GRAFI: ORDINAMENTO TOPOLOGICO

[Deme, seconda edizione] cap. 14

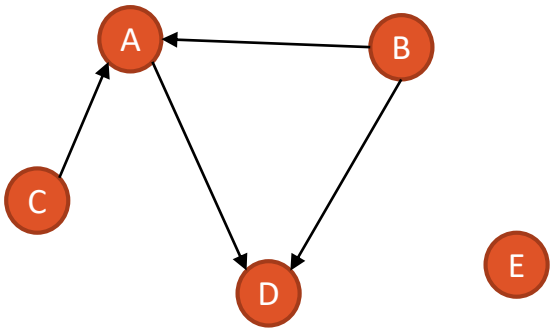
Sezione 14.4 fino a 14.4.1 inclusa

(dall'introduzione ai grafi) DAG

Un grafo **orientato aciclico** si chiama **DAG** (**D**irected **A**cyclic **G**raph). I DAG si usano spesso per rappresentare relazioni di precedenza tra processi da eseguire.

In particolare, dato un DAG, è sempre possibile ordinare i nodi in un **ordine topologico**, cioè in modo che non ci sia nessun arco all'indietro nell'ordinamento.

Questo grafo è un DAG



Relazioni d'ordine parziale

Una relazione di **ordine parziale** (su un insieme I) è definita come una relazione binaria $\leq$,

Riflessiva $x \leq x \; \forall x \in I$

Antisimmetrica $x \leq y \wedge y \leq x \Rightarrow x = y \qquad \forall x, y \in I$

Transitiva $x \leq y \wedge y \leq z \Rightarrow x \leq z \qquad \forall x, y, z \in I$

La coppia $(I, \leq)$ si definisce insieme **parzialmente ordinato**.

Relazione di raggiungibilità (nei DAG)

In un DAG, la relazione di **raggiungibilità** (è raggiungibile da) è una **relazione d'ordine parziale**.

È **riflessiva**: per definizione, ogni vertice è raggiungibile da se stesso con un cammino degenere di lunghezza 0.

È **antisimmetrica**: se v è raggiungibile da u e u è raggiungibile da v, allora v e u sono coincidenti (cioè non vi può essere una coppia di nodi distinti mutuamente raggiungibili).

È **transitiva**: se v è raggiungibile da u e w è raggiungibile da v, allora w è raggiungibile da u, percorrendo (in sequenza) il cammino da u a v e da v a w.

Relazioni d'ordine totale (lineare)

Una relazione d'ordine parziale può venire completata (generalmente in più modi) in una relazione d'**ordine totale**.

Una relazione d'ordine totale ($<$) (($I, <$) è un insieme totalmente ordinato) è una relazione

Riflessiva

Antisimmetrica

Transitiva

Totale $\forall x, y \in I, \quad x < y \vee y < x$

Ordine topologico

Dato un **grafo orientato aciclico**, un **ordine topologico** è un ordine lineare dei suoi nodi tale che se nel grafo vi è un arco (u, v), allora u precede v nell'ordine; cioè:

Definizione 1.

ordine topologico su un grafo orientato aciclico G = (V, E) è una relazione d'ordine totale $<$ sull'insieme V dei nodi, tale che

$$\langle u, v \rangle \in E \Rightarrow u < v$$

Equivalentemente, è un ordine lineare dei suoi nodi tale che se nel grafo vi è un cammino da u a v, allora u precede v in tale ordine; cioè:

Definizione 2.

ordine topologico su un grafo orientato aciclico G = (V, E) è:
una relazione d'ordine totale $<$ sull'insieme V dei nodi, tale che

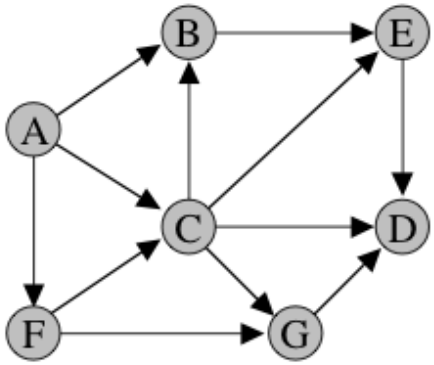
$$u \to v \Rightarrow u < v$$

(esiste un cammino da u a v $\Rightarrow$ u $<$ v)

