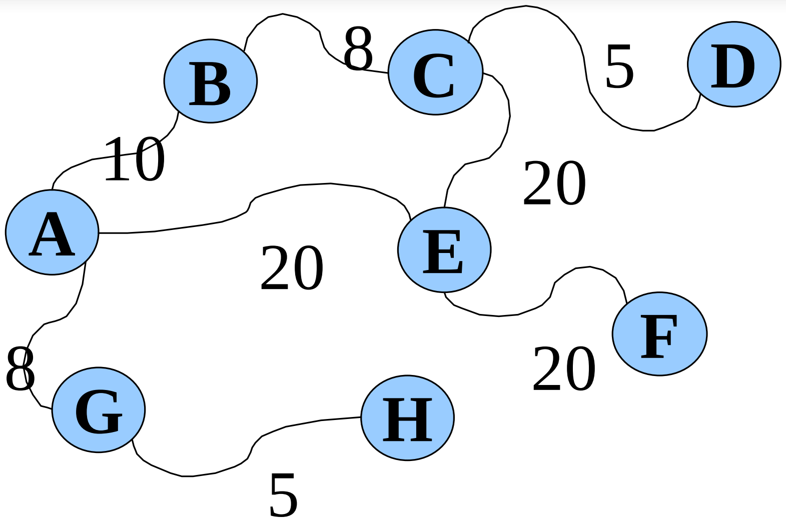
Università degli Studi di Palermo
Facoltà di Ingegneria - Corso di Laurea in Ingegneria Informatica
A.A. 2001/2002

CORSO DI FONDAMENTI DI INFORMATICA

Grafi



Ing. Ignazio Infantino

Sommario

- Definizioni
- Rappresentazione dei grafi
- Algoritmi di visita
- Esempi in C

Grafo orientato

Un grafo orientato (o diretto o di-grafo) G è una coppia (V,E), dove

- V è l'insieme (finito) dei vertici
- E è l'insieme (finito) degli archi, che corrisponde ad una relazione binaria su V

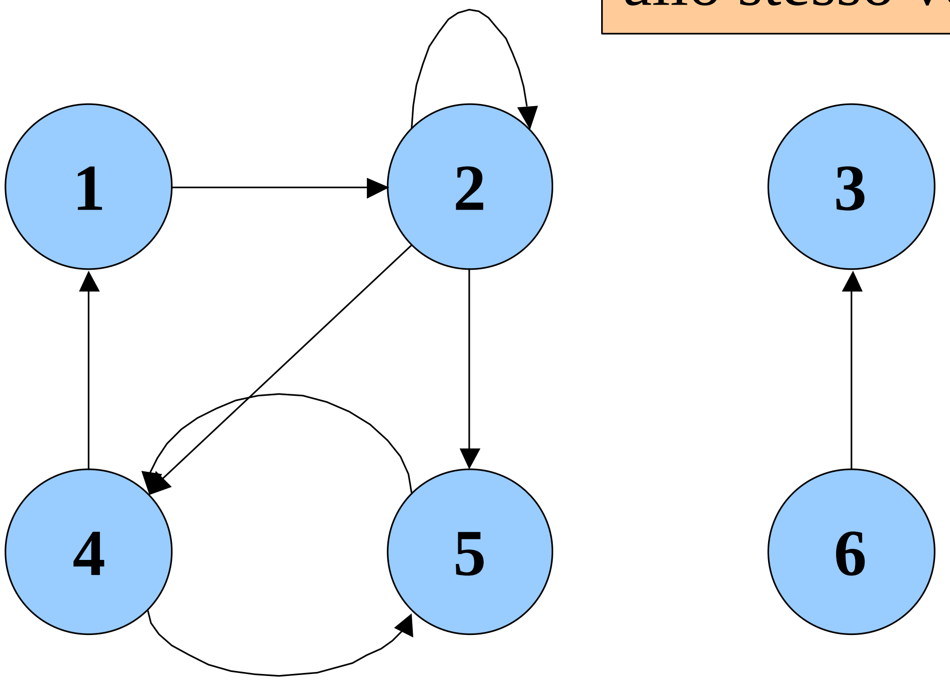
Rappresentazione

I grafi orientati si rappresentano spesso disegnando

- i vertici sotto forma di cerchi
- gli archi sotto forma di archi

Esempio

Cappio
arco da un vertice
allo stesso vertice

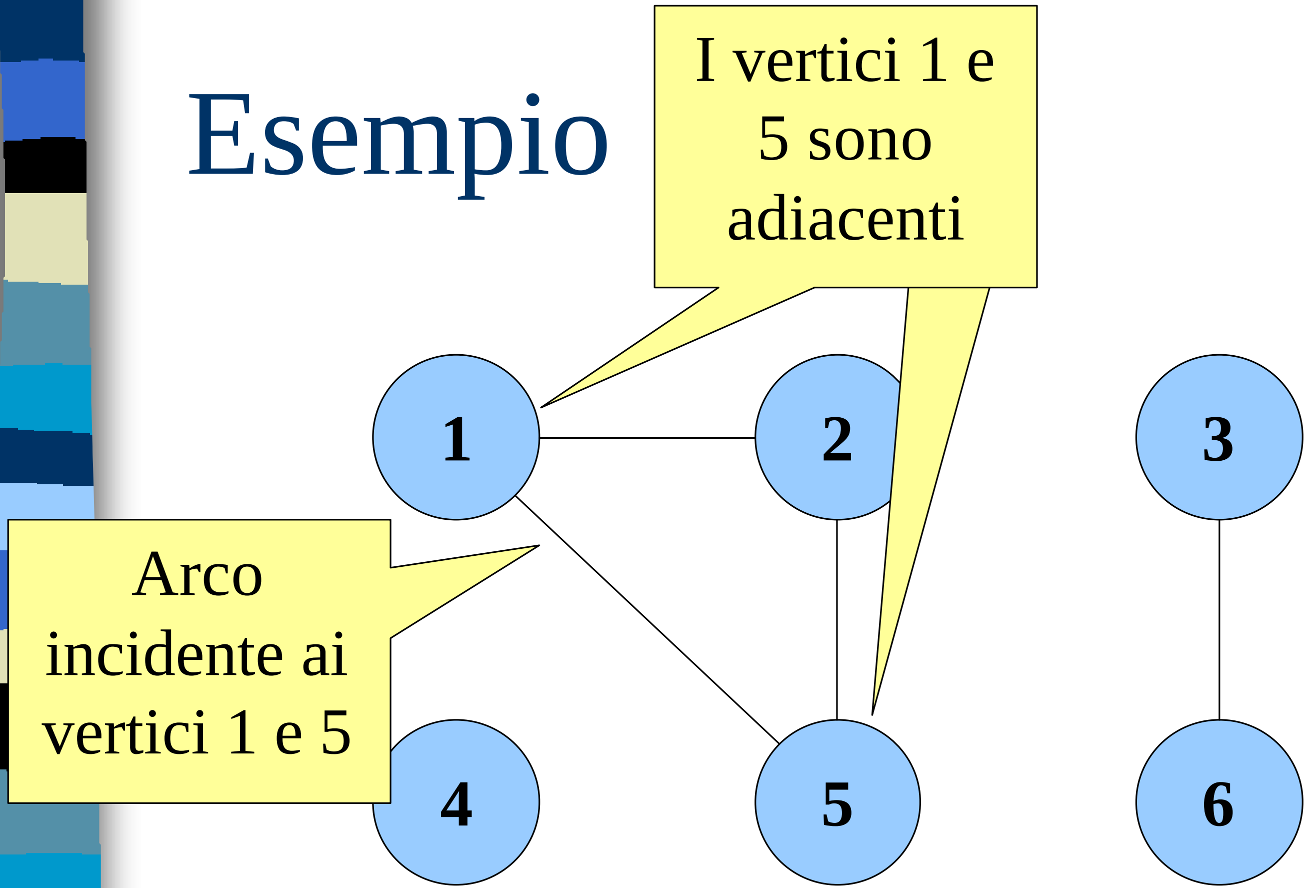


G
V=(1,2,3,4,5,6)
E=[(1,2),(2,2),(2,4),(2,5),(4,1),(4,5),(5,4),(6,3)]

Grafo non orientato

- Un grafo non orientato è ancora una coppia G=(V,E), ma l'insieme E è costituito in questo caso da coppie non ordinate di vertici.
- Gli archi sono rappresentati da linee
- Non sono ammessi cappi

Esempio



G
V=(1,2,3,4,5,6)
E=[(1,2),(1,5),(2,5),(3,6)]

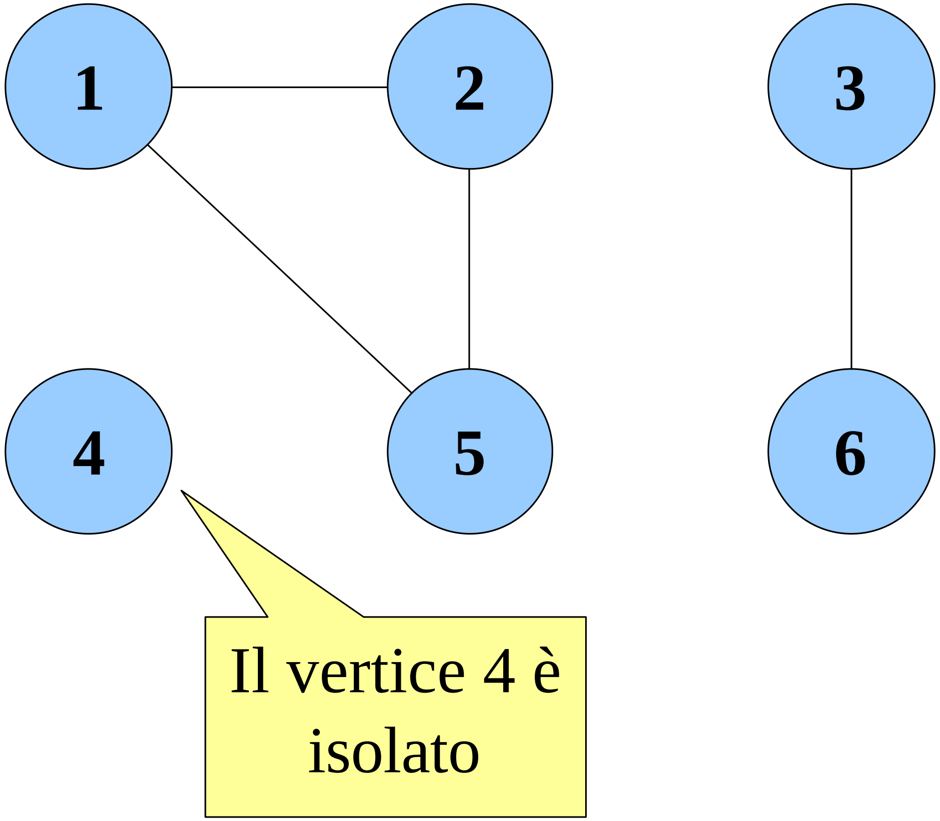
Grado

In un grafo non orientato il grado di un vertice è il numero di archi incidenti.

In un grafo orientato:

- Il grado entrante è il il numero di archi entranti
- Il grado uscente è il numero di archi uscenti
- Il grado è la somma del grado entrante e del grado uscente
- Un vertice di grado 0 si dice isolato

Esempio



Cammino

Un cammino da un vertice u ad un vertice u' in un grafo G(V,E) è una sequenza di vertici (v_0, v_1, v_2, …, v_k) tale che u= v_0 e u'= v_k, con (v_{i-1}, v_i,) $\in$ E per i=1, 2, …, k.

K è la lunghezza del cammino.

Se esiste un cammino da u a u' allora si dice che u' è raggiungibile da u.

Un cammino è semplice se tutti i vertici che lo compongono sono distinti.

Cicli

Un ciclo è un cammino in cui $v_0 = v_k$.

Un cappio è un ciclo di lunghezza 1.

Un grafo senza cicli si dice aciclico.

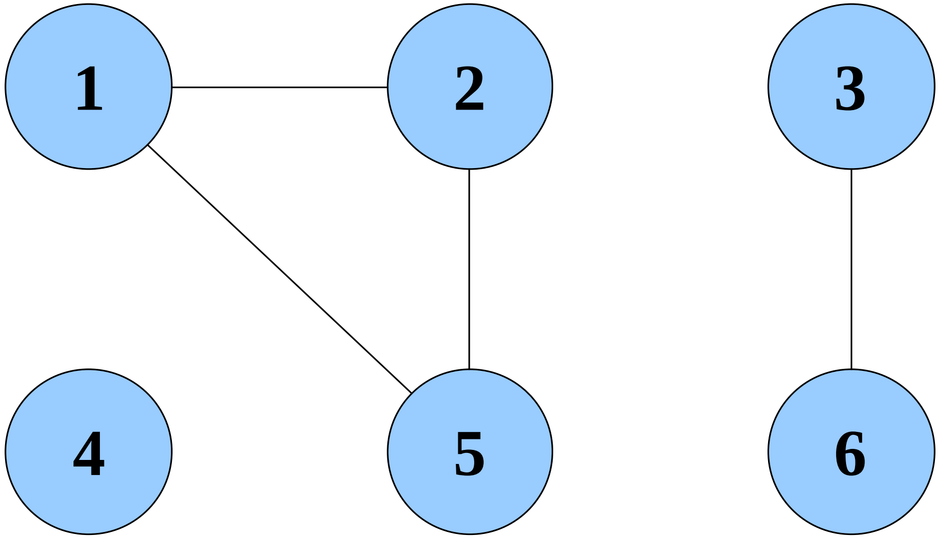
Raggiungibilità

Un grafo non orientato è connesso se per ogni coppia di vertici esiste un cammino che li collega.

I sottografi connessi di dimensione massima si dicono componenti connesse.

Un grafo connesso è composto da una sola componente connessa.

Esempio



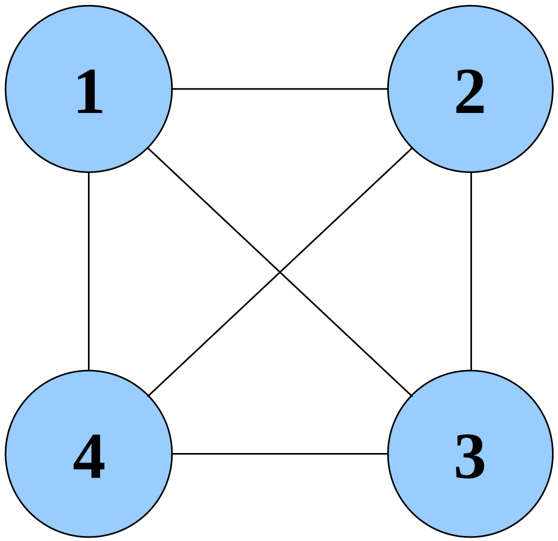
Il grafo ha 3 componenti connesse:
(1,2,5) (3,6) (4)

Raggiungibilità

Un grafo orientato è fortemente connesso se per ogni coppia ordinata di vertici (u,u’) esiste un cammino che collega u ad u’.

Grafo completo

Un grafo si dice completo se data una qualsiasi coppia di vertici, questi sono adiacenti.

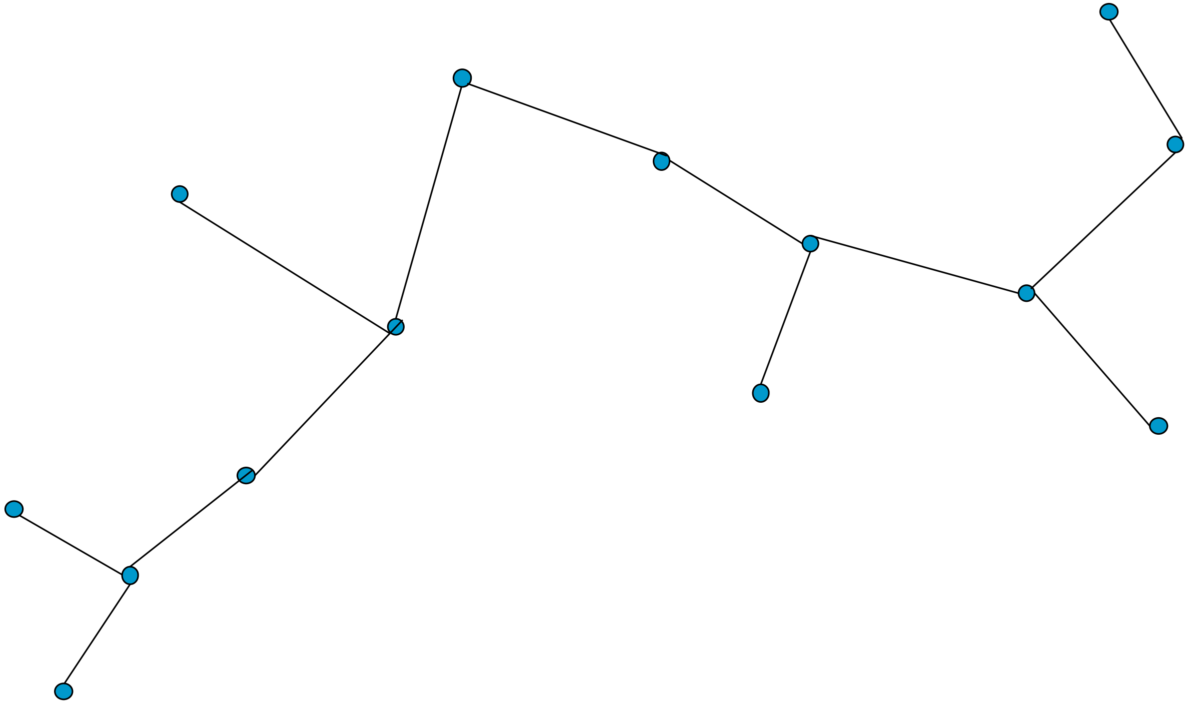


Alberi e foreste

Un grafo non orientato aciclico si dice foresta.

