Àrvores Geradoras Mínimas

Disciplina: Técnicas de Programação Avançada

Mateus Conrad B. da Costa

Ifes - Campus Serra

17 de dezembro de 2024

Definição de Àrvore Geradora e Àrvore Geradora Mínima

- Uma **árvore geradora** de um grafo conexo $G = (V, E)$ **contém todos os vértices** de G com o mínimo de arestas.
- Uma **árvore geradora mínima (MST)** é a **árvore geradora com o menor peso total**, onde o peso é definido pela soma dos pesos das arestas.

Conceito de Corte

- Um **corte** é uma partição $(S, V - S)$ do conjunto de vértices V de um grafo G.
- Uma aresta e **cruza o corte** se seus vértices pertencem a lados opostos da partição.
- Um corte **respeita um conjunto** A de arestas se nenhuma das arestas de A cruza o corte.
- Uma **aresta leve** é a aresta de peso mínimo entre as arestas que cruzam o corte.

Determinando Arestas Seguras

- **Teorema:** Se $(S, V - S)$ é um corte que respeita o conjunto de arestas A, e (u, v) é uma **aresta leve** que cruza o corte, então (u, v) é uma **aresta segura** para A.

Propriedade dos Ciclos

- Seja T uma árvore geradora de um grafo G.
- Sejam e uma aresta de G fora de T e C um ciclo formado por $e + T$.
- Se T é uma **árvore geradora mínima**, para cada aresta f em C:
$$w(f) \leq w(e)$$
- Se $w(f) > w(e)$, é possível obter uma árvore geradora de menor peso trocando-se f por e.

Algoritmo de Prim-Jarnik

- O algoritmo de **Prim-Jarnik** é semelhante ao algoritmo de Dijkstra para caminhos mínimos.
- Ele parte de um vértice arbitrário e constrói a MST:
  1. Escolhe um vértice inicial.
  2. Encontra a **aresta mínima** conectando a MST ao grafo restante.
  3. Repete o processo até incluir todos os vértices.

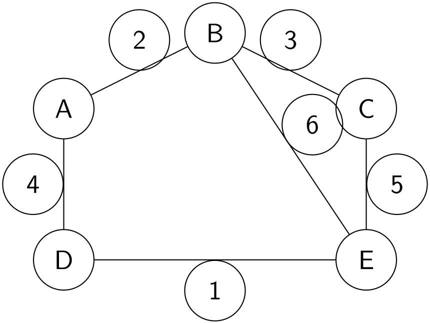
Algoritmo de Prim-Jarnik

Algorithm 1 Prim-Jarnik

Require: Grafo $G = (V, E)$, pesos w, vértice inicial r

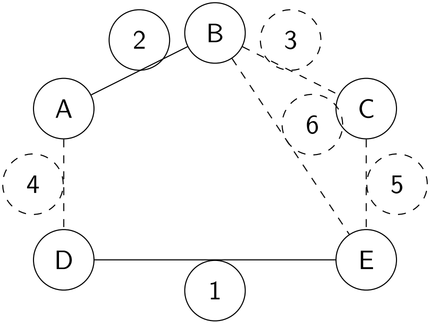
```
for cada vértice u ∈ V do
    custo[u] ← ∞
    p[u] ← nil
end for
custo[r] ← 0
Q ← V
while Q não vazio do
    u ← removeMenor(Q)
    for cada v adjacente a u do
        if v ∈ Q e w(u, v) < custo[v] then
            p[v] ← u
            custo[v] ← w(u, v)
        end if
    end for
```

Exemplo de Grafo - Etapa 1



- **Etapa 1:** Selecionamos o vértice A como ponto de partida.
- **Aresta escolhida:** (A, B) com peso 2.

Exemplo de Grafo - Etapa 2



- **Etapa 2:** Selecionamos a aresta (D, E) com peso 1.
- **Arestas selecionadas:** (A, B) e (D, E).

Algoritmo de Kruskal

- O algoritmo de **Kruskal** inicia com uma floresta de $|V|$ árvores disjuntas (cada vértice é uma **árvore**).
- **Passos:**
  1. Ordena todas as arestas por peso.
  2. Percorre a lista de arestas:
     - Adiciona a aresta se ela não formar ciclo na floresta.
  3. Repete até que todas as **árvores** estejam conectadas.

Algoritmo de Kruskal

Algorithm 2 Kruskal

Require: Grafo $G = (V, E)$, pesos w

$T \leftarrow \emptyset$ {Inicializa a floresta}
Ordena as arestas E em ordem não decrescente de $w(e)$
for cada aresta $(u, v) \in E$ em ordem **do**
    if u e v estão em componentes disjuntos **then**
        Adiciona (u, v) a T
        Une os componentes de u e v
    end if
end for
Saída: Árvore Geradora Mínima T