Aula 6 - Aplicações de DFS (parte II)

Luís Felipe

UFF

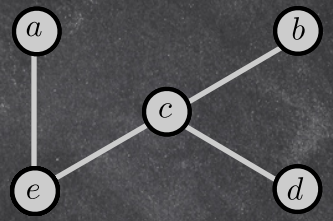
26 de Março de 2026

Luís Felipe

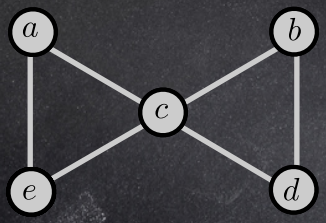
26/03/26

Articulação

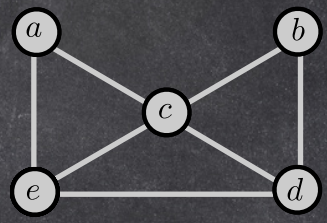
Um vértice v de um grafo G é de articulação se, e somente se, $w(G \setminus v) > w(G)$. Isto é, o número de componentes conexas de $G - v$ é maior do que o número de componentes conexas de G.



- Os vértices c e e são de articulação.
- Os vértices a, b e d não são de articulação.



c é articulação



Não existe articulação

Luís Felipe

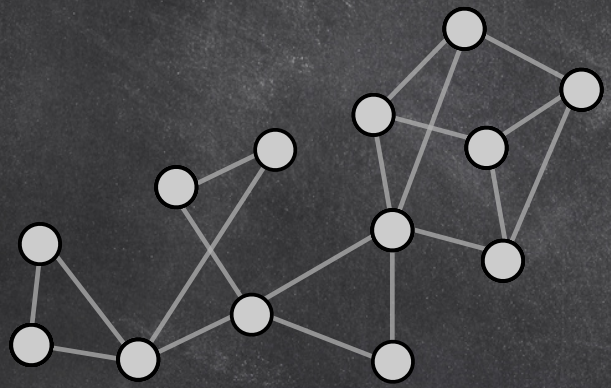
26/03/26

Blocos

Um bloco é um grafo conexo que não possui articulação.

Um bloco de um grafo G é um subgrafo de G maximal com relação à propriedade de ser bloco.

Exemplo:



4 blocos

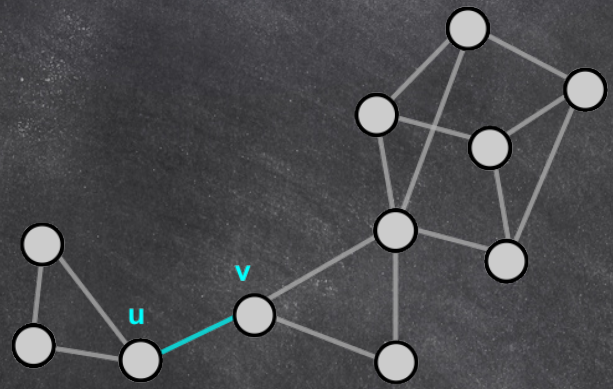
- Fato: Dois blocos diferentes em um grafo têm no máximo um vértice em comum.
- Fato: Cada aresta de um grafo G está em um único bloco. Portanto, os blocos formam uma partição de $E(G)$.

Luís Felipe

26/03/26

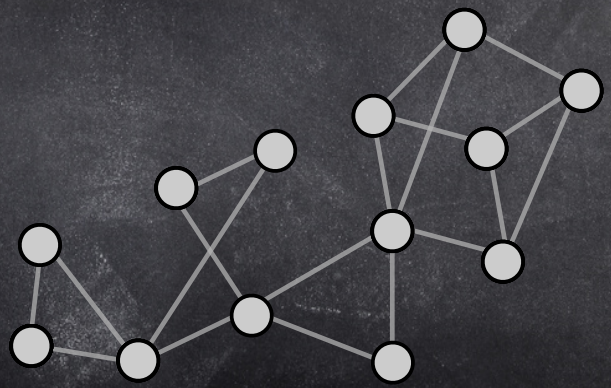
Blocos

Se um bloco de um grafo é uma única aresta, então essa aresta é chamada de ponte.