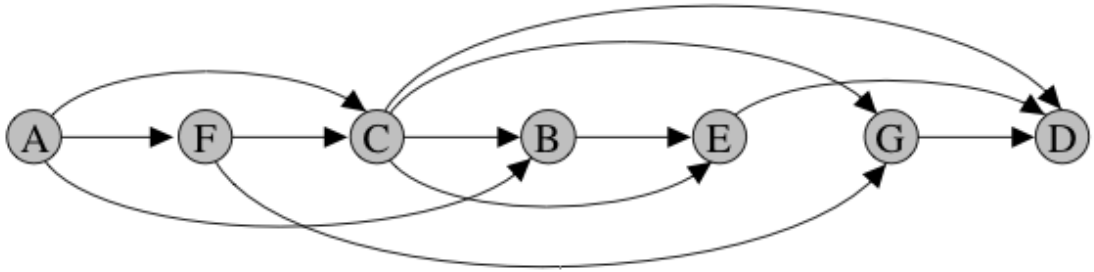
Proprietà ed esempi

Un grafo orientato aciclico **possiede sempre un ordine topologico**.

Un DAG **può possedere diversi** ordini topologici.



(a)

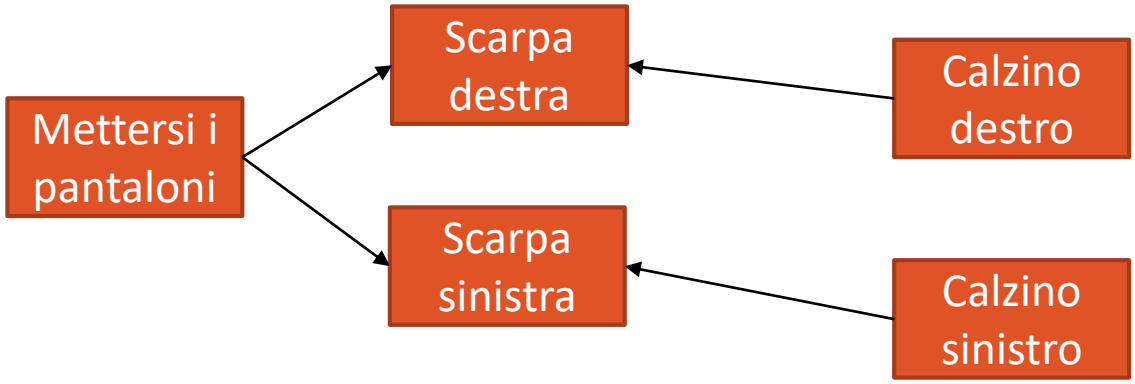


(b)

(a) Un grafo aciclico; **(b) Ordine topologico dei nodi del grafo (a)**

Esempio (motivante)

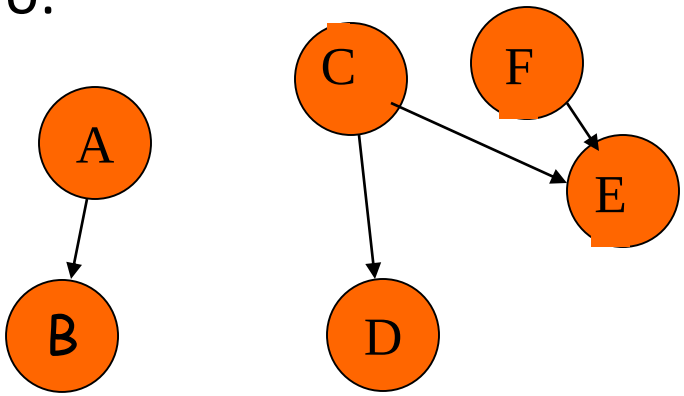
DAG che rappresenta **precedenza tra processi** (non eseguibili in parallelo)



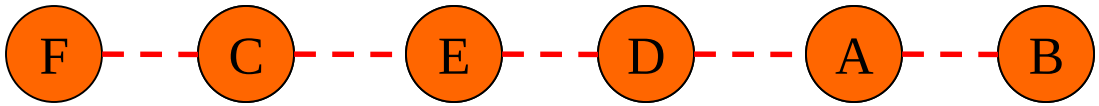
Ordinamento topologico

Esempio (1/2)

