Grafi

Un grafo G è una coppia (V, E) dove

- V è un insieme di vertici(nodi)
- E un insieme di coppie di vertici(archi,spigoli):
  ogni arco connette due vertici

Sostanzialmente V rappresenta un insieme di oggetti mentre E la relazione tra essi

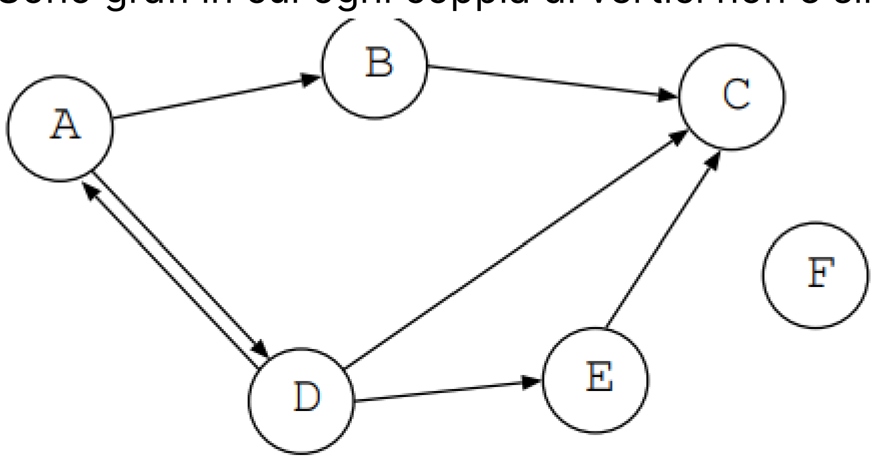
Esistono due tipi di grafi:

- orientati
- non orientati

Terminologia

Grafi orientati

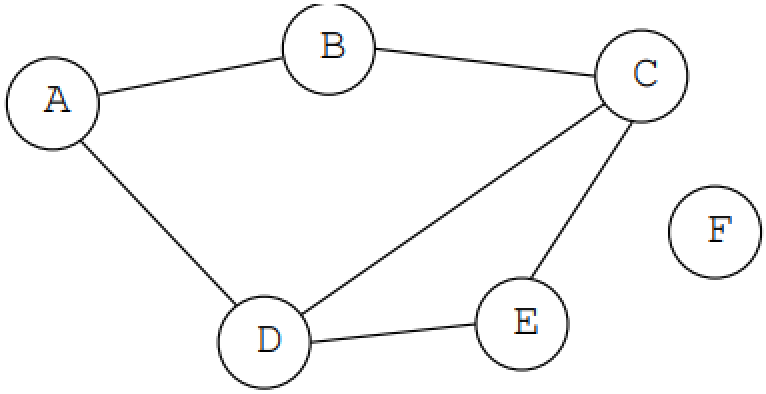
Sono grafi in cui ogni coppia di vertici non è simmetrica.



(A,D) e (D,A) denotano due archi diversi dunque i due vertici non sono uguali

Grafi non orientati

Sono grafi in cui ogni coppia di vertici è simmetrica.



(A,D) e (D,A) denotano lo stesso arco dunque i vertici sono uguali

Incidenza/Adiacenza

Un vertice (X, Y) è **incidente** da X a Y
(A,B) è incidente da A a B

Inoltre un vertice X si dice adiacente(*neighbour*) a Y solo se $(Y, X) \in E$.
In una grafo non orientato la relazione di adiacenza è simmetrica.

Nella figura del grafo orientato, B è adiacente ad A,ma non il contrario.
Nella figura del grafo non orientato invece, B è adiacente ad A e viceversa

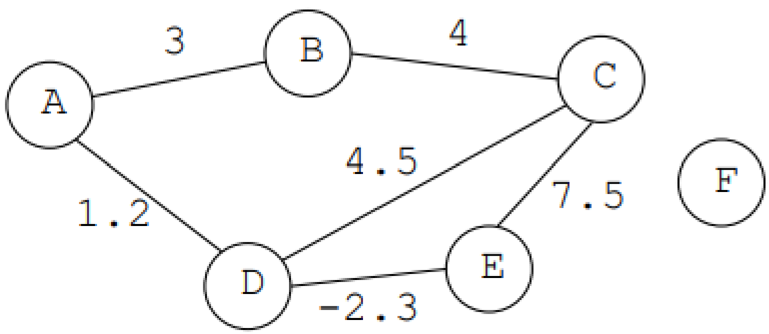
Grado

- Grafo **non orientato**:
  Il grado di un vertice è il numero di archi che da esso partono.
- Grafo **orientato**:
  - Grado **entrante** di un vertice:
    numero di archi che "entrano" in esso
  - Grado **uscente** di un vertice:
    numero di archi che escono da esso

