

# Data Mining

## Linkanalyse

**Dr. Hanna Köpcke**  
**Wintersemester 2020**

**Abteilung Datenbanken, Universität Leipzig**  
**<http://dbs.uni-leipzig.de>**

# Übersicht

## Hochdimensionale Daten

Clustering

Dimensions-  
reduktion

Empfehlungs-  
systeme

Assoziations-  
regeln

Locality Sensitive  
Hashing

Supervised ML

## Graphdaten

Community  
Detection

PageRank

Web Spam

## Datenströme

Windowing

Filtern

Momente

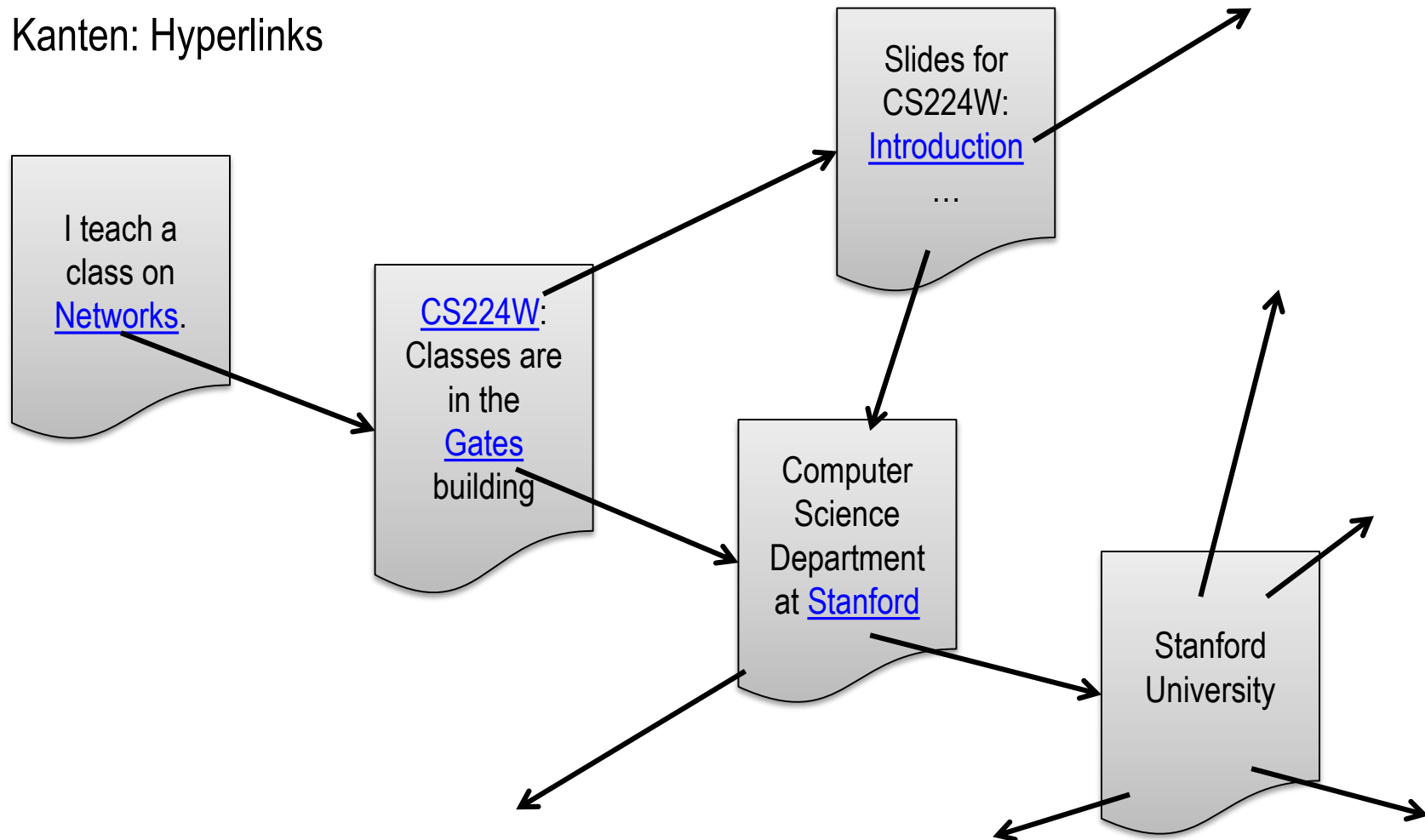
Web Advertising

# Inhaltsverzeichnis

- **Einführung**
- **PageRank**
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- **Link-Spam**

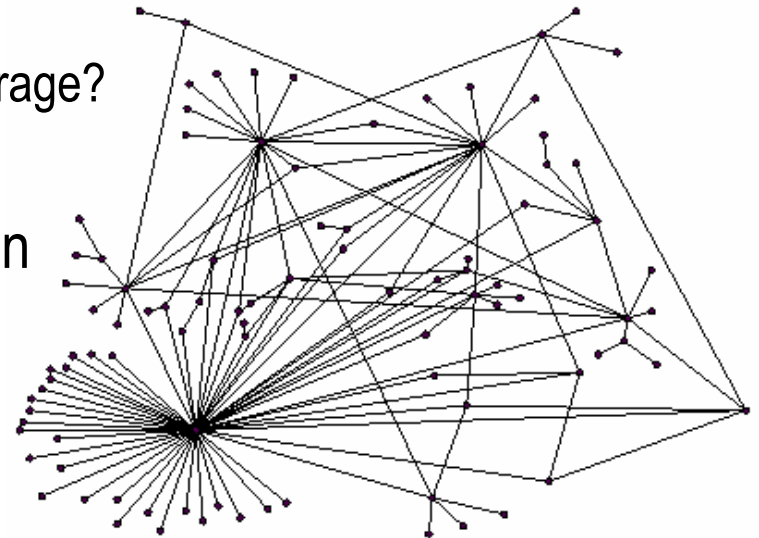
# Graphdaten: World Wide Web

- Das Web als ein gerichteter Graph:
  - Knoten: Webseiten
  - Kanten: Hyperlinks



