



**Università  
degli Studi  
di Palermo**



# Misure di Similarità

CORSO DI BIG DATA  
a.a. 2021/2022

Prof. Roberto Pirrone

# Sommario

- Distanza e similarità
- Misure di similarità per dati quantitativi
- Misure di similarità per dati categorici e testuali
- Similarità per serie temporali e sequenze discrete
- Similarità tra grafi

# Distanza e similarità

- La *distanza* tra due oggetti  $O_i$  e  $O_j$ ,  $dist(O_i, O_j)$  è una funzione che ritorna un valore tanto più piccolo quanto  $O_i$  e  $O_j$  sono simili
- La *similarità* tra due oggetti  $O_i$  e  $O_j$ ,  $sim(O_i, O_j)$  è una funzione che ritorna un valore tanto più grande quanto  $O_i$  e  $O_j$  sono simili
- Le funzioni di distanza o similarità sono in genere espresse
  - In forma chiusa
  - Tramite formulazione algoritmica

# Dati multidimensionali quantitativi

- La distanza più comune è la norma  $L_p$

$$\text{Dist}(\bar{X}, \bar{Y}) = \left( \sum_{i=1}^d |x_i - y_i|^p \right)^{1/p}$$

- $p=1 \rightarrow$  Manhattan distance
- $p=2 \rightarrow$  norma euclidea che è invariante alla rotazione dello spazio dei dati

$$\text{Dist}(\bar{X}, \bar{Y}) = \left( \sum_{i=1}^d a_i \cdot |x_i - y_i|^p \right)^{1/p}$$

# Dati multidimensionali quantitativi

- *Curse of dimensionality*: Le norme perdono di capacità discriminativa tra i dati al crescere della dimensionalità di questi ultimi
- Si consideri la variabile casuale generata calcolando la distanza di Manhattan di un punto  $X_i = (Y_1, \dots, Y_n)^T$  generato con distribuzione uniforme all'interno dell'ipercubo unitario  $d$ -dimensionale  $[0,1]^d$ :

$$\text{Dist}(\bar{O}, \bar{X}) = \sum_{i=1}^d (Y_i - 0)$$

# Dati multidimensionali quantitativi

- *Curse of dimensionality*: Le norme perdono di capacità discriminativa tra i dati al crescere della dimensionalità di questi ultimi
- Si consideri la variabile casuale generata calcolando la distanza di Manhattan di un punto  $X_i = (Y_1, \dots, Y_n)^T$  generato con distribuzione uniforme all'interno dell'ipercubo unitario  $d$ -dimensionale  $[0,1]^d$ :

