Análise e Síntese de Algoritmos

Algoritmos Elementares em Grafos [CLRS, Cap. 22]

2011/2012

Contexto

- Revisão [CLRS, Cap.1-13]
  - Fundamentos; notação; exemplos
- Algoritmos em Grafos [CLRS, Cap.21-26]
  - Algoritmos elementares
  - Árvores abrangentes
  - Caminhos mais curtos
  - Fluxos máximos
- Programação Linear [CLRS, Cap.29]
  - Algoritmos e modelação de problemas com restrições lineares
- Técnicas de Síntese de Algoritmos [CLRS, Cap.15-16]
  - Programação dinâmica
  - Algoritmos greedy
- Tópicos Adicionais [CLRS, Cap.32-35]
  - Emparelhamento de Cadeias de Caracteres
  - Complexidade Computacional
  - Algoritmos de Aproximação

Resumo

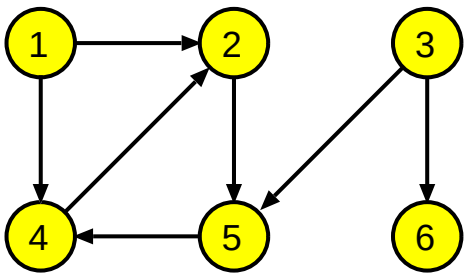
Grafo

Grafo

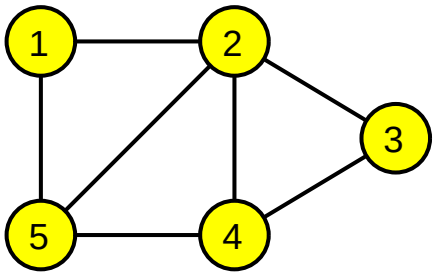
Grafo, $G = (V, E)$, definido por um conjunto V de vértices e um conjunto E de arcos

- Arcos representam ligações entre pares de vértices: $E \subseteq V \times V$
  - Grafo esparso se $|E| << |V \times V|$
  - Caso contrário diz-se que é um grafo denso
- Grafos podem ser dirigidos ou não dirigidos
  - Existência (ou não) da noção de direcção nos arcos

Grafo Dirigido



Grafo Não Dirigido

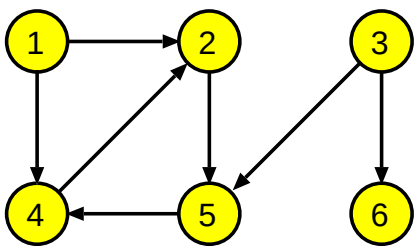


Representação de Grafos

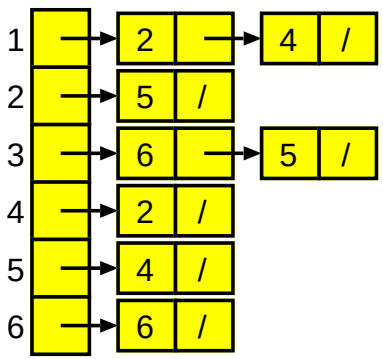
Representação dos arcos

- Matriz de adjacências: arcos representados por matriz
  - Para grafos densos
- Listas de adjacências: arcos representados por listas
  - Para grafos esparsos

Grafo Dirigido



Listas de Adjacências



Matriz de Adjacências

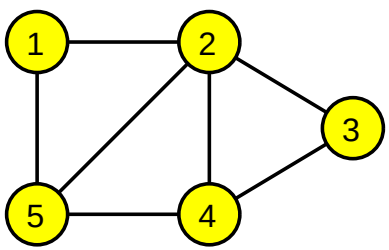
	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

Representação de Grafos

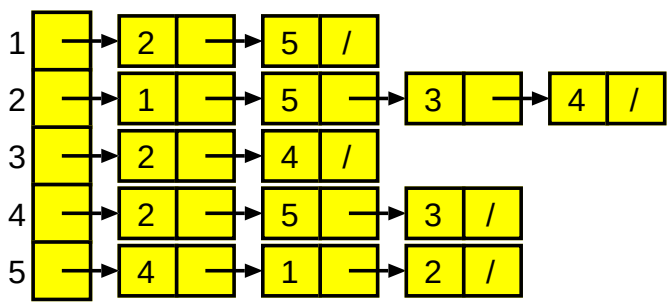
Representação dos arcos

- Matriz de adjacências: arcos representados por matriz
  - Para grafos densos
- Listas de adjacências: arcos representados por listas
  - Para grafos esparsos

Grafo Não Dirigido



Listas de Adjacências



Matriz de Adjacências

	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

Representação de Grafos

Matriz de Adjacências

- $\Theta(V^2)$ para qualquer grafo

Listas de adjacências

- Tamanho das listas é $|E|$ para grafos dirigidos
- Tamanho das listas é $2|E|$ para grafos não dirigidos
- Tamanho total das listas de adjacências é $O(V+E)$

Grafos pesados

- Existência de uma função de pesos $\omega : E \rightarrow IR$
- Função de pesos ω associa um peso a cada arco

Procura em Largura Primeiro (Breadth-First Search)

Dados $G = (V, E)$ e vértice fonte s, o algoritmo BFS explora sistematicamente os vértices de G para descobrir todos os vértices atingíveis a partir de s

- Fronteira entre nós descobertos e não descobertos é expandida uniformemente
  - Nós à distância k descobertos antes de qualquer nó à distância $k + 1$
- Cálculo da distância: menor número de arcos de s para cada vértice atingível
- Identificação de árvore Breadth-First (BF): caminho mais curto de s para cada vértice atingível v

BFS: Notação

Implementação

- $color[v]$: cor do vértice v, (branco, cinzento ou preto)
  - branco: não visitado
  - cinzento: já visitado, mas algum dos adjacentes não visitado ou procura em algum dos adjacentes não terminada
  - preto: já visitado e procura nos adjacentes já terminada
- $\pi[v]$: predecessor de v na árvore BF
- $d[v]$: tempo de descoberta do vértice v

Outras definições

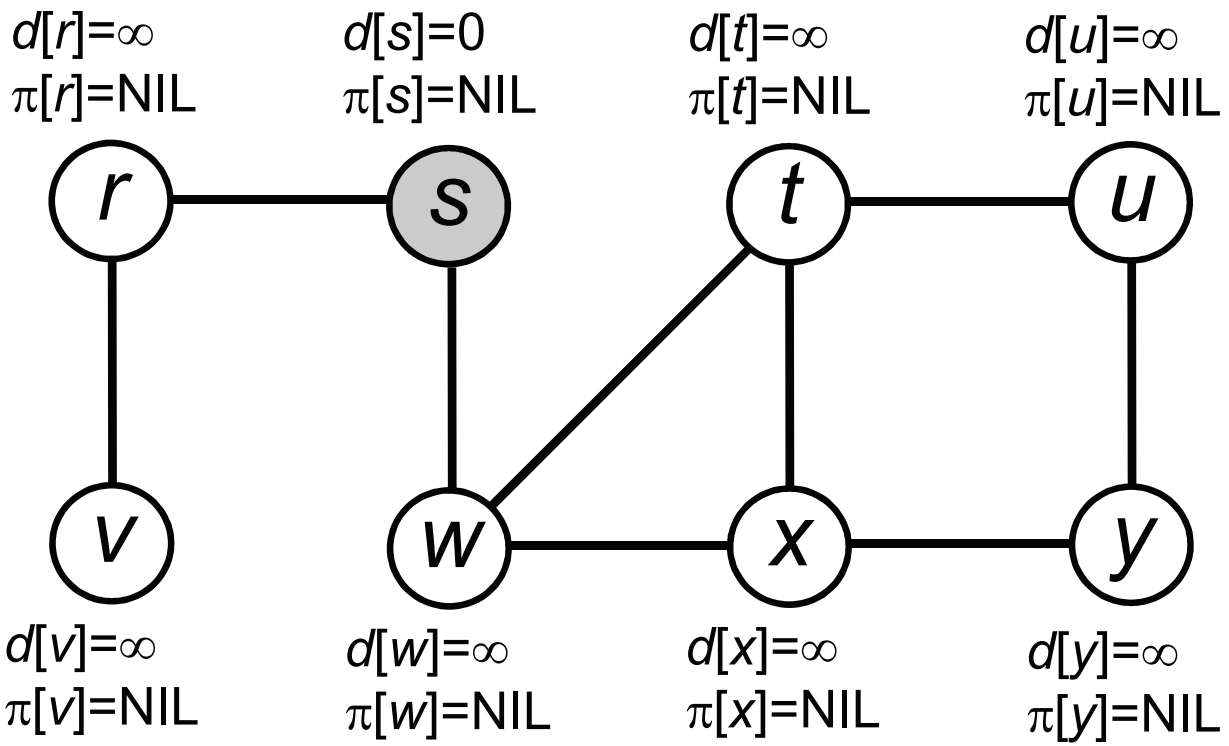
- Caminho mais curto definido como o caminho de s para v composto pelo menor número de arcos
- $\delta(s,v)$: denota o comprimento do caminho mais curto de s a v

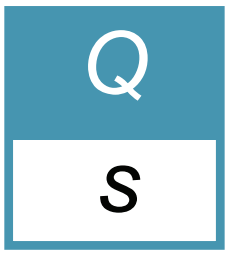
BFS: Pseudo-Código

BFS(G, s)

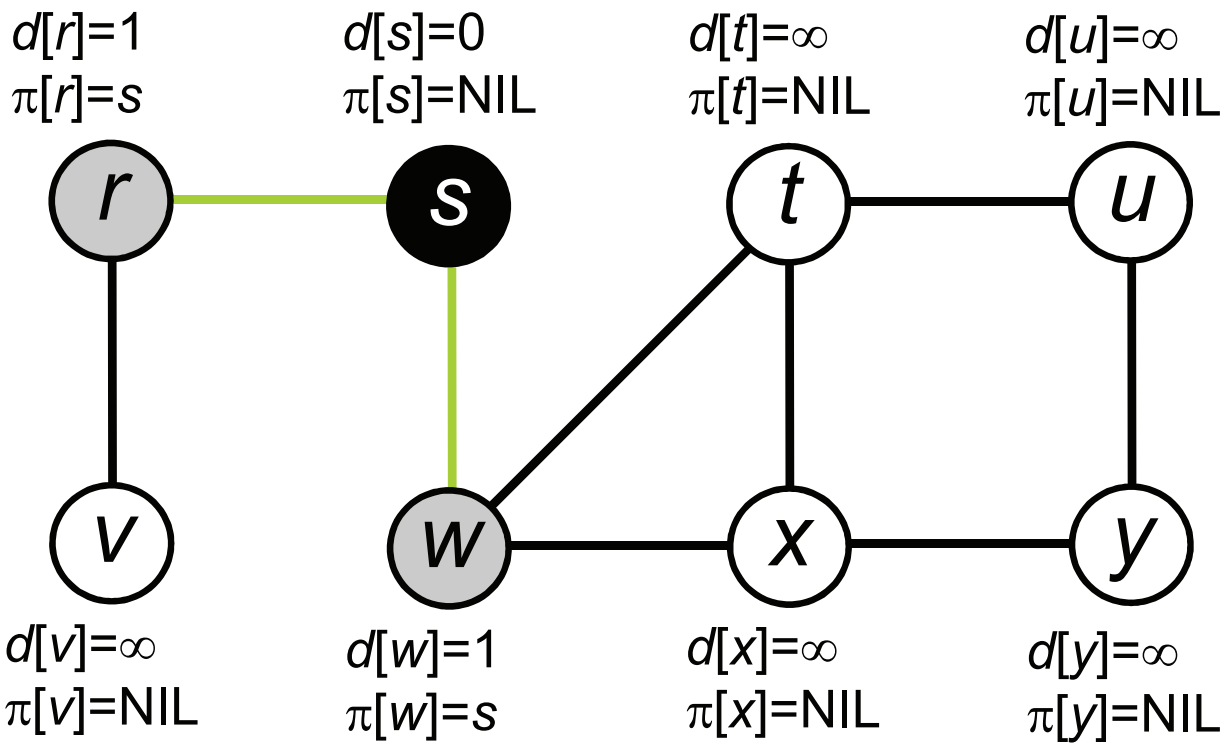
```
 1  for each vertex u ∈ V[G] − {s}                ▷ Inicialização
 2        do color[u] ← white; d[u] ← ∞; π[u] ← NIL
 3  color[s] ← gray; d[s] ← 0; π[s] ← NIL
 4  Q = {s}
 5  while Q ≠ ∅                                   ▷ Ciclo Principal
 6        do u ← Dequeue(Q)
 7           for each v ∈ Adj[u]
 8                do if color[v] = white
 9                      then color[v] ← gray; d[v] ← d[u] + 1; π[v] ← u
10                           Enqueue(Q, v)
11           color[u] ← black
```

