Componentes fortemente conexas

CLRS Cap 22.5

Busca em profundidade

DFS (G)

para cada $u \in V(G)$ **faça**
  $u.\text{cor} \leftarrow \text{branco}$ $u.\pi \leftarrow \text{nil}$
$\text{tempo} \leftarrow 0$
para cada $u \in V(G)$ **faça**
  se $u.\text{cor} = \text{branco}$
    então DFS-Visit(u)

Busca em profundidade

DFS (G)

para cada $u \in V(G)$ **faça**
    $u.\mathrm{cor} \leftarrow \mathrm{branco}$ $u.\pi \leftarrow \mathrm{nil}$
$\mathrm{tempo} \leftarrow 0$
para cada $u \in V(G)$ **faça**
    se $u.\mathrm{cor} = \mathrm{branco}$
        então DFS-Visit(u)

DFS-Visit(u)

$u.\mathrm{cor} \leftarrow \mathrm{cinzento}$ $u.d \leftarrow \mathrm{tempo}$ $\mathrm{tempo} \leftarrow \mathrm{tempo} + 1$
para cada $v \in u.\mathrm{Estrela}$ **faça**
    se $u.\mathrm{cor} = \mathrm{branco}$
        então $v.\pi \leftarrow u$
            DFS-Visit(v)
$u.\mathrm{cor} \leftarrow \mathrm{preto}$
$u.f \leftarrow \mathrm{tempo}$ $\mathrm{tempo} \leftarrow \mathrm{tempo} + 1$

A floresta DF

Cada componente da floresta DF é uma árvore enraizada, orientada a partir da raiz.

A floresta DF

Cada componente da floresta DF é uma árvore enraizada, orientada a partir da raiz.

A cada chamada de DFS-Visit no laço principal, os vértices lá processados fazem parte da componente cuja raiz é o argumento da chamada.

A floresta DF

Cada componente da floresta DF é uma árvore enraizada, orientada a partir da raiz.

A cada chamada de DFS-Visit no laço principal, os vértices lá processados fazem parte da componente cuja raiz é o argumento da chamada.

Se o grafo é não dirigido, isso determina as componentes conexas do grafo.

A floresta DF

Cada componente da floresta DF é uma árvore enraizada, orientada a partir da raiz.

A cada chamada de DFS-Visit no laço principal, os vértices lá processados fazem parte da componente cuja raiz é o argumento da chamada.

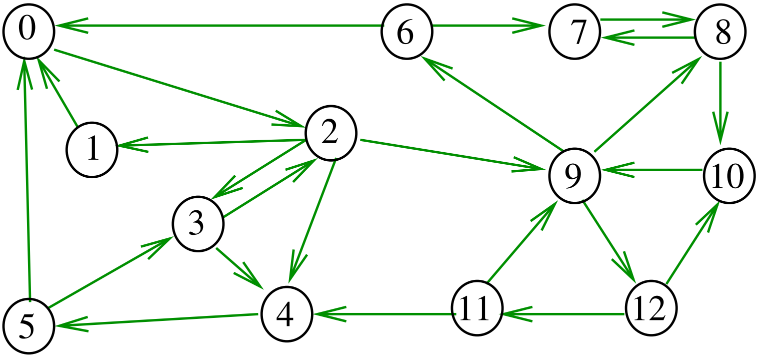
Se o grafo é não dirigido, isso determina as componentes conexas do grafo.

E se o grafo é dirigido?
O que conseguimos inferir de uma floresta DF?

Digrafos fortemente conexos

Um digrafo é fortemente conexo se e somente se para cada par $\{s, t\}$ de seus vértices, existem caminhos de s a t e de t a s.

Exemplo: um digrafo fortemente conexo



Componentes fortemente conexas

Seja G um digrafo e u e v vértices de G.

Escrevemos $u \rightsquigarrow v$ se existe caminho de u para v em G.

Componentes fortemente conexas

Seja G um digrafo e u e v vértices de G.

Escrevemos $u \rightsquigarrow v$ se existe caminho de u para v em G.

Seja C um conjunto maximal de vértices de G tal que $u \rightsquigarrow v$ e $v \rightsquigarrow u$ quaisquer que sejam u e v em C.

