

Ranking:

PageRank als Potenzreihe

Die Wege

Seien u und v zwei Knoten und $l \in \mathbb{N}$. Wir definieren

$$a_0(u, v) = \begin{cases} 0 & \text{falls } u \neq v \\ 1 & \text{falls } u = v \end{cases}$$

$$a_l(u, v) = \sum_{w|(w,v) \in E} a_{l-1}(u, w) \frac{1}{\text{out}(w)}$$

Die $a_l(u, v)$ sind die Wahrscheinlichkeit, dass ein Zufalls Surfer (ohne Teleportation) in l Schritten in v ist, sofern er in u begonnen hat, d.h.

$$\begin{aligned} a_l(u, v) &= P(X_l = v \mid X_0 = u) \\ &= \sum_w P(X_{l-1} = w \mid X_0 = u) P(X_l = v \mid X_{l-1} = w) \\ &= \sum_{w|(w,v) \in E} a_{l-1}(u, w) \frac{1}{\text{out}(w)}. \end{aligned}$$

Die Wege

Alternativ ergibt sich:

$$a_0(u, v) = \begin{cases} 0 & \text{falls } u \neq v \\ 1 & \text{falls } u = v \end{cases}$$
$$a_l(u, v) = \frac{1}{\text{out}(u)} \sum_{w|(u,w) \in e} a_{l-1}(w, v)$$

Die Potenzreihe

Satz

Für jedes $v \in V$ ist

$$\text{PageRank}(v) = \sum_{l \in \mathbb{N}} (1-d)d^l \sum_{u \in V} a_l(u, v) e(u) \quad (1)$$

ist die eindeutig bestimmte Lösung der Gleichung

$$\text{PageRank}(v) = d \sum_{w \rightarrow v} \frac{\text{PageRank}(w)}{\text{out}(w)} + (1-d)e(v). \quad (2)$$

Beweis:

Dass (1) eine Lösung ist sieht man durch Einsetzen unter Anwendung der Definition der a_l (Übungsaufgabe).

Dass die Potenzreihe die eindeutig bestimmte Lösung ist, ist etwas schwieriger.

Eindeutigkeit der Lösung

Wir nehmen an, dass die Gleichung (2) zwei verschiedene Lösungen r_1 und r_2 hat.

Damit gilt:

$$r_1(v) - r_2(v) = d \sum_{w \rightarrow v} \frac{r_1(w) - r_2(w)}{\text{out}(w)}$$

und somit in Matrixschreibweise:

$$r_1 - r_2 = dM^T(r_1 - r_2),$$

wobei M die **normalisierte Adjazenzmatrix** ist.

Zur Erinnerung:

Die Spaltensummen von M^T sind entweder 1 oder 0 (Senken).

Eindeutigkeit der Lösung

Summiert man die Gleichung spaltenweise, so ergibt sich:

$$\begin{aligned} \sum_{v \in V} |r_1(v) - r_2(v)| &= d \sum_{v \in V} s_v |r_1(v) - r_2(v)| \\ &\leq d \sum_{v \in V} |r_1(v) - r_2(v)| \end{aligned}$$

wobei $s_v \geq 0$ die Summe der v zugeordneten Spalte ist.

Damit gilt $1 \leq d$ oder $r_1 = r_2$.

Für einen Dämpfungsfaktor $0 < d < 1$ ist die Lösung von (2) somit eindeutig bestimmt.



Der Zufalls-Surfer

$$\text{PageRank}(v) = \sum_{l \in \mathbb{N}} (1-d)d^l \sum_{u \in V} a_l(u, v) e(u)$$

ergibt eine andere Möglichkeit um $\text{PageRank}(v)$ zu interpretieren.

- Zu jedem Zeitpunkt t ist der Surfer auf einer Seite X_t .
- Er wählt die Startseite $X_0 = v$ mit Wahrscheinlichkeit $e(v)$ unter allen Seiten.
- In jedem Schritt entscheidet der Surfer mit Wahrscheinlichkeit $(1-d)$, ob er aufhört.
- Setzt er seinen Weg fort (Wahrscheinlichkeit d), so wählt er unter allen ausgehenden Links der aktuellen Seite einen zufällig aus und folgt ihm.
- Möchte er seinen Weg fortsetzen, befindet sich aber in einer Senke, so ist er *beleidigt* und hört gezwungenermaßen auf.

Der Zufalls-Surfer

$$\text{PageRank}(v) = \sum_{l \in \mathbb{N}} (1-d)d^l \sum_{u \in V} a_l(u, v) e(u)$$

- Der Surfer begann in u .
- Der Surfer machte l Schritte und hörte **freiwillig** auf.
- Der Surfer beendete nach l Schritten, **freiwillig** seinen Lauf in v .

Es **fehlt** die Wahrscheinlichkeit, dass der Surfer seinen Lauf gezwungenermaßen beendete, d.h. das er sich in einer Senke entschied weiter zu machen.

⇒ Senken verursachen einen Rang-Verlust.

Der Zufalls-Surfer

Satz

$$\sum_{v \in V} \text{PageRank}(v) = \|e\|_1 - \frac{d}{1-d} \sum_{v | \text{out}(v)=0} \text{PageRank}(v)$$

Der Verlust entspricht genau der Wahrscheinlichkeit, dass der Zufalls-Surfer seinen Lauf unfreiwillig beenden muss.

Die Teleportation

Obwohl der zweite Ansatz ohne Teleportation auskommt, sind die Ergebnisse äquivalent:

Satz

PageRank sei der Rang ohne Teleportation und $\text{PageRank}'$ der mit Teleportation. Für jeden Knoten v gilt

$$\text{PageRank}'(v) = \frac{\text{PageRank}(v)}{\|\text{PageRank}\|_1}.$$

D.h. beide Rankings unterscheiden sich nur durch die Normierung.

Die Berechnung

Zur Berechnung benutzen wir die folgende Iteration:

$$r^{(i+1)} = dM^t r^{(i)} + (1-d)e$$

für ein beliebiges $r^{(0)}$.

Unter direkter Benutzung des Links ergibt sich

$$r^{(i+1)}(v) = d \sum_{u|u \rightarrow v} \frac{r^{(i)}(u)}{\text{out}(u)} + (1-d)e(v)$$

Dies ergibt den folgenden Algorithmus.

Die Berechnung

Eingabe: Ein Graph $G = (V, E)$, ein V -Vektor $r^{(0)}$ (Startvektor),
eine Zahl $0 < d < 1$ und ein Personalisierungsvektor e

Ausgabe: Eine Approximation $r^{(i)}$ von PageRank

solange „nicht konvergiert“ **tue**

für alle $v \in V$ **tue**

$r^{(i+1)}(v) = (1 - d)e(v)$

für u mit $u \rightarrow v$ **tue**

$r^{(i+1)}(v) = r^{(i+1)}(v) + d \frac{r^{(i)}(u)}{\text{out}(u)}$

Ende

Ende

Ende

Die Parameter

Durch die Beschreibung als Potenzreihe, konnten die verschiedenen Parameter für PageRank sauber getrennt werden.

- Die Graphstruktur geht über die **Wegwahrscheinlichkeiten** $a_l(u, v)$ ein.
- Der **Dämpfungsfaktor** d beeinflusst die Reichweite der gegenseitigen Beeinflussung.
- Der **Personalisierungsvektor** e beschreibt eine ad-Hoc-Relevanz jeder Seite.

Welche Effekte haben die einzelnen Parameter?

Ranking:

PageRank: Dämpfung

Die Konvergenz

Zur Beurteilung der Güte der Approximation, betrachten wir die Konvergenz.

Satz

Für jedes $i \geq 1$ und jeden Startvektor $r^{(0)}$ gilt:

$$r^{(i)}(v) = d^i \sum_u a_i(u, v) r^{(0)}(u) + \text{PageRank}^{(i-1)}(v),$$

wobei

$$\text{PageRank}^{(i)}(v) := \sum_{l=0}^i (1-d)d^l \sum_u a_l(u, v) e(u)$$

die i -te **Partialsomme** von $\text{PageRank}(v)$ ist.

Beweis: Übungsaufgabe

Die Konvergenz

Korollar

Für $r^{(0)} = (1 - d)e$ gilt für jeden Knoten v

$$r^{(i)}(v) = \text{PageRank}^{(i)}(v).$$

Weiter gilt

$$\|\text{PageRank} - r^{(i+1)}\|_1 \leq d \|\text{PageRank} - r^{(i)}\|_1.$$

Beweis: Übungsaufgabe

Konsequenz: Die Dämpfung garantiert die Konvergenz.

Gleichzeitig kann man den absoluten Fehler sehr gut abschätzen.

Die Konvergenz

Für $r^{(0)} = (1 - d)e$ gilt offensichtlich

$$\begin{aligned} \|\text{PageRank} - r^{(i)}\|_1 &\leq d^i \|\text{PageRank} - r^{(0)}\|_1 \\ &\leq d^i (\|\text{PageRank}\|_1 + \|r^{(0)}\|_1) \\ &\leq 2(1 - d)d^i \end{aligned}$$

Mittels

$$\|\text{PageRank} - r^{(i)}\|_1 \leq 2(1 - d)d^i \leq \varepsilon$$

können wir somit garantieren, dass der absolute Fehler ε nicht überschreitet.

Es ergibt sich somit

$$2(1 - d)d^i < \varepsilon \quad \Leftrightarrow \quad i > \frac{\ln \varepsilon - 1 - \ln(1 - d)}{\ln d}$$

Für einen absoluten Fehler $\varepsilon = 0.00001$ und $d = 0.85$ ergeben sich so ca. **66 Iterationen**.

Die Gauß-Seidel-Iteration

Die Konvergenz lässt sich sogar verbessern, indem wir Speicher sparen!

In dem oben angegebenen Algorithmus müssen immer zwei Rankings pro Knoten gespeichert werden, nämlich $r^{(i)}(v)$ und $r^{(i+1)}(v)$.

Diese Vorgehensweise heißt **Jacobi-Iteration**.

Verwendet man stattdessen nur ein gespeichertes Ranking, so spricht man von einer **Gauß-Seidel-Iteration**.

Leider konvergiert diese nicht immer. Aber in unserem Fall tut sie es, und die resultierenden Werte sind auf jeden Fall nicht schlechter als vorher.

Die Gauß-Seidel-Iteration

Eingabe: Ein Graph $G = (V, E)$, ein V -Vektor $r^{(0)}$ (Startvektor)
eine Zahl $0 < d < 1$ und ein Personalisierungsvektor e

Ausgabe: Eine Approximation r von PageRank

für alle $v \in V$ **tue** $r(v) = r^{(0)}(v)$

solange nicht konvergiert tue

für alle $v \in V$ **tue**

$r(v) = (1 - d)e(v)$

für u mit $u \rightarrow v$ **tue**

$r(v) = r(v) + d \frac{r(u)}{\text{out}(u)}$

Ende

Ende

Ende

Die Gauß-Seidel-Iteration

Satz

$r^{(i)}$ sei der durch i Jacobi-Iterationen und $s^{(i)}$ der durch i Gauß-Seidel-Iterationen gewonnene Vektor. Es gilt

$$\lim_{i \rightarrow \infty} s^{(i)} = \lim_{i \rightarrow \infty} r^{(i)} = \text{PageRank}$$

und

$$\|\text{PageRank} - s^{(i)}\|_1 \leq \|\text{PageRank} - r^{(i)}\|_1.$$

Ranking:

PageRank: Personalisierung

PageRank als lineare Abbildung

Es gilt

$$\text{PageRank}(v) = \sum_u \sum_{l=0}^{\infty} d^l (1-d) a_l(u, v) e(u) = \sum_u a(u, v) e(u),$$

wobei

$$a(u, v) := \sum_{l=0}^{\infty} d^l (1-d) a_l(u, v).$$

Damit ergibt sich

$$\text{PageRank} = A^T e$$

mit $A^T = (a(u, v))_{u, v \in V}$.

Konsequenz

PageRank ist eine lineare Abbildung, die einen Personalisierungsvektor auf einen Rankingvektor abbildet.

Personalisierung

Perfekte Personalisierung:

- Der Nutzer weist **jeder** Seite eine Basis-Relevanz $e(v)$ zu.
- Über die PageRank Matrix A lässt sich dann ein individuelles Ranking $A^T e$ erstellen.
- **Probleme:**
 - ▶ A muss komplett bekannt sein.
 - ▶ $A^T e$ muss berechnet werden; Zeit $O(n^2)$
 - ▶ Der Nutzer muss e erstellen.

Deshalb ermöglicht man keine perfekte Personalisierung, sondern schränkt die Auswahlmöglichkeiten ein.

Modulare Personalisierung

Grundsätzliche Idee:

- Lasse nicht jeden Vektor e zu.
- Treffe eine Vorauswahl von k (linear unabhängigen) Personalisierungsvektoren e_1, \dots, e_k .
- Gebe dem Nutzer nur die Möglichkeit eine Linearkombination aus den k Vektoren zu wählen:

$$e = x_1 e_1 + \dots + x_k e_k \text{ mit } x_i \in [0, 1]$$

- In diesem Fall ergibt sich das Ranking als:

$$r = Ae = \sum_{i=1}^k x_i Ae_i.$$

Modulare Personalisierung

Konsequenz: Statt eines Rankings pro Seite, müssen k Rankings

$$r_i = Ae_j \text{ mit } 1 \leq i \leq k$$

berechnet werden.

Dies kann einfach durch gleichzeitige Berechnung erreicht werden.

Frage: Wie werden die Basisrankingvektoren gewählt?

Modulare Personalisierung

Mögliche Gruppenbildung:

- Themenbasierte Kategorien, z.B.
 - ▶ Wissenschaften (Informatik, Mathematik, Physik etc.)
 - ▶ Politik
 - ▶ Sport
 - ▶ Hobbies
- Vertrauensvolle Seiten, z.B.
 - ▶ Institutionen (Universitäten, Institute, Regierungen etc.)
 - ▶ Firmen einer Branche
 - ▶ Medien

Der Nutzer gibt den Kategorien eine Wertung und erhält so ein personalisiertes Ranking.

Zusätzlich sollte immer das undifferenzierte Ranking (d.h. alle gleich) mit einfließen, d.h. als $(k + 1)$ -te Kategorie verwendet werden.

Ranking:

PageRank: Graphstruktur

Die Wege Nachdem der Dämpfungsfaktor und der Personalisierungsvektor betrachtet wurden, bleibt die Graphstruktur.

Die Graphstruktur geht über die **Wegwahrscheinlichkeiten** $a_l(u, v)$ ein.

Beobachtung: Das Ranking eines Knotens v wird nur dann von dem Ranking eines Knotens u beeinflusst, wenn ein Weg von u nach v existiert.

Konsequenz: Wir können PageRank komponentenweise berechnen.

PageRank und die starken Komponenten

Satz

Sei $G = (V, E)$ ein gerichteter Multigraph und C eine seiner starken Komponenten. Dann konvergiert für jedes $v \in C$ die Folge $r^{(i)}(v)$ mit $r^{(0)}(v) = (1 - d)e(v)$ und

$$r^{(i+1)}(v) = d \sum_{u \in C | u \rightarrow v} \frac{r^{(i)}(u)}{\text{out}(u)} + d \sum_{u \notin C | u \rightarrow v} \frac{\text{PageRank}(u)}{\text{out}(u)} + (1 - d)e(v)$$

gegen $\text{PageRank}(v)$.

Beweis:

Zuerst stellen wir fest, dass die Partialsummen $\text{PageRank}^{(i)}(v)$ eine monoton steigenden Folge bilden, da alle Summanden ≥ 0 sind.

Wir beweisen nun induktiv, dass für jedes $i \geq 0$ Folgendes gilt:

$$\text{PageRank}(v) \geq r^{(i)}(v) \geq \text{PageRank}^{(i)}(v).$$

PageRank und die starken Komponenten

Induktionsanfang: Für $i = 0$ gilt offensichtlich:

$$r^{(0)}(v) = \text{PageRank}^{(0)}(v).$$

Induktionsannahme: Für alle v gilt

$$\text{PageRank}(v) \geq r^{(i)}(v) \geq \text{PageRank}^{(i)}(v).$$

Induktionsschritt: Es gilt

$$\begin{aligned} \text{PageRank}(v) &= d \sum_{u \in C|u \rightarrow v} \frac{\text{PageRank}(u)}{\text{out}(u)} + d \sum_{u \notin C|u \rightarrow v} \frac{\text{PageRank}(u)}{\text{out}(u)} + (1-d)e(v) \\ &\geq d \sum_{u \in C|u \rightarrow v} \frac{r^{(i)}(u)}{\text{out}(u)} + d \sum_{u \notin C|u \rightarrow v} \frac{\text{PageRank}(u)}{\text{out}(u)} + (1-d)e(v) \\ &= r^{(i+1)}(v) \\ &\geq d \sum_{u \in C|u \rightarrow v} \frac{\text{PageRank}^{(i)}(u)}{\text{out}(u)} + d \sum_{u \notin C|u \rightarrow v} \frac{\text{PageRank}^{(i)}(u)}{\text{out}(u)} + (1-d)e(v) \\ &= \text{PageRank}^{(i+1)}(v) \quad \square \end{aligned}$$

PageRank und die starken Komponenten

Das das Ergebnis auch für Approximationen gilt, ergibt sich aus

Satz

Sei $G = (V, E)$ ein gerichteter Multigraph und C eine seiner starken Komponenten und $j \in \mathbb{N}$. Für jeden Knoten u , der direkt zu einem Knoten v in C verlinkt ist sei ferner ein Wert $r(u)$ gegeben mit

$$\text{PageRank}(u) \geq r(u) \geq \text{PageRank}^{(j)}(u).$$

Dann gilt für jedes $v \in C$, $r^{(0)}(v) = (1 - d)e(v)$ und

$$r^{(i+1)}(v) = d \sum_{u \in C | u \rightarrow v} \frac{r^{(i)}(u)}{\text{out}(u)} + d \sum_{u \notin C | u \rightarrow v} \frac{r(u)}{\text{out}(u)} + (1 - d)e(v)$$

auch

$$\text{PageRank}(v) \geq r^{(j)}(v) \geq \text{PageRank}^{(j)}(v).$$

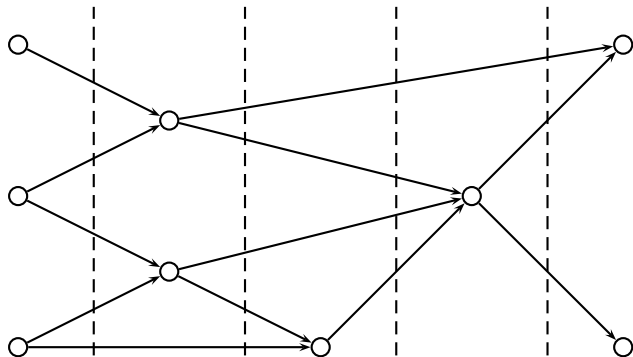
Konsequenz: Wir können komponentenweise iterieren und das Ergebnis ist nicht schlechter als vorher.

Der DAG der starken Komponenten

Die starken Komponenten von G bilden die Knoten eines DAG $S(G)$ (directed acyclic graph).

- Die Knoten von $S(G)$ sind die starken Komponenten $[v]$
- Für jede Kante (u, v) in G mit $[u] \neq [v]$ existiert eine Kante $([u], [v])$ in $S(G)$.
- Dieser Graph enthält keine Kreise.
- Die Knoten dieses Graphen, d.h. die starken Komponenten von G lassen sich in Schichten L_1, \dots, L_k einteilen, d.h.
 - ▶ für jedes $[v]$ existiert eine Zahl $1 \leq l[v] \leq k$, so dass $[v] \in L_{l[v]}$ und
 - ▶ für jede Kante $([u], [v])$ gilt $l[u] < l[v]$.
- Die Schichten können in linearer Zeit ermittelt werden.

Ein Beispiel



Die verteilte Berechnung von PageRank

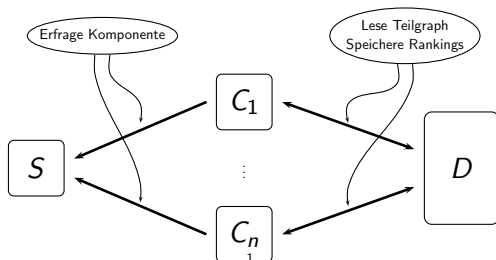
Konsequenz: Sind die Rankings von allen Komponenten in L_1, \dots, L_i bekannt, dann können die Rankings aller Komponenten in L_{i+1} unabhängig voneinander berechnet werden (d.h. gleichzeitig)

Das ermöglicht eine Client / Server basierte, verteilte Berechnung von PageRank.

Das System besteht aus

- einer **Datenbank D** , die den Graphen, die starken Komponenten und die Rankings speichert,
- einem **Server S** , der $S(G)$ kennt und eine Warteschlange von starken Komponenten verwaltet und
- **Klienten C_1, \dots, C_n** , die die komponentenweise Berechnung durchführen.

Die verteilte Berechnung von PageRank



- Zu Beginn initialisiert S für jede starke Komponente einen Zähler auf ihren Eingangsgrad in $S(G)$.
- Alle Komponenten ohne Vorgänger werden in die Warteschlange eingefügt.
- Ein Klient C_i erfragt bei S die ID einer Komponente aus der Warteschlange.
- C_i erhält den Teilgraphen und die notwendigen Ränge von D und führt lokal die Iteration durch.
- Anschließend sendet C_i zu D und meldet S , dass die Komponente fertig berechnet wurde.
- S senkt die Zähler der starken Komponenten auf die die abgearbeitete Komponente verwies.
- Sinkt der Zähler einer Komponente auf 0, wird sie in die Warteschlange eingereiht.
- Wenn die Warteschlange leer ist, signalisiert S das Ende der Berechnung.

Die verteilte Berechnung von PageRank

Zur Erinnerung: WebBase Crawl von 2001

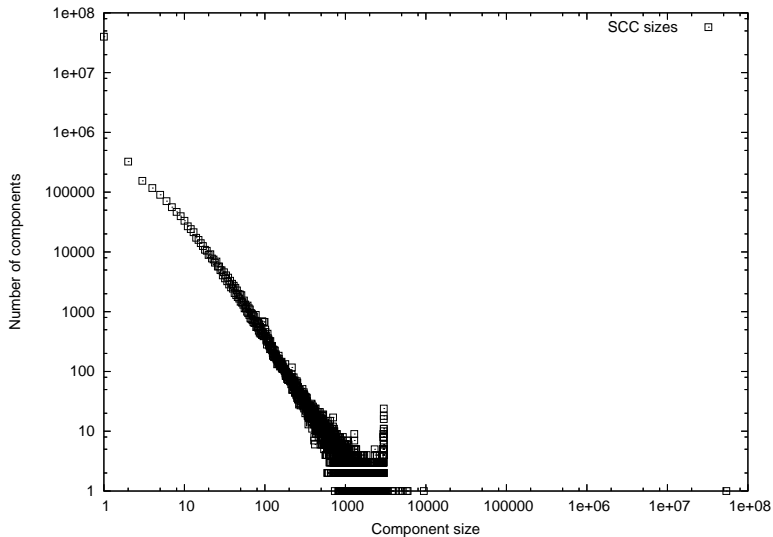
Anzahl der Knoten	118 142 115
Anzahl der Kanten	1 019 903 190
Anzahl der starken Komponenten	41 126 852
Durchschnittliche Größe	~ 2,8726
Größte starke Komponente:	53 891 939 Knoten
Zweitgrößte starke Komponente	9 428 Knoten
Drittgrößte starke Komponente	5 925 Knoten
Starke Komponenten der Größe 1	39 843 421
Starke Komponenten der Größe 2	323 994
Starke Komponenten der Größe 3	154 786

Die verteilte Berechnung von PageRank

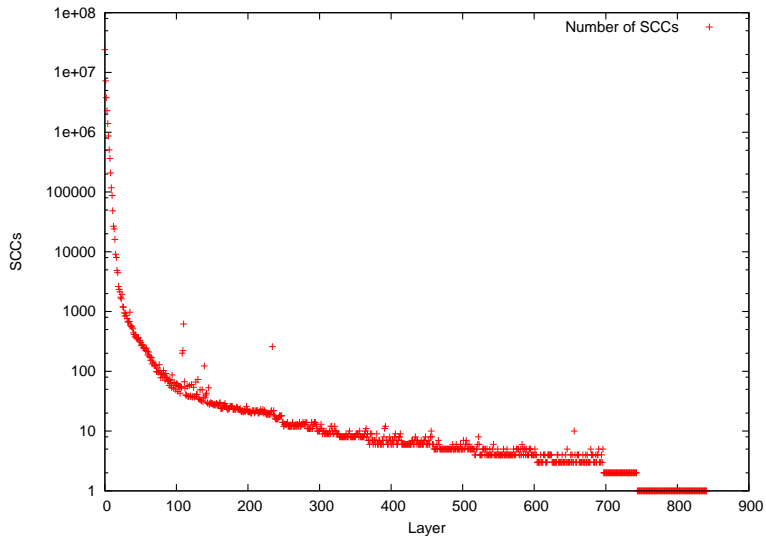
Konsequenzen:

- Das grösste zu behandelnde Teilproblem entspricht ca. 45% des gesamten Problemes.
- Alle anderen Komponenten können ohne Schwierigkeiten bearbeitet werden (einfacher PC reicht).
- Viele kleine Komponenten lassen sich ohne Iteration behandeln (direkte Lösung des Gleichungssystemes).

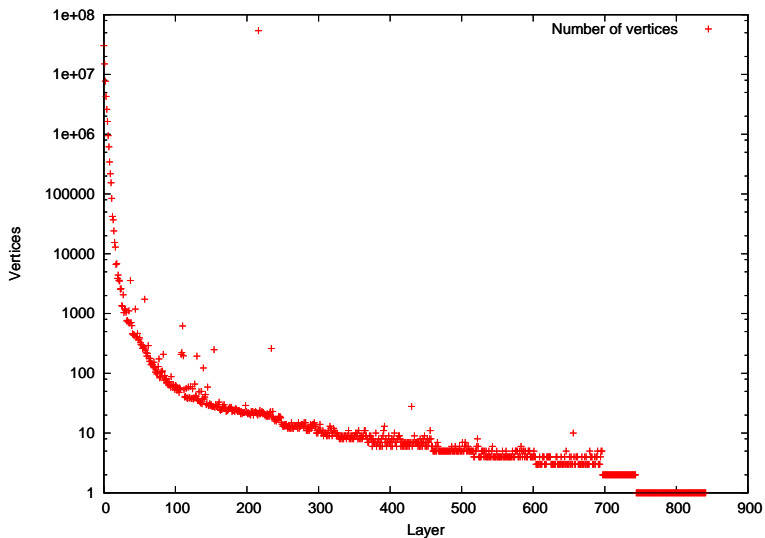
Die Starken Komponenten des WWW



Die Schichten des WWW



Die Schichten des WWW



Simulation der verteilten Berechnung

Die verteilte Berechnung von PageRank wurde simuliert, um Schätzungen für den erreichbaren Speedup zu erzielen.

- Wir nehmen an, dass die Kommunikation mit Server und Datenbank und die eigentliche Berechnung linear in der Anzahl der Knoten des jeweiligen Teilgraphen ist.
- Der Server benutzt eine FIFO Warteschlange.

Eine untere Schranke erhält man über die maximale Anzahl der Knoten auf einem Pfad durch den DAG von einer Quelle zu einer Senke.

Aus den verwendeten Daten ergab sich diese Schranke als **53 903 795 vertices**. (ca. 45 % des gesamten Graphen, größte SCC 53 891 939 Knoten).

Simulation der verteilten Berechnung

