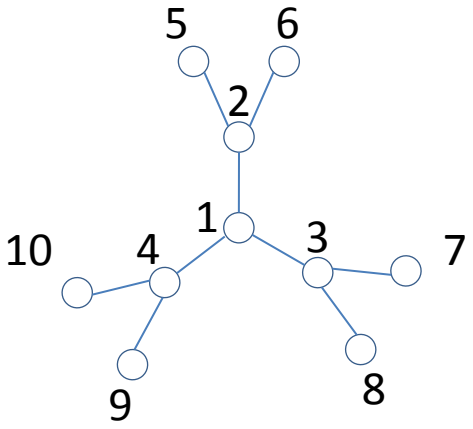
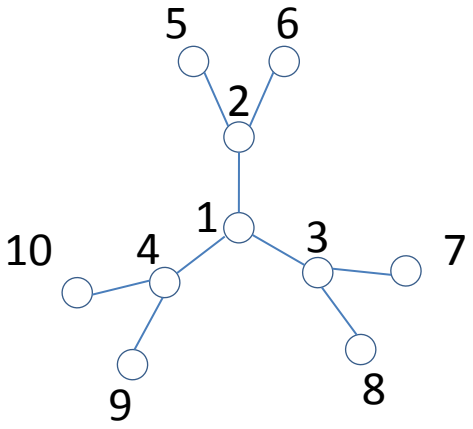


DIFFUSIONE SU GRAFO (rete) E RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE



Cammini aleatori su grafo (Random walk)

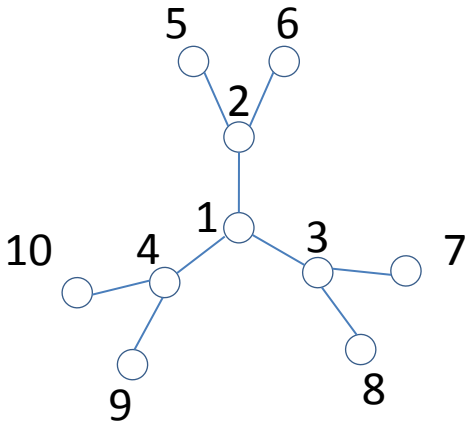
«processo di esplorazione casuale di un grafo mediante dei cammini che partono da uno o più nodi»

Un **grafo** è un oggetto che, in matematica, viene utilizzato per rappresentare una struttura composta da vertici (nodi) e archi (collegamenti tra nodi). In inglese arco (nel senso in cui viene usato nei grafi) si dice **edge**.

Matematicamente un grafo G si indica con:

G=(V,E) questo vuol dire che G è composto da un insieme di vertici (nodi) **V** e da un insieme di archi **E**.

V è un **vettore** (con un elemento per nodo)
E è una **matrice** v x v (se i nodi sono 10 allora è una matrice 10 x 10)



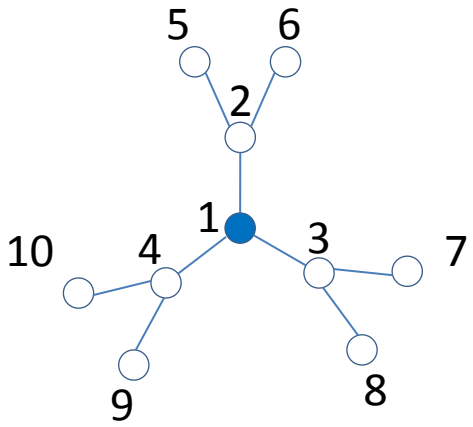

```
> I <- matrix(rep(0,100), nrow=10)
> I[1,2]=1
> I[2,1]=1
> I[1,3]=1
> I[3,1]=1
> I[1,4]=1
> I[4,1]=1
> I[2,5]=1
> I[5,2]=1
> I[2,6]=1
> I[6,2]=1
> I[3,7]=1
> I[7,3]=1
> I[3,8]=1
> I[8,3]=1
> I[4,9]=1
> I[9,4]=1
> I[4,10]=1
> I[10,4]=1
> sum(I)
[1] 18
>
```

```
> I
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    1    1    1    0    0    0    0    0     0
 [2,]    1    0    0    0    1    1    0    0    0     0
 [3,]    1    0    0    0    0    0    1    1    0     0
 [4,]    1    0    0    0    0    0    0    0    1     1
 [5,]    0    1    0    0    0    0    0    0    0     0
 [6,]    0    1    0    0    0    0    0    0    0     0
 [7,]    0    0    1    0    0    0    0    0    0     0
 [8,]    0    0    1    0    0    0    0    0    0     0
 [9,]    0    0    0    1    0    0    0    0    0     0
[10,]    0    0    0    1    0    0    0    0    0     0
```

