Algoritmi e Strutture Dati (Mod. B)

Algoritmi su grafi

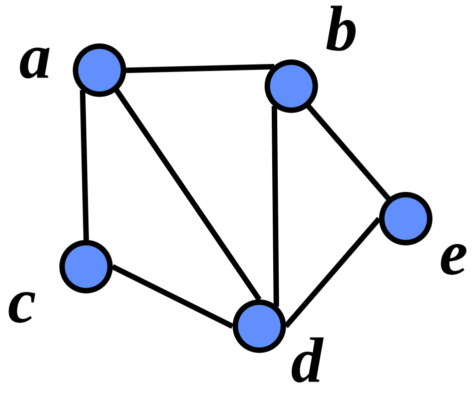
Ricerca in profondità

(Depth-First Search) Parte I

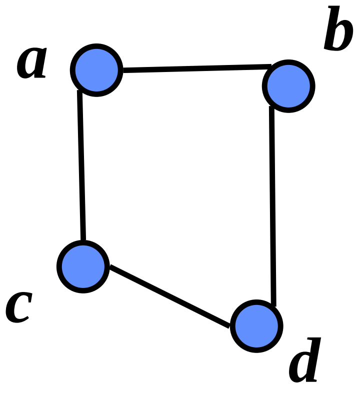
Sottografo di copertura

Un *sottografo* di $G=(V,E)$ è un grafo $H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$.

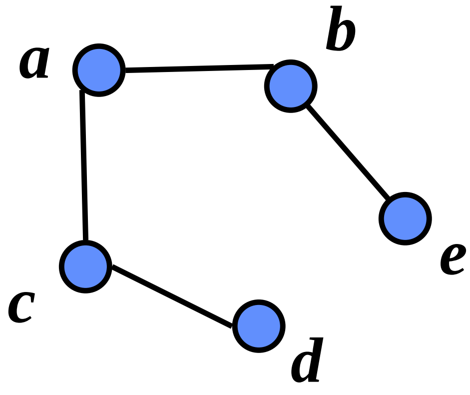
- *H'* è un *sottografo di copertura* (o *di supporto* o sotto-grafo *"spanning"*) di G se
  - $\boxed{V^* = V}$ e $E^* \subseteq E$



