Teoria dos Grafos
Período 2003.1

Árvore de Cobertura ou Árvore Geradora

Árvore de Cobertura ou Árvore Geradora

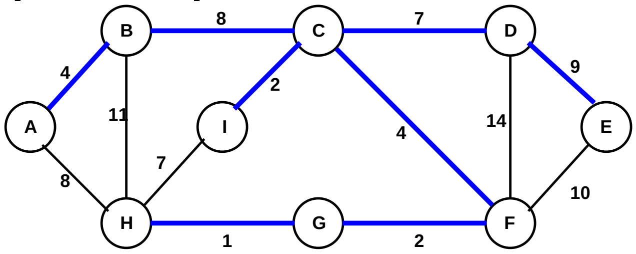
Considere os seguintes problemas:

- **Problema 1:** Temos um mapa modelado por um grafo: os vértices correspondem a cidades e os arcos representam estradas de terra batida entre cidades adjacentes, com os rótulos indicando a respectiva distância. O Governo do Estado planeja asfaltar algumas estradas, tornando possível sair de qualquer cidade para outra em estrada asfaltada. Que estradas deveriam ser asfaltadas? Seria possível decidir como minimizar o total de asfalto a ser gasto?

- **Problema 2:** No sistema de abastecimento de água de Campina Grande, existem vários tanques para armazenamento e tratamento da água que vem do açude de Boqueirão. Como definir a forma de interligar esses tanques sabendo que, em princípio, qualquer par de tanques pode ser interligado. É desnecessário dizer que a solução ideal é aquela em que todos os tanques serão abastecidos e que o custo das obras seja mínimo.

Os dois problemas acima são conhecidos como o problema de conexão mínima. Este problema, na teoria dos grafos é conhecido como o problema de encontrar a árvore de cobertura (geradora) mínima para o grafo.

Uma *árvore de cobertura* ou *árvore geradora* de um grafo conectado G é um subgrafo que forma uma árvore e que inclui cada um dos vértices de G. Uma *árvore de cobertura mínima* para um grafo valorado é uma árvore de cobertura em que a soma dos pesos das arestas é mínima.



A figura acima apresenta um grafo e sua árvore de cobertura mínima (em azul).

Sobre árvore de cobertura podemos dizer:

- O conceito de árvore de cobertura só existe para grafos conectados.
- Se o grafo tem n vértices, a árvore de cobertura em $n-1$ vértices.
- Para determinar uma árvore de cobertura:
  - Se o grafo G não tem ciclos, G é uma árvore de cobertura.
  - Se G tem ciclo, é necessário remover recursivamente arcos (até achar uma árvore), mantendo o grafo conectado.

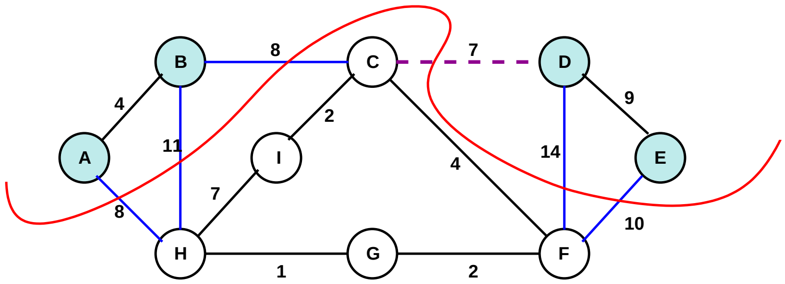
No caso de árvore de cobertura mínima, uma solução genérica utiliza uma estratégia gulosa. Em cada passo da esratégia gulosa, uma nova aresta (u, v) é escolhida para compor a solução A. Em cada passo, A contém um subconjunto de alguma árvore de cobertura mínima. A nova aresta escolhida deve respeitar essa propriedade (essa seria a nossa função de viabilidade). Dizemos, portanto, que uma aresta que pode ser incluído no conjunto A é uma aresta segura.

Uma solução genéria seria:

SolucaoGenerica(G, w**)**
$A \leftarrow \emptyset$
while A não é árvore de cobertura **do**
  Encontrar uma aresta segura (u, v) para A
  $A \leftarrow A \cup \{(u, v)\}$
return A

Uma forma de reconhecer uma aresta segura é:

- Definir um corte no grafo, particionando o conjunto de vértices.
- Uma aresta cruza o corte se cada um dos vértices que formam a aresta está em partições diferentes.
- O corte deve respeitar A. Isso acontece se nenhuma aresta de A cruza o corte.
- A aresta segura é a aquela de menor peso que cruza o corte.



Vamos mostrar duas formas diferentes de implementar a escolha da aresta segura.

Algoritmo de Kruskal

- Kruskal utiliza conjuntos disjuntos.
- O tempo de execução depende de como conjuntos disjuntos foram implementados.
- Kruskal é $O(E \log E)$

Kruskal(G, w**)**
$\triangleright$ Considerar os arcos ordenados (decrescente) em relação ao peso w

$A \leftarrow \emptyset$
for cada vértice $v \in V$ **do**
  CriaConjunto(v)
for cada arco $(u, v) \in E$ **do**
  if EncontraConjunto(u) $\neq$**EncontraConjunto**(v) **then**
    $A \leftarrow A \cup \{(u, v)\}$
    Uniao(u, v)
return A

Algoritmo de Prim

- A solução se parece com a solução de Dijkstra para encontrar o menor caminho.
- Utiliza uma fila de prioridade como estrutura auxiliar.
- Prim é $O(E \log V)$.

Prim(G, w, r)

$Q \leftarrow V[G]$
for cada $u \in Q$ **do**
  $chave[u] \leftarrow \infty$
$chave[r] \leftarrow 0$
$\pi[r] \leftarrow NIL$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ **ExtraiMenor**(Q)
  for cada $v \in Adj[u]$ **do**
    if $v \in Q$ and $w(u, v) < chave[v]$ **then**
      $\pi[v] \leftarrow u$
      $chave[v] \leftarrow w(u, v)$