Peso

Un grafo è pesato se associamo a G anche la funzione peso W.
Dunque otteniamo (G, W) dove:

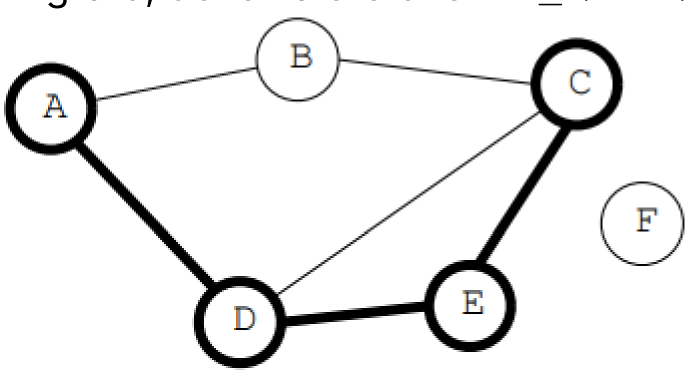
- G è un grafo
- W è la funzionepeso: $W : E \rightarrow R$ con R che indica l'insieme dei numeri reali.



$W((A, B)) = 3, W((C, F)) = \infty$

Sottografo

Dato $G = (V, E)$ un grafo, definiamo il sottografo H come
$H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$.
Essendo H un grafo, deve valere che $E^* \subseteq V^* \times V^*$

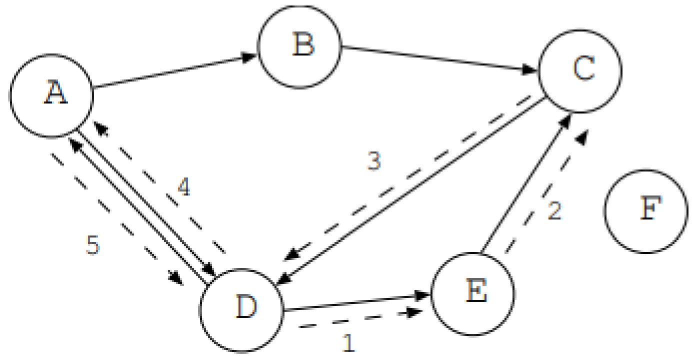


$V^* = \{A, C, D, E\}$, $E^* = \{(A, D), (D, E), (E, C)\}$
Esempio di un sottgrafo

Cammino

Dato $G = (V, E)$ un grafo, un cammino nel grafo G è una sequenza di vertici $v_1, v_2, .., v_n$ tale che $(v_i, v_{i+1}) \in E$ con $1 \leq i \leq n$.

La lunghezza di un cammino è il numero di passaggi(numero di vertici - 1).



D, E, C, D, A, D è un cammino nel G
D, E, C, B, A, D non è un cammino nel G

Viene detto **cammino semplice** se tutti i vertici del cammino sono distinti, fatta eccezione per il primo e l'ultimo che possono essere uguali.

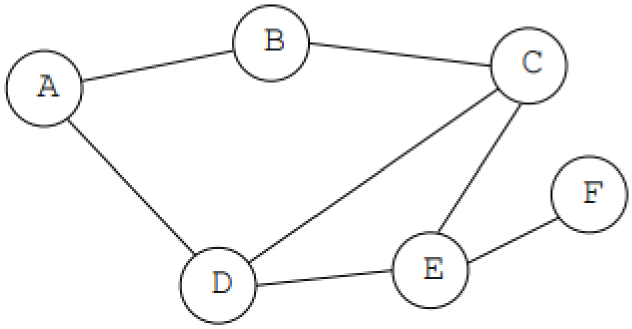
Raggiungibilità

Se esiste un cammino p tra i vertici X e Y, allora si dice che Y è raggiungibile da X.
Per definizione, X è sempre raggiungibile da X.

Inoltre in un grafo
- **Orientato:** la relazione di raggiungibilità non è simmetrica
- **Non orientato:** la relazione di raggiungibilità è simmetrica

Grafi connessi

Se G è un grafo **non orientato**, lo definiamo **connesso** se esiste un cammino da ogni vertice ad ogni altro vertice.



Da A posso raggiungere {B,C,D,E,F}.
Da B posso raggiungere {A,C,D,E,F}.
Da C posso raggiungere {A,B,D,E,F} e così via.
Dunque questo grafo è connesso.
Se F non fosse stato collegato ad E, allora questo grafo non sarebbe stato connesso.