G



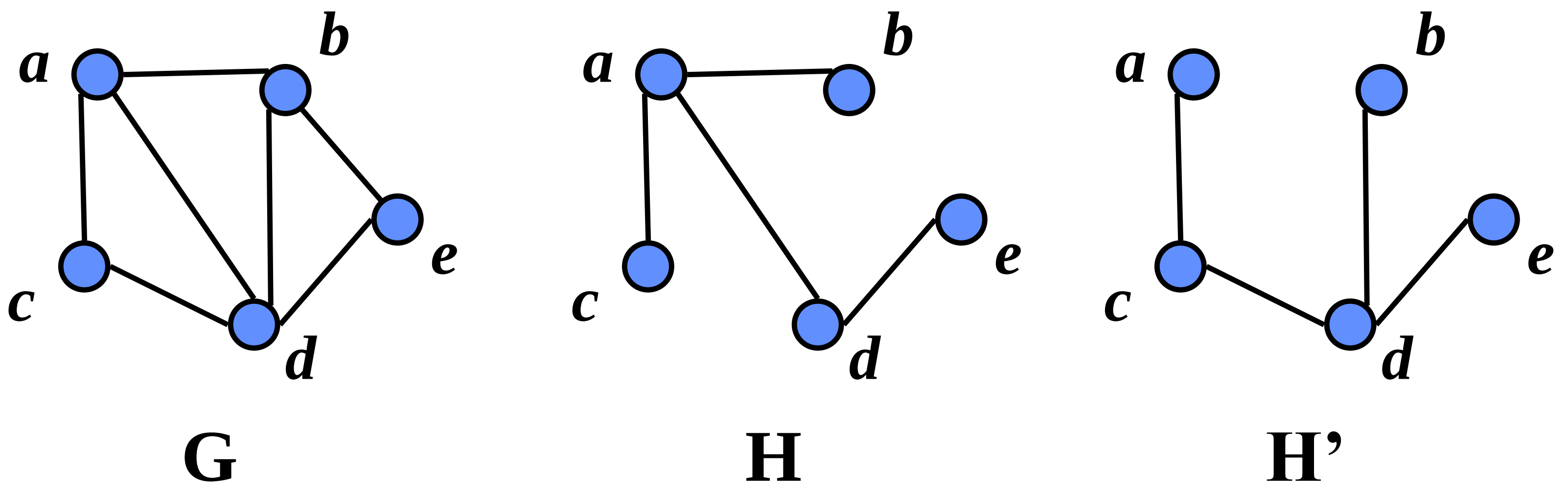
H



H'

Albero di copertura

- **Un grafo *H = (V*, E*)* è un *albero di copertura* (o *albero "spanning"*) *del grafo* *G=(V,E)* se**
  - ***H* è un *grafo di copertura* di *G***
  - ***H* è un *albero***



Visita in Profondità (DFS)

- **Tecnica di visita di un grafo**
  - **È una variazione della *visita in profondità* per alberi binari**
- **La visita di s procede come segue:**
  - **Si visitano ricorsivamente tutti i vertici adiacenti ad *s*;**
  - **Si termina la visita del vertice *s* e si ritorna.**
- **Bisogna evitare di rivisitare vertici già visitati**
  - **Bisogna anche qui evitare i cicli**
  - **Nuovamente, quando un vertice è stato scoperto e (poi) visitato viene marcato opportunamente (*colorandolo*)**

Algoritmo DFS

Manterremo traccia del *momento* (`tempo`) in cui ogni vertice *v* viene *scoperto* e del momento in cui viene *visitato* (o *terminato*).

Useremo inoltre due array $d[v]$ e $f[v]$ che registrano il momento in cui *v* verrà scoperto e quello in cui verrà visitato.

La variabile globale *tempo* serve a registrare il passaggio del tempo.

Il tempo viene usato per *studiare* le *proprietà* di *DFS*

DFS: intuizioni

I passi dell'algoritmo *DFS*

- **si parte da un vertice *non visitato s* e lo si visita**
- **si sceglie un vertice *non scoperto* adiacente ad *s*.**
- **da *s* si attraversa quindi un percorso di vertici adiacenti (visitandoli) finché possibile (*DFS-Visita*):**
  - **cioè finché non si incontra un vertice già scoperto/visitsto**
- **appena si resta "*bloccati*" (tutti gli archi da un vertice sono stati scoperti), si torna indietro (*backtracking*) di un passo (vertice) nel percorso attraversato (aggiornando il vertice *s* al vertice corrente dopo il passo all'indietro).**
- **si ripete il processo ripartendo dal passo.**