Dato il grafo:

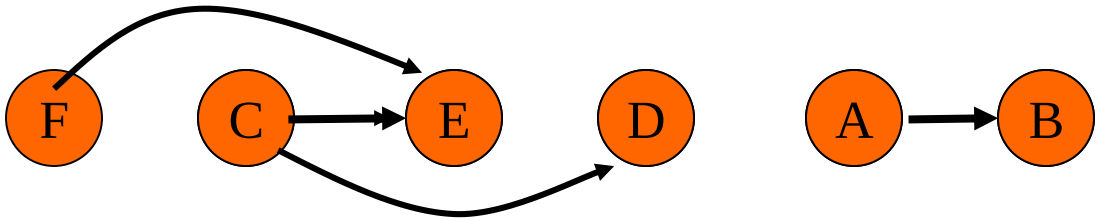


l'ordine:

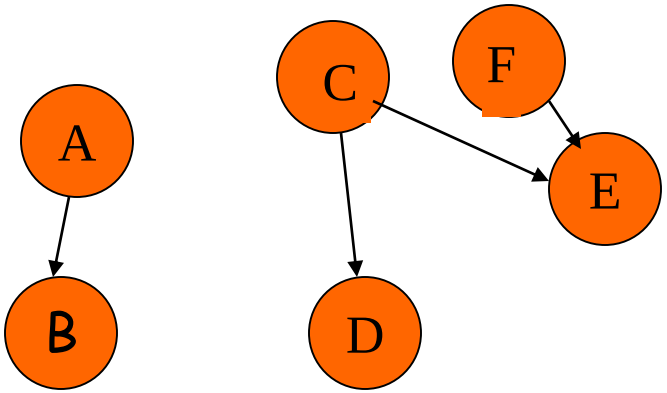


è un ordine topologico

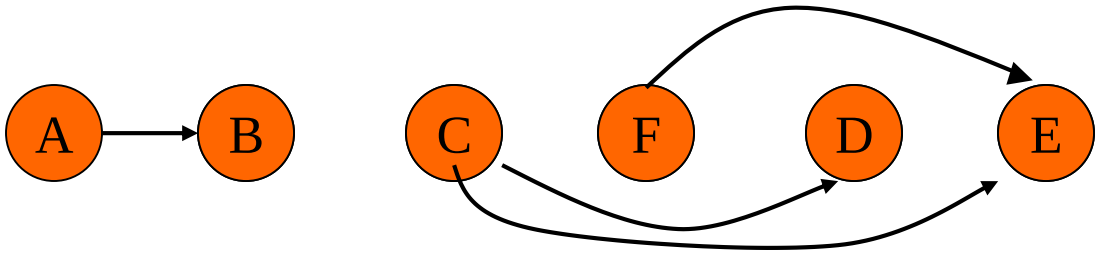
Infatti disegnando gli archi del grafo essi risultano **tutti orientati nello stesso verso** (da sinistra verso destra):

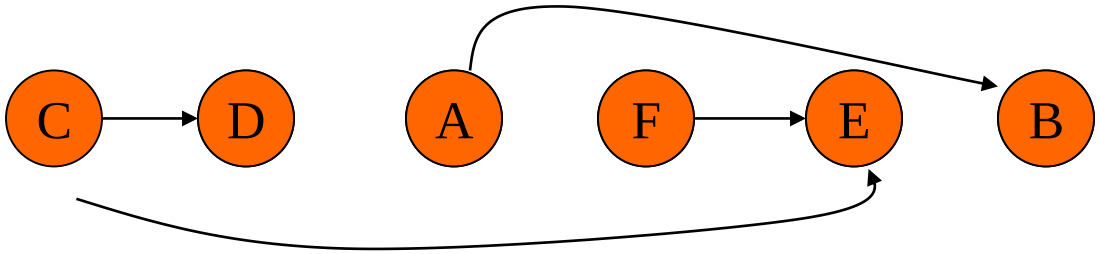


Esempio (2/2)



Ma non è l'unico, anche i seguenti sono ordini topologici





Dato un DAG, come trovare un ordinamento topologico?

