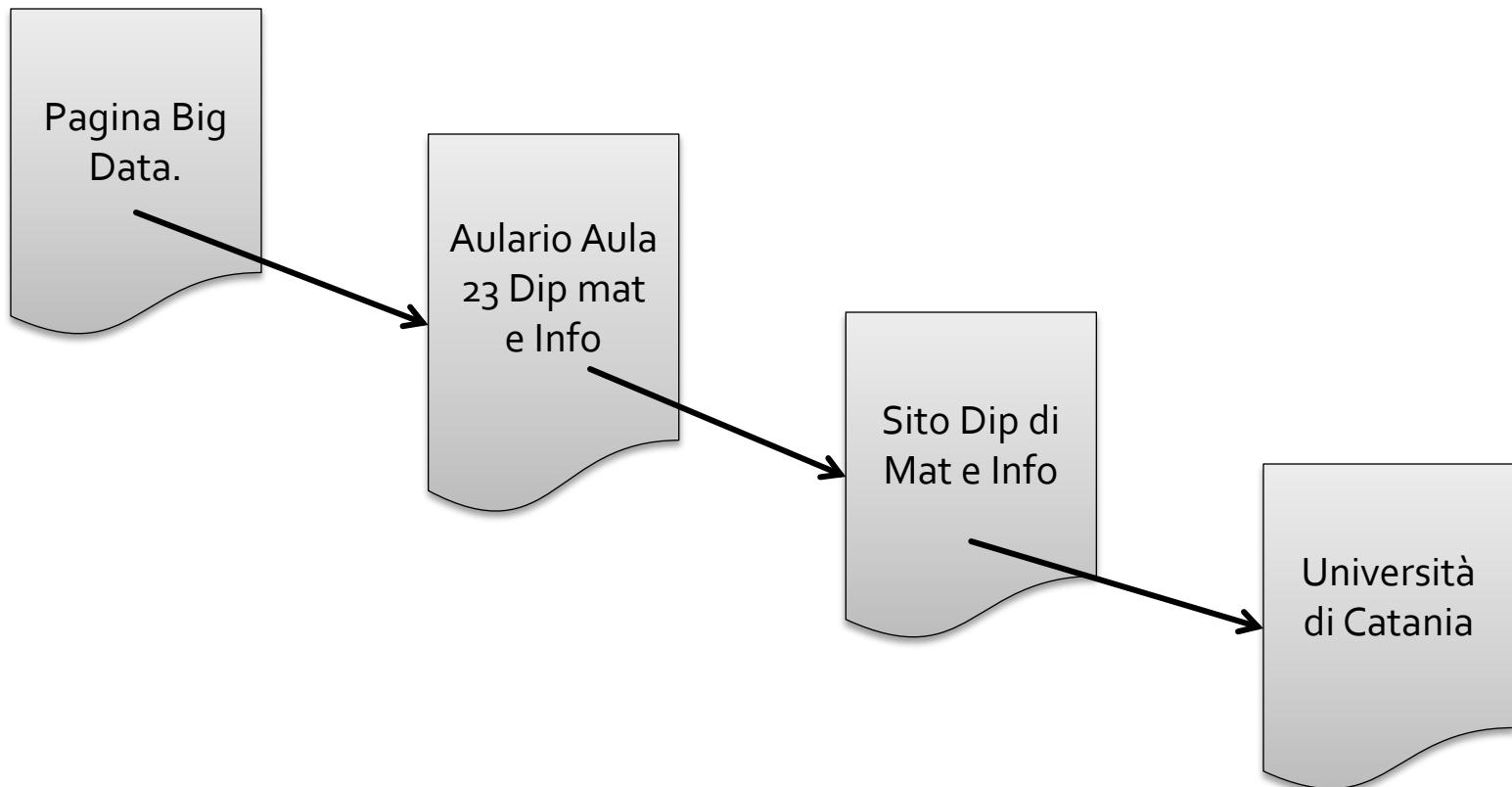


Analisi di grandi grafi: Link Analysis, PageRank

Il web come un grafo

■ Il web come un grafo diretto:

- Nodi: pagine web
- archi: link ipertestuali



Questione

- **Com'è organizzato il Web?**
- **Primo tentativo:** Human curated **Web directory**
 - Yahoo, DMOZ, LookSmart
- **Secondo tentativo:** **Web Search**
 - **Information Retrieval:**
Trovare documenti rilevanti in un insieme piccolo e autorevole
 - Articoli di quotidiani, brevetti, ecc.
 - **Ma:** Il web è immenso, pieno di documenti non strutturati, spam, roba senza senso ecc.



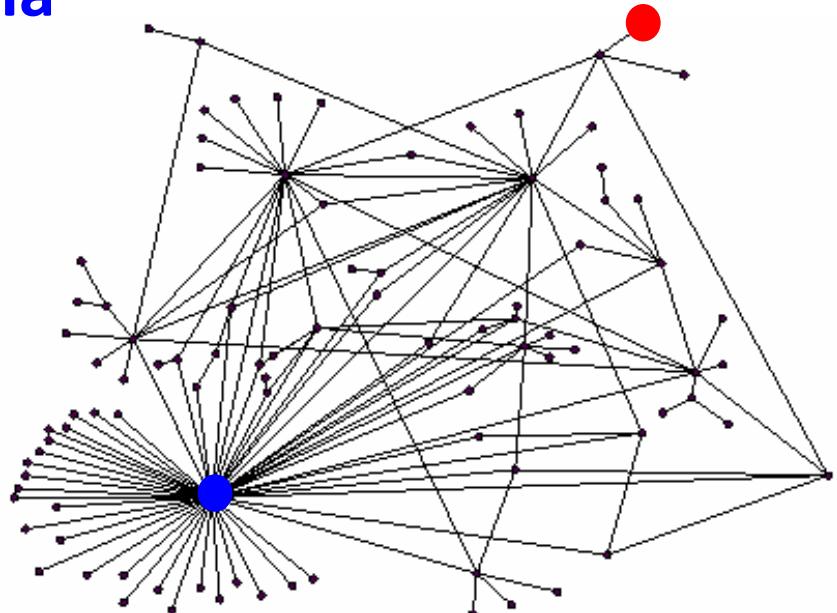
Web Search: 2 sfide

2 sfide per il web search:

- **(1) il Web contiene tante sorgenti di informazione**
A quale “credere”?
 - **Trick:** Le pagine affidabili possono puntare l’una all’altra
- **(2) Qual è la “migliore” risposta ad una query “giornali”?**
 - Non c’è una sola risposta
 - **Trick:** Le pagine che conoscono i giornali potrebbero puntare tutte a diversi quotidiani.

Ranking dei nodi del grafo

- **Le pagine web non sono ugualmente “importanti”**
www.pippo.net vs. www.unict.it
- C’è una grande variabilità
nella connettività del grafo del web.
**rank delle pagine in base alla
struttura dei link!**



Algoritmi per l'analisi dei link

- Copriremo i seguenti **approcci** per calcolare l'**importanza** dei nodi in un grafo:
 - Page Rank
 - Topic-Specific (Personalized) Page Rank
 - Random walk with restart
 - Web Spam Detection
 - Hub and Authority

PageRank: The “Flow” Formulation

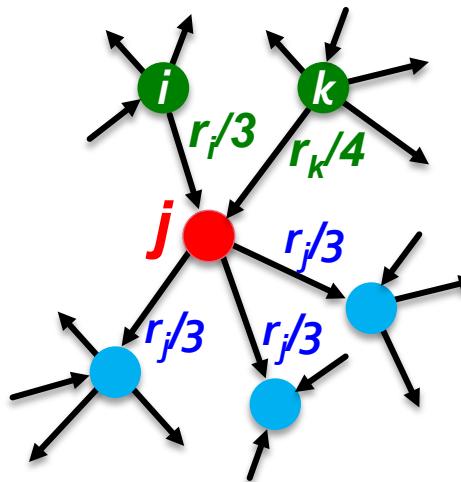
Link come voti

- **Idea: link come voti**
 - Una pagina è importante se possiede tanti link
 - Entranti? Uscenti?
- **Pensiamo i link come i voti:**
 - www.unict.it quanti link entranti ha?
 - www.pippo.net quanti link entranti ha?
- **Ma tutti i link sono uguali?**
 - I link da pagine web importanti contano di più
 - Abbiamo la ricorsione!

Formulazione ricorsiva di base

- Ogni voto di un link è proporzionale
all'importanza della sua pagina sorgente
- Se la pagina j con importanza r_j ha n link uscenti, ogni link prende r_j/n voti
- L'importanza di j è la somma dei voti dei suoi link entranti

$$r_j = r_i/3 + r_k/4$$

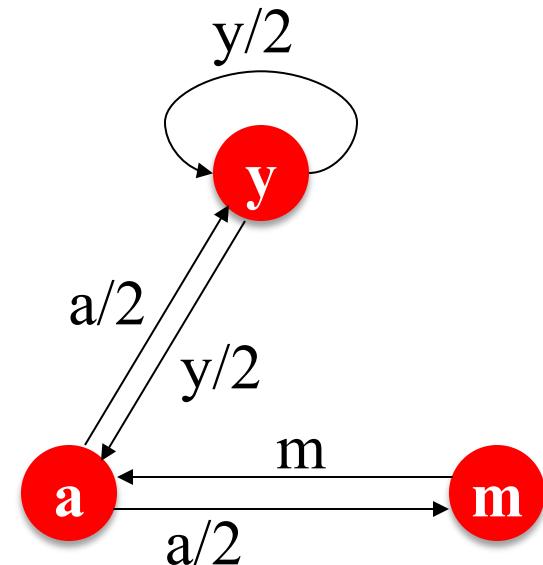


PageRank: Il modello “Flusso”

- Un "voto" da una pagina importante vale più
- Una pagina è importante se viene puntata da altre pagine importanti
- Definiamo il “rank” r_j per la pagina j

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

d_i ... out-degree del nodo i



$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

Risolviamo le equazioni di Flusso

- **3 equazioni, 3 incognite, no vincoli**
 - Non c'è una sola soluzione
 - Tutte le soluzioni sono equivalenti a meno di un fattore di scala
- **Un vincolo addizionale rende la soluzione unica:**
 - $r_y + r_a + r_m = 1$
 - **Soluzione:** $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$
- **Con un solver di equazioni lineari (es. Gaussian elimination) risolviamo sistemi piccoli, ma..**
- **abbiamo bisogno di una nuova formulazione per large web-size graph!**

Equazioni:

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

PageRank: Formulazione matriciale

- **Matrice di adiacenza Stocastica M**
 - Sia d_i il numero di out-link di i
 - Se $i \rightarrow j$, allora $M_{ji} = \frac{1}{d_i}$ altrimenti $M_{ji} = 0$
 - M è una matrice stocasticaⁱ colonna
 - La somma delle colonne è pari a 1
- **Vettore Rank r :** vettore con una entry per ogni pagina
 - r_i è lo score d'importanza della pagina i
 - $\sum_i r_i = 1$
- **L'equazione di flusso può essere riscritta come**

$$\mathbf{r} = \mathbf{M} \cdot \mathbf{r}$$

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

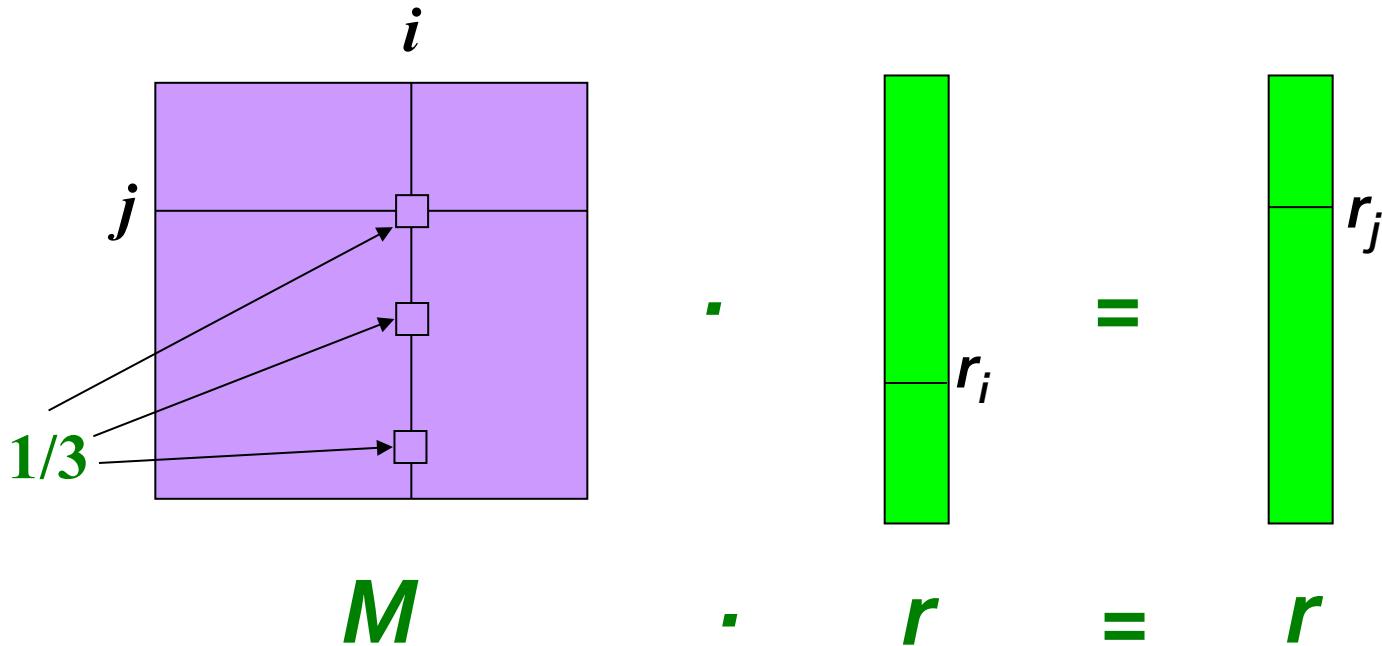
Esempio

- Equazione flusso:
- Formulazione matriciale

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

$$M \cdot r = r$$

- Supponiamo che la pagina i linka 3 pagine, incluso j



Formulazione autovettori

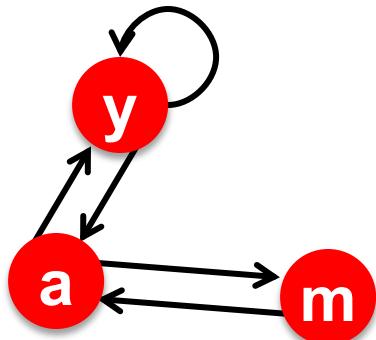
- L'equazione la possiamo riscrivere come

$$\mathbf{r} = \mathbf{M} \cdot \mathbf{r}$$

- Quindi il **vettore dei rank** \mathbf{r} è un **autovettore** della matrice stocastica del web \mathbf{M}
 - Infatti, il primo autovettore, con corrispondente autovalore **1**
 - Il più grande autovalore per \mathbf{M} è **1** poiché \mathbf{M} è stocastica sulle colonne (con entry non-negative)