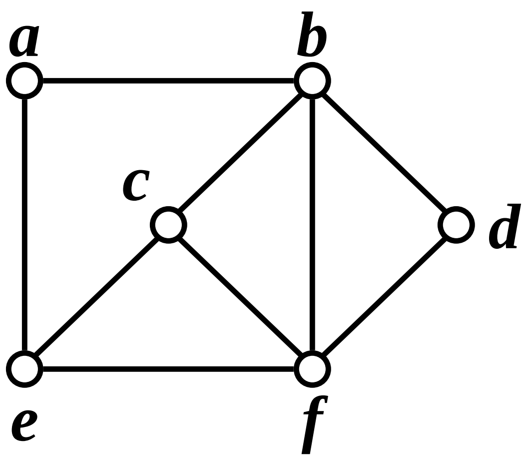
DFS: DFS-Visita

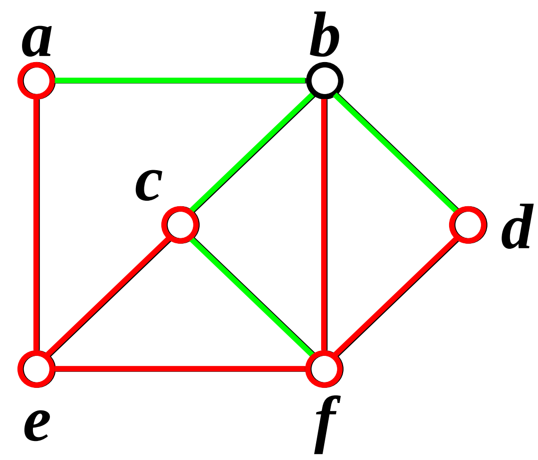
DFS-Visita*: algoritmo principale della *DFS

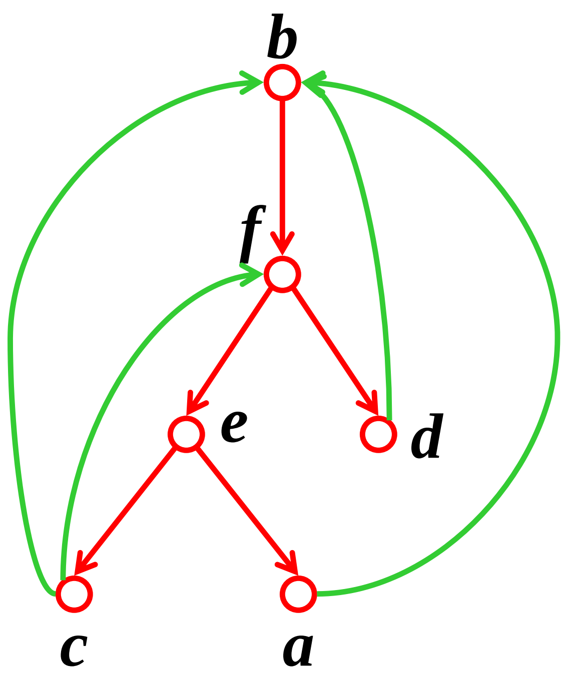
sia dato un vertice *u* di colore bianco in ingresso

- **visitare il vertice *u:* colorare *u* di grigio e assegnare il tempo di inizio visita *d*[*u*]**
- **visitare in *DFS ricorsivamente* ogni vertice bianco adiacente ad *u* con *DFS-Visita***
- **colorare di nero *u* e assegnare il tempo di fine visita *f*[*u*].**