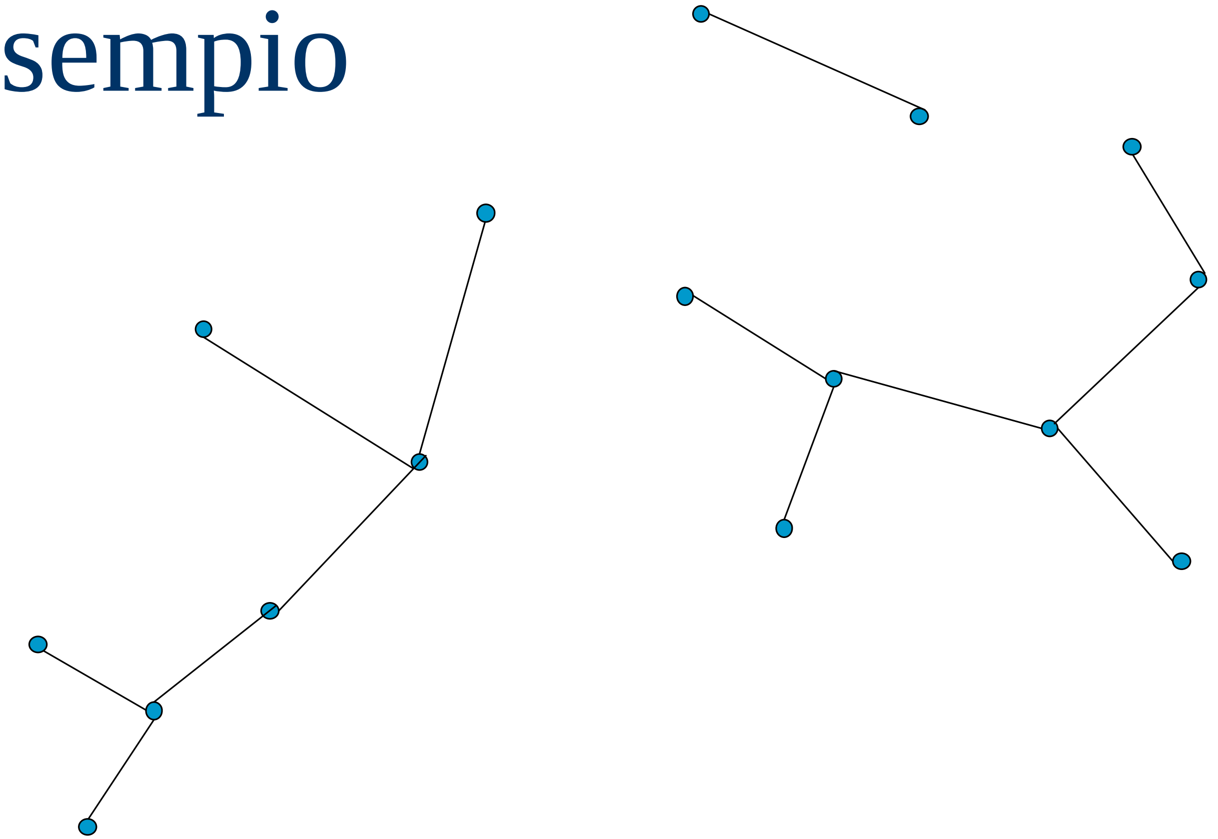
Un grafo no orientato aciclico connesso si dice albero.

Esempio



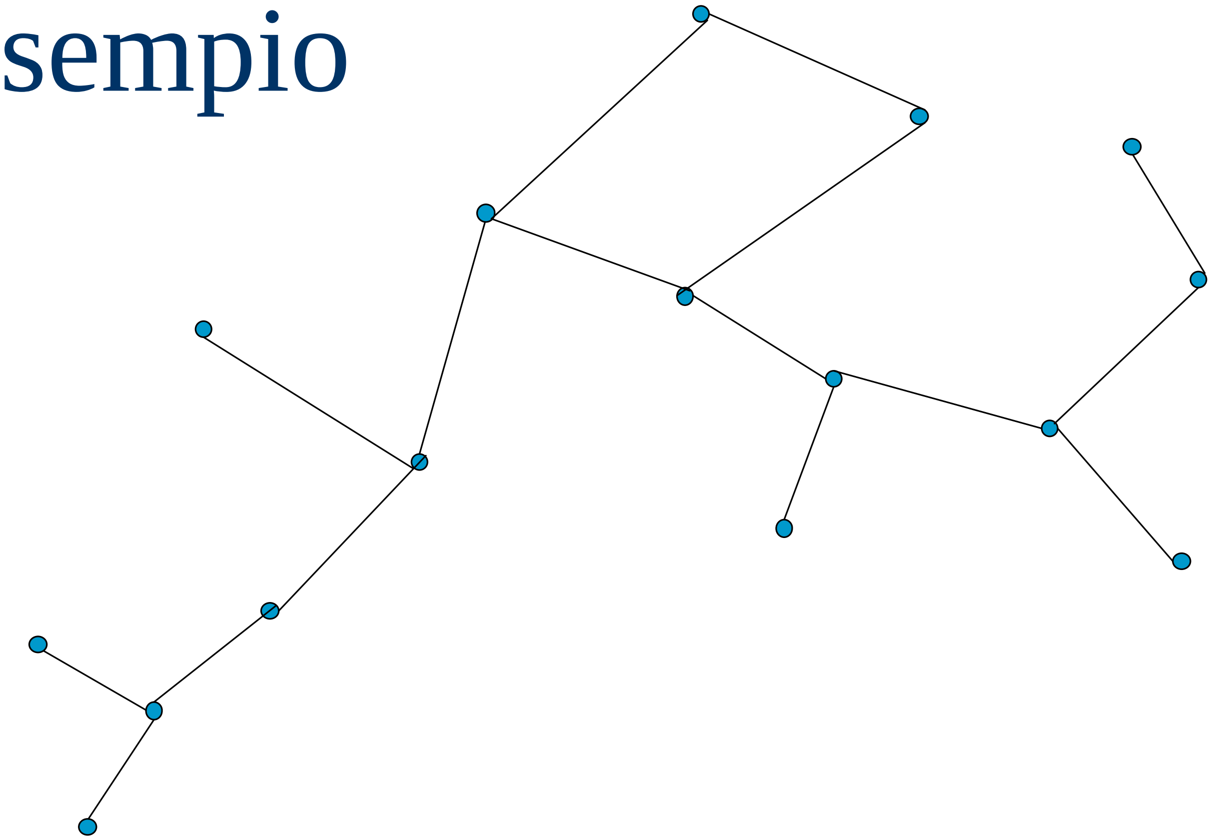
Il grafo è un albero

Esempio



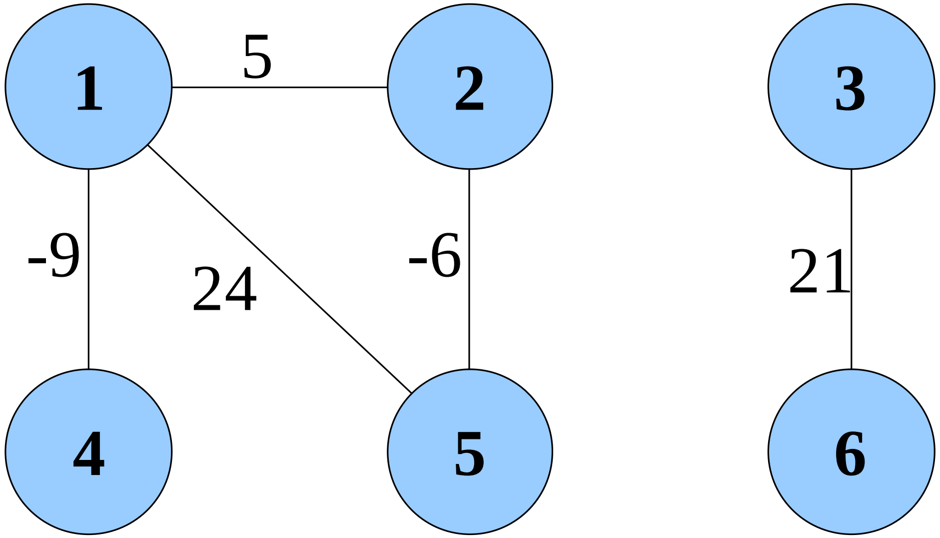
Il grafo è una foresta

Esempio



Il grafo è né un albero né una foresta dato che contiene un ciclo.

Grafi pesati

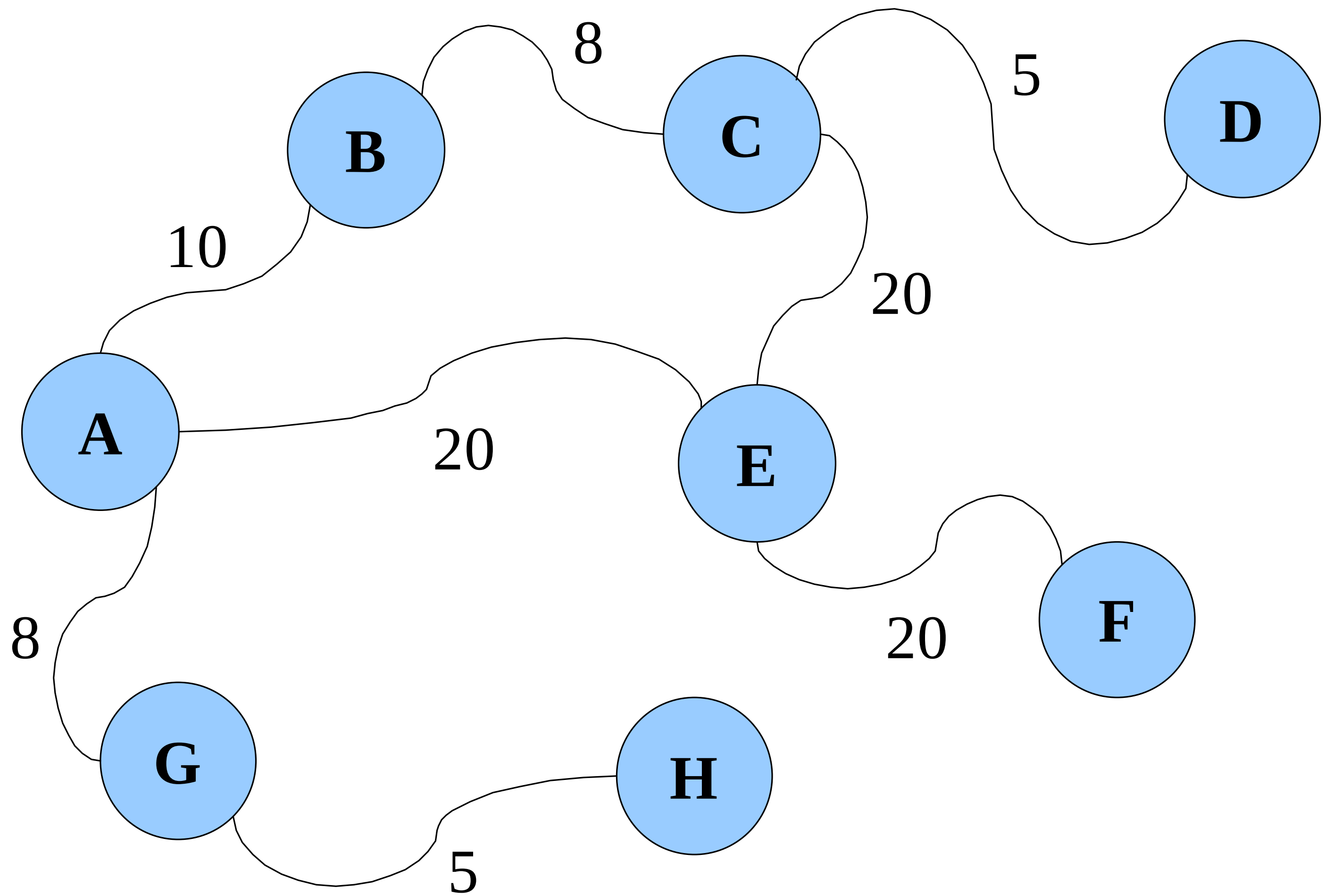


Nei grafi pesati ad ogni arco è associato un peso

Esempio di applicazione

- Una società telefonica gestisce un certo numero di centrali (a cui sono collegati gli utenti), collegate tra loro da canali dotati ciascuno di una certa banda.
- Per rappresentare in ciascun istante la situazione, tenendo conto di eventuali guasti a canali e centrali, si può utilizzare un grafo non orientato pesato.
- IL peso di ciascun arco corrisponde lla sua capacità (o banda).

Connessioni telefoniche



Conn. telefoniche: problemi

- Il grafo è connesso? Se non lo è, ci saranno connessioni non possibili?
- Quali sono i canali critici, ossia tali per cui la mancanza del corrispondente arco rende il grafo non connesso?
- Quali sono le centrali critiche, tali per cui la mancanza del corrispondente vertice (o degli archi incidenti) rende il grafo non connesso?

