

Resumo

- 1 Revisão do conteúdo e objetivo
- 2 Componentes fortemente conexas
- 3 Síntese

Resumo

- 1 Revisão do conteúdo e objetivo
- 2 Componentes fortemente conexas
- 3 Síntese

Revisão do conteúdo

Vimos diversos usos do algoritmo **DFS**: componentes conexas em grafos não direcionados; ordenação topológica e suas aplicações ...

Objetivo

Identificar as **componentes fortemente conexas** de um **grafo direcionado**.

Objetivo

Identificar as **componentes fortemente conexas** de um **grafo direcionado**.

É um problema bem mais complexo do que em grafos não direcionados.

Resumo

- 1 Revisão do conteúdo e objetivo
- 2 Componentes fortemente conexas
- 3 Síntese

Componentes fortemente conexas

Lembrando:

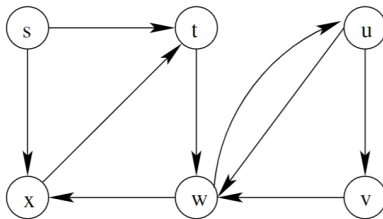
Grafo direcionado fuertemente conexo

Um grafo direcionado $G = (V, A)$ é fortemente conexo se para quaisquer $u, v \in V$, existe um caminho (direcionado) de u a v em G .

Lembrando:

Um grafo direcionado $G = (V, A)$ é fortemente conexo se para quaisquer $u, v \in V$, existe um caminho (direcionado) de u a v em G .

Nem todo grafo é fortemente conexo ...



Componentes fortemente conexas

Componente fortemente conexa

Uma **componente fortemente conexa** de um grafo direcionado $G = (V, A)$ é um subconjunto **maximal** de vértices $C \subseteq V$ tal que o subgrafo induzido por C é fortemente conexo.

C é **maximal** se não existe outro conjunto de vértices C' , que contém a C , tal que o subgrafo induzido por C' é fortemente conexo.

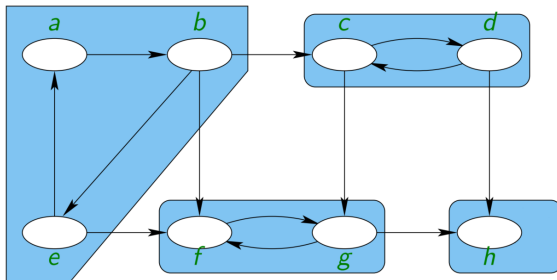
Componentes fortemente conexas

Componente fortemente conexa

Uma **componente fortemente conexa** de um grafo direcionado $G = (V, A)$ é um subconjunto **maximal** de vértices $C \subseteq V$ tal que o subgrafo induzido por C é fortemente conexo.

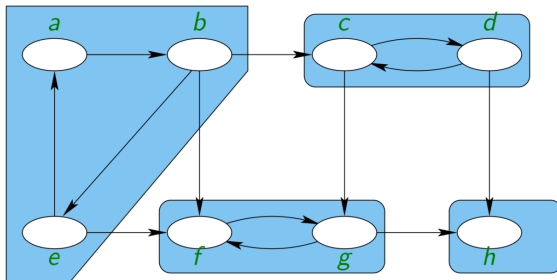
C é **maximal** se não existe outro conjunto de vértices C' , que contém a C , tal que o subgrafo induzido por C' é fortemente conexo. = Se não conseguimos “ampliar” C .

Componentes fortemente conexas



Um grafo direcionado e suas **componentes fortemente conexas**.

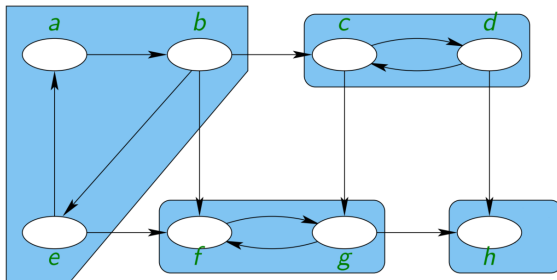
Componentes fortemente conexos



Um grafo direcionado e suas **componentes fortemente conexas**.

Problema: dado um grafo G , encontrar suas componentes fortemente conexas (CFC).

Componentes fortemente conexas



Um grafo direcionado e suas **componentes fortemente conexas**.

Problema: dado um grafo G , encontrar suas componentes fortemente conexas (CFC).
O algoritmo que identifica as CFC de G é baseado no grafo de componentes de G .