NOTA: \mathbf{x} è un Autovettore con il corrispondente λ se:
 $\mathbf{Ax} = \lambda\mathbf{x}$

Esempio: Equazioni di flusso & M



	y	a	m
y	1/2	1/2	0
a	1/2	0	1
m	0	1/2	0

$$r = M \cdot r$$

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}$$

Metodo Power Iteration

- Dato il grafo del web con n nodi, dove i nodi sono le pagine e gli archi sono i link ipertestuali
- Power iteration: semplice schema iterativo
 - Supponiamo dia vere N pagine web
 - Inizializziamo: $\mathbf{r}^{(0)} = [1/N, \dots, 1/N]^T$
 - Iterare: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
 d_i out-degree del node i
 - Stop quando $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$

PageRank: soluzione

■ Power Iteration:

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$

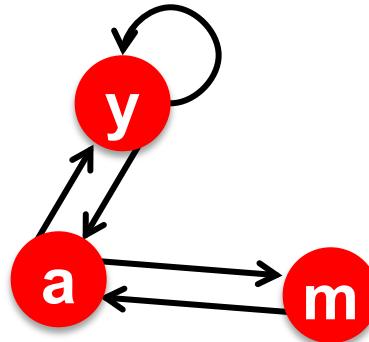
■ 2: $r = r'$

■ Goto 1

■ Esempio:

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix}$$

Iterazione 0, 1, 2, ...



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	1
m	0	$\frac{1}{2}$	0

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

PageRank: soluzione

■ Power Iteration:

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$

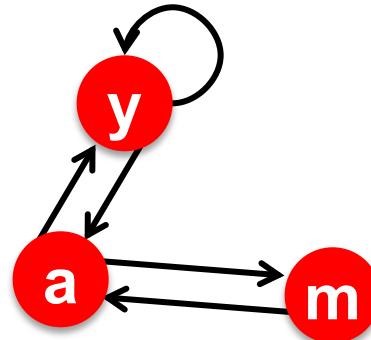
■ 2: $r = r'$

■ Goto 1

■ Esempio:

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \dots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & 3/15 \end{matrix}$$

Iterazione 0, 1, 2, ...



	y	a	m
y	1/2	1/2	0
a	1/2	0	1
m	0	1/2	0

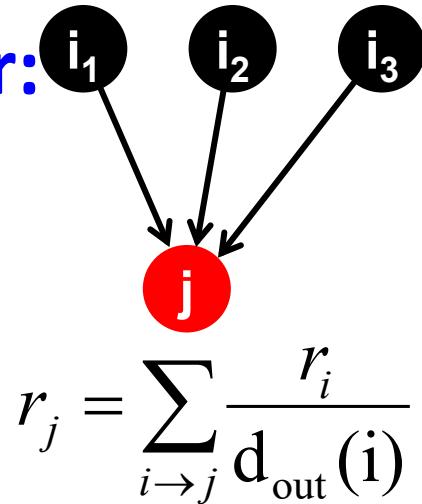
$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

Random Walk surfer

- Immaginiamo un random web surfer:
 - A tempo t , il surfer è sulla pagina i
 - A tempo $t + 1$, il surfer segue un out-link da i uniformemente random
 - Arriva ad una pagina j linkata da i
 - Ripete il processo all'infinito
- Sia:
 - $p(t)$... il vettore la cui i -esima coordinata è la probabilità che il surfer sia alla pagina i a tempo t
 - $p(t)$ è la distribuzione di probabilità sulle pagine web

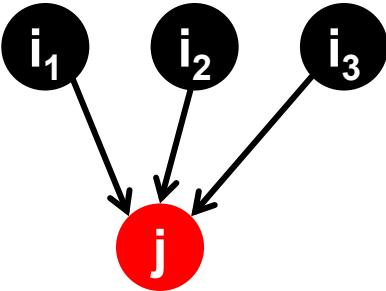


Distribuzione stazionaria

- Dove sarà il random surfer a tempo $t+1$?

- Segue un link in modo random

$$p(t + 1) = M \cdot p(t)$$



$$p(t + 1) = M \cdot p(t)$$

- Supponiamo che raggiunge uno stato

$$p(t + 1) = M \cdot p(t) = p(t)$$

quindi $p(t)$ è la **distribuzione stazionaria** del random walk

- Il nostro vettore r sodisfa $r = M \cdot r$

- Quindi, r è la distribuzione stazionaria della random walk

Esistenza e unicità

- **Risultato centrale nella teoria delle random walk (note come catene di Markov):**

Per grafi che soddisfano **certe condizioni**, la **distribuzione stazionaria è unica** e sarà raggiunta a prescindere da quali saranno le probabilità iniziali a tempo $t = 0$

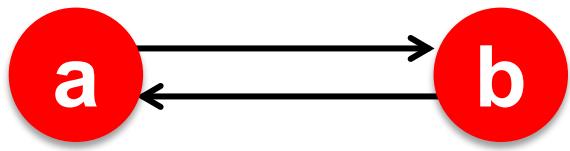
PageRank: La formulazione Google

PageRank: tre domande

$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i} \quad \text{equivalentemente} \quad r = Mr$$

- Converge?
- Converge a quello che desideriamo?
- I risultati sono ragionevoli?

Converge?



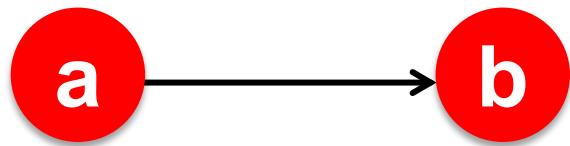
$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i}$$

■ Esempio:

$$\begin{array}{lcl} r_a & = & 1 & 0 & 1 & 0 \\ r_b & & 0 & 1 & 0 & 1 \end{array}$$

Iterazione 0, 1, 2, ...

Converge a ciò che vogliamo?



$$r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i}$$

■ Esempio:

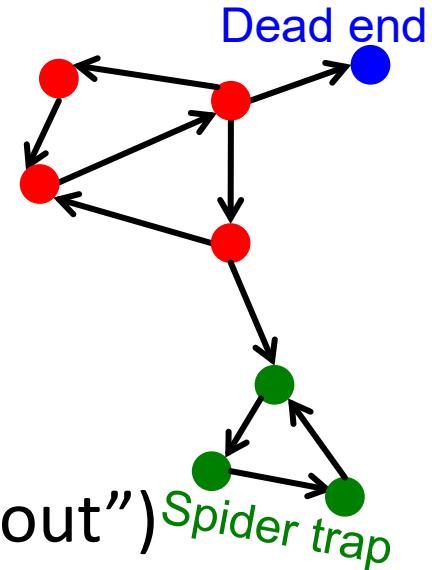
$$\begin{array}{ll} r_a & = \\ r_b & = \end{array} \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

Iterazione 0, 1, 2, ...

PageRank: Problemi

2 problemi:

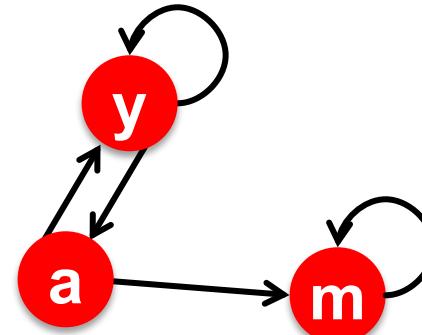
- (1) Alcune pagine sono **dead end** (no out-link)
 - Random walk non ha dove andare
 - Causano la perdita dell'importanza ("leak out")
- (2) **Spider trap:**
(tutti gli out-link sono nello stesso gruppo)
 - Random walker resta catturato in una trappola
 - La spider-trap assorbirà tutta l'importanza



Problem: Spider Trap

■ Power Iteration:

- Set $r_j = 1$
- $r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$
- And iterate



m is a spider trap

	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	0
m	0	$\frac{1}{2}$	1

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2$$

$$r_m = r_a/2 + r_m$$

■ Esempio:

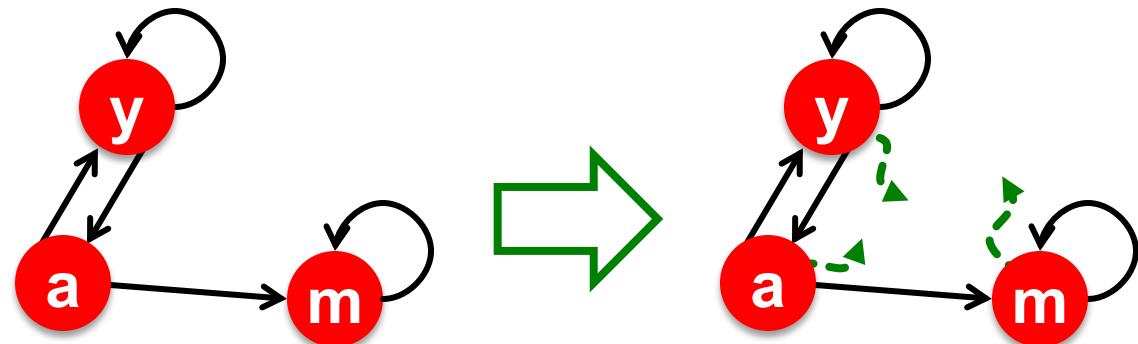
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \dots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{matrix}$$

Iteration 0, 1, 2, ...

Tutto lo score viene "assorbito" dal nodo m.

Soluzione: Teleport!

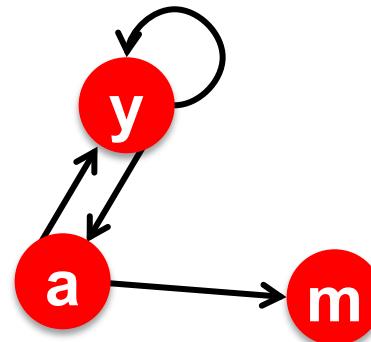
- La soluzione di Google per le spider trap: **Ad ogni time step, il random surfer ha due opzioni**
 - Con probabilità β , segue un link in nodo random
 - Con probabilità $1-\beta$, salta ad una pagina random
 - Valori comuni per β sono nel range 0.8 - 0.9
- Il surfer sarà teletrasportato fuori dalla spider-trap in un numero ragionevole di time step



Problema: Dead End

■ Power Iteration:

- Set $r_j = 1$
- $r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$
- Ed itera



	y	a	m
y	1/2	1/2	0
a	1/2	0	0
m	0	1/2	0

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2$$

$$r_m = r_a/2$$

■ Esempio:

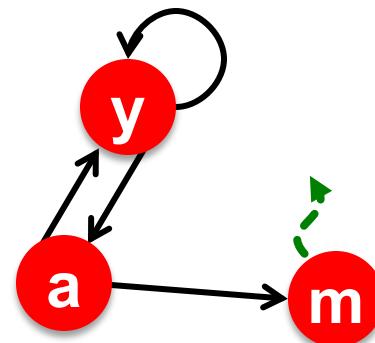
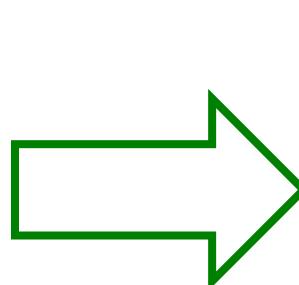
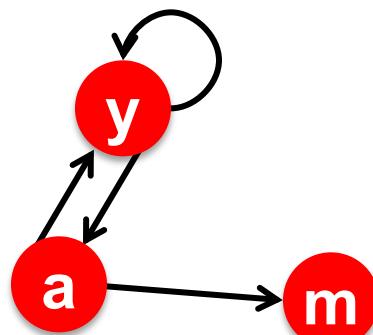
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \dots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & 0 \end{matrix}$$

Iterazione 0, 1, 2, ...