Se G invece è un grafo **orientato**,allora

- è **fortemente connesso** se esiste un cammino da ogni vertice ad ogni altro vertice.
- è **debolmente connesso** se, trasformandalo in un grafo non orientato diventa connesso.

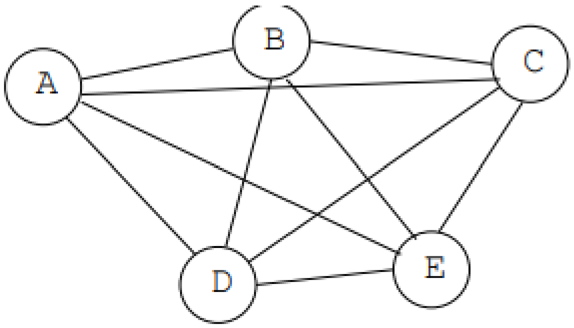
Ciclo

In un grafo un **ciclo** è un cammino $x_1, x_2, .., x_n$ con $n > 2$ e $x_1 = x_n$ che non attraversa lo stesso arco due volte(quest'ultima condizione vale solo per i grafi **non orientati**).

Se il grafo non presenta cicli allora viene detto **aciclico**.
Un grafo orientato aciclico viene detto anche **Directed Acyclic Graph**(DAG)

Grafo completo

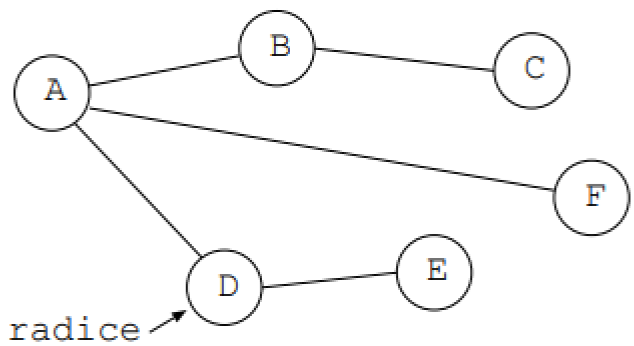
Un grafo G viene detto **completo** se esiste un arco per ogni coppia di vertici.



Se un grafo completo ha n vertici, allora il numero di archi è uguale a $\binom{n}{2} = \frac{n(n-1)}{2}$

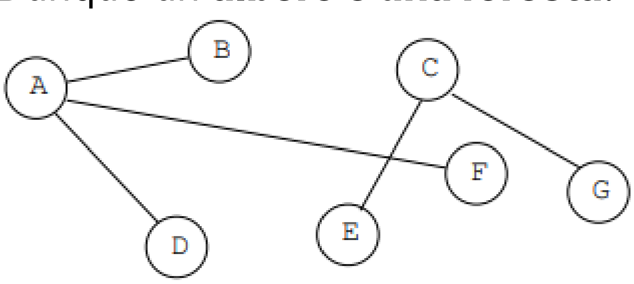
Alberi e foreste

Un albero è un grafo **non orientato**,**connesso** e **aciclico.**
Se non viene definita la radice dell'albero, allora l'albero viene detto **libero**, se no viene detto **radicato.**



Albero libero

Una foresta invece è un grafo **non orientato**, **aciclico** ma non necessariamente **connesso.**
Dunque un **albero è una foresta.**



Foresta che contiene due alberi

Matrici e liste di adiacenza

Un grafo è possibile anche rappresentarlo, oltre al classico metodo grafico, tramite **Matrici di adiacenza** oppure **Liste di adiacenza.**

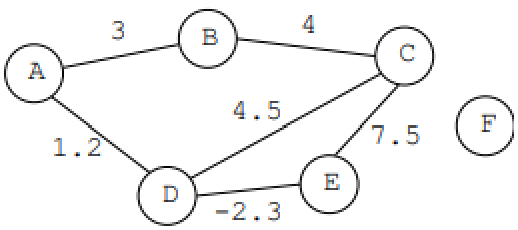
Matrici di adiacenza

E' una matrice $n \times n$ con $n =$ numero di vertici.

$$M(x,y) = \begin{cases} 1 & \text{se } (x,y) \in E \\ 0 & \text{altrimenti} \end{cases}$$

dati x e y due vertici, se (x, y) è un arco(E), allora mettiamo 1 nella posizione (x, y) della matrice, 0 altrimenti.

Nel caso di un grafo **pesato**, la funzione diventa: $M(x, y) = W(x, y)$ con W che definisce la funzione peso.