Componentes fortemente conexas

Seja G um digrafo e u e v vértices de G.

Escrevemos $u \rightsquigarrow v$ se existe caminho de u para v em G.

Seja C um conjunto maximal de vértices de G tal que $u \rightsquigarrow v$ e $v \rightsquigarrow u$ quaisquer que sejam u e v em C.

C é uma componente fortemente conexa de G.

Componentes fortemente conexas

Seja G um digrafo e u e v vértices de G.

Escrevemos $u \rightsquigarrow v$ se existe caminho de u para v em G.

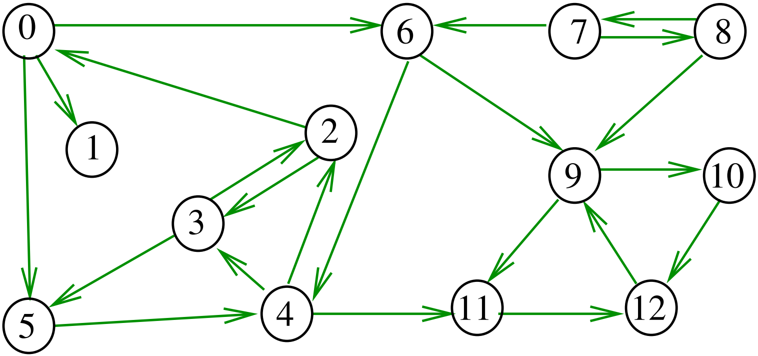
Seja C um conjunto maximal de vértices de G tal que $u \rightsquigarrow v$ e $v \rightsquigarrow u$ quaisquer que sejam u e v em C.

C é uma componente fortemente conexa de G.

Diferentes componentes fortemente conexas são disjuntas, ou seja, as componentes determinam uma partição de V_G.

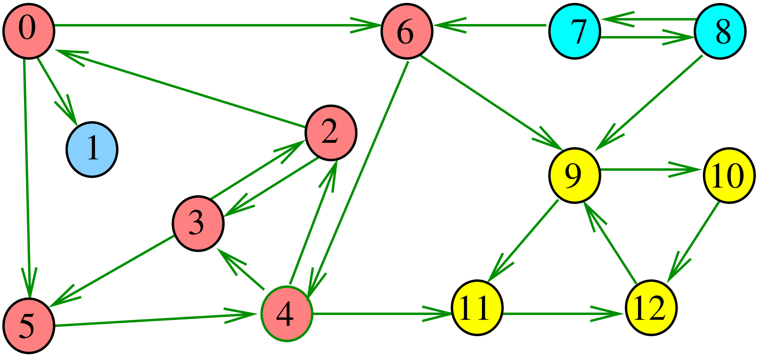
Componentes fortemente conexas

Exemplo: 4 componentes fortemente conexas



Componentes fortemente conexas

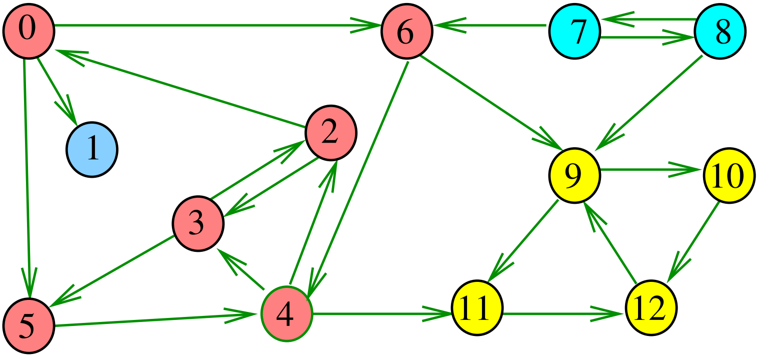
Exemplo: 4 componentes fortemente conexas



Problema: Dado digrafo G, encontrar
todas as componentes fortemente conexas de G.

