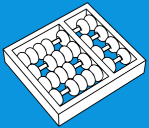
Conexidade. Árvore geradora mínima

MO417 - Complexidade de Algoritmos I

05/24

20



Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP

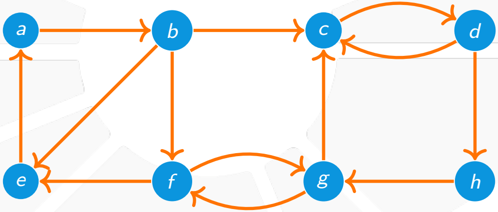
“Minimum Spanning Trees ... or how to bring the world together on a budget.”

Seth James Nielson

Componentes fortemente conexas

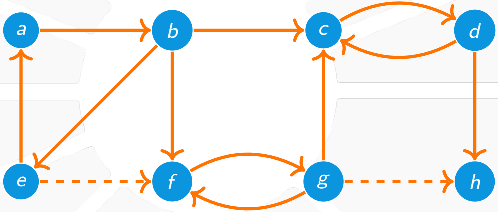
Grafo fortemente conexo

Um grafo direcionado $G = (\mathbf{V}, \mathbf{E})$ é **FORTEMENTE CONEXO** se para todo par de vértices **u,v** de G, existe um caminho direcionado de **u** a **v**.



Grafo fortemente conexo

Nem todo grafo direcionado é fortemente conexo:



Componente fortemente conexa

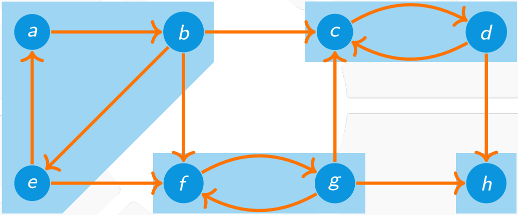
Uma **COMPONENTE FORTEMENTE CONEXA** de um grafo direcionado $G = (\mathbf{V}, \mathbf{E})$ é um subconjunto de vértices $\mathbf{C} \subseteq \mathbf{V}$ tal que:

(1) O subgrafo induzido por **C** é fortemente conexo e

(2) **C** é maximal com respeito à propriedade (1).

Componente fortemente conexa

Podemos **PARTICIONAR** um grafo direcionado em componentes fortemente conexas.

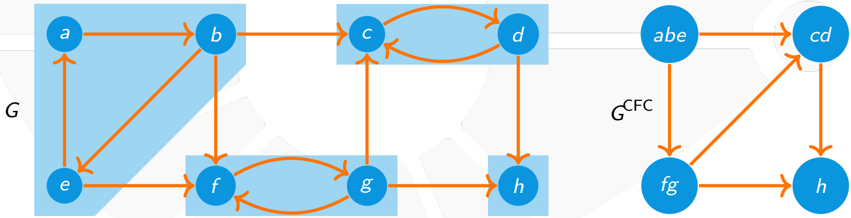


Como encontrar as componentes fortemente conexas?

Grafo componente

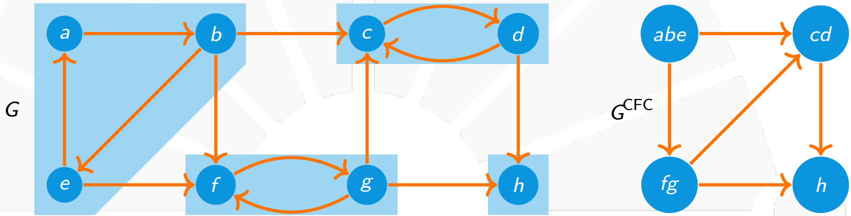
O **GRAFO COMPONENTE** de um grafo direcionado $G = (\mathbf{V}, \mathbf{E})$ é um grafo direcionado, denotado por G^{CFC} em que:

- Cada vértice é uma componente fortemente conexa.
- Existe aresta **(C,D)** se houver **(u,v)** $\in$ **E** com **u** $\in$ **C** e **v** $\in$ **D**.



Note que G^{CFC} é acíclico. Por quê?

Grafo componente



Considere uma busca em profundidade sobre G:

- Se **u** for o último vértice finalizado,
- então **u** deve pertencer a uma fonte de G^{CFC}. Por quê?
- As componentes de G^{CFC} são visitadas em **ORDEM TOPOLÓGICA**.