Il PageRank viene “perso” perché la matrice non è stocastica

Soluzione: Teleport!

- **Teleport:** segui un link random con probabilità 1 dal dead-end
- Modifica la matrice in modo corrispondente



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	0
m	0	$\frac{1}{2}$	0

	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$
a	$\frac{1}{2}$	0	$\frac{1}{3}$
m	0	$\frac{1}{2}$	$\frac{1}{3}$

Perché il Teleport risolve i problemi?

Perché i dead-end e le spider trap sono un problema
e perché il teleport li risolve?

- **Spider-trap** non sono un problema ma con le trap gli score del PageRank non sono quello che volgiamo
 - **Soluzione:** Non rimanere mai bloccati in una spider trap.
Con il teleport ne usciamo in un numero finito di passi
- **Dead-end** sono un problema
 - Matrice non stocastica a colonne le assunzioni iniziali non sono rispettate
 - **Soluzione:** Rendere la matrice stocastica sempre. Il teleport consente di uscire sempre dal dead end.

Soluzione: Random Teleport

■ Soluzione di Google che risolve tutto:

Ad ogni step, il random surfer ha due opzioni

- Con probabilità β , segue il link random
- Con probabilità $1-\beta$, salta ad una pagina web random

■ Equazione PageRank [Brin-Page, 98]

$$r_j = \sum_{i \rightarrow j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N}$$

d_i ... out-degree
del nodo i

In questa formulazione M non ha dead-end. Possiamo sia preprocessare la matrice M e rimuovere tutti i dead-end oppure seguire il random teleport con probabilità 1 per uscire dal dead end

The Google Matrix

- **Equazione PageRank** [Brin-Page, '98]

$$r_j = \sum_{i \rightarrow j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N}$$

- **Matrice A di Google:**

$[1/N]_{N \times N}$...matrice $N \times N$
dove tutte le entry sono $1/N$

$$A = \beta M + (1 - \beta) \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}_{N \times N}$$

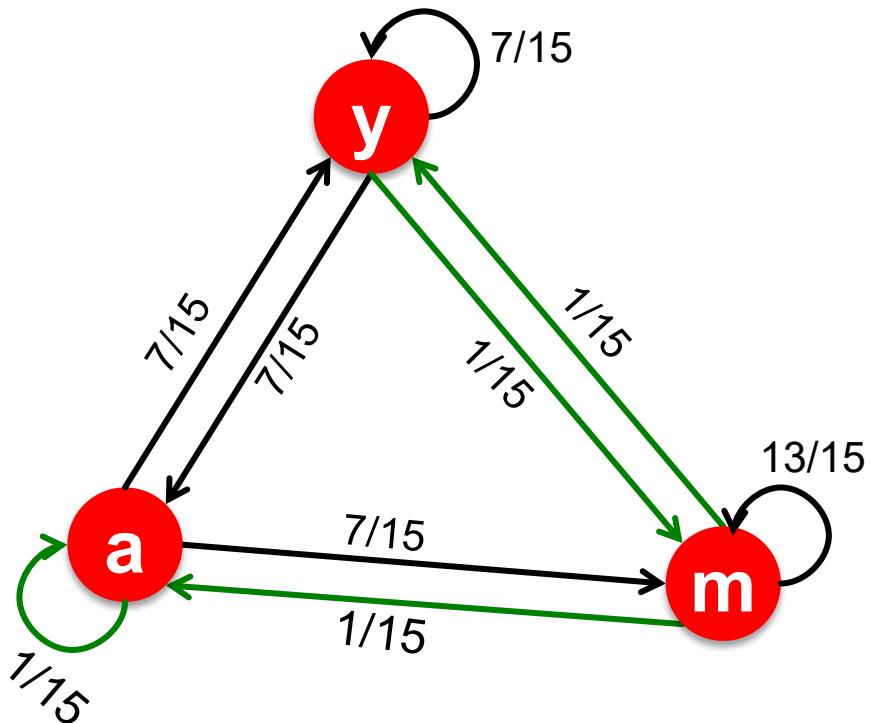
- **Problema ricorsivo:** $r = A \cdot r$

Power iteration funziona!

- **Cosa è β ?**

- In pratica $\beta = 0.8, 0.9$ (5 step in media per il jump)

Random Teleport ($\beta = 0.8$)



$$\begin{array}{c}
 M \\
 \boxed{\begin{matrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{matrix}} \\
 + 0.2 \\
 \boxed{\begin{matrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{matrix}} \\
 [1/N]_{NxN} \\
 A \\
 \boxed{\begin{matrix} y & 7/15 & 7/15 & 1/15 \\ a & 7/15 & 1/15 & 1/15 \\ m & 1/15 & 7/15 & 13/15 \end{matrix}}
 \end{array}$$

y	1/3	0.33	0.24	0.26		7/33
a	=	1/3	0.20	0.20	0.18	...
m		1/3	0.46	0.52	0.56	21/33

Come calcoliamo realmente il
PageRank?

Calcolare il Page Rank

■ Passo chiave moltiplicazione matrice-vettore

- $r^{\text{new}} = A \cdot r^{\text{old}}$

- Semplice se abbiamo memoria a sufficienza per mantenere A , r^{old} , r^{new}

- **Diciamo $N = 1$ miliardo di pagine** $A = \beta \cdot M + (1-\beta) [1/N]_{N \times N}$

- 4 byte per ogni entry
- 2 miliardi di entries per i vettori, circa 8GB

- **A ha N^2 entry**
 - 10^{18} numero davvero grande!

$$A = 0.8 \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} + 0.2 \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{7}{15} & \frac{7}{15} & \frac{1}{15} \\ \frac{7}{15} & \frac{1}{15} & \frac{1}{15} \\ \frac{1}{15} & \frac{7}{15} & \frac{13}{15} \end{bmatrix}$$

Formulazione Matriciale

- Supponiamo di avere N pagine
- Consideriamo la pagina i , con d_i out-link
- Abbiamo $M_{ji} = 1/|d_i|$ quando $i \rightarrow j$ e $M_{ji} = 0$ altrimenti
- Il **random teleport** è equivalente a:
 - Aggiungere un **teleport link** da i a qualsiasi altra pagina e settare la probabilità a $(1-\beta)/N$
 - La probabilità di raggiungere un out-link viene ridotta da $1/|d_i|$ a $\beta/|d_i|$
 - **Equivalentemente:** Tassiamo ogni pagina una frazione $(1-\beta)$ del suo score e lo ridistribuiamo uniformemente

Rielaboriamo le equazioni

- $r = A \cdot r$, dove $A_{ji} = \beta M_{ji} + \frac{1-\beta}{N}$
- $r_j = \sum_{i=1}^N A_{ji} \cdot r_i$
- $$\begin{aligned} r_j &= \sum_{i=1}^N \left[\beta M_{ji} + \frac{1-\beta}{N} \right] \cdot r_i \\ &= \sum_{i=1}^N \beta M_{ji} \cdot r_i + \frac{1-\beta}{N} \sum_{i=1}^N r_i \\ &= \sum_{i=1}^N \beta M_{ji} \cdot r_i + \frac{1-\beta}{N} \quad \text{poiché } \sum r_i = 1 \end{aligned}$$
- **Otteniamo:** $r = \beta M \cdot r + \left[\frac{1-\beta}{N} \right]_N$

Note: Assumiamo che \mathbf{M} non ha dead-end

$[x]_N$... vettore di lunghezza N con tutte le entry x

Formulazione per matrice sparsa

- **Equazione PageRank rielaborata**

$$\mathbf{r} = \beta \mathbf{M} \cdot \mathbf{r} + \left[\frac{1 - \beta}{N} \right]_N$$

- Dove $\left[(1-\beta)/N \right]_N$ vettore con tutte le N entries $(1-\beta)/N$
- \mathbf{M} è una **matrice sparsa!** (senza dead-end)
 - 10 link per nodo, approssimativamente $10N$ entry
 - Per ogni iterazione
 - Calcolare $\mathbf{r}^{\text{new}} = \beta \mathbf{M} \cdot \mathbf{r}^{\text{old}}$
 - Aggiungere valore costante $(1-\beta)/N$ ad ogni entry di \mathbf{r}^{new}
 - Se \mathbf{M} contiene dead-end $\sum_j r_j^{\text{new}} < 1$ e
dobbiamo quindi rinormalizzare \mathbf{r}^{new} affinché la somma sia 1

PageRank: Algoritmo completo

- **Input: Grafo G e parametro β**
 - Grafo diretto G (con spider trap e dead end)
 - Parametro β
- **Output: vettore PageRank r^{new}**

- **Set:** $r_j^{old} = \frac{1}{N}$
- **repeat until convergence:** $\sum_j |r_j^{new} - r_j^{old}| > \varepsilon$
 - $\forall j: r_j'^{new} = \sum_{i \rightarrow j} \beta \frac{r_i^{old}}{d_i}$
 $r_j'^{new} = \mathbf{0}$ if in-degree of j is 0
 - **Now re-insert the leaked PageRank:**
 $\forall j: r_j^{new} = r_j'^{new} + \frac{1-S}{N}$ **where:** $S = \sum_j r_j'^{new}$
 - $r^{old} = r^{new}$

Codifica matrice sparsa

- Codifichiamo la matrice sparsa usando solo le entry diverse da 0
 - Spazio proporzionale al numero di link
 - $10N$, or $4*10^9$ miliardi = 40GB
 - Non entra in memoria ma su disco

source node	degree	destination nodes
0	3	1, 5, 7
1	5	17, 64, 113, 117, 245
2	2	13, 23

Algoritmo di base: step di update

- Supponiamo di avere RAM a sufficienza per tenere r^{new} in memoria
 - Memorizziamo r^{old} e \mathbf{M} su disco
- 1 step di power-iteration è:

Initialize all entries of $r^{old} = (1-\beta) / N$

For each page i (of out-degree d_i):

Read into memory: $i, d_i, \text{dest}_1, \dots, \text{dest}_{d_i}, r^{old}(i)$

For $j = 1 \dots d_i$

$r^{new}(\text{dest}_j) += \beta r^{old}(i) / d_i$

0	r^{new}
1	
2	
3	
4	
5	
6	

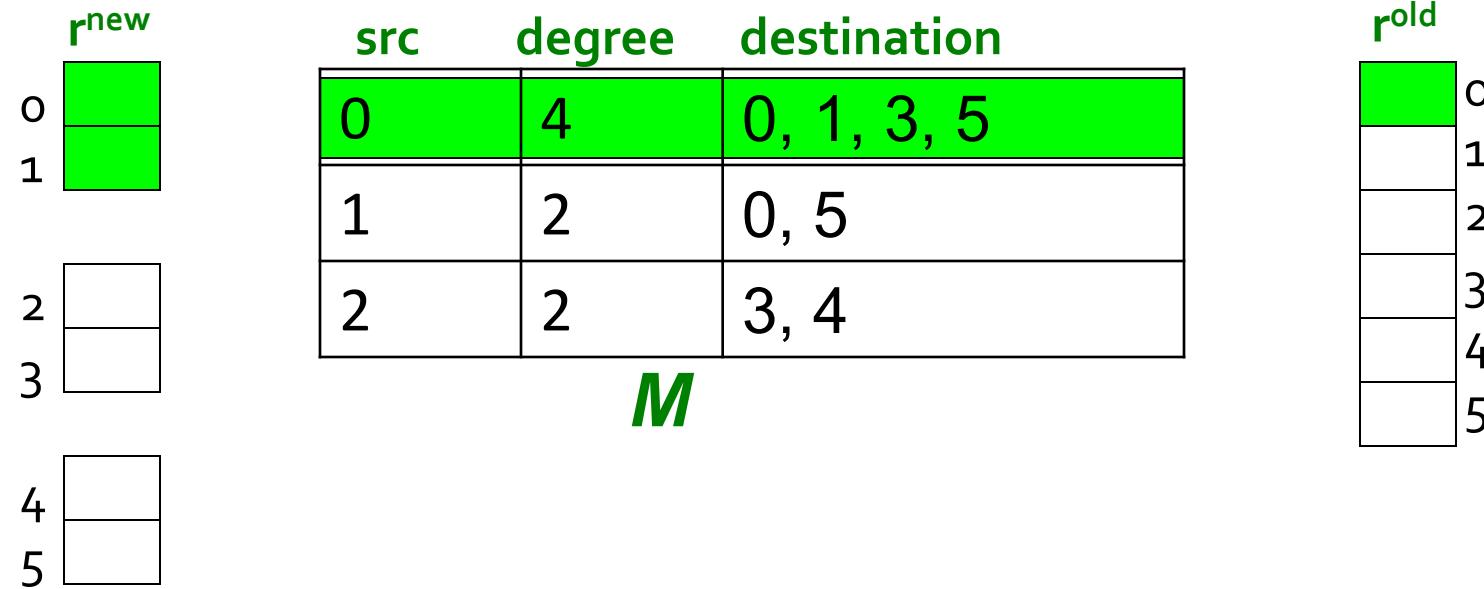
	source	degree	destination
0	0	3	1, 5, 6
1	1	4	17, 64, 113, 117
2	2	2	13, 23

0	r^{old}
1	
2	
3	
4	
5	
6	

Analisi

- Supponiamo di avere RAM a sufficienza per tenere r^{new} in memoria
 - Memorizziamo r^{old} e M su disco
- In ogni iterazione dobbiamo:
 - Leggere r^{old} e M
 - Scrivere r^{new} su disco
 - Costo per iterazione del Power method:
 $= 2|r| + |M|$
- Domanda:
 - Che facciamo se r^{new} non entra in memoria?

