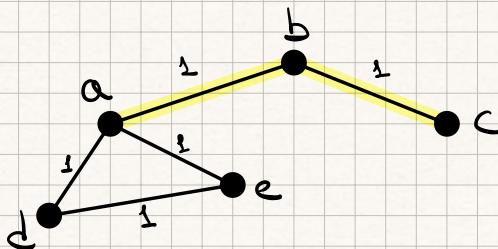


## METRICHE

### 1) GEODESIC DISTANCE

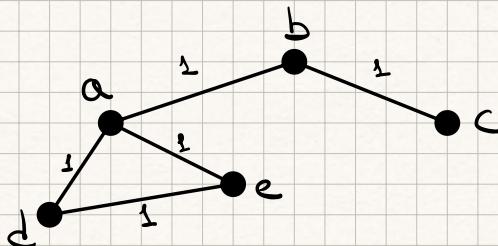
Per due vertici la geodesic distance è il percorso più breve che li unisce.



$$\text{geodesicD}(a, c) = 2$$

### 2) ECCENTRICITY OF A VERTICE $v$

È la massima Geodesic Distance fra  $v$  e tutti gli altri vertici



$$\text{geodesicD}(a, b) = 1$$

$$\text{geodesicD}(a, c) = 2$$

$$\text{geodesicD}(a, d) = 1$$

$$\text{geodesicD}(a, e) = 1$$

$$\text{Eccent}(a) = 2$$

### 3) RAGGIO

Fra tutte le Eccent( $v_i$ )  $\forall v_i \in \text{Grafo}$ , prendo la minore

### 3) DIAMETRO

Fra tutte le Eccent( $v_i$ )  $\forall v_i \in \text{Grafo}$ , prendo la maggiore. I vertici che vengono uniti dal diametro sono detti vertici periferici (peripheral).

# SIMILARITÀ NEI GRAFI

## 1) STRUCTURAL CONTEXT-BASED SIMILARITY

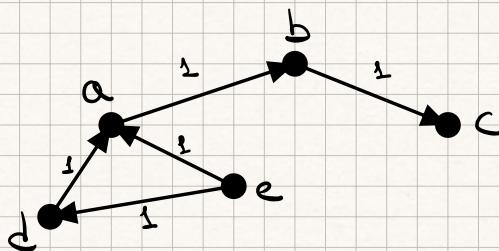
Due nodi sono considerati simili se hanno un vicinato simile.

### SIMRANK

È un tipo di similarità structural-context

Definiamo in un grafo diretto  $G = (V, \delta)$

- 1) Individual in-neighborhood of  $v$ :  $I(v) = \{u \mid (v, u) \in \delta\}$
- 2) Individual out-neighborhood of  $v$ :  $O(v) = \{w \mid (v, w) \in \delta\}$



$$I(a) = \{d, e\}$$

$$O(a) = \{b\}$$

Il SimRank è definito come:

$$S(u, v) = \frac{C}{|I(u)| |I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} S(x, y)$$

Dove  $C$  è una costante fra 0 e 1.

Se un vertice non ha vicini allora  $S(u, v) = 0$

**Come calcolare il SimRank?** Procedura iterativa

In modo iterativo. A ogni iterazione ci cerchiamo  $\lambda^2$  valori  $S_i(u, v)$ , uno per ogni coppia di vertici.

Nella prima iterazione cerchiamo  $S_0$ :

$$1) S_0(u, v) = \begin{cases} 0 & u \neq v \\ 1 & u = v \end{cases}$$

Nelle iterazioni successive:

$$2) \begin{cases} s_{i+1}(u, v) = \frac{c}{|I(u)||I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} s_i(x, y) & \text{per } u \neq v \\ s_{i+1}(u, v) = 1 & \text{per } u = v \end{cases}$$

Note:  $A^2$  cresce di  $c$ ,  $s_i(u,v)$  non decresce.