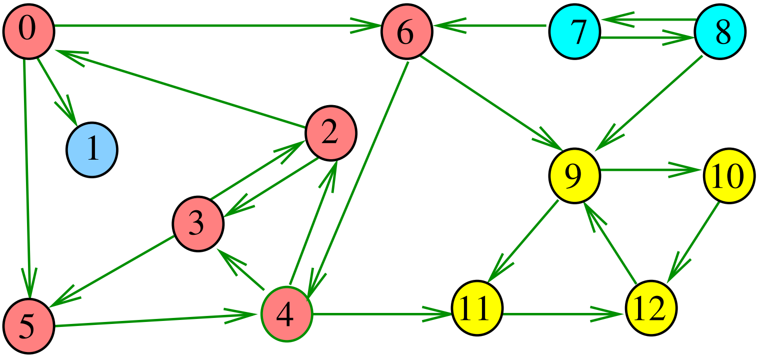
Exemplo

v	0	1	2	3	4	5	6	7	8	9	10	11	12
id[v]	2	1	2	2	2	2	2	3	3	0	0	0	0



Exemplo

v	0	1	2	3	4	5	6	7	8	9	10	11	12
id[v]	2	1	2	2	2	2	2	3	3	0	0	0	0



Algoritmos Tarjan, Kosaraju e Sharir

Robert Endre Tarjan (1972),
Sambasiva Rao Kosaraju (1978)
e Micha Sharir (1981) desenvolveram
algoritmos que consomem tempo $\mathrm{O}(n + m)$
para calcular os componentes f.c. de um digrafo G.

Esses algoritmos utilizam DFS
de uma maneira fundamental.

Algoritmos Tarjan, Kosaraju e Sharir

Robert Endre Tarjan (1972),
Sambasiva Rao Kosaraju (1978)
e Micha Sharir (1981) desenvolveram
algoritmos que consomem tempo $\mathrm{O}(n + m)$
para calcular os componentes f.c. de um digrafo G.

Esses algoritmos utilizam DFS
de uma maneira fundamental.

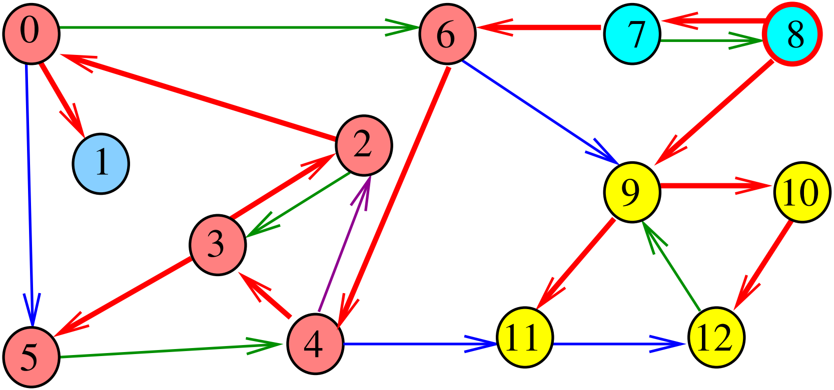
Tarjan realiza apenas um passo DFS sobre o digrafo.

Kosaraju e Sharir fazem duas passadas DFS.

Discutiremos o algoritmo de Kosaraju e Sharir.

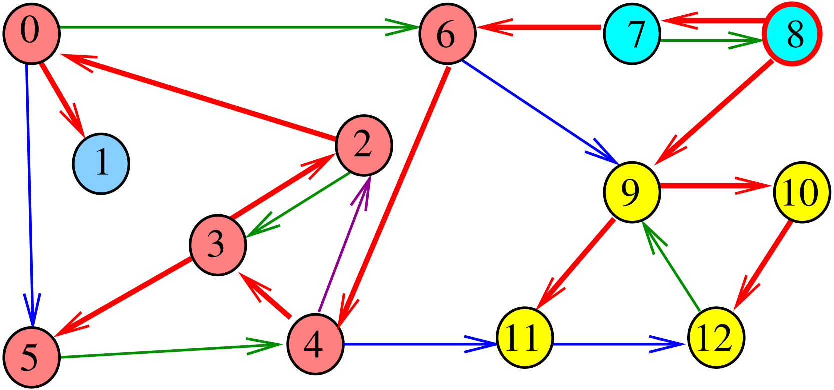
Propriedade

Vértices de um componente fortemente conexo são uma subarborescência em uma floresta DF.