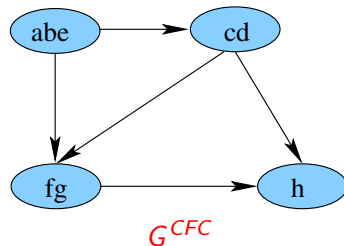
Componentes fortemente conexas

Grafo de componentes

Um grafo de componentes G^{CFC} de um grafo direcionado G é um grafo direcionado, no qual cada vértice no G^{CFC} é uma componente em G , e há aresta (C_1, C_2) em G^{CFC} se existe uma aresta (u, v) em G , tal que $u \in C_1$ e $v \in C_2$.

Grafo de componentes

G



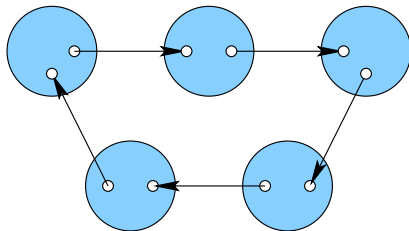
Componentes fortemente conexas

Propriedade fundamental: o grafo de componentes é acíclico.

Componentes fortemente conexos

Propriedade fundamental: o grafo de componentes é acíclico.

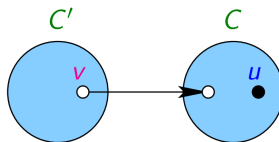
Se G^{CFC} contivesse um ciclo, então alguma componente fortemente conexa estaria contida em outro subgrafo fortemente conexo de G .



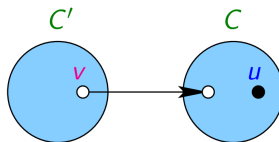
Componentes fortemente conexas

Considere uma busca em profundidade sobre G , e seja u o **último** vértice a ser **finalizado**. A componente fortemente conexa de u é **uma fonte** no grafo de componentes G^{CFC} .

Componentes fortemente conexos

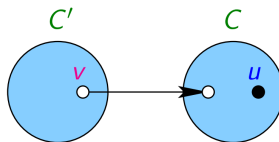


Dica: Compare $d[u]$ com $f[v]$, utilizando a linha do tempo da execução do DFS.



Caso $d[u] < f[v]$: (chegue a uma contradição).

Componentes fortemente conexos



Dica: Compare $d[u]$ com $f[v]$, utilizando a linha do tempo da execução do DFS.

Caso $d[u] < f[v]$: (chegue a uma contradição).

Caso $f[v] < d[u]$: (chegue a uma contradição).

Componentes fortemente conexas

Grafo transposto

Seja $G = (V, E)$ um grafo direcionado. O **grafo transposto** de G é o grafo $G^T = (V, E^T)$ tal que $E^T = \{(u, v) : (v, u) \in E\}$.

Componentes fortemente conexas

Grafo transposto

Seja $G = (V, E)$ um grafo direcionado. O **grafo transposto** de G é o grafo $G^T = (V, E^T)$ tal que $E^T = \{(u, v) : (v, u) \in E\}$.

Ou seja, G^T é obtido a partir de G invertendo as orientações das arestas.

Componentes fortemente conexas

Grafo transposto

Seja $G = (V, E)$ um grafo direcionado. O **grafo transposto** de G é o grafo $G^T = (V, E^T)$ tal que $E^T = \{(u, v) : (v, u) \in E\}$.

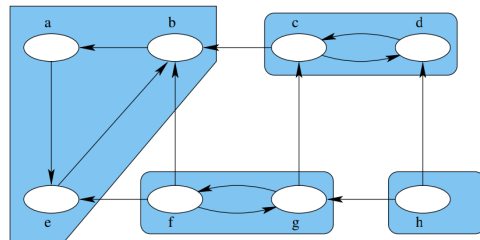
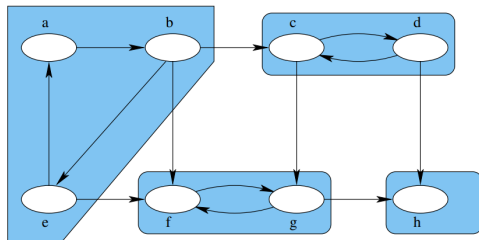
Ou seja, G^T é obtido a partir de G invertendo as orientações das arestas.