$$\text{Dist}(\bar{O}, \bar{X}) = \sum_{i=1}^d (Y_i - 0)$$

$$\mu = d/2, \quad \sigma = \sqrt{d/12}$$

# Dati multidimensionali quantitativi

- Curse of dimensionality

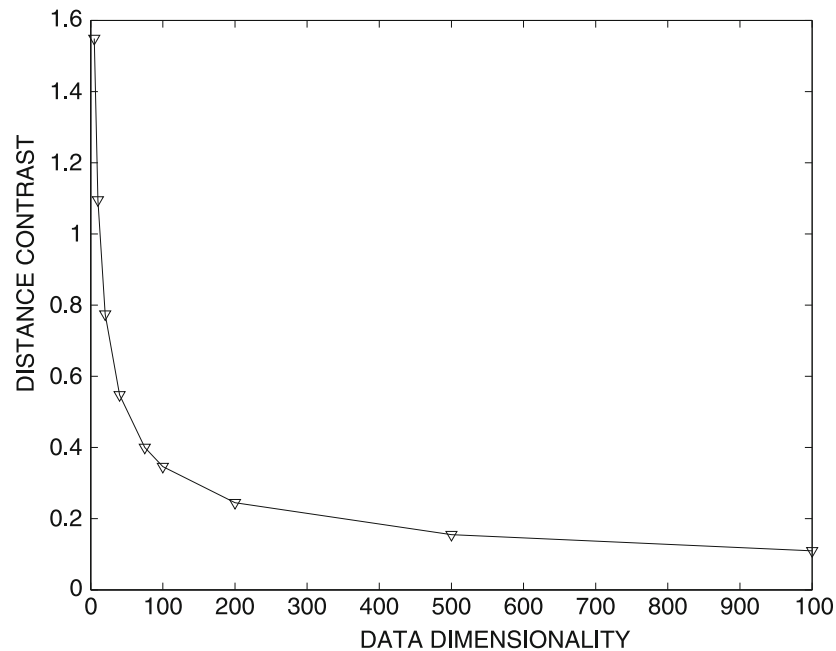
- Per la legge dei grandi numeri, la stragrande maggioranza dei valori di distanza starà nell'intervallo  $[D_{min}, D_{max}] = [\mu \pm 3\sigma] = 6\sigma = \sqrt{3d}$
- Il *contrasto* della distanza di Manhattan, cioè la dimensione relativa dei valori misurati di distanza rispetto al valor medio delle grandezze in gioco è dunque:

$$\text{Contrast } (d) = \frac{D_{\max} - D_{\min}}{\mu} = \sqrt{12/d}$$

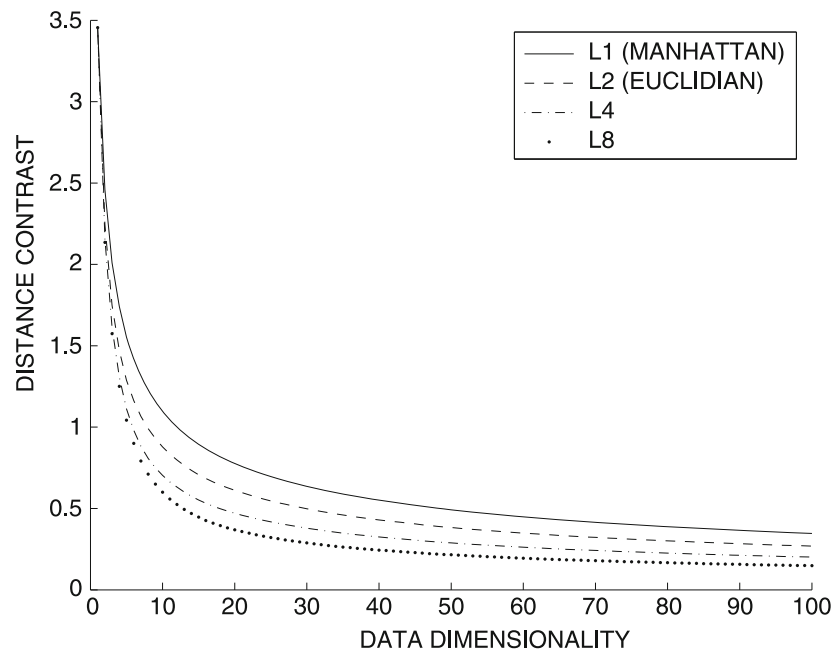
- All'aumentare di  $p$ , questo valore decresce ancora più velocemente con  $d$

# Dati multidimensionali quantitativi

- Curse of dimensionality



(a) Contrasts with dimensionality



(b) Contrasts with norms



# Dati multidimensionali quantitativi

- Curse of dimensionality
  - È importante eliminare le feature irrilevanti
  - È importante ridurre il rumore sui dati perché piccole variazioni in alta dimensionalità possono mascherare l'effetto della similarità