Rappresentazioni dei grafi

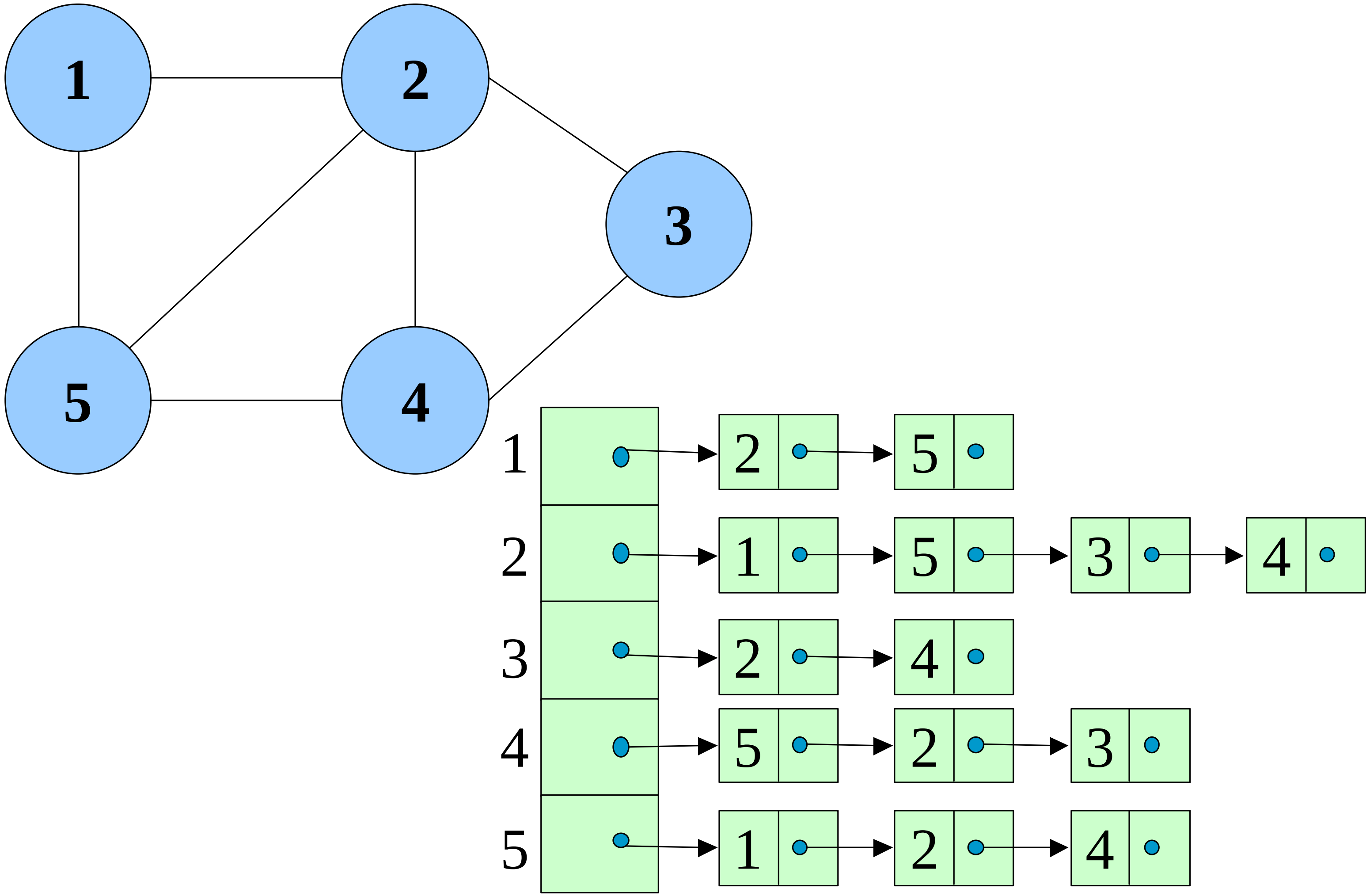
Esistono due modi principali per rappresentare un grafo G=(V,E):

- Liste di adiacenza
- Matrici di adiacenza

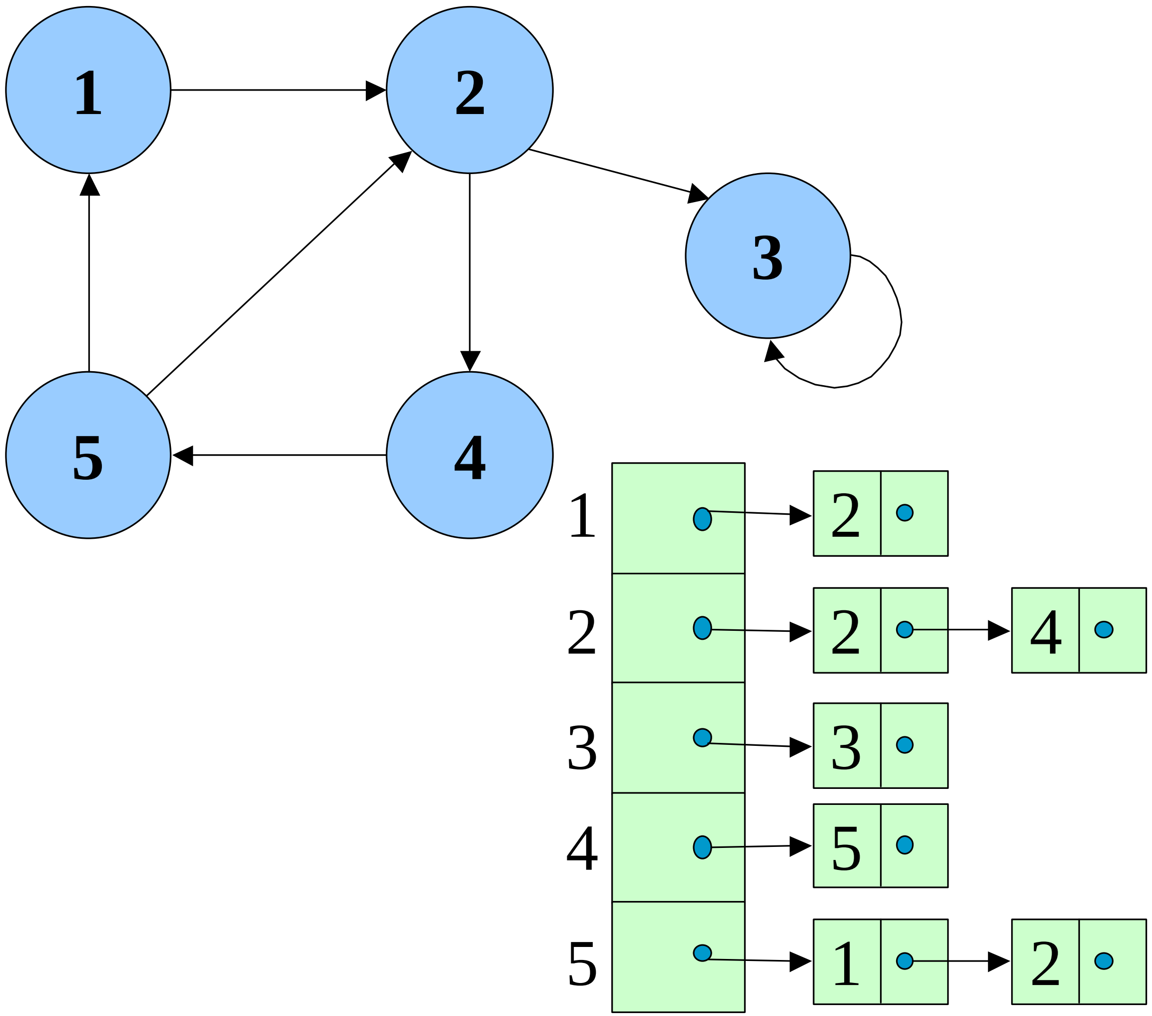
Liste di adiacenza

Dato un grafo G(V,E), esso può essere rappresentato tramite un vettore A, il cui elemento A[i] contiene il puntatore alla lista dei vertici adiacenti al vertice i.

Esempio



Esempio



Uso della memoria

Se il grafo è orientato, il numero di elementi presenti nelle liste è pari al numero di elementi dell'insieme E.

Se il grafo è non orientato, si hanno il doppio degli elementi presenti in E.

Limiti

Il principale limite della rappresentazione tramite liste di adiacenza consiste nella difficoltà di verificare se esiste un arco tra il vertice u ed il vertice u’.

In tal caso è necessario scandire l’intera lista dei vertici adiacenti a u (o a u’).

Matrice di adiacenza

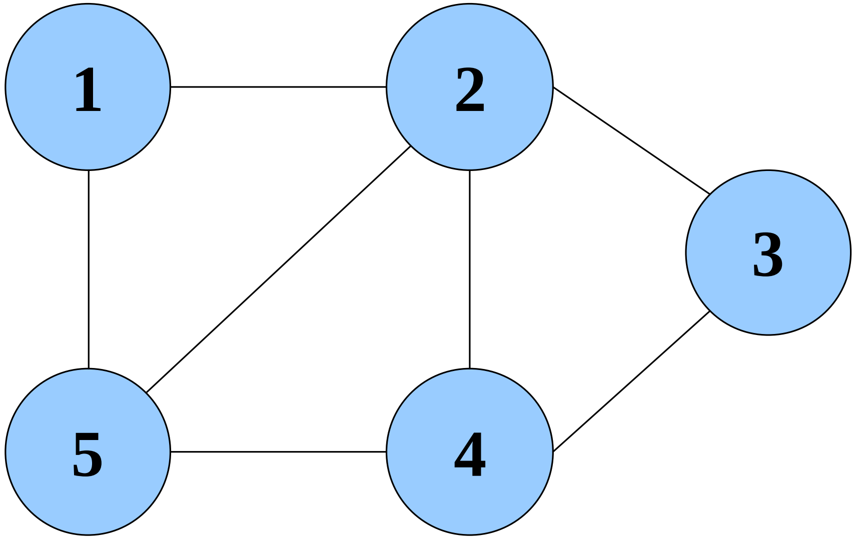
Consiste in una matrice di dimensione $|V| \times |V|$. L'elemento a_{ij} vale

- 1 se $(i,j) \in E$
- 0 altrimenti

Nel caso di grafi non orientati la matrice è simmetrica.

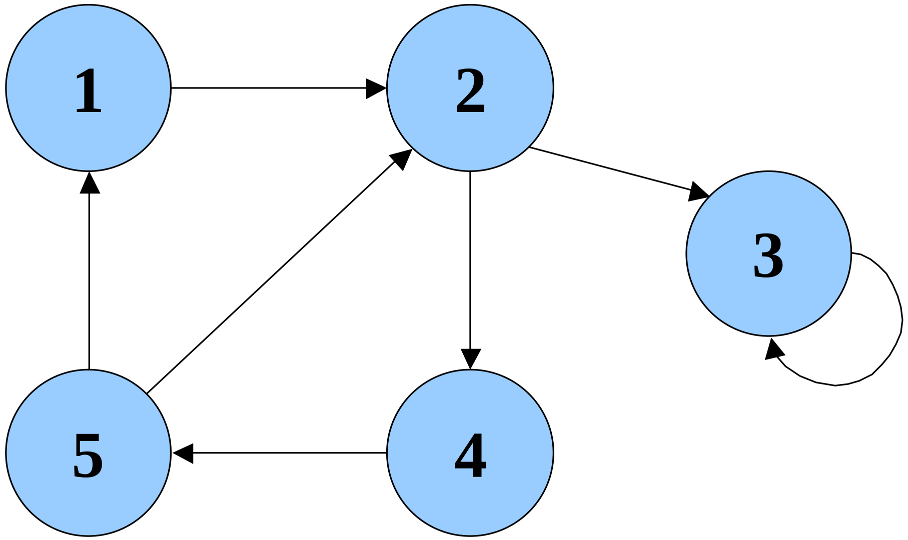
Nel caso di grafi pesati l'elemento a_{ij}, qualora non sia nullo, assume il valore del peso dell'arco.

Esempio



	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

Esempio



	1	2	3	4	5
1	0	1	0	0	0
2	0	0	1	1	0
3	0	0	1	0	0
4	0	0	0	0	1
5	1	1	0	0	0

Confronto

Rispetto alla rappresentazione tramite liste di adiacenza, quella tramite matrice:

- Richiede più memoria (a meno che il grafo non sia particolarmente denso)
- Permette un accesso più efficiente alla topologia del grafo

Algoritmi di visita

L'operazione corrispondente ad esplorare un grafo a partire da un vertice dato (sorgente) toccando tutti i vertici da questo raggiungibili si definisce visita.

Esistono due principali algoritmi di visita:

- Visita in ampiezza
- Visita in profondità

Visita in ampiezza

La visita in ampiezza (o breadth-first search, BFS) consiste nel visitare i vertici del grafo in livelli:

1. L=0, S_L=(sorgente)
2. Si visitano tutti i vertici S_{L+1} non ancora visitati e adiacenti ad almeno un vertice in S_L
3. L=L+1
4. Si ripete da 2 sino a che S_L è vuoto

Albero BFS

La visita in ampiezza produce un albero BFS, nel quale

- La radice è il vertice sorgente
- I vertici sono quelli del grafo
- Gli archi sono un sottoinsieme di quelli del grafo

Pseudo-codice (1)

BSF(G,s)

- for ogni vertice u $\in$ V[G]-s
- do color[u] $\leftarrow$ WHITE
- d[u] $\leftarrow \infty$
- π[u] $\leftarrow$ NIL
- color[s] $\leftarrow$ GRAY
- d[s] $\leftarrow$ 0
- π[s] $\leftarrow$ NIL
- Q $\leftarrow$ s

Pseudo-codice (2)

```
while Q ≠ ∅
  do u ← head[Q]
     for ogni v ∈ Adj[u]
        do if color[v] = WHITE
             then color[v] ← GRAY
                  d[v] ← d[u]+1
                  π[v] ← u
                  Enqueue(Q,v)
     Dequeue(Q)
     color[u] ← BLACK
```

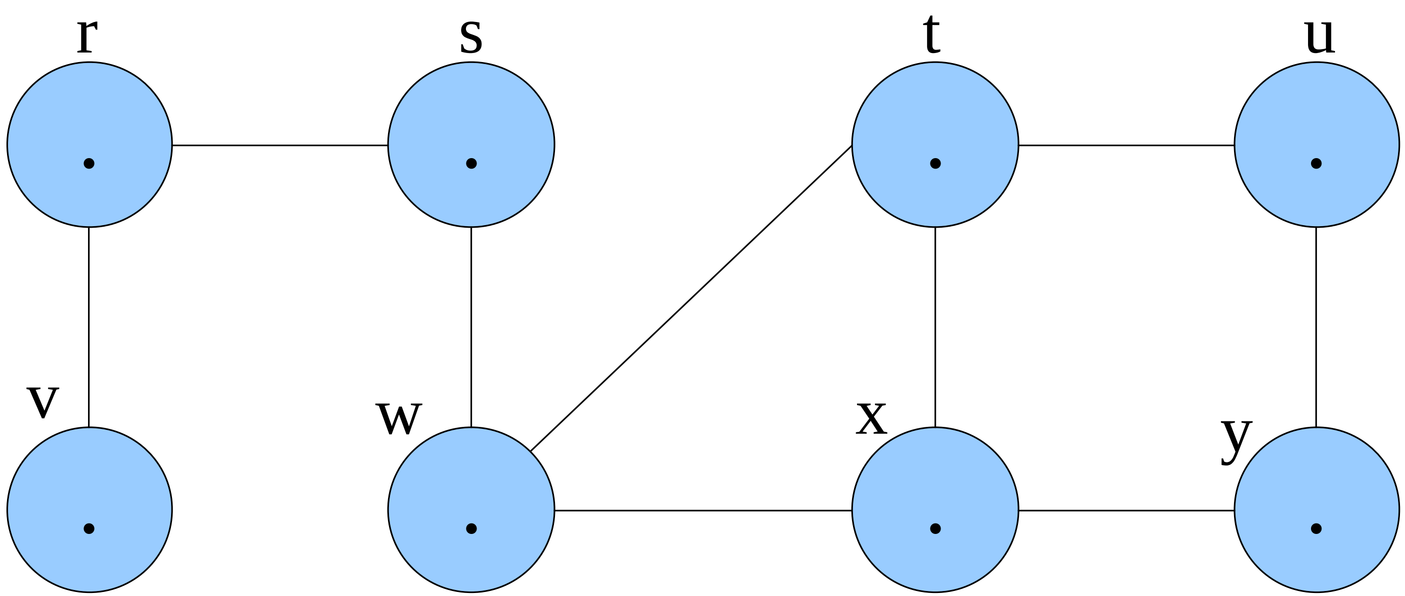
Nota

La colorazione dei vertici segue il seguente criterio:

- Inizialmente tutti i vertici sono bianchi
- Un vertice è colorato in grigio quando viene raggiunto per la prima volta
- Un vertice è colorato in nero quando tutti i vertici ad esso adiacenti e non ancora visitati sono stati inseriti nella coda

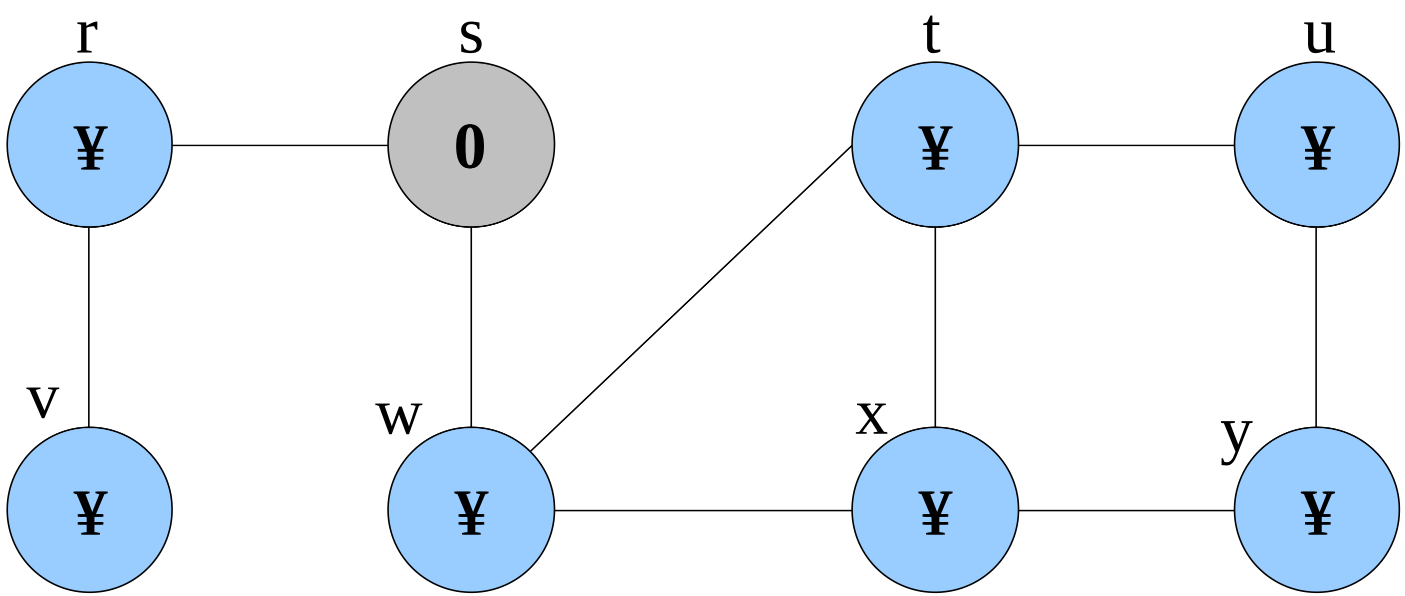
Esempio (0)