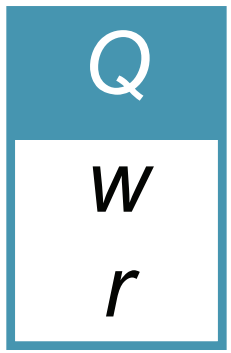
BFS: Exemplo


d[r]=∞
π[r]=NIL
d[s]=0
π[s]=NIL
d[t]=∞
π[t]=NIL
d[u]=∞
π[u]=NIL
r
s
t
u
v
w
x
y
d[v]=∞
π[v]=NIL
d[w]=∞
π[w]=NIL
d[x]=∞
π[x]=NIL
d[y]=∞
π[y]=NIL

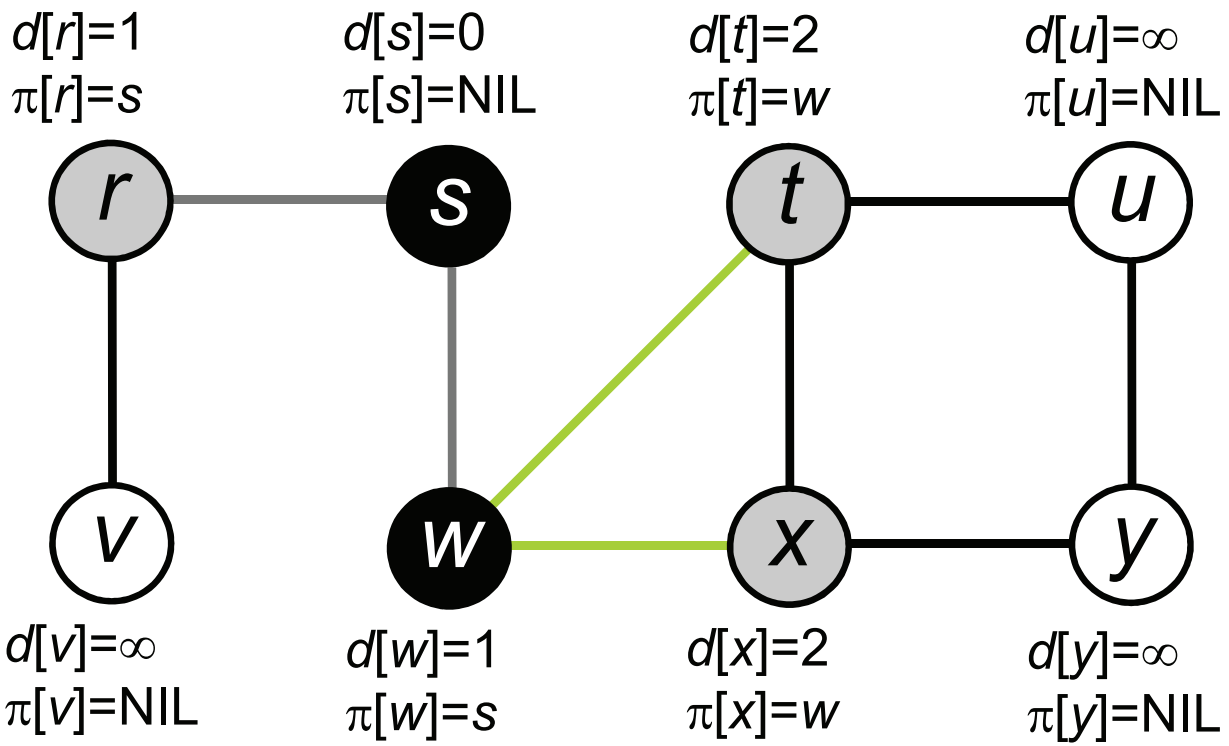

Q
s

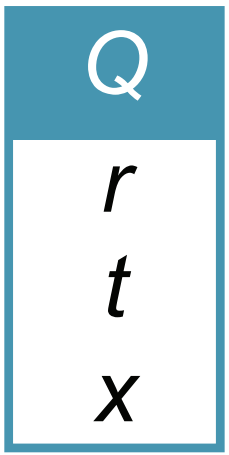
BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=∞
π[t]=NIL
d[u]=∞
π[u]=NIL
r
s
t
u
v
w
x
y
d[v]=∞
π[v]=NIL
d[w]=1
π[w]=s
d[x]=∞
π[x]=NIL
d[y]=∞
π[y]=NIL

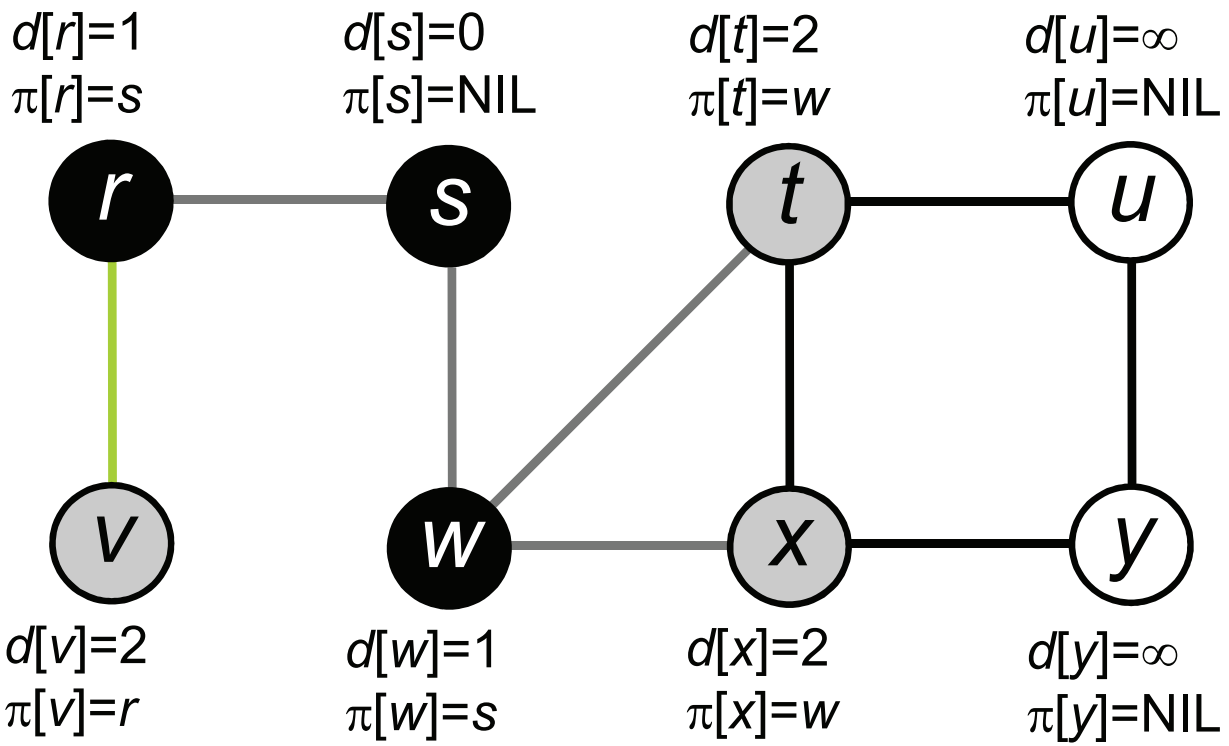

Q
w
r

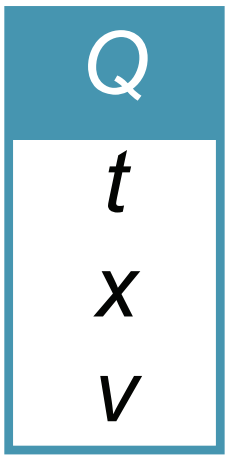
BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=∞
π[u]=NIL
r
s
t
u
v
w
x
y
d[v]=∞
π[v]=NIL
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=∞
π[y]=NIL

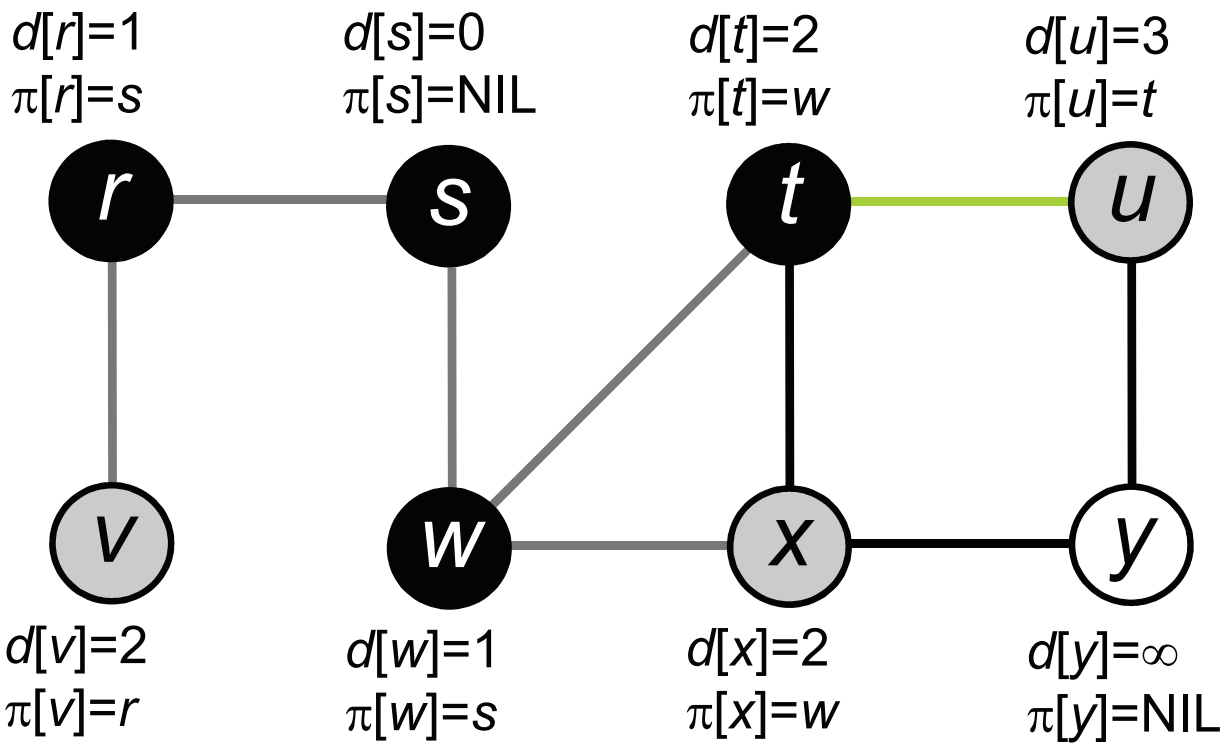

Q
r
t
x

BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=∞
π[u]=NIL
r
s
t
u
v
w
x
y
d[v]=2
π[v]=r
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=∞
π[y]=NIL