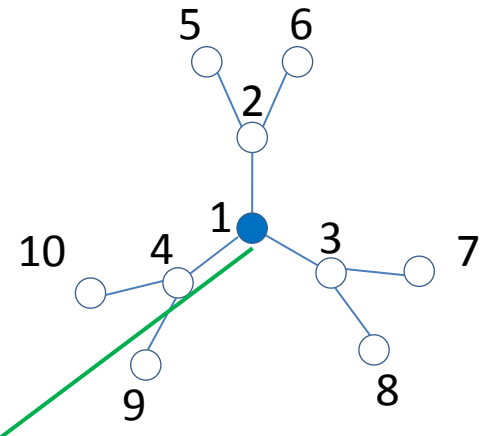
Prob.norm() del package **NetPreProc** normalizza una matrice in modo che la somma dei valori sulle **righe** sia 1 .

```
> library(NetPreProc)
> Prob.norm(I)
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333
 [5,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
> Q <- Prob.norm(I)
>
```

```
> I
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    1    1    1    0    0    0    0    0     0
 [2,]    1    0    0    0    1    1    0    0    0     0
 [3,]    1    0    0    0    0    0    1    1    0     0
 [4,]    1    0    0    0    0    0    0    0    1     1
 [5,]    0    1    0    0    0    0    0    0    0     0
 [6,]    0    1    0    0    0    0    0    0    0     0
 [7,]    0    0    1    0    0    0    0    0    0     0
 [8,]    0    0    1    0    0    0    0    0    0     0
 [9,]    0    0    0    1    0    0    0    0    0     0
[10,]    0    0    0    1    0    0    0    0    0     0
> Q <- Prob.norm(I)
> Q
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]     [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333
 [5,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
> t(Q)
           [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000    1    1    0    0    0     0
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000    0    0    1    1    0     0
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000    0    0    0    0    1     1
 [5,] 0.0000000 0.3333333 0.0000000 0.0000000    0    0    0    0    0     0
 [6,] 0.0000000 0.3333333 0.0000000 0.0000000    0    0    0    0    0     0
 [7,] 0.0000000 0.0000000 0.3333333 0.0000000    0    0    0    0    0     0
 [8,] 0.0000000 0.0000000 0.3333333 0.0000000    0    0    0    0    0     0
 [9,] 0.0000000 0.0000000 0.0000000 0.3333333    0    0    0    0    0     0
[10,] 0.0000000 0.0000000 0.0000000 0.3333333    0    0    0    0    0     0
> probvector <- rep(0,10)
> probvector[1]<-1
> probvector
 [1] 1 0 0 0 0 0 0 0 0 0
>
```



```
> I
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    1    1    1    0    0    0    0    0     0
 [2,]    1    0    0    0    1    1    0    0    0     0
 [3,]    1    0    0    0    0    0    1    1    0     0
 [4,]    1    0    0    0    0    0    0    0    1     1
 [5,]    0    1    0    0    0    0    0    0    0     0
 [6,]    0    1    0    0    0    0    0    0    0     0
 [7,]    0    0    1    0    0    0    0    0    0     0
 [8,]    0    0    1    0    0    0    0    0    0     0
 [9,]    0    0    0    1    0    0    0    0    0     0
[10,]    0    0    0    1    0    0    0    0    0     0
> Q <- Prob.norm(I)
> Q
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333
 [5,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
> t(Q)
           [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000    1    1    0    0    0     0
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000    0    0    1    1    0     0
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000    0    0    0    0    1     1
 [5,] 0.0000000 0.3333333 0.0000000 0.0000000    0    0    0    0    0     0
 [6,] 0.0000000 0.3333333 0.0000000 0.0000000    0    0    0    0    0     0
 [7,] 0.0000000 0.0000000 0.3333333 0.0000000    0    0    0    0    0     0
 [8,] 0.0000000 0.0000000 0.3333333 0.0000000    0    0    0    0    0     0
 [9,] 0.0000000 0.0000000 0.0000000 0.3333333    0    0    0    0    0     0
[10,] 0.0000000 0.0000000 0.0000000 0.3333333    0    0    0    0    0     0
> probvector <- rep(0,10)
> probvector[1]<-1
> probvector
 [1] 1 0 0 0 0 0 0 0 0 0
>
```



```
> Q
      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333
 [5,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
> p0 %*% Q
     [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
```

FASE 1 : per ogni riga moltiplico il valore presente in p0 per l'intera riga di Q. Ottengo una matrice.