Algoritmo di aggiornamento Block-based



- Spezzare r^{new} in k blocchi che entrano in memoria
- Scansionare M e r^{old} una volta per ogni blocco

Analisi del Block Update

- **Simile a nested-loop join nei database**
 - Spezzare r^{new} in k che entrano in memoria
 - Scansionare M e r^{old} una volta per ogni blocco
- **Costo totale:**
 - k scansioni di M ed r^{old}
 - **Costo per iterazione del Power method:**
$$k(|M| + |r|) + |r| = k|M| + (k+1)|r|$$
- **Possiamo fare di meglio?**
 - **Suggerimento:** M è molto più grande di r (circa 10-20x), dobbiamo evitare di leggere k volte M per iterazione

Block-Stripe Update Algorithm

	src	degree	destination
r^{new} 0 1	0	4	0, 1
	1	3	0
2 3	2	2	1
	0	4	3
2 3	2	2	3
4 5	0	4	5
	1	3	5
4 5	2	2	4

Spezziamo M in strisce! Ogni striscia contiene solo i nodi destinazione del corrispondente blocco di r^{new}

Analisi Block-Stripe

- Spezzare M in strisce
 - Ogni striscia contiene solo i nodi destinazione del corrispondente blocco di r^{new}
- Overhead per striscia
 - trascurabile
- Cost per iterazione del Power method:
 $= |M|(1+\varepsilon) + (k+1)|r|$

Problemi del Page Rank

- **Misura la popolarità generica di una pagina**
 - Sbilanciato contro le topic-specific authority
 - **Soluzione:** Topic-Specific PageRank
- **Usa una misura singola di importanza**
 - Altri modelli usano più misure
 - **Soluzione:** Hubs-and-Authorities
- **Suscettibile al Link spam**
 - Link artificiali creati per fare il boosting del pagerank di alcuni siti
 - **Soluzione:** TrustRank

Topic-Specific PageRank

- Invece di una popolarità generica, possiamo misurare la popolarità all'interno di un argomento?
- **Goal:** valutare le pagine Web non solo in base alla loro popolarità, ma a quanto siano vicine ad un particolare argomento, ad esempio "Sport" o "storia"
- **Consente di rispondere alle query di ricerca in base agli interessi dell'utente**
 - **Esempio:** la query "Trojan" vuole pagine diverse a seconda che siate interessati allo sport, alla storia e alla sicurezza del computer

Topic-Specific PageRank

- Random walker ha una probabilità piccola di essere teletrasportato in un particolare istante
- **Teleport può portare a:**
 - **Standard PageRank:** Qualsiasi pagina con la stessa probabilità
 - Risolve spider-trap e dead-end
 - **Topic Specific PageRank:** A un topic-specific set di pagine “rilevanti” (**teleport set**)
- **Idea: Aggiungere un Bias alla random walk**
 - Quando il walker si teletrasporta, sceglie una pagina da un insieme S
 - S contiene solo pagine che sono rilevanti per il particolare topic
 - Es, Open Directory (DMOZ) pagine per un dato topic/query
 - Per ogni teleport set S , otteniamo un vettore r_s specifico

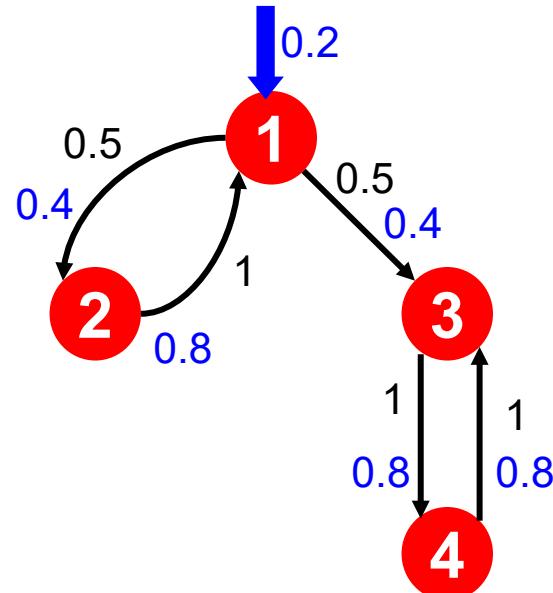
Formulazione matriciale

- Per far sì che queste funzioni dobbiamo aggiornare il teletrasporto nella formulazione del PageRank:

$$A_{ij} = \begin{cases} \beta M_{ij} + (1 - \beta)/|S| & \text{if } i \in S \\ \beta M_{ij} + 0 & \text{otherwise} \end{cases}$$

- A è ancora stocastica
- Pesiamo allo stesso modo tutte le pagine nel teleport set S
 - Potremmo anche assegnare pesi differenti alle pagine!
- Calcoliamo il tutto come per il PageRank standard:
 - Moltiplichiamo per M , aggiungiamo il vettore del teleport
 - Tenendo sempre in considerazione che la matrice M è sparsa

Esempio: Topic-Specific PageRank



Supponiamo $S = \{1\}$, $\beta = 0.8$

Node	Iteration				
	0	1	2	...	stable
1	0.25	0.4	0.28		0.294
2	0.25	0.1	0.16		0.118
3	0.25	0.3	0.32		0.327
4	0.25	0.2	0.24		0.261

$S=\{1\}$, $\beta=0.90$:

$r=[0.17, 0.07, 0.40, 0.36]$

$S=\{1\}$, $\beta=0.8$:

$r=[0.29, 0.11, 0.32, 0.26]$

$S=\{1\}$, $\beta=0.70$:

$r=[0.39, 0.14, 0.27, 0.19]$

$S=\{1,2,3,4\}$, $\beta=0.8$:

$r=[0.13, 0.10, 0.39, 0.36]$

$S=\{1,2,3\}$, $\beta=0.8$:

$r=[0.17, 0.13, 0.38, 0.30]$

$S=\{1,2\}$, $\beta=0.8$:

$r=[0.26, 0.20, 0.29, 0.23]$

$S=\{1\}$, $\beta=0.8$:

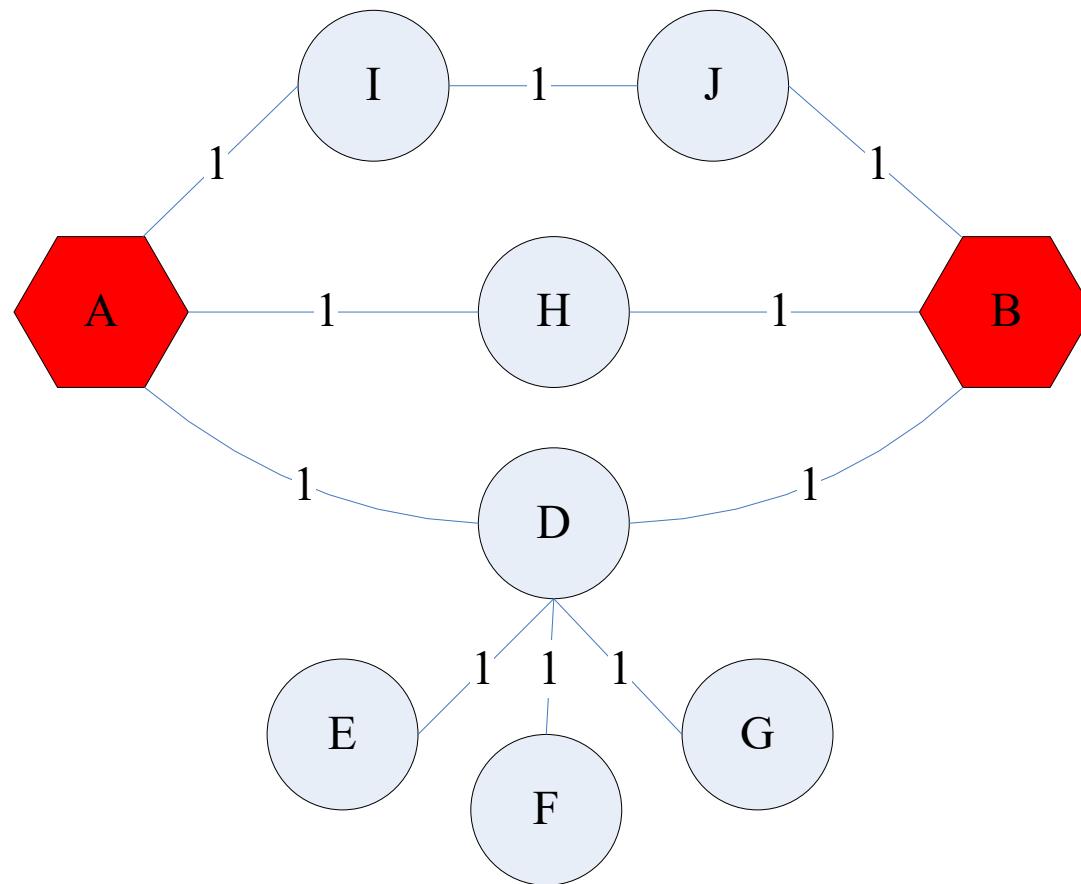
$r=[0.29, 0.11, 0.32, 0.26]$

Come scoprire il Vettore dei Topic S

- **Creiamo differenti PageRank per differenti topic**
 - Es. 16 categorie DMOZ top-level :
 - arts, business, sport,...
- **Quale topic usare?**
 - L'utente lo può scegliere da un menu
 - Classificare la query in un topic
 - Usare il **contesto** della query
 - Es. La query è eseguita a partire da una pagina web che parla di un topic ben preciso
 - History delle query es., “Calcio” Seguito da “Totti”
 - Contesto dell'utente. Per esempio i suoi bookmark,...

Applicazioni per misurare la Prossimità nei Grafi

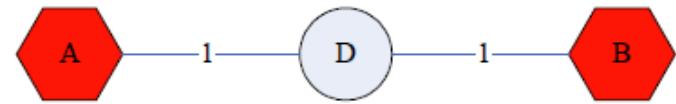
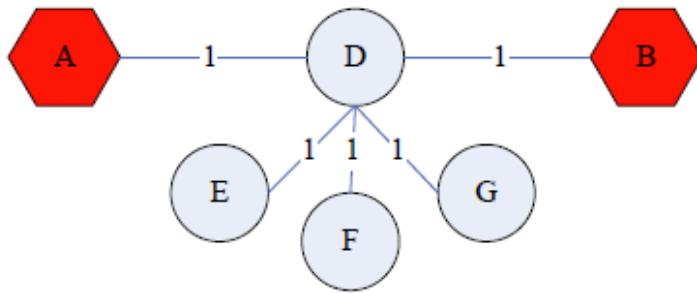
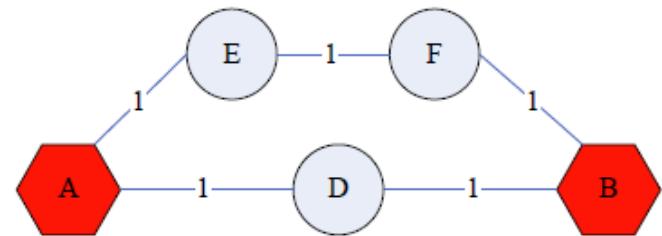
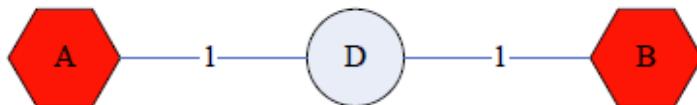
Prossimità nei grafi



a.k.a.: Relevance, Closeness, 'Similarity'...

Buona misura di prossimità?

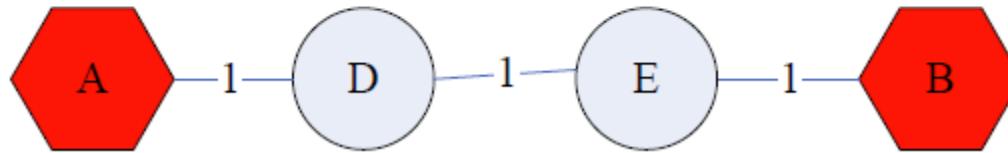
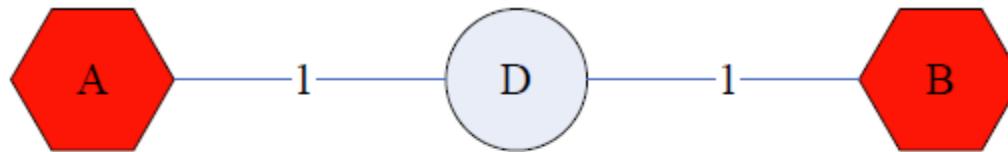
- **Shortest path non va bene**



- **Non efficace sui nodi con degree-1 (E, F, G)!**
- Relazioni multiple

Buona misura di prossimità?