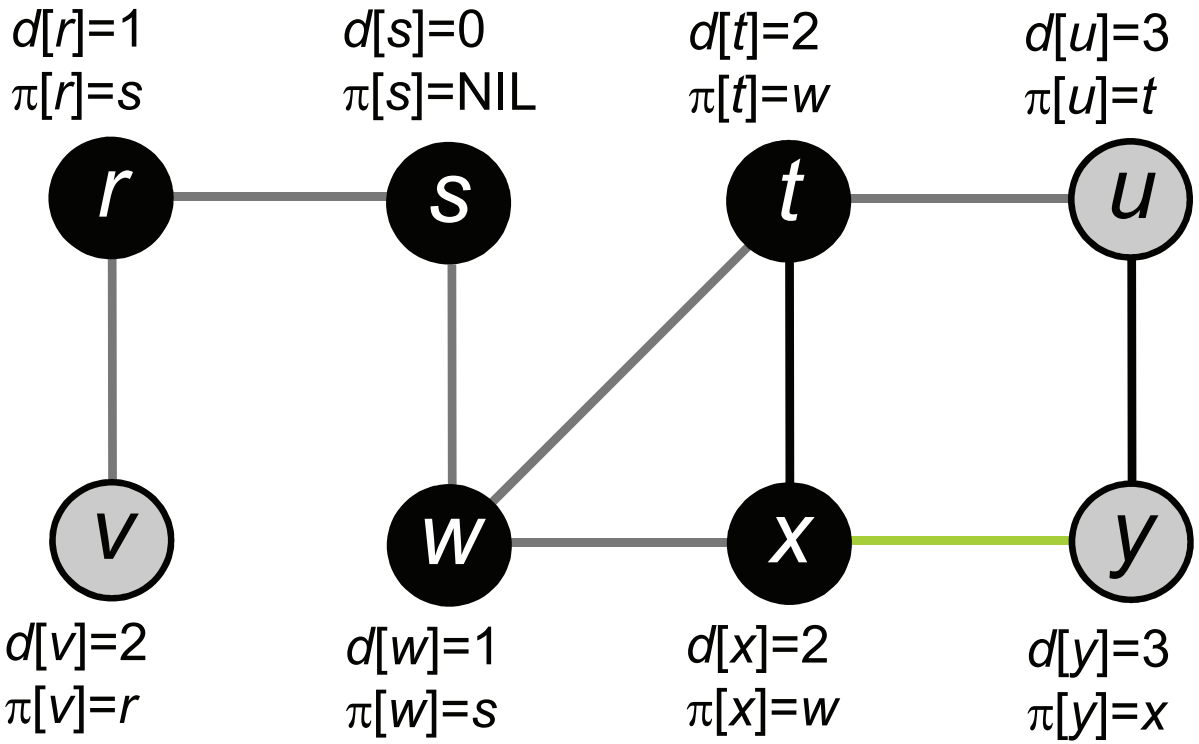

Q
t
x
v

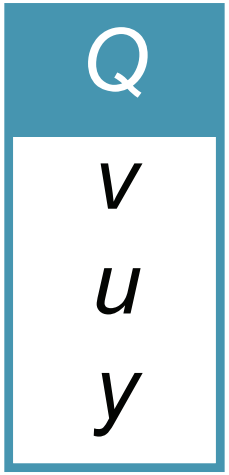
BFS: Exemplo



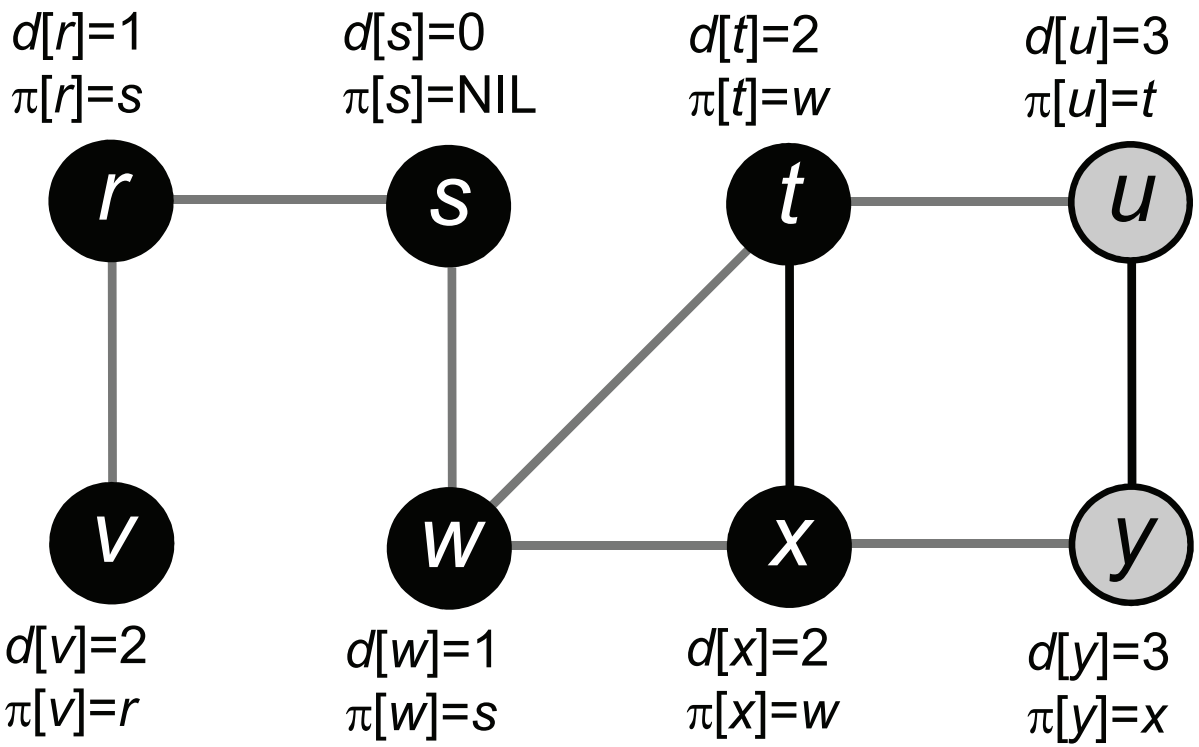
Q
x
v
u

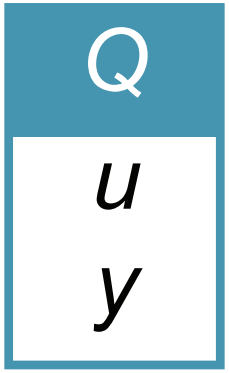
BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=3
π[u]=t
r
s
t
u
v
w
x
y
d[v]=2
π[v]=r
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=3
π[y]=x

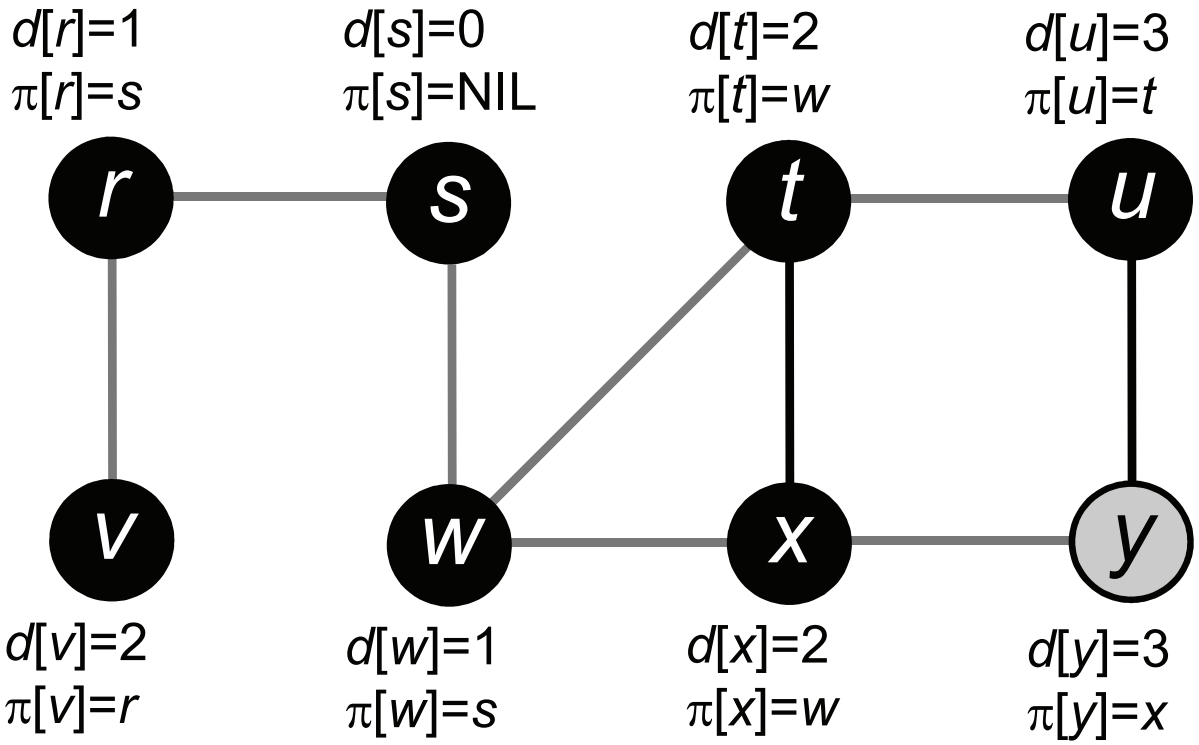

Q
v
u
y

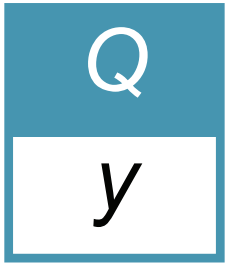
BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=3
π[u]=t
r
s
t
u
v
w
x
y
d[v]=2
π[v]=r
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=3
π[y]=x

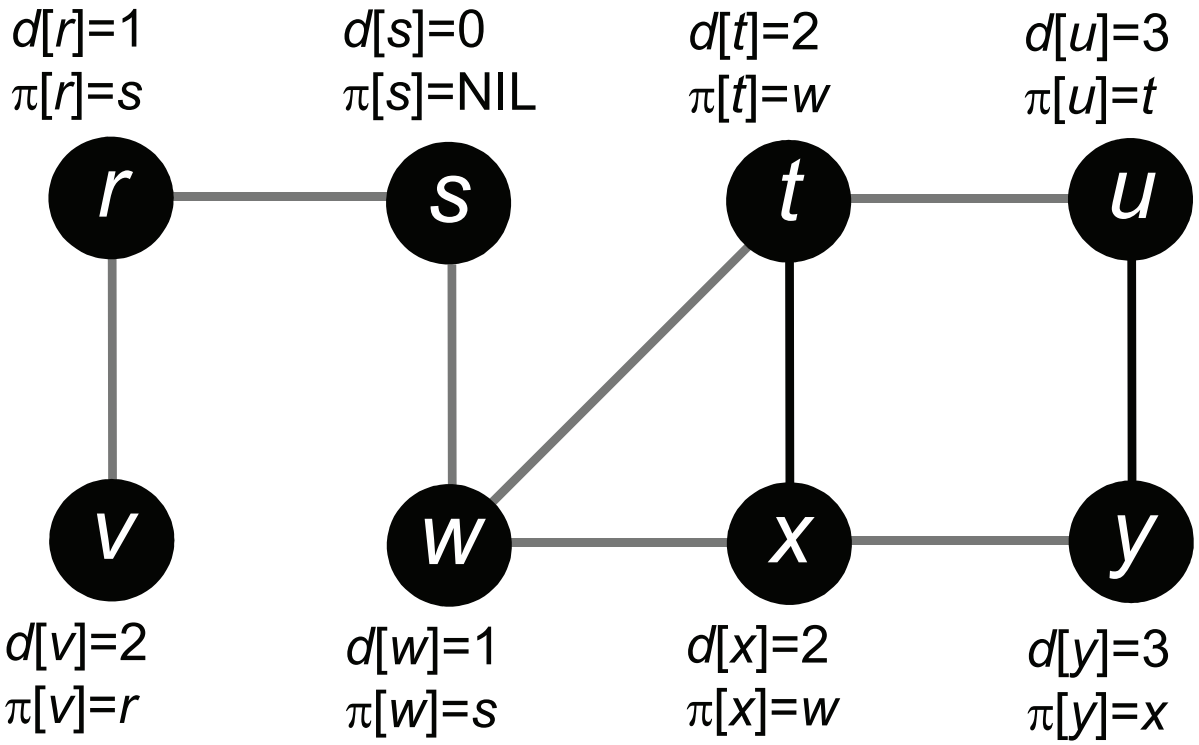

Q
u
y

BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=3
π[u]=t
r
s
t
u
v
w
x
y
d[v]=2
π[v]=r
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=3
π[y]=x


Q
y

BFS: Exemplo


d[r]=1
π[r]=s
d[s]=0
π[s]=NIL
d[t]=2
π[t]=w
d[u]=3
π[u]=t
r
s
t
u
v
w
x
y
d[v]=2
π[v]=r
d[w]=1
π[w]=s
d[x]=2
π[x]=w
d[y]=3
π[y]=x


Q

BFS: Complexidade

Complexidade

Tempo de execução: $O(V+E)$

- Inicialização: $O(V)$
- Para cada vértice
  - Colocado na fila apenas 1 vez: $O(V)$
  - Lista de adjacências visitada 1 vez: $O(E)$