FASE 2 : per ogni colonna della matrice ottenuta in FASE 1 faccio la somma. Ottengo un valore per ogni colonna … quindi un **vettore**.

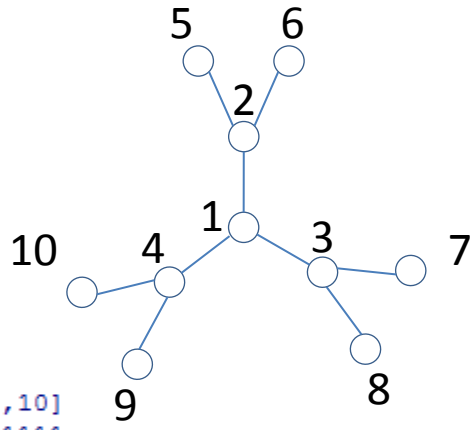
In R :

vettore %*% matrice

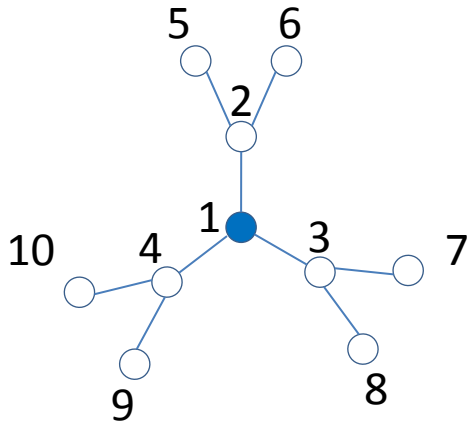
La somma del vettore deve essere 1

Deve essere normalizzata con Prob.norm()

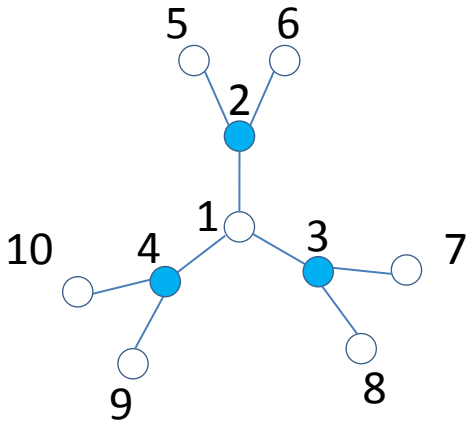
```
> probvector <- rep(0,10)
> probvector[1]<-1
> probvector
 [1] 1 0 0 0 0 0 0 0 0 0
>
> p0 <- probvector
> p0 %*% Q
     [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
> p1 <- p0 %*% Q
> p1 %*% Q
          [,1] [,2] [,3] [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
[1,] 0.3333333    0    0    0 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111
> p2 <- p1 %*% Q
> sum(p0)
[1] 1
> sum(p1)
[1] 1
> sum(p2)
[1] 1
>
```