# Dati multidimensionali quantitativi

- Proximity thresholding
  - Si suddividono le dimensioni dei dati in  $k_d$  intervalli a profondità costante in modo da rendere costante la probabilità che due vettori condividano lo stesso intervallo in una data dimensione
  - $\mathcal{S}(X, Y, k_d)$  sia l'insieme delle dimensioni in cui le componenti di  $X$  e  $Y$  cadono nello stesso intervallo e siano  $n_i$  e  $m_i$  rispettivamente il minimo ed il massimo valore lungo la dimensione  $i \in \mathcal{S}(X, Y, k_d)$

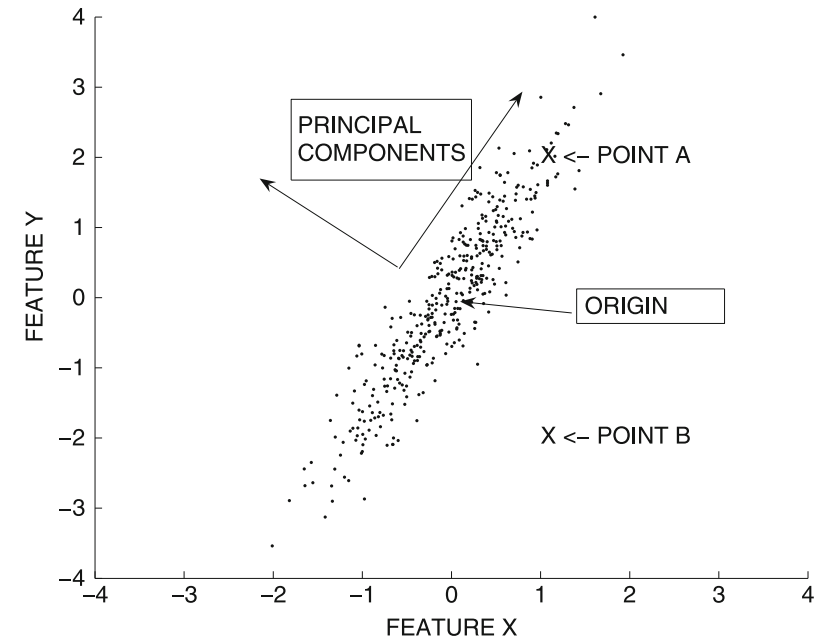
$$PSelect(\bar{X}, \bar{Y}, k_d) = \left[ \sum_{i \in \mathcal{S}(\bar{X}, \bar{Y}, k_d)} \left( 1 - \frac{|x_i - y_i|}{m_i - n_i} \right)^p \right]^{1/p}$$

# Dati multidimensionali quantitativi

- Distanza di Mahalanobis

$$Maha(\bar{X}, \bar{Y}) = \sqrt{(\bar{X} - \bar{Y})\Sigma^{-1}(\bar{X} - \bar{Y})^T}$$

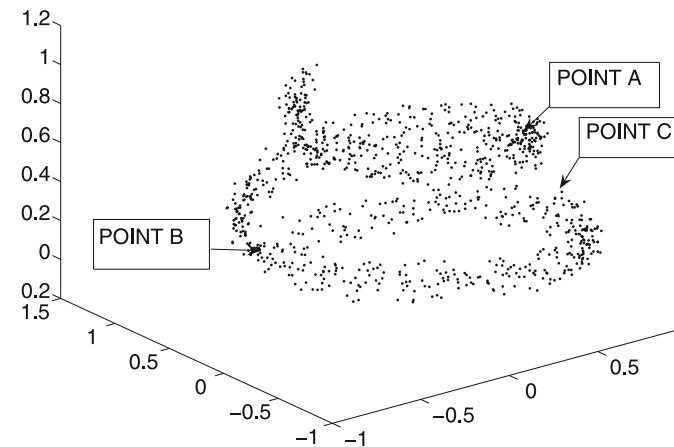
- Tiene esplicitamente conto della covarianza tra i dati che possono avere una particolare distribuzione nello spazio
- Diviene banalmente la distanza euclidea dopo aver ruotato i dati lungo le loro componenti principali ( $\Sigma$  diviene diagonale)