BFS: Resultados

Resultados

- Para qualquer arco (u,v) : $\delta(s,v) \leq \delta(s,u)+1$
- Se u atingível, então v atingível
  - caminho mais curto para v não pode ser superior a caminho mais curto para u mais arco (u,v)
  - Se u não atingível, então resultado é válido (independentemente de v ser atingível)
- No final da BFS: $d[u] = \delta(s,u)$, para todo o vértice $u \in V$

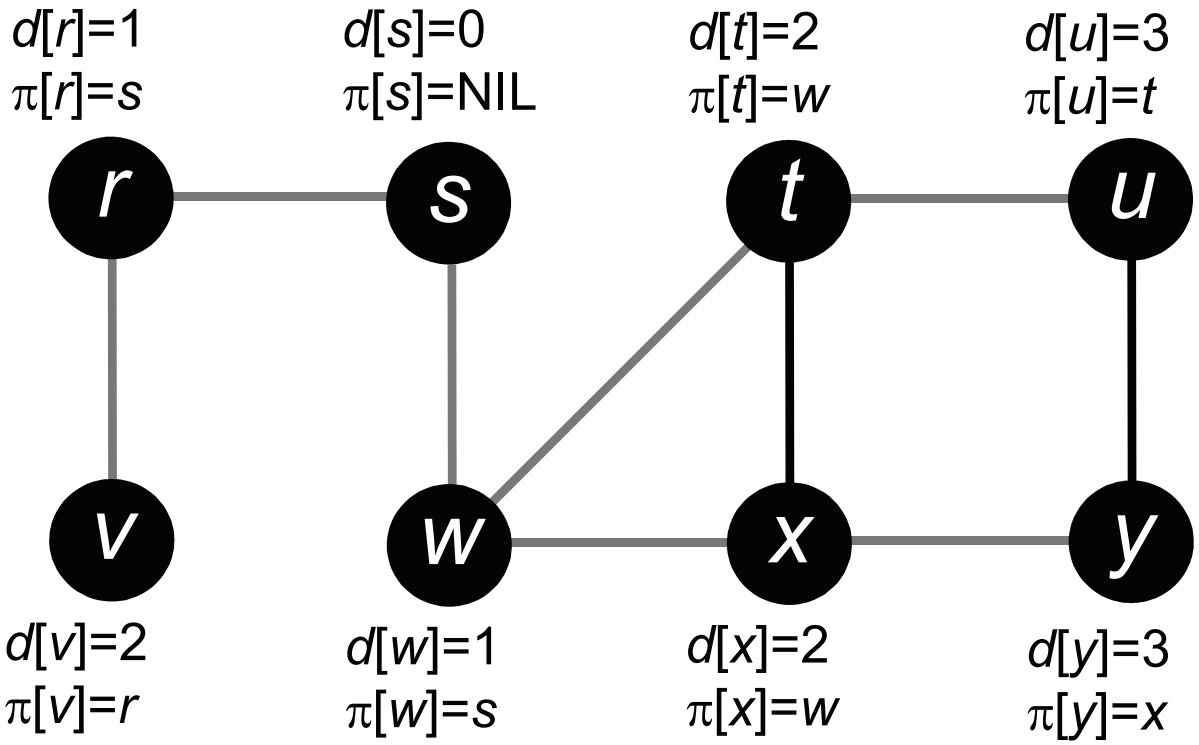
Árvore Breadth-First

Árvore Breadth-First

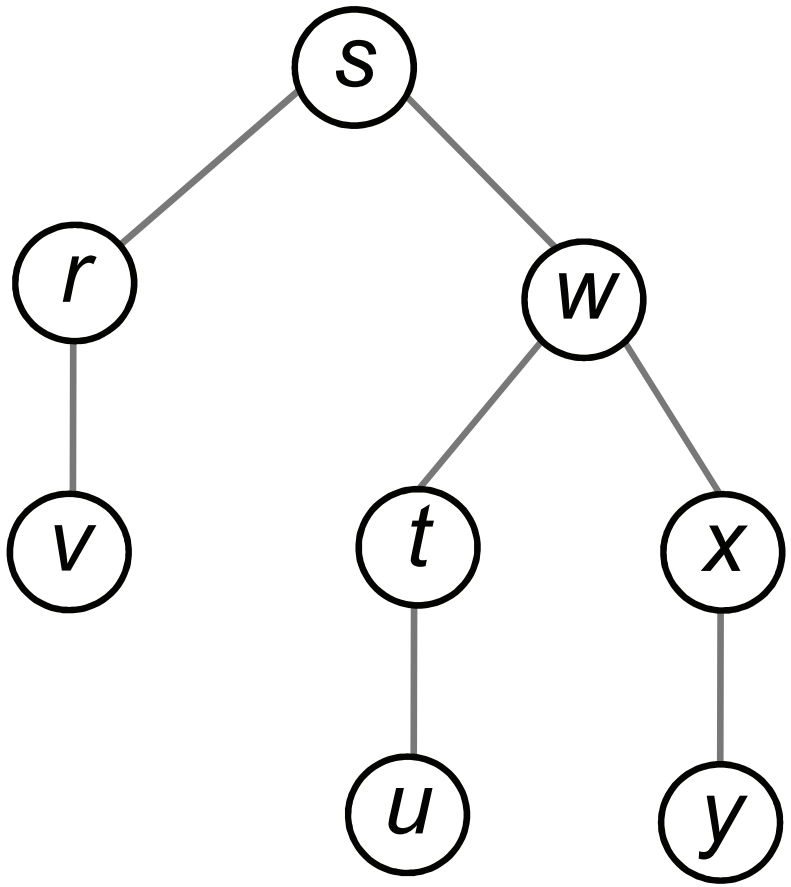
- Árvore BF é sub-grafo de G
- Vértices atingíveis a partir de s
- Arcos que definem a relação predecessor da BFS

- $G_\pi = (V_\pi, E_\pi)$
- $V_\pi = \{v \in V : \pi[v] \neq NIL\} \cup \{s\}$
- $E_\pi = \{(\pi[v], v) \in E : v \in V_\pi - \{s\}\}$

Árvore Breadth-First



Árvore Breadth-First



Procura Profundidade Primeiro (Depth-First Search)

Grafo pesquisado dando prioridade aos arcos dos vértices mais recentemente visitados

Resultados

Floresta Depth-First (DF)

- $G_\pi = (V, E_\pi)$
- $E_\pi = \{(\pi[v], v) : v \in V \wedge \pi[v] \neq NIL\}$
- Floresta DF composta por várias árvores DF

Implementação

- $color[v]$: cor do vértice v, (branco, cinzento ou preto)
- $\pi[v]$: predecessor de v na árvore DF
- $d[v]$: tempo de início (de visita do vértice)
- $f[v]$: tempo de fim (de visita do vértice)

DFS: Pseudo-Código

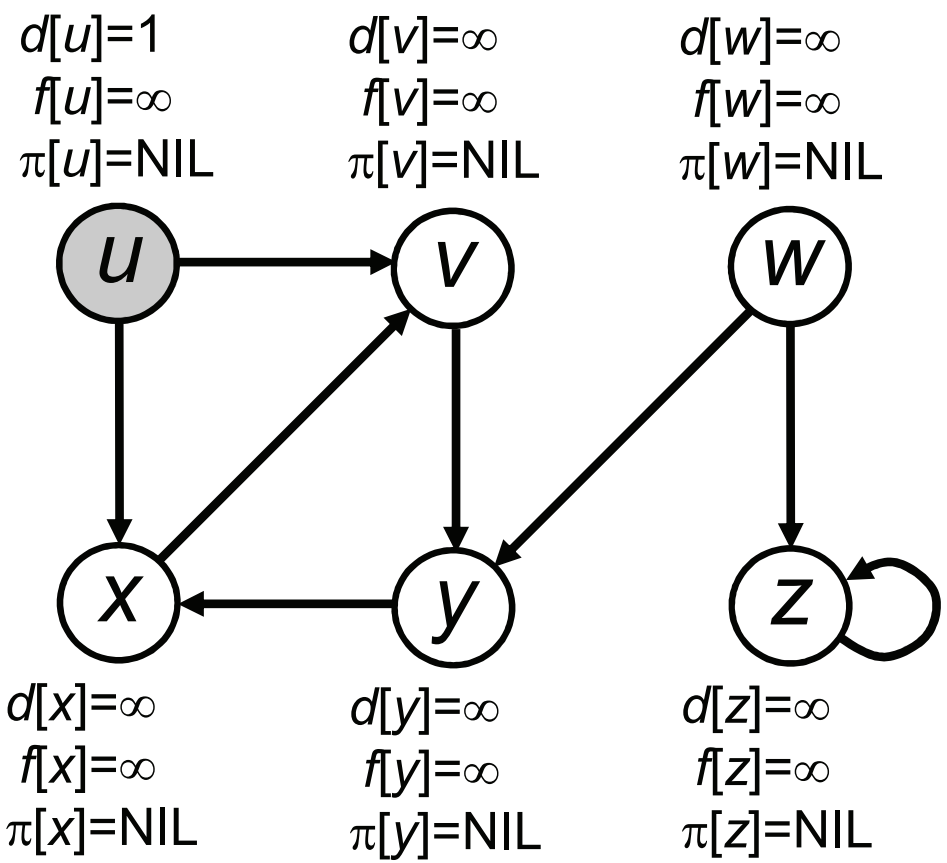
DFS(G)

```
1  for each vertex u ∈ V[G]
2      do color[u] ← white; π[u] = NIL
3  time ← 1
4  for each vertex u ∈ V[G]
5      do if color[u] = white
6          then DFS-Visit(u)
```

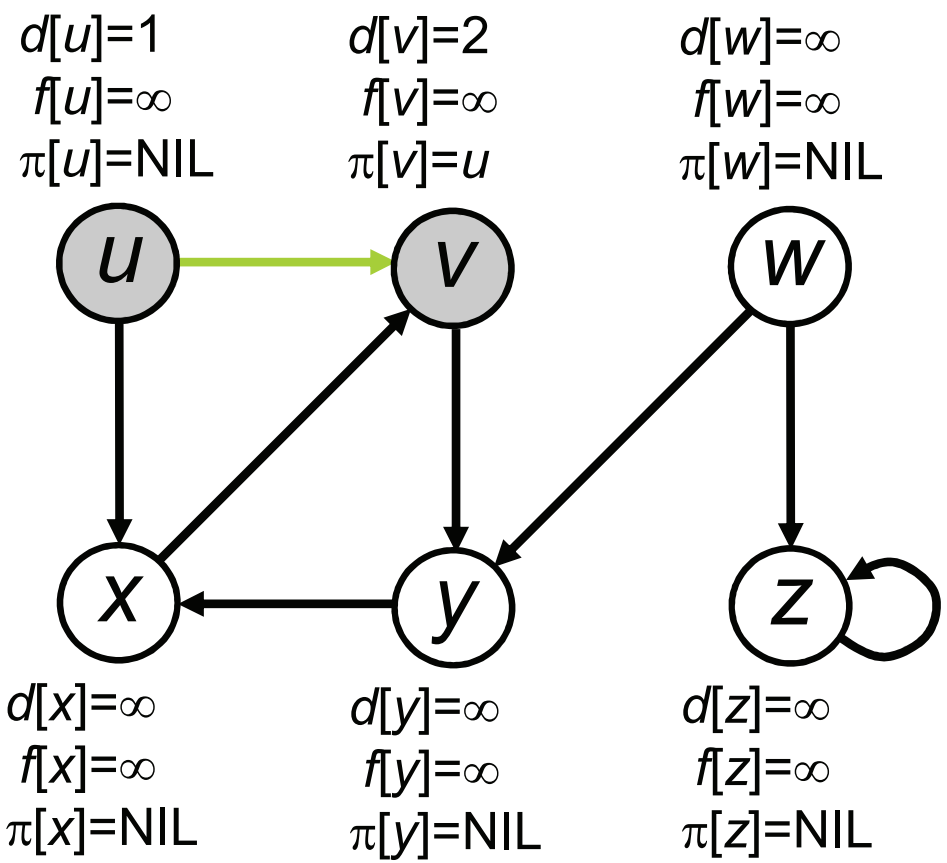
DFS-Visit(u)

```
1  color[u] ← gray
2  d[u] ← time
3  time ← time + 1
4  for each v ∈ Adj[u]
5      do if color[v] = white
6          then π[v] ← u
7               DFS-Visit(v)
8  color[u] ← black
9  f[u] ← time
10 time ← time + 1
```