Grafo transposto

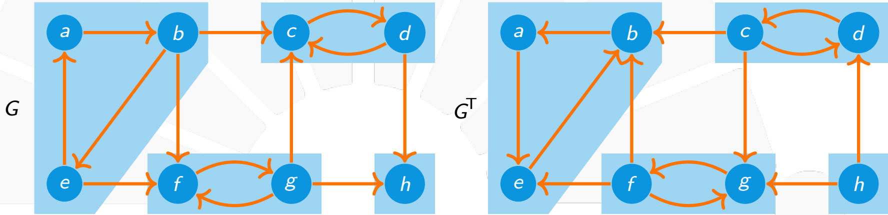
O **GRAFO TRANSPOSTO** de um grafo direcionado $G = (\mathbf{V}, \mathbf{E})$ é um grafo direcionado, denotado por G^{T}, que:

- Tem o mesmo conjunto de vértices **V**.
- Tem uma aresta **(u,v)** se houver uma aresta **(v,u)** em G.

Observações:

- G^{T} é obtido invertendo-se as arestas de G.
- Podemos calcular G^{T} em tempo $O(V + E)$.

Grafo transposto



- Note que G e G^{T} têm as mesmas componentes. Por quê?
- Componentes fontes para G são sorvedouros para G^{T}.
- Se **u** for um vértice de uma fonte em G^{CFC}, então, em G^{T}, os **VÉRTICES ALCANÇÁVEIS** de **u** formam uma componente!

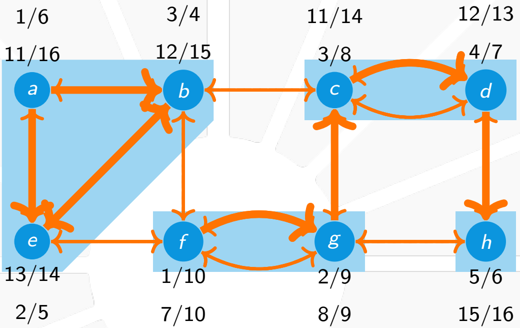
Algoritmo

Algoritmo: $\textsc{Componentes-Fortemente-Conexas}(G)$

1. execute $\mathrm{DFS}(G)$ e calcule $f[v]$ para cada $v \in V$
2. execute $\mathrm{DFS}(G^{\mathsf{T}})$ considerando os vértices em **ORDEM DECRESCENTE** de $f[v]$
3. **devolva** os conjuntos de vértices de cada árvore da floresta de busca encontrada

A complexidade de tempo é $O(V + E)$.

Exemplo



Execute $\mathrm{DFS}(G)$ e calcule $f[\mathbf{v}]$ para cada $\mathbf{v} \in \mathbf{V}$.
Execute $\mathrm{DFS}(G^{\mathsf{T}})$ considerando os vértices em **ORDEM DECRESCENTE** de $f[\mathbf{v}]$.
devolva os conjuntos de vértices de cada árvore da floresta de busca encontrada.

Correção

Teorema

O algoritmo Componentes-Fortemente-Conexas *determina as componentes fortemente conexas de G em tempo $O(V + E)$.*

Antes da demonstração, precisamos de uma preparação.

Lema auxiliar

Lema (1)

Sejam ***C*** *e* ***D*** *duas componentes fortemente conexas e considere vértices* ***u,v*** $\in$ ***C*** *e* ***u',v'*** $\in$ ***D****.*

- *Se existe algum caminho* ***u*** $\rightsquigarrow$ ***u'****,*
- *então* ***NÃO*** *existe um caminho* ***v'*** $\rightsquigarrow$ ***v****.*

- A prova segue da maximalidade de **C** e **D**.
- O lema implica que G^{CFC} é **ACÍCLICO**.

Definições auxiliares

Adotaremos a seguinte convenção:

- Consideramos que em uma execução do algoritmo d e f referem-se à busca em profundidade da linha 1 (em G).
- Para cada subconjunto **U** de vértices, definimos:

$$d(\mathbf{U}) = \min_{\mathbf{u} \in \mathbf{U}} \left\{ d[\mathbf{u}] \right\} \quad \text{e} \quad f(\mathbf{U}) = \max_{\mathbf{u} \in \mathbf{U}} \left\{ f[\mathbf{u}] \right\}$$

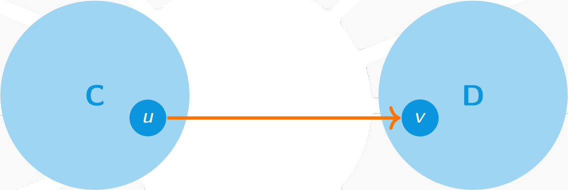
  Em outras palavras:

  - $d(\mathbf{U})$ é o **PRIMEIRO** instante em que um vértice de **U** é descoberto.
  - $f(\mathbf{U})$ é o **ÚLTIMO** instante em que um vértice de **U** é finalizado.