Chiamata ricorsiva





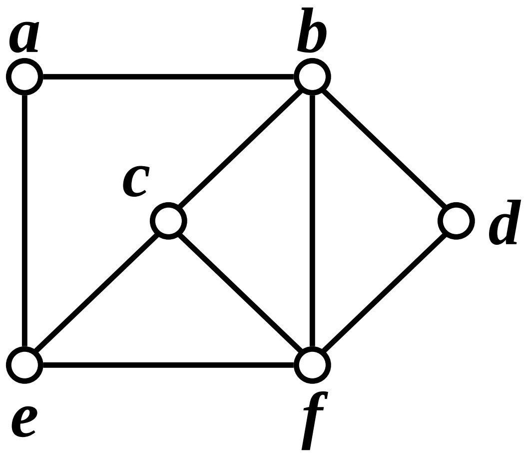


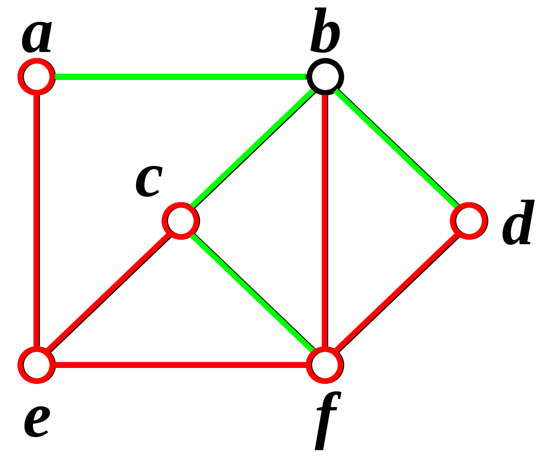
b f e c a d

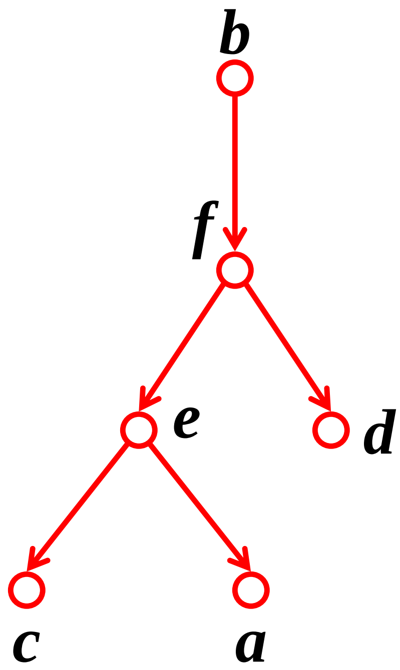
Albero di copertura Depth-first

Archi dell'albero

Archi di ritorno







b f e c a d

Albero di copertura è la Foresta Depth-First

Archi dell'albero

Algoritmo DFS

```
DSF(G:grafo)
   for each vertice u ∈ V
       do  colore[u] = Bianco
           pred[u] = NIL
   tempo = 0
   for each vertice u ∈ V
       do if  colore[u] = Bianco
           then  DFS-Visita(G,u)
```

```
DSF-Visita(G:grafo,u:vertice)
   colore[u] = Grigio
   d[u] = tempo = tempo + 1
   for each vertice v ∈ Adiac[u]
       do if colore[v] = Bianco
           then pred[v] = u
                DFS-Visit(G,v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```

Inizializzazione del grafo e della variabile tempo

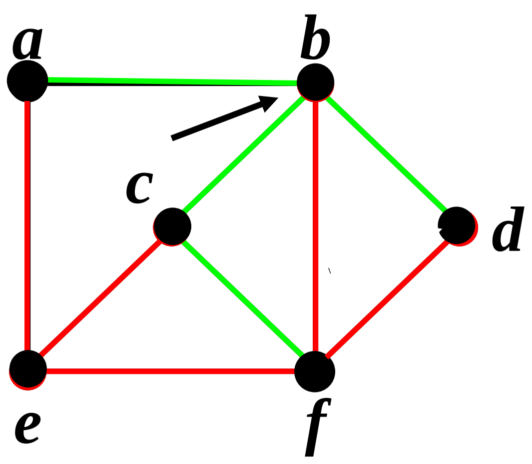
Abbreviazione per:
`tempo=tempo+1`
`d[u]=tempo`

Abbreviazione per:
`tempo=tempo+1`
`f[u]=tempo`

DFS: simulazone

```
DSF(G:grafo)
  for each vertice u ∈ V
      do  colore[u] = Bianco
          pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
      do if  colore[u] = Bianco
         then  DFS-Visita(G,u)
```

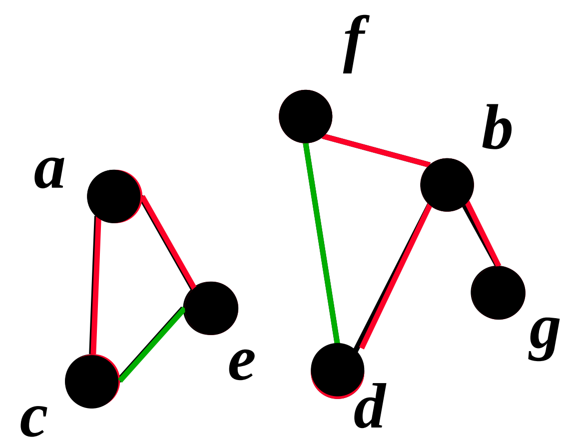
```
DSF-Visita(G:grafo,u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
     do if colore[v] = Bianco
        then pred[v] = u
             DFS-Visit(G,v)
 colore[u] = Nero
 f[u] = tempo = tempo + 1
```

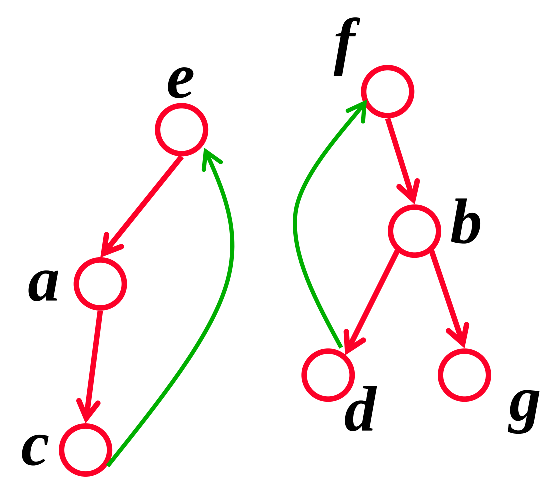


Alberi di copertura multipli