uv é uma ponte

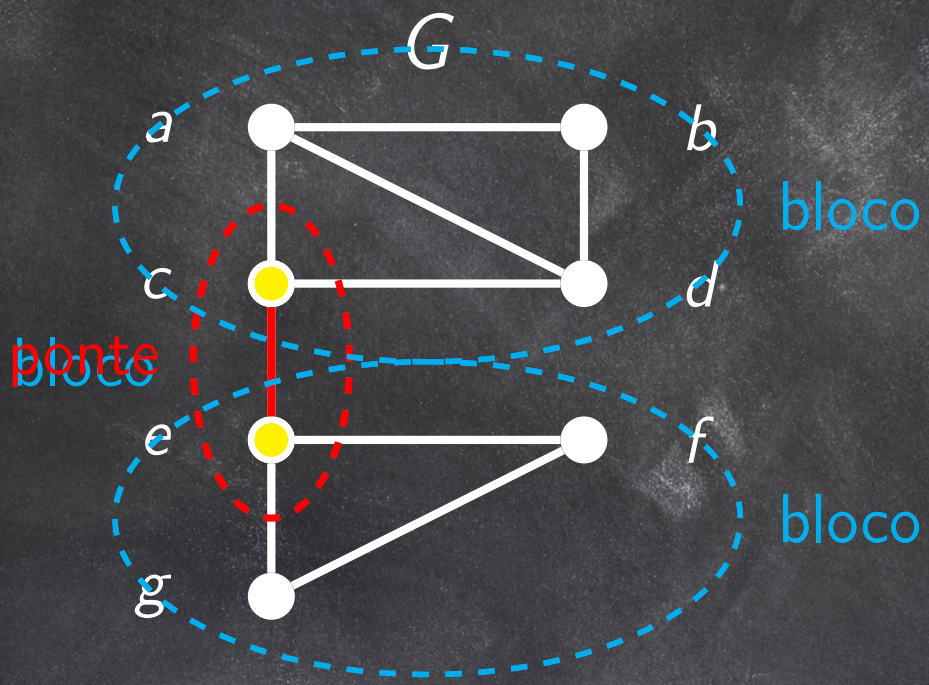
Em todo bloco que não seja uma ponte, existem dois caminhos internamente disjuntos entre qualquer par de vértices. Neste caso, o bloco é um subgrafo maximal biconexo.



Luís Felipe

26/03/26

Aplicação 6: Determinar (e encontrar) articulações e blocos



A remoção de uma ponte desconecta o grafo!

Luís Felipe

26/03/26

Sobre blocos e arestas de retorno

- Como já visto, os blocos de um grafo G determinam naturalmente uma partição do conjunto de arestas de G, isto é, cada aresta pertence a um e apenas um bloco de G.

- O mesmo não ocorre em relação aos vértices:
  - Se v pertence a mais de um bloco então v é uma articulação.
  - se v pertence a um único bloco então v não é uma articulação.

$back(v)$:

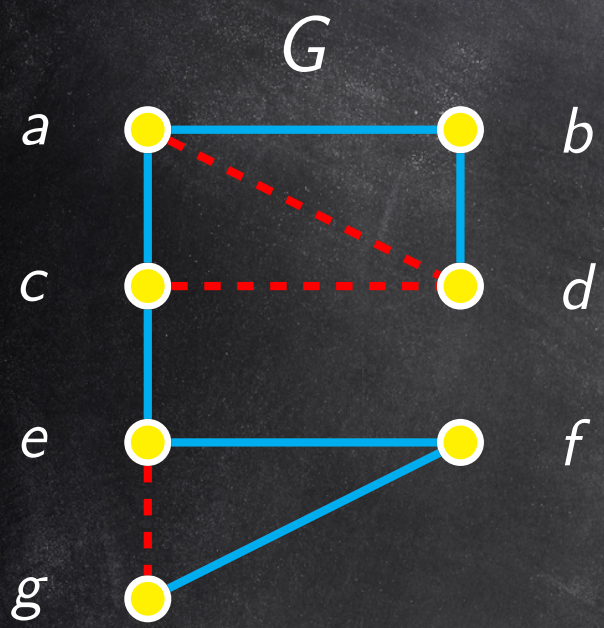
- Seja T uma árvore de profundidade do grafo G.
  $back(v)$ é o menor valor de $PE(w)$ tal que o vértice w pode ser alcançado a partir de v da seguinte forma:
  - A partir de v, usa-se 0 ou mais arestas de T seguindo o sentido para baixo na árvore;
  - em seguida, usa-se no máximo uma aresta de retorno para alcançar w.