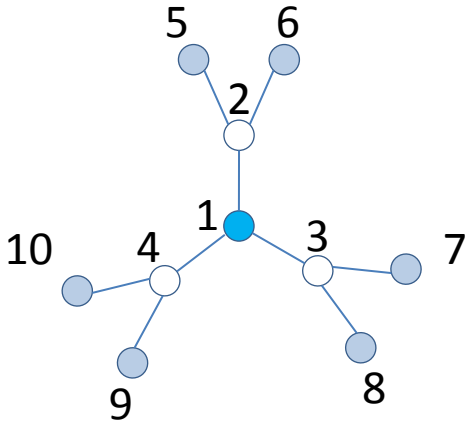
p0



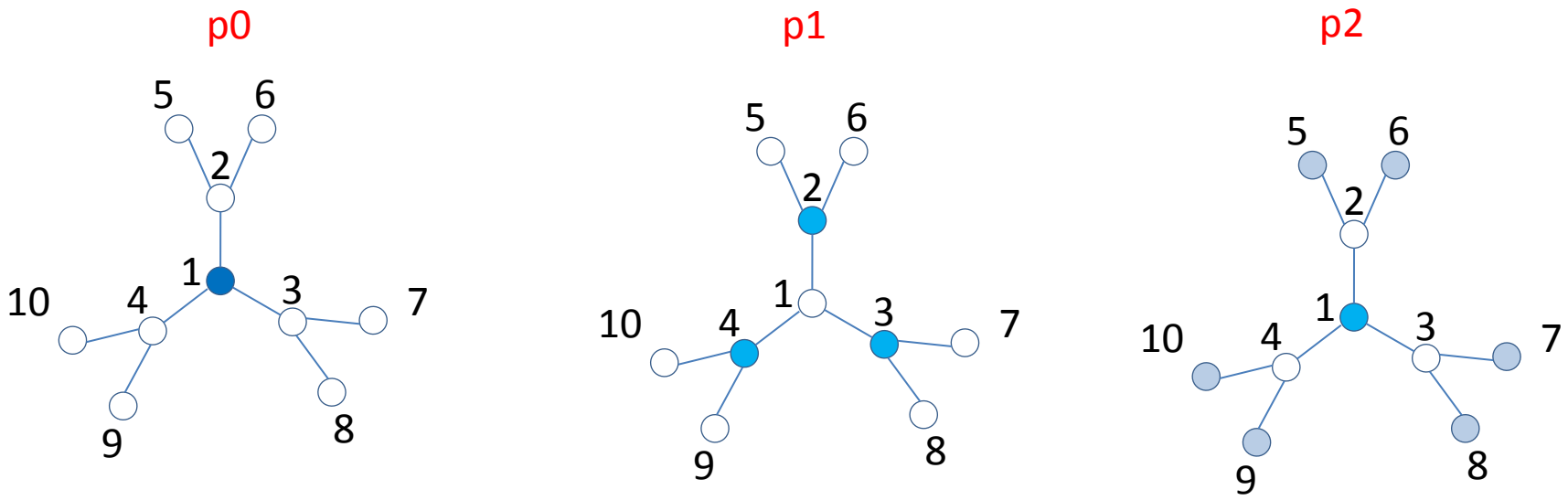
p1



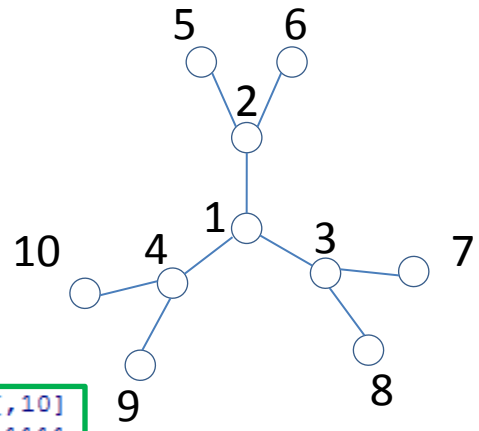
p2

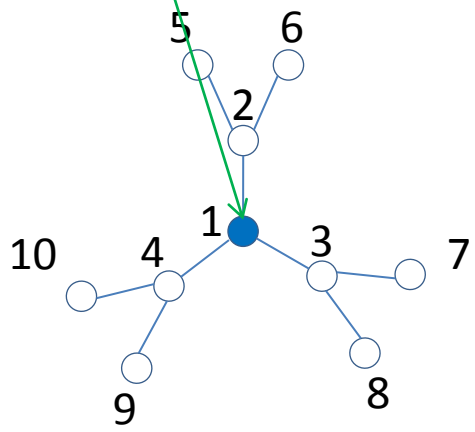


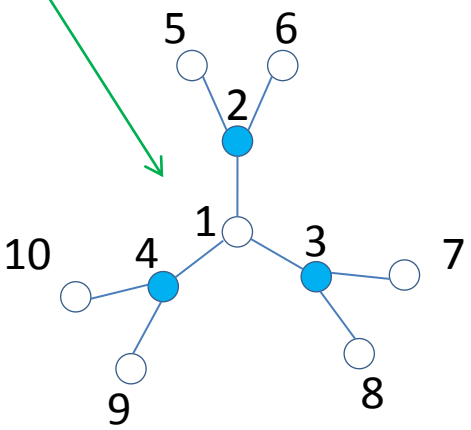
```
> p0
 [1] 1 0 0 0 0 0 0 0 0 0
> Q
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]     [,10]
 [1,] 0.0000000 0.3333333 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.3333333 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333 0.0000000 0.0000000
 [4,] 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333 0.3333333
 [5,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
> p0 %*% Q
     [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
> p1
     [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
> p1 %*% Q
          [,1] [,2] [,3] [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
[1,] 0.3333333    0    0    0 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111
>
```

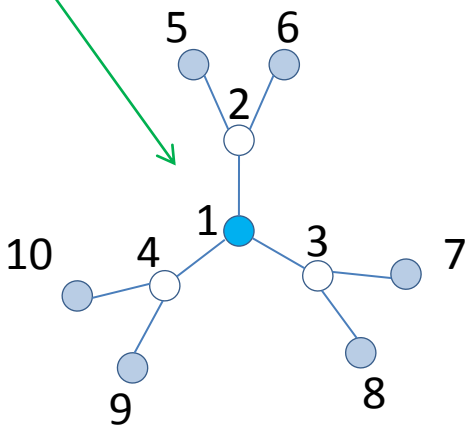


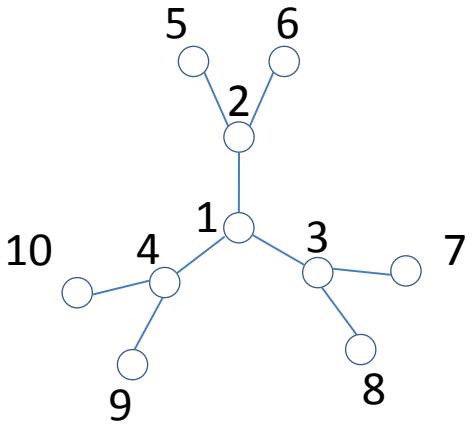
```
> probvector <- rep(0,10)
> probvector[1]<-1