Vertice sorgente: s



Esempio (1)

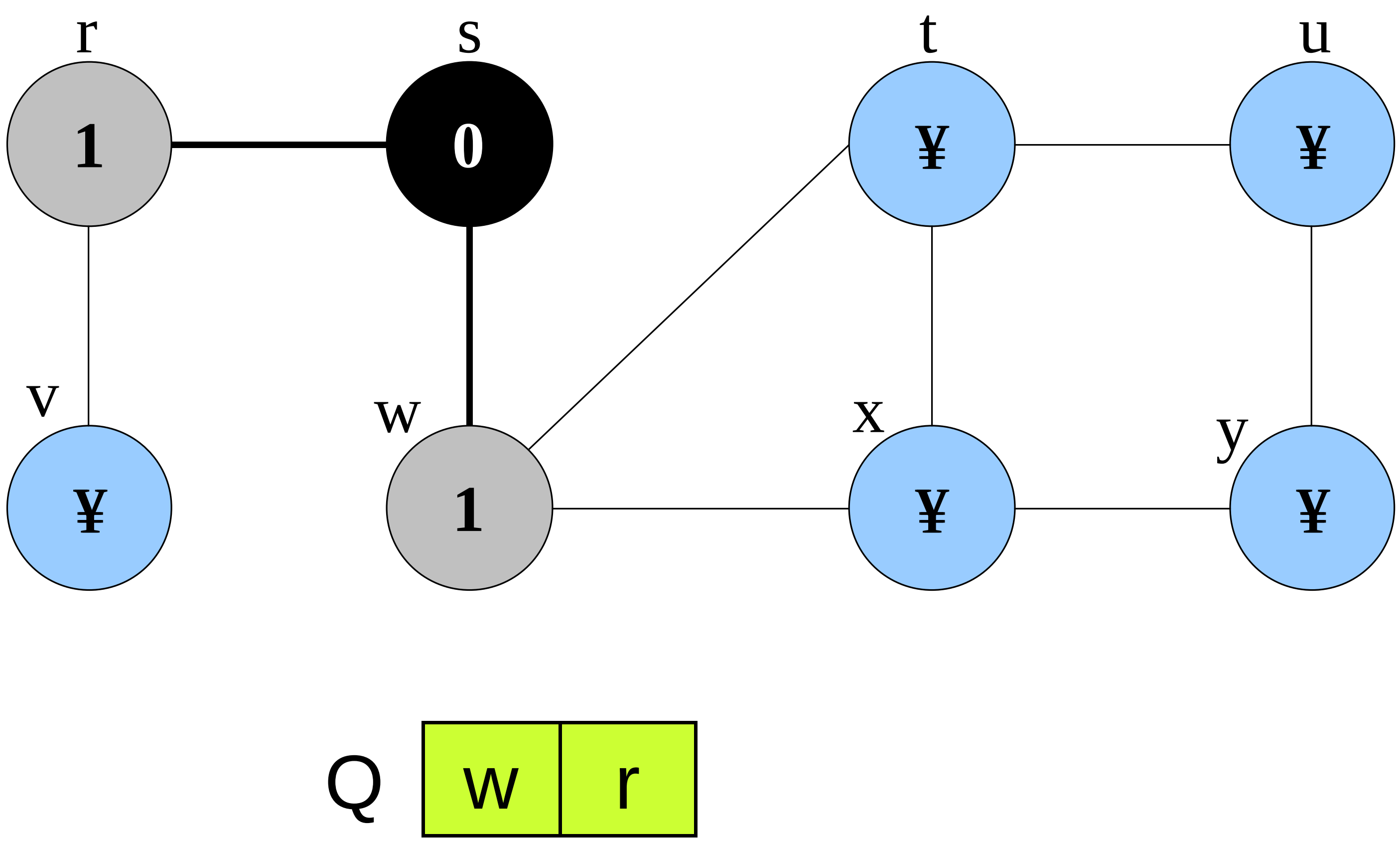
Inizializzazione



Q | s

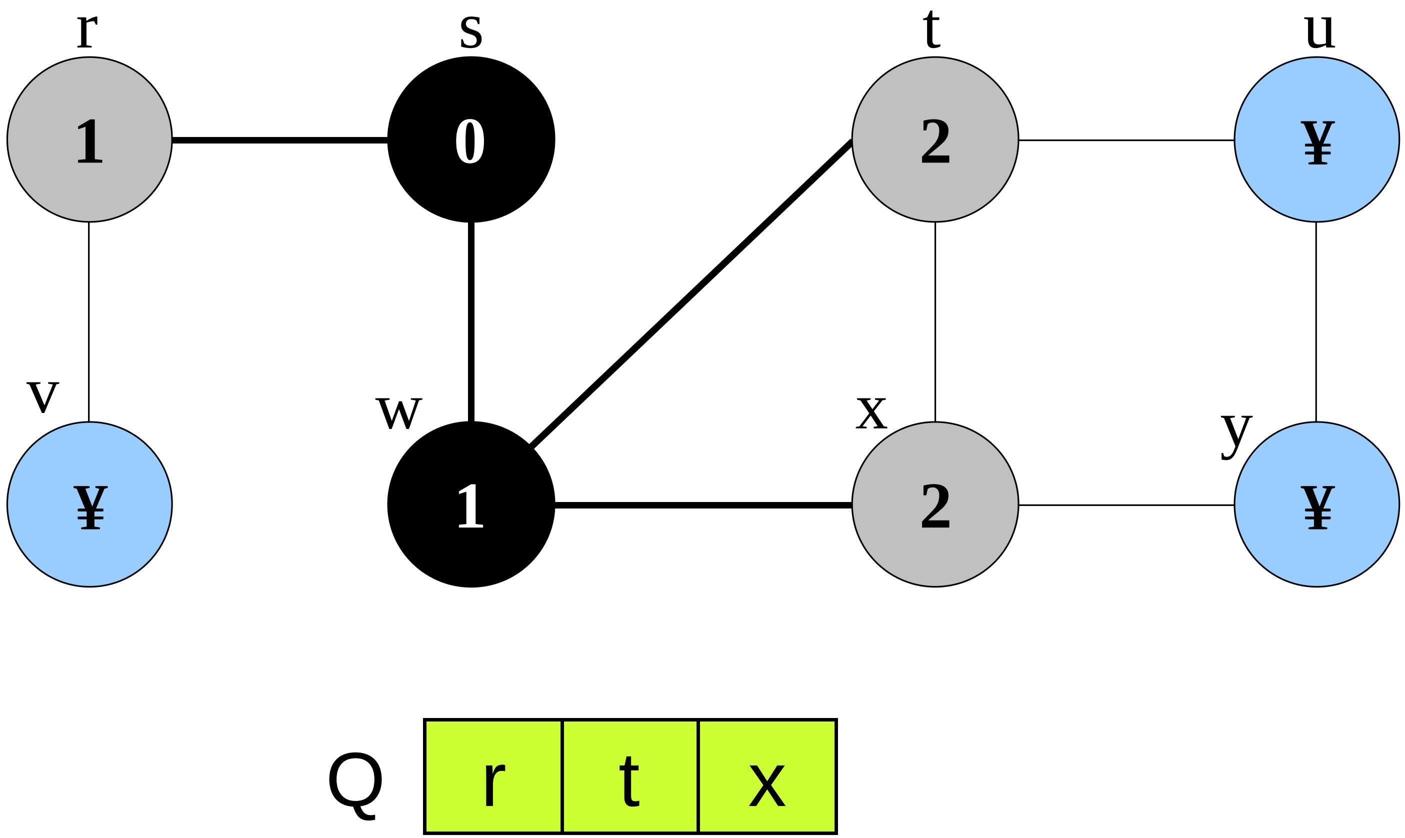
Esempio (2)

- Estrae s da Q
- Metti in Q e colora in grigio i vertici adiacenti a s
- Colora s in nero



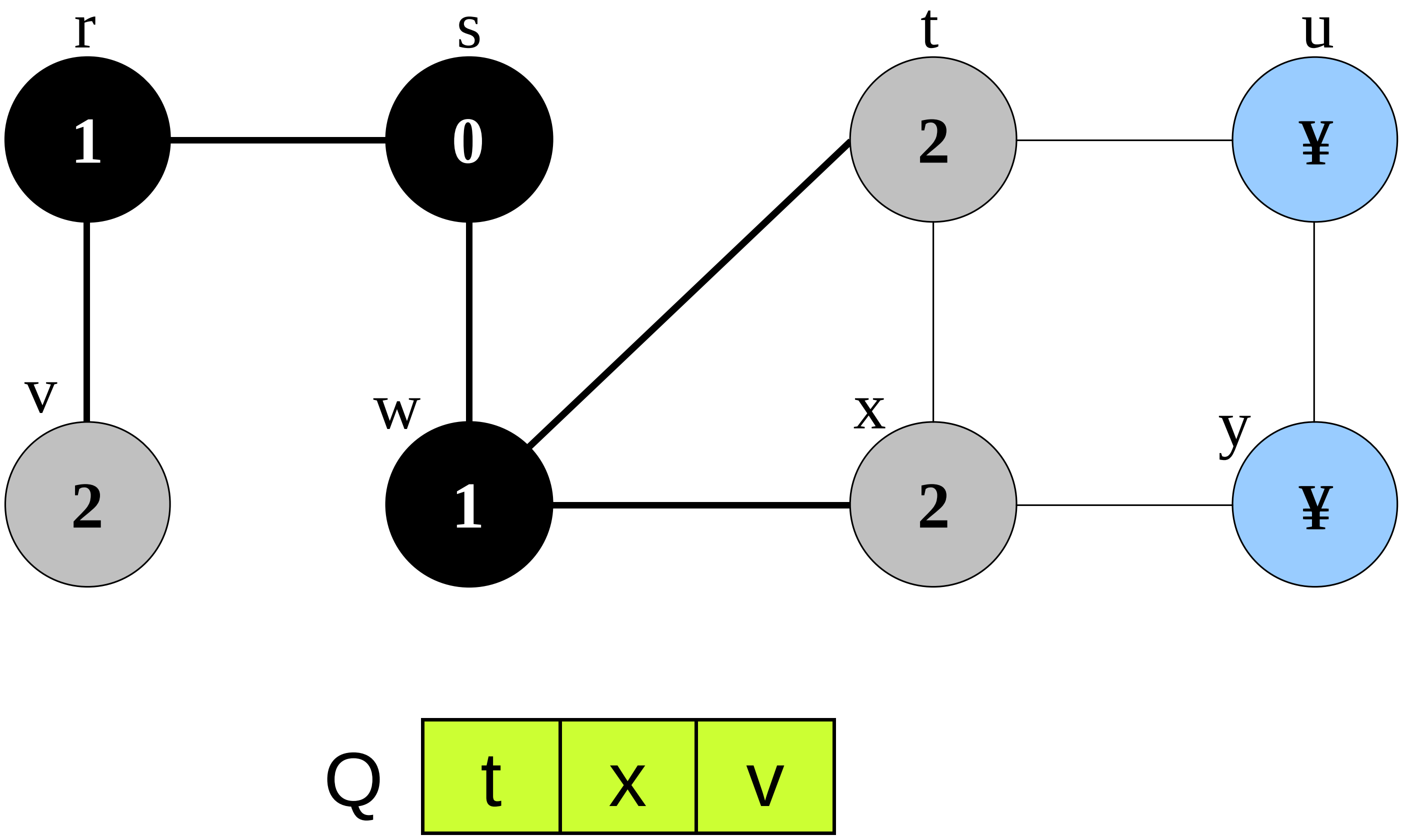
Esempio (3)

- Estrae w da Q
- Metti in Q e colora in grigio i vertici adiacenti a w
- Colora w in nero



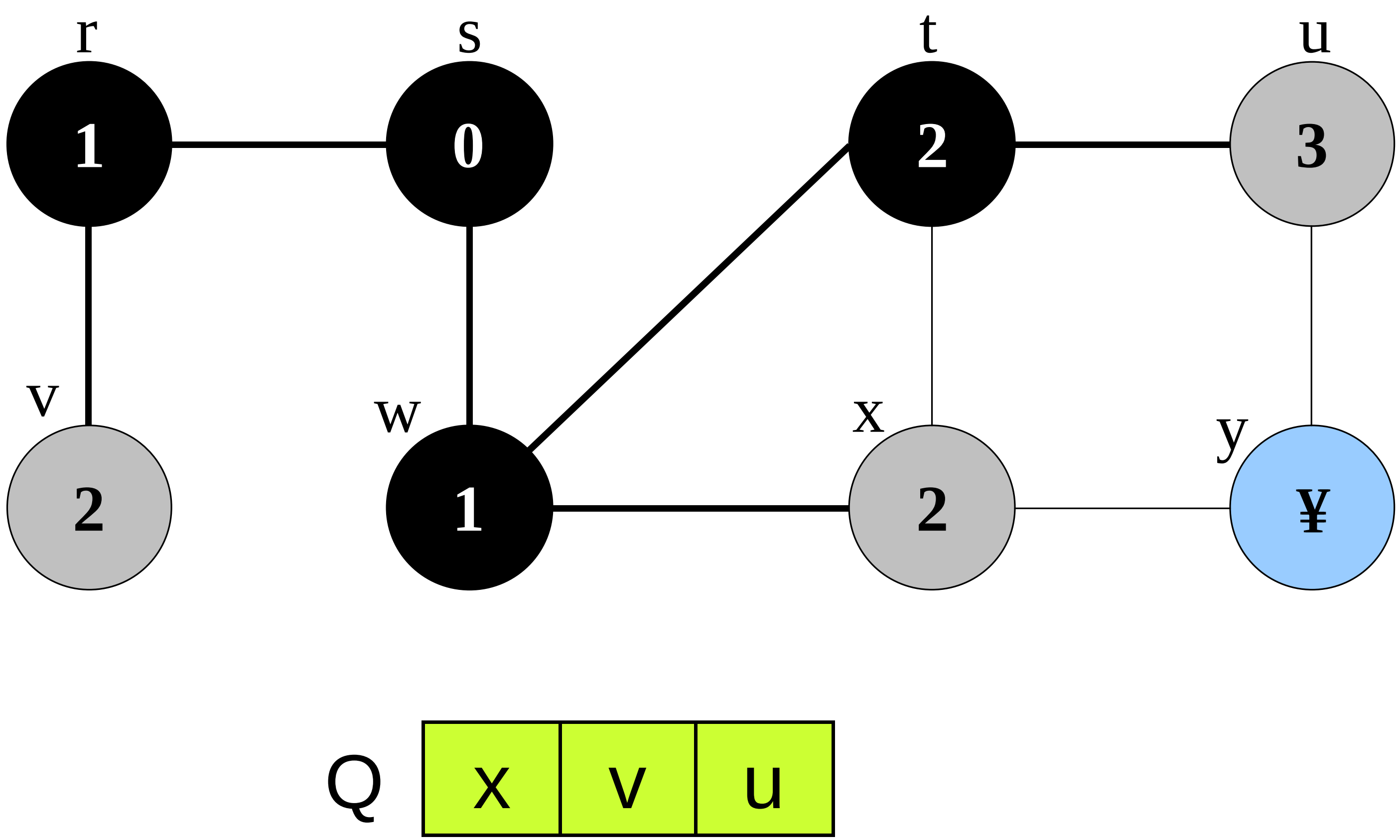
Esempio (4)

- Estrae r da Q
- Metti in Q e colora in grigio i vertici adiacenti a r
- Colora r in nero