```
DSF(G:grafo)
  for each vertice u ∈ V
      do  colore[u] = Bianco
          pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
      do if  colore[u] = Bianco
         then  DFS-Visita(G,u)
```

```
DSF-Visita(G:grafo,u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
     do if colore[v] = Bianco
        then pred[v] = u
             DFS-Visit(G,v)
 colore[u] = Nero
 f[u] = tempo = tempo + 1
```





Tempo di esecuzione di DFS

```
DSF(G:grafo)
   for each vertice u ∈ V
          do  colore[u] = Bianco
              pred[u] = NIL
   tempo = 0
   for each vertice u ∈ V
          do if colore[u] = Bianco
             then DFS-Visita(G,u)
```

$\Theta(|V|)$

$\Theta(|V|)$

$|V|$ *volte*

```
DSF-Visita(G:grafo,u:vertice)
   colore[u] = Grigio
   d[u] = tempo = tempo + 1
   for each vertice v ∈ Adiac[u]
         do if colore[v] = Bianco
             then pred[v] = u
                  DFS-Visit(G,v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```

$\Theta(|E_u|)$

Chiamata solo per vertici non ancora visitati

Tempo di esecuzione di DFS

```
DSF(G:grafo)
  for each vertice u ∈ V
       do  colore[u] = Bianco
           pred[u] = NIL
   tempo = 0
  for each vertice u ∈ V
       do if  colore[u] = Bianco
           then  DFS-Visita(G,u)
```

