

Aula 25: Grafos: algoritmos elementares (II)

David Déharbe

Programa de Pós-graduação em Sistemas e Computação

Universidade Federal do Rio Grande do Norte

Centro de Ciências Exatas e da Terra

Departamento de Informática e Matemática Aplicada

Download me from <http://DavidDeharbe.github.io>



Ordenação topológica

Componentes fortemente conectados

Referência: Cormen, cap 23.

Ordenação topológica

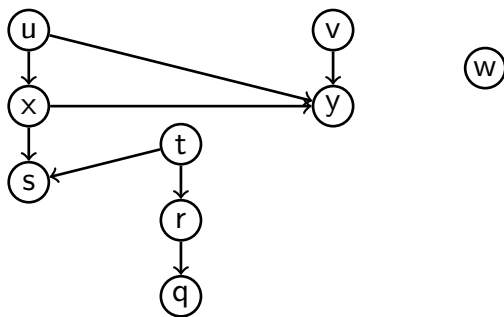
- ▶ Grafos dirigidos acíclicos indicam uma relação de precedência
 - ▶ $u \prec v$: u vem antes de v , aresta (u, v)
 - ▶ relação parcial
 - ▶ exemplos: pré-requisitos entre componentes curriculares, entre tarefas, etc.
- ▶ Ordenação topológica:
 - ▶ entrada: uma relação de precedência
 - ▶ saída: um “escalonamento” dos vértices que respeita a precedência
- ▶ exemplo: grade curricular

Definição

Definição (Grafo dirigido acíclico, DAG)

Um *grafo dirigido acíclico*, ou DAG^1 é um grafo dirigido G tal que se existe uma aresta dirigida de (u, v) , então não existe caminho de v até u .

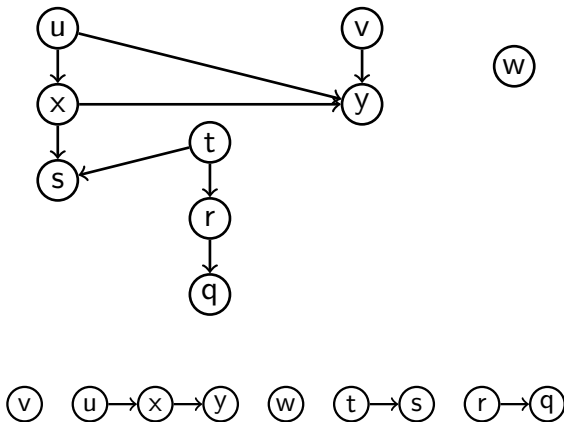
Não há ciclo em DAGs.



¹Do inglês *Directed Acyclic Graph*.

Exemplo

Ordenação topológica



Algoritmo

Ordenação topológica

- ▶ Aplicar DFS;
- ▶ Retornar a lista dos vértices em ordem decrescente do atributo f .

TOPOLOGICAL-SORT(G)

 aplique DFS a G

 inserindo cada v na cabeça de uma lista quando é finalizado

return a lista dos vértices



Algoritmo detalhado

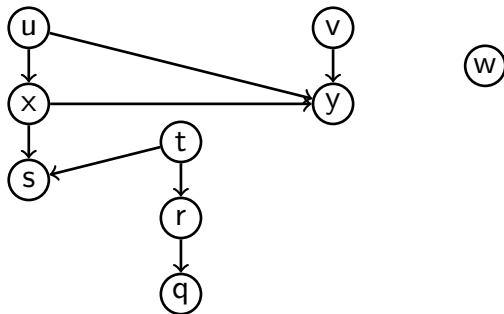
Ordenação topológica

```
TOPOLOGICAL-SORT( $G$ )  
  for  $v \in G.V$   
     $v.visited = \text{FALSE}$   
   $l = \text{EMPTY-LIST}$   
  for  $v \in G.V$   
    if  $\neg v.visited$   
      TOPOLOGICAL-SORT-VISIT( $v$ )  
  return  $l$ 
```

```
TOPOLOGICAL-SORT-VISIT( $v, l$ )  
   $v.visited = \text{TRUE}$   
  for  $w \in v.adj$   
    if  $\neg w.visited$   
      TOPOLOGICAL-SORT-VISIT( $w, l$ )  
   $l = \text{PUSH-FRONT}(v, l)$ 
```

Algoritmo

Exemplo



Complexidade

Ordenação topológica

- ▶ DFS: $\Theta(|V| + |E|)$
- ▶ + $\Theta(1)$ cada inserção de vértice $\equiv \Theta(|V|)$
- ▶ = $\Theta(|V| + |E|)$

1. Lema: caracterização de DAG por arestas
2. Teorema: correção do algoritmo proposto

Lema (Arestas e DAG)

Um grafo dirigido $G = (V, E)$ é acíclico (um DAG) se e somente se qualquer aplicação de DFS(G) encontra nenhuma aresta de volta.



