Aula 10 - Busca em largura (BFS)

Luís Felipe

UFF

14 de Abril de 2026

Luís Felipe

14/04/26

Busca em largura (BFS)

- A Busca em Largura (*Breadth-First Search – BFS*) explora o grafo por níveis.
- A ideia é visitar primeiro:
  - todos os vértices vizinhos do vértice inicial;
  - depois os vizinhos desses vértices;
  - e assim sucessivamente.
- Para controlar a ordem da exploração usamos uma fila.
- A BFS constrói uma floresta de largura, onde:
  - cada vértice tem um pai;
  - cada vértice tem um nível na busca.

Luís Felipe

14/04/26

Busca em largura (BFS)

```
1  t <- 0                          # relogio (tempo global da busca)
2  F <- {}                         # fila auxiliar da BFS
3
4  para todo vertice v em V(G) faca
5      L(v) <- 0                   # indice de visita de v (0 = nao visitado)
6      pai(v) <- null              # pai de v na floresta de largura
7
8  enquanto existe v em V(G) tal que L(v) = 0 faca
9      nivel(v) <- 0               # v será raiz de uma nova arvore da BFS
10     t <- t + 1
11     L(v) <- t                   # registra ordem de visita
12     colocar v na fila F         # inicia exploracao a partir de v
13
14     enquanto F != {} faca
15         v <- primeiro elemento de F
16         retirar v de F          # remove o vertice atual da fila
17
18         para todo vertice w em N(v) faca
19             se L(w) = 0 entao
20                 visitar aresta vw          # w esta sendo descoberto
21                 pai(w) <- v                # v torna-se pai de w
22                 nivel(w) <- nivel(v) + 1   # w fica no proximo nivel
23                 t <- t + 1
24                 L(w) <- t                  # registra ordem de visita
25                 colocar w no final da fila F
26
27             senao se nivel(w) = nivel(v) entao
28                 se pai(w) = pai(v) entao
29                     vw eh aresta "irmao"   # vertices com mesmo pai
30                 senao
31                     vw eh aresta "primo"   # mesmo nivel, pais distintos
32
33             senao se nivel(w) = nivel(v) + 1 entao
34                 vw eh aresta "tio"         # conecta niveis consecutivos
```

Luís Felipe

14/04/26

Busca em largura (BFS) em Python

```
from collections import deque
t = 0                                   # relogio (tempo global da busca)
F = deque()                             # fila auxiliar da BFS
L = [0]*n                               # indice de visita (0 = nao visitado)
pai = [-1]*n                            # pai na floresta de largura
nivel = [-1]*n                          # nivel de cada vertice
tipo = dict()                           # classificacao das arestas

for a in range(n):                      # percorre todos os vertices (BFS completa)
    if L[a] == 0:                       # se a ainda nao foi visitado
        nivel[a] = 0                    # a sera raiz de uma nova arvore da BFS
        t += 1
        L[a] = t                        # registra ordem de visita
        F.append(a)                     # inicia exploracao a partir de a

        while F:                        # enquanto houver vertices na fila
            v = F.popleft()             # remove o primeiro elemento da fila

            for w in adj[v]:            # percorre todos os vizinhos de v
                e = (min(v,w), max(v,w))# representa {v,w} de forma unica (não duplica vw e wv)

                if L[w] == 0:           # w esta sendo descoberto
                    pai[w] = v          # v torna-se pai de w
                    nivel[w] = nivel[v] + 1
                    t += 1
                    L[w] = t            # registra ordem de visita
                    F.append(w)         # coloca w no final da fila
                    tipo[e] = "pai"

                elif nivel[w] == nivel[v]:
                    if pai[w] == pai[v]:
                        tipo[e] = "irmao"
                    else:
                        tipo[e] = "primo"

                elif nivel[w] == nivel[v] + 1:
                    tipo[e] = "tio"
```

