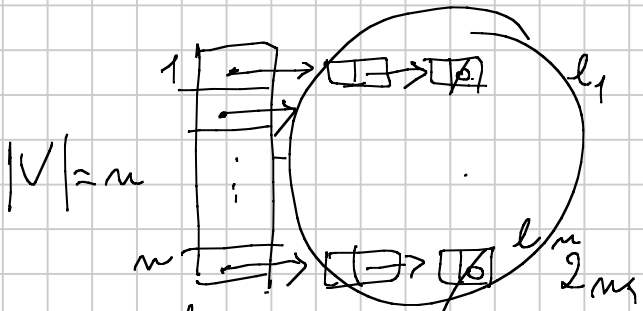
GRAFI — Visita in ampiezza BFS

Titolo nota 17/03/2016



$|V| = n$

grafo non orientato

lista di adiacenza

s

cicli

colore (basta una marca di 1 bit) se solo Visita BFS

π

d

spazio $= \Theta(n)$

nella coda vanno a finire tutti i vertici una sola volta

tempo $= \Theta(n+m)$

$\Theta(n) + \Theta(n+m) = \Theta(n+m)$

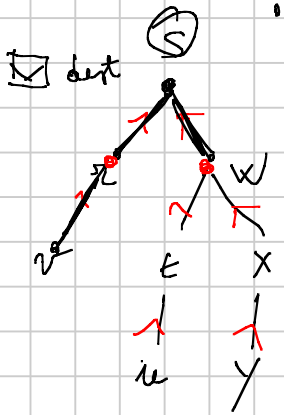
$O(n^2)$

n e m

$\sum_{i=1}^{n} l_i = 2m$

albero BFS = albero dei predecessori.

sequenza BFS = s, r, w, v, t, x, u, y



dest = y

π(nodo)

albero implicitamente memorizzato nei nodi di G

dest	next
v	r
t	w
u	w
x	w
y	w
r	r
w	w

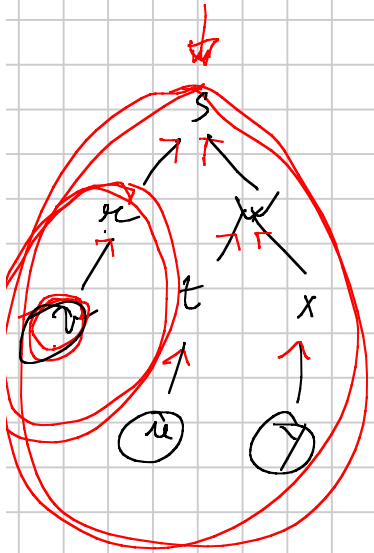
tutti i nodi raggiungibili da s

cammini minimi da s a tutti gli altri nodi

Calcolo i vertici del cammino da s a v

- BFS(s)

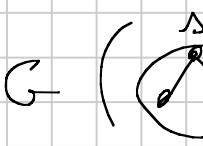
```
PRINT-PATH(G, s, v);
if (v == s) stampa s;
else if (v.π == NIL) print "non ci sono cammini"
else { PRINT-PATH(G, s, v.π);
       print v;
}
```

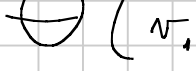


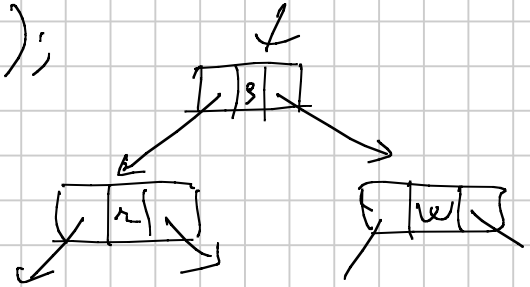
$O(n)$

\# nodo

1 puntatore







s, r, v

Grafo dinamico $\Theta(n+m)$ operazioni di SEARCH, INSERT nel dizionario + $\Theta(n)$ EN-DE

Titolo nota 17/03/2016

non possiamo usare le liste di adiacenza (struttura statica)

Soluzione: si inserisce il vertice in un Dizionario

Adj(u) = si costruisce al momento come fusione dei link presenti sulla pagina Web

n operazioni di EN-DE $= 2n$

$|V| = n$

$|E| = m$

```
BFS-Esplora (G, s):
Q = nuovaCoda();
D = nuovoDizionario;
→ ENQUEUE (Q, s);
→ inserisci (D, s);
```

```
while Q ≠ ∅ {
    u = DEQUEUE (Q);
    * esamino u* ;
    ∀ vertice v ∈ Adj(u) {
m       if (search (D, v) == NIL) {
n           ENQUEUE (Q, v);
n           INSERT (D, v);
    }}
```