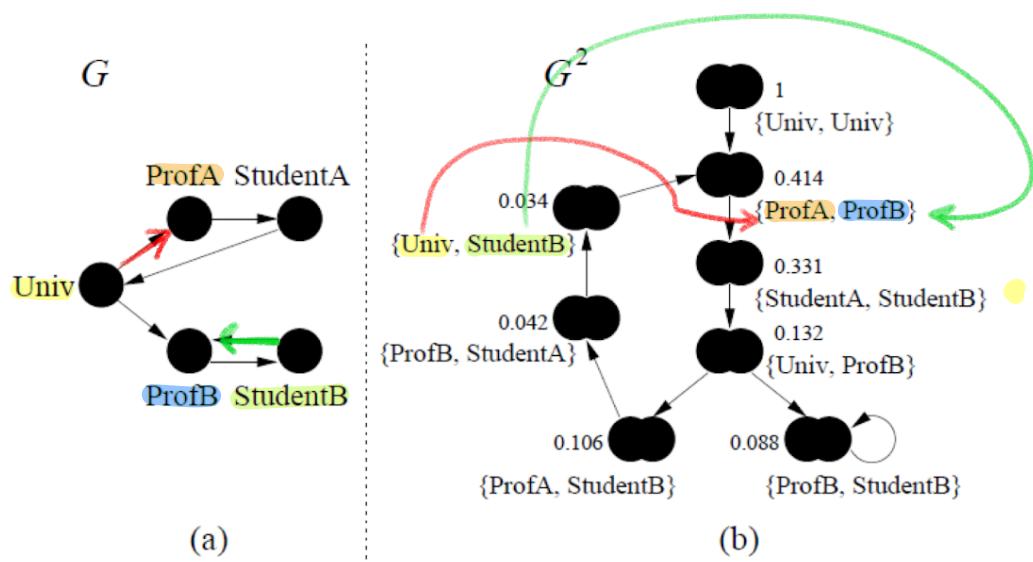
COMPLICATITÀ

$O(KM^2d_2)$  dove  $d$  è la media dei valori  $|I(u)||I(v)|$ . e dove  $K$  è il numero delle iterazioni.

Grafo  $G^2$

Un nodo in  $G^2$  è formato da una coppia di nodi in  $G$ , secondo le seguenti regole:

in  $G^2$ :  $(a,b) \rightarrow (c,d)$  se in  $G$ :  $a \rightarrow c$  e  $b \rightarrow d$ .



## Intuizione della formula $s_i(u, v)$

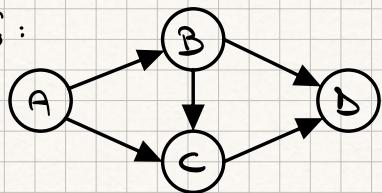
È una formula ricorsiva che parte dai nodi  $u$  e  $v$  per poi ricorsivamente analizzare tutti i percorsi  $u \rightarrow v$ , analizzando ogni volta i nodi che hanno avuto in ingresso al nodo corrente. Più sono contenuti i nodi, più il valore di  $s(u, v)$  diminuirà perché analizzeremo nodi singolari dopo aver passato molti nodi con bassa o nulla similarità  $s(x, y)$ .

La similarità di due vertici in un grafo è data da quanto vengono riferiti dagli stessi vertici.

In altre parole, più vicini condividono più i due vertici sono simili.