Luís Felipe

14/04/26

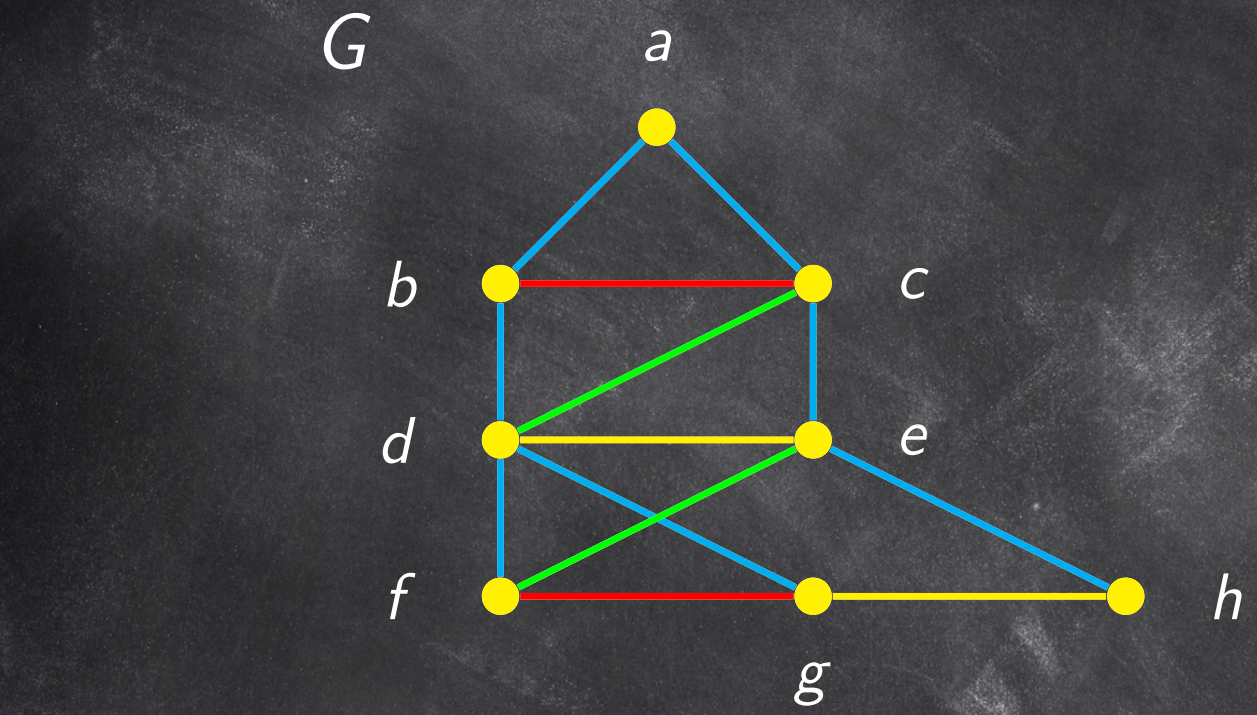
Classificação das arestas na BFS

- Aresta pai: descobre um novo vértice.
- Aresta irmão: liga vértices do mesmo nível com o mesmo pai.
- Aresta primo: liga vértices do mesmo nível com pais distintos.
- Aresta tio: liga um vértice a outro do nível seguinte já descoberto.

Luís Felipe

14/04/26

O que a BFS está fazendo



v	a	b	c	d	e	f	g	h
L(v)	1	2	3	4	5	6	7	8

$F = \emptyset$

$F = a$

Luís Felipe

14/04/26

Intuição da busca em largura

- A busca em largura segue a ideia de:

"Esgotar o pai antes de processar os filhos"

- Ou seja:
  - todos os vizinhos de um vértice são processados antes dos vértices do próximo nível;
  - a próxima aresta visitada sempre parte do vértice mais antigo na busca.
- Por isso a BFS utiliza uma fila.

Luís Felipe

14/04/26

Complexidade da busca em largura

- A complexidade da BFS é:

$$O(n + m)$$

  onde:
  - $n = |V(G)|$ é o número de vértices
  - $m = |E(G)|$ é o número de arestas

- Cada vértice é visitado no máximo uma vez.
- Cada aresta é examinada no máximo uma vez.

Luís Felipe

14/04/26

Floresta de largura

- A BFS constrói uma floresta de largura T.
- Essa floresta é uma floresta geradora de G:
  - todos os vértices alcançados pela busca pertencem a T;
  - ao final da busca, todos eles possuem $L(v) \neq 0$.
- Apenas as arestas-pai pertencem à floresta.
- As demais arestas são classificadas como:
  - arestas irmão
  - arestas primo
  - arestas tio

Luís Felipe

14/04/26

Caracterização das arestas-pai (arestas azuis)

Seja vw uma aresta de G.

vw é uma aresta-pai da floresta de largura

se e somente se

$$nivel(w) = nivel(v) + 1$$

e

$$L(w) = 0$$

no momento da visita.

- Nesse caso, w está sendo descoberto pela busca.

Luís Felipe

14/04/26

Caracterização das arestas-tio (arestas verdes)

Seja vw uma aresta de G.

vw é uma aresta-tio

se e somente se

$$nivel(w) = nivel(v) + 1$$

e

$$L(w) \neq 0$$

no momento da visita.

- Ou seja, w já havia sido descoberto anteriormente.

Luís Felipe

14/04/26

Caracterização das arestas-irmão (arestas vermelhas)

Seja vw uma aresta de G.

vw é uma aresta-irmão

se e somente se

$$nivel(w) = nivel(v)$$

e

$$pai(w) = pai(v)$$

- Ou seja, v e w têm o mesmo pai na floresta.