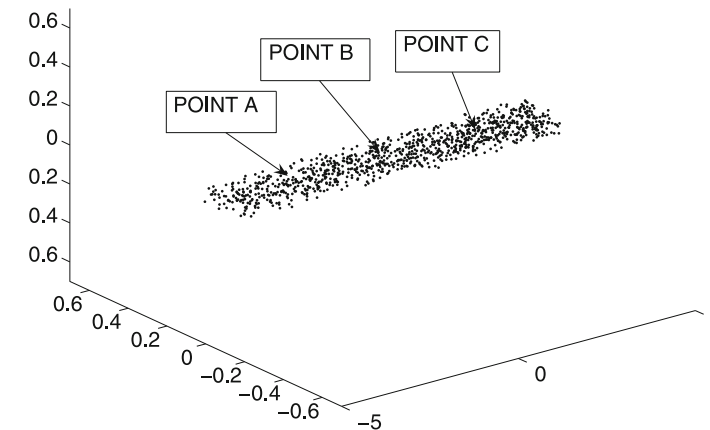
# Dati multidimensionali quantitativi

- ISOMAP embedding

- Si calcolano i  $k$  vicini di ogni punto (i  $k$  punti *ad esso più vicini* di ogni altro ovvero « $k$  nearest neighbors»)
- Si crea un grafo pesato  $G$  in cui ogni punto è un nodo connesso con i  $k$  vicini con archi il cui peso è la distanza
- Per ogni coppia di punti  $X, Y$  la distanza si ottiene come costo del cammino minimo lungo il grafo tra il nodo  $X$  ed il nodo  $Y$



(a) A and C seem close  
(original data)



(b) A and C are actually far away  
(ISOMAP embedding)

# Dati binari, categorici e testuali

- La similarità si calcola tra i singoli attributi, ognuno dei quali varia all'interno di un proprio insieme di valori discreti

$$Sim(\bar{X}, \bar{Y}) = \sum_{i=0}^d S(x_i, y_i)$$

- La scelta di  $S(x_i, y_i)$  determina i vari tipi di similarità

# Dati binari, categorici e testuali

- Inverse occurrence frequency
  - $p_k(x)$  è la frazione di record per cui il  $k$ -esimo attributo vale  $x$

$$S(x_i, y_i) = \begin{cases} 1/p_k(x_i)^2 & x_i = y_i \\ 0 & \text{otherwise} \end{cases}$$

- Goodall similarity

$$S(x_i, y_i) = \begin{cases} 1 - p_k(x_i)^2 & x_i = y_i \\ 0 & \text{otherwise} \end{cases}$$

# Dati binari, categorici e testuali

- Distanza coseno
  - Le norme  $L_p$  non gestiscono bene il problema di documenti aventi lunghezza differente e sono sensibili a lunghi documenti per cui tendono a riportare valori sempre più alti
  - Il coseno dell'angolo tra due vettori è invariante rispetto alla loro lunghezza
  - Sia  $h(.)$  la funzione TF-IDF di ogni termine nello spazio LSA

$$\cos(\bar{X}, \bar{Y}) = \frac{\sum_{i=1}^d h(x_i) \cdot h(y_i)}{\sqrt{\sum_{i=1}^d h(x_i)^2} \cdot \sqrt{\sum_{i=1}^d h(y_i)^2}}$$

# Dati binari, categorici e testuali

- Coefficiente di Jaccard

- Usato per i dati binari che rappresentano insiemi:  $X$  è un vettore di bit che indica l'appartenenza o meno di un «lessico» di elementi dati ad un insieme  $S_X$

$$J(\bar{X}, \bar{Y}) = \frac{\sum_{i=1}^d x_i \cdot y_i}{\sum_{i=1}^d x_i^2 + \sum_{i=1}^d y_i^2 - \sum_{i=1}^d x_i \cdot y_i} = \frac{|S_X \cap S_Y|}{|S_X \cup S_Y|}$$