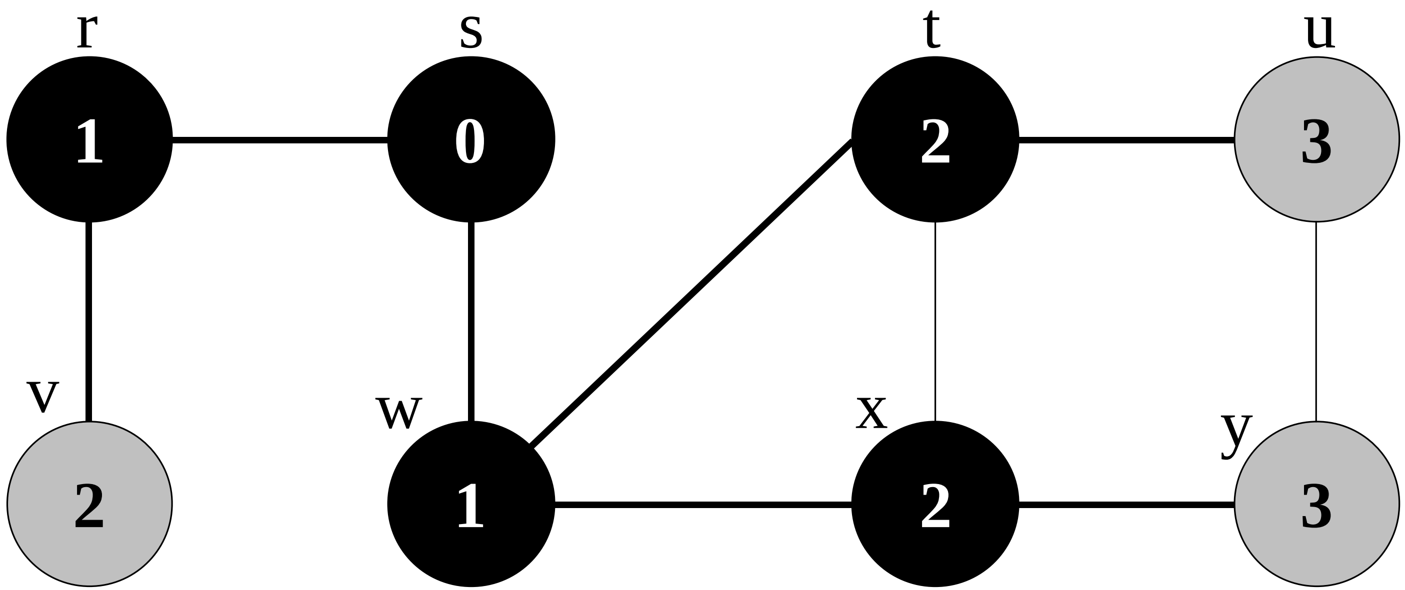
Esempio (5)

- Estrae t da Q
- Metti in Q e colora in grigio i vertici adiacenti a t
- Colora t in nero



Esempio (6)

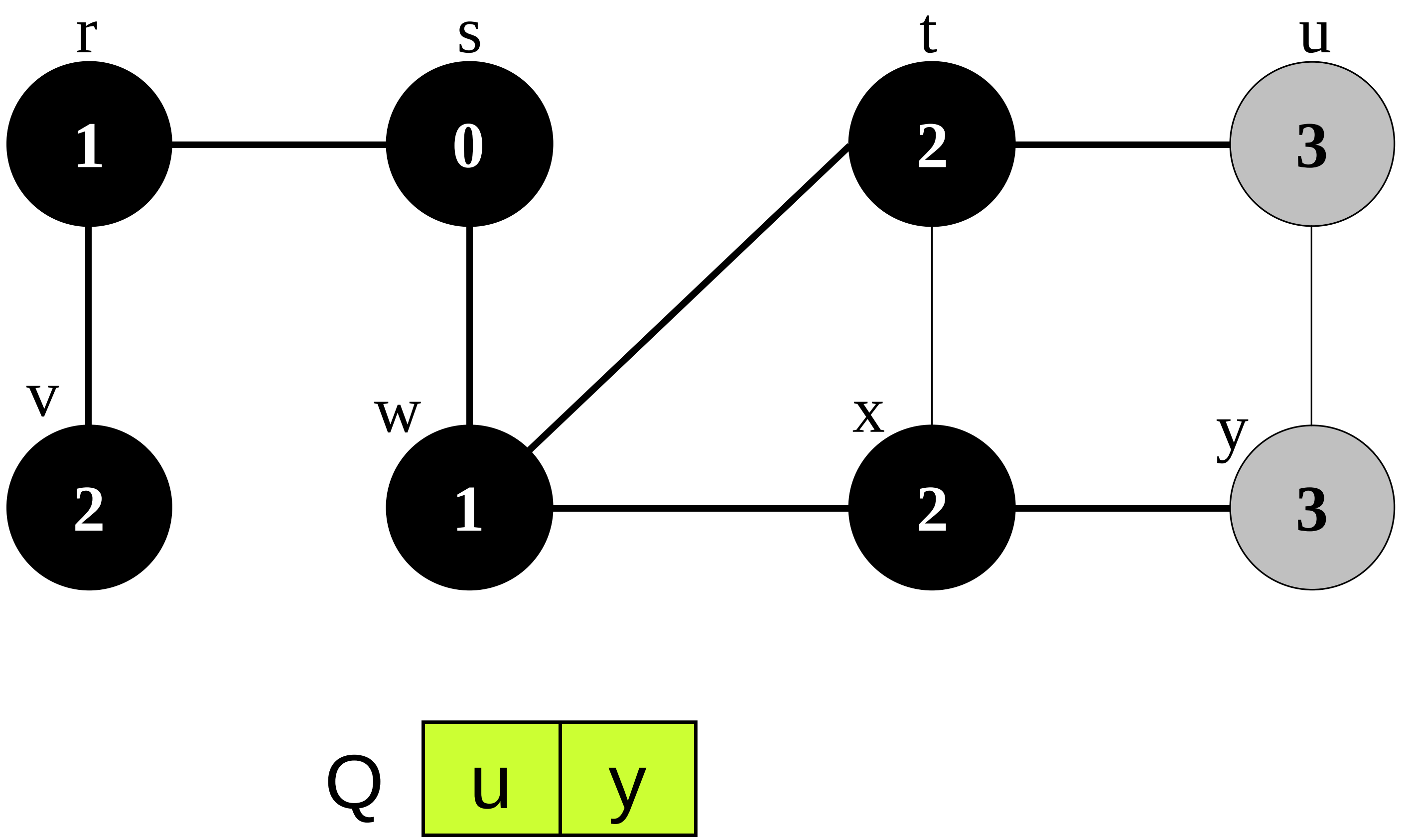
- Estrae x da Q
- Metti in Q e colora in grigio i vertici adiacenti a x
- Colora x in nero



Q	v	u	y

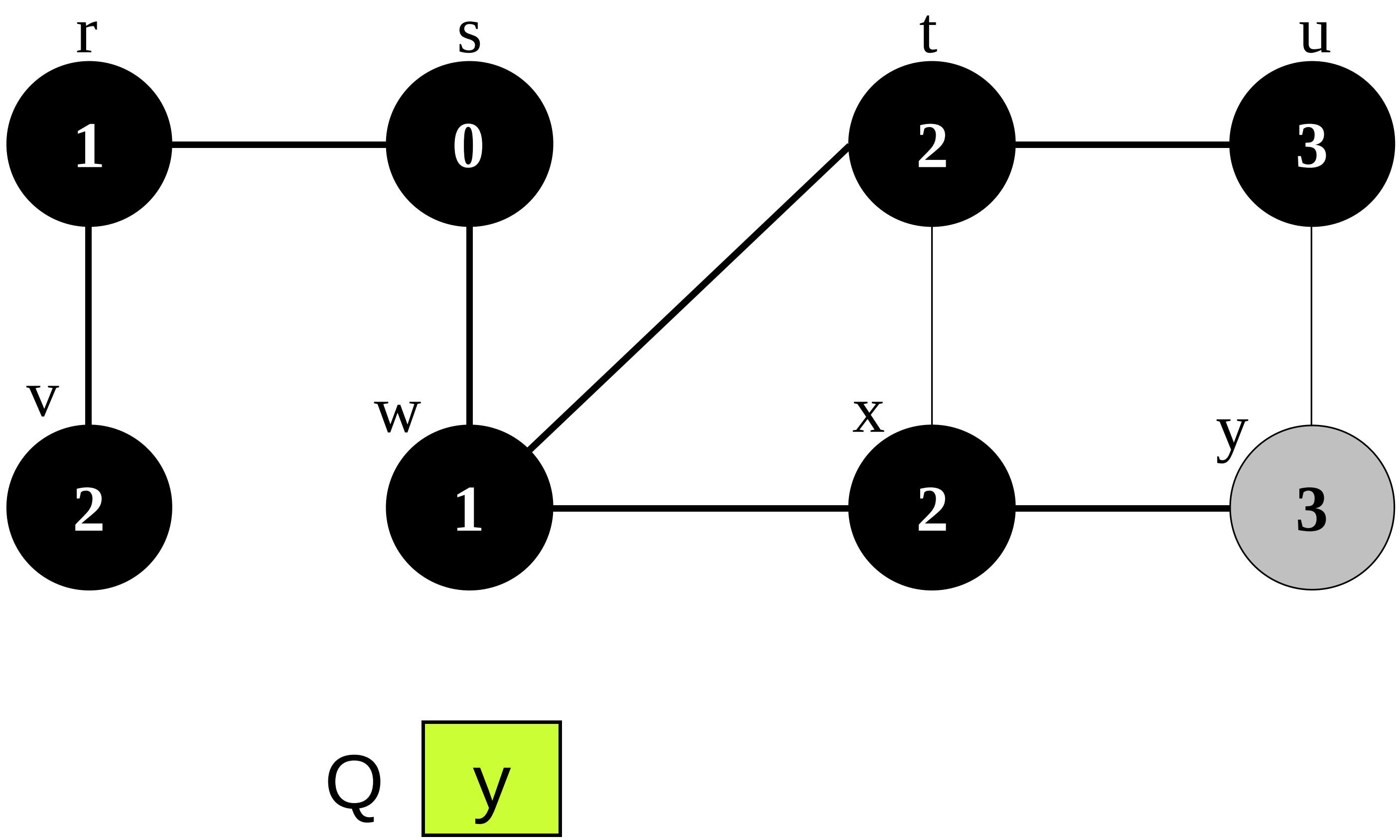
Esempio (7)

- Estrae v da Q
- Metti in Q e colora in grigio i vertici adiacenti a v
- Colora v in nero



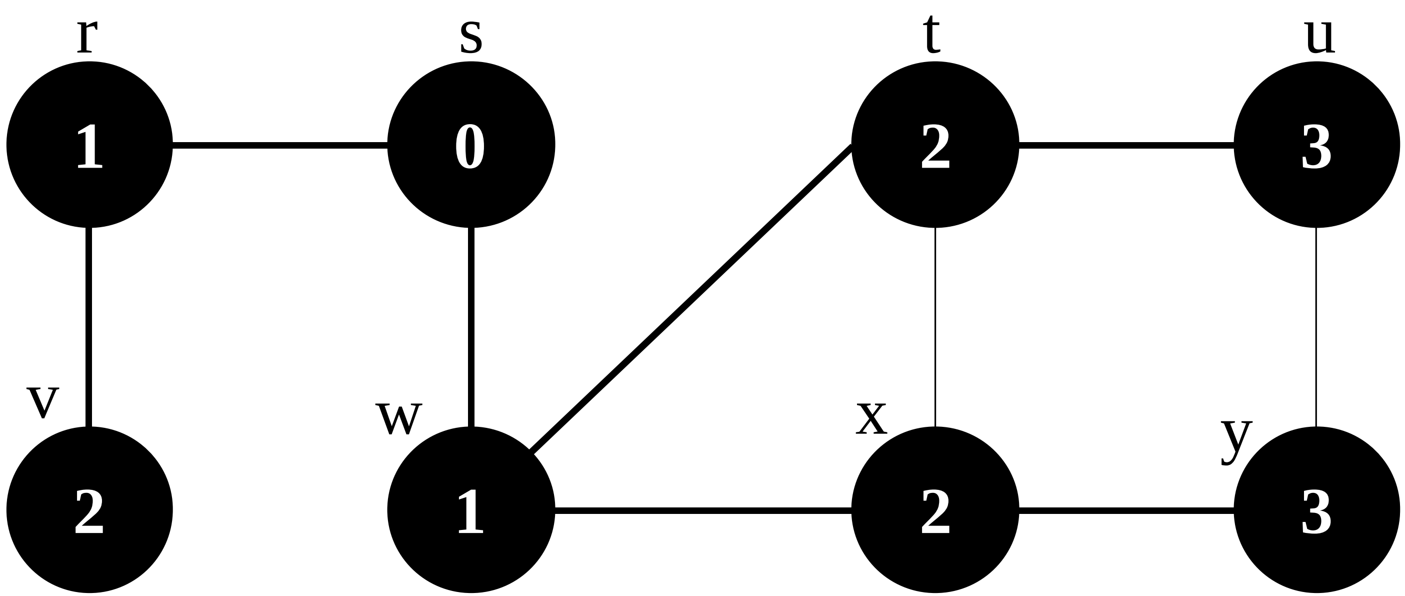
Esempio (8)

- Estrae u da Q
- Metti in Q e colora in grigio i vertici adiacenti a u
- Colora u in nero



Esempio (9)

- Estrae y da Q
- Metti in Q e colora in grigio i vertici adiacenti a y
- Colora y in nero



Q

Cammini minimi

Dati due vertici s e v in un grafo non pesato, il numero minimo di archi su un cammino da s a v si definisce distanza sul cammino minimo.

La procedura BFS calcola le distanze sul cammino minimo per ogni vertice di un grafo (rispetto ad un vertice sorgente).

Visita in profondità

La visita in profondità (Depth-First Search, o DFS) segue un approccio opposto a BFS.

Ad ogni passo si visita un vertice adiacente all'ultimo vertice visitato.

Quando non ne esistono, si ritorna indietro all'ultimo vertice visitato che abbia vertici adiacenti non visitati, e li si visita.

La procedura DFS è normalmente implementata attraverso una procedura ricorsiva.