Propriedade

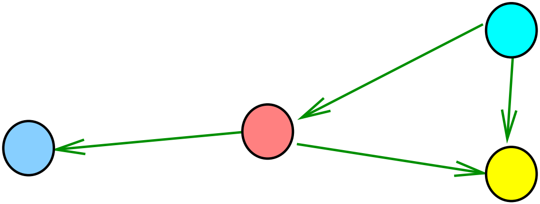
Vértices de um componente fortemente conexo são uma subarborescência em uma floresta DF.



Ao entrar num componente f.c., examinaremos todos os seus vértices na mesma chamada de DFS-Visit.

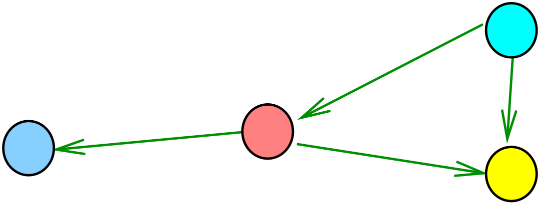
Digrafos dos componentes

O digrafo dos componentes de G tem um vértice para cada componente fortemente conexo e um arco U-W se G possui um arco com ponta inicial em U e ponta final em W.



Digrafos dos componentes

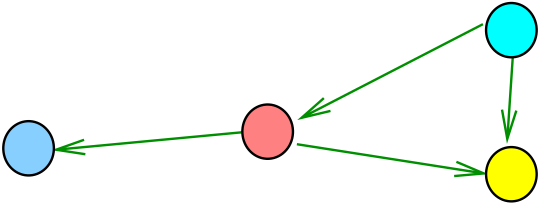
O digrafo dos componentes de G tem um vértice para cada componente fortemente conexo e um arco U-W se G possui um arco com ponta inicial em U e ponta final em W.



Digrafo dos componentes é um DAG!
(DAG: directed acyclic graph)

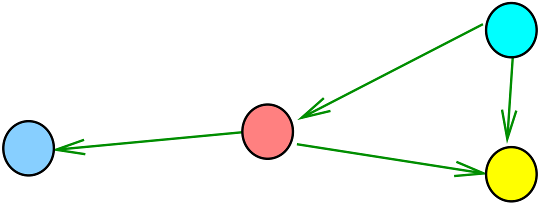
Ideia ...

Visitar as componentes
numa ordem topológica do digrafo das componentes...



Ideia ...

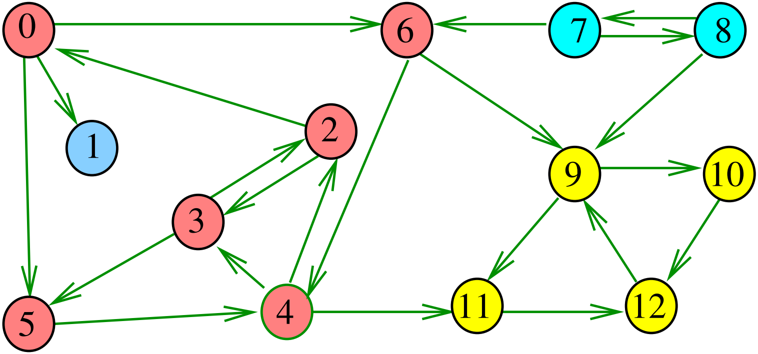
Visitar as componentes
numa ordem topológica do digrafo das componentes...



Primeiro a componente amarela e a azul (ou vice-versa),
depois a ciam, e por último a vermelha.

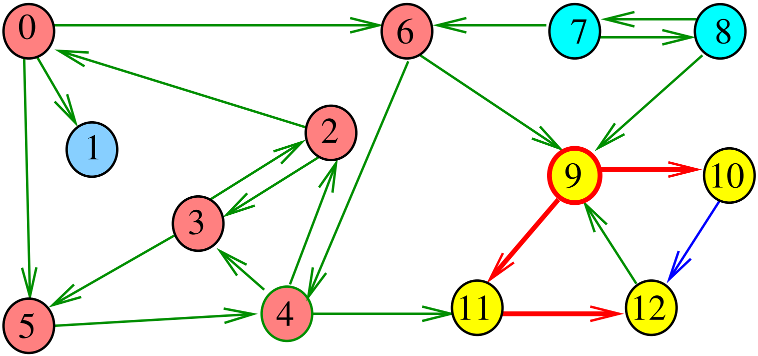
Ideia ... G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...