> probvector
 [1] 1 0 0 0 0 0 0 0 0 0
>
> p0 <- probvector
> p0 %*% Q
     [,1]      [,2]      [,3]      [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0 0.3333333 0.3333333 0.3333333    0    0    0    0    0     0
> p1 <- p0 %*% Q
> p1 %*% Q
          [,1] [,2] [,3] [,4]      [,5]      [,6]      [,7]      [,8]      [,9]      [,10]
[1,] 0.3333333    0    0    0 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111
> p2 <- p1 %*% Q
> sum(p0)
[1] 1
> sum(p1)
[1] 1
> sum(p2)
[1] 1
>
```







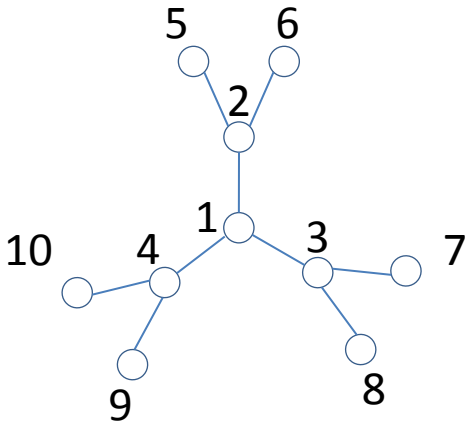




RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE

Come si collega quello che abbiamo detto sui random walk con il riposizionamento di farmaci?

Riposizionare significa trovare farmaci **simili** ad altri farmaci che hanno **proprietà note** (ad esempio l'efficacia nel trattamento di una determinata patologia). Il punto cruciale è avere un insieme di molecole per le quali siamo in grado di definire una qualche nozione di simi-larità (es. similarità struttura molecolare) ed una serie di proprietà per ciascuna di esse.

