Titolo nota 26/02/2016

VISITE di GRAFI

Visita in Ampiezza
BFS
"breadth first search"

Visita in profondità
DFS
"Depth first search"

$G = (V, E)$
sorgente $s \in V$
scopre i vertici in ordine di distanza crescente dalla sorgente

- si utilizza una CODA
- colorazione dei vertici ($\simeq$ marcatura)

Bianco: non ancora scoperto

Grigio: vertice scoperto

Nero: vertice scoperto la cui lista di adiacenza è stata completamente esaminata

Titolo nota 26/02/2016

- Scopre tutti e soli i vertici raggiungibili da s (tutto G, se G è connesso)
- calcola le distanze dalla sorgente di tutti i vertici scoperti.

$\forall u \in V$

$u.color$
$u.d$ // distanza da s
$u.\pi$ // predecessore di u: è il vertice v dal quale u è stato scoperto, esplorando l'arco (v, u)

Titolo nota 26/02/2016

```
BFS(G,s)   // G rappresentato con liste di adiacenza
 for ogni vertice v ∈ V \ {s} {
        v.color = Bianco
        v.d = ∞
        v.π = NIL
 }                                   } Θ(|V|)
 s.color = Grigio
 s.π = NIL
 s.d = 0
 Q = nuova Coda()
 Enqueue(Q, s);
 while (Q ≠ ∅) {
      u = Dequeue(Q)
      for ogni v ∈ Adj[u] {
          if (v.color == Bianco) {
```

Titolo nota 26/02/2016

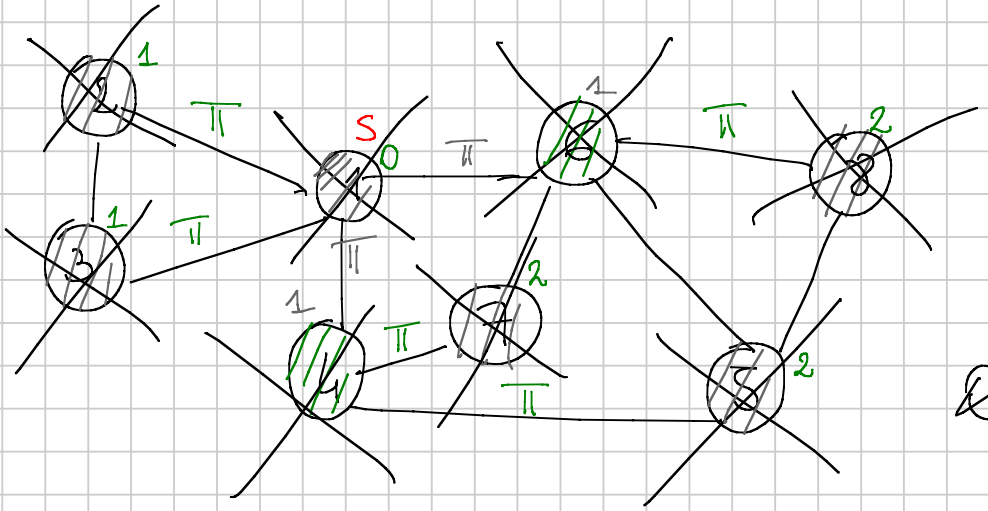
$\Theta(1)$ {

```
        v.color = Grigio;
        v.d = u.d + 1;
        v.π = u;
        Enqueue (Q, v);
    }
}
u.color = Nero;
}
```

$\Theta(|Adj[u]|)$ $O(|E|)$

ESEMPIO

Titolo nota 26/02/2016

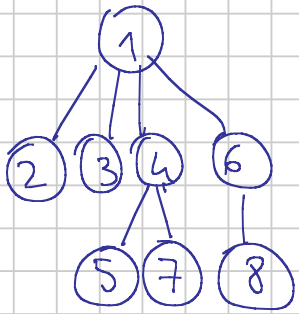


S = 1

Q = 1, 2, 3, 4, 6, 5, 7, 8

BFS: 1, 2, 3, 4, 6, 5, 7, 8

livello: 0 | 1 | 2



Albero BF

Titolo nota 26/02/2016

ALBERO BF

$G = (V, E)$

- costruito con la BFS
- è un sottografo di G (albero)

$$BF = (V_\pi, E_\pi)$$

$$V_\pi = \{ v \in V \mid v.\pi \neq NIL \} \cup \{s\}$$

$$E_\pi = \{ (v.\pi, v) \mid v \in V_\pi \setminus \{s\} \}$$

BF è connesso, $|E_\pi| = |V_\pi| - 1$ (CNES affinché BF sia un albero)

$\Rightarrow$

BF è un albero

PROPRIETÀ: $\forall v \in V_\pi$, il cammino $s \rightsquigarrow v$ nell'albero BF è un cammino minimo in G.

Titolo nota 26/02/2016

Analisi

1) i vertici entrano e escono dalla coda al più 1 volta (quando da Biondi diventano Grigi)
2) inserimenti e estrazioni dalla coda costano complessivamente $O(|V|)$
3) la lista $Adj(u)$ di un vertice u viene esaminata al più una volta (quando u è estratto dalla coda)
4) il costo del while è $O(|E|)$

↓

ogni arco è esaminato al più 2 volte se il grafo è non orientato, oppure 1 volta se il grafo è orientato.