Vediamo due metodi:

- Uno semplice, basato su un'intuizione
- Uno più complesso, ma che risolve il problema in tempo **lineare**

Sorgenti e Pozzi

Per comodità, diamo le seguenti due definizioni:

nodo sorgente: nodo che non ha archi entranti;

nodo pozzo: nodo che non ha archi uscenti.

Proprietà. In un grafo orientato aciclico vi sono **almeno un nodo pozzo e almeno un nodo sorgente**.

(ovvio, perché in un insieme parzialmente ordinato costituito da un numero finito di elementi vi sono sicuramente sia elementi minimali che elementi massimali).

Un algoritmo astratto di ordinamento topologico

1. Un qualunque nodo **sorgente** può essere scelto come primo elemento in un ordine topologico.
2. Se **rimuoviamo** tale nodo e tutti gli archi da esso uscenti, un qualunque nodo sorgente nel grafo così ottenuto può essere scelto come secondo elemento.

E così via.

ORDINAMENTO-TOPOLOGICO **(G)**

```
G' <- fai una copia di G
ord <- lista vuota di vertici
while esiste u senza archi entranti in G'
    appendi u come ultimo elemento di ord
    rimuovi da G' u e tutti i suoi archi uscenti
if G' non è vuoto then errore il grafo non è aciclico
else return ord
```

Correttezza algoritmo astratto

Dimostriamo che **ord è un ordinamento topologico di G**

Denotiamo con ord_i e G'_i il contenuto di ord e G' all'iterazione i-esima

Per ogni i, ord_i e V'_i (l'insieme dei vertici di G'_i) formano una partizione dei vertici di G

Dimostriamo per **induzione** su i che

INV: ad ogni istante del ciclo **non può esserci nessun cammino in G che porta da un vertice in G'_i ad uno in ord_i** (quindi «all'indietro» nell'ordinamento topologico)

PASSO BASE i = 0

$\mathrm{ord}_0 = \emptyset$ e $G'_0 = G$. la condizione (non esiste nessun cammino G'_0 a ord_0) è banalmente verificata perché ord_0 non contiene vertici.

Correttezza algoritmo astratto - II

PASSO INDUTTIVO: ad un istante k, non c'è nessun cammino in G dai vertici in G'_k ai vertici in ord_k ⇒ ad un istante k+1, non c'è nessun cammino in G dai vertici in G'_{k+1} ai vertici in ord_{k+1}

Al passo k-esimo, scegliamo un vertice u senza archi entranti in G'_k.

