Análise de Algoritmos

Parte destes slides são adaptações de slides

do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

BFS e DFS

CLRS 22 Elementary Graph Algorithms

CLRS 22.2 e 22.3

Busca em largura

BFS (G, s)

1 **para cada** $u \in V(G) \setminus \{s\}$ **faça**
2 $\quad \text{color}[u] \leftarrow \text{branco} \quad d[u] \leftarrow \infty \quad \pi[u] \leftarrow \text{nil}$
3 $Q \leftarrow \emptyset$ $\quad \triangleright$ fila dos vértices descobertos
4 $\text{color}[s] \leftarrow \text{cinzento} \quad d[s] \leftarrow 0 \quad \pi[s] \leftarrow \text{nil}$
5 INSIRA(Q, s)

Busca em largura

```
BFS (G, s)
 1  para cada u ∈ V(G) \ {s} faça
 2      color[u] ← branco    d[u] ← ∞    π[u] ← nil
 3  Q ← ∅          ▷ fila dos vértices descobertos
 4  color[s] ← cinzento    d[s] ← 0    π[s] ← nil
 5  INSIRA(Q, s)

 6  enquanto Q ≠ ∅ faça
 7      u ← REMOVA(Q)
 8      para cada v ∈ adj[u] faça
 9          se color[v] = branco
10              então color[v] ← cinzento
11                    d[v] ← d[u] + 1
12                    π[v] ← u
13                    INSIRA(Q, v)
14      color[u] ← preto
15  devolva d, π
```

Correção

Seja $\delta(s,u)$ a distância de s a u em G.

Lema: $\delta(s,v) \leq \delta(s,u)+1$ para toda aresta uv.

Lema: $d[v] \geq \delta(s,v)$ para todo vértice ao final da BFS.

Lema: Se Q contém os vértices $v_1, v_2, \ldots, v_r$ nesta ordem, então $d[v_r] \leq d[v_1]+1$ e $d[v_i] \leq d[v_{i+1}]$ para $i = 1, \ldots, r-1$.

Corolário: Se v_i entra na fila antes de v_j, então $d[v_i] \leq d[v_j]$ quando v_j entra na fila.

Correção

Seja $\delta(s,u)$ a distância de s a u em G.

Lema: $\delta(s,v) \leq \delta(s,u)+1$ para toda aresta uv.

Lema: $d[v] \geq \delta(s,v)$ para todo vértice ao final da BFS.

Lema: Se Q contém os vértices $v_1, v_2, \ldots, v_r$ nesta ordem, então $d[v_r] \leq d[v_1]+1$ e $d[v_i] \leq d[v_{i+1}]$ para $i = 1, \ldots, r-1$.

Corolário: Se v_i entra na fila antes de v_j, então $d[v_i] \leq d[v_j]$ quando v_j entra na fila.

Teorema: Ao final da BFS, $d[v] = \delta(s,v)$ para todo $v \in V$. Ademais, para todo v alcançável de s com $v \neq s$, um caminho mais curto de s a v consiste em um caminho mais curto de s a $\pi[v]$ seguido da aresta $\pi[v]v$.

Busca em profundidade

```
DFS (G)
 1  para cada u ∈ V(G) faça
 2      color[u] ← branco        π[u] ← nil
 3  tempo ← 0
 4  para cada u ∈ V(G) faça
 5      se color[u] = branco
 6          então DFS-Visit(u)
```

Busca em profundidade

DFS (G)

```
 1  para cada u ∈ V(G) faça
 2      color[u] ← branco        π[u] ← nil
 3  tempo ← 0
 4  para cada u ∈ V(G) faça
 5      se color[u] = branco
 6          então DFS-Visit(u)
```

DFS-Visit(u)

```
 1  color[u] ← cinzento    d[u] ← tempo ← tempo + 1
 3  para cada v ∈ adj(u) faça
 4      se color[v] = branco
 5          então π[v] ← u
 6                DFS-Visit(v)
 7  color[u] ← preto
 8  f(u) ← tempo           tempo ← tempo + 1
```

Descrição

Vértice branco: ainda não descoberto

Vértice cinzento: descoberto mas não processado

Vértice preto: processado

Descrição

Vértice branco: ainda não descoberto

Vértice cinzento: descoberto mas não processado

Vértice preto: processado

DFS devolve em π uma DF floresta.