Pseudo-codice (1)

```
DSG(G)
for ogni vertice u ∈ V[G]
  do color[u] ← WHITE
     π[u] ← NIL
time ← 0
for ogni vertice u ∈ V[G]
  do if color[u] = WHITE
       then DFS-Visit(u)
```

Psedo-codice (2)

```
DFS-Visit(u)
    color[u] ← GRAY
    d[u] ← time ← time+1
    for v ∈ Adj[u]
        do if color[v]=WHITE
              then π[v] ← u
                   DFS-Visit(v)
    color[u] ← BLACK
    f[u] ← time ← time+1
```

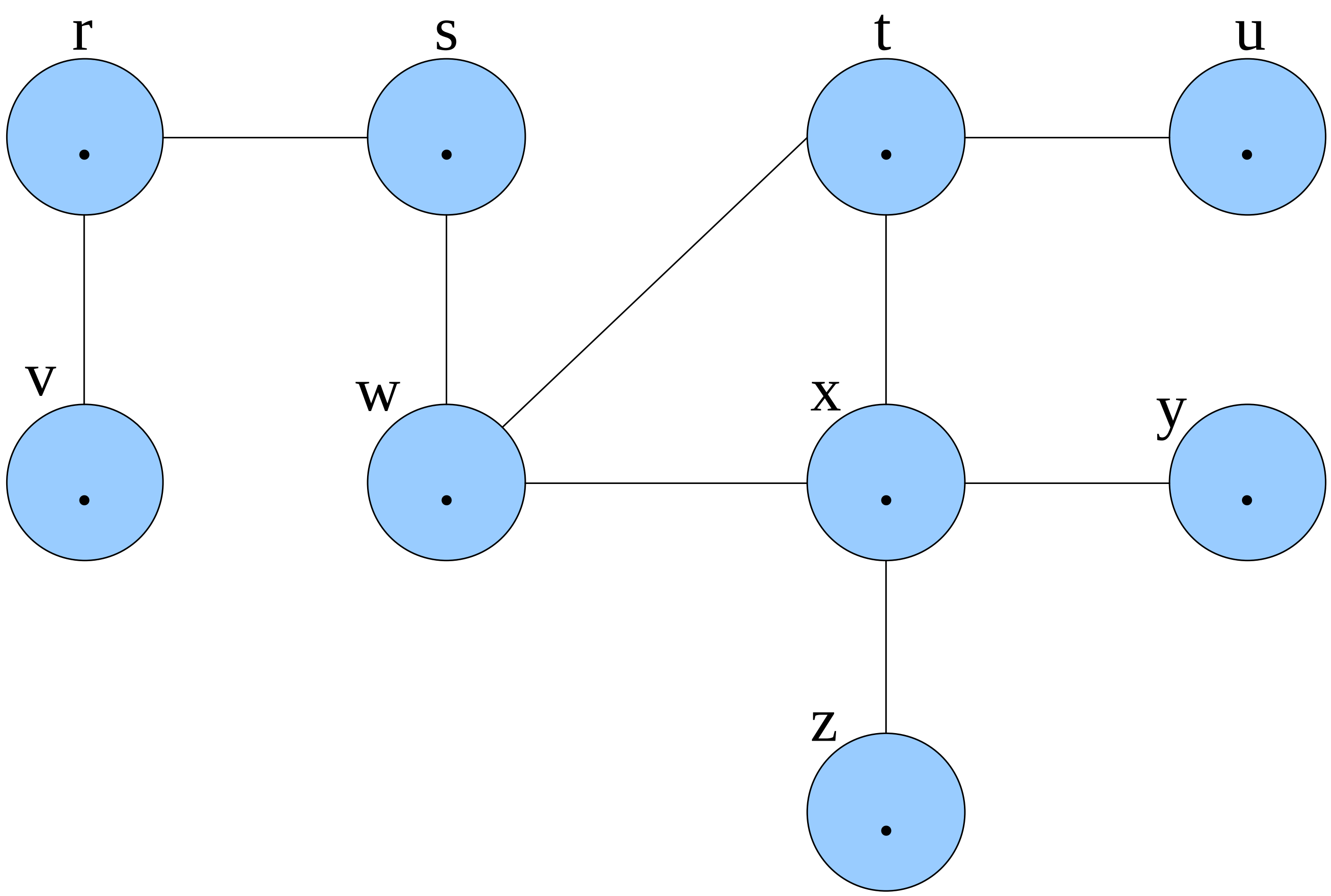
Nota

La colorazione dei vertici segue il seguente criterio:

- Inizialmente tutti i vertici sono bianchi
- Un vertice è colorato in grigio quando viene ragiunto per la prima volta
- Un vertice è colorato in nero quando tutti i vertici ad esso adiacenti sono stati visitati.

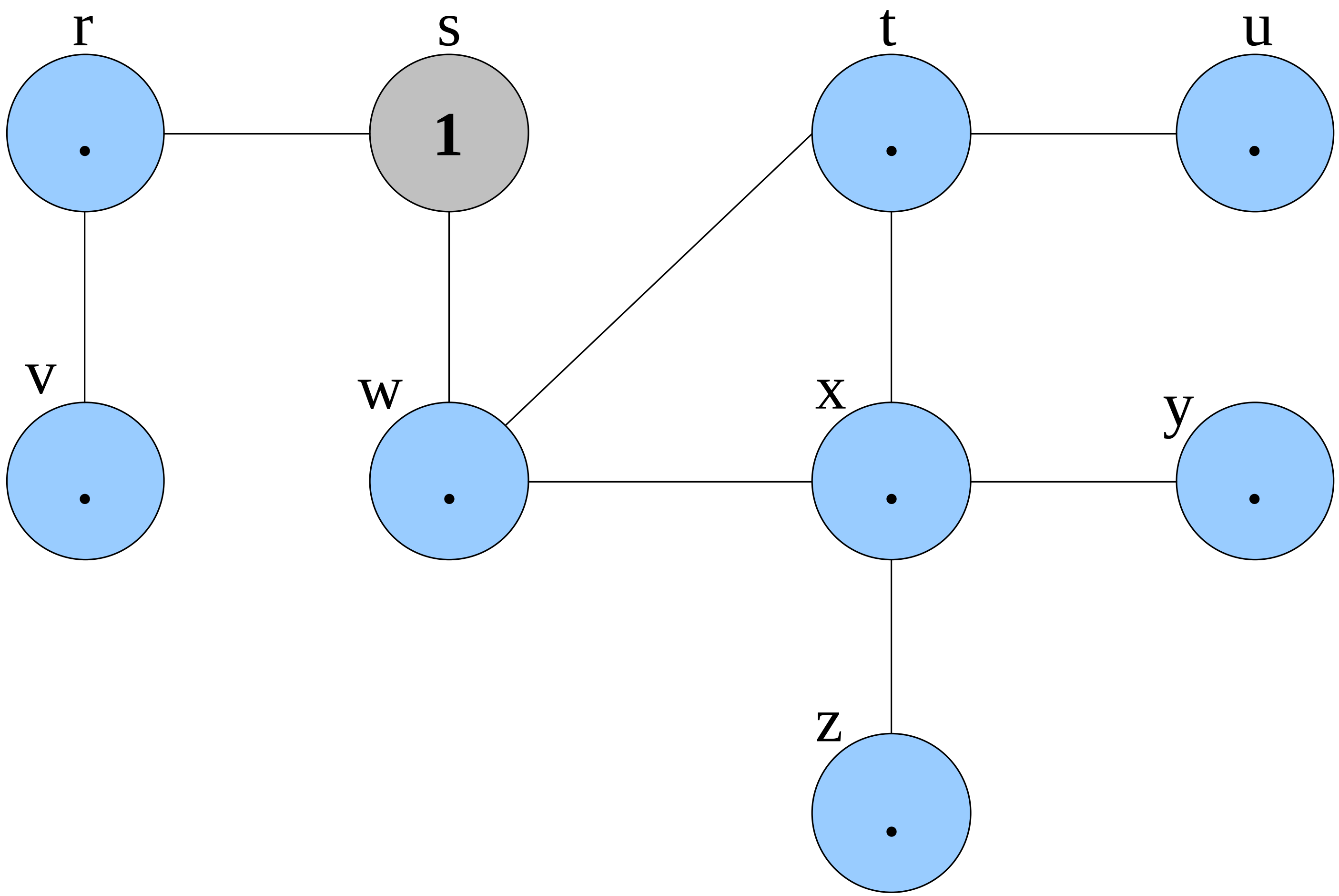
Esempio (0)

Vertice sorgente: s



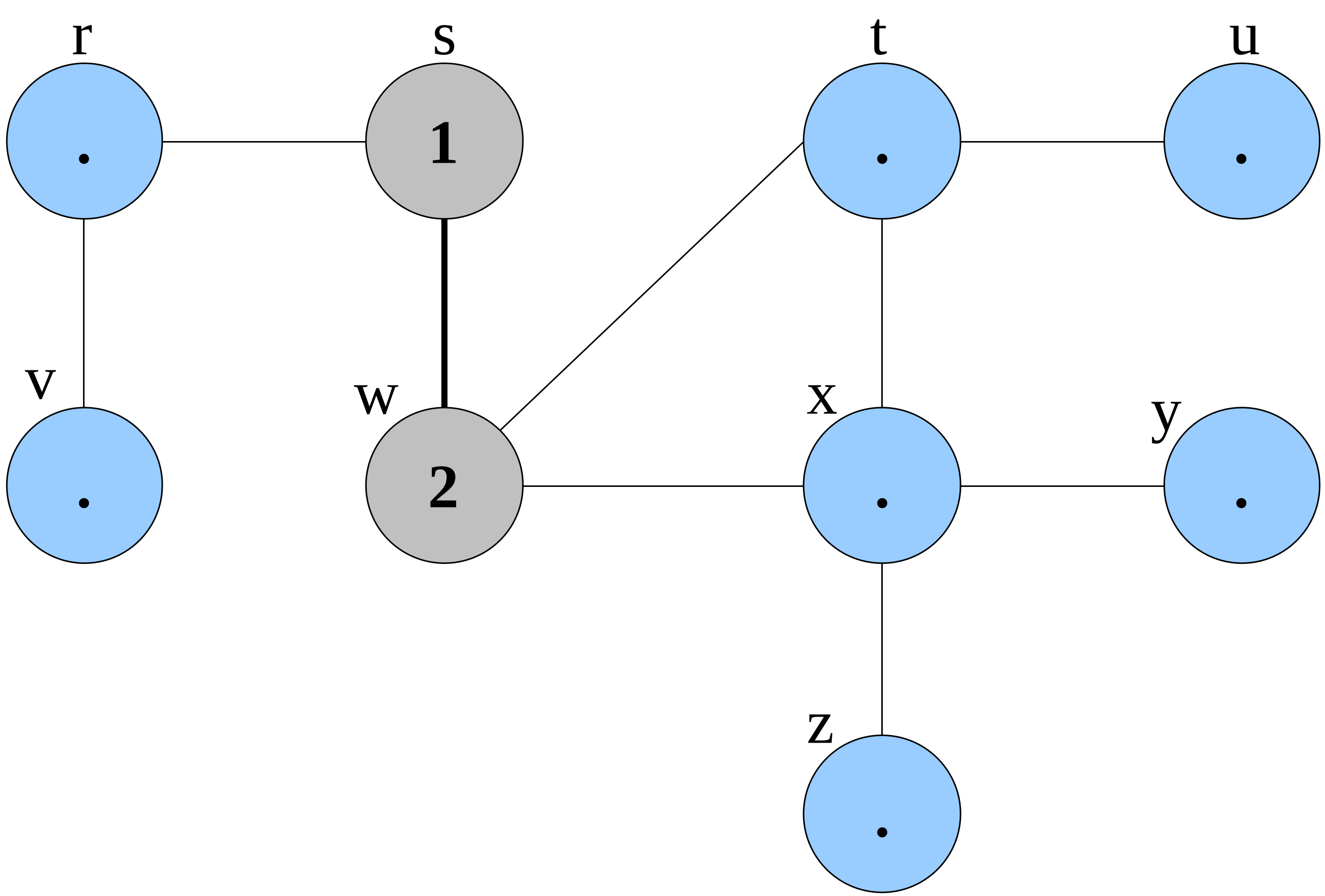
Esempio (1)

Visita s



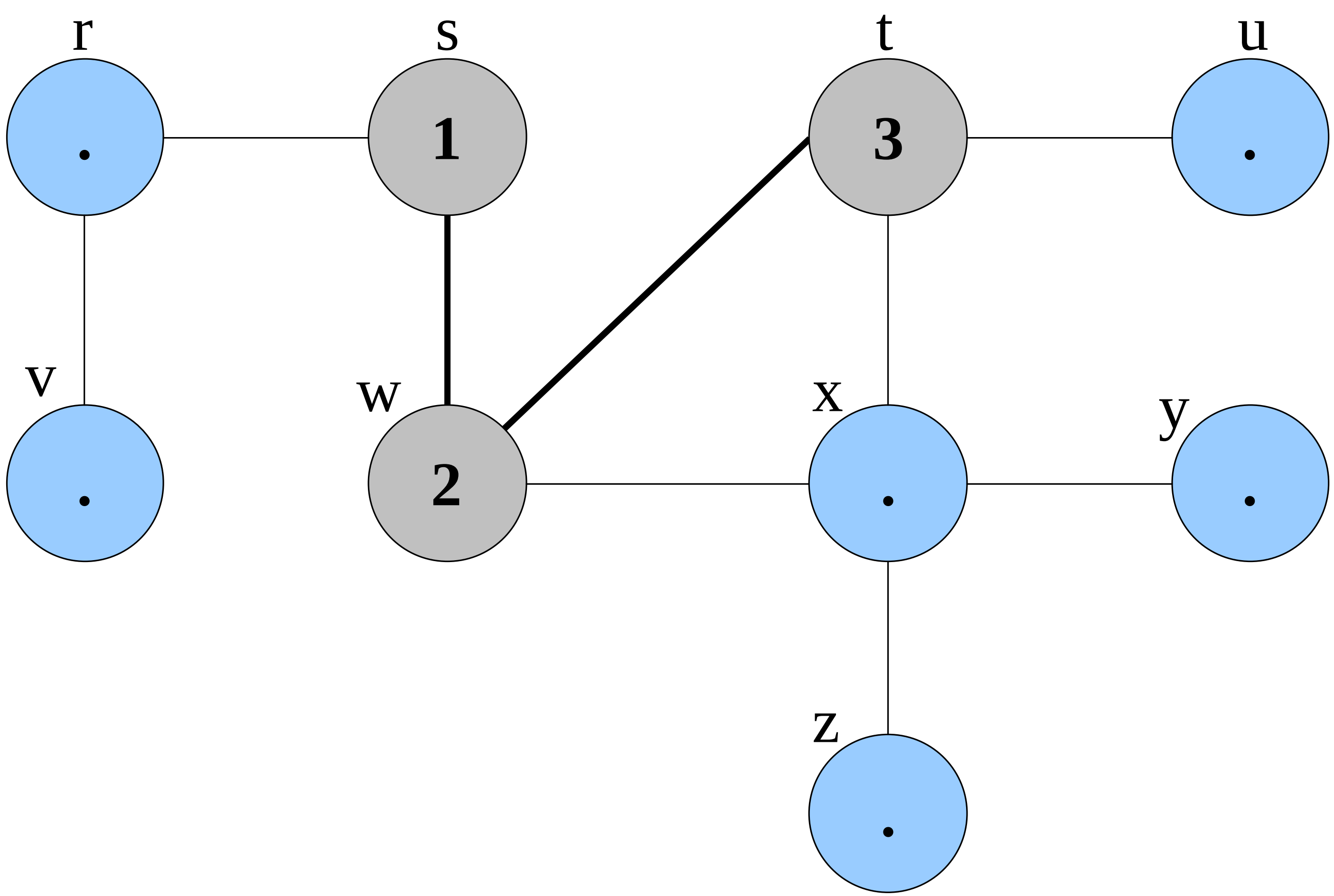
Esempio (2)

Visita w



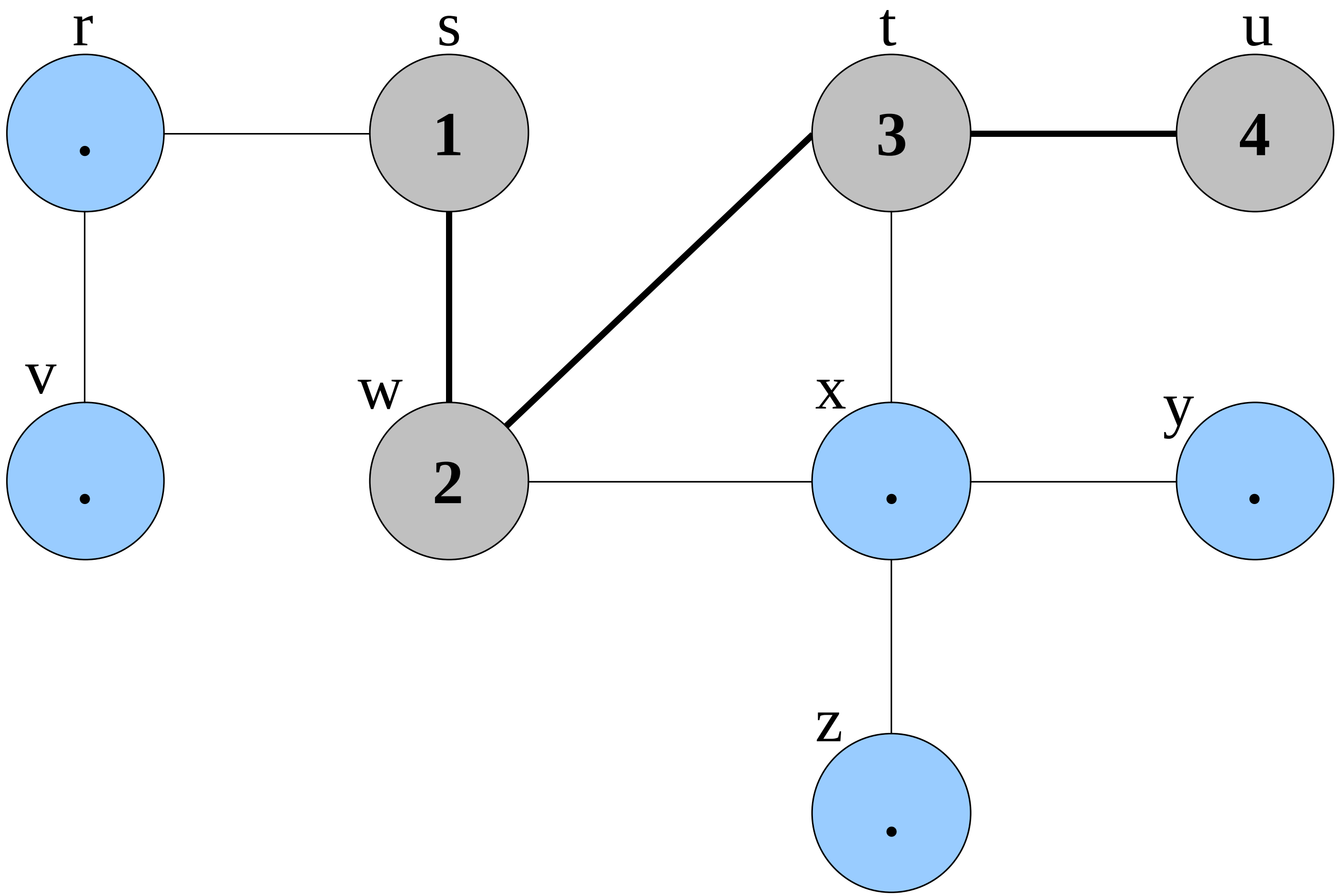
Esempio (3)