BFS-ESPLORA $\Theta(n+m)$ operazioni di SEARCH e INSERT su D

D è una tabella hash

SEARCH + INSERT

$\Theta(1)$ in media

$\Downarrow$

$\Theta(n+m)$

D è Albero AVL

SEARCH + INSERT

$O(\log n)$

$\Downarrow$

$\Theta((n+m)\log n)$

Applicazioni cammini minimi

Titolo nota 17/03/2016

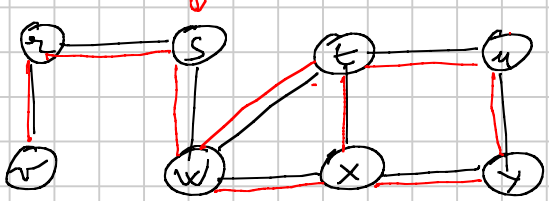
G è connesso ? Visita BFS;

G è non connesso se ∃ un vertice $\neq s$: $\Pi.v = NIL$ $\Theta(n)$

G è ciclico ?
(se contiene almeno un ciclo)

$\Rightarrow$ con visita ~~DFS~~

Eseguire una visita BFS che produca la sequenza di archi visitati

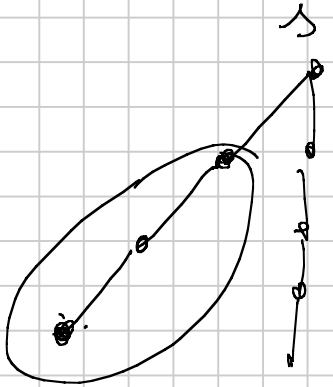


(s,r), (r,v), (s,w) (w,t) (w,x)
(t,u) (t,x) (u,y) (x,y)

Visita DFS - Visita in profondità.

Titolo nota 17/03/2016

(corrisponde alla visita anticipata in un albero



DFS (Comune, ----)

v ha un colore $\leq$ bianco / grigio / nero

π predecessore

$v.d$ = tempo inizio visita di v

$v.f$ = tempo fine visita del vertice v

time = globale

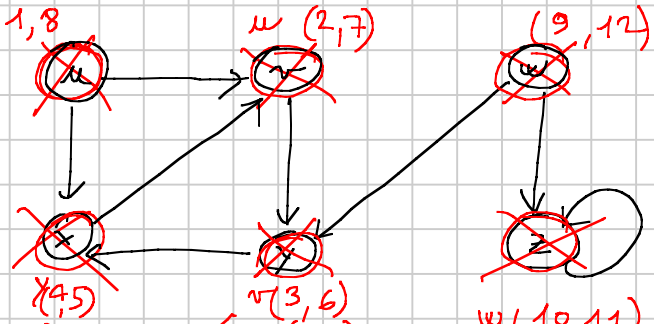
Visita Tutti i nodi di G anche se non raggiungibili.

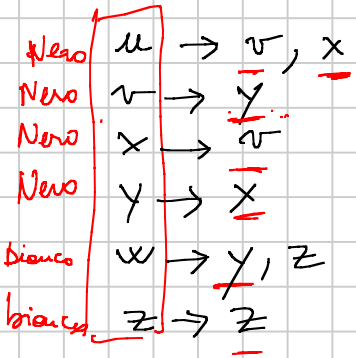
Titolo nota 17/03/2016

```
DFS(G);
for ogni vertice v ∈ G { u.color = bianco; u.π = NIL };
time = 0;
for ogni v ∈ G
    if (u.color == bianco)  DFS_Visit(G,u);
DFS_Visit(G,u);
time = time + 1; u.d = time; u.color = grigio;
for ogni v ∈ Adj(u) {
    if (v.color == bianco) {
        v.π = u;
        DFS_Visit(G,v);
}}
u.color = black;
time = time + 1;
u.f = time;
```

G orientato DFS

Titolo nota 17/03/2016





liste di adiacenza

$m + n$ spazio

time = ~~0~~ 1

~~2 3 4 5~~

~~6 7 8 9~~

~~10~~, 11, 12

DFS_~~Visit~~(u)

DFS_~~Visit~~(v)

DFS_~~Visit~~(y)

DFS_~~Visit~~(x)

DFS_~~Visit~~(w)

DFS_~~Visit~~(z)

Stampare gli archi della visita DFS

Complessità di DFS(G)

spazio. $O(n)$ ≡ profondità della pila di ricorsione

tempo = $\Theta(n + m)$

sequenza di visita dei vertici = u, v, y, x, w, z DFS