Lema (Arestas e DAG)

Um grafo dirigido $G = (V, E)$ é acíclico (um DAG) se e somente se qualquer aplicação de $\text{DFS}(G)$ encontra nenhuma aresta de volta.

Plano de prova

- ▶ (\Leftarrow) nenhuma aresta de volta \Rightarrow nenhum ciclo
- ▶ (\Rightarrow) nenhum ciclo \Rightarrow nenhuma aresta de volta

Aciclicidade e arestas

Demonstração

Demonstração.

(\Leftarrow) nenhuma aresta de volta \Rightarrow nenhum ciclo
 \equiv ciclo \Rightarrow aresta de volta

- ▶ G possui um ciclo c
- ▶ seja v o primeiro vértice de c encontrado em uma busca em profundidade
- ▶ seja (u, v) a aresta de c chegando em v
- ▶ pelo teorema do caminho branco: na etapa $v.d$, há um caminho branco até u
- ▶ u torna-se um descendente de v na floresta em profundidade
- ▶ logo, (u, v) é uma aresta de volta.



Aciclicidade e arestas

Demonstração

Demonstração.

(\Rightarrow) nenhum ciclo \Rightarrow nenhuma aresta de volta

\equiv

aresta de volta \Rightarrow ciclo

- ▶ seja (u, v) uma aresta de volta.
- ▶ logo v é um ancestral de u na floresta de profundidade.
- ▶ então há um caminho de v até u , passando por u : é um ciclo



Teorema (Correção do algoritmo TOPOLOGICAL-SORT)

$\text{TOPOLOGICAL-SORT}(G)$ *produz uma ordenação topológica de um grafo dirigido acíclico G .*

Demonstração.

- ▶ Basta mostrar que: em um DAG, se há uma aresta (u, v) , então $v.f < u.f$.
- ▶ (u, v) não é uma aresta de volta
- ▶ Quando encontrado, v é branco ou preto
 - ▶ se v for branco, é um descendente de u e $v.f < u.f$
 - ▶ se v for preto, já foi finalizado e $v.f < u.d < u.f$.



Componentes fortemente conectados

- ▶ **SCC**: *Strongly connected components*
- ▶ Decomposição de um grafo dirigido em grafos menores
- ▶ Permite, para alguns problemas de grafos, aplicar estratégia de divisão e conquista.

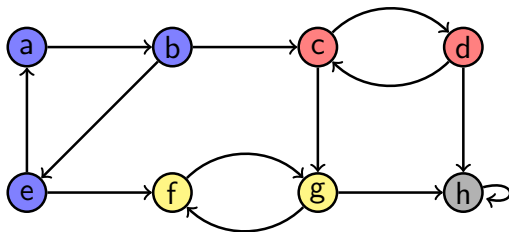


Definição

- Notação: $u \rightsquigarrow v$ existe um caminho de u até v

Definição (Componente fortemente conectado)

Um *componente fortemente conectado* de um grafo $G = (V, E)$ é um conjunto máximo de vértices $U \subseteq V$ tal que para qualquer $(u, v) \in U^2$, então $u \rightsquigarrow v$ e $v \rightsquigarrow u$.

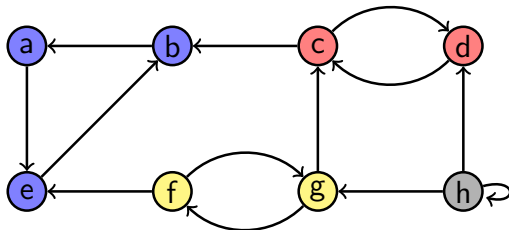


Matriz transposta

O algoritmo para encontrar os SCC de G utiliza o grafo transposto de G .