Visita t



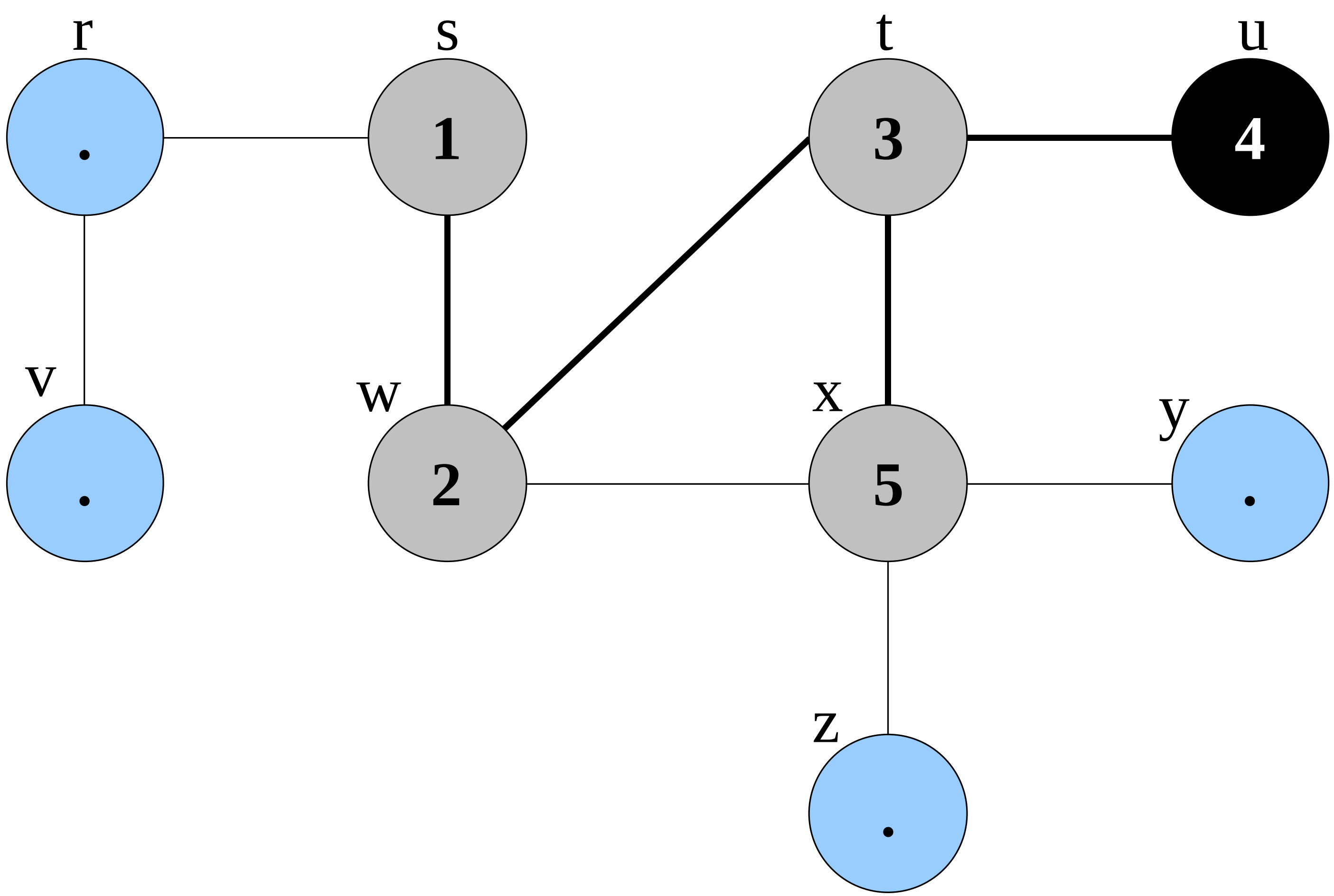
Esempio (4)

Visita u



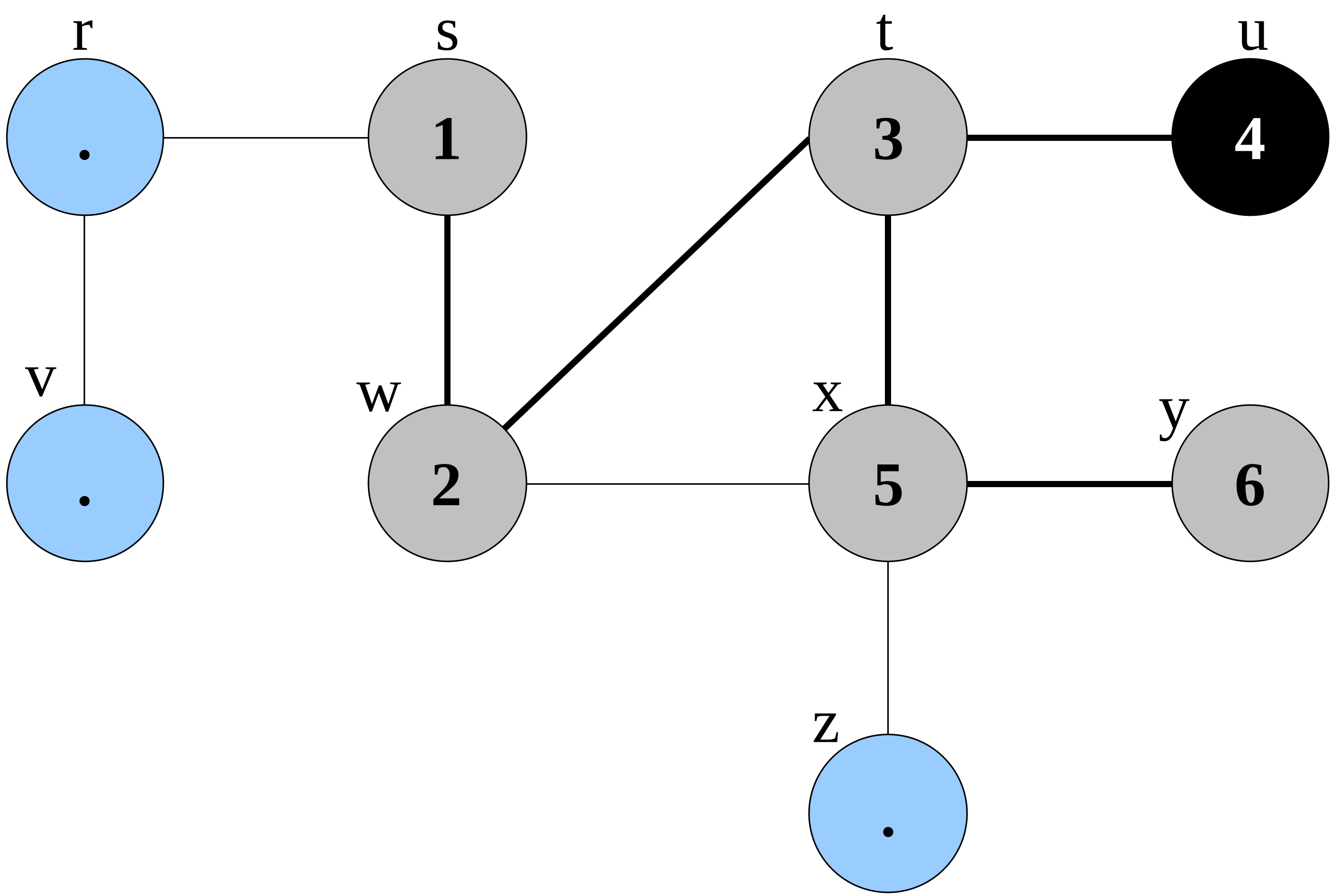
Visita x

Esempio (5)



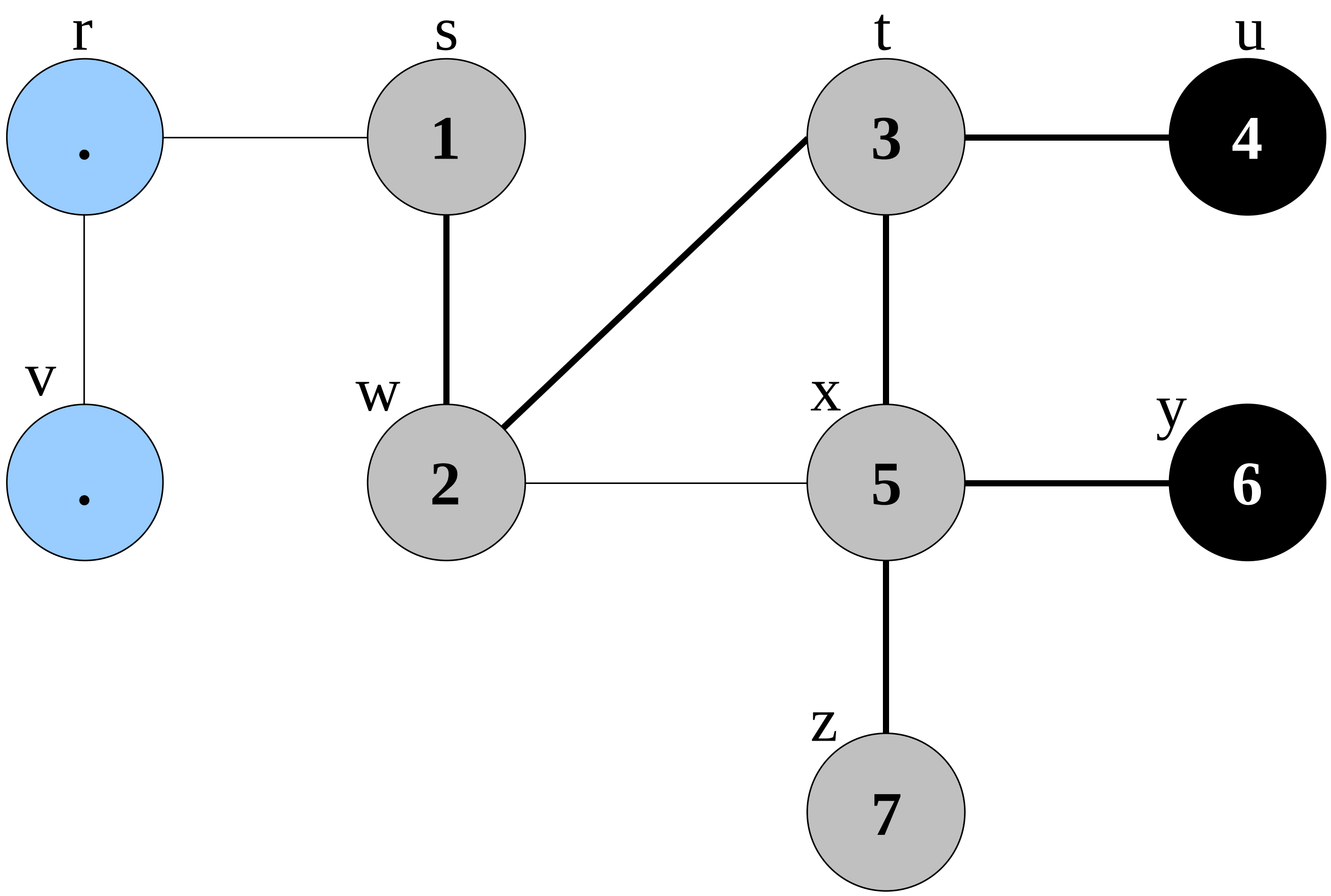
Esempio (6)

Visita y



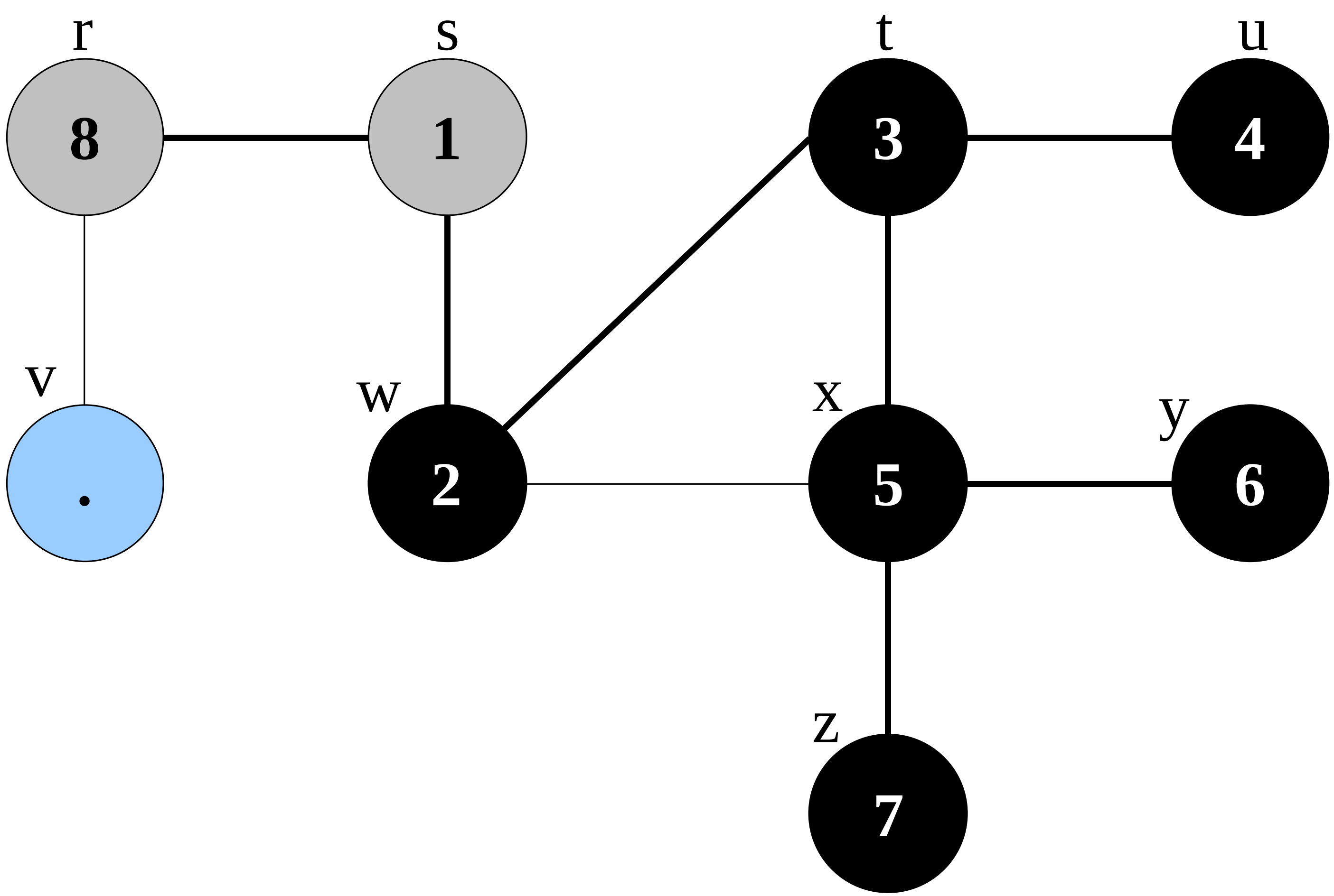
Esempio (7)