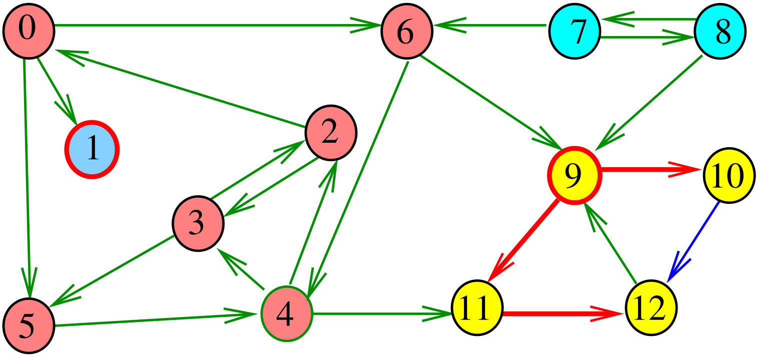
Ideia ... G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...



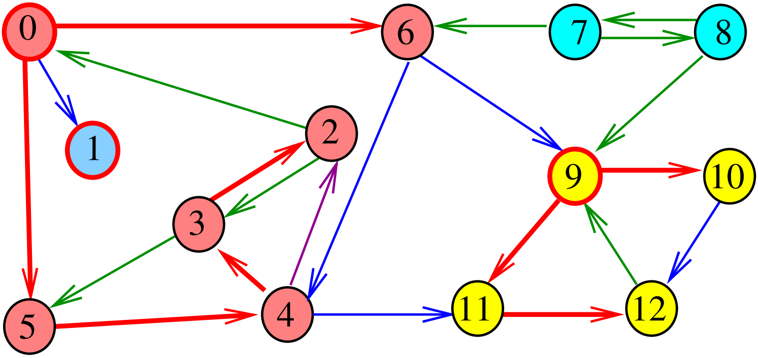
Ideia ... G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...



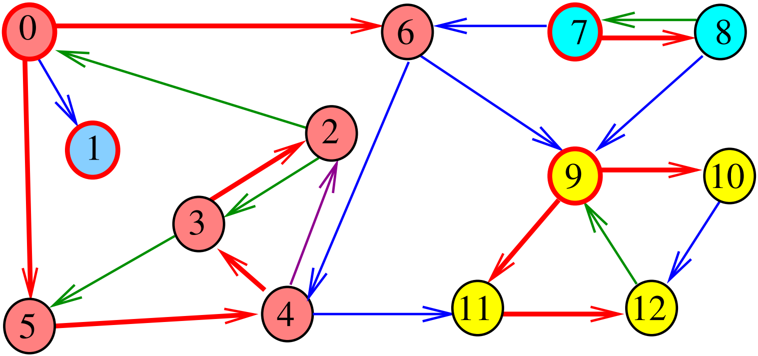
Ideia . . . G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...



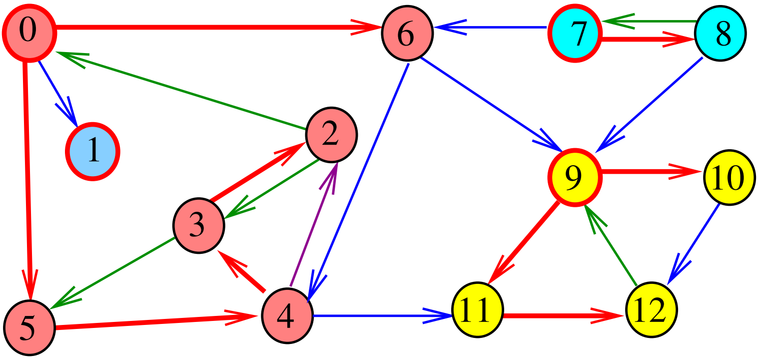
Ideia . . . G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...



Ideia . . . G e DFS

Visitar as componentes
numa ordem topológica do digrafo das componentes...