Dada uma representação de listas de adjacências de G é possível obter a representação de listas de adjacências de G^T em tempo $\Theta(V + E)$.

Componentes fortemente conexos

Ideia fundamental para entender o algoritmo para encontrar as componentes f.c.:

- Os grafos G e G^T têm as mesmas componentes.
- Se iniciamos uma BP em G^T começando no “último” vértice finalizado u (último vértice a ser finalizado em uma busca em profundidade anterior), a **primeira árvore encontrada** terá os vértices da componente f.c. de u .



Componentes fortemente conexas

Desenvolvendo essa ideia:

- **Fontes** no grafo de componentes de G , são **sorvedouros** no grafo de componentes de G^T .

Componentes fortemente conexas

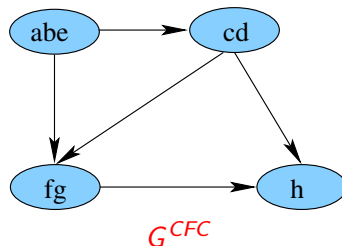
Desenvolvendo essa ideia:

- **Fontes** no grafo de componentes de G , são **sorvedouros** no grafo de componentes de G^T .
- Se temos um **vértice u de uma fonte** em G^{CFC} , os **vértices alcançáveis** de u em G^T formam uma componente f.c. de G .

Componentes fortemente conexos

Desenvolvendo essa ideia:

- **Fontes** no grafo de componentes de G , são **sorvedouros** no grafo de componentes de G^T .
- Se temos um **vértice u de uma fonte** em G^{CFC} , os **vértices alcançáveis** de u em G^T formam uma **componente f.c.** de G .



Componentes fortemente conexas: o algoritmo

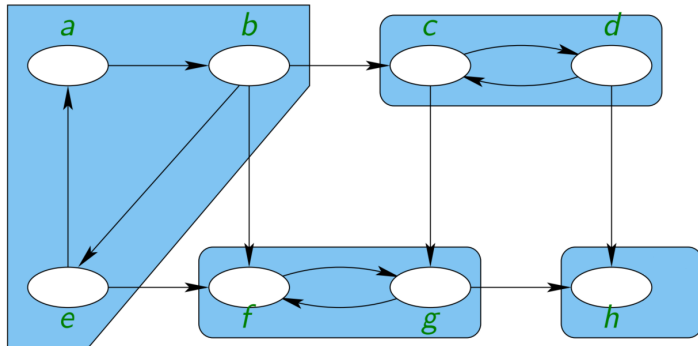
Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

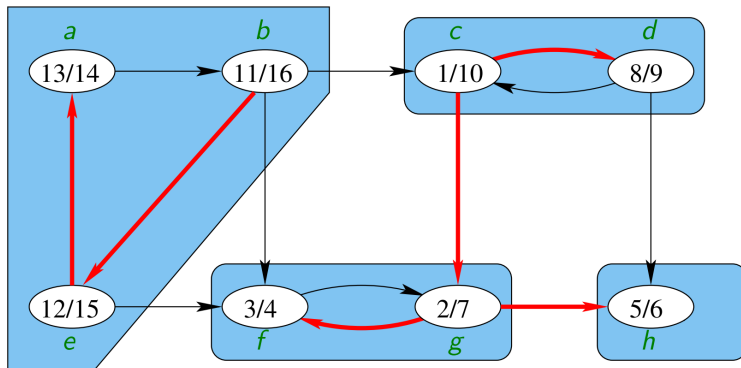
→ Os conjuntos de vértices devolvidos são as componentes fortemente conexas de G .

Em alguns livros é chamado de Algoritmo de Kosaraju.

Componentes fortemente conexos

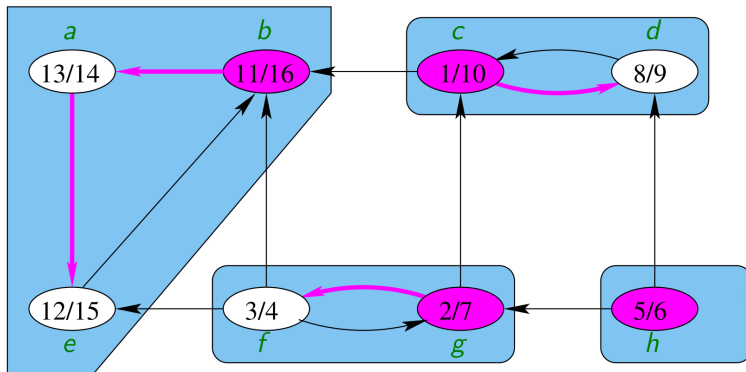


Componentes fortemente conexos



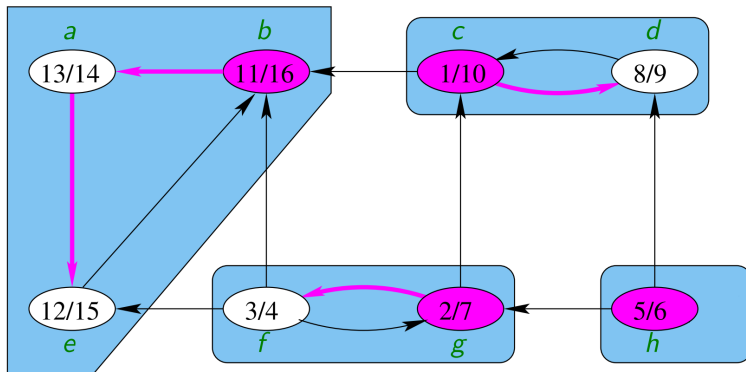
1. Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.

Componentes fortemente conexos



2. Executar $DFS(G^T)$, considerando os vértices em ordem decrescente de $f[v]$.

Componentes fortemente conexos



3. Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca.

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.
2. Calcular G^T , executar $DFS(G^T)$

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.
2. Calcular G^T , executar $DFS(G^T)$ – Tempo $O(V + E)$.

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.
2. Calcular G^T , executar $DFS(G^T)$ – Tempo $O(V + E)$.
3. Devolver os **conjuntos de vértices** de **cada árvore**

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.
2. Calcular G^T , executar $DFS(G^T)$ – Tempo $O(V + E)$.
3. Devolver os **conjuntos de vértices** de **cada árvore** – Tempo $O(V)$.

Componentes fortemente conexos: complexidade de tempo

Algoritmo CFC(G)

- 1 Executar $DFS(G)$, calculando $f[v]$ para cada $v \in V$.
- 2 Executar $DFS(G^T)$, considerando os vértices em **ordem decrescente** de $f[v]$.
- 3 Devolver os **conjuntos de vértices** de **cada árvore** da floresta de busca encontrada.

Complexidade de tempo:

1. Executar $DFS(G)$ – Tempo $O(V + E)$.
 2. Calcular G^T , executar $DFS(G^T)$ – Tempo $O(V + E)$.
 3. Devolver os **conjuntos de vértices** de **cada árvore** – Tempo $O(V)$.
- Complexidade $O(V + E)$.

Componentes fortemente conexas: correção do algoritmo

Correção do algoritmo **CFC**

O algoritmo **CFC** determina as componentes fortemente conexas de G .

Para provar a correção do algoritmo, precisamos de **dois lemas**.

Componentes fortemente conexas

Lema 1

Sejam C_1 e C_2 duas componentes f.c. de G , e sejam $u_1, v_1 \in C_1$, $u_2, v_2 \in C_2$.

Se existe um caminho de u_1 a u_2 em G , então **não** existe um caminho de v_2 a v_1 em G .

Prova:

Componentes fortemente conexas

Lema 1

Sejam C_1 e C_2 duas componentes f.c. de G , e sejam $u_1, v_1 \in C_1$, $u_2, v_2 \in C_2$.

Se existe um caminho de u_1 a u_2 em G , então **não** existe um caminho de v_2 a v_1 em G .

Prova: O grafo de componentes é acíclico, pela maximalidade de cada componente.
Se tivéssemos um caminho de v_2 a v_1 , existiria um ciclo de comprimento 2 no grafo de componentes.

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no Passo 1 do algoritmo.

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no Passo 1 do algoritmo.
- Seja U um subconjunto de vértices de G .

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no Passo 1 do algoritmo.
- Seja U um subconjunto de vértices de G .

$$d(U) = \min_{u \in U} \{d[u]\}$$

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no Passo 1 do algoritmo.
- Seja U um subconjunto de vértices de G .

$$d(U) = \min_{u \in U} \{d[u]\}$$

- Ou seja, $d(U)$ é o instante em que o **primeiro vértice** de U foi descoberto.

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no Passo 1 do algoritmo.
- Seja U um subconjunto de vértices de G .

$$d(U) = \min_{u \in U} \{d[u]\}$$

- Ou seja, $d(U)$ é o instante em que o **primeiro vértice** de U foi descoberto.

$$f(U) = \max_{u \in U} \{f[u]\}$$

Componentes fortemente conexos: definições auxiliares

Definições auxiliares:

- Daqui pra frente $d[v]$, $f[v]$ referem-se à busca em profundidade em G feita no **Passo 1** do algoritmo.
- Seja U um subconjunto de vértices de G .

$$d(U) = \min_{u \in U} \{d[u]\}$$

- Ou seja, $d(U)$ é o instante em que o **primeiro vértice** de U foi descoberto.

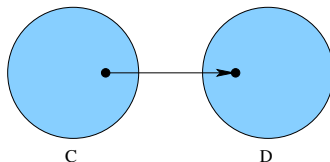
$$f(U) = \max_{u \in U} \{f[u]\}$$

- Ou seja, $f(U)$ é o instante em que o **último vértice** de U foi finalizado.

Componentes fortemente conexas

Lema 2

Sejam C e D duas componentes fortemente conexas no grafo direcionado $G = (V, A)$. Se existe um arco $(u, v) \in A$ tal que $u \in C$ e $v \in D$, então $f(C) > f(D)$.

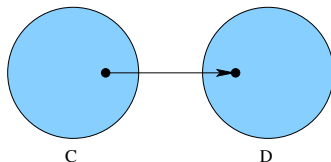


Prova

Componentes fortemente conexas

Lema 2

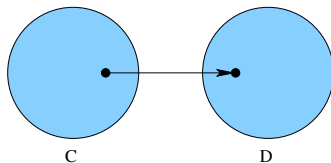
Sejam C e D duas componentes fortemente conexas no grafo direcionado $G = (V, A)$. Se existe um arco $(u, v) \in A$ tal que $u \in C$ e $v \in D$, então $f(C) > f(D)$.



Prova (Caso 1): $d(C) < d(D)$. Seja x o primeiro vértice de C que foi descoberto, logo $d[x] = d(C)$. No instante $d[x]$, existe um caminho branco de x a todo vértice de $C \cup D$. Pelo Teorema do Caminho Branco, todos os vértices de $C \cup D$ são descendentes de x . Portanto $f(D) < f[x] = f(C)$.

Componentes fortemente conexos

Existe um arco $(u, v) \in A$ tal que $u \in C$ e $v \in D \Rightarrow f(C) > f(D)$



Prova (Caso 2): $d(C) > d(D)$. O primeiro vértice de $C \cup D$ a ser descoberto pertence a D . Como D é fortemente conexo, todos os vértices de D serão finalizados antes de qualquer vértice de C ser descoberto, portanto $f(C) > f(D)$.

Componentes fortemente conexas

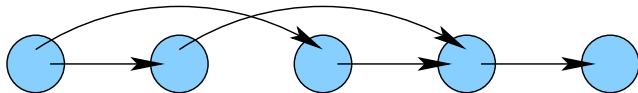
Corolário

Sejam C e D duas componentes fortemente conexas no grafo direcionado G . Se existe um arco (u, v) no grafo G^T tal que $u \in C$ e $v \in D$, então $f(C) < f(D)$.

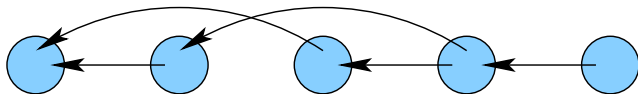
Veja que os grafos G e G^T têm as mesmas componentes f.c.: é suficiente aplicar o **Lema 2** para o arco (v, u) em G .

Componentes fortemente conexos vs ordenação topológica

O grafo G^{CFC} é direcionado e acíclico \Rightarrow possui uma **ordenação topológica**. Veja que nós **já temos** essa **O.T.**: é a ordenação dos vértices (componentes f.c. de G) em ordem decrescente do valor $f(C)$.



Quando $\text{DFS}(G^T)$ começa a construir uma nova árvore, ele escolhe como **raiz** um vértice de uma componente f.c. que é uma **fonte** do grafo G^{CFC} “**desconsiderando**” **as componentes com vértices já visitados anteriormente**.



Componentes fortemente conexas

Correção do algoritmo **CFC**

O algoritmo **CFC** determina as componentes fortemente conexas de G .

Prova por **indução** no número k de árvores produzidas.

Componentes fortemente conexas

Correção do algoritmo CFC

O algoritmo **CFC** determina as componentes fortemente conexas de G .

Prova por **indução** no número k de árvores produzidas.

Vamos provar que as k primeiras árvores produzidas correspondem a componentes f.c. de G , para todo k .

Base: $k = 0$ (trivial)

Hipótese de indução: as primeiras k árvores produzidas são componentes f.c.

Componentes fortemente conexas

Passo de indução: considere a $(k + 1)$ -ésima árvore produzida pelo algoritmo.
Vamos mostrar que o conjunto de vértices dessa árvore é uma componente f.c.

Componentes fortemente conexas

Passo de indução: considere a $(k + 1)$ -ésima árvore produzida pelo algoritmo. Vamos mostrar que o conjunto de vértices dessa árvore é uma componente f.c.

Seja u a raiz dessa árvore, e seja C a componente fortemente conexa de u .

Pela escolha do algoritmo, $f[u] = f(C) > f(D)$ para qualquer outra componente fortemente conexa D de vértices ainda não visitados em $\mathbf{DFS}(G^T)$.

Componentes fortemente conexas

Passo de indução: considere a $(k + 1)$ -ésima árvore produzida pelo algoritmo. Vamos mostrar que o conjunto de vértices dessa árvore é uma componente f.c.

Seja u a raiz dessa árvore, e seja C a componente fortemente conexa de u .

Pela escolha do algoritmo, $f[u] = f(C) > f(D)$ para qualquer outra componente fortemente conexa D de vértices ainda não visitados em $\text{DFS}(G^T)$.

Quando a busca começa em u , todos os vértices de C são brancos. Pelo **Teorema do Caminho Branco**, todos os vértices de C tornam-se descendentes de u na árvore.

Componentes fortemente conexas

Passo de indução: considere a $(k + 1)$ -ésima árvore produzida pelo algoritmo. Vamos mostrar que o conjunto de vértices dessa árvore é uma componente f.c.

Seja u a raiz dessa árvore, e seja C a componente fortemente conexa de u .

Pela escolha do algoritmo, $f[u] = f(C) > f(D)$ para qualquer outra componente fortemente conexa D de vértices ainda não visitados em $\mathbf{DFS}(G^T)$.

Quando a busca começa em u , todos os vértices de C são brancos. Pelo **Teorema do Caminho Branco**, todos os vértices de C tornam-se descendentes de u na árvore.

Pelo **Corolário**, qualquer arco que sai de C só pode entrar em uma das k componentes fortemente conexas já visitadas.

Componentes fortemente conexas

Passo de indução: considere a $(k + 1)$ -ésima árvore produzida pelo algoritmo. Vamos mostrar que o conjunto de vértices dessa árvore é uma componente f.c.

Seja u a raiz dessa árvore, e seja C a componente fortemente conexa de u .

Pela escolha do algoritmo, $f[u] = f(C) > f(D)$ para qualquer outra componente fortemente conexa D de vértices ainda não visitados em $\text{DFS}(G^T)$.

Quando a busca começa em u , todos os vértices de C são brancos. Pelo **Teorema do Caminho Branco**, todos os vértices de C tornam-se descendentes de u na árvore.

Pelo **Corolário**, qualquer arco que sai de C só pode entrar em uma das k componentes fortemente conexas já visitadas. Portanto:

- (1) os vértices de C estão na árvore de busca em profundidade de G^T com raiz u ; e
- (2) apenas esses vértices estão na árvore.

Resumo

- 1 Revisão do conteúdo e objetivo
- 2 Componentes fortemente conexas
- 3 Síntese

Síntese

- O algoritmo **CFC(G)** identifica as **componentes fortemente conexas** de um **grafo direcionado**.
- As componentes fortemente conexas são geradas seguindo uma **ordenação topológica** dentro do grafo de componentes.

- Identificar grupos de **sítios web** relacionados, mas **não alcançáveis** desde a parte da Web indexada pelos motores de busca (**deep web**, ou **web obscura**, corresponde à parte não indexada, e **surface web** é a parte indexada).
- Detectar grupos fortemente conectados de indivíduos (**comunidades**) dentro das redes sociais.

Material bibliográfico e exercícios

T. Cormen et al. Algoritmos - Teoria e Prática (3a ed.). – **Cap. 22** (22.5)

Exercícios: ver exercícios no final do capítulo 22.5.

Dúvidas

Dúvidas?