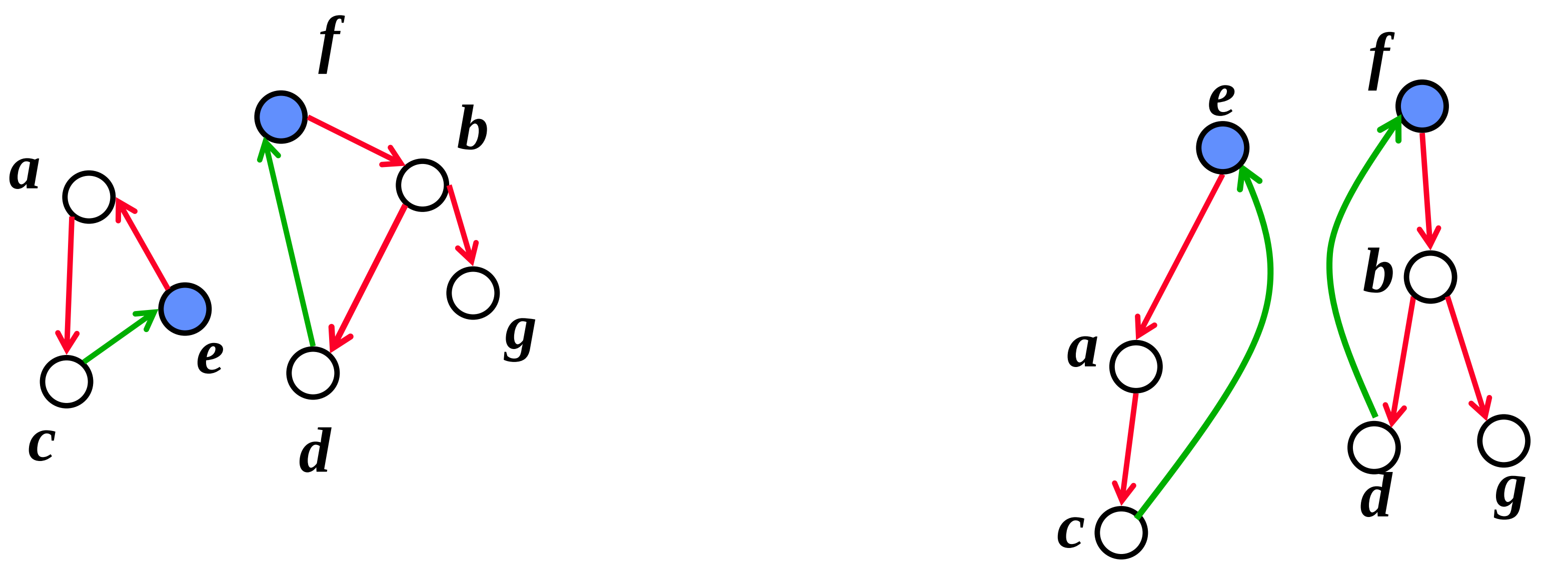
$\Theta(|V|+|E|)$

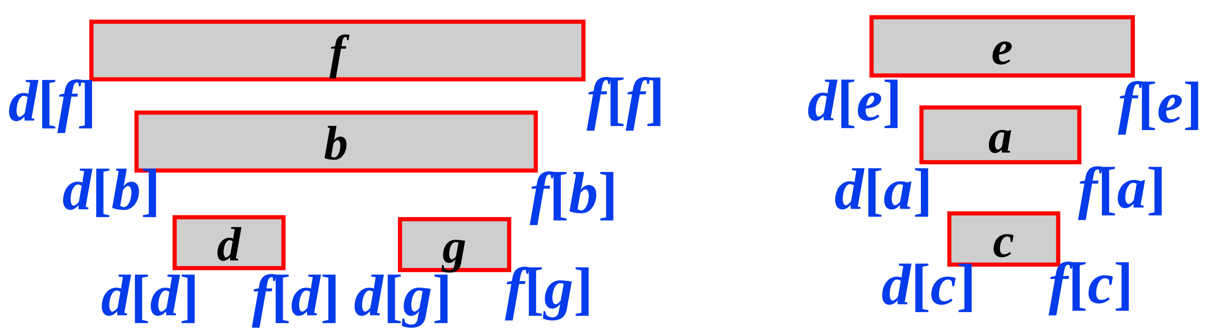
Proprietà di DFS: struttura a parentesi

Teorema: **In ogni DFS di un grafo *G*, per ogni coppia di vertici *u* e *v*, *una sola* delle condizioni seguenti vale:**

- **Gli intervalli [$d[u], f[u]$] e [$d[v], f[v]$] sono interamente disgiunti**
- **L'intervallo [$d[u], f[u]$] è interamente contenuto nell'intervallo [$d[v], f[v]$] e *u* è in *discendente* di *v* nell'albero DF.**
- **L'intervallo [$d[v], f[v]$] è interamente contenuto nell'intervallo [$d[u], f[u]$] e *v* è in *discendente* di *u* nell'albero DF.**

Struttura a parentesi: intuizione





Proprietà di DFS: struttura a parentesi

Dimostrazione: **Due sono i casi**

- $d[u] < d[v]$

  Due sottocasi:

  ① $d[v] < f[u]$. **Quindi** v **è scoperto mentre** u **è ancora grigio. Questo implica che** v **è *discendente* di** u **(*perché?*)**

  Inoltre, v **è stato scoperto più recentemente di (dopo)** u**; perciò la sua lista di archi uscenti viene esplorata, e** v **viene visitato (terminato) e a** $f[v]$ **assegnato un valore.**