# World Wide Web

- Wie kann das Web organisiert werden?
- 1. Versuch: Webverzeichnis: Yahoo, Curlie (DMOZ)
- 2. Versuch: Suchmaschine
  - Webcrawler durchlaufen Web
  - Inverted Index für Suchanfragen
- Herausforderungen für Suchmaschinen
  1. Welche Seiten sind vertrauenswürdig und enthalten tatsächlich die relevanten Informationen zu einer Anfrage?
  2. Welches ist die „beste“ Antwort auf eine Anfrage?
- Verwendung der **Hyperlink-Struktur** um die Bedeutung einer Webseite zu ermitteln

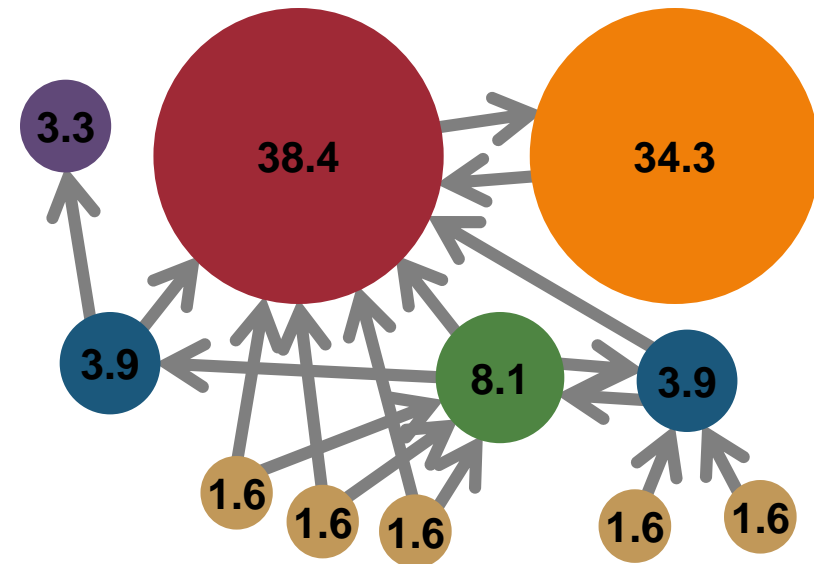


# Inhaltsverzeichnis

- Einführung
- PageRank
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- Link-Spam

# PageRank

- **Idee:** Webseite ist wichtig, falls sie von vielen Nutzern besucht wird
- Anstatt das Verhalten direkt zu beobachten, wird angenommen, dass Nutzer den Hyperlinks zufällig folgen: **Random-Surfer-Model**
  - Beginne auf einer beliebigen Seite
  - Wiederhole: Folge einem zufällig ausgewählten Hyperlink dieser Seite
- **PageRank** einer Webseite: „*Neigung*“, dass ein *Random-Surfer* diese *Webseite besucht*
- Folgerungen:
  - Webseite ist *wichtig (hoher PageRank)*, falls viele Hyperlinks auf sie verweisen
  - Hyperlinks von *wichtigen* Webseiten haben höheres Gewicht

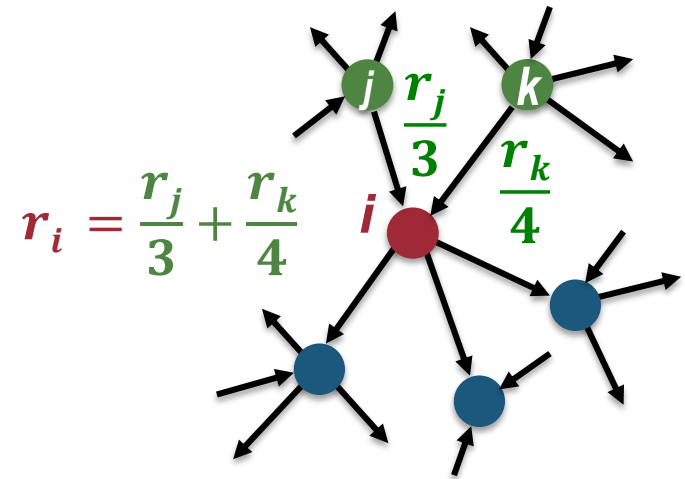


# Einfacher PageRank

- Einfacher PageRank:

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

- $d_j$ : Anzahl der Webseiten mit mind. einem Hyperlink ausgehend von  $j$  (Outdegree)
- $j \rightarrow i$ : Summe läuft über alle auf  $i$  verweisenden Webseiten  $j$



- Sei  $n$  die Anzahl der Knoten (Webseiten)
- Lineares Gleichungssystem aus  $n$  Gleichungen und  $n$  Unbekannten
- Eindeutige Lösung falls Einschränkung, z.B. auf  $\sum_i r_i = n$
- *Gaußsches Eliminationsverfahren* nur bei Graphen mit kleinem  $n$  möglich
- *Alternative Berechnung für riesigen Webgraph notwendig*



# Matrixformulierung

- Stochastische Adjazenzmatrix  $M$  ( $n \times n$ )

$$M_{ij} = \begin{cases} \frac{1}{d_j}, & j \rightarrow i \\ 0, & \text{sonst} \end{cases}$$

- Für den PageRank-Vektor  $\mathbf{r} = (r_1, r_2, \dots, r_n)^T$  gilt:

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

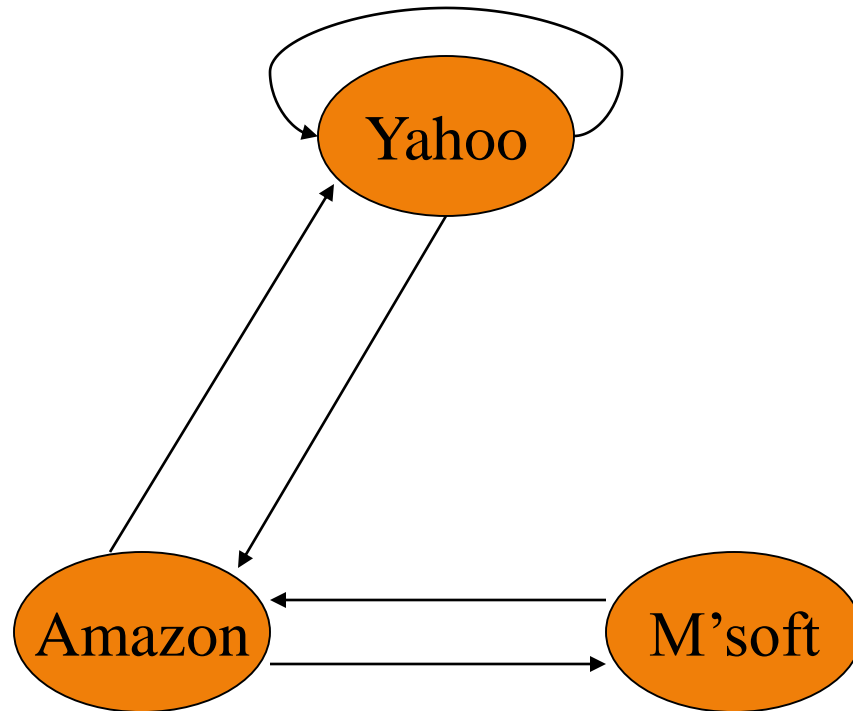
$$\mathbf{r} = M \cdot \mathbf{r}$$

The diagram illustrates the matrix equation  $\mathbf{r} = M \cdot \mathbf{r}$ . On the left, a green vertical rectangle represents the vector  $\mathbf{r}_i$ . This is followed by an equals sign. In the center, a red square matrix represents  $M$ , with a row index  $i$  and a column index  $j$ . To the right of the matrix is a dot, followed by another green vertical rectangle representing the vector  $\mathbf{r}_j$ .

# Power Iteration Method

- *Power Iteration Method:*
  - Zu Beginn:  $\mathbf{r}^{(0)} = (1, 1, \dots, 1)^T$
  - Iteration:  $\mathbf{r}^{(t+1)} = M \cdot \mathbf{r}^{(t)}$
  - Stopp, falls  $\|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}\| < \varepsilon$
- Vektor  $\mathbf{r}^{(t)} = M^t \cdot \mathbf{r}^{(0)}$  gibt mit  $\frac{r_i^{(t)}}{n}$  die Wahrscheinlichkeit, dass sich der Random-Surfer zum Zeitpunkt  $t$  auf einer Seite  $i = 1, \dots, n$  befindet
- Falls Algorithmus konvergiert, gilt mit  $\mathbf{r} := \lim_{t \rightarrow \infty} M^t \cdot \mathbf{r}^{(0)}$ 
$$\mathbf{r} = M \cdot \mathbf{r}$$
- Vektor  $\mathbf{r}$  gibt den PageRank

# Beispiel: Mini-WWW



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

$$\begin{aligned}y &= \frac{1}{2} y + \frac{1}{2} a \\a &= \frac{1}{2} y + m \\m &= \frac{1}{2} a\end{aligned}$$

# Beispiel: Mini-WWW

- Iteration  $\mathbf{r}^{(t+1)} = M \cdot \mathbf{r}^{(t)}$  mit  $\mathbf{r}^{(t)} = (y^{(t)}, a^{(t)}, m^{(t)})$

$$- y^{(t+1)} = \frac{1}{2}y^{(t)} + \frac{1}{2}a^{(t)}$$

$$- a^{(t+1)} = \frac{1}{2}y^{(t)} + m^{(t)}$$

$$- m^{(t+1)} = \frac{1}{2}a^{(t)}$$

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

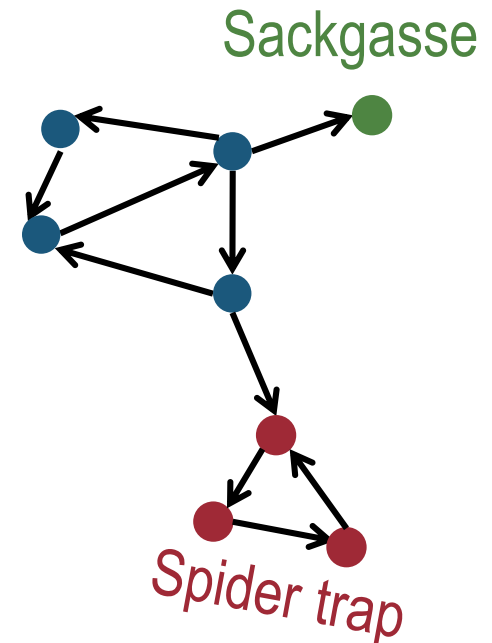
y		1	1	5/4	9/8		6/5
a	=	1	3/2	1	11/8	...	6/5
m		1	1/2	3/4	1/2		3/5

