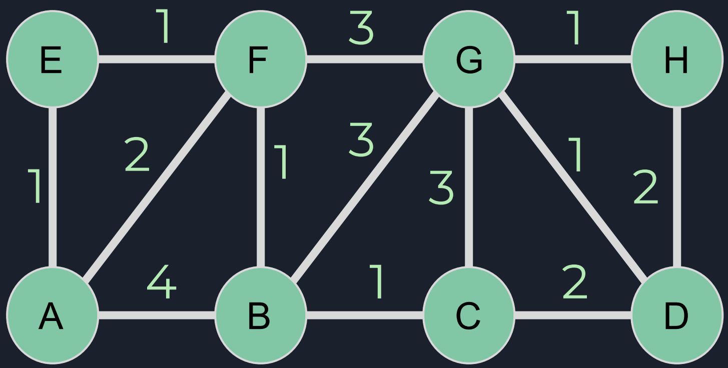


Q = {}

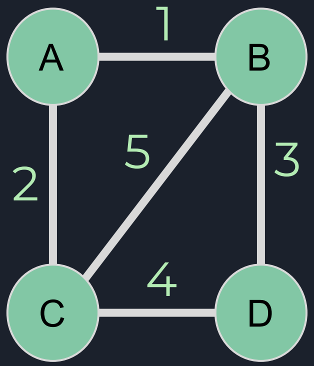
ÁRVORE GERADORA MÍNIMA



GRAFO INICIAL



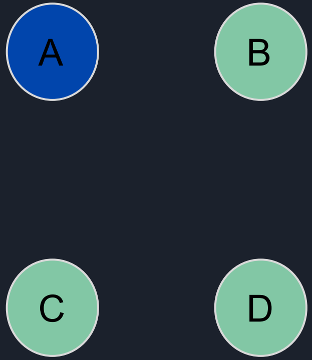
PARALELO ENTRE PRIM E KRUSKAL



PRIM

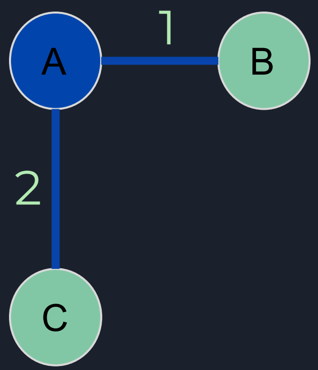
- (A,0,*)
- (B,∞,*)
- (C,∞,*)
- (D,∞,*)

KRUSKAL

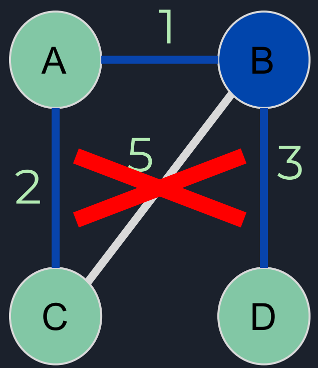


- (A-B,1)
- (A-C,2)
- (B-D,3)
- (C-D,4)
- (B-C,5)

PRIM

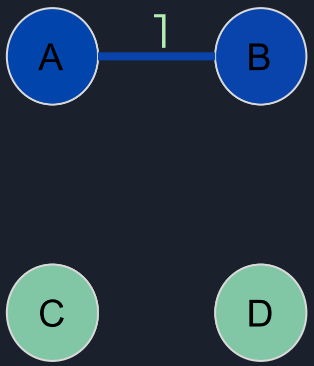


(A,0,*)
(B,1,A)
(C,2,A)
(D,∞,*)

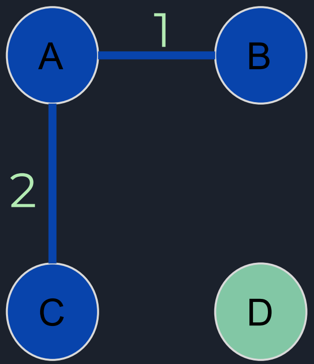


(A,0,*)
(B,1,A)
(C,2,A)
(D,3,B)