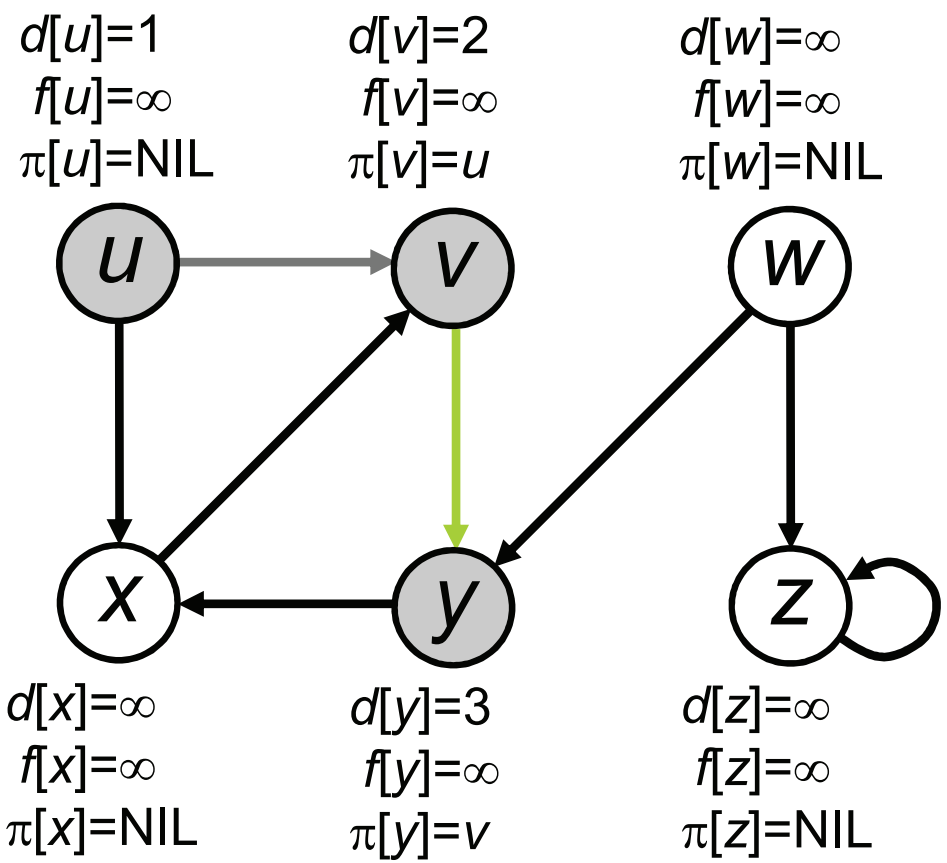
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=∞
f[v]=∞
π[v]=NIL
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
x
y
z
d[x]=∞
f[x]=∞
π[x]=NIL
d[y]=∞
f[y]=∞
π[y]=NIL
d[z]=∞
f[z]=∞
π[z]=NIL

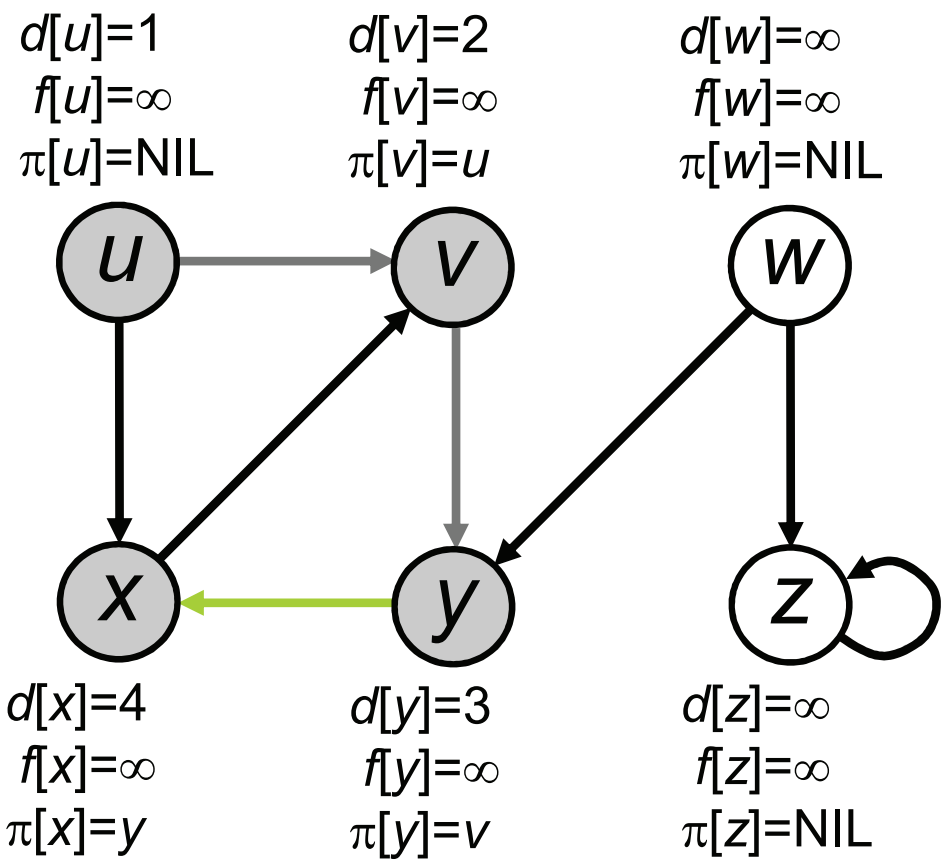
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
x
y
z
d[x]=∞
f[x]=∞
π[x]=NIL
d[y]=∞
f[y]=∞
π[y]=NIL
d[z]=∞
f[z]=∞
π[z]=NIL

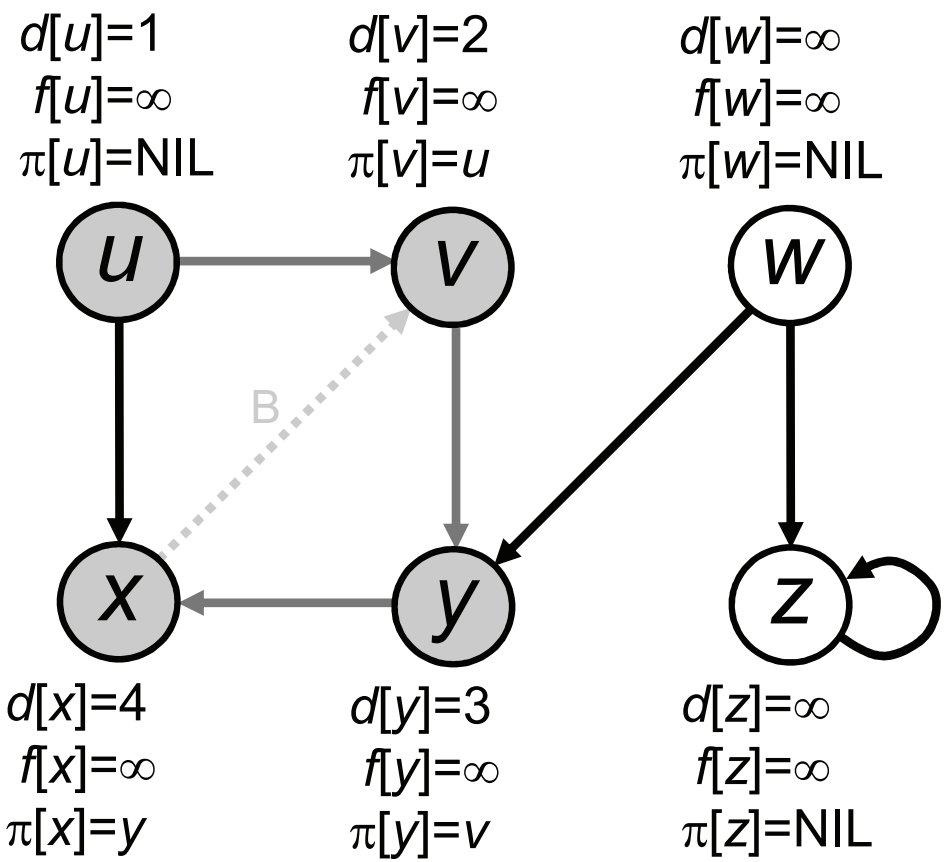
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
x
y
z
d[x]=∞
f[x]=∞
π[x]=NIL
d[y]=3
f[y]=∞
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

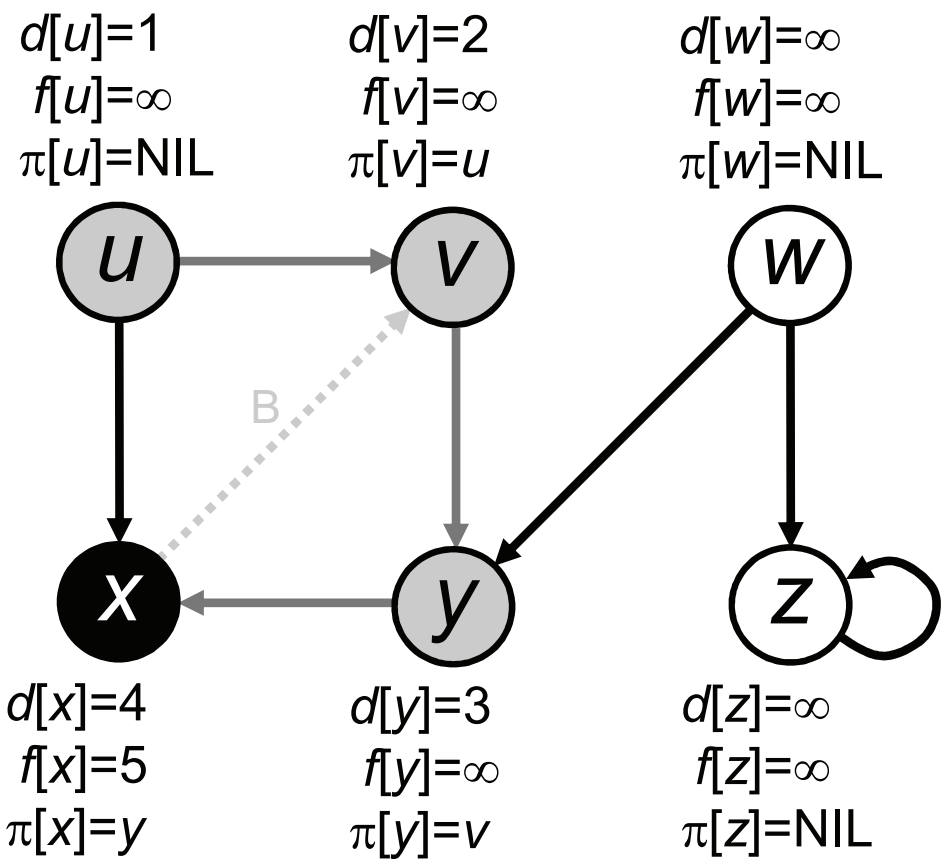
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
x
y
z
d[x]=4
f[x]=∞
π[x]=y
d[y]=3
f[y]=∞
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