Definição (Grafo transposto)

O *grafo transposto* de um grafo dirigido $G = (V, E)$ é o grafo dirigido $G^T = (V, E^T)$, onde $E^T = \{(u, v) | (v, u) \in E\}$.

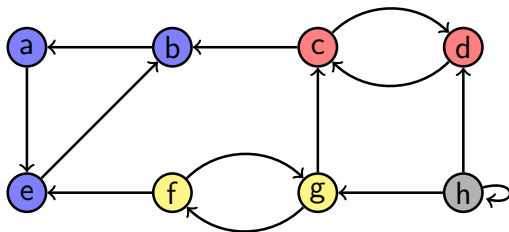


Matriz transposta

O algoritmo para encontrar os SCC de G utiliza o grafo transposto de G .

Definição (Grafo transposto)

O *grafo transposto* de um grafo dirigido $G = (V, E)$ é o grafo dirigido $G^T = (V, E^T)$, onde $E^T = \{(u, v) | (v, u) \in E\}$.



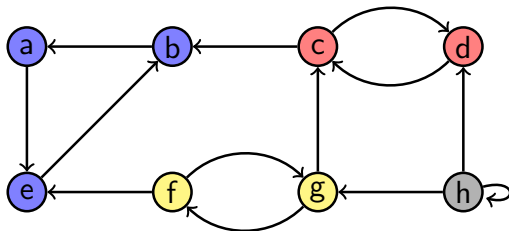
Complexidade?

Matriz transposta

O algoritmo para encontrar os SCC de G utiliza o grafo transposto de G .

Definição (Grafo transposto)

O *grafo transposto* de um grafo dirigido $G = (V, E)$ é o grafo dirigido $G^T = (V, E^T)$, onde $E^T = \{(u, v) | (v, u) \in E\}$.



Complexidade? $\Theta(|V| + |E|)$

Algoritmo

As etapas principais

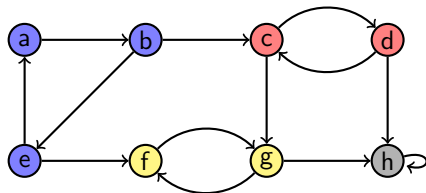
GRAPH-SCC(G)

- 1 DFS(G)
- 2 $G^T = \text{GRAPH-TRANSPOSE}(G)$
- 3 DFS(G^T) t. q. laço principal processa vértices por f decrescente
- 4 cada árvore da floresta de profundidade resultado é um SCC



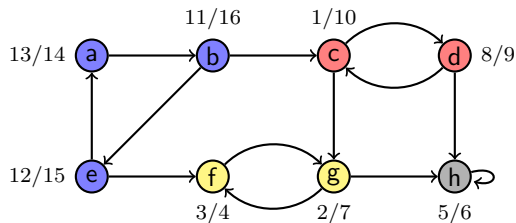
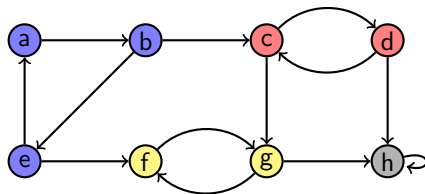
Ilustração

Algoritmo



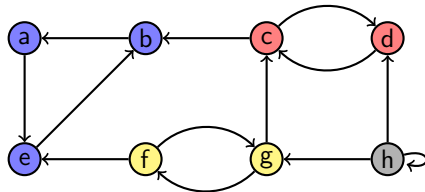
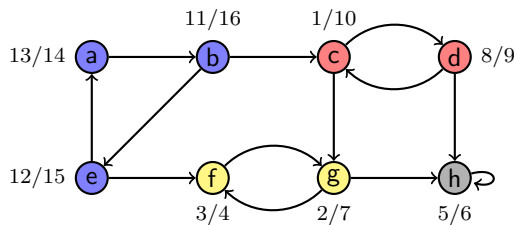
Ilustração

Algoritmo



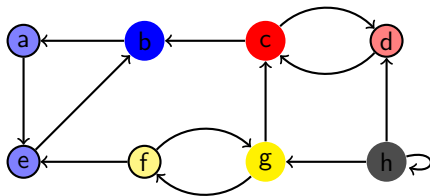
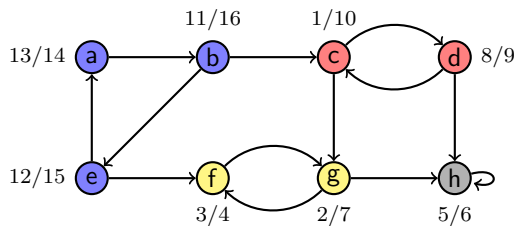
Ilustração