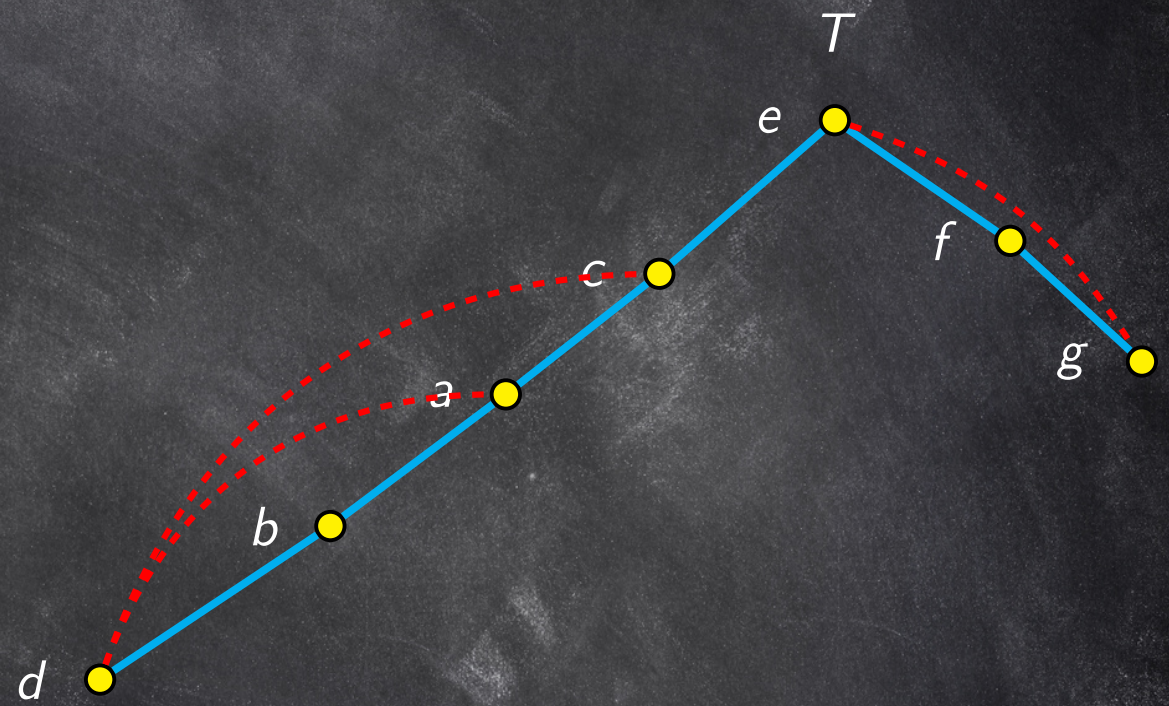
Obs.: $back(v)$ mede o quão perto da raiz o subgrafo de v consegue "voltar" usando uma aresta de retorno.

Luís Felipe

26/03/26

Exemplo





v	a	b	c	d	e	f	g
PE(v)	3	4	2	5	1	10	11
PS(v)	8	7	9	6	14	13	12
back(v)	2	2	2	2	1	1	1

Luís Felipe

26/03/26

Como calcular $back(v)$

Recorrência para $back(v)$

$$back(v) = \min \begin{cases} PE(v), \\ \min\{\, back(w) \mid w \text{ é filho de } v \text{ em } T \,\}, \\ \min\{\, PE(w) \mid vw \text{ é aresta de retorno} \,\} \end{cases}$$

Obs.: $back(v)$ é o menor tempo de entrada que pode ser alcançado a partir do subgrafo enraizado em v usando arestas da árvore DFS e no máximo uma aresta de retorno.