Outro lema auxiliar

Lema (2)

*Sejam **C** e **D** duas componentes fortemente conexas. Se existe aresta **(u,v)** tal que $\mathbf{u} \in \mathbf{C}$ e $\mathbf{v} \in \mathbf{D}$, então $f(\mathbf{C}) > f(\mathbf{D})$.*

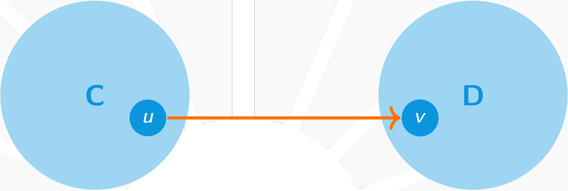


Corolário

*Se G^T tem aresta **(u,v)** tal que $\mathbf{u} \in \mathbf{C}$ e $\mathbf{v} \in \mathbf{D}$, então $f(\mathbf{C}) < f(\mathbf{D})$.*

Segue do fato de que G e G^T têm as mesmas componentes.

Demonstração do lema



Suponha que $d(\mathbf{C}) < d(\mathbf{D})$.

- Isso é, **C** foi descoberto antes de **D**.
- Seja **x** o primeiro vértice de **C** a ser descoberto ($d[\mathbf{x}] = d(\mathbf{C})$).
- No instante $d[\mathbf{x}]$, existia um caminho branco de **x** a cada um dos vértices em $\mathbf{C} \cup \mathbf{D}$.
- Então, todos os vértices de $\mathbf{C} \cup \mathbf{D}$ são descendentes de **x**.
- Portanto, $f(\mathbf{D}) < f[\mathbf{x}] \leq f(\mathbf{C})$.

Demonstração do lema

Agora, suponha que $d(\mathbf{C}) > d(\mathbf{D})$:

- Assim, o primeiro vértice a ser descoberto está em **D**.
- Logo, cada um dos vértices de **D** é finalizado antes de qualquer vértice de **C** ser descoberto.
- Portanto, $f(\mathbf{C}) > f(\mathbf{D})$.

Demonstração do teorema

Teorema

O algoritmo Componentes-Fortemente-Conexas *determina as componentes fortemente conexas de G em tempo $O(V + E)$.*

Demonstração:

- Provaremos que as **PRIMEIRAS** k árvores produzidas na linha 3 correspondem a componentes fortemente conexas.
- A prova é por indução em k.
- Quando $k = 0$, a afirmação é trivial, então tome $k \geq 1$.
- Suponha que as primeiras $k - 1$ árvores produzidas correspondem a componentes fortemente conexas.

Demonstração do teorema

Considere a *k*-ésima árvore produzida pelo algoritmo:

- Sejam **u** a raiz dessa árvore de busca e **C** a componente fortemente conexa que contém **u**.
- Mostraremos que a árvore produzida contém **TODOS** os vértices de **C** e **SOMENTE** os vértices de **C**.
- Isso completará a indução e a prova do teorema.

Demonstração do teorema

A árvore contém **TODOS** os vértices de **C**:

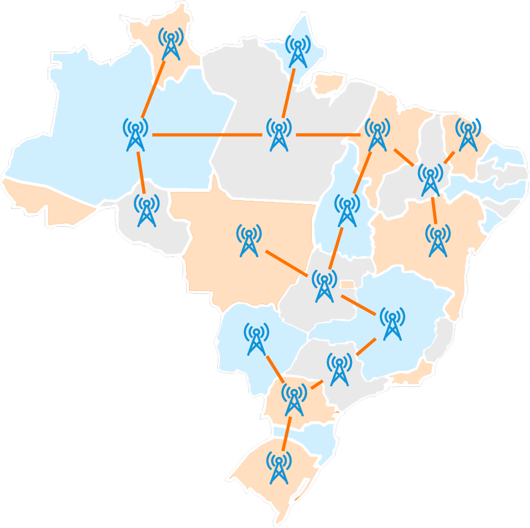
- Considere o instante $d[\mathbf{u}]$, em que **u** é descoberto.
- Por indução, nenhum vértice de **C** foi finalizado.
- Então, no instante $d[\mathbf{u}]$ os vértices de **C** são brancos.
- Portanto, todos os vértices de **C** tornam-se descendentes de **u** na árvore de busca de G^T.