$\pi[u]$: predecessor ou pai de u na DF floresta

Descrição

Vértice branco: ainda não descoberto

Vértice cinzento: descoberto mas não processado

Vértice preto: processado

DFS devolve em π uma DF floresta.

$\pi[u]$: predecessor ou pai de u na DF floresta

Faz duas marcas de tempo:

$d[u]$: momento da descoberta de u
$f[u]$: momento do término de u

Descrição

Vértice branco: ainda não descoberto

Vértice cinzento: descoberto mas não processado

Vértice preto: processado

DFS devolve em π uma DF floresta.

$\pi[u]$: predecessor ou pai de u na DF floresta

Faz duas marcas de tempo:

$d[u]$: momento da descoberta de u
$f[u]$: momento do término de u

u é branco antes de $d[u]$,
cinzento entre u entre $d[u]$ e $f[u]$,
preto depois de $f[u]$.

Consumo de tempo

Cada vértice é descoberto uma única vez, pois é branco e, ao ser descoberto, passa a ser cinzento, e depois preto.

Consumo de tempo

Cada vértice é descoberto uma única vez, pois é branco e, ao ser descoberto, passa a ser cinzento, e depois preto.

A lista de adjacência de cada vértice descoberto é percorrida uma única vez.

Consumo de tempo

Cada vértice é descoberto uma única vez, pois é branco e, ao ser descoberto, passa a ser cinzento, e depois preto.

A lista de adjacência de cada vértice descoberto é percorrida uma única vez.

Logo o consumo de tempo é $\mathrm{O}(n+m)$, onde $n = |V|$ e $m = |E|$, pois a inicialização custa $\Theta(n)$ e a soma do tamanho das listas de adjacências percorridas é $\mathrm{O}(m)$.

Consumo de tempo

Cada vértice é descoberto uma única vez, pois é branco e, ao ser descoberto, passa a ser cinzento, e depois preto.

A lista de adjacência de cada vértice descoberto é percorrida uma única vez.

Logo o consumo de tempo é $\mathrm{O}(n+m)$, onde $n = |V|$ e $m = |E|$, pois a inicialização custa $\Theta(n)$ e a soma do tamanho das listas de adjacências percorridas é $\mathrm{O}(m)$.

O consumo de tempo de uma DFS é linear no tamanho do grafo.

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

b) O intervalo $[d[u], f[u]]$ está contido no intervalo $[d[v], f[v]]$, e u é descendente de v.

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

b) O intervalo $[d[u], f[u]]$ está contido no intervalo $[d[v], f[v]]$, e u é descendente de v.

c) O intervalo $[d[v], f[v]]$ está contido no intervalo $[d[u], f[u]]$, e v é descendente de u.

Propriedades da DFS de um grafo

Mesma estrutura que sequência válida de parêntesis.

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

b) O intervalo $[d[u], f[u]]$ está contido no intervalo $[d[v], f[v]]$, e u é descendente de v.

c) O intervalo $[d[v], f[v]]$ está contido no intervalo $[d[u], f[u]]$, e v é descendente de u.

Corolário: Vértice v é um descendente próprio do vértice u na floresta resultante da DFS sse $d[u] < d[v] < f[v] < f[u]$.

Propriedades da DFS de um grafo

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

b) O intervalo $[d[u], f[u]]$ está contido no intervalo $[d[v], f[v]]$, e u é descendente de v.

c) O intervalo $[d[v], f[v]]$ está contido no intervalo $[d[u], f[u]]$, e v é descendente de u.

Corolário: Vértice v é um descendente próprio do vértice u na floresta resultante da DFS sse $d[u] < d[v] < f[v] < f[u]$.