## Esempio SimRank

$G:$

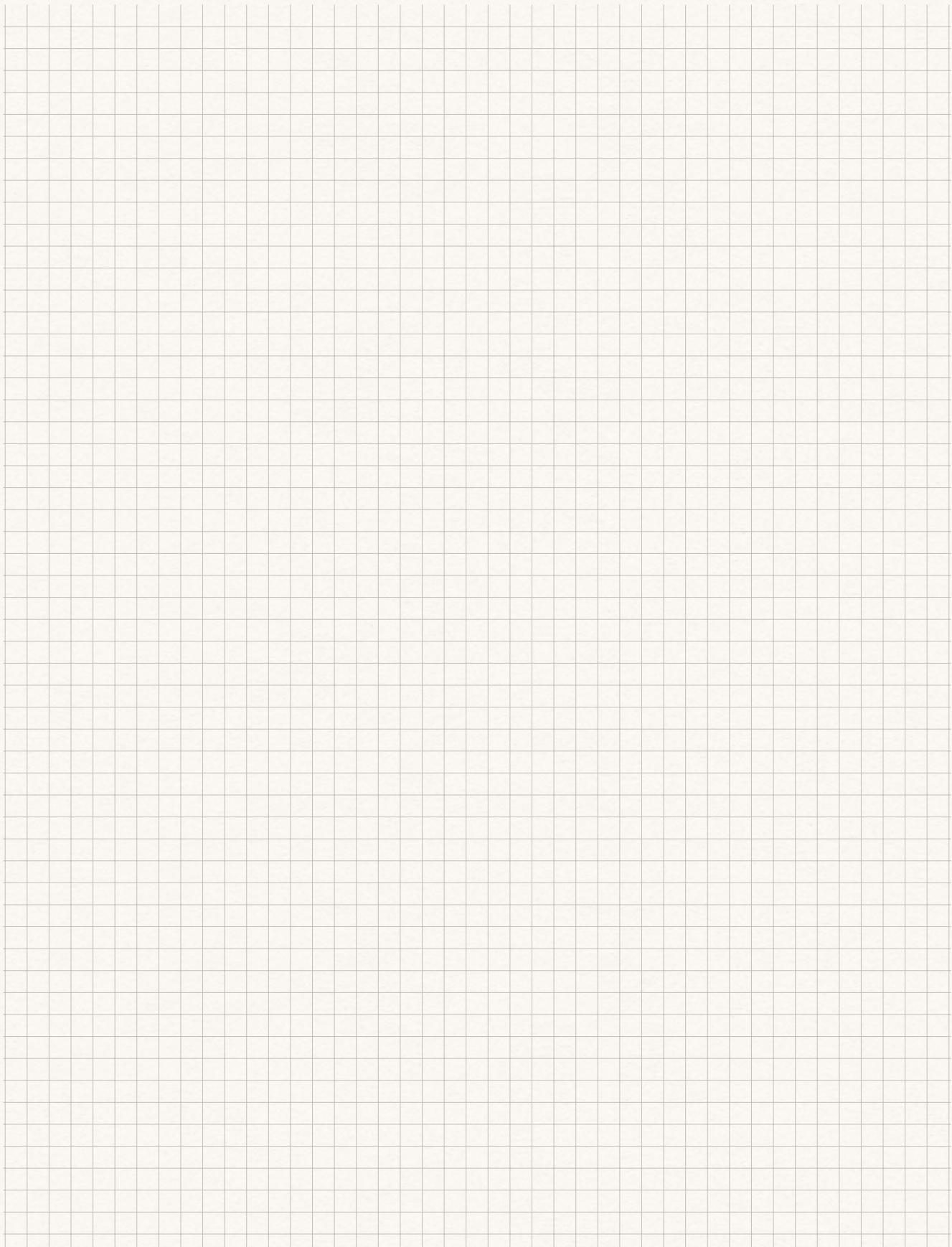


Calcoliamo  $s(A, D)$

$$I(A) = \{B, C\}$$

$$I(D) = \{B, C\}$$

$$S_1(A, D) = \frac{c}{4} \left[ \underbrace{s_p(B, B)}_{1} + \underbrace{s_p(B, C)}_{\varnothing} + \underbrace{s_p(C, B)}_{\varnothing} + \underbrace{s_p(C, C)}_{1} \right]$$



## 2) SIMILARITY BASED ON RANDOM WALK

Dato delle probabilità che se un'informazione parta dal vertice A, e la stessa informazione parta anche dal vertice B, queste informazioni raggiungono un terzo vertice C sia partendo da A che da B.

Definiamo la **distanza prevista** fra due vertici  $u \in V$ :

$$d(u, v) = \sum_{t: u \rightarrow v} P[t] \ell(t)$$

Expected
Distance

Dove la somma è calcolata fra tutti i tour possibili fra  $u \in V$  ( $t: u \rightarrow v$ ) tale per cui non toccano  $v$  tranne come ultima meta'.