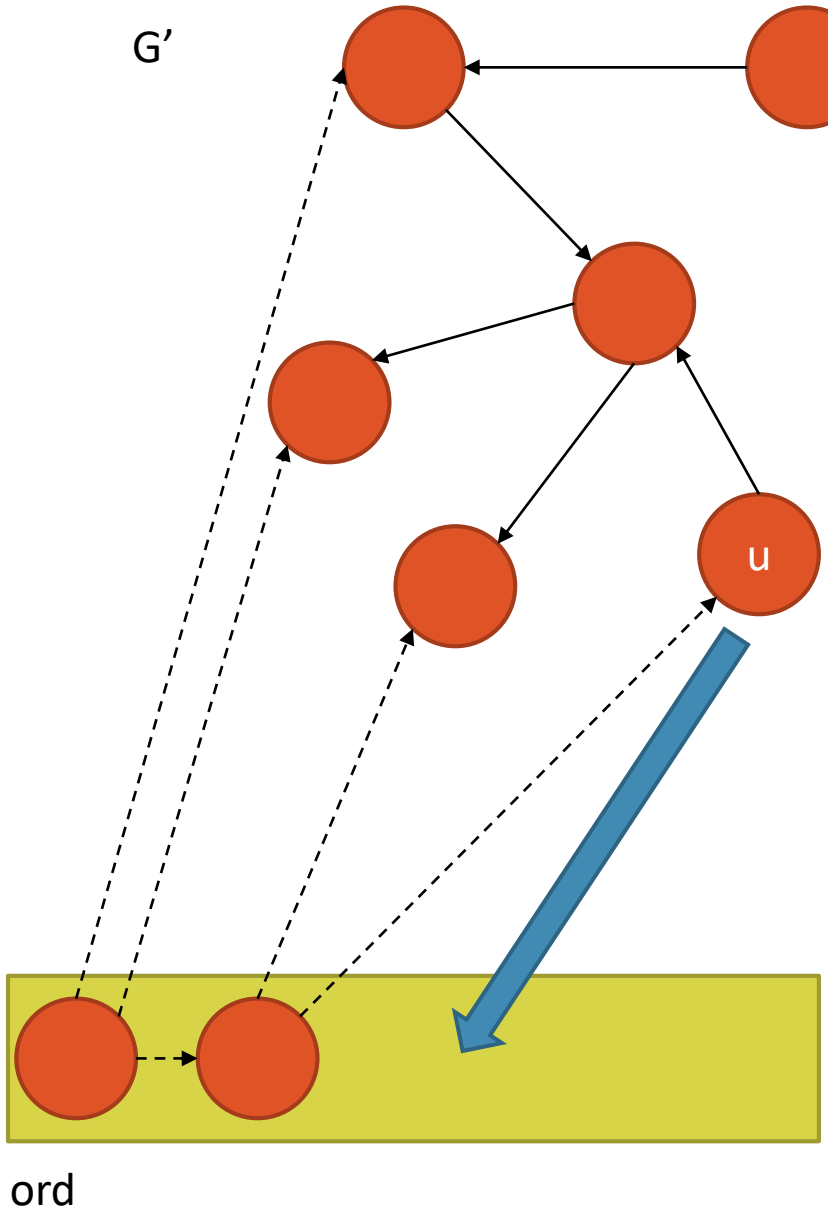
Per come è stato scelto u, non ci sono cammini in G che raggiungono u passando solo per vertici in G'_k.

L'unico modo per raggiungere u, potrebbe essere passando da un vertice in ord_k (ma partendo comunque da G'_k).

Ma questo non è possibile per ipotesi induttiva (perché i vertici in ord_k non sono raggiungibili da G'_k).

Di conseguenza, aggiungendo u ad ord_k all'istante k, all'istante k+1 **non c'è nessun cammino in G dai vertici in G'_{k+1} ai vertici in ord_{k+1}**

CVD.



Se sposto u in ord non può esistere nessun cammino che porta da un vertice in G’ ad u (o comunque, ad uno in ord).

Complessità dell'algoritmo semplice

Copia di G: O(n+m)

Numero di cicli: n

Verifica condizione del while: O(m) (possibili ottimizzazioni)

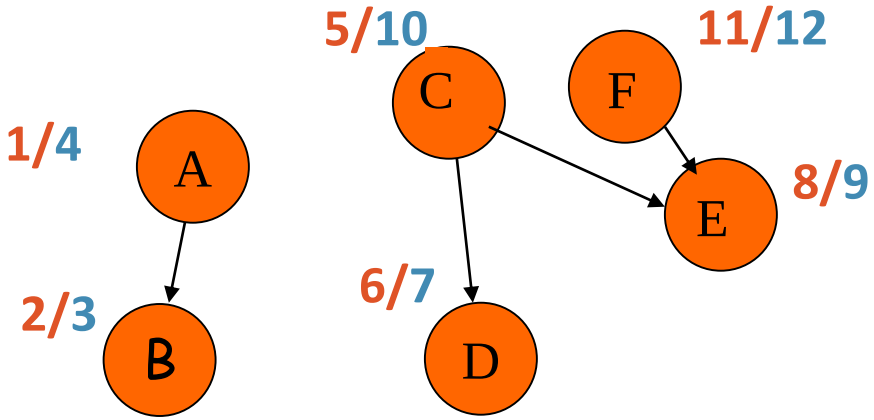
Rimuovi nodo O(m)