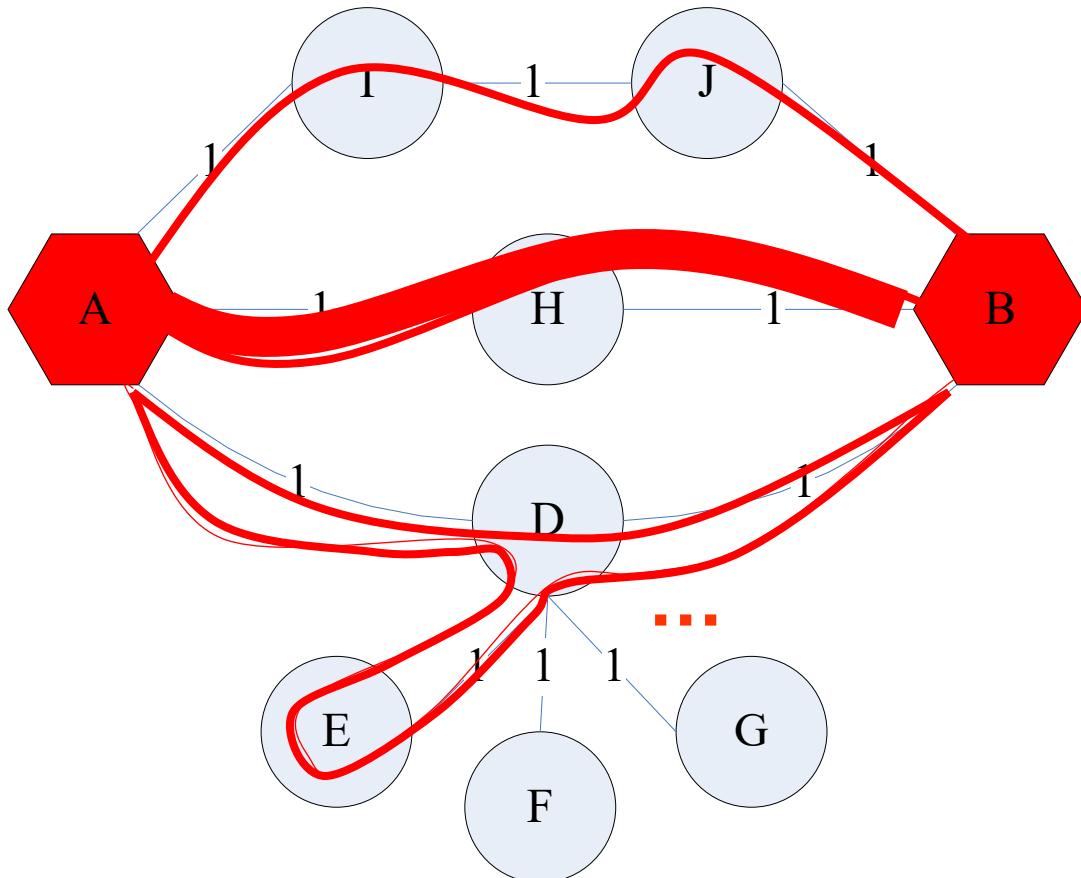
- Network flow non va bene:



- Non punisce cammini lunghi

Qual è una buona misura di prossimità?

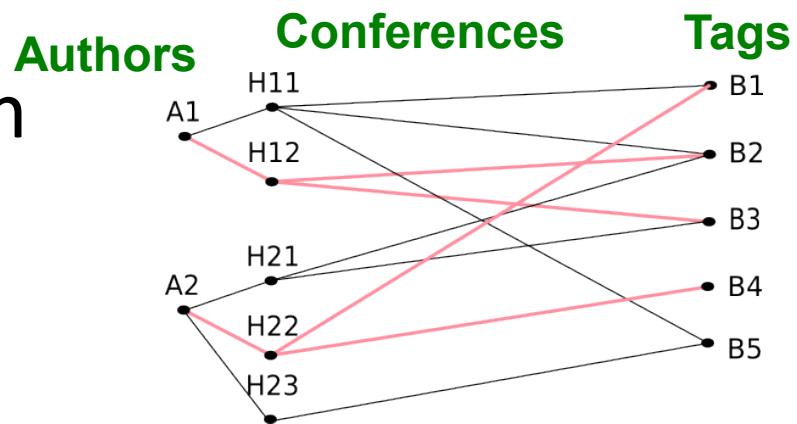
[Tong-Faloutsos, '06]



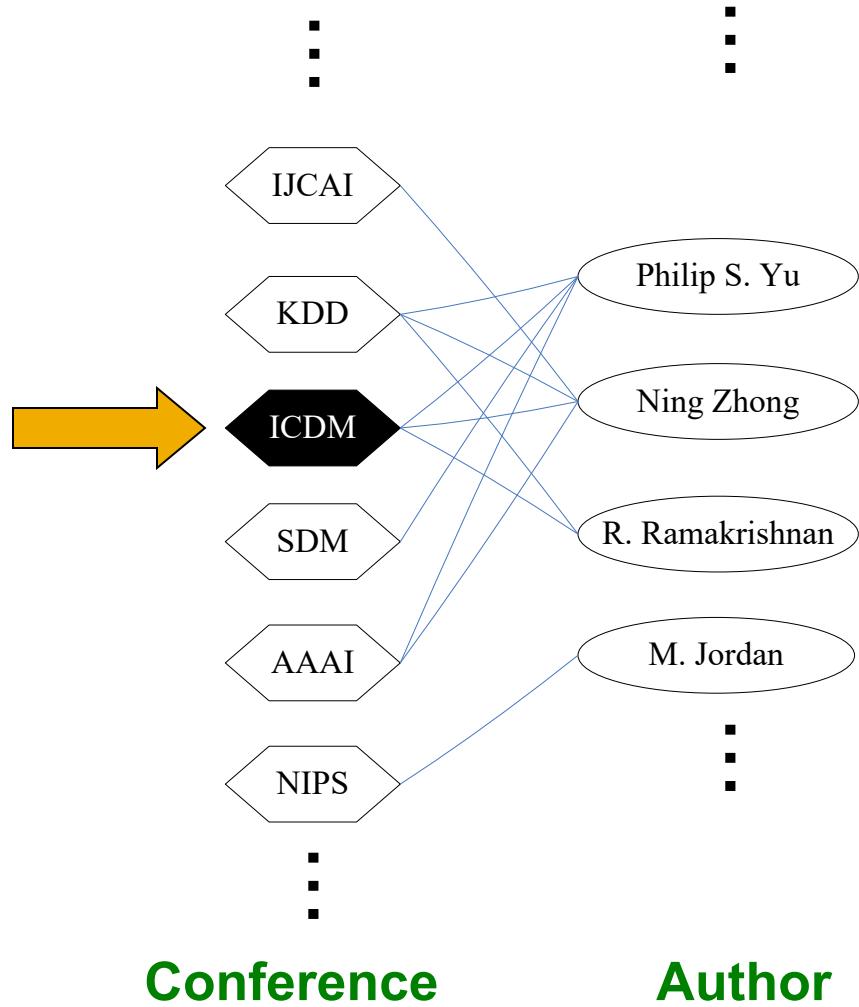
- Connessioni multiple
- Tipologia di connessione
 - Diretta & Indiretta
 - Lunghezza, grado, peso

SimRank: Idea

- **SimRank:** Random walk da **nodo fissato** su un grafo k -partito
- **Setting:** grafo k -partito con k tipi di nodi
 - Es. Autori, Conferenze, Tag
- **Topic Specific PageRank** dal nodo u : **teleport set $S = \{u\}$**
- Gli score daranno la similarità con il nodo u
- **Problema:**
 - Deve essere fatto per ogni nodo u
 - Idoneo per applicazioni sub-Web-scale



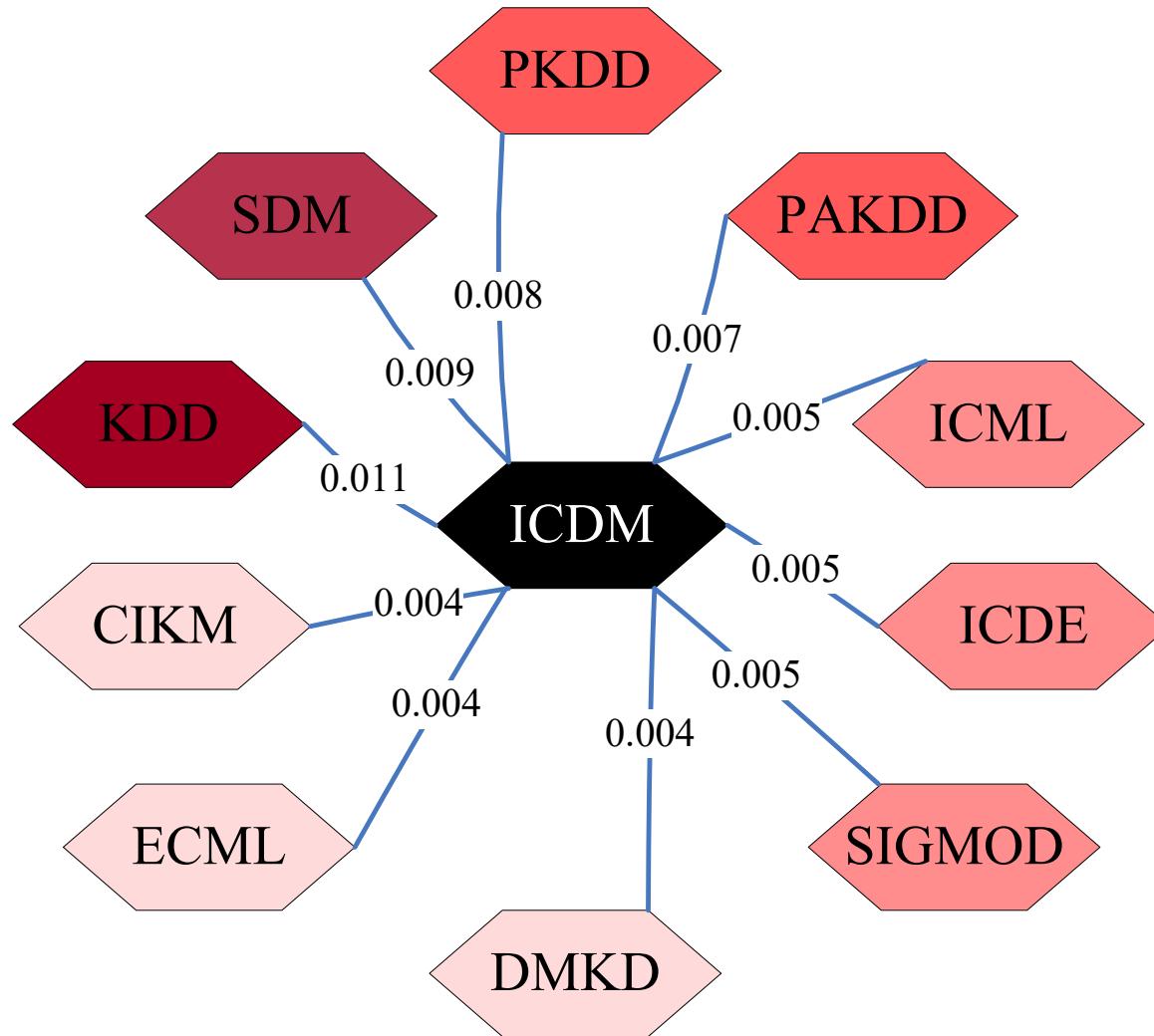
SimRank: Esempio



Q: Quali sono le conferenze simili a **ICDM**?

**A: Topic-Specific
PageRank con teleport
set $S=\{ICDM\}$**

SimRank: Esempio



PageRank: Sommario

- “Normal” PageRank:
 - Teleport uniformemente a tutti i nodi
 - Tutti i nodi stanno nel teleport set con la stessa probabilità: $S = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$
- Topic-Specific PageRank conosciuto come Personalized PageRank:
 - Teleports ad un set specifico di pagine
 - I nodi hanno probabilità differenti: $S = [0.1, 0, 0, 0.2, 0, 0, 0.5, 0, 0, 0.2]$
- Random Walk with Restarts:
 - Topic-Specific PageRank dove il teleport set è sempre lo stesso nodo. $S=[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]$

<https://colab.research.google.com/drive/1bltfV9XMr3ohXHMeUgBbXTmeJZOKM4dF?usp=sharing>

TrustRank: Come combattere il Web Spam

Cosa è il Web Spam?

- **Spamming:**
 - Qualsiasi azione deliberata per far avanzare una pagina web nelle posizioni dei risultati di un search engine, con un valore molto più alto di quello reale
- **Spam:**
 - Pagine Web che sono il risultato di spamming
- **Ampia definizione**
 - SEO = search engine optimization
- Approssimativamente il **10-15%** delle pagine web sono spam

Web Search

- **Primi search engine:**
 - Web crawling
 - Indicizzazione delle pagine con le parole contenute
 - Rispondere alle query (liste di parole) con le pagine che contengono le parole
- **Primi page ranking:**
 - Tentativo di ordinare le pagine in base all'**importanza**
 - **I primi motori di ricerca consideravano:**
 - **(1)** Numero di volte in cui la query appariva nel testo
 - **(2)** Posizione della parola su header, titolo ecc.