KRUSKAL

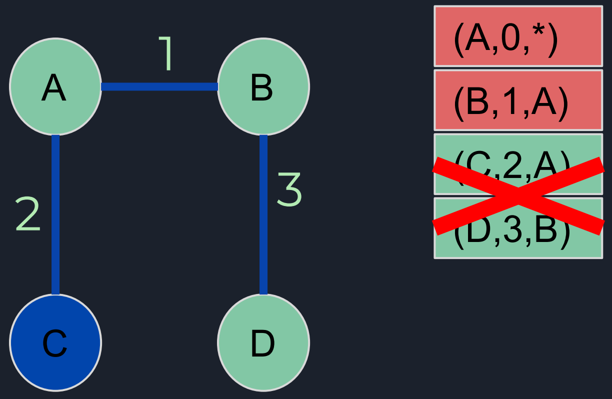


(A-B,1)
(A-C,2)
(B-D,3)
(C-D,4)
(B-C,5)

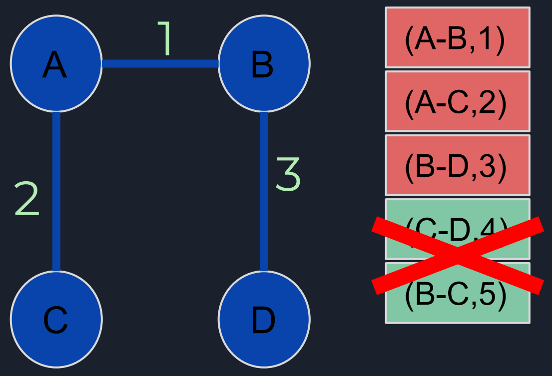


(A-B,1)
(A-C,2)
(B-D,3)
(C-D,4)
(B-C,5)

PRIM



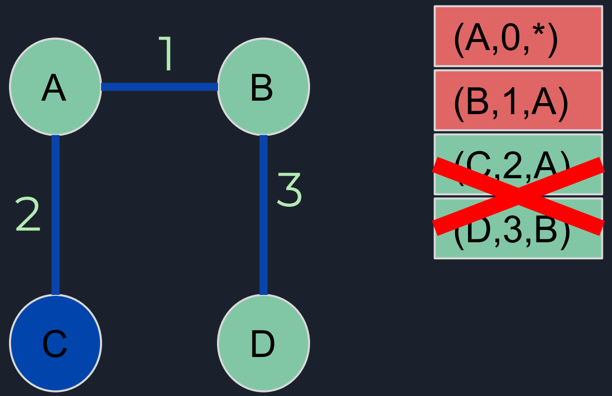
KRUSKAL



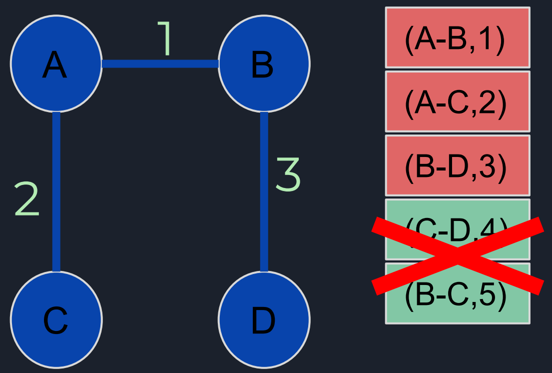
Resultado: Obtivemos a mesma
Árvore Geradora Mínima

Será sempre verdadeiro isso?