Algoritmo



Ilustração

Algoritmo



GRAPH-SCC(G)

- 1 DFS(G) // $\Theta(|V| + |E|)$
- 2 $G^T = \text{GRAPH-TRANSPOSE}(G)$ // $\Theta(|V| + |E|)$
- 3 DFS(G^T) // $\Theta(|V| + |E^T|) = \Theta(|V| + |E|)$

GRAPH-SCC(G)

- 1 DFS(G) // $\Theta(|V| + |E|)$
- 2 $G^T = \text{GRAPH-TRANSPOSE}(G)$ // $\Theta(|V| + |E|)$
- 3 DFS(G^T) // $\Theta(|V| + |E^T|) = \Theta(|V| + |E|)$

$$\Theta(|V| + |E|)$$

Correção (roteiro)

Algoritmo

1. propriedade dos caminhos entre vértices de um mesmo SCC
2. propriedade sobre busca em profundidade e vértices de um SCC
3. noção de **antepassado** de um vértice em uma busca em profundidade
4. relação entre antepassado na busca e ancestral no grafo
5. propriedade sobre antepassado e SCC
6. propriedade sobre antepassados dos vértices de um mesmo SCC
7. correção do algoritmo



Caminhos entre vértices de um mesmo SCC

Correção

Lema (Caminhos entre vértices de um mesmo SCC)

Seja u e v dois vértices quaisquer de um mesmo SCC. Então todos os vértices nos caminhos entre u e v estão neste mesmo SCC.

Caminhos entre vértices de um mesmo SCC

Correção

Lema (Caminhos entre vértices de um mesmo SCC)

Seja u e v dois vértices quaisquer de um mesmo SCC. Então todos os vértices nos caminhos entre u e v estão neste mesmo SCC.

Demonstração.

Seja w um vértice no caminho de u até v . Logo $u \rightsquigarrow w$.
Precisamos verificar que $w \rightsquigarrow u$.

- ▶ Como w está no caminho de u até v , então $w \rightsquigarrow v$.
- ▶ Como u e v estão no mesmo SCC, $v \rightsquigarrow u$.

Logo $w \rightsquigarrow u$.



Busca em profundidade e vértices de um SCC

Correção

Teorema (Busca em profundidade e vértices de um SCC)

Em qualquer busca em profundidade, todos os vértices em um SCC encontram-se em uma mesma árvore da busca em profundidade.



Busca em profundidade e vértices de um SCC

Correção

Teorema (Busca em profundidade e vértices de um SCC)

Em qualquer busca em profundidade, todos os vértices em um SCC encontram-se em uma mesma árvore da busca em profundidade.

Demonstração.

Seja u o primeiro vértice do SCC encontrado na busca em profundidade, e v qualquer outro vértice do SCC.

- ▶ na etapa $u.d$, todos os demais vértices do SCC estão brancos
- ▶ pelo teorema do caminho branco, v é um descendente de u na floresta de profundidade

Logo, u e v estão na mesma árvore de profundidade.



Antepassado na busca em profundidade

Correção

- ▶ $v.d$ e $v.f$ são os valores obtidos em $\text{DFS}(G)$

Definição (Antepassado em uma busca em profundidade)

Em uma busca em profundidade, para qualquer vértice u , o antepassado de u , denotado $\phi(u)$, é o vértice v tal que $u \rightsquigarrow v$ com o maior valor de f .



Antepassado na busca em profundidade

Correção

- ▶ $v.d$ e $v.f$ são os valores obtidos em $\text{DFS}(G)$

Definição (Antepassado em uma busca em profundidade)

Em uma busca em profundidade, para qualquer vértice u , o antepassado de u , denotado $\phi(u)$, é o vértice v tal que $u \rightsquigarrow v$ com o maior valor de f .

- ▶ um antepassado por SCC (\approx representante do componente)
- ▶ primeiro vértice do SCC descoberto na busca em profundidade de G
- ▶ último a ser finalizado
- ▶ é a raiz da árvore de profundidade na busca em profundidade de G^T



Antepassado na busca em profundidade

Correção

- ▶ $v.d$ e $v.f$ são os valores obtidos em $\text{DFS}(G)$

Definição (Antepassado em uma busca em profundidade)

Em uma busca em profundidade, para qualquer vértice u , o antepassado de u , denotado $\phi(u)$, é o vértice v tal que $u \rightsquigarrow v$ com o maior valor de f .