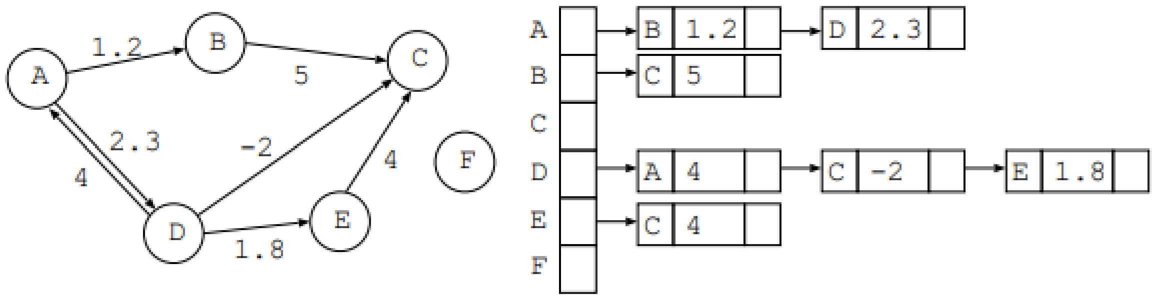


$$\begin{array}{c|cccccc|} A & 0 & 3 & 0 & 1.2 & 0 & 0 \\ B & 3 & 0 & 4 & 0 & 0 & 0 \\ C & 0 & 4 & 0 & 4.5 & 7.5 & 0 \\ D & 1.2 & 0 & 4.5 & 0 & -2.3 & 0 \\ E & 0 & 0 & 7.5 & -2.3 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Esempio di matrice di adiacenza in un grafo pesato

Liste di adiacenza

E' un array lungo n(numero di vertici) dove ogni elemento $A[i]$ è una lista che rappresenta le adiacenze dell'$i - esimo$ elemento.
$L(x)$ è la lista di adiacenza del vertice x che contiene ogni elemento y tale che $(x, y) \in E$.
In un grafo **pesato**, ad ogni elemento associamo anche il valore di $W(x, y)$.



Lista di adiacenza di un grafo pesato

Operazioni su matrici di adiacenza(grafo non orientato)

operazione	tempo di esecuzione
grado(x)	$O(n)$
archiIncidenti(x)	$O(n)$
sonoAdiacenti(x, y)	$O(1)$
aggiungiVertice(x)	$O(n^2)$
aggiungiArco(x, y)	$O(1)$
rimuoviVertice(x)	$O(n^2)$
rimuoviArco(x, y)	$O(1)$

n è il numero di vertici

- **grado(x)**: devo contare il numero di elementi $\neq$ da 0 di $A[x]$
- **archiIncidenti(x)**: stessa cosa sopra
- **sonoAdiacenti(x,y)**: basta vedere se $A[x][y] = A[y][x]$
- **aggiungiVertice(x):** devo ricreare un nuovo array con dimensione maggiore, quindi devo copiare tutti gli elementi nel nuovo array.
- **aggiungiArco(x,y):** basta aggiungere un valore in $A[x][y]$ e in $A[y][x]$
- **rimuoviVertice(x,y):** devo ricreare un nuovo array con dimensione minore, quindi devo copiare tutti gli elementi nel nuovo array.
- **rimuoviArco(x,y):** basta impostare a 0 il valore di $A[x][y]$ (e $A[y][x]$?)

Operazioni su liste di adiacenza(grafo non orientato)

operazione	tempo di esecuzione
grado(x)	$O(\delta(x))$
archiIncidenti(x)	$O(\delta(x))$
sonoAdiacenti(x, y)	$O(\min(\delta(x), \delta(y)))$
aggiungiVertice(x)	$O(1)$
aggiungiArco(x, y)	$O(1)$
rimuoviVertice(x)	$O(m)$
rimuoviArco(x, y)	$O(\delta(x) + \delta(y))$

$\delta(x)$ è il numero degli adiacenti, n è il numero di vertici e m il numero di archi

- **grado(x)**: bisogna scorrere tutta la lista di x lunga $\delta(x)$ per contare il numero di adiacenze e quindi il grado.
- **archiIncidenti(x)**: stesso discorso di sopra
- **sonoAdiacenti(x,y)**: bisogna scorrere la liste di x o di y per vedere se x è presente in y o viceversa.
  Il tempo di esecuzione è dato dalla lunghezza minore delle due liste.
- **aggiungiVertice(x)**: basta aggiungere un elemento alla lista di x
- **aggiungiArco(x,y)**: basta aggiungere un elemento alla lista di x e un elemento alla lista di y
- **rimuoviVertice(x)**:boh
- **rimuoviArco(x,y)**: devo scorrere le liste di x e y e rimuovere i due elementi che rappresentano l'arco