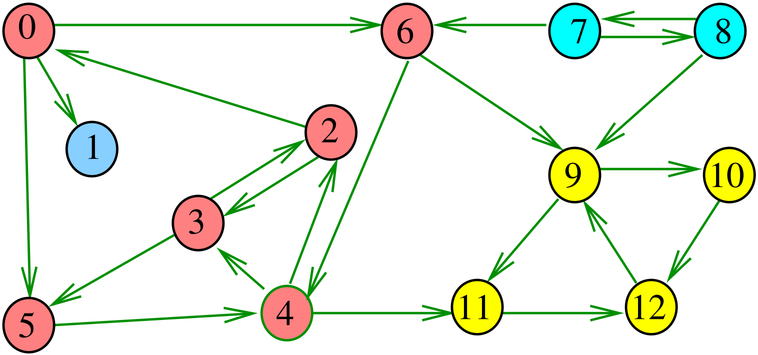


Cada chamada à DFS-Visit devolve
uma componente fortemente conexa.

Propriedade

Um digrafo G e seu digrafo reverso R têm as mesmas componente fortemente conexas.

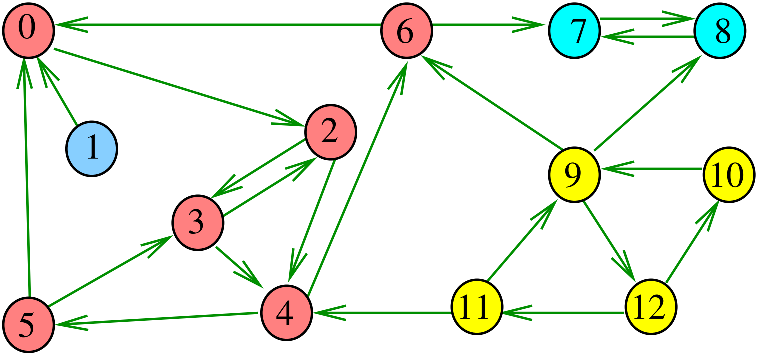
Exemplo: Digrafo G



Propriedade

Um digrafo G e seu digrafo reverso R têm as mesmas componente fortemente conexas.

Exemplo: Digrafo reverso R de G



DFS e componentes fortemente conexas

Considere o vetor f obtido de uma DFS no G.

DFS e componentes fortemente conexas

Considere o vetor f obtido de uma DFS no G.

Fato. Se $f[v] > f[w]$ e existe um caminho de w a v, então existe um caminho de v a w.

DFS e componentes fortemente conexas

Considere o vetor f obtido de uma DFS no G.

Fato. Se $f[v] > f[w]$ e existe um caminho de w a v,
então existe um caminho de v a w.

Em outras palavras:

Fato. Se existe um caminho de w a v e $f[v] > f[w]$,
então v e w estão em um mesmo componente fortemente conexo.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

    Execute uma DFS em G calculando $u.f$ para cada u.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Execute uma DFS em G calculando $u.f$ para cada u.

Construa G^r.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Execute uma DFS em G calculando $u.f$ para cada u.

Construa G^r.

Execute uma DFS em G^r considerando os vértices de G^r em ordem decrescente do valor de f calculado acima.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

> Execute uma DFS em G calculando $u.f$ para cada u.
>
> Construa G^r.
>
> Execute uma DFS em G^r considerando os vértices de G^r em ordem decrescente do valor de f calculado acima.
>
> Devolva as componentes da floresta DF construída para G^r.

Segunda versão (para a simulação)

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

  Construa G^r.

Segunda versão (para a simulação)

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Construa G^r.

Execute uma DFS em G^r calculando $u.f$ para cada u.

Segunda versão (para a simulação)

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Construa G^r.

Execute uma DFS em G^r calculando $u.f$ para cada u.

Execute uma DFS em G considerando os vértices de G em ordem decrescente do valor de f calculado acima.

Segunda versão (para a simulação)

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

Construa G^r.

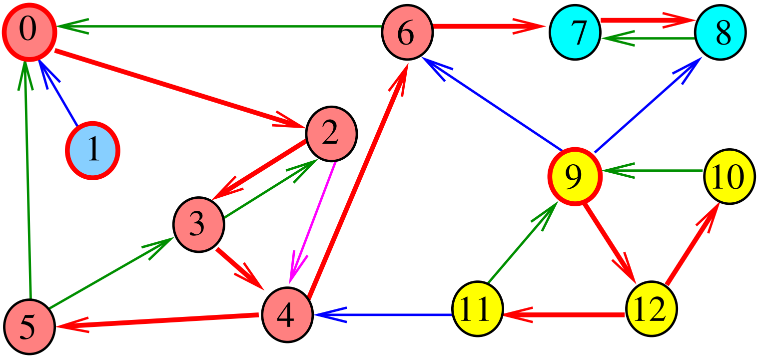
Execute uma DFS em G^r calculando $u.f$ para cada u.