*Anche nota come Intersection over Union(IoU)*

- Può essere usato come similarità tra vettori multidimensionali



# Similarità tra serie temporali

- Fattori da tenere in considerazione:
  - Normalizzazione degli attributi comportamentali in fase di pre-processing
  - Traslazione dell'attributo temporale per riferire due serie allo stesso intervallo di tempo
  - Scalatura dell'attributo temporale per serie che descrivono lo stesso fenomeno, ma a scale diverse (*time warping*)
  - Presenza di segmenti rumorosi che inducono il match tra due serie lungo intervalli non contigui

# Similarità tra serie temporali

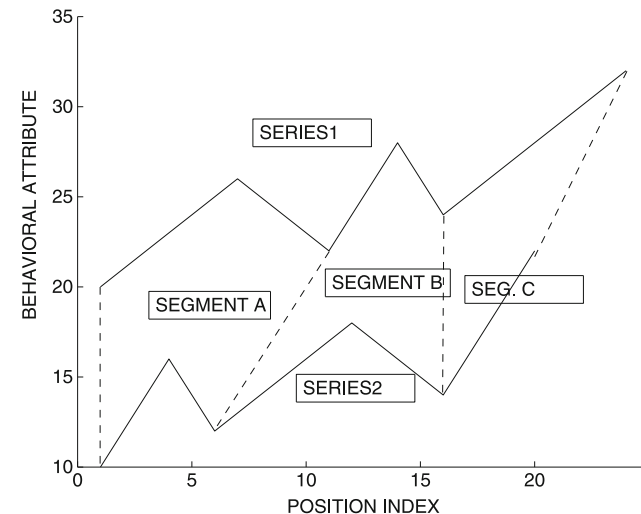
- Uso delle norme  $L_p$ :
  - Una norma funziona perfettamente su due sequenze trasformate con HWT
  - HWT normalizzata induce una rotazione degli assi rispetto ad un sistema di riferimento ortonormale che rappresenta diverse scale temporali
  - Necessitano di sequenze della stessa lunghezza

# Similarità tra serie temporali

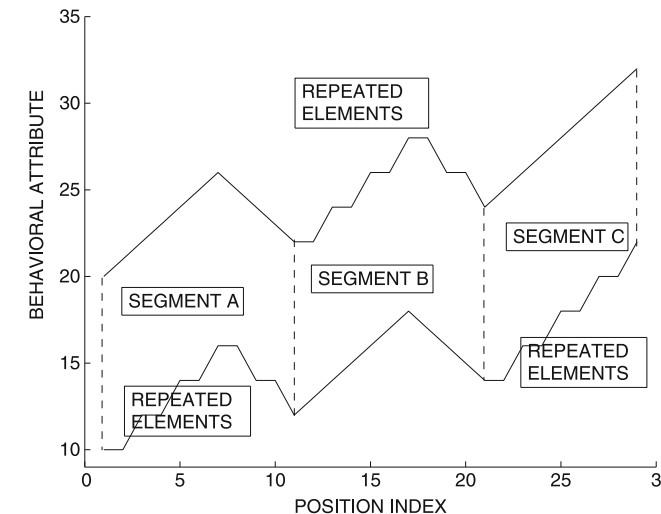
- Dynamic Time Warping (DTW):

- Necessità di adattare due serie temporali alla stessa scala, ma con coefficiente variabile da segmento a segmento
  - Speech recognition: gestione delle diverse velocità del parlato

- Consente di applicare una norma alle serie scalate che vengono riportate alla stessa lunghezza