Propriedades da DFS de um grafo

Teorema: Para quaisquer vértices u e v, exatamente uma das três condições abaixo valem na floresta resultante:

a) Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos, e nem u é descendente de v, nem v é descendente de u.

b) O intervalo $[d[u], f[u]]$ está contido no intervalo $[d[v], f[v]]$, e u é descendente de v.

c) O intervalo $[d[v], f[v]]$ está contido no intervalo $[d[u], f[u]]$, e v é descendente de u.

Corolário: Vértice v é um descendente próprio do vértice u na floresta resultante da DFS sse $d[u] < d[v] < f[v] < f[u]$.

Teorema (do caminho branco): v é descendente de u sse, no momento $d[u]$ em que u é descoberto, v pode ser alcançado de u por caminho com apenas vértices brancos.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para
um ascendente na DF floresta.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para um ascendente na DF floresta.

c) Arestas para frente: de um vértice para um descendente na DF floresta.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para um ascendente na DF floresta.

c) Arestas para frente: de um vértice para um descendente na DF floresta.

d) Arestas cruzadas: todas as outras arestas.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para um ascendente na DF floresta.

c) Arestas para frente: de um vértice para um descendente na DF floresta.

d) Arestas cruzadas: todas as outras arestas.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para um ascendente na DF floresta.

c) Arestas para frente: de um vértice para um descendente na DF floresta.

d) Arestas cruzadas: todas as outras arestas.

Teorema: Se o grafo não é dirigido,
então só há arestas da árvore e arestas de retorno.

Classificação das arestas

Quatro tipos de arestas derivadas de uma DFS:

a) Árestas da árvore: arestas da DF floresta.

b) Árestas de retorno: de um vértice para um ascendente na DF floresta.

c) Arestas para frente: de um vértice para um descendente na DF floresta.

d) Arestas cruzadas: todas as outras arestas.

Teorema: Se o grafo não é dirigido,
então só há arestas da árvore e arestas de retorno.

Aplicação: ordenação topológica.

Ordenação topológica

Digrafo $G = (V, A)$

Ordenação topológica: Permutação de V
em que u vem antes de v sempre que $(u, v) \in A$.

Ordenação topológica

Digrafo $G = (V, A)$

Ordenação topológica: Permutação de V
em que u vem antes de v sempre que $(u, v) \in A$.

Digrafo G é acíclico se não tem circuito dirigido.

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Ordenação topológica

Digrafo $G = (V, A)$

Ordenação topológica: Permutação de V
em que u vem antes de v sempre que $(u, v) \in A$.

Digrafo G é acíclico se não tem circuito dirigido.

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Ordenação topológica

Digrafo $G = (V, A)$

Ordenação topológica: Permutação de V
em que u vem antes de v sempre que $(u, v) \in A$.

Digrafo G é acíclico se não tem circuito dirigido.

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Ordenação topológica

Digrafo $G = (V, A)$

Ordenação topológica: Permutação de V
em que u vem antes de v sempre que $(u, v) \in A$.

Digrafo G é acíclico se não tem circuito dirigido.

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Devolva a lista ligada.

Ordenação topológica

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Devolva a lista ligada.

Ordenação topológica

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Devolva a lista ligada.

Consumo de tempo: linear no tamanho do grafo

Ordenação topológica

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Devolva a lista ligada.

Consumo de tempo: linear no tamanho do grafo

Lema: Um digrafo é acíclico sse
a DFS não detecta nenhuma aresta de retorno.

Ordenação topológica

Problema: Dado digrafo G acíclico,
encontrar uma ordenação topológica de G.

Algoritmo:

Execute uma DFS no grafo calculando $f[u]$ para cada u.

Quando cada vértice u termina,
insira-o no início de uma lista ligada.

Devolva a lista ligada.

Consumo de tempo: linear no tamanho do grafo

Lema: Um digrafo é acíclico sse
a DFS não detecta nenhuma aresta de retorno.

Teorema: O algoritmo acima calcula uma ordenação topológica do grafo.