Temos:

1. $u.f \leq \phi(u).f$
2. $u \rightsquigarrow v \Rightarrow \phi(v).f \leq \phi(u).f$
3. $\phi(\phi(u)) = \phi(u)$



Antepassado na busca em profundidade

Correção

$$u.f \leq \phi(u).f$$

- ▶ Pois $u \rightsquigarrow u$, e $\forall v | u \rightsquigarrow v \cdot v.f \leq \phi(u).f$

$$u \rightsquigarrow v \Rightarrow \phi(v).f \leq \phi(u).f$$

- ▶ Pois $\{w \cdot v \rightsquigarrow w\} \subseteq \{w \cdot u \rightsquigarrow w\}$

$$\phi(\phi(u)) = \phi(u)$$

- ▶ Pois $\phi(u).f \leq \phi(\phi(u)).f$,
- ▶ e, como $u \rightsquigarrow \phi(u)$, então $\phi(\phi(u)).f \leq \phi(u).f$,
- ▶ temos $\phi(u).f = \phi(\phi(u)).f$, e
- ▶ cada vértice tem um valor de f diferente.

Antepassados e ancestrors

Correção

Teorema (Antepassado é ancestral)

Em um grafo dirigido $G = (V, E)$, o antepassado $\phi(u)$ de qualquer $u \in V$ em qualquer busca em profundidade de G , sempre é um ancestral de u .



Teorema (Antepassado é ancestral)

Em um grafo dirigido $G = (V, E)$, o antepassado $\phi(u)$ de qualquer $u \in V$ em qualquer busca em profundidade de G , sempre é um ancestral de u .

Demonstração.

Por caso sobre a cor de $\phi(u)$ na etapa $u.d$



Teorema (Antepassado é ancestral)

Em um grafo dirigido $G = (V, E)$, o antepassado $\phi(u)$ de qualquer $u \in V$ em qualquer busca em profundidade de G , sempre é um ancestral de u .

Demonstração.

Por caso sobre a cor de $\phi(u)$ na etapa $u.d$

- ▶ Se for GRAY, então $\phi(u)$ é ancestral de u .



Antepassados e ancestrors

Correção

Teorema (Antepassado é ancestral)

Em um grafo dirigido $G = (V, E)$, o antepassado $\phi(u)$ de qualquer $u \in V$ em qualquer busca em profundidade de G , sempre é um ancestral de u .

Demonstração.

Por caso sobre a cor de $\phi(u)$ na etapa $u.d$

- ▶ Se for BLACK, então foi finalizado, e $\phi(u).f < u.f$.

Contradiz $u.f \leq \phi(u).f$.



Antepassados e ancestrors

Correção

Teorema (Antepassado é ancestral)

Em um grafo dirigido $G = (V, E)$, o antepassado $\phi(u)$ de qualquer $u \in V$ em qualquer busca em profundidade de G , sempre é um ancestral de u .

Demonstração.

Por caso sobre a cor de $\phi(u)$ na etapa $u.d$

- ▶ Se for WHITE, consideramos a cor dos vértices no caminho de u até $\phi(u)$
 - ▶ todos são brancos, logo $\phi(u)$ é descendente de u ,
 $\phi(u).f < u.f$: **contradição**
 - ▶ senão, seja t o último vértice não branco do caminho:
 - ▶ t não pode ser preto, pois tem um sucessor branco
 - ▶ então há um caminho branco de t até $\phi(u)$, e
 - ▶ $\phi(u)$ é descendente de t (teorema do caminho branco)
 - ▶ logo $t.f > \phi(u).f$
 - ▶ **contradição** por definição de $\phi(u)$ e $u \rightsquigarrow t$.



Corolário (Antepassado e SCC)

Em qualquer busca em profundidade de um grafo dirigido $G = (V, E)$, para qualquer $u \in V$, ambos u e $\phi(u)$ pertencem ao mesmo SCC.

Corolário (Antepassado e SCC)

Em qualquer busca em profundidade de um grafo dirigido $G = (V, E)$, para qualquer $u \in V$, ambos u e $\phi(u)$ pertencem ao mesmo SCC.