Visita z



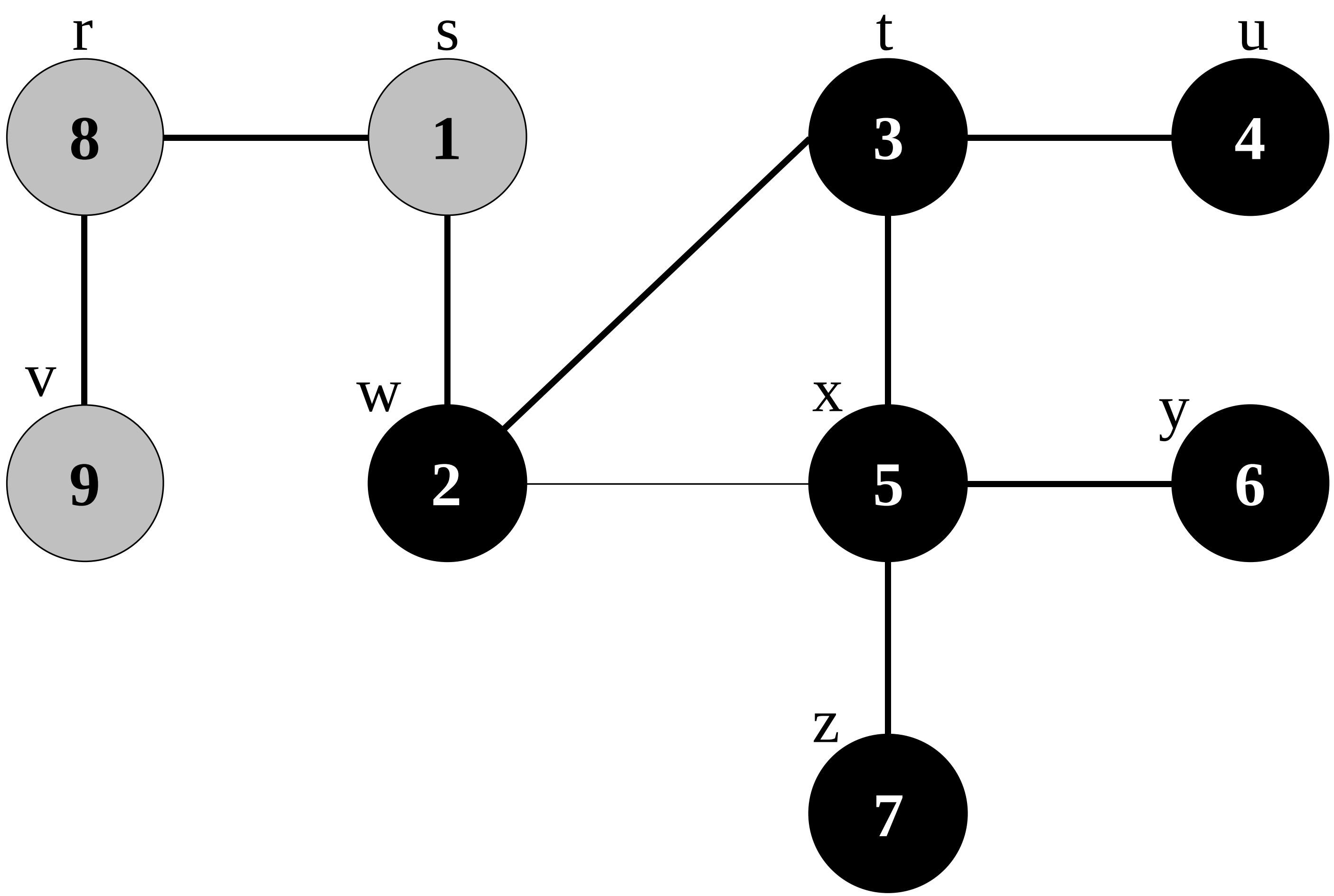
Esempio (8)

Visita r



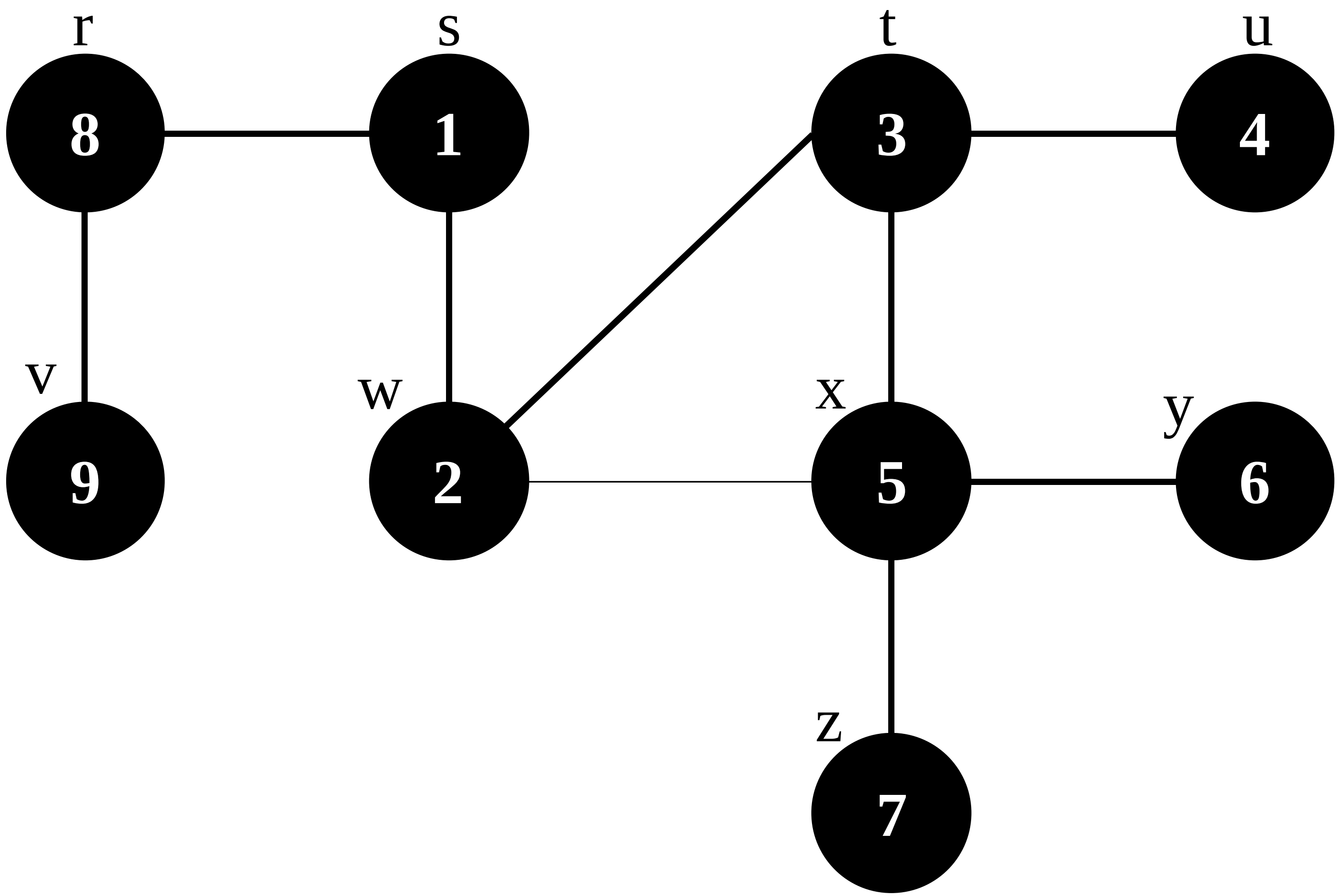
Esempio (9)

Visita v



Esempio (10)

Concludi visita



Foresta DFS

- La procedura DFS costruisce una foresta DFS, composta da uno o più alberi DFS.
- Gli archi che compongono la foresta sono detti archi dell’albero.

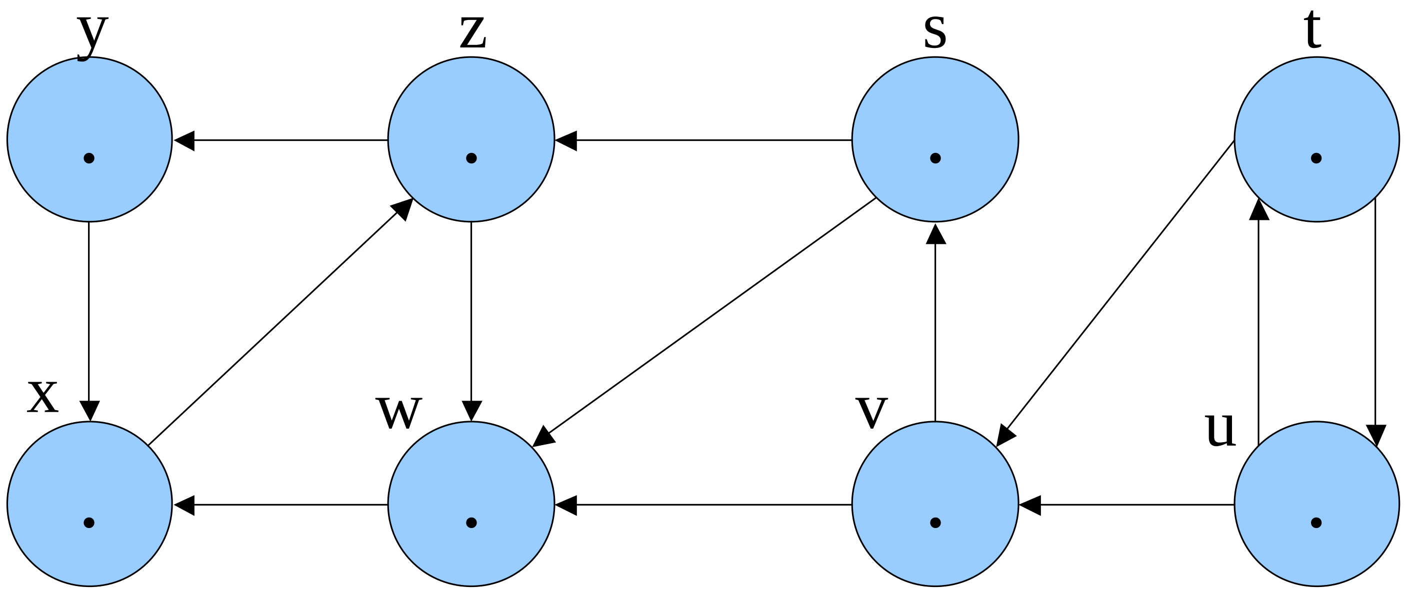
Classificazione degli archi

In un grafo orientato, gli archi possono essere raggruppati in 4 categorie:

- Archi dell'albero (T)
- Archi all'indietro (B): non sono archi T e connettono un vertice ad un suo antenato
- Archi in avanti (F): non sono archi T e connettono un vertice ad un suo discendente
- Archi di attraversamento (C): gli archi rimanenti.

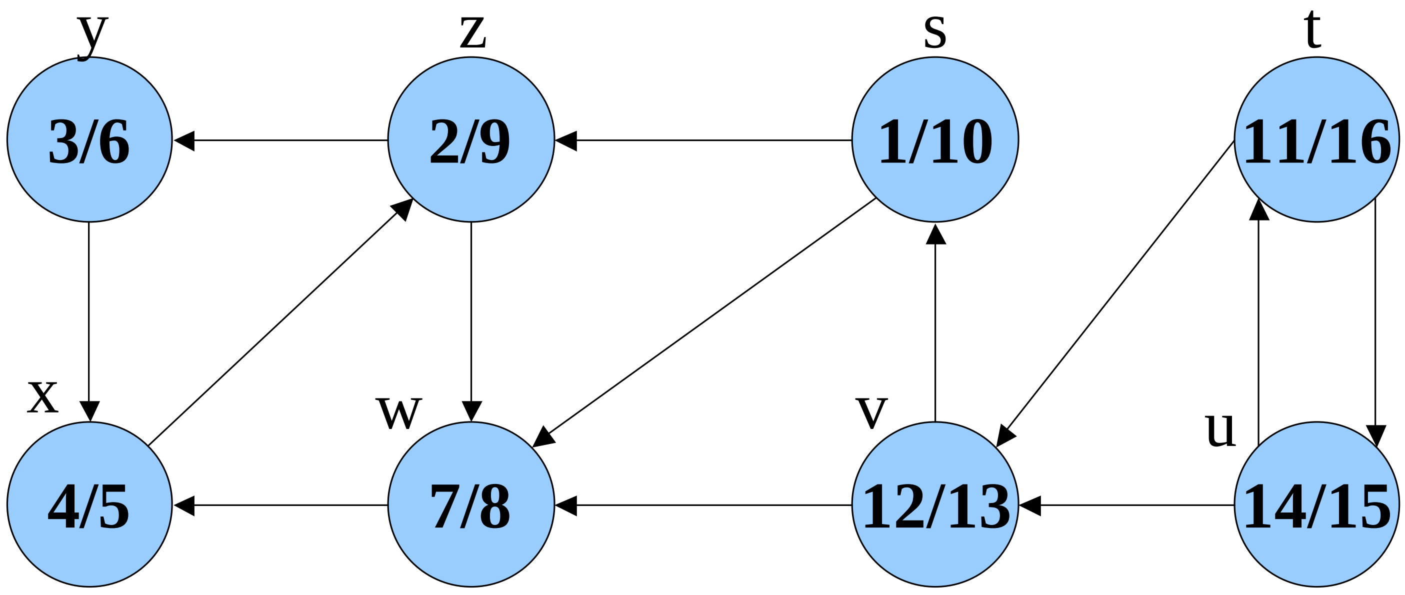
Esempio (0)

Vertice sorgente: s



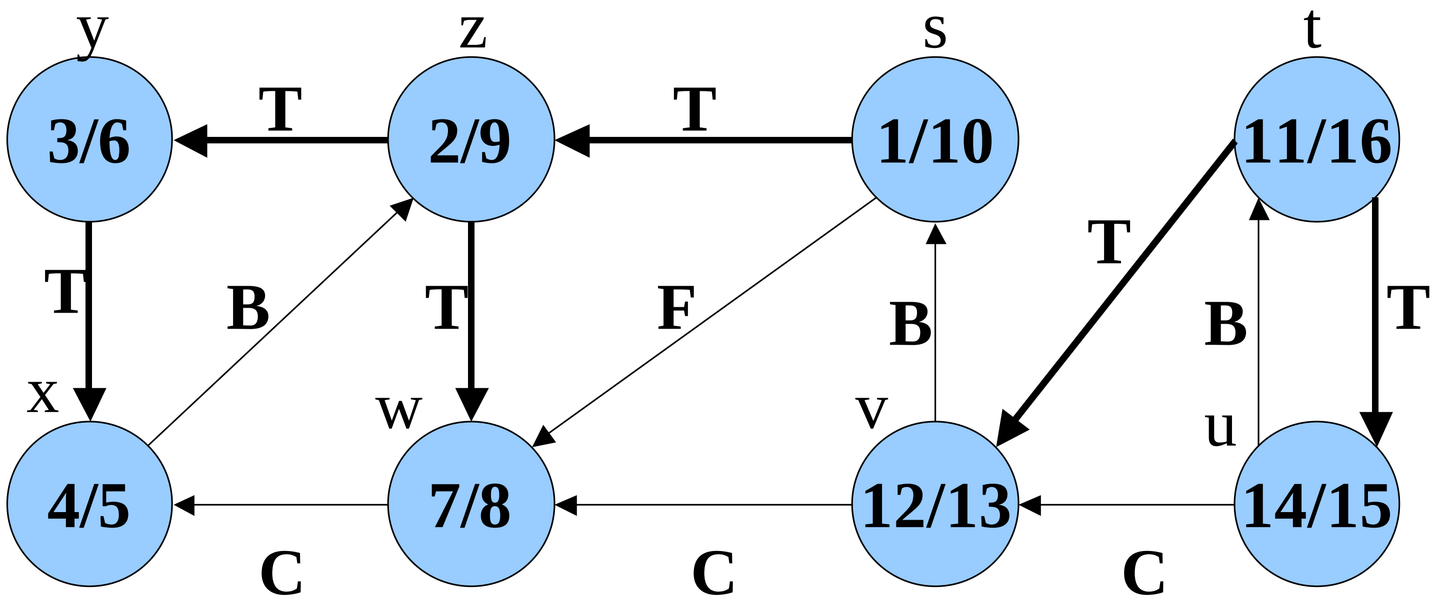
Esempio (1)

Visita in profondità



Esempio (2)

Classificazione degli archi



Classificazione degli archi

La procedura DFS può essere facilmente modificata in modo da classificare gli archi. Ogni volta che si incontra un arco (u,v) si considera il colore del vertice v in quel momento:

- Se è bianco l’arco è un arco dell’albero
- Se è grigio è un arco all’indietro
- Se è nero l’arco è un arco in avanti (se $d[u]<d[v]$) o un arco di attraversamento (se $d[u]>d[v]$).