# Inhaltsverzeichnis

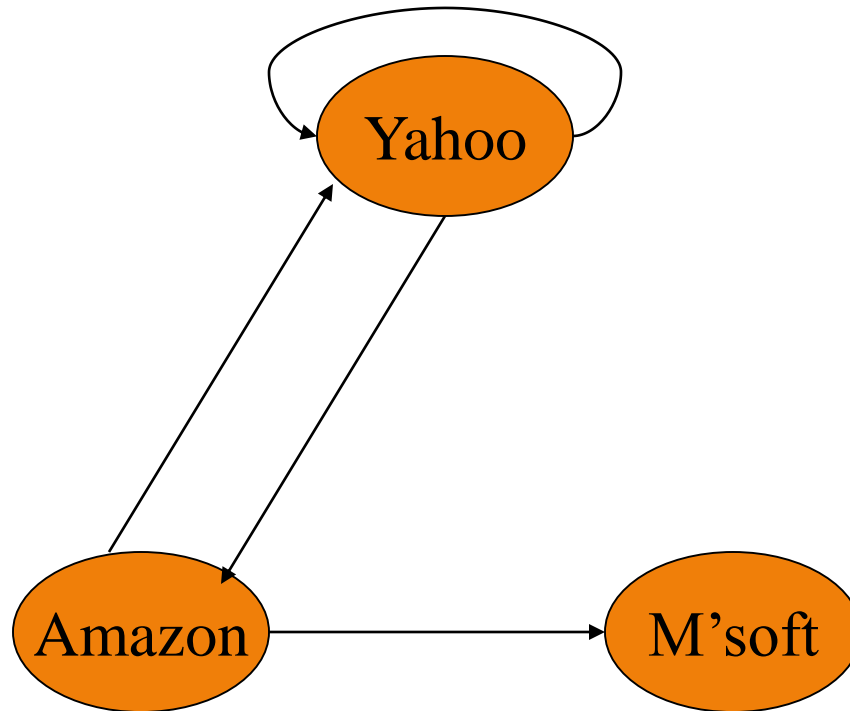
- **Einführung**
- **PageRank**
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- **Link-Spam**

# PageRank: Probleme

- Konvergenz der Power-Iteration-Method ist nicht sicher
- Das reale WWW ist komplex
- **Sackgassen:** Webseiten ohne Hyperlinks
  - Random-Surfer kann Sackgasse nicht verlassen
  - Gewichtungen „verschwinden“ über Sackgasse
- **Spider Traps:** Gruppen von Webseiten ohne ausgehende Hyperlinks
  - Random-Surfer kann Gruppe nicht verlassen
  - Gewichtungen konzentrieren sich auf Gruppe



# Problem: Sackgasse

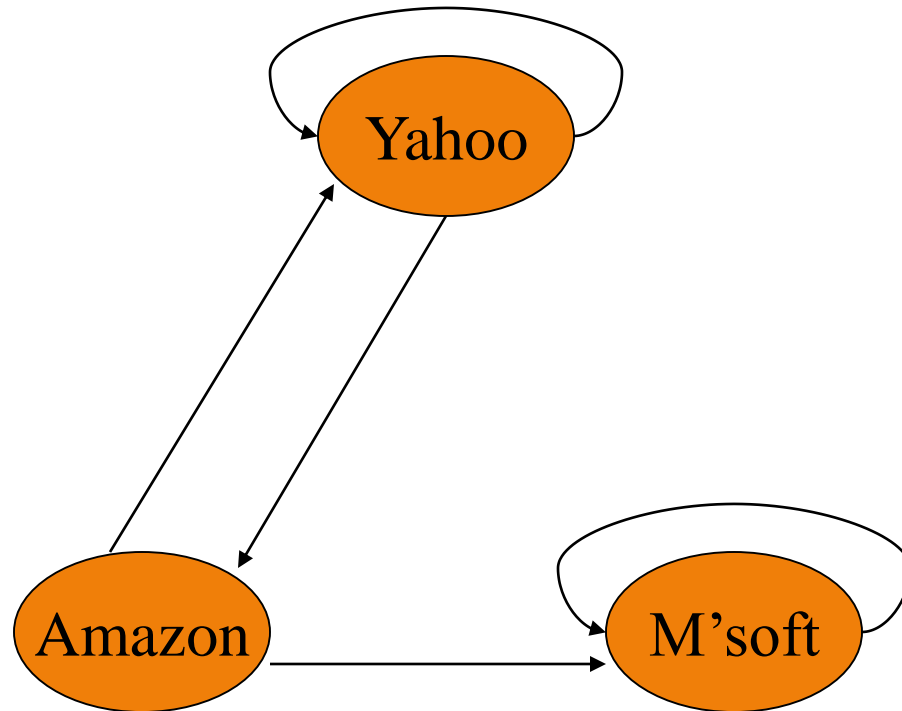


M      y      a      m

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

$$\begin{aligned} y &= \frac{1}{2} y + \frac{1}{2} a \\ a &= \frac{1}{2} y \\ m &= \frac{1}{2} a \end{aligned}$$

# Problem: Spider Trap



M      y      a      m

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

$$\begin{aligned}y &= \frac{1}{2} y + \frac{1}{2} a \\a &= \frac{1}{2} y \\m &= \frac{1}{2} a + m\end{aligned}$$

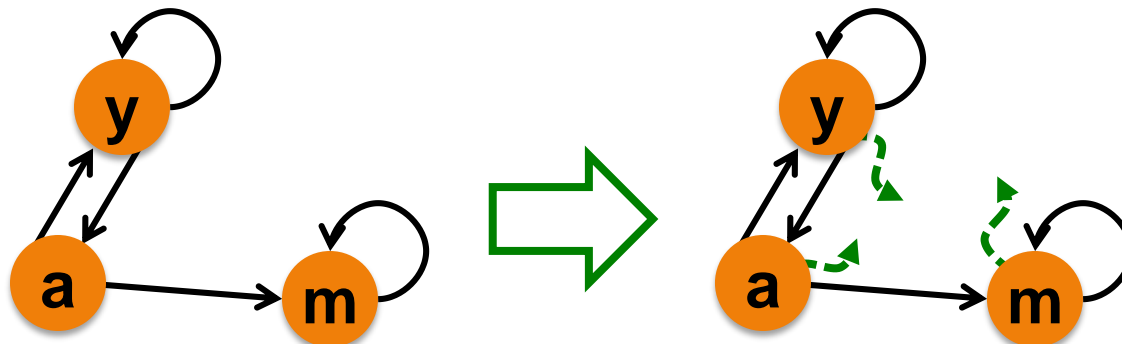


# PageRank (allgemeine Form)

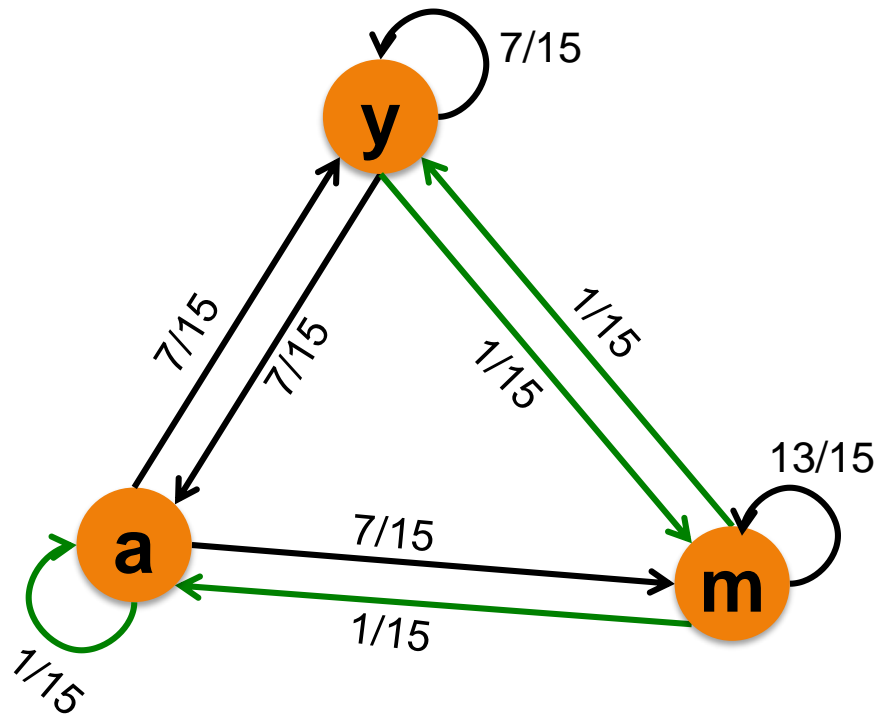
- PageRank [Brin & Page, 1998, <http://infolab.stanford.edu/~backrub/google.html>]

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

- Interpretation:
  - Mit Wahrscheinlichkeit  $\beta$ , folge einem zufällig gewählten Hyperlink
  - Mit Wahrscheinlichkeit  $1 - \beta$ , wechsel zu einer zufälligen Seite
- Gewöhnlich setzt man  $\beta$  auf einen Wert zwischen 0.8 und 0.9
- Annahme: Keine Sackgassen*



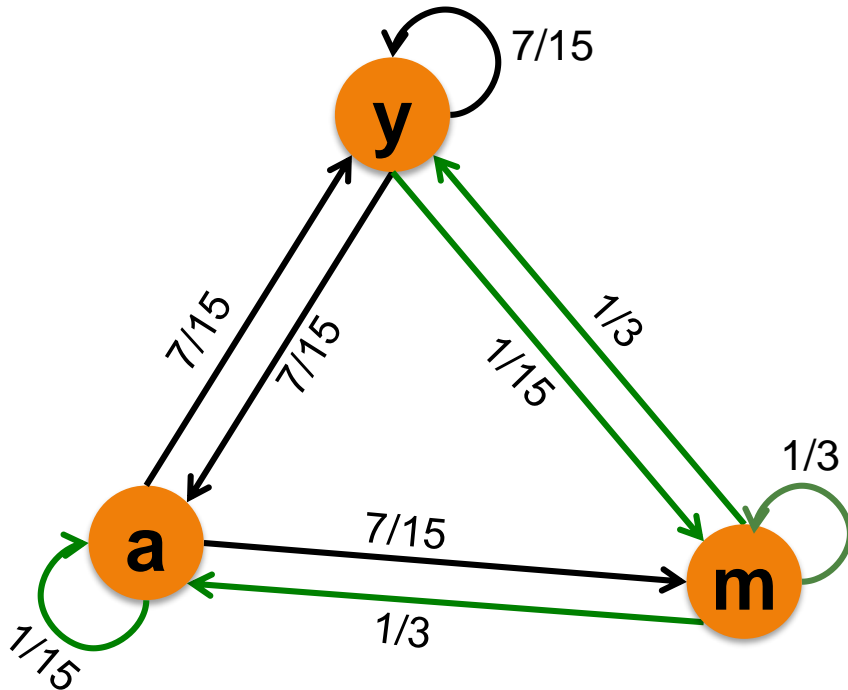
# Beispiel



$$\begin{array}{c} \mathbf{M} \\ 0.8 \end{array} \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \\
 = \begin{array}{c} y \\ a \\ m \end{array} \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

y		1	1	0.84	0.78		7/11
a	=	1	0.6	0.60	0.54	...	5/11
m		1	1.4	1.56	1.69		21/11

# Beispiel: mit Sackgasse



**Annahme:** falls Knoten eine Sackgasse ist, wechsel mit Wahrscheinlichkeit 1 zu einer zufälligen Seite

$$0.8 \begin{matrix} & \mathbf{M} \\ \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \end{bmatrix} & + 0.2 ? \end{matrix}$$

$$= \begin{matrix} & \begin{matrix} y & a & m \end{matrix} \\ \begin{matrix} y \\ a \\ m \end{matrix} & \begin{bmatrix} 7/15 & 7/15 & \mathbf{1/3} \\ 7/15 & 1/15 & \mathbf{1/3} \\ 1/15 & 7/15 & \mathbf{1/3} \end{bmatrix} \end{matrix}$$



Alternative Berechnung  
des PageRank

# PageRank: Vollständiger Algorithmus

- Eingabe: Gerichteter Graph  $G$  aus  $n$  Knoten (inkl. Sackgassen und Spider Traps) und Parameter  $\beta$
- Ausgabe: PageRank Vector  $r^{new}$

- Intitialisiere:  $\forall i: r_i^{old} = 0, r_i^{new} = 1$
- Wiederhole
  1. Berechne  $\sum_j |r_j^{new} - r_j^{old}|$  und Abbruch, falls  $< \varepsilon$
  2.  $r^{old} \leftarrow r^{new}$
  3.  $\forall i: r_i^{new} \leftarrow \sum_{j \rightarrow i} \beta \frac{r_j^{old}}{d_j}$
  4.  $S \leftarrow \sum_i r_i^{new}$
  5.  $\forall i: r_i^{new} = r_i^{new} + \frac{1-S}{n}$

# Inhaltsverzeichnis

- **Einführung**
- **PageRank**
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- **Link-Spam**

# Berechnung des PageRank

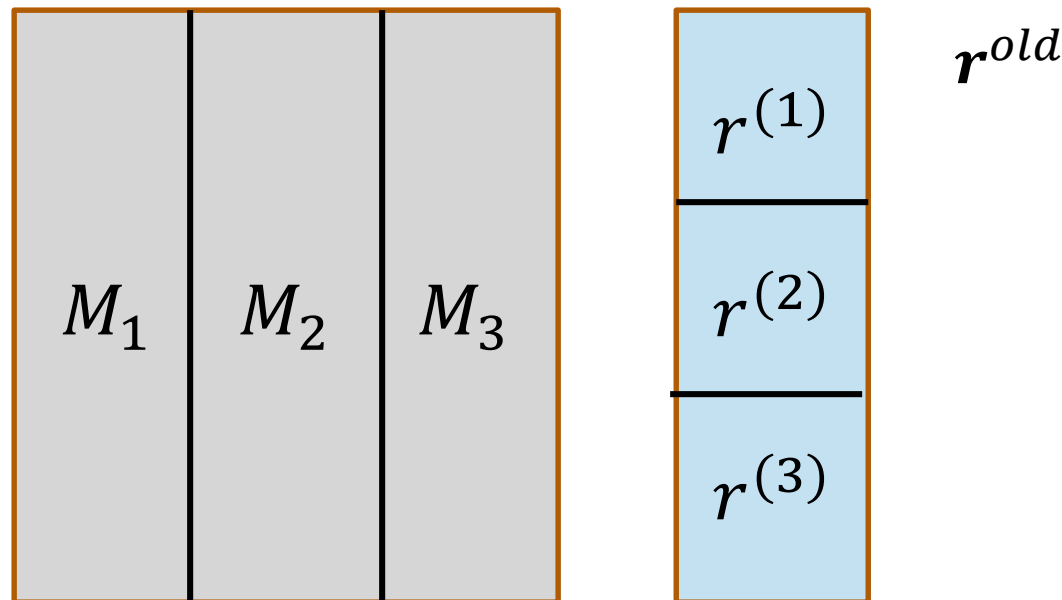
- Stochastische Adjazenzmatrix  $\mathbf{M}$  ist sehr groß und liegt verteilt vor
- Initialisierung von  $\mathbf{r}^{old}$  ( $\forall i: r_i^{old} = 0$ ) und Vektor  $\mathbf{r}^{new}$  ( $\forall i: r_i^{new} = 1$ ) im Hauptspeicher

Wiederhole:

1. Berechne  $\sum_j |r_j^{new} - r_j^{old}|$  und Abbruch, falls  $< \varepsilon$
2.  $\mathbf{r}^{old} \leftarrow \mathbf{r}^{new}$
3. **Matrix-Vektor-Produkt:**  $\mathbf{r}^{new} \leftarrow \beta \cdot \mathbf{M} \cdot \mathbf{r}^{old}$
4.  $S \leftarrow \sum_j r_j^{new}$
5.  $\forall i: r_i^{new} += 1 - \frac{S}{n}$

# Verteilte Berechnung des PageRank

- Matrix-Vektor-Produkt über **MapReduce**
- *Problem*: PageRank-Vektor  $r^{old}$  passt nicht in Hauptspeicher
  - Mind. 1 Billion ( $10^{12}$ ) Webseiten
  - Einfache/doppelte Genauigkeit für Wert: 4 Byte/8 Byte
  - 4TB/8TB Speicherbedarf für Vektor
- *Unterteilung des Vektors  $r^{old}$  in Abschnitte*



# Verteilte Berechnung des PageRank

- Jeder Map-Task ist nur für einen Streifen  $M_k$  zuständig und benötigt auch nur den dazugehörigen Teil  $r^{(k)}$  von  $r^{old}$

Wiederhole:

1. Berechne  $\sum_j |r_j^{new} - r_j^{old}|$  und Abbruch, falls  $< \varepsilon$
2.  $r^{old} \leftarrow r^{new}$  (**nur relevanter Teil  $r^{(k)}$** )

3. **Matrix-Vektor-Produkt:**  $\forall i: r_i^{new} \leftarrow \sum_{j \rightarrow i} \beta \frac{r_j^{old}}{d_j}$

- Eingabe: Elemente  $(i, j, m_{ij})$  des Streifens  $M_k$
- Map: Elemente  $(i, j, m_{ij})$  auf  $(i, m_{ij}r_j)$
- Reducer für Schlüssel  $i$  bekommt Liste  $[m_{i1}r_1, \dots, m_{in}r_n]$  und berechnet  $(i, \sum_j \beta m_{ij}r_j)$  also  $r_i^{new}$

4.  $S \leftarrow \sum_j r_j^{new}$

5.  $\forall i: r_i^{new} = r_i^{new} + \frac{1-S}{n}$

4 MapReduce-Prozeduren



# Blockmatrix

- Unterteilung von  $\mathbf{r}^{old}$  in  $k$  Blöcke der Länge  $l$
- **Zusätzliche Unterteilung der Matrix in Quadrate der Länge  $l$**
- *Vorteil:* Ein Map-Task ist nur für ein Quadrat verantwortlich und somit schneller fertig
- *Nachteile:*
  - Insg.  $k^2$  (statt nur  $k$ ) Map-Tasks
  - Ein Block von  $\mathbf{r}^{old}$  wird von insgesamt  $k$  Map-Tasks gelesen

$M_{11}$	$M_{12}$	$M_{13}$
$M_{21}$	$M_{22}$	$M_{23}$
$M_{31}$	$M_{32}$	$M_{33}$

$r_1$
$r_2$
$r_3$

$\mathbf{r}^{old}$

# Effiziente Repräsentation der Matrix $M$

- Effizientere Repräsentation der Matrix  $M$  möglich, da spärlich besetzt und nur zwei verschiedene Werte pro Spalte
- Für jede Spalte: Speichere  $d_i$  (Anzahl der Hyperlinks) und Liste der Zeilen mit einem Wert ungleich Null:  $dest_1, \dots, dest_{d_i}$
- Unterteilung von  $M$  in Spalten:

		$i$	$d_i$	$dest_1, \dots, dest_{d_i}$		
0		0	3	1, 4, 5		0
1		1	4	1,2,3,5		1
2		2	2	3,4		2
3		...	...	...		3
4						4
5						5

# Blockmatrix aus 3x3 Quadrate

		$i$	$d_i$	$dest_1, \dots, dest_{d_i}$		
$r^{new}$	0	0	3	1	$r^{old}$	0
	1	1	4	1		1
2	0	0	3		2	2
	1	1	4	2,3		3
3	0	0	3	4,5	3	4
	1	1	4	5		5
4	2	2	2		4	4
	3	3	...	...		5
5	2	2	2	3	5	4
	3	3	...	...		5
...						

# Inhaltsverzeichnis

- **Einführung**
- **PageRank**
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- **Link-Spam**

# Themenspezifischer PageRank

- PageRank misst die allgemeine Beliebtheit einer Webseite und vernachlässigt somit evtl. wichtige themenspezifische Quellen
- **Ziel:** Bewertung von Webseiten nicht nur nach Popularität, sondern auch nach Thema, z.B. Wissenschaft, Kunst, Natur, Motorsport, ...
  - Berücksichtigung des Kontexts oder der Interessen eines Nutzers bei Suchanfragen
  - Beispiel: Anfrage nach „Jaguar“ soll, je nach Interesse des Nutzers, entweder zu Webseiten mit Tieren oder mit Autos führen
- **Idee:** Höhere Gewichtung für themenspezifische Seiten
  - Menge  $S$  mit themenspezifischen Seiten
  - Einen PageRank-Vektor  $\mathbf{r}^{(S)}$  für jedes  $S$
  - Falls  $i \in S$ :  $r_i^{(S)} = \sum_{j \rightarrow i} \beta \frac{r_j}{d_j} + (1 - \beta) \frac{1}{|S|}$ ; Sonst:  $r_i^{(S)} = \sum_{j \rightarrow i} \beta \frac{r_j}{d_j}$
  - Mit Wahrscheinlichkeit  $1 - \beta$ : Wähle zufällig eine Seite aus  $S$
  - Auch Webseiten „in der Nähe“ von Elementen aus  $S$  werden höher gewichtet

# Zusammenstellen der Menge $S$

- Verwendung der Seiten eines Webverzeichnisses, z.B. [Curlie](#) (DMOZ)
- Erweiterung mit *ähnlichen* Webseiten unter Verwendung *typischer* Worte
  - Typische Wörter = Wörter, die generell selten aber relative häufig in einem Dokument eines bestimmten Themas vorkommen
  - z.B. über Tf-idf-Maß eines Wortes  $i$  in Dokument  $j$ :  $TF \cdot IDF_{ij} = TF_{ij} \cdot IDF_i$ 
    - Absolute Häufigkeit  $f_{ij}$  eines Wortes  $i$  in Dokument  $j$
    - Relative Häufigkeit des Wortes  $i$  in Dokument  $j$ :  $TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$
    - Sei  $n_i$  die Anzahl der insgesamt  $N$  Dokumente, die das Wort  $i$  enthalten
    - Inverse Dokumenthäufigkeit:  $IDF_i = \log_2 \frac{N}{n_i}$
- Auch der Kontext bzw. die Interessen eines Nutzers können über diese Methode erschlossen werden:
  - Wörter in vergangenen Anfragen
  - Wörter auf Seiten mit Bookmark
  - Wörter auf aktueller Seite (von wo Anfrage gesendet)

# Inhaltsverzeichnis

- **Einführung**
- **PageRank**
  - Probleme
  - Effiziente Berechnung
  - Themenspezifischer PageRank
- **Link-Spam**

# Web Spamming

- **Web Spamming:** alle Unternehmungen um eine Webseite im Ranking einer Suchmaschine zu verbessern **ohne** die „Bedeutung“ der Seite für eine Suchanfrage zu erhöhen
- **Term-Spamming:** Manipulation des Inhalts einer Webseite
  - Beispiel: ein Verkäufer von T-Shirts lässt seine Webseite so aussehen, als ob interessante Information zu Sport/Filmen zu finden sind
  - Einfügen von relevanten Wörtern bzw. einer vollständigen Kopie einer relevanten Webseite und ändern der Textfarbe auf Hintergrundfarbe
- Lösung von Suchmaschinen: Anstelle der Textes einer Webseite, Verwendung des Textes in und um Verlinkungen zu einer Webseite
- PageRank verhindert den Versuch diese Lösung zu umgehen, indem man tausende irrelevante Seiten mit Verlinkungen auf Zielseite erstellt
- **Link-Spamming:** Erstellen einer Struktur von Verlinkungen um PageRank zu erhöhen



# Google-Bombe (2006)



## Web

Results 1 - 10 of about 969,000 for [miserable failure](#). (0.06 seconds)

### [Biography of President George W. Bush](#)

Biography of the president from the official White House web site.

[www.whitehouse.gov/president/gwbbio.html](http://www.whitehouse.gov/president/gwbbio.html) - 29k - [Cached](#) - [Similar pages](#)

[Past Presidents](#) - [Kids Only](#) - [Current News](#) - [President](#)

[More results from www.whitehouse.gov »](#)

### [Welcome to MichaelMoore.com!](#)

Official site of the gadfly of corporations, creator of the film Roger and Me and the television show The Awful Truth. Includes mailing list, message board, ...

[www.michaelmoore.com/](http://www.michaelmoore.com/) - 35k - Sep 1, 2005 - [Cached](#) - [Similar pages](#)

### [BBC NEWS | Americas | 'Miserable failure' links to Bush](#)

Web users manipulate a popular search engine so an unflattering description leads to the president's page.

[news.bbc.co.uk/2/hi/americas/3298443.stm](http://news.bbc.co.uk/2/hi/americas/3298443.stm) - 31k - [Cached](#) - [Similar pages](#)

### [Google's \(and Inktomi's\) Miserable Failure](#)

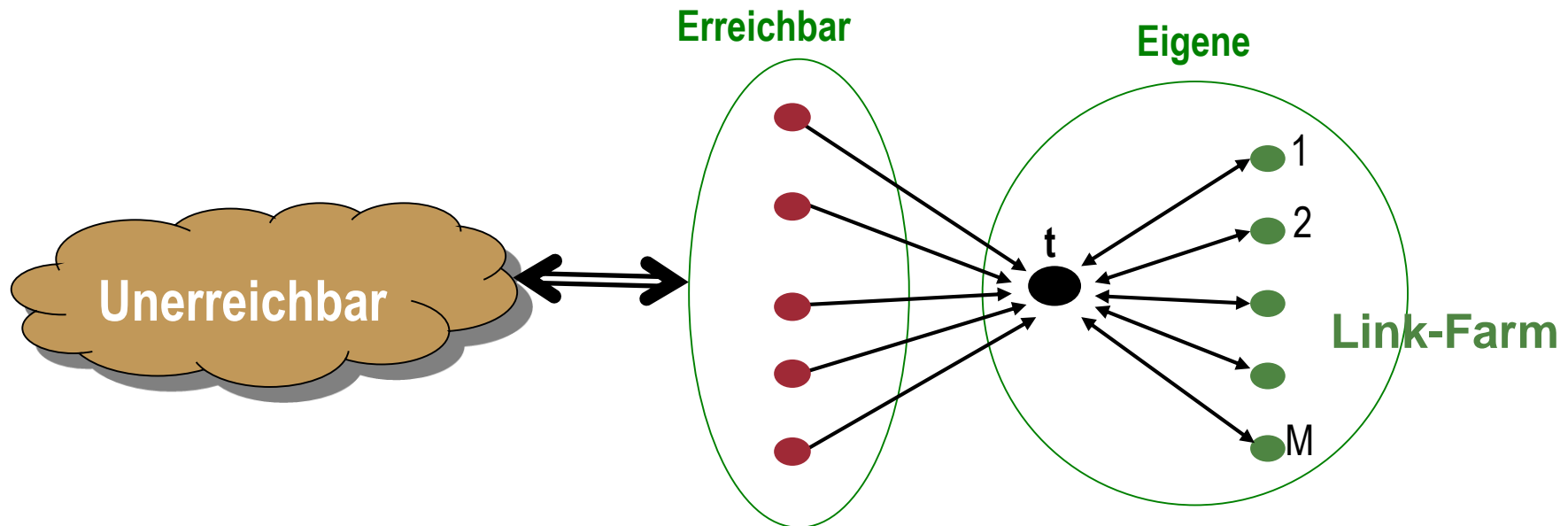
A search for **miserable failure** on Google brings up the official George W.

Bush biography from the US White House web site. Dismissed by Google as not a ...

[searchenginewatch.com/sereport/article.php/3296101](http://searchenginewatch.com/sereport/article.php/3296101) - 45k - Sep 1, 2005 - [Cached](#) - [Similar pages](#)

# Link-Spamming

- Aus Sicht des Spammer: 3 Arten von Webseiten
  - Unerreichbare Seiten
  - Erreichbare/editierbare Seiten (z.B. über Kommentare auf Blogs/Nachrichtenseiten)
  - Eigene Seiten
- Vorgehen:
  - Möglichst viele Verlinkungen von erreichbaren Seiten zur Zielseite  $t$
  - Erstellen einer **Link-Farm** um PageRank weiter zu erhöhen



# Funktionsweise der Link-Farm

- PageRank-Gewicht aus den erreichbaren Seiten:  $x$
- PageRank der Zielseite  $t$ :  $y$
- PageRank einer der insgesamt  $M$  Farm-Seiten:  $\frac{\beta y}{M} + 1 - \beta$
- Dann:

$$y = x + \beta M \left[ \frac{\beta y}{M} + 1 - \beta \right] + 1 - \beta = x + \beta^2 y + \beta(1 - \beta)M + 1 - \beta$$

$$\Rightarrow y = \boxed{\frac{x}{1 - \beta^2}} + \boxed{\frac{\beta}{1 + \beta} M} + \boxed{\frac{1}{(1 + \beta)}}$$

Vervielfachung  
von  $x$

Beliebig großes  $y$   
durch großes  $M$

Vernachlässigbar

# TrustRank

- Google erkennt das klassische Design einer Link-Farm und wird die Webseiten aus dem Index entfernen
  - Außerdem versucht Google neue Designs zu erkennen
  - *Krieg* zwischen Google und Spammer
- Alternative: Verwendung des **TrustRank**: Themenspezifischer PageRank mit einer Auswahl vertrauenswürdiger Seiten als Menge  $S$ 
  - **Annahme**: Vertrauenswürdige Seiten verlinken selten auf Spam
  - Vertrauen in beliebige Seite nimmt ab mit Entfernung zu vertrauenswürdiger Seite
  - Vertrauen wird über alle Verlinkungen aufgeteilt da viele Verlinkungen oft mit einer ungenaueren Prüfung dieser einhergeht
- Auswahl der vertrauenswürdigen Seiten (ohne menschliche Inspektion):
  - Seiten mit sehr hohem PageRank
  - Verwendung vertrauenswürdiger Domains, z.B. .edu, .gov, ...