Execute uma DFS em G considerando os vértices de G em ordem decrescente do valor de f calculado acima.

Devolva as componentes da floresta DF construída para G.

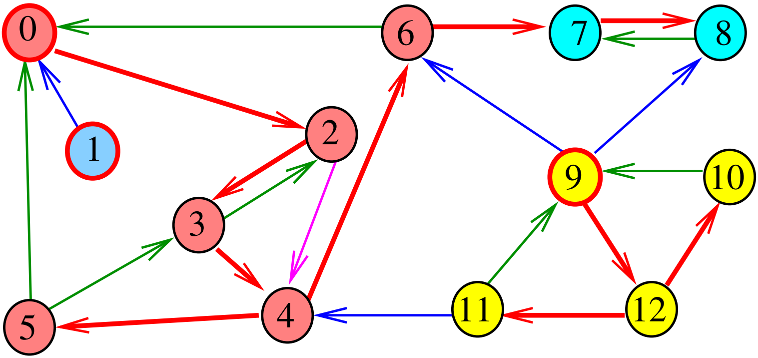
Digrafo reverso R e DFS

v	0	1	2	3	4	5	6	7	8	9	10	11	12
$f[v]$	7	8	6	5	4	3	2	1	0	12	9	10	11



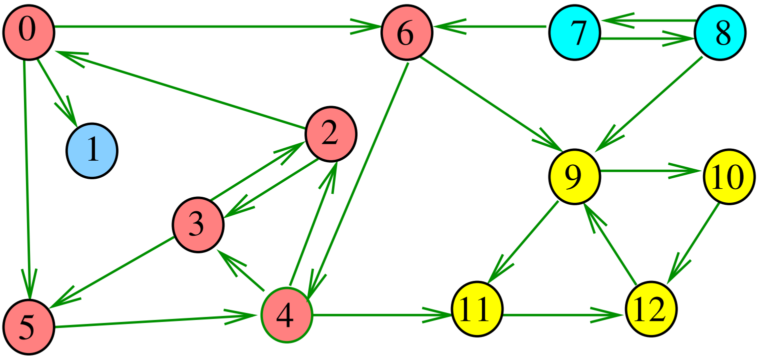
Digrafo reverso R e DFS

v	0	1	2	3	4	5	6	7	8	9	10	11	12
$f[v]$	7	8	6	5	4	3	2	1	0	12	9	10	11
i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	8	7	6	5	4	3	2	0	1	10	11	12	9



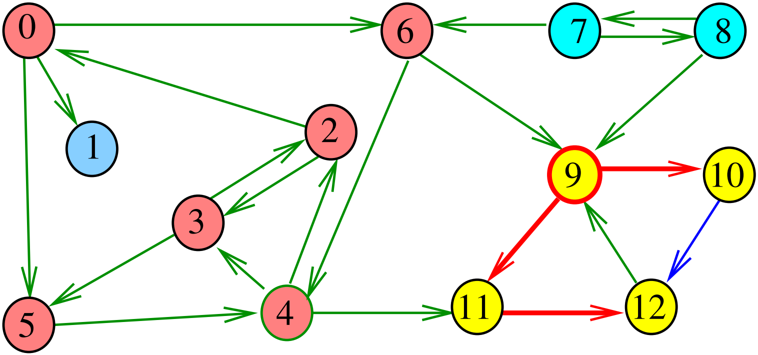
Digrafo G e DFS

i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	8	7	6	5	4	3	2	0	1	10	11	12	9