TOT: **O(m*n)**

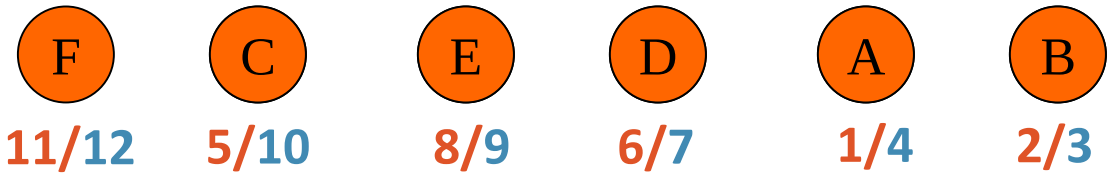
Ora vediamo un algoritmo più efficiente

Una proprietà della visita in profondità

Visitando il grafo **in profondità** considerando i vertici in ordine alfabetico: A B C D E F si ottengono i seguenti tempi di inizio e fine visita:

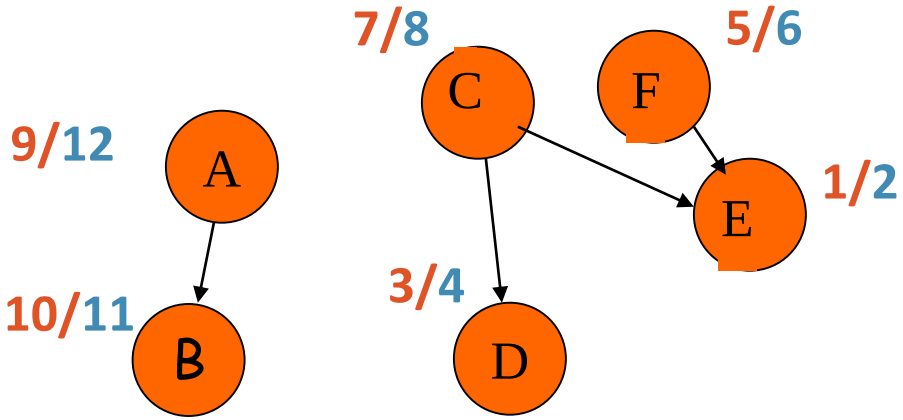


Che riportati in un ordinamento topologico sono

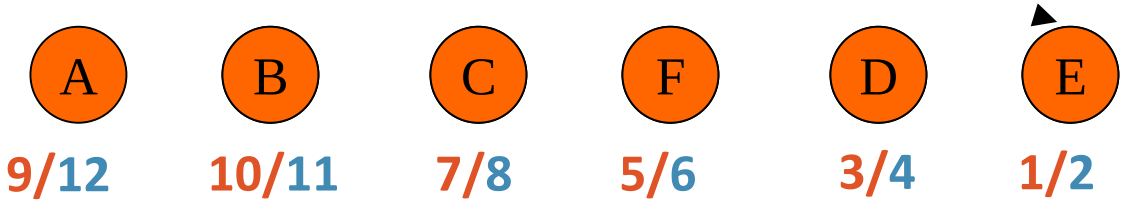


Una proprietà della visita in profondità - II

Facciamo la stessa cosa per gli altri due esempi

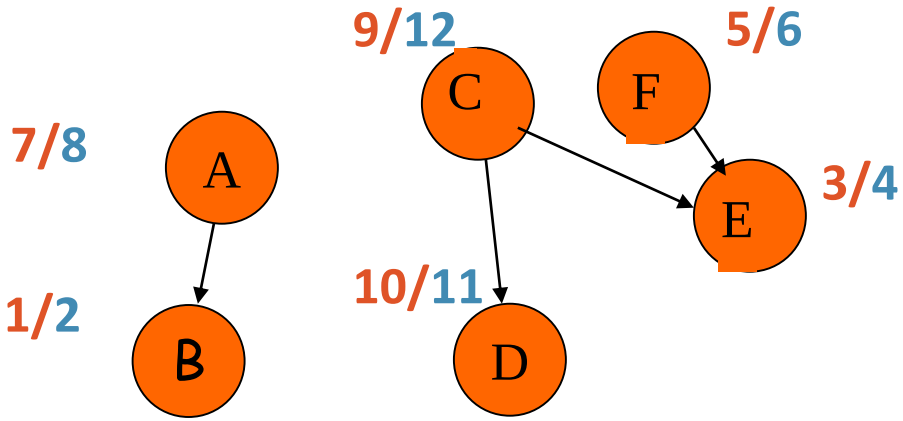


Che riportati in un ordinamento topologico sono

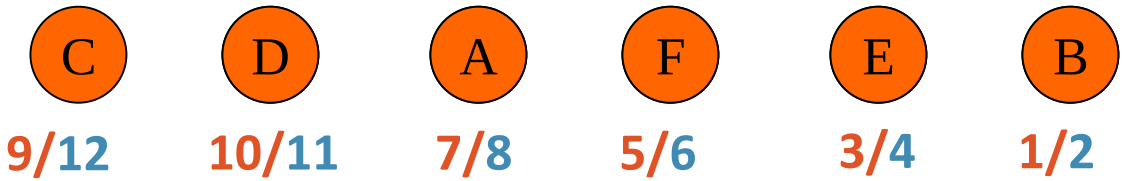


Una proprietà della visita in profondità - III