$\ell(t)$  è la lunghezza del tour in questione

$P[t]$  è la probabilità di attraversare quel tour.

$$P[t] = \begin{cases} \prod_{i=1}^{k-1} \frac{1}{|\alpha(x_i)|} & \text{per } \ell(t) > 0 \\ \emptyset & \text{per } \ell(t) = 0 \end{cases}$$

Così  $\alpha(x_i)$  out-neighborhood.

$x_i$  è uno dei vertici che compongono il tour  $t$ :  
 $t = \langle x_1, \dots, x_k \rangle$ .

**Intuizione Expected Distance:**

L'intuizione della formula della distanza prevista è che ci dà la distanza che un viaggiatore partendo da  $u$ , prendendo ogni volta scelte casuarine, arriva a  $v$ .

## EXPECTED MEETING DISTANCES ( $\delta_{\text{MD}}$ )

Indica il numero di passi che due viaggiatori, uno che parte da  $u$  e l'altro da  $v$ , ci mettono per incontrarsi. **SIMMETRICA** per definizione.

### Definizione rigorosa

Bisogna usare il concetto di  $G^2$ .

Formalmente  $\delta_{\text{MD}}(u, v)$  è semplicemente la distanza prevista in  $G^2$  che parte da  $(u, v)$  per arrivare a qualsiasi modo  $(x, x) \in V^2$  detto singleton.

$$m(u, v) = \sum_{t: (u, v) \rightsquigarrow (x, x)} P[t] \ell(t)$$

$G^2$  è spesso NON Fortemente connesso, perciò se non ci sono tour per  $(u, v)$ :  $m(u, v) = \infty$ .

Questo può essere problematico nel definire alcuni percorsi che portano a singleton e altri a  $(u, v)$ .

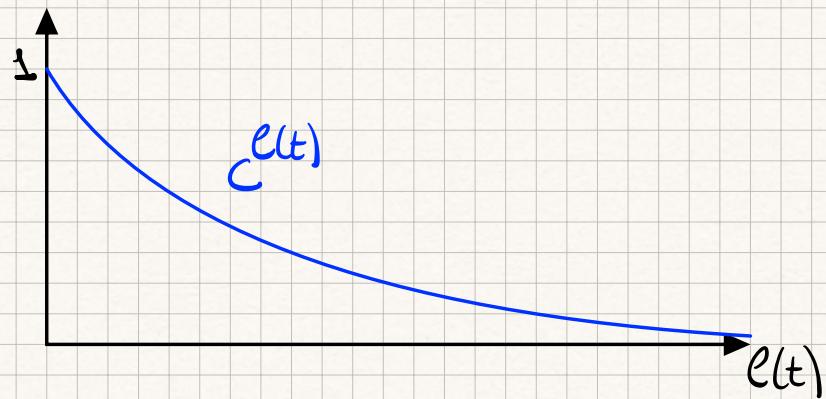
## EXPECTED- $f$ MEETING DISTANCE

Invece di calcolare la lunghezza di un tour  $\ell(t)$ , possiamo usare una funzione  $f(\ell(t))$  monotonica non negativa, che ha dominio  $[0, \infty)$ :

$$s_f(u, v) = \sum_{t: (u, v) \rightsquigarrow (x, x)} P[t] C^{\ell(t)} \quad \text{con } C \in [0, 1]$$

Dove  $C$  sta ad indicare la probabilità che a ogni passo del tour continui a camminare.

Questa metrica si sposa meglio col classico concetto di similitudine perché due vertici vicini hanno una  $C(t)$  più piccola. Essendo  $C(t)$  esponenziale se è  $\neq \emptyset$  allora  $C^0 = 1$ , viceversa  $C^\infty = \emptyset$ , questo perché  $C \in [\emptyset, 1]$ :