  Quindi $[d[v], f[v]]$ **è totalmente incluso in** $[d[u], f[u]]$

  ② $f[u] < d[v]$. **Poiché** $d[u] < f[u]$**, segue che** $[d[u], f[u]]$ **e** $[d[v], f[v]]$ **sono totalmente disgiunti**

- $d[u] > d[v]$

Proprietà di DFS: struttura a parentesi

Dimostrazione: **Due sono i casi**

- $d[u] < d[v]$ ✓
- $d[u] > d[v]$

  Due sottocasi: il ragionamento è simile a prima ma con i ruoli di u **e** v **invertiti**

  ① $d[u] < f[v]$.

  Risulta che $[d[u], f[u]]$ **è completamente incluso in** $[d[v], f[v]]$ **e** ***u discendente*** **di** v

  ② $f[v] < d[u]$.

  Poiché $d[u] < f[u]$, **segue che** $[d[v], f[v]]$ **e** $[d[u], f[u]]$ **sono totalmente disgiunti (e in due sottoalberi distinti)**

Proprietà di DFS: struttura a parentesi

Corollario: **Un vertice** ***v*** **è un** ***discendente*** **di** ***u*** **nella** ***foresta DF*** **di un grafo** ***G*** **se e solo se**

$$d[u] < d[v] < f[v] < f[u].$$

Dimostrazione: **Immediata conseguenza del teorema precedente.**

Proprietà di DFS: percorso bianco

Teorema: **Nella *foresta DF* di un grafo *G*, un vertice *v* è *discendente* del vertice *u* <u>*se e solo se*</u> al tempo *d*[*u*] in cui la ricerca visita *u*, il vertice *v* può essere raggiunto da *u* lungo un *percorso composto* da *soli vertici bianchi*.**

Dimostrazione:

solo se: **Assumiamo che *v* sia discendente di *u* nella *foresta DF* e che *w* sia un arbitrario vertice nel percorso tra *u* e *v* nella *foresta DF*.**

Allora anche *w* è discendente di *u* nella *foresta DF*.

Per il corollario precedente, $d[u] < d[w]$, quindi *w* è bianco al tempo $d[u]$

Proprietà di DFS: percorso bianco

Teorema: **Nella *foresta DF* di un grafo *G*, un vertice *v* è *discendente* del vertice *u* *se e solo se* al tempo *d*[*u*] in cui la ricerca visita *u*, il vertice *v* può essere raggiunto da *u* lungo un *percorso composto* da *soli vertici bianchi*.**

Dimostrazione:

se: **Assumiamo che *v* sia il *primo* vertice *raggiungibile* da *u* *lungo un percorso bianco* al tempo *d*[*u*], ma che non diventi un discendente di *u* nell'*albero DF*.**

Quindi tutti i vertici che precedono *v* nel percorso saranno discendenti di *u*.

Sia *w* il predecessore di *v* nel percorso (*v* è quindi adiacente a *w*).

Proprietà di DFS: percorso bianco

Teorema: **Nella *foresta DF* di un grafo *G*, un vertice *v* è *discendente* del vertice *u* *se e solo se* al tempo *d*[*u*] in cui la ricerca visita *u*, il vertice *v* può essere raggiunto da *u* lungo un *percorso composto* da *soli vertici bianchi*.**

Dimostrazione:

se: **per il *Corollario precedente*, abbiamo che *f*[*w*]<*f*[*u*].**

Poiché *v*∈*Adiac*[*w*], la chiamata a *DFS-Visita(w)* garanti-sce che *v* venga visitato (e *terminato*) prima di *w*.

Perciò, *f*[*v*]<*f*[*w*]<*f*[*u*].

Poiché quindi *v* è *bianco* al tempo *d*[*u*], vale *d*[*u*]< *d*[*v*],

e il *Corollario precedente* garantisce che *v* deve essere un discendente di *u* nell'*albero DF*.