I primi spammer

- Il web commerciale...
- **exploit search engine** per fare in modo che gli utenti arrivassero ai loro siti
- **Esempio:**
 - Venditori di magliette che appaiono quando si richiedono “film”
- **Diverse tecniche per ottenere alta rilevanza/importanza per una pagina web**

I primi Spammer: Term Spam

■ Come fare?

- (1) Aggiungere la parola film 1,000 volte nella pagina
- Settare il testo bianco
- (2) Oppure, eseguire la query film sul proprio search engine
- Vedere le pagine che appaiono per prime cosa contengono
- Copiare il contenuto nella propria pagina e renderlo invisibile
- **Queste tecniche ed altre sono note come spam**

This Exploit Allows Me To Hack Any Vibecoder -
YouTube

Soluzione di Google allo spam

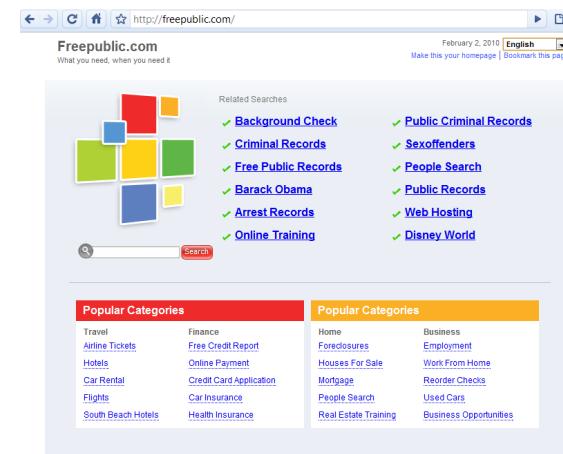
- **Credere in quello che dicono le persone di te e non quello che dici tu su te stesso**
 - Usare le parole che compaiono in link e nel testo circostante
- PageRank tool per misurare l'importanza di una pagina web

Perché Funziona?

- **Il venditore di magliette perde**
 - Parlare di film non gli serve perché gli altri non dicono che lui parla di film
 - La sua pagina non è importante
- **Esempio:**
 - Crea 1,000 pagine, ognuna si collega all'altra con la parola “film” nell’ancora
 - Le pagine non hanno link quindi hanno un PageRank basso
 - Quindi il venditore non può essere importante per i film come IMDB

Google vs. Spammer: Round 2!

- Una volta che Google è diventato il motore di ricerca dominante, gli spammer hanno iniziato a sviluppare modi per ingannare Google
- **Spam farm** sviluppate per concentrare il page rank su un'unica pagina.
- **Link spam:**
 - Creare strutture di link che fanno il boosting del PageRank di una particolare pagina



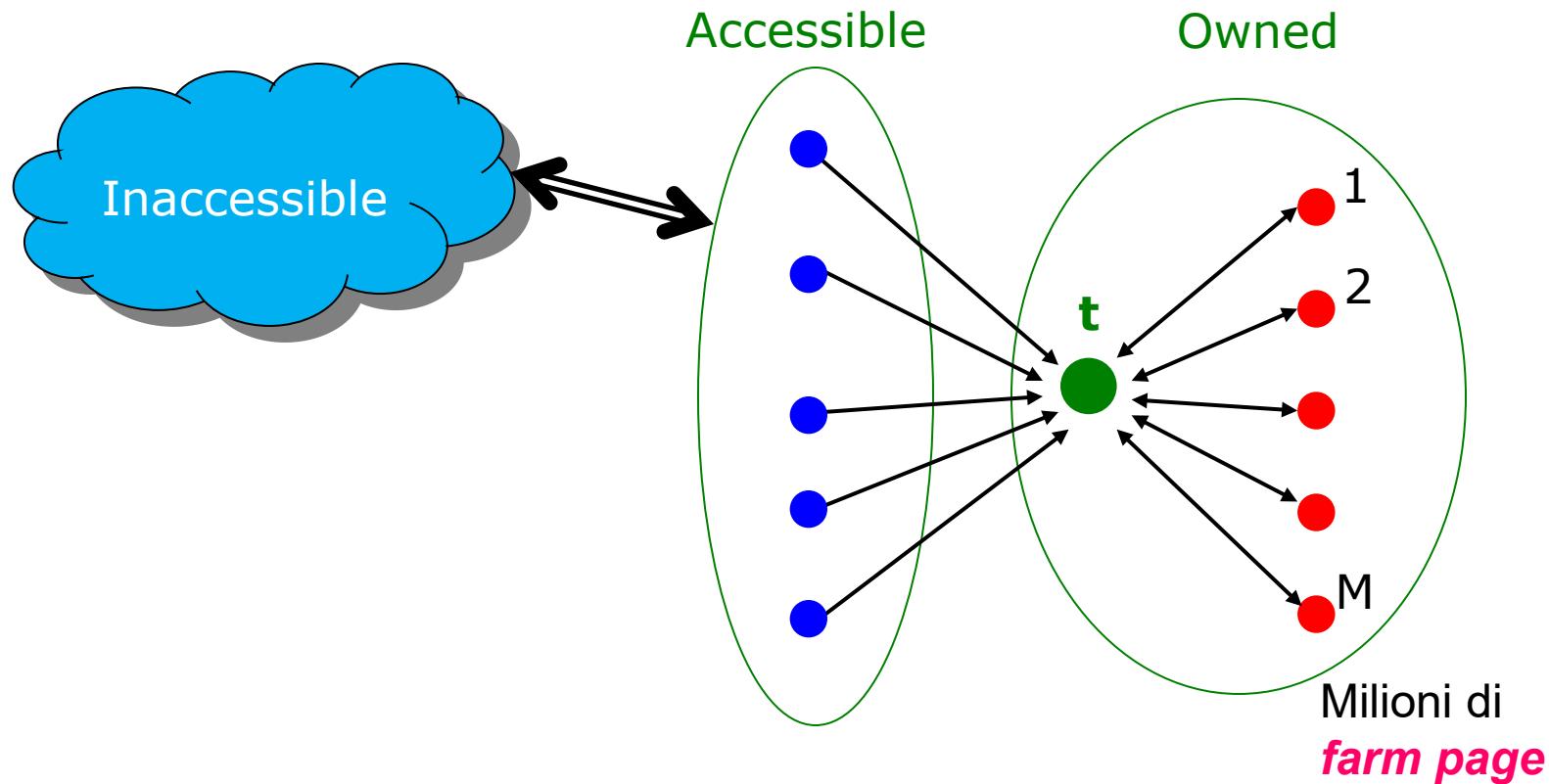
Link Spamming

- **Tre tipi di pagine dal punto di vista dello spammer**
 - **Pagine inaccessibili**
 - **Pagine accessibili**
 - e.g., blog commenti
 - Lo spam posta link sulle sue pagine
 - **Possedute dallo spammer**
 - Controllate totalmente dallo spammer
 - Su diversi domini

Link Farm

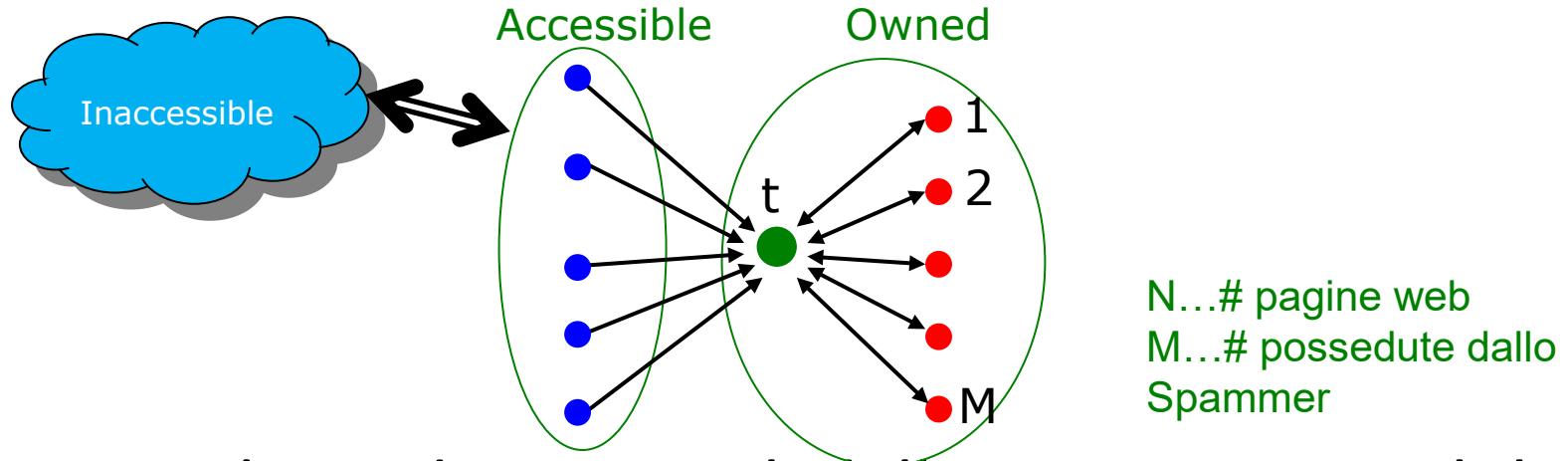
- **Goal dello Spammer:**
 - Massimizzare il PageRank della pagina target t
- **Tecnica:**
 - Ottenere il maggior numero di link da pagine accessibili per la pagina di destinazione t
 - Crea "link farm" per ottenere PageRank con effetto moltiplicatore

Link Farm



Uno delle più comuni ed efficaci organizzazioni
per una link farm

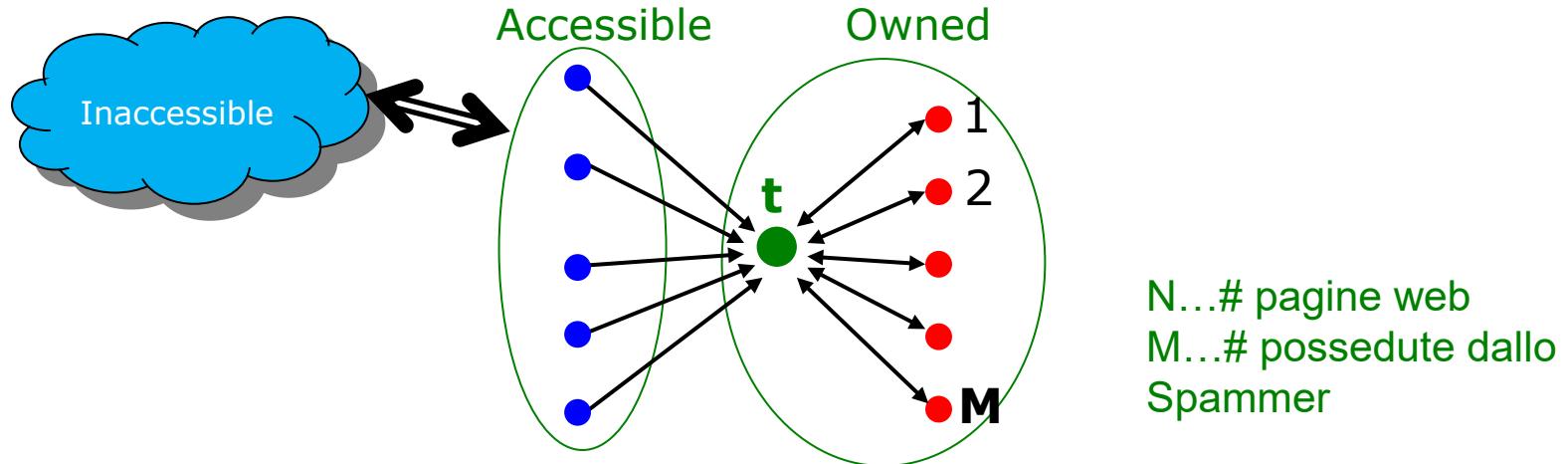
Analisi



- x : Contributo di PageRank dalle pagine accessibili
- y : PageRank della pagina target t
- Rank di ogni pagina “farm” = $\frac{\beta y}{M} + \frac{1-\beta}{N}$
- $y = x + \beta M \left[\frac{\beta y}{M} + \frac{1-\beta}{N} \right] + \frac{1-\beta}{N}$
 $= x + \beta^2 y + \frac{\beta(1-\beta)M}{N} + \boxed{\frac{1-\beta}{N}}$

Molto piccolo
Risolviamo per y
- $y = \frac{x}{1-\beta^2} + c \frac{M}{N}$ dove $c = \frac{\beta}{1+\beta}$

Analisi



- $y = \frac{x}{1-\beta^2} + c \frac{M}{N}$ con $c = \frac{\beta}{1+\beta}$
- Per $\beta = 0.85$, $1/(1-\beta^2) = 3.6$
- Effetto moltiplicatore acquisito per PageRank
- Rendendo M grande, possiamo ottenere **y grande quanto vogliamo**

TrustRank: combattere il Web Spam

Combattere lo spam

■ Combattere lo spam

- Usare metodi statistici
- Simili al filtering delle email di spam
- Trovare pagine duplicate

■ Combattere il link spam

- Trovare e mettere in blacklist strutture che sembrano simili alle spam farm
- **TrustRank** = topic-specific PageRank con teleport a set di pagine trusted
 - Esempio: domini tipo .edu, ecc.