A árvore contém **SOMENTE** vértices de **C**:

- Suponha que existe uma aresta **(x,v)** que sai de **C**.
- Seja **D** a componente fortemente conexa que contém **v**.
- Pelo corolário do Lema 2, temos que $f(\mathbf{C}) < f(\mathbf{D})$.
- Então, descobrimos vértices de **D** antes de **u**.
- Por indução, todos vértices de **D** já foram finalizados.
- Portanto, a árvore só contém vértices de **C**.

Árvore geradora mínima

Motivação



Motivação

Na situação anterior:

- Temos um conjunto de torres.
- Para conectar duas torres usamos um cabeamento.
- Queremos que todas elas estejam interconectadas.

Como **MINIMIZAR** o comprimento do cabeamento?

- Este é um problema de otimização.
- A entrada pode ser modelada como um grafo

Árvore geradora mínima

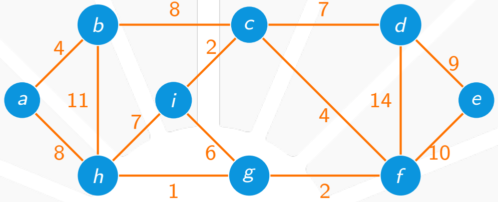
Problema (Árvore geradora mínima (AGM))

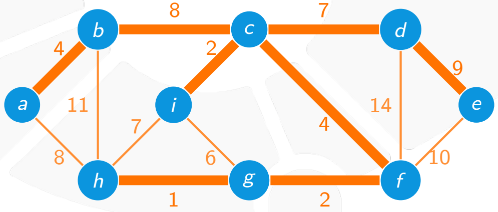
Entrada: *Um grafo conexo* $G = (V, E)$ *com peso* $w(u,v) \geq 0$ *para cada aresta* (u,v).

Solução: *Um subgrafo gerador conexo* T *de* G.

Objetivo: ***MINIMIZAR***: $w(T) = \sum_{(u,v) \in E[T]} w(u,v)$.

Exemplo


a
b
c
d
e
f
g
h
i
4
8
11
8
7
2
7
1
6
2
4
14
9
10


a
b
c
d
e
f
g
h
i
4
8
11
8
7
2
7
1
6
2
4
14
9
10

Refletindo um pouco

Observações:

- Se o grafo fosse desconexo, então não haveria solução.
- Supomos que **NÃO** há arestas de peso negativo.

Algumas perguntas:

- Por que dizemos que uma solução ótima é uma **ÁRVORE**?
- Vale se houvesse arestas de peso negativo na entrada?

Algoritmos estudados

Veremos dois algoritmos **GULOSOS**:

1. Algoritmo de Prim.
2. Algoritmo de Kruskal.

Esquema dos algoritmos

Ideia:

- Construiremos uma árvore incrementalmente.
- Denotamos por **A** um conjunto de arestas da árvore.
- Garantimos que **A** seja um subconjunto de uma AGM.
- Mantemos essa invariante em cada iteração:
    1. No início da iteração, **A** satisfaz a invariante.
    2. Selecionamos **(u,v)** tal que **A** $\cup$ **{(u,v)}** também a satisfaça.
    3. Adicionamos **(u,v)** ao conjunto **A**.

Dizemos que uma tal **(u,v)** é uma **ARESTA SEGURA**.

Algoritmo genérico

Algoritmo: $\textsc{AGM-Generico}(G, w)$

$A \leftarrow \emptyset$
enquanto *A não é uma árvore geradora*
    encontre uma aresta segura (u, v) para A
    $A \leftarrow A \cup \{(u, v)\}$
devolva A

O algoritmo está correto:

- O algoritmo devolve uma árvore geradora **A**.
- **A** é subgrafo de alguma AGM.
- Logo, **A** é também mínima.

Algoritmo genérico

O algoritmo está bem definido?

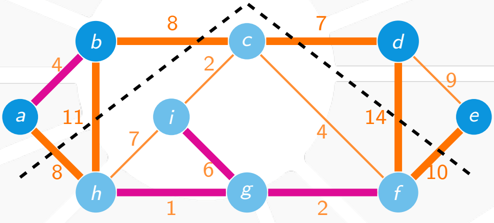
- Se a iteração executa, então **A** não é árvore geradora.
- Logo, **A** não contém todas arestas de alguma AGM T.
- Assim qualquer aresta de **E[T]** $\setminus$ **A** é segura.

Os algoritmos que veremos diferem em como encontrar uma **ARESTA SEGURA**.

Como encontrar arestas seguras?

Considere um grafo $G = (\mathbf{V}, \mathbf{E})$ e seja $\mathbf{S} \subset \mathbf{V}$.