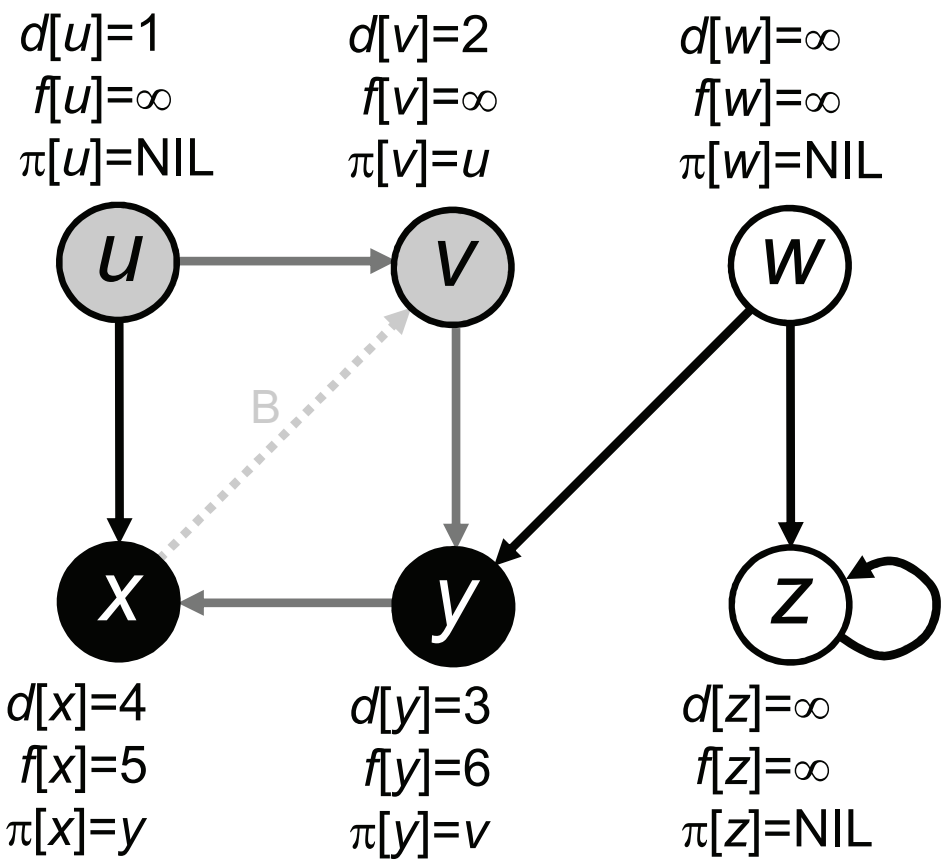
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
B
x
y
z
d[x]=4
f[x]=∞
π[x]=y
d[y]=3
f[y]=∞
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

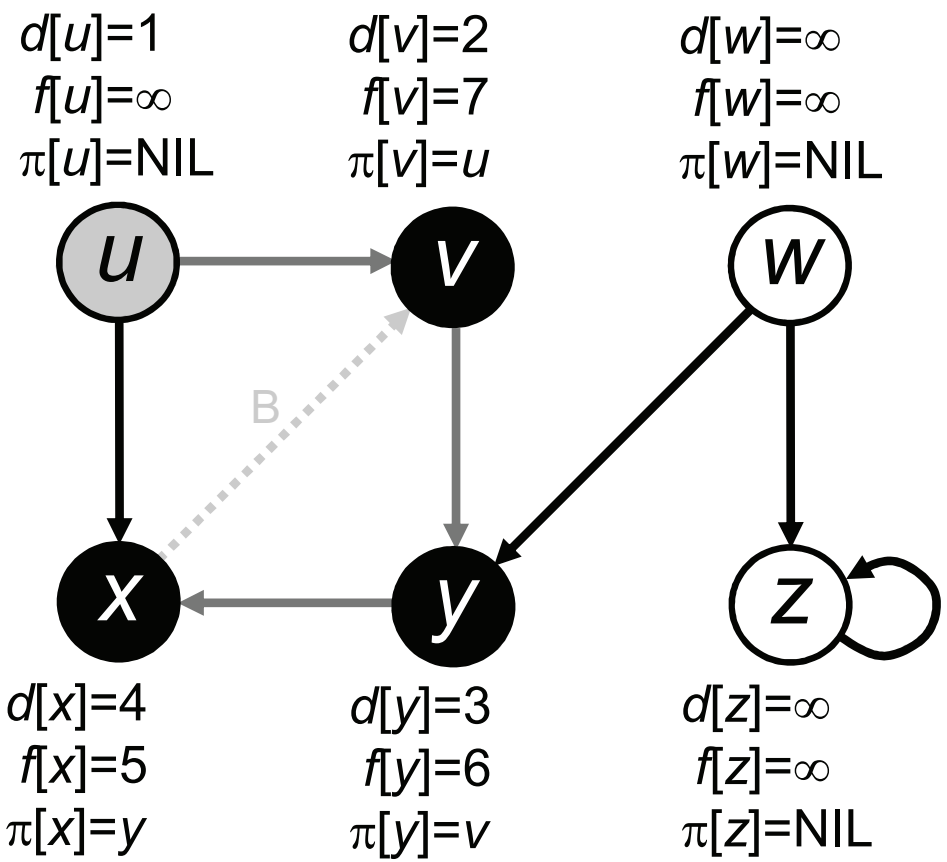
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
B
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=∞
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

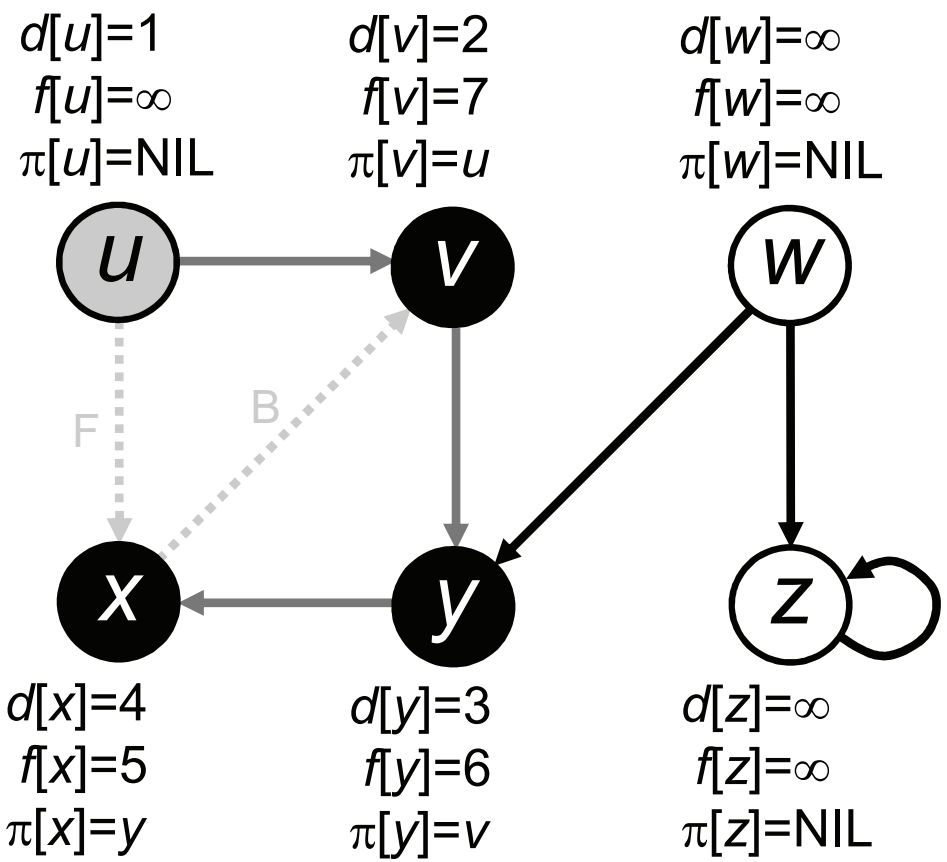
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=∞
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
B
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

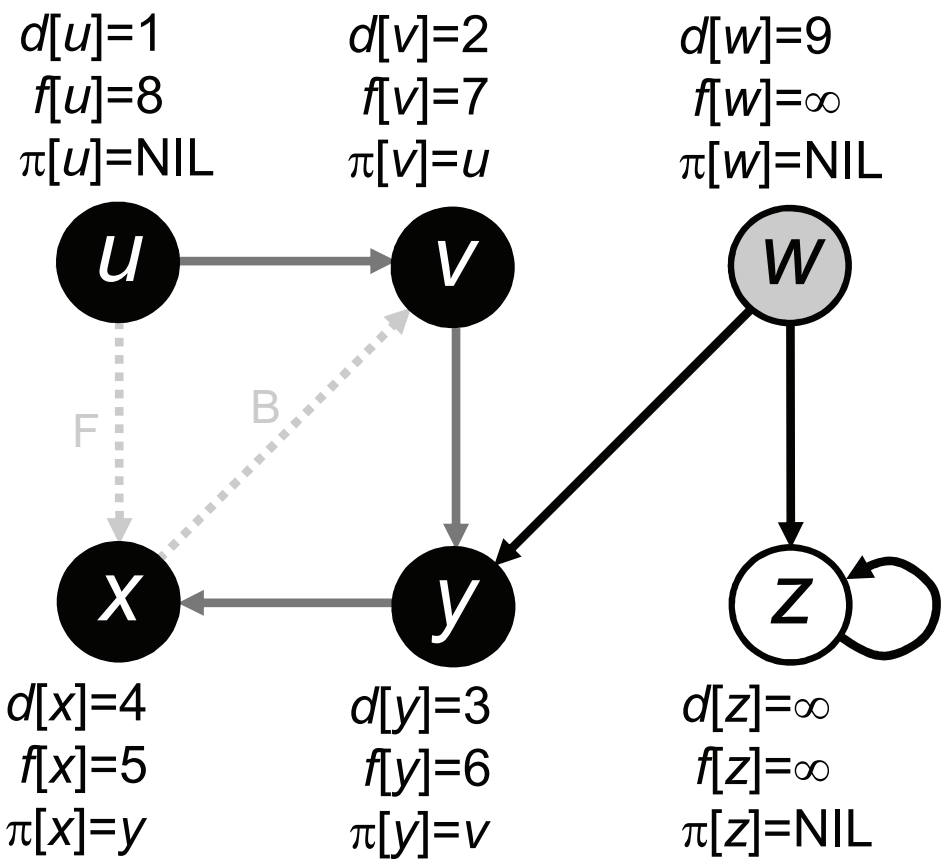
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
B
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

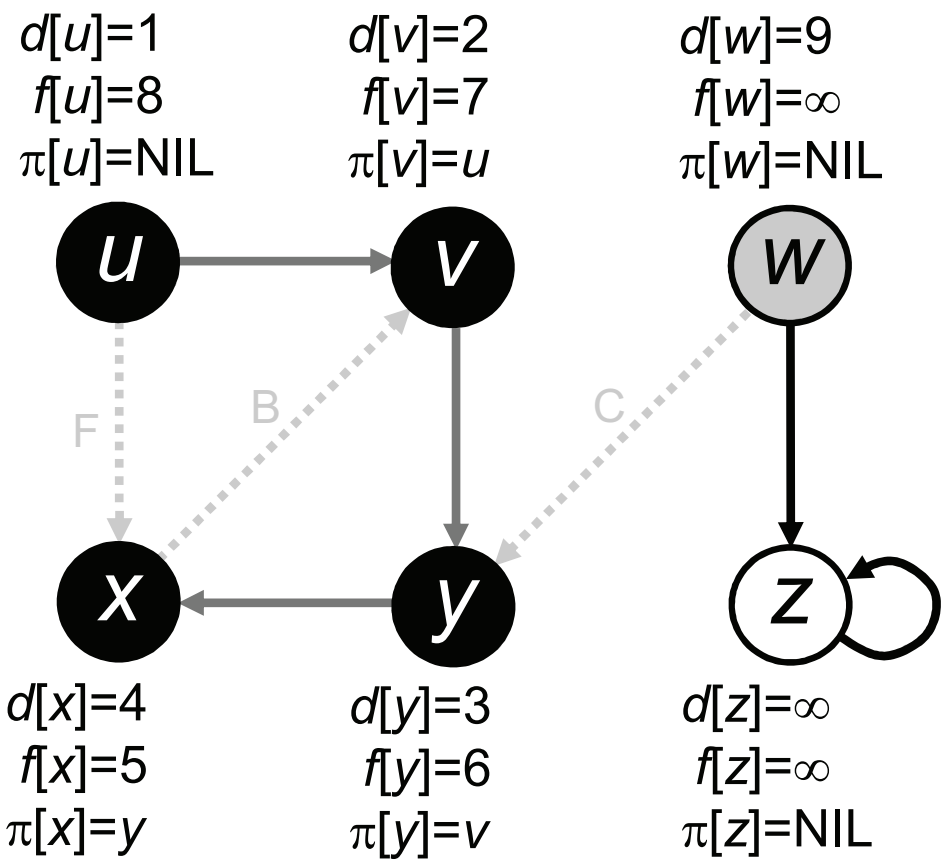
DFS: Exemplo


d[u]=1
f[u]=∞
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=∞
f[w]=∞
π[w]=NIL
u
v
w
F
B
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