TrustRank: Idea

- **Principio di base: Approssimare l'isolamento**
E' raro che una pagina "buona" punti ad una pagina "cattiva" (spam)
- Insieme di **pagine seed** dal web
- Un **oracolo (umano)** per identificare le pagine del seed set
 - **Task dispendioso**, il seed set dovrebbe essere più piccolo possibile

Trust Propagation

- Il seed set di pagine **buone** è il set di **trusted pages**
- Effettuare un topic-sensitive PageRank con **teleport set = trusted pages**
 - **Propagare la trust attraverso i link:**
 - Ogni pagina riceverà un valore di trust tra **0 e 1**
- **Soluzione 1:** Usare un valore **threshold** e marcare tutte le pagine sotto il threshold come **spam**

Modello semplice: Trust Propagation

- Ad ogni pagina web del trust set associare un valore di “trust” pari a 1
- Supponiamo che il trust di una pagina p sia t_p
 - La pagina p ha un insieme di out-link o_p
- Per ogni $q \in o_p$, p conferisce la trust a q
 - $\beta t_p / |o_p|$ for $0 < \beta < 1$
- Trust è additiva
 - Trust di p è la somma di tutte le trust conferite a p dalle pagine che la linkano
- Notiamo la similarità a Topic-Specific PageRank
 - Entro un fattore di scala, TrustRank = PageRank con le pagine trust come teleport set

Perché è una buona idea

- **Attenuazione del Trust:**
 - Il degree di trust conferito da ogni pagina trusted decresce con la distanza nel grafo
- **Trust splitting:**
 - Maggiore è il numero di out-link da una pagina, tanto meno l'autore della pagina indica ogni out-link
 - Trust è **spartito** attraverso gli out-link

Scegliere il Seed Set

- **Due considerazioni in conflitto:**
 - Manualmente curato. Le pagine seed sono scelte dall'uomo. Questo deve essere il più piccolo possibile
 - Deve assicurare che ogni **pagina buona** ottenga un adeguato trust rank, quindi abbiamo bisogno che le pagine buone siano raggiungibili dal seed set con uno short path

Approcci per la costruzione del Seed Set

- Supponiamo di volere k pagine seed
- Come facciamo?
- **(1) PageRank:**
 - Scegliere le top k pagine ottenute col PageRank
 - In teoria non si possono ottenere pagine cattive con un rank alto
- **(2) Usare domini trusted** la cui appartenenza è controllata, es. .edu, .mil, .gov

HITS: Hubs and Authorities

Hubs and Authorities

- **HITS (Hypertext-Induced Topic Selection)**
 - Misura di importanza delle pagine web simile a PageRank
 - Proposto nello stesso periodo del PageRank ('98)
- **Goal:** Supponiamo di voler trovare buoni quotidiani
 - Non proprio quotidiani. Trovare “esperti” – persone che linkano in modo coordinato a buoni quotidiani
- **Idea: Ancora una volta i link come voti**
 - Una pagina è più importante se ha più link
 - Entranti? Uscenti?

Troviamo i quotidiani

■ Hubs e Authorities

Ogni pagina ha 2 score:

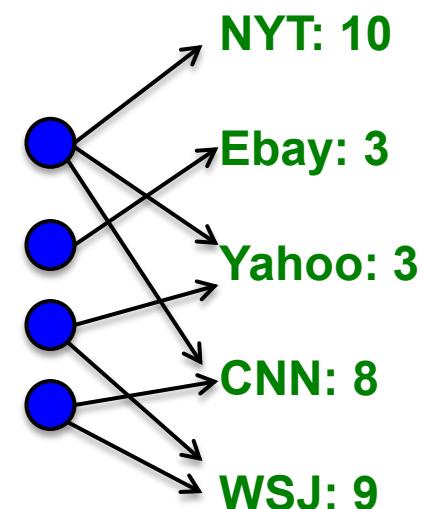
- **Qualità di esperto (hub):**

- Voti totali ricevuti dalle authority che puntano a loro

- **Qualità sui contenuti (authority):**

- Voti totali ricevuti dagli esperti

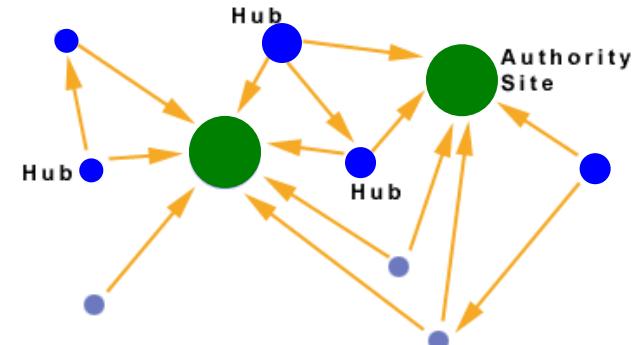
■ Principio di miglioramenti progressivi



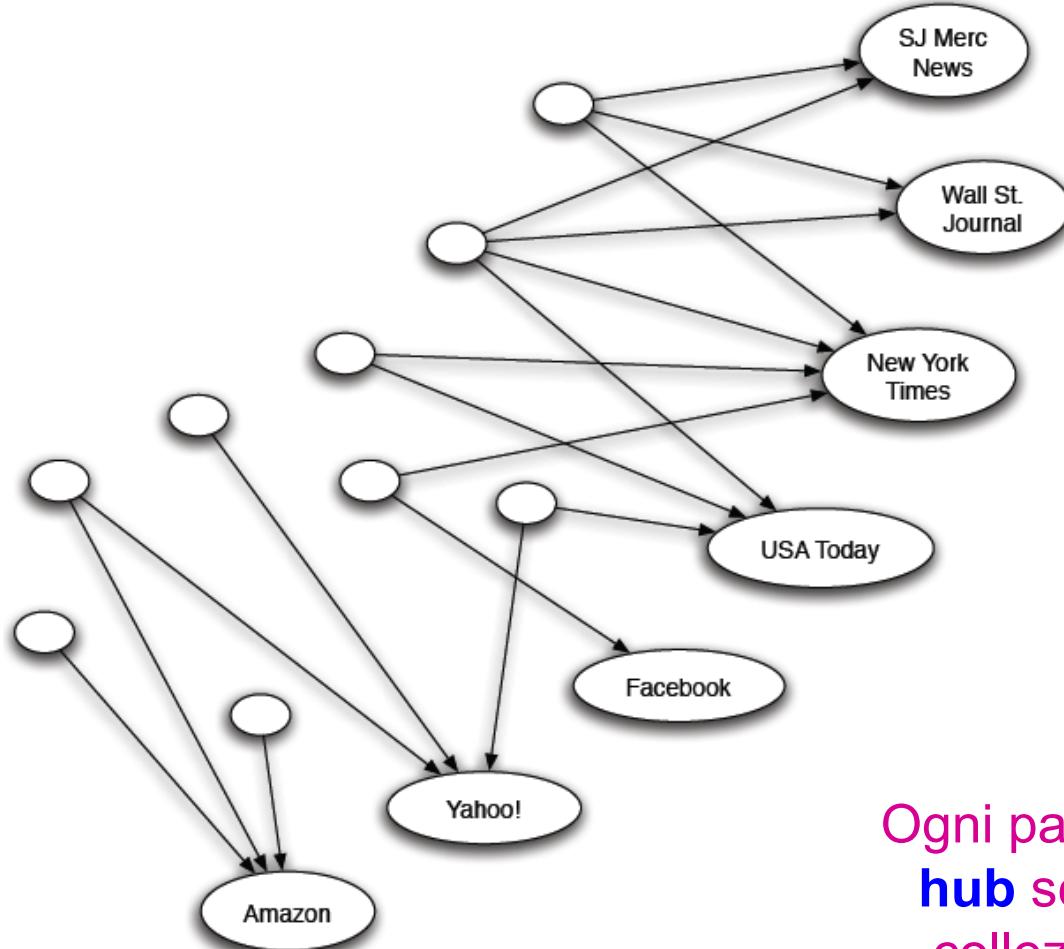
Hubs and Authorities

Le pagine interessanti ricadono in due classi:

1. **Authority** sono pagine che contengono informazioni utili
 - Home page dei quotidiani
 - Home page di corsi
 - Home page di costruttori di automobili
2. **Hub** pagine che si collegano alle authority
 - Elenchi di quotidiani
 - Elenchi di corsi
 - Elenchi dei costruttori di automobili europei