Luís Felipe

26/03/26

DFS com cálculo de $back(v)$

```python
def dfs(v, adj, pe, ps, back, pai, t):
    t[0] += 1
    pe[v] = t[0]

    back[v] = pe[v]

    for w in adj[v]:

        if pe[w] == 0:                      # w ainda não foi visitado pela DFS.

            pai[w] = v                      # vw se torna uma aresta da DFS

            dfs(w, adj, pe, ps, back, pai, t)

            back[v] = min(back[v], back[w])

        elif ps[w] == 0 and w != pai[v]:    # w já foi descoberto; w não terminou; w não é pai de v

            back[v] = min(back[v], pe[w])

    t[0] += 1
    ps[v] = t[0]
```

Luís Felipe

26/03/26

Detectando vértices de articulação

Teorema Seja T a árvore DFS.

- Se v é a raiz da DFS:

  v é articulação sse possui ≥ 2 filhos em T

- Se v não é raiz da DFS:

  v é articulação sse existe filho w tal que

  $$back(w) \geq PE(v)$$

Obs.:

$back(w) = PE(v)$ significa que a subárvore de w consegue voltar exatamente para v, mas não para cima de v. Então, ao remover v, ela também se desconecta.

$back(w) > PE(v)$ significa que a subárvore de w não consegue voltar nem para v; ela só alcança vértices com PE maior que o de v, isto é, vértices mais abaixo na DFS. Então, a remoção de v também desconecta essa parte.

Prova (caso da raiz)

Suponha que v seja a raiz da DFS.

($\Rightarrow$) Suponha que v seja articulação.
Então a remoção de v aumenta o número de componentes conexas de G.
Logo existem pelo menos dois subgrafos distintos ligados a v que ficam desconectados após remover v.
Na árvore DFS, cada um desses subgrafos corresponde a uma subárvore distinta de v. Portanto v possui pelo menos dois filhos em T.

($\Leftarrow$) Suponha agora que v possui dois filhos w_1 e w_2 em T. Como não existe aresta de retorno entre nós que não sejam ancestrais em T, então não existem arestas entre essas subárvores. Assim, ao remover v, essas subárvores ficam desconectadas, e portanto v é articulação.

Luís Felipe

26/03/26

Prova (vértice não raiz)

Suponha que v não seja a raiz da DFS.

(⇒) Suponha que v seja articulação.
Então existe um filho w de v cuja subárvore fica desconectada do restante do grafo quando v é removido.
Isso significa que nenhum vértice da subárvore de w possui aresta de retorno para um ancestral de v. Caso contrário, existiria um caminho que conectaria essa subárvore ao restante do grafo sem passar por v.
Logo o menor tempo de descoberta alcançável a partir da subárvore de w satisfaz $back(w) \geq PE(v)$.

(⇐) Suponha agora que existe um filho w de v tal que $back(w) \geq PE(v)$.
Pela definição de $back(w)$, nenhum vértice da subárvore de w possui aresta de retorno para um ancestral de v.
Assim todo caminho da subárvore de w para vértices acima de v passa necessariamente por v. Portanto, ao remover v, essa subárvore fica desconectada do restante do grafo.
Logo v é um vértice de articulação.

Luís Felipe

26/03/26

Detecção de articulações

```
1   def dfs_articulacoes(v, adj, pe, ps, back, pai, t, art):
2       t[0] += 1
3       pe[v] = t[0]
4
5       back[v] = pe[v]
6       filhos = 0
7
8       for w in adj[v]:
9           if pe[w] == 0:
10
11              pai[w] = v
12              filhos += 1
13
14              dfs_articulacoes(w, adj, pe, ps, back, pai, t, art)
15
16              back[v] = min(back[v], back[w])
17
18              if pai[v] != -1 and back[w] >= pe[v]:   # o filho w e toda a subárvore abaixo não
19                                                     # conseguem voltar para nenhum ancestral de v
20                                                     # usando uma aresta de retorno.
21                  art[v] = True
22
23          elif ps[w] == 0 and w != pai[v]:           # w já foi descoberto, mas ainda não terminou
24                                                     # de ser processado
25
26              back[v] = min(back[v], pe[w])
27
28      if pai[v] == -1 and filhos >= 2:               # caso especial da raiz ser articulação
29          art[v] = True
30
31      t[0] += 1
32      ps[v] = t[0]
```

Luís Felipe

26/03/26

Encontrando blocos (componentes biconexas)

Problema: Obtenha blocos do grafo

Ideia do algoritmo:

- Durante a DFS mantemos uma pilha de arestas.
- Sempre que uma aresta é visitada ela é empilhada.
- Quando encontramos um vértice v tal que

$$back(w) \geq PE(v)$$

então todas as arestas da pilha até vw formam um bloco.