Demonstração.

- ▶ Por definição de ϕ , temos $u \rightsquigarrow \phi(u)$.
- ▶ Pelo teorema “antepassado é ancestro”, $\phi(u) \rightsquigarrow u$.



Antepassados dos vértices de um SCC

Correção

Teorema (Antepassados dos vértices de um SCC)

Em um grafo dirigido $G = (V, E)$, os vértices u e v pertencem ao mesmo SCC se, e somente se, possuem o mesmo antepassado na busca em profundidade de G .



Antepassados dos vértices de um SCC

Correção

Teorema (Antepassados dos vértices de um SCC)

Em um grafo dirigido $G = (V, E)$, os vértices u e v pertencem ao mesmo SCC se, e somente se, possuem o mesmo antepassado na busca em profundidade de G .

Demonstração.

- ▶ (\Rightarrow)
- ▶ (\Leftarrow)



Antepassados dos vértices de um SCC

Correção

Teorema (Antepassados dos vértices de um SCC)

Em um grafo dirigido $G = (V, E)$, os vértices u e v pertencem ao mesmo SCC se, e somente se, possuem o mesmo antepassado na busca em profundidade de G .

Demonstração.

- ▶ (\Rightarrow) u e v pertencem ao mesmo SCC:
 - ▶ logo $\{w \cdot u \rightsquigarrow w\} = \{w \cdot v \rightsquigarrow w\}$.
 - ▶ por definição de ϕ , $\phi(u) = \phi(v)$.
- ▶ (\Leftarrow)



Antepassados dos vértices de um SCC

Correção

Teorema (Antepassados dos vértices de um SCC)

Em um grafo dirigido $G = (V, E)$, os vértices u e v pertencem ao mesmo SCC se, e somente se, possuem o mesmo antepassado na busca em profundidade de G .

Demonstração.

- ▶ (\Rightarrow)
- ▶ $(\Leftarrow) \phi(u) = \phi(v)$:
 - ▶ pelo corolário “Antepassado e SCC” u e $\phi(u)$ estão no mesmo SCC
 - ▶ v e $\phi(v)$ estão no mesmo SCC,
 - ▶ logo u e v estão no mesmo SCC.

Intuição do algoritmo

- ▶ $\text{DFS}(G)$ marca os antepassados com o maior valor de f (e o menor valor de d) de cada SCC.
- ▶ $\text{DFS}(G^T)$
 - ▶ começa com um vértice v_1 de maior f , que é um antepassado
 - ▶ visita todos os vértices do SCC de v_1
 - ▶ continua com um vértice v_2 de maior f entre os não visitados, também é um antepassado
 - ▶ visita todos os vértices do SCC de v_2
 - ▶ e assim sucessivamente



Teorema da correção

Correção

Teorema (Correção de GRAPH-SCC)

Seja G um grafo dirigido qualquer, $\text{GRAPH-SCC}(G)$ calcula corretamente os componentes fortemente conectados de G .



Teorema da correção

Correção

Teorema (Correção de GRAPH-SCC)

Seja G um grafo dirigido qualquer, $\text{GRAPH-SCC}(G)$ calcula corretamente os componentes fortemente conectados de G .

Roteiro da demonstração:

- ▶ indução
- ▶ número de árvores de profundidade encontrados em $\text{DFS}(G^T)$.
- ▶ mostramos que, assumindo que as árvores anteriores são SCC, cada nova árvore formada é um SCC
- ▶ trivial para a primeira árvore (não existe árvores anteriores)



Prova do teorema da correção

Correção

Demonstração.

- ▶ Seja T uma árvore de profundidade de raiz r produzida por $\text{DFS}(G^T)$.
- ▶ Seja $C(r) = \{w \cdot \phi(w) = r\}$ (é um SCC)
- ▶ Mostramos que u é incluído em T se e somente se $u \in C(r)$
 - ▶ (\Leftarrow)
 - ▶ (\Rightarrow)



Prova do teorema da correção

Correção

Demonstração.

- ▶ Seja T uma árvore de profundidade de raiz r produzida por $\text{DFS}(G^T)$.
- ▶ Seja $C(r) = \{w \cdot \phi(w) = r\}$ (é um SCC)
- ▶ Mostramos que u é incluído em T se e somente se $u \in C(r)$
 - ▶ (\Leftarrow)
 - ▶ teorema “busca em profundidade e vértices de um SCC”
 - ▶ cada vértice em $C(r)$ termina em uma mesma árvore de profundidade
 - ▶ como $r \in C(r)$, e r é a raiz de T , então cada elemento de $C(r)$ termina precisamente em T .
 - ▶ (\Rightarrow)

Prova do teorema da correção

Correção

Demonstração.

- ▶ Seja T uma árvore de profundidade de raiz r produzida por $\text{DFS}(G^T)$.
- ▶ Seja $C(r) = \{w \cdot \phi(w) = r\}$ (é um SCC)
- ▶ Mostramos que u é incluído em T se e somente se $u \in C(r)$
 - ▶ (\Leftarrow)
 - ▶ (\Rightarrow) Mostramos que, para um vértice w , se $\phi(w).f < r.f$, ou $\phi(w).f > r.f$, w não pertence a T



Prova do teorema da correção

Correção

Demonstração.

- ▶ Seja T uma árvore de profundidade de raiz r produzida por $\text{DFS}(G^T)$.
- ▶ Seja $C(r) = \{w \cdot \phi(w) = r\}$ (é um SCC)
- ▶ Mostramos que u é incluído em T se e somente se $u \in C(r)$
 - ▶ (\Leftarrow)
 - ▶ (\Rightarrow) Mostramos que, para um vértice w , se $\phi(w).f < r.f$, ou $\phi(w).f > r.f$, w não pertence a T
 - ▶ $\phi(w).f < r.f$
 - ▶ se w for colocado em T , então $w \rightsquigarrow r$
 - ▶ logo, $\phi(w).f \leq \phi(r).f = r.f$
 - ▶ **contradição**

Prova do teorema da correção

Correção

Demonstração.

- ▶ Seja T uma árvore de profundidade de raiz r produzida por $\text{DFS}(G^T)$.
- ▶ Seja $C(r) = \{w \cdot \phi(w) = r\}$ (é um SCC)
- ▶ Mostramos que u é incluído em T se e somente se $u \in C(r)$
 - ▶ (\Leftarrow)
 - ▶ (\Rightarrow) Mostramos que, para um vértice w , se $\phi(w).f < r.f$, ou $\phi(w).f > r.f$, w não pertence a T
 - ▶ $\phi(w).f > r.f$
 - ▶ por hipótese de indução, quando r for selecionado em $\text{DFS}(G^T)$, $w.f$ terá sido inserido na árvore de raiz $\phi(w)$.
 - ▶ um vértice é inserido exatamente em uma árvore.

Teorema da correção

Correção

Teorema (Correção de GRAPH-SCC)

Seja G um grafo dirigido qualquer, $\text{GRAPH-SCC}(G)$ calcula corretamente os componentes fortemente conectados de G .



Teorema da correção

Correção

Teorema (Correção de GRAPH-SCC)

Seja G um grafo dirigido qualquer, $\text{GRAPH-SCC}(G)$ calcula corretamente os componentes fortemente conectados de G .

Demonstração.

- ▶ indução
- ▶ número de árvores de profundidade encontrados em $\text{DFS}(G^T)$.
- ▶ mostramos que, assumindo que as árvores anteriores são SCC, cada nova árvore formada é um SCC
- ▶ trivial para a primeira árvore (não existe árvores anteriores)



Exercícios

1. Como pode evoluir a quantidade de SCC em um grafo quando uma aresta é adicionada? removida?
2. Seja G um grafo dirigido. O grafo dos componentes de G é o grafo $G^{SCC} = (V^{SCC}, E^{SCC})$, tal que V^{SCC} contem um vértice por SCC de G , e E^{SCC} contem a aresta (u, v) se existe uma aresta entre um vértice de u e um vértice de v no grafo G .

Mostre que o grafo dos componentes conectados de G é um DAG.

3. Escreva um algoritmo que calcula o grafo dos componentes de $G = (V, E)$, com complexidade $O(|V| + |E|)$.

Nota: o grafo resultado deve ter, ao máximo, uma aresta entre cada par de vértices.

4. Um grafo é semi-conectado se, para qualquer par de vértices (u, v) , ou $u \rightsquigarrow v$ ou $v \rightsquigarrow u$.

Escreva um algoritmo eficiente que testa se um grafo é semi-conectado. Mostre que seu algoritmo é correto.