(a) Original series



(b) Warped series

# Similarità tra serie temporali

- Dynamic Time Warping (DTW):
  - Necessità di adattare due serie temporali alla stessa scala, ma con coefficiente variabile da segmento a segmento
    - Speech recognition: gestione delle diverse velocità del parlato
  - È definita in maniera ricorsiva ed è indipendente dalla lunghezza delle due serie
  - Parte dalla considerazione che una distanza tra due serie di  $n$  elementi può essere riscritta come  $\text{dist}(X_n, Y_n) = \text{dist}(X_{n-1}, Y_{n-1}) + g(|x_n - y_n|)$

# Similarità tra serie temporali

- Dynamic Time Warping (DTW):

- $X = (x_1, \dots, x_m)$   $Y = (y_1, \dots, y_n)$

$$DTW(i, j) = \text{distance}(x_i, y_j) + \min \begin{cases} DTW(i, j-1) & \text{ripeti } x_i \\ DTW(i-1, j) & \text{ripeti } y_j \\ DTW(i-1, j-1) & \text{non ripetere} \end{cases}$$

$$DTW(0, 0) = 0, \quad DTW(j, 0) = DTW(0, j) = \infty \forall i, j$$

- Si calcola iterativamente con un doppio ciclo **for** su  $i$  e  $j$
- Può essere calcolata solo su una finestra per cui  $|i-j| \leq w$  altrimenti  $\rightarrow \infty$
- Si può estendere a serie temporali con attributi multidimensionali

# Similarità tra serie temporali

- Window based matching:
  - Date due serie  $X$  e  $Y$ , si estraggono una serie di finestre non sovrapponibili escludendo gli eventuali intervalli rumorosi  $(X_1, \dots, X_r)$  e  $(Y_1, \dots, Y_r)$
  - Uno schema generale per il calcolo della similarità è:

$$Sim(\bar{X}, \bar{Y}) = \sum_{i=1}^r Match(\bar{X}_i, \bar{Y}_i)$$

- $Match(.,.)$  può essere implementato in diversi modi

# Similarità tra serie discrete

- Seguono lo stesso principio generale delle serie temporali
- Si può pensare di applicare le norme e la DTW è possibile
- Ci sono due approcci caratteristici
  - Edit distance (distanza di Levenshtein)
  - Longest common subsequence (LCSS)

# Similarità tra serie discrete

- Edit distance (distanza di Levenshtein)
  - Date due sequenze  $X = (x_1, \dots, x_m)$  e  $Y = (y_1, \dots, y_n)$ , la distanza  $Edit(i,j)$  tra la sottosequenza  $X_i$  e la sottosequenza  $Y_j$  è lo *sforzo minimo* necessario a trasformare una sequenza in un'altra in termini di *operazioni di edit*
    - Cancellazione dell'ultimo elemento di  $X_i$  per fare il match con l'ultimo di  $Y_j$
    - Inserimento di un elemento in coda a  $X_i$  per fare il match con l'ultimo di  $Y_j$
    - Sostituzione dell'ultimo elemento di  $X_i$  con l'ultimo di  $Y_j$  se sono diversi
  - È una distanza, ma è asimmetrica  $\rightarrow Edit(i,j) \neq Edit(j,i)$



# Similarità tra serie discrete

- Edit distance (distanza di Levenshtein)

$$Edit(i, j) = \min \begin{cases} Edit(i-1, j) + \text{Costo Cancellazione} \\ Edit(i, j-1) + \text{Costo Inserzione} \\ Edit(i-1, j-1) + I_{ij} \cdot (\text{Costo Sostituzione}) \end{cases}$$

$$I_{ij} = \begin{cases} 0 & x_i = y_j \\ 1 & \text{altrimenti} \end{cases} ; \quad \begin{aligned} Edit(i, 0) &= i \text{ cancellazioni} \\ Edit(0, j) &= j \text{ inserzioni} \end{aligned}$$

- La scelta dell'implementazione delle primitive di cancellazione, inserimento e sostituzione in termini numerici può consentire anche l'applicazione alle serie temporali