Luís Felipe

26/03/26

Demarcadores e blocos

Observação:

- Um filho w de v tal que

$$back(w) \geq PE(v)$$

  é chamado de demarcador de v.
- Essa condição indica que a subárvore enraizada em w não alcança ancestrais de v por arestas de retorno.
- Consequentemente, quando essa desigualdade é satisfeita, as arestas acumuladas na pilha desde vw formam um bloco.
- Assim, os demarcadores são exatamente os pontos da DFS em que um bloco é identificado e retirado da pilha.

Luís Felipe

26/03/26

Comportamento da pilha de arestas

Durante a execução da DFS para encontrar blocos:

- Cada vez que uma aresta de árvore ou de retorno é encontrada, ela é empilhada.
- A pilha armazena as arestas do bloco atualmente em construção.
- Quando, para um filho w de v, vale

$$back(w) \geq PE(v),$$

  conclui-se que a subárvore de w não alcança ancestrais de v.
- Nesse momento, todas as arestas da pilha desde vw até o topo são desempilhadas e formam um bloco.
- Assim, a pilha cresce enquanto o bloco está "aberto" e é esvaziada parcialmente quando um bloco é finalizado.

Luís Felipe

26/03/26

Cálculo dos blocos

```
1  def dfs_blocos(v, adj, pe, ps, back, pai, t, pilha):
2      t[0] += 1
3      pe[v] = t[0]
4      back[v] = pe[v]
5
6      for w in adj[v]:
7          if pe[w] == 0:
8
9              pilha.append((v, w))
10             pai[w] = v
11
12             dfs_blocos(w, adj, pe, ps, back, pai, t, pilha)
13
14             back[v] = min(back[v], back[w])
15
16             if back[w] >= pe[v]:
17
18                 bloco = []
19
20                 while True:
21                     e = pilha.pop()
22                     bloco.append(e)
23
24                     if e == (v, w):
25                         break
26
27                 print("Bloco:", bloco)
28
29         elif ps[w] == 0 and w != pai[v]:
30
31             pilha.append((v, w))
32
33             back[v] = min(back[v], pe[w])
34
35     t[0] += 1
36     ps[v] = t[0]
```

Luís Felipe

26/03/26

Demarcadores

Pergunta: Cada bloco possui seu próprio demarcador?

Resposta: Sim, cada bloco identificado pelo algoritmo está associado a uma aresta de árvore vw tal que

$$back(w) \geq PE(v).$$

- O vértice w é um demarcador de v.
- Quando essa condição é satisfeita, a aresta vw marca exatamente o ponto em que o algoritmo reconhece o fim de um bloco.
- As arestas removidas da pilha até vw constituem esse bloco.
- Portanto, cada bloco produzido pelo algoritmo possui um demarcador associado.
- Desse modo, o número de demarcadores coincide com o número de blocos encontrados pela DFS.

Luís Felipe

26/03/26

Raiz da DFS e articulações

Afirmação: Se a busca em profundidade se iniciar por um vértice que não é articulação, então a raiz da árvore DFS terá apenas um filho.

- A raiz de uma DFS é articulação se, e somente se, possui dois ou mais filhos na árvore de profundidade.
- Logo, se o vértice inicial não é articulação, ele não pode ter dois ou mais filhos na árvore DFS.
- Portanto, a raiz terá necessariamente apenas um filho.

Luís Felipe

26/03/26

Articulações, blocos e pontes

A mesma estrutura de DFS permite identificar diferentes propriedades do grafo:

- Articulação: um vértice v é articulação se existe um filho w tal que

$$back(w) \geq PE(v),$$

com tratamento especial para a raiz.

- Bloco: quando essa condição ocorre, as arestas da pilha até vw formam uma componente biconexa.

- Ponte: uma aresta vw é ponte quando

$$back(w) > PE(v).$$

Nesse caso, não existe aresta de retorno ligando a subárvore de w a v ou a algum ancestral de v.