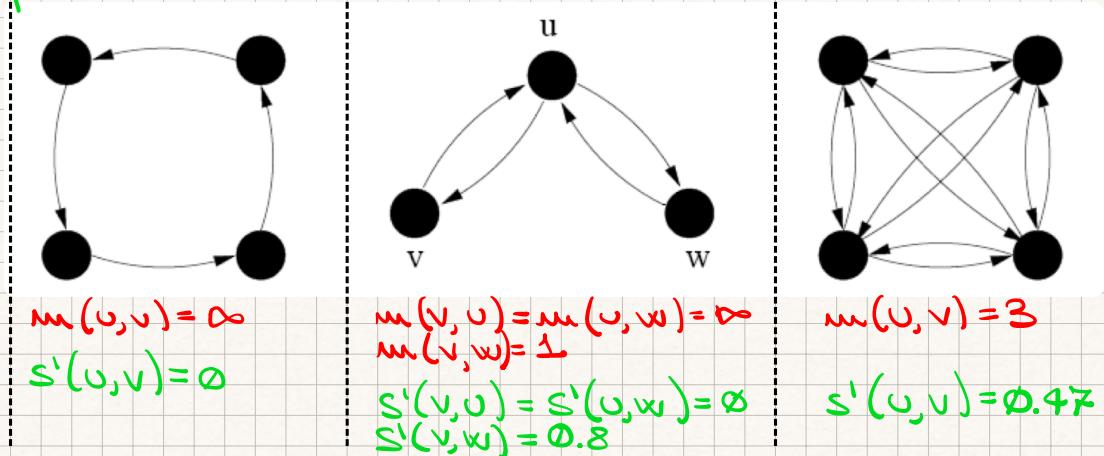


Di conseguenza si:

- 1)  $S'(\alpha, b) = \emptyset$  non ci sono tour fra  $\alpha$  e  $b$
- 2)  $S'(\alpha, b) = 1$  allora  $\alpha = b$
- 3)  $S'(\alpha, b) \in ]0, 1[$  per gli altri tipi di  $\alpha$  e  $b$ .

$\tilde{\epsilon}$  dimostrato che  $S(u, v) = S'(u, v)$  presso lo stesso parametro  $C$ .

Esempio con  $C = 0.8$ :



## SPARSEST CUT

Intuitivamente per cercare i cluster in un grafo dobbiamo tagliare il grafo stesso in parti in modo tale che i vertici contenuti in una di queste parti siano ben connessi fra loro, mentre i vertici di cluster diversi abbiano connessioni ben più blande.

### Definiamo un $CUT(S, T)$

Prendiamo il grafico  $G = (V, \delta)$  diretto.

Chiamiamo  $CUT(S, T)$  una partizione dell'insieme di vertici  $V$  in  $G$  dove  $V = S \cup T$  e  $S \cap T = \emptyset$

Il **cut set** di un  $CUT(S, T)$  è il set di vertici che vengono tagliati:  $\{(u, v) \in \delta \mid u \in S, v \in T\}$

Dimensione di un cut è il numero di archi nel cut set. Per archi pesati il valore del cut è la somma dei pesi.

### MINIMUM CUT

È il taglio che ha minore dimensione del cut set.

### SPARSITÀ

Scegliere un cut dove, per ogni vertice  $u$  del cut set la maggior parte degli archi connessi a  $u$ , portano a un solo cluster.

La misura di sparsità di un taglio  $C = (S, T)$  è:

$$\Phi = \frac{\text{Cut-size}}{\min\{|S|, |T|\}}$$

Il taglio con  $\Phi$  minore è il più sparso.

## Problemi con la sparsità.

- 1) Favorisce i tagli dove ci sono pochi archi da effettivamente togliere, che bilanciano i due set di vertici
- 2) È un problema di tipo NP-Hard

## MODULARITY OF A CLUSTER

Per K cluster, è espressione della loro qualità:

$$Q = \sum_{i=1}^K \left( \frac{e_i}{|E|} - \left( \frac{d_i}{2|E|} \right)^2 \right)$$

Dove  $e_i$  è il numero di archi fra i vertici del  $i$ -esimo cluster. Mentre  $d_i$  è la somma del numero di archi di ogni vertice del cluster (la somma dei gradi di ogni vertice del cluster)

Quindi  $\frac{e_i}{|E|}$  è la probabilità che un arco cade nel  $i$ -esimo cluster. Mentre  $\frac{d_i}{2|E|}$  è la probabilità che un vertice casuale appartenga al cluster  $i$ -esimo.