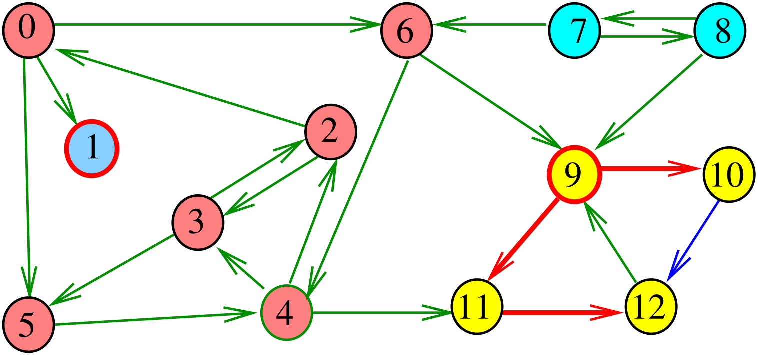
Digrafo G e DFS

i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	8	7	6	5	4	3	2	0	1	**10**	**11**	**12**	**9**



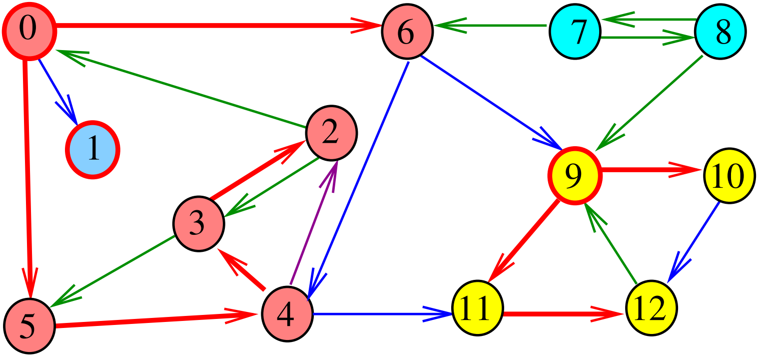
Digrafo G e DFS

i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	8	7	6	5	4	3	2	0	**1**	**10**	**11**	**12**	**9**



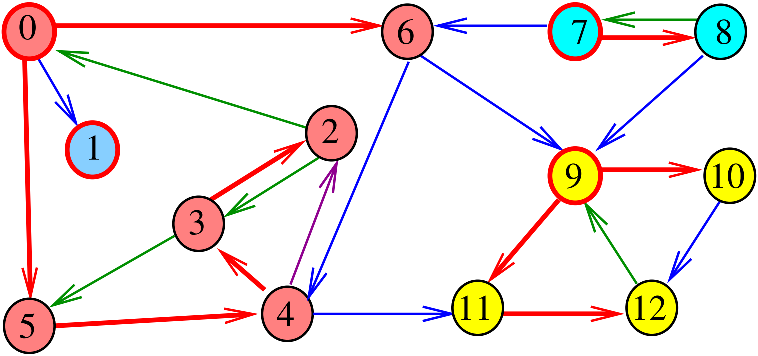
Digrafo G e DFS

i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	8	7	**6**	**5**	**4**	**3**	**2**	**0**	**1**	**10**	**11**	**12**	**9**



Digrafo G e DFS

i	0	1	2	3	4	5	6	7	8	9	10	11	12
$f^{-1}[i]$	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**0**	**1**	**10**	**11**	**12**	**9**



Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

- Execute uma DFS em G calculando $u.f$ para cada u.
- Construa G^r.
- Execute uma DFS em G^r considerando os vértices de G^r em ordem decrescente do valor de f calculado acima.
- Devolva as componentes da floresta DF construída para G^r.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

- Execute uma DFS em G calculando $u.f$ para cada u.
- Construa G^r.
- Execute uma DFS em G^r considerando os vértices de G^r em ordem decrescente do valor de f calculado acima.
- Devolva as componentes da floresta DF construída para G^r.

Consumo de tempo: linear no tamanho de G.

Algoritmo de Kosaraju-Sharir

Seja $G = (V, E)$ um digrafo.

Seja G^r o digrafo reverso de G (todos os arcos invertidos).

Kosaraju-Sharir(G)

- Execute uma DFS em G calculando $u.f$ para cada u.
- Construa G^r.
- Execute uma DFS em G^r considerando os vértices de G^r em ordem decrescente do valor de f calculado acima.
- Devolva as componentes da floresta DF construída para G^r.

Consumo de tempo: linear no tamanho de G.

Teorema: Kosaraju-Sharir(G) calcula corretamente as componentes fortemente conexas de G.