Luís Felipe

14/04/26

Caracterização das arestas-primo (arestas amarelas)

Seja vw uma aresta de G.

vw é uma aresta-primo

se e somente se

$$nivel(w) = nivel(v)$$

e

$$pai(w) \neq pai(v)$$

- Ou seja, v e w estão no mesmo nível, mas possuem pais diferentes.

Luís Felipe

14/04/26

Propriedade fundamental da BFS

Seja vw uma aresta de G e T uma floresta de largura.

$$|nivel(v) - nivel(w)| \leq 1$$

- Ou seja, vértices ligados por uma aresta diferem em no máximo um nível.
- Em particular:
  - vértices na fila da BFS diferem em no máximo um nível.

Luís Felipe

14/04/26

Por que os níveis diferem no máximo 1?

Considere uma aresta vw de um grafo G.
Suponha que durante a BFS:

$$nivel(v) = k$$

Quando v é processado:

- todos os seus vizinhos são examinados;
- se w ainda não foi visitado, então:

$$nivel(w) = k + 1$$

Se w já havia sido visitado, então necessariamente:

$$nivel(w) \in \{k - 1,\ k,\ k + 1\}$$

Observe que ao visitar v, visitamos seus vizinhos que:

- Já estavam na árvore. Nesse caso, alguém que está na fila mas não na sua vez e ainda será visitado, ou seja, no nível $k - 1$ ou k. Dado que a gente explora todo um nível para depois "descer"), ou;
- Ainda não estão na árvore. Nesse caso, alguém que vai para o nível seguinte, $k + 1$.

Luís Felipe

14/04/26

Aplicação: cálculo de distâncias

Problema:

- Dado um grafo conexo G e um vértice x, determinar a distância de x a todos os vértices.

Solução:

- Executar BFS em G com raiz x.
- Ao final da busca:

$$dist(x, v) = nivel(v)$$

- Um caminho mínimo de x a v é o caminho de x a v na árvore de largura.

Luís Felipe

14/04/26

Aplicação: menor caminho em labirinto

Problema:

- Encontrar o menor caminho da entrada até a saída de um labirinto.

Modelagem:

- representar o labirinto como um grafo;
- entrada e saída são vértices x e y.

Solução:

- executar BFS a partir de x;
- se $L(y) \neq 0$, então y é alcançável;
- o caminho em T de x até y é o caminho mínimo, $dist(x, y) = nivel(y)$.

Luís Felipe

14/04/26

Aplicação: grafos bipartidos

Problema:

- decidir se um grafo G é bipartido.

Solução:

- executar uma BFS em G;
- G é bipartido se e somente se a floresta não contém:
  - arestas-irmão
  - arestas-primo
- essas arestas fecham ciclos ímpares.

Para definir a 2-coloração do grafo bipartido:

- níveis pares $\rightarrow$ uma cor
- níveis ímpares $\rightarrow$ outra cor

Luís Felipe

14/04/26

Exercício

Dado:

- um grafo conexo G
- uma árvore geradora T de G

Elaborar um algoritmo para decidir se T pode ser produzida por uma busca em largura.
Ou seja:

existe uma BFS em G cuja árvore de largura é T?

Solução: Para cada vértice r de T, enraíze T em r e calcule o nível de cada vértice. Em seguida, verifique se toda aresta $vw \in E(G) \setminus E(T)$ satisfaz $|nivel(v) - nivel(w)| \leq 1$. Se isso ocorrer para alguma escolha de raiz r, então T pode ser produzida por uma busca em largura; caso contrário, não.
Uma vez que a árvore foi enraizada, arestas ligam pai e filho, e portanto têm diferença de nível igual a 1. Assim, basta testar as arestas fora de T.
Complexidade: $O(n(n+m))$. Cada uma das n escolhas de raiz calculamos os níveis em $O(n)$ e testamos as arestas em $O(m)$.

Luís Felipe

14/04/26

Exercício

Elaborar uma adaptação da busca em largura para aplicá-la a digrafos.

Dica:

- considerar apenas as arestas orientadas no sentido da exploração.

Luís Felipe

14/04/26

Exercício

```
1  desmarcar todos os vertices
2
3  escolher uma raiz v
4  pai(v) <- null
5  marcar v
6  inserir v em D
7
8  enquanto D != {} faca
9      v <- primeiro elemento de D
10
11     se existe w em N(v) desmarcado entao
12         visitar aresta vw
13         pai(w) <- v
14         marcar w
15         inserir w em D
16     senao
17         remover v de D
```

Mostre que:

- se D é uma pilha $\rightarrow$ DFS
- se D é uma fila $\rightarrow$ BFS