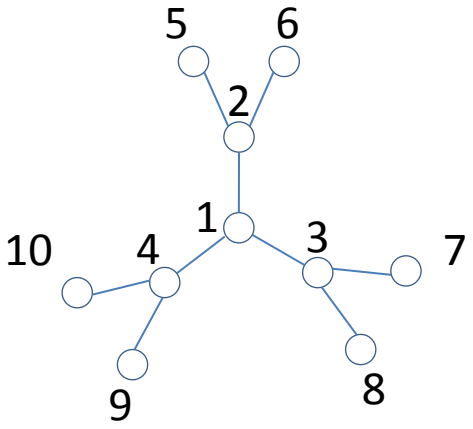


RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE

Quali informazioni abbiamo sui farmaci approvati da FDA?

- Una matrice di similarità che esprime, per ogni possibile coppia di molecole quanto sono simili → MATRICE CHE RAPPRESENTA GLI **ARCHI** DI UN **GRAFO** (in esso i farmaci sono i nodi)

- Una matrice che ci dice, per ogni farmaco, a quali categorie terapeutiche appartengono. l'appartenenza o meno a una data categoria terapeutica **è una proprietà delle molecole**.



NB: un vettore costruito così indica che l'inizio dei cammini aleatori avverrà con la stessa probabilità sui nodi 2 e 3.

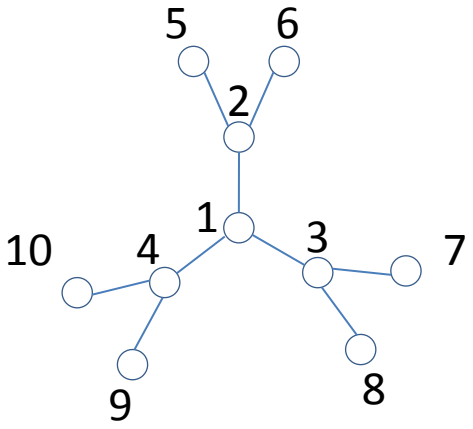
RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE

Quali informazioni abbiamo sui farmaci approvati da FDA?

- La matrice di similarità va normalizzata con Prob.norm()

- Prendo una **colonna** (vettore) della matrice delle categorie terapeutiche. Supponiamo (solo per fare un esempio) che i farmaci siano 10. (e gli elementi del vettore, quindi, 10). Poi supponiamo che le penicilline siano gli elementi **2** e **3** del vettore (ossia i nodi 2 e 3 del grafo) . possiamo impostare il vettore iniziale delle probabilità in questo modo:

> p0.penicilline <- c(0, **0.5, 0.5**, 0, 0, 0, 0, 0, 0, 0)



RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE

Dopo aver convertito la matrice di similarità tra farmaci in una matrice delle probabilità di transizione (Q) ed aver moltiplicato il vettore p0.penicilline per la matrice (usando %*%) otteniamo un vettore in cui i valori esprimono, per ogni molecola (ossia per ogni nodo del grafo) la **probabilità di appartenere alla categoria terapeutica «penicilline»**.

La categoria terapeutica del risultato è definita da quale colonna della matrice delle etichette (1 0) abbiamo usato per costruire il vettore p0 .

RIPOSIZIONAMENTO DI FARMACI SULLA BASE DELLA SIMILARITA' DI STRUTTURA MOLECOLARE

Dopo aver convertito la matrice di similarità tra farmaci in una matrice delle probabilità di transizione (Q) ed aver moltiplicato il vettore p0.penicilline per la matrice (usando %*%) otteniamo un vettore in cui i valori esprimono, per ogni molecola (ossia per ogni nodo del grafo) la **probabilità di appartenere alla categoria terapeutica «penicilline»**.

La categoria terapeutica del risultato è definita da quale colonna della matrice delle etichette (1 0) abbiamo usato per costruire il vettore p0 .