È la funzione obiettivo da massimizzare per avere cluster di qualità.

$|E|$  è il numero totale di archi nel grafo.

## SCAN

Definiamo l'immadiato vicino di un vertice come tutti quei vertici contenuti one-hop dal vertice in questione.  $\Gamma(u) = \{v \mid (u, v) \in \delta\} \cup \{u\}$

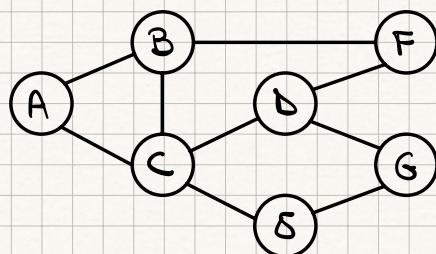
## STRUCTURES SIMILARITY

$$\sigma(v, w) = \frac{|\Gamma(v) \cap \Gamma(w)|}{\sqrt{|\Gamma(v)| \cdot |\Gamma(w)|}}$$

Questo valore dipende dal numero di vicini che i due vertici condividono, perché più ne condividono più l'insieme dato dall'intersezione fra  $\Gamma(v)$  e  $\Gamma(w)$  è grande, e di conseguenza c'è grande anche  $\sigma(v, w)$ .

## Esempio

1)  $\Gamma(A) = \{B, C\}$   
 $\Gamma(D) = \{C, F, G\}$



$$\sigma(A, D) = \frac{1}{\sqrt{5}}$$

2)  $\Gamma(A) = \{A, C\}$   
 $\Gamma(G) = \{D, E\}$

$$\Gamma(A, G) = \emptyset$$

Per definire i membri di un cluster SCAN usa un parametro di threshold  $\epsilon$ . È un parametro per la similarità, perché definisce l' $\epsilon$ -vicinato ( $v$ )  $N_\epsilon(v)$ :

$$N_\epsilon(v) = \{w \in T(v) \mid \sigma(v, w) \geq \epsilon\}$$

### CORE VERTICES:

Definiamo  $v$  come core-vertex se  $|N_\epsilon(v)| \geq \mu$ , dove  $\mu$  è un ulteriore threshold:

$$\text{COR}_\epsilon, \mu(v) = |N_\epsilon(v)| \geq \mu$$

Difatti SCAN usa un approccio density-based (simile a DBSCAN) dove però segue i core-vertices. Se un vertice  $v$  è nel  $\epsilon$ -vicinato di un core-vertex allora  $v$  è assegnato allo stesso cluster del core- $v$ . Il processo continua finché nessun cluster cresce.

### VERTEX DIRECT REACHED

Un vertice  $w$  è direct-reached da un core  $v$  se

$$\text{DirReach}_\epsilon, \mu(v, w) \iff \text{COR}_\epsilon, \mu(v) \wedge w \in N_\epsilon(v)$$

### STRUCTURE REACHABLE

Un vertice  $v$  è raggiunto da un core-vertex se esiste una catena di vertici  $w_1, \dots, w_n$  tale per cui  $m \leftarrow w_1, w_i \leftarrow w_{i+1}, w_n \leftarrow v$ .

## STRUCTURES CONNECTED

Quando due vertici sono connessi a un terzo  
terzo-vertex in comune ai due.

$$\text{CONNECT}_{\varepsilon, \mu}(v, w) \Leftrightarrow \exists u \in V: \text{REACH}_{\varepsilon, \mu}(v, u) \wedge \text{REACH}_{\varepsilon, \mu}(u, w)$$

## STRUCTURES CONNECTED CLUSTER

Un cluster così definito deve possedere le seguenti  
proprietà:

3) CONNECTIVITY:  $\forall v, w \in C: \text{CONNECT}_{\varepsilon, \mu}(v, w)$

2) MAXIMALITY:  $\forall v, w \in V: v \in C \wedge \text{REACH}_{\varepsilon, \mu}(v, w) \Leftrightarrow w \in C$