PRIM



KRUSKAL

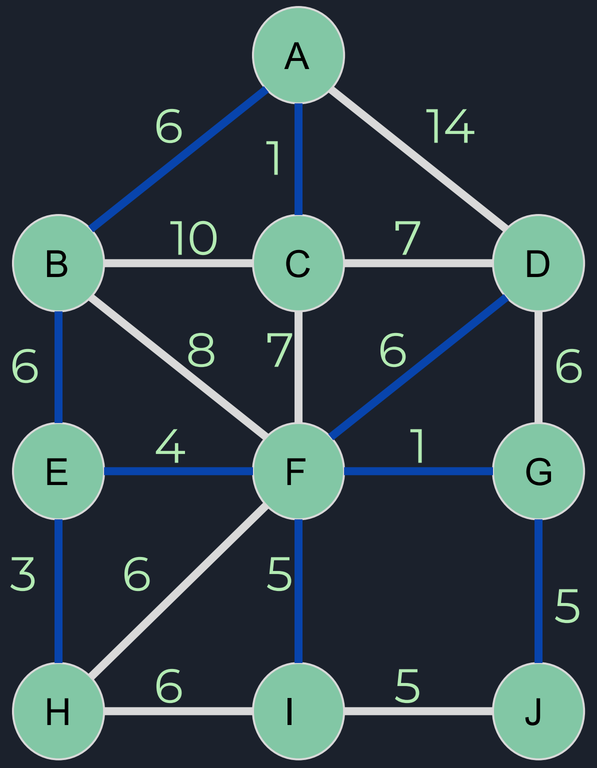


Resultado: Obtivemos a mesma
Árvore Geradora Mínima

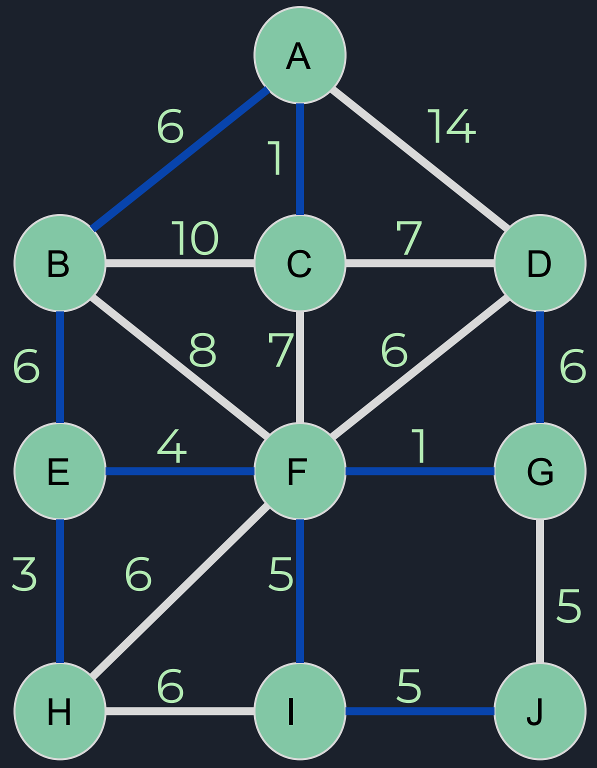
Será sempre verdadeiro isso?

NÃO! Os custos dos pesos serão sempre
os mesmos, porém, os caminhos podem
sofrer leves mudanças

PRIM



KRUSKAL



COMPLEXIDADE

MST-Prim(G, w, r)

O(V*log(V)) + O(E*log(V))

O(E*log(V))

```
for cada vértice u ∈ V(G)
        chave[u] = ∞
        π[u] = 0
chave[r] = 0
Q = V(G)
while Q ≠ Ø
        u = Extract-Min(Q)
        for cada v ∈ Adj[u]
            if (v ∈ Q ) && (w(u, v) < chave[v])
                π[v] = u
                chave[v] = w(u, v)
```