# Similarità tra serie discrete

- Longest common subsequence (LCSS)
  - È una similarità → al crescere di LCSS la similarità tra le due sequenze aumenta

$$LCSS(i, j) = \max \begin{cases} LCSS(i-1, j-1) + 1 & \text{se } x_i = y_j \\ LCSS(i-1, j) & \text{altrimenti se cancelliamo } x_i \\ LCSS(i, j-1) & \text{altrimenti se cancelliamo } y_j \end{cases}$$

$$LCSS(i, 0) = LCSS(0, j) = 0 \quad \forall i, j$$

# Similarità nei grafi

- Similarità tra nodi in un grafo
  - Si usano distanze nei casi in cui si possono associare dei costi ai nodi o agli archi
  - Si usano similarità nei casi in cui il grafo è descritto in termini di pesi
  - Omofilia: due nodi sono tanto più simili quanto più sono connessi tra loro attraverso cammini che siano *quanti più possibile* o *quanto più brevi possibile*

# Similarità nei grafi

- Cammino minimo (algoritmo di Dijkstra)
  - Si esplorano tutti i nodi a partire dal nodo sorgente  $s$ , a partire dal quale misuriamo la distanza, verso un certo nodo  $j$  di destinazione
  - Ad ogni arco  $(i,j)$  è associato un costo  $c_{ij}$
  - Di volta in volta, scelto il nodo  $i$ -esimo con il costo minimo del cammino  $SP(s,i)$  tra i nodi già visitati, i suoi vicini  $j$  ricevono un'etichetta con il costo del cammino minimo trovato sinora partendo dal nodo sorgente  $s$ :

$$SP(s, j) = \min \{ SP(s, j), SP(s, i) + c_{ij} \}$$

Inizializzazione:  $SP(s, s) = 0, \quad SP(s, j) = \infty \quad \forall j \neq s$

# Similarità nei grafi

- Cammino minimo (algoritmo di Dijkstra)
  - L'approccio è lineare nel numero di archi del grafo
  - Ogni nodo viene visitato esattamente una volta e si ottiene il calcolo delle distanze di  $s$  da tutti gli altri nodi del grafo in una sola passata

$$SP(s, j) = \min \{ SP(s, j), SP(s, i) + c_{ij} \}$$

Inizializzazione:  $SP(s, s) = 0, \quad SP(s, j) = \infty \quad \forall j \neq s$

# Similarità nei grafi

- Random Walk
  - Si utilizza in grafi in cui due nodi possono essere connessi da molti cammini contemporaneamente
  - Un nodo  $A$  potrebbe essere più simile a  $B$  cui è connesso da tre cammini piuttosto che a  $C$  cui è connesso da un cammino solo anche se più breve

# Similarità nei grafi

- Random Walk
  - Da  $s$  si dipartono dei cammini casuali verso gli altri nodi
  - Ogni passo da un nodo all'altro è gestito da una probabilità legata al peso  $w_{ij}$  dell'arco tra i due
  - Ogni nodo  $j$  ha una *probabilità di restart* cioè di far ritornare indietro il cammino verso  $s$
  - I cammini sono variabili casuali con una distribuzione polarizzata verso  $s$
  - La similarità si ottiene *massimizzando* questa probabilità → cammini più probabili connettono  $s$  ai suoi nodi più simili

# Similarità nei grafi

- Similarità tra grafi
  - Si ricerca un *isomorfismo* tra strutture a grafo
  - Il problema è complicato dal fatto che più nodi possono avere la stessa etichetta (ad es. molecole)
  - Ci sono diversi approcci



# Similarità nei grafi

- Similarità tra grafi
  - Massimo sotto-grafo comune
  - Similarità basata sulla presenza di sotto-strutture
    - Si contano le sottostrutture più frequenti analogamente alle stringhe
    - fingerprint molecolari
  - Graph-edit distance
    - Analoga alla string-edit: le operazioni sono quelle di inserimento di nodi, inseriemnto e cancellazione di archi e sostituzione della label di un nodo