Facciamo la stessa cosa per gli altri due esempi

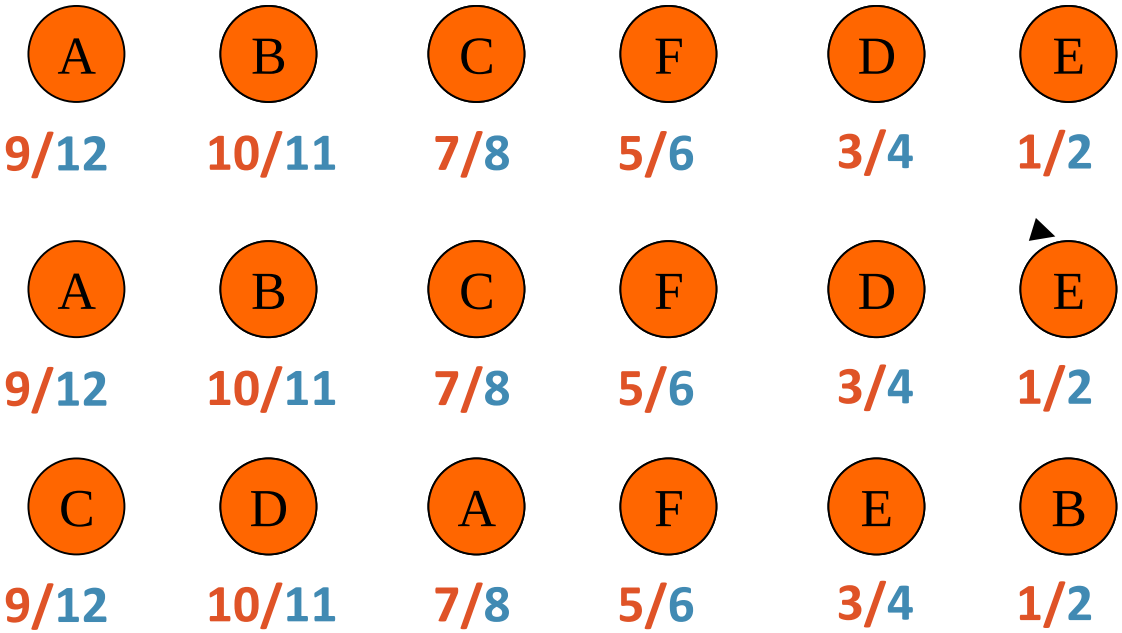


Che riportati in un ordinamento topologico sono



Una proprietà della visita in profondità - IV

Che cosa hanno in comune i tre ordinamenti?

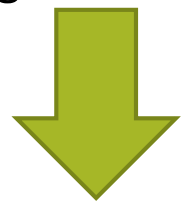


I vertici sono sempre in **ordine decrescente dei tempi di fine visita**

Osservazione

E’ vero che ogni ordinamento dei vertici di un DAG secondo i tempi di fine visita di una qualunque visita DFS del grafo è un ordinamento topologico?

Se la risposta fosse affermativa, si potrebbe ottenere un algoritmo per calcolare un ordinamento topologico con una semplice modifica alla struttura standard dell’algoritmo di **visita DFS**.