O(V*log(V)) ⇒ Iterar vértices e Fila de prioridade

O(E*log(V)) ⇒ Visitar todas as arestas adjacentes

O(V) ⇒ Iterar sobre cada vértice

O(log(V)) ⇒ Criar e inicializar a fila de prioridades

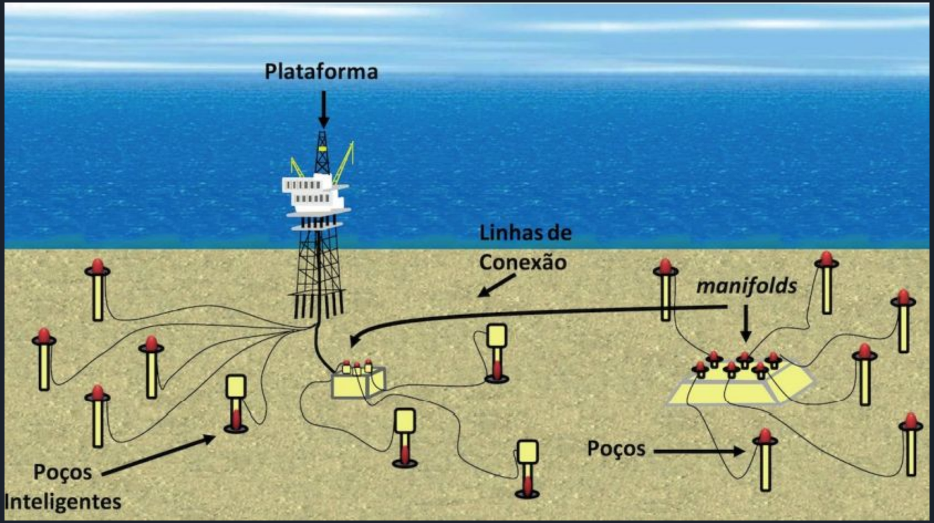
O(log(V)) ⇒ Extrair o vértice de menor valor

O(E*log(V)) ⇒ Visitar todas as arestas adjacentes

O(log(V)) ⇒ Atualizar valores na fila de prioridades

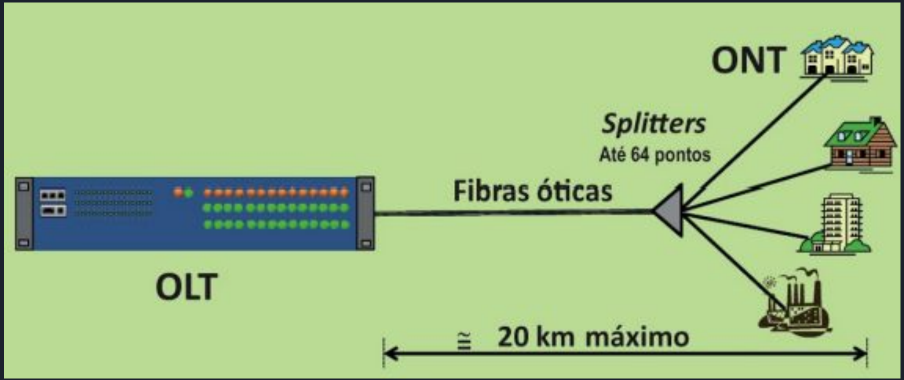
EXEMPLOS DE APLICAÇÃO

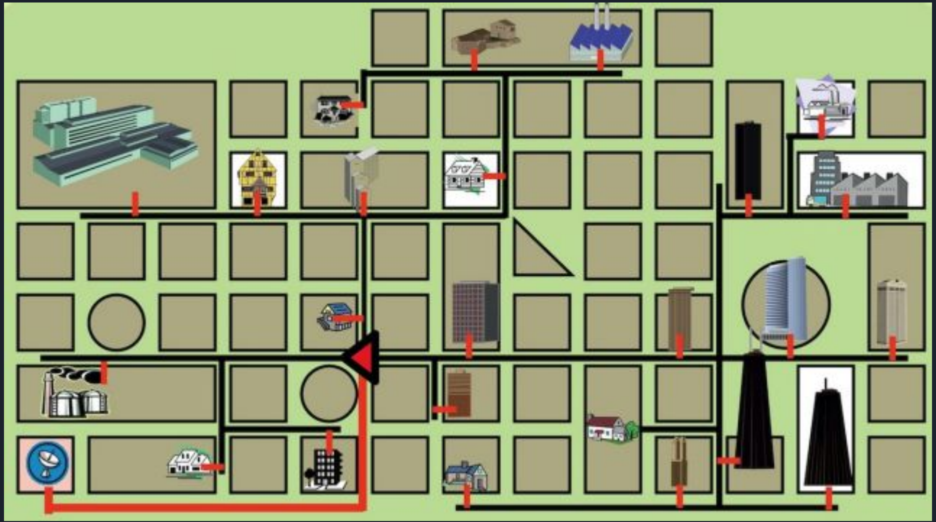
Campos de petróleo Off-Shore são sistemas extremamente complexos que visam coletar e direcionar o óleo dos poços em solo submarino até um ponto de transporte.