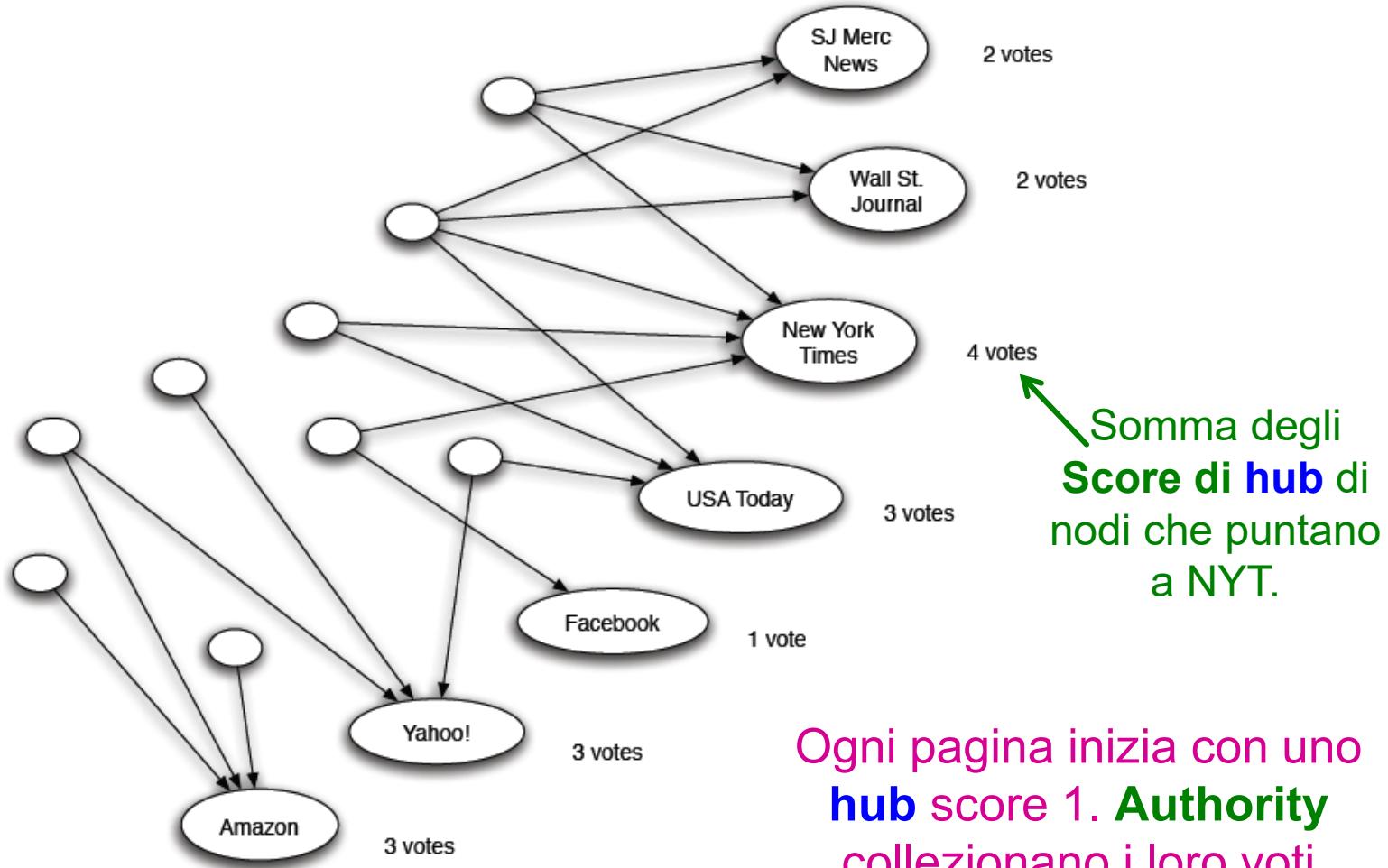


Contare in-link: Authority



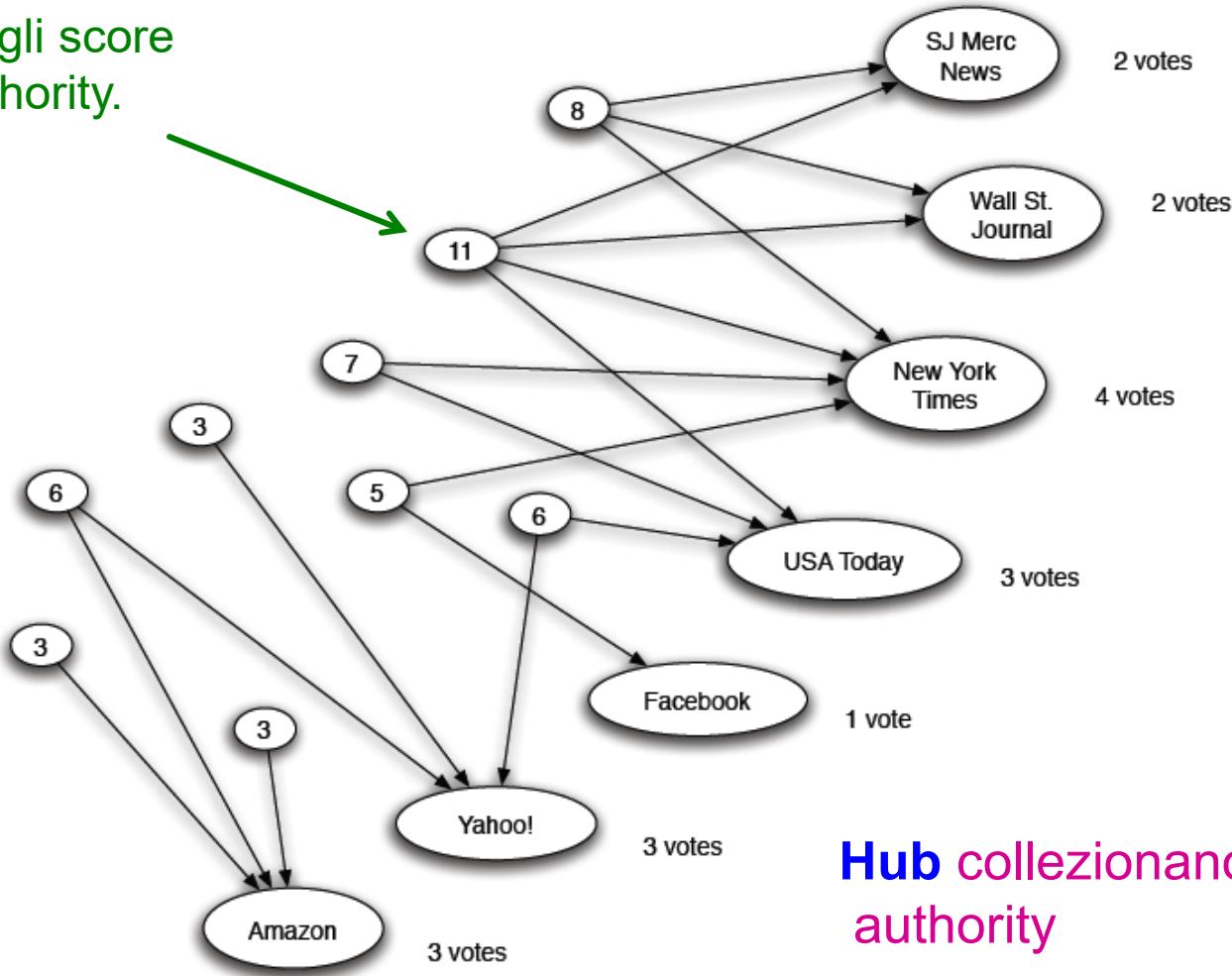
Ogni pagina inizia con uno **hub** score 1. **Authority** collezionano i loro voti

Contare gli in-link: Authority



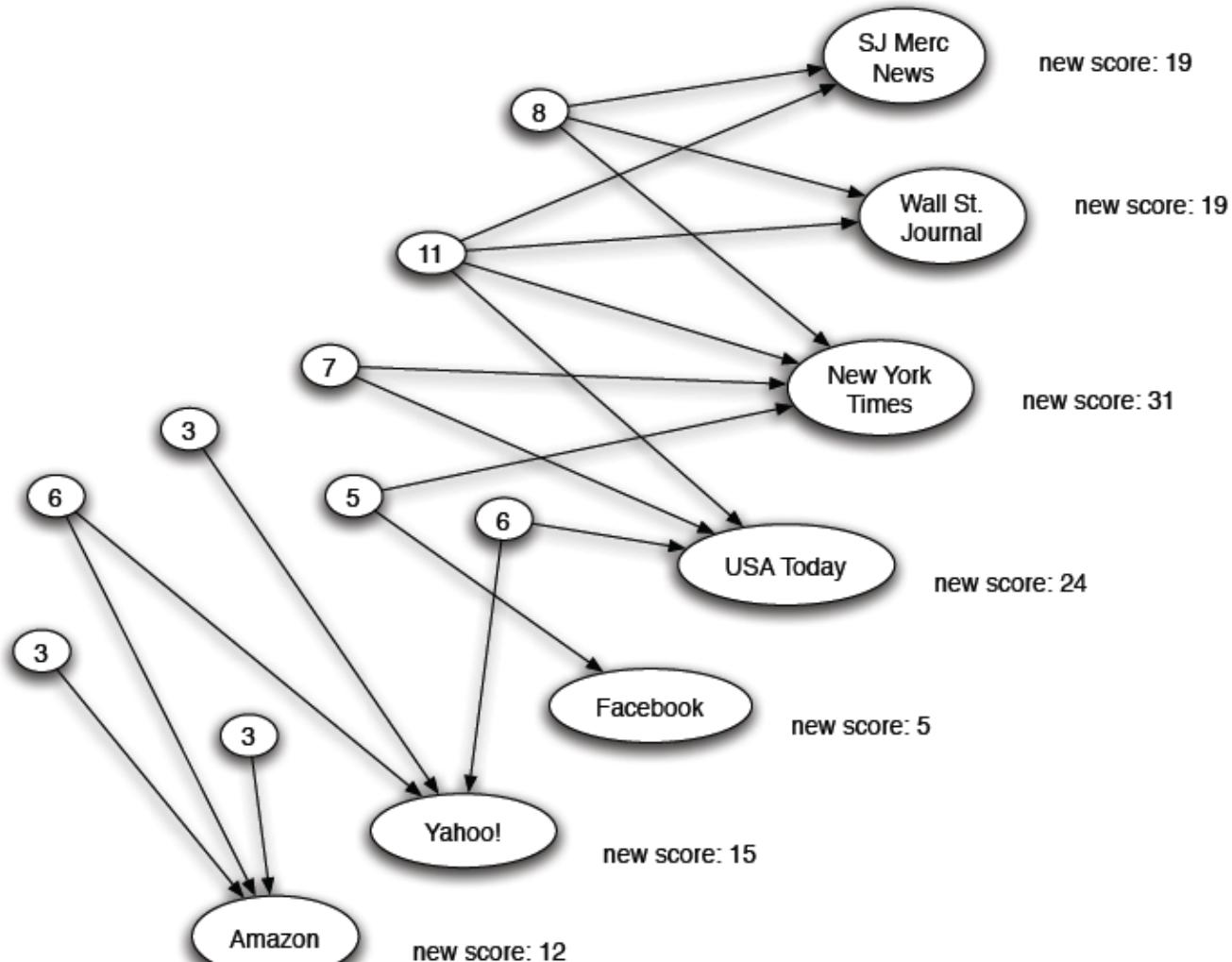
Qualità di esperto: Hub

Somma degli score
degli authority.



Hub collezionano i voti delle
authority

Ridistribuire i voti



Authority nuovamente collezionano
Gli score degli **hub**

Definizione di mutua ricorsione

- **Un buon hub linka diverse buone authority**
- **Una buona authority è linkata da diversi buoni hub**
- **Modelliamo usando due score per ogni nodo**
 - **Hub** score e **Authority** score
 - Rappresentati dai vettori h e a

Hubs and Authorities

- **Ogni pagina i ha 2 score:**

- Authority score: a_i
- Hub score: h_i

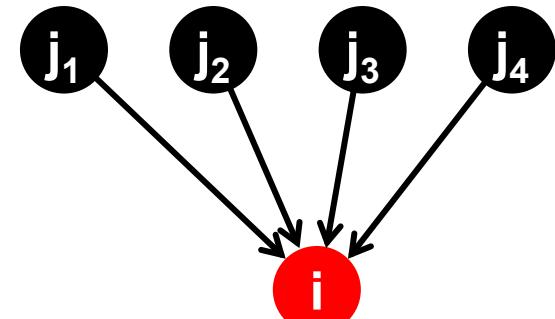
Algoritmo HITS:

- Inizializza: $a_j^{(0)} = 1/\sqrt{N}$, $h_j^{(0)} = 1/\sqrt{N}$

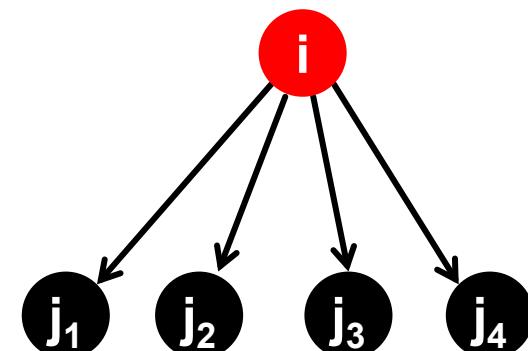
- Itera fino alla convergenza:

- $\forall i$: Authority: $a_i^{(t+1)} = \sum_{j \rightarrow i} h_j^{(t)}$
- $\forall i$: Hub: $h_i^{(t+1)} = \sum_{i \rightarrow j} a_j^{(t)}$
- $\forall i$: normalizza:

$$\sum_i (a_i^{(t+1)})^2 = 1, \sum_j (h_j^{(t+1)})^2 = 1$$



$$a_i = \sum_{j \rightarrow i} h_j$$



$$h_i = \sum_{i \rightarrow j} a_j$$

Hubs and Authorities

- **HITS converge ad un singolo punto stabile**
- **Notazione:**
 - Vettore $a = (a_1 \dots, a_n)$, $h = (h_1 \dots, h_n)$
 - Matrice di adiacenza A ($N \times N$): $A_{ij} = 1$ se $i \rightarrow j$, 0 altrimenti
- **Quindi $h_i = \sum_{i \rightarrow j} a_j$**
può essere riscritto come $h_i = \sum_j A_{ij} \cdot a_j$
Ne segue che: $h = A \cdot a$
- **In modo simile , $a_i = \sum_{j \rightarrow i} h_j$**
può essere riscritto come $a_i = \sum_j A_{ji} \cdot h_j$
$$a = A^T \cdot h$$

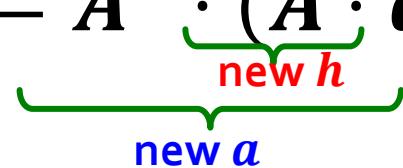
Hubs and Authorities

■ Algoritmo HITS in notazione vettoriale

- Set: $a_i = h_i = \frac{1}{\sqrt{n}}$

Ripeti fino alla convergenza:

- $h = A \cdot a$
- $a = A^T \cdot h$
- Normalizza a e h
- Quindi: $a = A^T \cdot \underbrace{(A \cdot a)}_{\text{new } h}$



Criterio di convergenza:

$$\sum_i (h_i^{(t)} - h_i^{(t-1)})^2 < \varepsilon$$

$$\sum_i (a_i^{(t)} - a_i^{(t-1)})^2 < \varepsilon$$

a è aggiornato (in 2 step):

$$a = A^T(A \cdot a) = (A^T A) a$$

h è aggiornato (in 2 step):

$$h = A(A^T h) = (A A^T) h$$

Power method

Esistenza e unicità

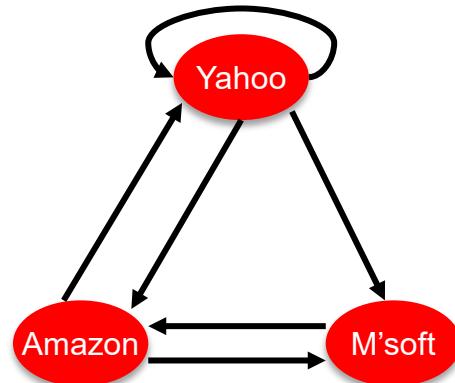
- $h = \lambda A a$ $\lambda = 1 / \sum h_i$
- $a = \mu A^T h$ $\mu = 1 / \sum a_i$
- $h = \lambda \mu A A^T h$
- $a = \lambda \mu A^T A a$

- Sotto una assunzione ragionevole di A ,
HITS **converge a dei vettori h^* e a^* :**
 - h^* **autovettore principale** di $A A^T$
 - a^* **autovettore principale** di $A^T A$

Esempio HITS

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$



$$h(\text{yahoo}) = .58 \quad .80 \quad .80 \quad .79 \quad \dots \quad .788$$

$$h(\text{amazon}) = .58 \quad .53 \quad .53 \quad .57 \quad \dots \quad .577$$

$$h(\text{m'soft}) = .58 \quad .27 \quad .27 \quad .23 \quad \dots \quad .211$$

$$a(\text{yahoo}) = .58 \quad .58 \quad .62 \quad .62 \quad \dots \quad .628$$

$$a(\text{amazon}) = .58 \quad .58 \quad .49 \quad .49 \quad \dots \quad .459$$

$$a(\text{m'soft}) = .58 \quad .58 \quad .62 \quad .62 \quad \dots \quad .628$$

PageRank e HITS

- **PageRank and HITS due soluzioni per lo stesso problema:**
 - **Qual è il valore di un in-link da u a v ?**
 - Nel PageRank, the il valore del link dipende dai coloro che *linkano a u*
 - Nel modello HITS, dipende dal valore degli altri link *uscenti da u*
- **I destini di PageRank e HITS dopo il 1998 sono stati differenti**