Titolo nota 26/02/2016

$\sum_{v \in V} |Adj[v]| = \sum_{v \in V} \delta(v) =$ → $|E|$ se G è orientato

→ $2|E|$ se G è non orientato.

$$\Rightarrow T(|V|, |E|) = O(|V| + |E|)$$

$$= \Theta(|V| + |E|) \text{ se } G \text{ è connesso}$$

↳ lineare

BFS è un algoritmo ottimo

(dimensione dell'input)

Procedura PRINT-PATH

Titolo nota 26/02/2016

stampa i vertici di un cammino minimo $s \rightsquigarrow v$

```
Print-Path(G, s, v)        // G è già stato visitato BFS(G, s)
if (v == s) print s;
else if (v.π == NIL)       // v non è raggiungibile da s
        print "non ci sono cammini s ⇝ v"
else { PrintPath(G, s, v.π);
       print v;
}
```

$T(s, v) \rightarrow$ lineare nella lunghezza del cammino $s \rightsquigarrow v$ (quindi nella distanza(s, v))

Titolo nota 26/02/2016

"ESPLORAZIONE" di un Graf (es: graf del Web)

∀ vertice u: funzione Adj(u) che estrae dal contenuto di u le connessioni ai vertici adiacenti a u (es: url dei link nella pagina u)

```
BFS esplora (s)
Q = nuova Coda();
D = nuovo Dizionario(); // "marcatore" dei vertici visitati.
Enqueue(Q, s);
Insert(D, s);
while (Q ≠ ∅) {
    u = Dequeue(Q);
```

Titolo nota 26/02/2016

```
* esamina u *
for ogni v ∈ Adj(u) {
    if( search(D, v) == NIL) {   // v non era già stato scoperto
        Insert(D, v);
        Enqueue(Q, v);
    }
}
}
```

Titolo nota 26/02/2016

Visita in Profondità (DFS)

- procede in profondità a partire dai vertici incontrati
- visita tutto il grafo (selezionando eventualmente più di una sorgente)
- natura ricorsiva (la visita riparte dal nuovo vertice appena scoperto)
- costruisce una FORESTA DF (un albero DF per ogni componente connessa del grafo)

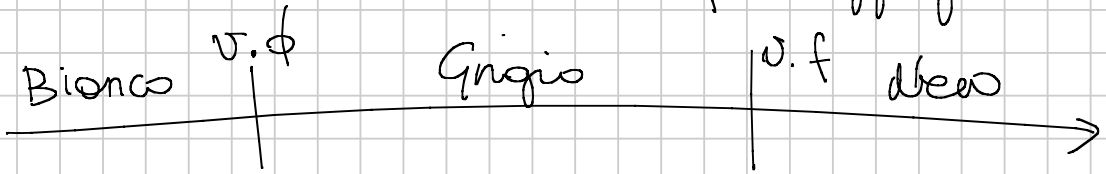
- NON calcola le distanze

Titolo nota 26/02/2016

→ $\forall v \in V$

$v.d \equiv$ momento in cui v è stato scoperto e colorato di grigio

$v.f \equiv$ momento in cui v diventa nero (tutto ciò di cui è raggiungibile da v è stato scoperto)



$$1 \leq v.d < v.f \leq 2|V|$$

Titolo nota 26/02/2016

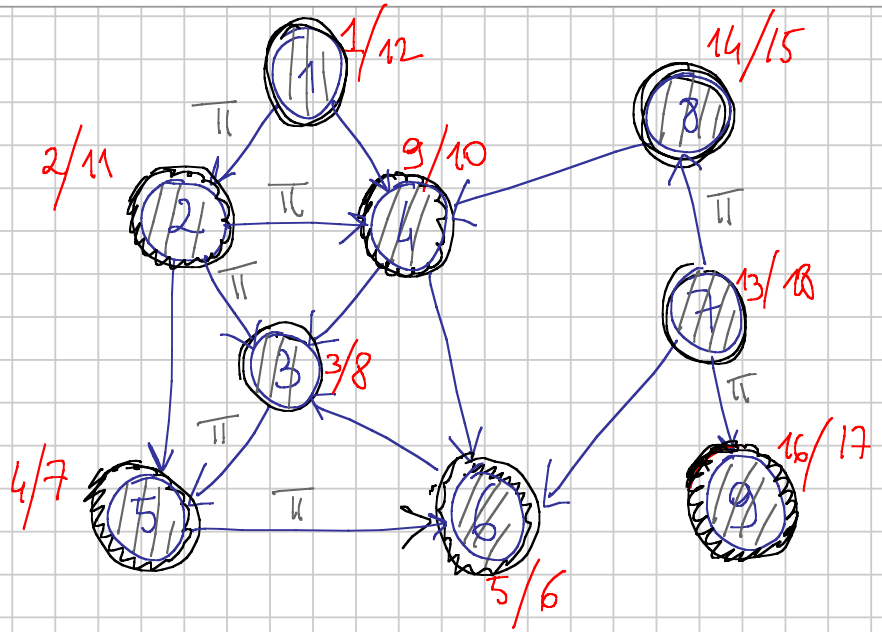
```
DFS(G)
for ogni v ∈ V {
        v.color = Bianco;
        v.π = NIL
}
time = 0;
for ogni vertice v ∈ V {
        if (v.color == Bianco) DFS-visit(G, v); // v è scelto come sorgente
}
```

Titolo nota 26/02/2016

```
DFS-visit (G, u)
time ++;
u.d = time;
u.color = Grigio;
for ogni v ∈ Adj[u] {   // ispezioniamo l'arco (u,v)
        if (v.color == Bianco) {
            v.π = u;
            DFS-visit (G, v);
        }
}
u.color = Nero;
time ++;
u.f = time
```

ESEMPIO

Titolo nota 26/02/2016



DFS : 1, 2, 3, 5, 6, 4, 7, 8, 9

(1 sorgente, 7 sorgente)

BFS: 1, 2, 4, 3, 5, 6

FORESTA DF