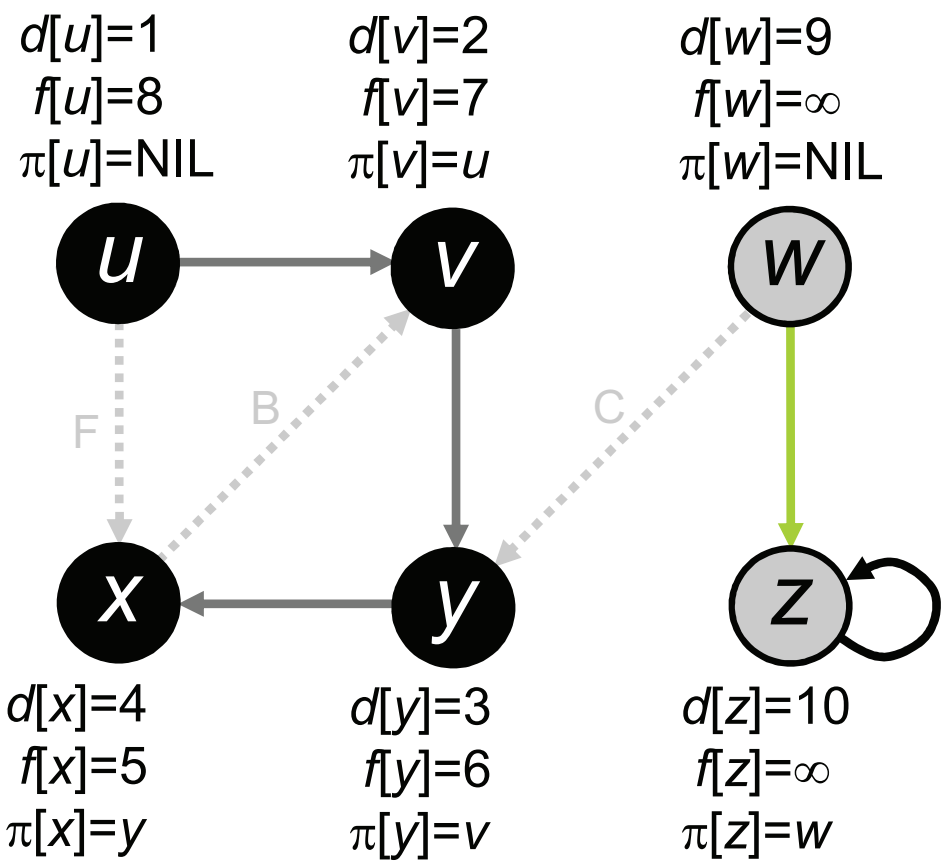
DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=∞
π[w]=NIL
u
v
w
F
B
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

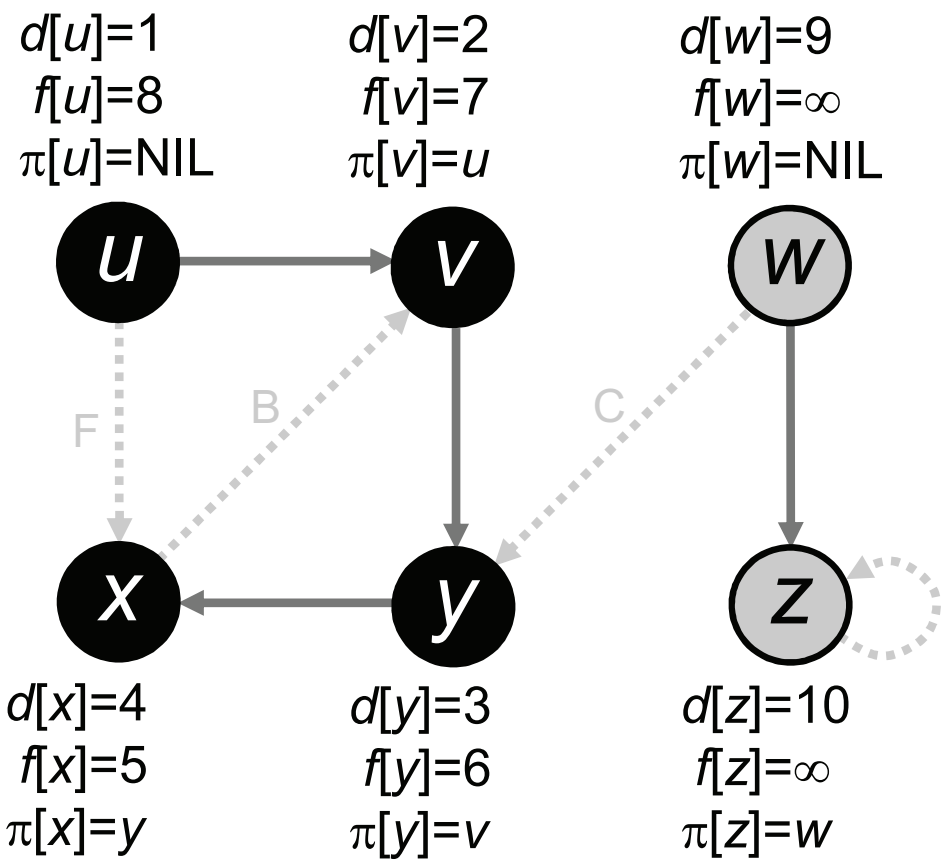
DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=∞
π[w]=NIL
u
v
w
F
B
C
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=∞
f[z]=∞
π[z]=NIL

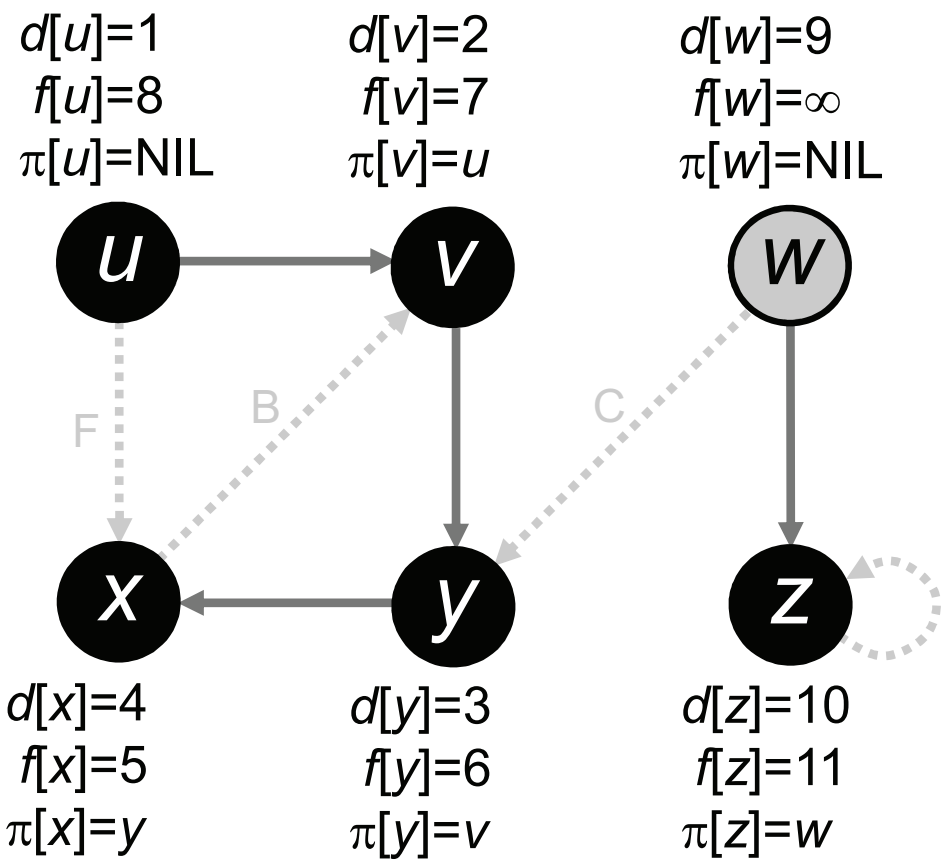
DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=∞
π[w]=NIL
u
v
w
F
B
C
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=10
f[z]=∞
π[z]=w

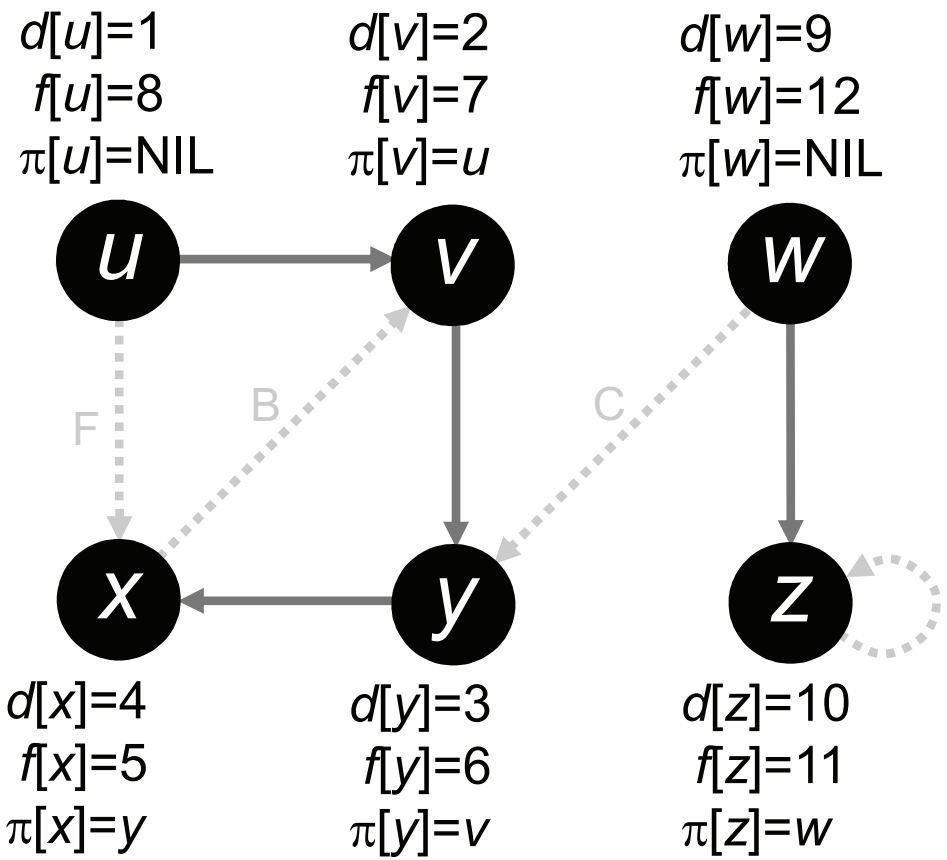
DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=∞
π[w]=NIL
u
v
w
F
B
C
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=10
f[z]=∞
π[z]=w

DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=∞
π[w]=NIL
u
v
w
F
B
C
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=10
f[z]=11
π[z]=w

DFS: Exemplo


d[u]=1
f[u]=8
π[u]=NIL
d[v]=2
f[v]=7
π[v]=u
d[w]=9
f[w]=12
π[w]=NIL
u
v
w
F
B
C
x
y
z
d[x]=4
f[x]=5
π[x]=y
d[y]=3
f[y]=6
π[y]=v
d[z]=10
f[z]=11
π[z]=w

DFS: Complexidade

Complexidade

Tempo de execução: $O(V+E)$

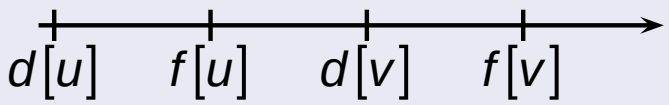
- Inicialização: $O(V)$
- Chamadas a DFS-Visit dentro de DFS: $O(V)$
- Arcos analisados em DFS-Visit: $\Theta(E)$
  - Chamadas a DFS-Visit dentro de DFS-Visit: $O(V)$
  - Mas $\sum_{v \in V} |Adj[v]| = \Theta(E)$

DFS: Resultados

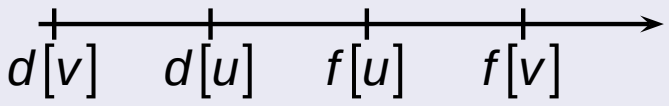
Resultados

Numa DFS de $G = (V, E)$, para cada par de vértices u e v apenas um dos 3 casos seguintes é verdade:

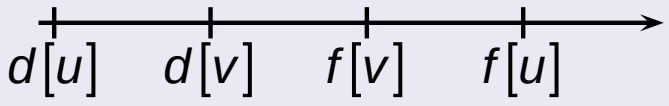
- Os intervalos $[d[u], f[u]]$ e $[d[v], f[v]]$ são disjuntos



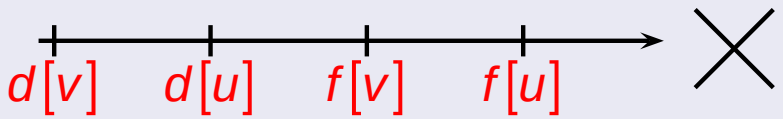
- $[d[u], f[u]] \subset [d[v], f[v]]$ e u é descendente de v na árvore DF



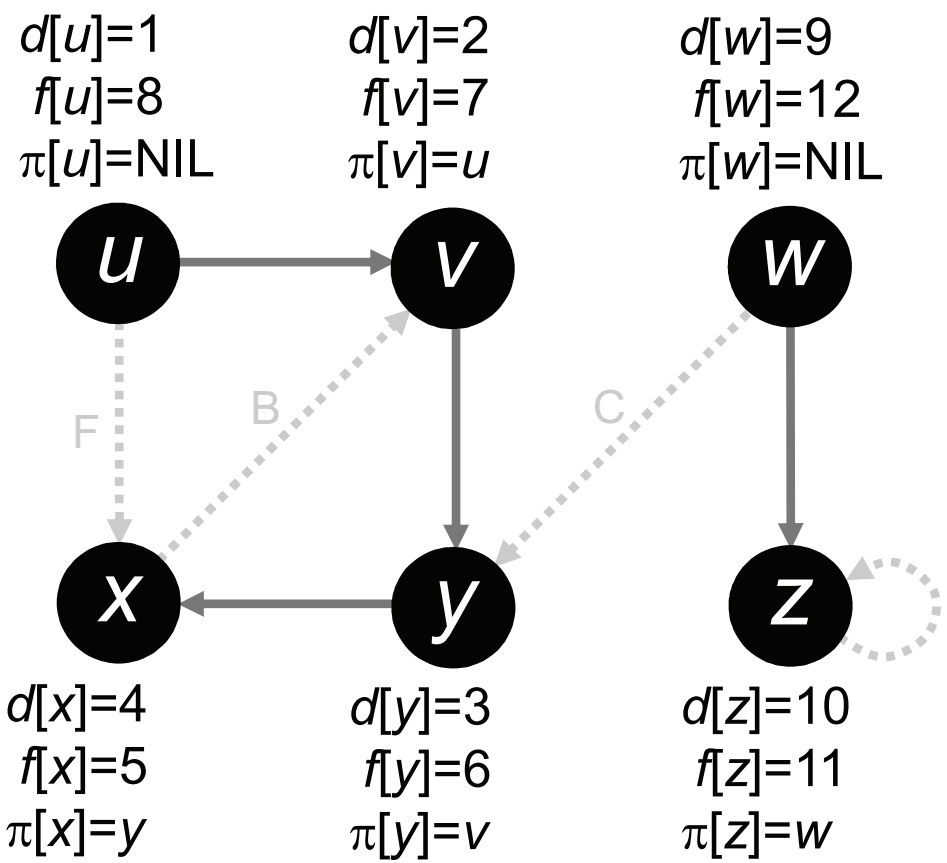
- $[d[v], f[v]] \subset [d[u], f[u]]$ e v é descendente de u na árvore DF



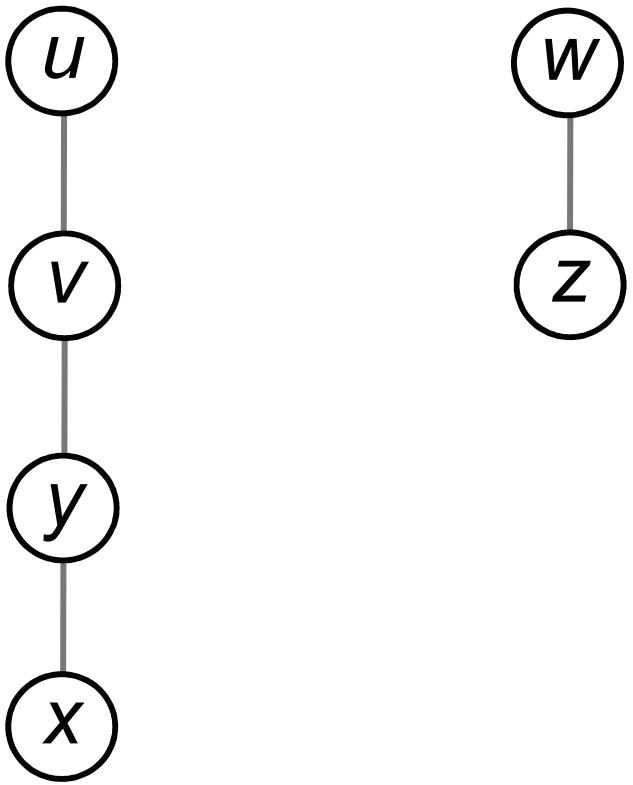
- A seguinte situação não pode ocorrer



Floresta Breadth-First



Floresta Breadth-First



DFS: Classificação de Arcos

Classificação de arcos (u, v)

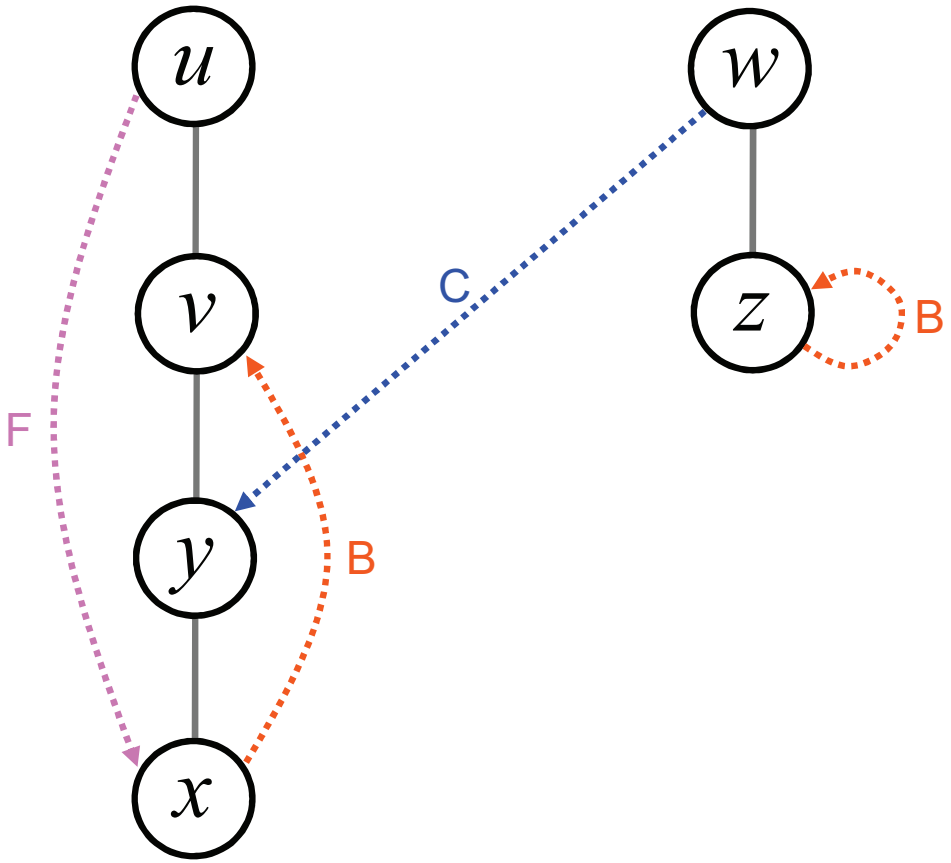
- Arcos de árvore: (tree edges)
  - arcos na floresta DF, G_π
  - (u, v) é arco de árvore se v foi visitado devido ao arco (u, v) ser visitado
- Arcos para trás: (back edges)
  - ligam vértice u a vértice v antecessor na mesma árvore DF
- Arcos para a frente: (forward edges)
  - ligam vértice v a vértice descendente na mesma árvore DF
- Arcos de cruzamento: (cross edges)
  - Na mesma árvore DF, se u (ou v) não antecessor de v (ou u)
  - Entre árvores DF diferentes

DFS: Classificação de Arcos

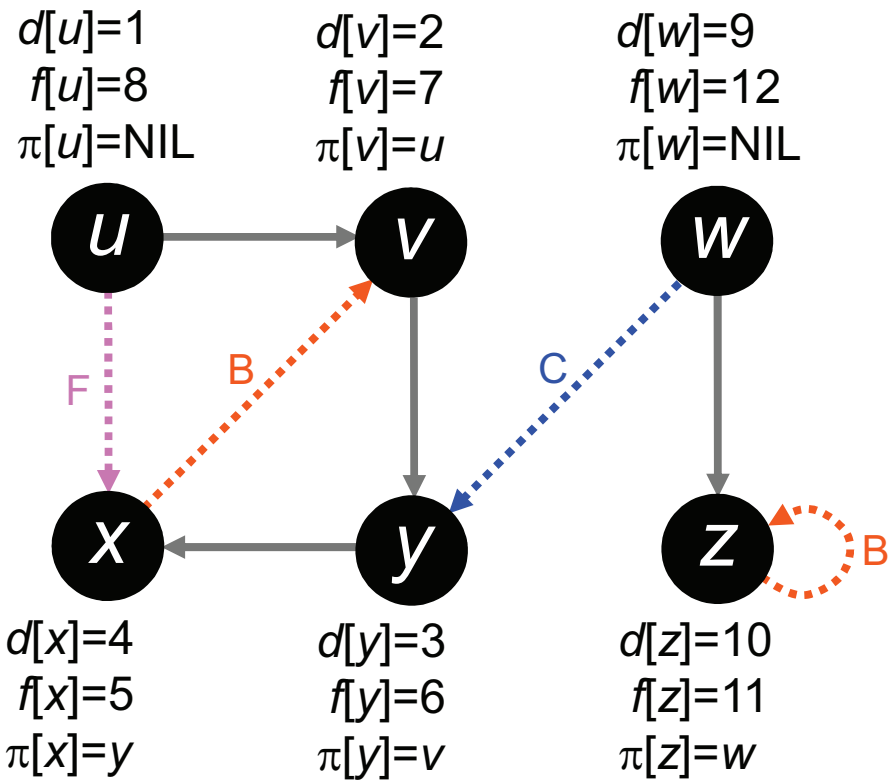
Classificação de arcos (u, v)

- Arcos de árvore: (tree edges)
  - $d[u] < d[v] < f[v] < f[u]$
  - $color[v] = white$ quando (u, v) é analisado
- Arcos para trás: (back edges)
  - $d[v] < d[u] < f[u] < f[v]$
  - $color[v] = gray$ quando (u, v) é analisado
- Arcos para a frente: (forward edges)
  - $d[u] < d[v] < f[v] < f[u]$
  - $color[v] = black$ quando (u, v) é analisado
- Arcos de cruzamento: (cross edges)
  - $d[v] < f[v] < d[u] < f[u]$
  - $color[v] = black$ quando (u, v) é analisado

DFS: Classificação de Arcos



DFS: Classificação de Arcos



Arcos de árvore (tree edges)

B — Arcos para trás (back edges)

F — Arcos para a frente (forward edges)

C — Arcos de cruzamento (cross edges)

DFS: Resultados

Resultados

Dado $G = (V, E)$, não dirigido, cada arco é arco de árvore ou arco para trás

- i.e., não existem arcos para a frente e de cruzamento

Numa floresta DF, v é descendente de u se e só se quando u é descoberto existe um caminho de vértices brancos de u para v

- v descendente de $u \Rightarrow$ existe caminho de vértices brancos de u para v
  - Qualquer vértice w descendente de u verifica $[d[w], f[w]] \subset [d[u], f[u]]$, pelo que w é branco quando u é descoberto

Resumo

Representação de grafos

- Listas de adjacências
- Matrizes de adjacências

Algoritmos Elementares

- Procura em Largura Primeiro (BFS)
  - Caminhos mais curtos no número de arcos
- Procura em Profundidade Primeiro (DFS)
  - Tempos de início ($d[]$) e de fim ($f[]$)
  - Classificação de arcos