EXEMPLOS DE APLICAÇÃO

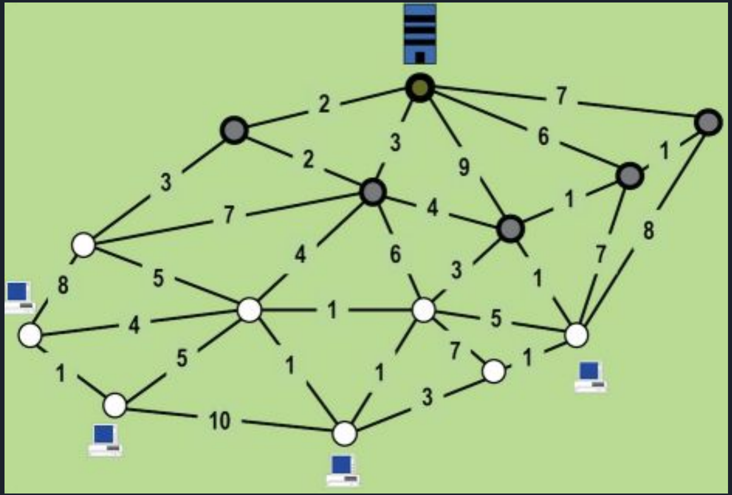
Em um sistema de transmissão de uma rede óptica passiva, o sinal óptico é transmitido por uma rede de distribuição. Na fibra óptica são feitas derivações através do uso de splitters (divisores ópticos passivos).

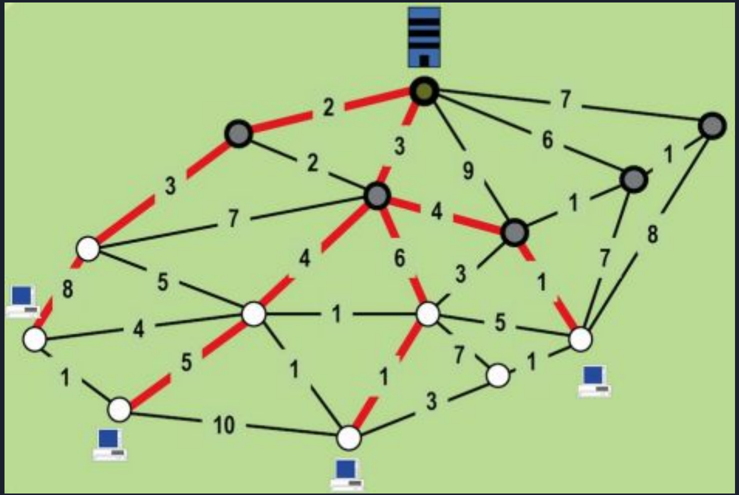




EXEMPLOS DE APLICAÇÃO

Otimização de distribuição de sinal em redes. Um sinal é gerado em um ponto da rede, transita codificado até pontos de decodificação e é distribuído aos usuários.





OUTRAS APLICAÇÕES

- Projeto de redes de computadores e de comunicação;
- Instalações telefônicas, hidráulicas, elétricas, de petróleo e gás;
- Análise de agrupamentos;
- Análise genética;
- Análise de padrões de distribuição espacial de esporos;
- Astronomia (determinação de agrupamento de quasars);
- Geração de limites de problemas NP-Difíceis;
- Computação móvel;
- Modelos de localização de interação de partículas em fluxo turbulento de fluidos;

OBRIGADO PELA ATENÇÃO!

CÓDIGO [Algoritmo-Prim/prim.cpp at main · JeffersonEstevo/Algoritmo-Prim (github.com)]