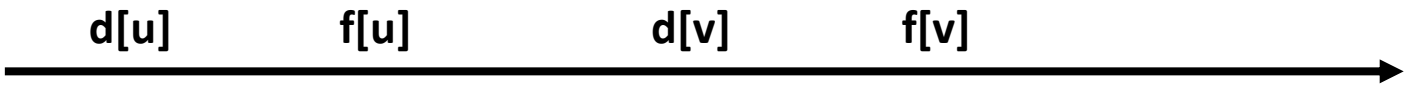
Basterebbe creare una **lista dei vertici in ordine decrescente dei tempi di fine visita**.

Teorema dell'ordinamento topologico

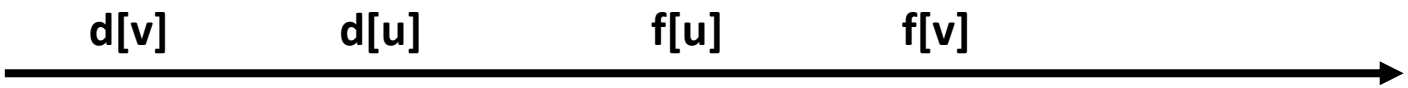
Teorema. Una (qualunque) DFS di un grafo orientato aciclico associa ai vertici tempi di fine visita tali che:

f[v] < f[u] per ogni arco <u, v> del grafo.

Dimostrazione. Supponiamo **per assurdo** che per almeno un arco <u, v> si abbia **f[u] < f[v]** (f[v] = f[u] è ovviamente impossibile). Abbiamo 2 possibilità:



Impossibile perché u non può diventare nero prima che v diventi grigio, ossia prima che tutti i suoi adiacenti siano stati scoperti.



Impossibile perché u sarebbe discendente di v in un albero della foresta e l'arco <u, v> sarebbe un arco all'indietro, ma G è un grafo aciclico.

CVD.

Modifichiamo una visita DFS per ottenere un ordinamento topologico

Sfruttando il teorema dell'ordinamento topologico, possiamo modificare una visita DFS in maniera che mantenga **la «storia» dei vertici chiusi**, in ordine inverso a quello di chiusura (da quello chiuso più recentemente a quello chiuso più anticamente).

Manteniamo un insieme ordinato (una lista o un vettore) ed inseriamo in testa i vertici man mano che vengono chiusi (cioè quando diventano neri).

Alla fine della visita, l'insieme ordinato conterrà l'ordinamento topologico del grafo G.

Tecnicamente, possiamo usare una lista, inserendo in testa i vertici chiusi, o un vettore, trattandolo come uno stack. Nell'algoritmo di seguito abbiamo usato quest'ultima soluzione.

Algoritmo di ordinamento topologico basato su DFS.

```
TOPOLOGICAL-SORT (G)
    INIZIALIZZA (G)
    ord <- un vettore di lunghezza n
    t <- n-1
    for ogni u ∈ V do
        if color [u] = white
        then DFS-TOPOLOGICAL (G, u, ord, t)
    return ord

DFS-TOPOLOGICAL (G, u, ord, t)
  color [u] <- gray
  d[u] <- time <- time+1
  for ogni v adiacente ad u
    if color[v] = white
      π[v] <- u
      DFS-TOPOLOGICAL (G, v, ord, t)
  color[u] <- black
  f[u] <- time <- time+1
  ord[t] <- u
  t--
```

Attenzione: in questo algoritmo non c'è un controllo che il grafo sia un DAG, ma ci vuole.

Complessità dell'ordinamento topologico

Abbiamo dimostrato che l'ordinamento topologico si può eseguire con una visita in profondità sul grafo G.

Di conseguenza, la complessità è la stessa di quella della visita DFS, che è **O(n+m)**.

Cosa devo aver capito fino ad ora

- Cos'è un DAG
- Cos'è un ordinamento topologico su un DAG
- Algoritmo astratto di ordinamento topologico
- Cosa accomuna un ordinamento topologico ed un albero di visita DFS
- Teorema dell'ordinamento topologico
- Algoritmo per l'ordinamento topologico basato su DFS

…se non ho capito qualcosa

- Alzo la mano e chiedo
- Ripasso sul libro
- Chiedo aiuto sul forum
- Chiedo o mando una mail al docente