INTRODUZIONE PROGETTI ESAME

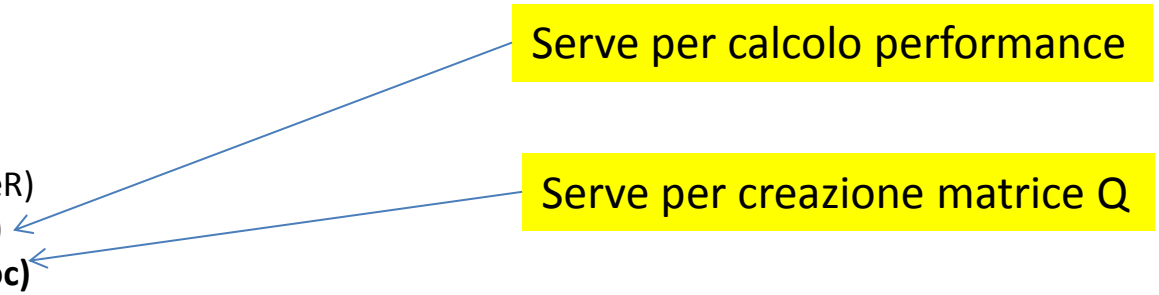
Il progetto d'esame consiste nell'effettuare un'analisi di riposizionamento di farmaci basata su rete di similarità tra molecole.
Il set di farmaci che utilizzeremo sono quelli utilizzati durante le esercitazioni in laboratorio.
I file necessari per l'esame sono disponibili a questo indirizzo:

http://homes.di.unimi.it/re/Corsi/Dati_Esame_Info16/

Creare una directory sul desktop e assegnare ad essa il nome **test**

Copiate i file scaricati dall'indirizzo sopra riportato al suo interno. Poi aprite R.

INTRODUZIONE PROGETTI ESAME




```
library(rcdk)
library(ChemmineR)
library(PerfMeas)
library(NetPreProc)

setwd("Z:\\Desktop\\test")          # ci si posiziona in un folder sul desktop contenente materiale esercitazione
                                    # questo è uno dei momenti critici. Se provi a testare comandi puoi, per favore,
                                    # inviarmi la stringa del path che ustilizzi? Giusto per sicurezza

dir()                               # Devono esserci i file "DDsimData.rda" e "RWR.R"
load("DDsimData.rda")
ls()                                # Devono esserci le variabili "DD.chem.data" "DrugBank.Cat"
```

INTRODUZIONE PROGETTI ESAME

Inizio esercitazione : riposizionamento di farmaci mediante random walk su reti farmacologiche

Creazione matrice di transizione

```
Q <- Prob.norm(DD.chem.data)
dim(Q)
```

Visualizzazione categorie terapeutiche farmaci:

```
dimnames(DrugBank.Cat)[[2]]
```

INTRODUZIONE PROGETTI ESAME

Come ho trovato questo indice?

Quante sono le penicilline nel dataset?

Questo file contiene le funzioni che useremo per l'esame. E' codice R. Per caricare usiamo source()

```
# ========  PENICILLINE ========================================================

# Costruzione vettore etichette (categoria terapeutica: Penicilline)

labels <- DrugBank.Cat[,41]
sum(labels)
which(labels==1)
indpos <- which(labels==1)

# Caricamento funzioni da file sorgente
source("RWR.R")

# TEST 1: random walk su rete farmacologica (categoria terapeutica: penicilline)
# (tipo random walk: random walk con restart 60% )

rw.penicilline <- RWR(W=Q, ind.positives=indpos, norm=FALSE)
str(rw.penicilline)
AUC.single(rw.penicilline$p, labels)                    # risultato?


# TEST 2: random walk su rete farmacologica (categoria terapeutica: penicilline)
# (tipo random walk: random walk con restart 60% )
# numero folds per cross validazione: 5

rw.penicilline.cv5 <- RW.cv(W=Q, ind.pos=indpos, fun=RWR, k=5, norm=FALSE)
str(rw.penicilline.cv5)
AUC.single(rw.penicilline.cv5, labels)                  # risultato?
```

INTRODUZIONE PROGETTI ESAME

Come ho trovato questo indice?

N. Antiviral agents nel dataset?

```
# ========  ANTI VIRAL AGENTS ================================================

# TEST 1b: random walk su rete farmacologica (categoria terapeutica: penicilline)
# (tipo random walk: random walk con restart 60% )

labels <- DrugBank.Cat[,23]
indpos <- which(labels==1)

rw.ava <- RWR(W=Q, ind.positives=indpos, norm=FALSE)
str(rw.ava)
AUC.single(rw.ava$p, labels)                                    # risultato?

# TEST 2b: random walk su rete farmacologica (categoria terapeutica: penicilline)
# (tipo random walk: random walk con restart 60% )
# numero folds per cross validazione: 5

rw.ava.cv5 <- RW.cv(W=Q, ind.pos=indpos, fun=RWR, k=5, norm=FALSE)
str(rw.ava.cv5)
AUC.single(rw.ava.cv5, labels)                                  # risultato?
```