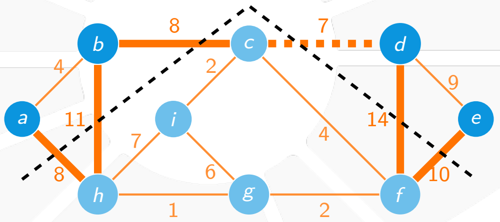
- Denote por $\delta(\mathbf{S})$ o conjunto de arestas de G com um extremo em $\mathbf{S}$ e outro em $\mathbf{V} \setminus \mathbf{S}$.
- Lembre-se de que um tal conjunto é chamado de **CORTE**.



Um corte $\delta(\mathbf{S})$ **RESPEITA** um conjunto $\mathbf{A}$ de arestas se não contiver nenhuma aresta de $\mathbf{A}$.

Arestas leves

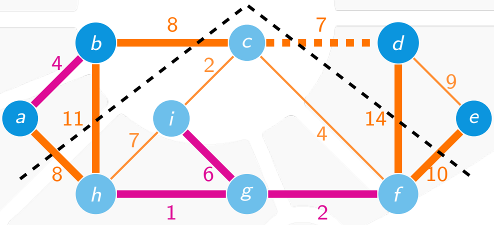
Uma aresta de um corte $\delta(S)$ é **LEVE** se tem o menor peso entre as arestas do corte.



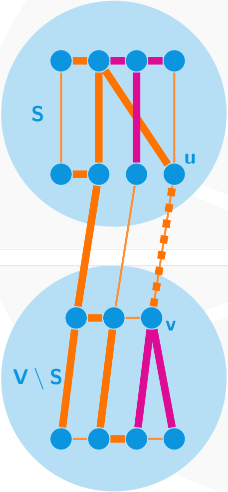
Arestas leves

Teorema

*Seja (G, w) um grafo com pesos nas arestas e suponha que **A** é um subconjunto de arestas de uma AGM de G. Se $\delta(\mathbf{S})$ é um corte que respeita **A** e **(u,v)** é uma aresta leve desse corte, então **(u,v)** é uma **ARESTA SEGURA**.*



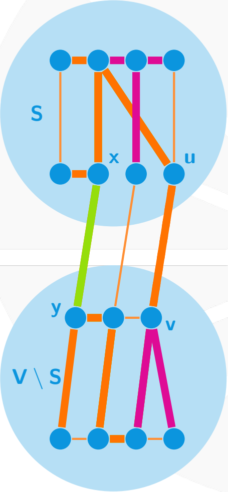
Demonstração do teorema



Seja **T** uma AGM que contém **A**:

- Tome δ(**S**) um corte que respeita **A**.
- Seja **(u,v)** uma **ARESTA LEVE** deste corte.
- Se **(u,v)** estiver em T, então não há nada a mostrar.
- Se **(u,v)** **NÃO** for uma aresta de T:
  - Construiremos uma AGM T' que contém **A** $\cup$ **{(u,v)}**,
  - concluindo que **(u,v)** é **SEGURA**.

Demonstração do teorema



Suponha que **(u,v)** **NÃO** for uma aresta de T:

- Existe um único caminho **P** de **u** a **v** em T.
- **u** a **v** estão em lados opostos do corte δ(**S**).
- Então, alguma aresta de **P** pertence ao corte.
- Seja **(x,y)** uma tal aresta (note que **(x,y)** $\notin$ **A** pois o corte respeita **A**).
- Defina $T' = T -$ **(x,y)** $+$ **(u,v)**.
- Observe que T' é uma árvore geradora.
- Como **(u,v)** é uma **ARESTA LEVE** de δ(**S**), temos que w(**u,v**) $\leq w$(**x,y**).
- Logo, $w(T') = w(T) - w(\mathbf{x,y}) + w(\mathbf{u,v}) \leq w(T)$.
- Portanto, T' é uma **AGM**.

Consequência para os algoritmos

Corolário

Seja (G, w) um grafo com pesos nas arestas e suponha que ***A*** *é um subconjunto de arestas de uma AGM de G. Se* ***C*** *são os vértices de uma componente de $G_A = (\boldsymbol{V}, \boldsymbol{A})$ e* ***(u,v)*** *é uma aresta leve do corte $\delta(\boldsymbol{C})$, então* ***(u,v)*** *é uma* ***ARESTA SEGURA****.*

Isso sugere uma estratégia iterativa:

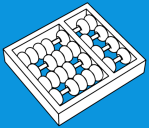
- Prim e Kruskal implementam essa ideia.
- Tais algoritmos farão uso desse corolário.

Conexidade. Árvore geradora mínima

MO417 - Complexidade de Algoritmos I

05/24

20



Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP