PageRank-Algorithmus

Benedikt Wolters
Matr.-Nr. 300037

31. Januar 2012

Proseminar Algorithms and Data Structures WS 2011/12

Lehrstuhl für Informatik 2, RWTH Aachen

Betreuer: Dipl.-Inform. Haidi Yue

1 Einleitung

Bedingt durch das enorme Angebot des Webs und dessen stetiges Wachstum wird es für den Benutzer immer wichtiger, dass Suchresultate anhand ihrer Relevanz und Wichtigkeit dargestellt werden.
Dazu bedarf es eines Systems, Webseiten entsprechend zu gewichten. Die Beurteilung der Wichtigkeit einer Webseite im Internet unterliegt vielen Gesichtspunkten. Diese sind zum Teil nicht objektiver Natur, da sie individuell bestimmt werden: Sie sind abhängig von Interessen, Wissensstand, Gesinnung und Erfahrungen des Suchenden. Darüber hinaus gibt es jedoch feste Kriterien, um allgemeine Aussagen über die Gewichtung einer Webseite zu ermitteln.
Diese Arbeit setzt sich mit dem PageRank-Algorithmus [4] auseinander, einem objektiven Verfahren, das die Wichtigkeit einer Webseite anhand von Linkstrukturen im Web bestimmt – als Spiegel und Abbild der Popularität und des allgemeinen Interesses an dieser Seite.
Das drastische Wachstum des Webs stellt eine der größten Herausforderungen an Web-Suchmaschinen dar. Während 1994 der World Wide Web Worm – eine der ersten Suchmaschinen – ca. 110.000 Dokumente indiziert und katalogisiert hatte, wurde 2008 laut Google eine Indexgröße von über 1 Billion erreicht [2]. Parallel zum Wachstum des Webs nahm auch die Zahl der Anfragen an Suchmaschinen enorm zu. Bereits 1997 erhielt die Suchmaschine Altavista 20 Millionen Anfragen am Tag [4].
Durch dieses schnelle Wachstum ist Skalierbarkeit hinsichtlich gleicher Qualität der Suchergebnisse eine der größten Anforderungen an eine moderne Suchmaschine. Darüber hinaus entwickelte sich das Web von einem primär akademischen Netzwerk zu einer kommerziellen Marketingplattform – ökonomische Kriterien und Marketingstrategien bestimmten nun das Interesse, das eigene Angebot in den Top-10-Suchergebnissen erscheinen zu lassen. Entsprechend wuchs der Drang, Suchergebnisse durch Tricks zum eigenen Vorteil zu manipulieren. Das Resultat solcher Tricks war eine Fülle von Junk-Ergebnissen, deren vorrangige Auflistung man zu vermeiden suchte. Daher wurde – zur Ermittlung der Wichtigkeit einer Website – die Idee der Hypertext-Matching-Analyse [3] mit dem PageRank-Indikator verbunden. Das Ergebnis war die – zunächst als reines Forschungsprojekt ins Leben gerufene – Webseite Google. Der enorme Erfolg dieser Suchmaschine sowie die geschickte Vermarktung der Suchergebnisse mit Werbung machten Google nicht nur zum Suchmaschinen-Marktführer, sondern auch zu einem der führenden IT-Unternehmen auf der Welt.
Will man Webseiten durchsuchen, liegt es nahe, die Anzahl der Suchwörter und ihre Position als Index zur Sortierung der Suchresultate zu wählen. Dies lässt sich jedoch leicht durch Mehrfachnennung entsprechender Schlüsselwörter manipulieren. Zudem sind naive Suchverfahren nicht besonders effizient, da für jedes Suchwort zunächst die Häufigkeit seiner Nennung berechnet

werden muss.
Ein Zwischenspeichern für jedes mögliche Suchwort erfordert entsprechend hohe Speicherkapazität und skaliert somit schlecht. Selbst wenn diese Probleme der Skalierbarkeit in den Griff zu bekommen sind, lässt sich dennoch keine Aussage über die Gewichtung der Webseite im Gesamtkontext des Webs machen. Insbesondere besteht die Gefahr der Manipulation durch gezielte Mehrfachnennung von Schlüsselworten. Dementsprechend suchen wir also Gewichtungskriterien, die möglichst unabhängig vom eigentlichen Inhalt einer Webseite sind.
Diese Arbeit geht in Abschnitt 2 zunächst auf die Kodierung des Webs als Graphs und dessen speziellen Eigenschaften ein. Darauf aufbauend wird der PageRank in Abschnitt 3 zunächst formal definiert und so modelliert, dass im Anschluss verschiedene Berechnungsverfahren diskutiert werden können. Im vorletzten Abschnitt wird auf Optimierungsvarianten dieser Berechnungsverfahren eingegangen.

2 Das Web als Graph

2.1 Struktur des Web Graphen

Um das Modell des PageRanks zu formalisieren, bedarf es zunächst einiger Grundannahmen: Wir modellieren das Web als gerichteten Graphen G von HTML-Seiten, die sich untereinander verlinken. Jede Seite ist ein Knoten k dieses Graphen. Die Hyperlinks auf einer Seite sind gerichtete Kanten und zeigen auf andere Webseiten. Wir definieren den Eingangsgrad (in-degree, $in(k)$) eines Knotens, welcher die Anzahl der eingehenden Links (Kanten) darstellt. Analog definieren wir den Ausgangsgrad (out-degree, $out(k)$) für ausgehende Links.

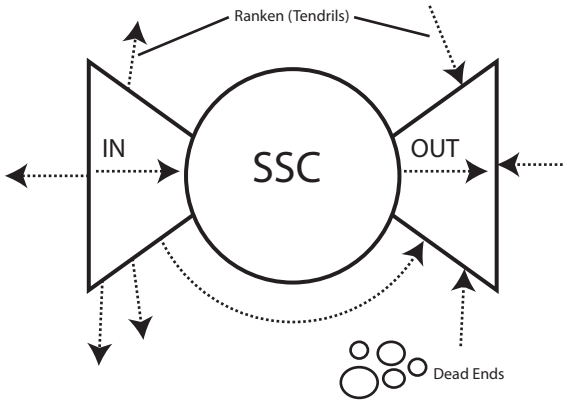

Abbildung 1:
Makroskopische Struktur des Webs nach [5]

Da das Web sehr groß ist, hat es eine komplexe Struktur. Die Struktur des

Web Graphen wird in [5] und [15] behandelt. Das Web enthält so gesehen vier Bestandteile (siehe Abbildung1): einen zentralen Kern aller Seiten, die sich gegenseitig verlinken und so eine große starke Zusammenhangskomponente (SSC) bilden. Dies ist das Herz des Webs. Wir nennen die weiteren Bestandteile IN und OUT. IN beinhaltet Seiten, die auf die SSC verlinken, die jedoch nicht aus der SSC verlinkt sind. OUT beinhaltet Seiten, die von der SSC erreichbar sind, jedoch nicht zu dieser zurückführen. Diese bilden den vierten Bestandteil, die sogenannten getrennten Komponenten (DEAD ENDS). Darüber hinaus existieren sogenannte TENDRILS, die von der SSC isoliert sind und diese nicht verlinken. Allerdings sind Verlinkungen zwischen IN, OUT und TENDRILS prinzipiell möglich.

2.2 In- und Out-Degree-Verteilungen

Der Web Graph ist ein skalenfreies Netz, das heißt: Eingangsgrad und Ausgangsgrad der Knoten sind nach den stochastischen Potenzgesetzen verteilt. Untersuchungen in [5] zeigen, dass die Eingangs- und Ausgangsgrade der jeweiligen Knoten potenzverteilt sind:

$$P_{in}(k)\,,\, P_{out}(k) \sim k^{-\alpha},\, 2 < \alpha < 3 \tag{1}$$

In anderen Worten: Die Wahrscheinlichkeit, dass ein Knoten einen Eingangsgrad bzw. Ausgangsgrad k hat, ist potenzverteilt. Im Fall des Eingangsgrads liegt α etwa bei 2,1 und beim Ausgangsgrad etwa bei 2,72 [5, 13, 15]. Dies ist insbesondere deshalb erwähnenswert, da es zeigt, dass zwischen den einzelnen Knoten enorme Unterschiede im Verlinkungsgrad existieren. Es zeigt ebenfalls, dass Webseiten, gemessen an ihrer Wichtigkeit, die durch Eingangslinks charakterisiert wird, unterschiedlich wichtig sind.

3 Der PageRank

Um die relative Wichtigkeit einer Webseite zu modellieren, benutzen wir den PageRank. Die Grundidee des PageRanks besteht darin, dass Webseiten gegenseitig über ihre Wichtigkeit "abstimmen". Dieser Abstimmvorgang geschieht über die gegenseitige Verlinkung bzw. das sich gegenseitige Zitieren. Die Gewichtung einer Webseite bestimmt sich im Rahmen des PageRank-Konzepts also aus der Wichtigkeit der Webseiten, die auf diese verlinken. Deren Rang bestimmt sich ebenfalls aus dem Rang der auf sie verlinkenden Webseiten. Dies resultiert in der Problemstellung der rekursiven Berechnung der Wichtigkeit einer Webseite aus der Bedeutsamkeit anderer Webseiten, die auf diese verweisen.

3.1 Definition

Sei u eine Webseite. Für u sei der PageRank $P(u)$ einer Seite u wie folgt definiert:

$$\tilde{P}(u) = \frac{(1-d)}{N} + d \cdot \sum_{i \in L(u)} \frac{P(i)}{N_i} \tag{2}$$

wobei

- $L(u)$ die Menge der Seiten, die einen Link auf u haben,
- N_i Anzahl der Seiten, auf die die Seite i verlinkt,
- N die Gesamtanzahl aller Webseiten ist und
- d ein Dämpfungsfaktor mit $0 \leq d \leq 1$ (typischerweise $d \approx 0,85$). Der genaue Zweck dieses Faktors wird im folgenden Abschnitt erläutert.

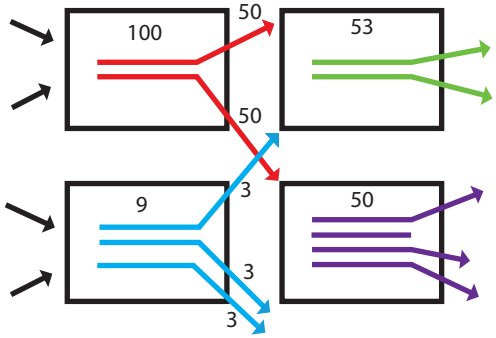

Abbildung 2:
Vererbung der Wichtigkeit zwischen Seiten durch gegenseitige Verlinkung ($d = 1$) [14]

Um zu vermeiden, dass nicht der gesamte PageRank einer Seite auf eine andere projiziert wird, wird ihr PageRank durch die Anzahl der ausgehenden Links geteilt. Ohne diese Sicherung könnte eine wichtige Seite einer unwichtigen Seite durch Verlinkung ihren PageRank übertragen und sie fälschlicherweise wichtig erscheinen lassen. Je mehr ausgehende Links eine Seite i hat, desto weniger gibt sie PageRank an die Seite k weiter (siehe Abbildung 2). Durch die Multiplikation der Summe mit dem Dämpfungsfaktor d wird das Ausmaß der Weitergabe des PageRanks sukzessive verringert.

3.2 Dead Ends und Spider Traps

3.2.1 Spider Traps

Spider Traps (auch Rank Sink genannt) sind Schleifen im Web Graph, aus denen kein Link aus der Schleife hinausführt. Webseiten, die auf eine solche

Schleife verlinken, würden ihren PageRank bei einer rekursiven Berechnung ohne Dämpfungsfaktor und den Summanden $(1-d)$ immer wieder erneut an die Webseiten in der Schleife vererben und im Gegenzug niemals PageRank von den Seiten innerhalb der Schleife erhalten, da keine Links aus der Schleife führen. Dies würde die Gesamtverteilung des PageRanks beeinflussen. Um dem entgegenzuwirken, wird der Dämpfungsfaktor d eingesetzt. Dieser kompensiert das Wachstum innerhalb der Schleife. Des Weiteren wird der an die Schleifen verloren gegangene PageRank durch eine Rank Source kompensiert. Diese Rank Source wird durch den Summanden $(1-d)$ umgesetzt. L. Page und S. Brin geben in [4] für d den Wert 0,85 als erprobt an.

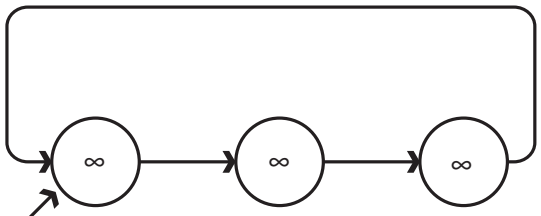

Abbildung 3:
Spider Trap Verhalten durch Rank Vererbung [14]

3.2.2 Dead Ends

Ein weiteres Problem stellen Dead Ends dar (auch Dangling Links genannt). Dead Ends sind Webseiten, die zwar verlinkt werden, aus denen aber keine Links hinausführen (vgl. Abbildung 1). In der Praxis können dies auch Seiten sein, die noch nicht erfasst wurden. Diese Links stellen ein Problem für das Modell dar, da unklar ist, wie der PageRank von dort aus weiterverteilt wird. Da Dead Ends den PageRank keiner anderen Seite verändern, werden sie vor der Berechnung entfernt und per Annahme ausgeschlossen. Dies verändert zwar die Normalisierung der PageRank Verteilung etwas, da nun die Links anderer Seiten auf die Dead Ends entfallen, sollte nach [14] jedoch keine großen Effekt haben. In der Praxis werden Dead Ends vor der PageRank-Berechnung entfernt und, nachdem die Verteilung berechnet wurde, wieder hinzugefügt. Alternativ ist es auch möglich beim Auftreten von Dead Ends den Zufallssurfer auf eine zufällige Seite teleportieren zu lassen. Dies ist allerdings in der Praxis ungünstig, da man diesen Sonderfall später bei der Optimierung getrennt betrachten müsste.

3.3 Das Modell des Zufallssurfers

Das PageRank-Konzept bildet das Verhalten eines Zufallssurfers auf dem Web Graphen ab. Der Zufallssurfer befindet sich zu einem Zeitpunkt $t = 0$ auf einer Webseite und klickt unabhängig vom Inhalt dieser auf Links und kommt so zur nächsten Seite. Die Wahrscheinlichkeit, dass der Zufallssurfer

einen bestimmten Link anklickt, sei dabei gleichverteilt und berechnet sich aus der Anzahl der Links, die er zu Auswahl hat.
Dabei wandert der Zufallssurfer nicht endlos von Link zu Link durch das Web, sondern bricht nach einer Zeit gelangweilt ab und entschließt sich, zu einer zufälligen Seite zu teleportieren. Die Konstante $(1-d)$ gibt die Wahrscheinlichkeit des Abbruchs gleichverteilt an alle Webseiten an.

3.4 Modellierung als Markov-Prozess

Betrachtet man das Modell des Zufallssurfers und die daraus resultierende Formel, so liegt es nahe, die PageRank-Verteilung als stochastischen Prozess aufzufassen. Ein Websurfer wählt ausgehend von einer Webseite u entweder

- mit einer Wahrscheinlichkeit von d eine Seite i, die von u verlinkt wird, oder
- mit Wahrscheinlichkeit $(1-d)$ teleportiert er sich zu einer anderen Seite. Diese Teleportation ist gleichverteilt auf alle Webseiten. Somit folgt eine Wahrscheinlichkeit von $\frac{(1-d)}{N}$, dass der Websurfer auf einer zufälligen Seite des Graphen landet.

Wir nehmen weiter an, dass Dead Ends bereits in einer Vorverarbeitung aus dem Graphen entfernt wurden. Wir definieren eine sogenannte Linkmatrix $A \in \mathbb{R}^{N \times N}$ als gewichtete Adjazenzmatrix des Web Graphen.

$$A_{ij} = \begin{cases} \frac{1}{N_j} & \text{, falls } j \text{ auf } i \text{ verlinkt bzw. wenn } i \in L(j) \\ 0 & \text{, sonst} \end{cases} \tag{3}$$

Durch die Konstruktion von A folgt:

$$\sum_{i=1}^{N} A_{ij} \quad = \quad 1,$$

mit $1 \leq j \leq N$ und $A_{ij} \geq 0$ für alle i, j.

Definition 1. Eine beliebige reelle Matrix $M \in R^{n \times n}$ heißt **nicht-negativ**, wenn alle Einträge größer gleich null sind. Eine quadratische nicht-negative reelle Matrix heißt **Markov-Matrix**, wenn ihre Spaltensumme für jede Spalte eins ist. Ein Vektor $u \in \mathbb{R}^n$ nennen wir **Wahrscheinlichkeitsvektor**, wenn seine Spaltensumme 1 ist und alle Einträge größer gleich null sind.

Bemerkung 2. Offensichtlich ist A aus (3) eine Markov-Matrix.

Satz 3. *Jede Markov-Matrix $M \in \mathbb{R}^{n \times n}$ hat einen nicht negativen Eigenvektor zum Eigenwert 1.*

Definition 4. Der Spektralradius einer $\rho(A)$ einer Matrix $A \in \mathbb{R}^{n\times n}$ ist der Betrag des betragsmäßig größten Eigenwerts von A: $\rho(A) = \max\limits_{1\leq i\leq n} |\lambda_i(A)|$. Wobei $\lambda_i(M)$ der i-te Eigenwert von M ist.

Theorem 5. *Der Spektralradius einer Markov-Matrix $\rho(M)$, $M \in \mathbb{R}^{n\times n}$ ist 1.*

Beweis. Per Definition von M gilt $\| M \|_1 = \max\limits_{j=1,\dots,n} \sum_{i=1}^{n} |m_{ij}| = 1$. Es gilt $\rho(M) := \max\limits_{1\leq i\leq n} |\lambda_i(M)|$. Wobei $\lambda_i(M)$ der i-te Eigewert von M ist. Wegen Satz 3 gilt somit $\rho(M) \geq 1$. Gleichzeitig gilt aber $\rho(M) \leq \| M \|_1 = 1$. Denn $\| M \|_1 = \sup\limits_{x\neq 0} \frac{\| Mx \|_1}{\| x \|_1} \geq \frac{\| Mv \|_1}{\| v \|_1} = \frac{\| \lambda v \|_1}{\| v \|_1} = |\lambda|$. $\Rightarrow \rho(M) = 1$ □

Definition 6. Ist $M \in \mathbb{R}^{n\times n}$ eine Markov-Matrix und $v \in \mathbb{R}^n$, so wird die Folge v, Mv, M^2v,... in $\mathbb{R}^n$ **Markov-Prozess** mit Anfangswert v genannt. Sei $S := \{S_1, \dots, S_{n+1}\}$ eine endliche Zustandsmenge. Eine **Markov-Kette** ist ein stochastischer Prozess $(X_t)_{t\in\mathbb{N}_0}$ mit Werten in S zu einer diskreten Zeit $t = 0, 1, 2, \dots$, der folgende sogenannte **Markov-Bedingung** erfüllt:

$$P(X_{t+1} = S_{n+1} \,|\, X_0 = S_0, \dots, X_n = S_n) = P(X_{t+1} = S_{n+1} \,|\, X_n = S_n)$$

Die Markov-Bedingung sagt also aus, dass das System nur vom aktuellen Zustand und nicht von den vorigen Zuständen abhängt (Gedächtnislosigkeit).

Sei nun weiter $v = \begin{pmatrix} v_1 \\ \vdots \\ v_N \end{pmatrix} \in \mathbb{R}^N$ der PageRank-Vektor, wobei v_i der PageRank $\tilde{P}(i)$ der Webseite i ist. Wir können somit die PageRank-Formel für alle Knoten im Graph als lineares Gleichungssystem umschreiben:

$$v = \frac{1-d}{N} \cdot e + d \cdot A \cdot v \,,\; e = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^N \tag{4}$$

Diese Gleichung ist rekursiv. Da v die Wahrscheinlichkeit beschreibt, mit der der Zufallssurfer auf der jeweiligen Webseite i mit Wahrscheinlichkeit v_i landet, muss gelten:

$$\sum_{i=1}^{N} v_i = 1 \tag{5}$$

Somit ist v ein Wahrscheinlichkeitsvektor. Insbesondere gilt: $e^T \cdot v = 1$. Weiter gilt:

$$\forall\, 1 \leq j \leq N : \frac{1}{N} \sum_{i=1}^{N} E_{ij} = 1, \tag{6}$$

mit $E_{ij} = 1$, $\forall i\,,\, j$. Wegen (6) und (5) lässt sich somit (4) wie folgt umstellen:

$$\begin{aligned} v &= \frac{1-d}{N} \cdot e \cdot (e^T \cdot v) + d \cdot A \cdot v && (7) \\ &= \left(\frac{1-d}{N} \cdot (e \cdot e^T) \cdot v + d \cdot A \cdot v \right) && (8) \\ &= \underbrace{\left(\frac{1-d}{N} \cdot E_{ij} + d \cdot A \right)}_{\tilde{A}} \cdot v && (9) \end{aligned}$$

Also ist v Eigenvektor von $\tilde{A}$ zum Eigenwert 1. Mit der Bedingung $0 \leq d \leq 1$ folgt aus (6),(5) zusätzlich:

$$\forall\, 1 \leq j \leq N : \sum_{i=1}^{N} \tilde{a}_{ij} = 1 \tag{10}$$

Bevor wir genauer auf die weiteren Eigenschaften von $\tilde{A}$ eingehen, definieren wir den Begriff der Primitivheit:

Definition 7. Eine nicht-negative Matrix $M \in \mathbb{R}^{n \times n}$ heißt **primitiv**, wenn sie irreduzibel ist und nur einen einzigen Eigenwert auf dem Spektralradius besitzt. Eine Matrix $M \in \mathbb{R}^{n \times n}$ heißt reduzibel (irreduzibel), wenn es eine (keine) Permutationsmatrix P gibt, sodass $P^T A P = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix}$, wobei X und Z quadratische Matrizen sind.

Bemerkung 8. Wegen (10) ist $\tilde{A}$ nicht-negativ. Als Markov-Matrix hat sie den Eigenwert 1 (Satz 3), welcher nach Theorem 5 auf dem Spektralradius liegt. Offensichtlich sind alle Einträge von $\tilde{A}$ wegen dem Summanden $\frac{1-d}{N} \cdot E_{ij}$in (8) ungleich null, sodass auch keine Permutation der Form $P^T \tilde{A} P = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix}$ existieren kann. Somit ist $\tilde{A}$ primitiv.

Wir wollen das Modell des Zufallssurfers nun als Markov-Kette betrachten, dabei ist $\tilde{A}$ die Zustandsübergangsmatrix. Als Startvektor wählen wir $v(0) = (p_1(0), \ldots, p_n(0))^T$ und $p_i(0) = \frac{1}{n}$. Wir berechnen die Wahrscheinlichkeit, dass der Zufallssurfer zum Zeitpunkt $t = 1$ auf einer Seite j ist. Dabei sei S_i der Zustand aus der endlichen Zustandsmenge S, in dem sich

der Zufallssurfer auf einer Seite i befindet.

$$\begin{aligned} p_j(1) &= P(X_1 = S_j) \\ &= \sum_{i=1}^{n} P(X_1 = S_j \wedge X_0 = S_i) \\ &= \sum_{i=1}^{n} P(X_0 = S_i) \cdot P(X_1 = S_j | X_0 = S_i) \\ &= \sum_{i=1}^{n} p_i(0) \cdot \tilde{a}_{ij} \end{aligned}$$

Daraus folgt, dass

$$v(1) = v(0) \cdot \tilde{A} \tag{11}$$

und durch sukzessives Einsetzen folgt gleichzeitig:

$$v(t) = v(0) \cdot \tilde{A}^t.$$

Satz 9. *(Perron-Frobenius-Theorem) Ist M eine primitive Markov-Matrix, so ist der Eigenvektor $x = (x_1, \ldots, x_n)^T$ mit $\sum_{i=1}^{n} x_i = 1$ zum Eigenwert 1* ***eindeutig bestimmt*** *und positiv. Weiter existiert $\lim_{t\to\infty} p(0)A^t = x$ und es gilt $Mx = x$.*

Für den Beweis von Satz 9 wird auf [16] verwiesen. In anderen Worten sagt der Satz für unser v aus, dass eine Stationäre Verteilung für unseren Markov-Prozess mit der Übergangsmatrix $\tilde{A}$ existiert. Des Weiteren sagt er aus, dass diese Verteilung eindeutig ist und sogar dem Eigenvektor zum Eigenwert 1 entspricht. Somit existiert der PageRank durch den Grenzwert $\lim_{t\to\infty} v(0)\tilde{A}^t = x$, welcher eindeutig bestimmt ist.

3.5 Berechnungsverfahren

Abschnitt 3.4 hat die mathematischen Grundlagen zur Existenz und Eindeutigkeit für die Berechnung des PageRanks gelegt. Im Folgenden beschäftigen wir uns mit dem Berechnungsverfahren des PageRanks.

3.5.1 Algebraische Berechnung

Wir wissen, dass der PageRank-Vektor der einzige Eigenvektor der Matrix $\tilde{A}$ zum Eigenwert 1 ist. Daher könnte man nun auf die Idee kommen, diesen Eigenvektor v durch Lösung des lineares Gleichungssystems

$$(I + d \cdot A)v = \frac{1-d}{N} \cdot e$$

zu berechnen. Dazu existieren in der Numerischen Linearen Algebra viele Verfahren: Diese sind in der Praxis jedoch nicht durchführbar, da die Dimension von A (für Webseiten $n > 10^9$) sehr groß ist. Beispielsweise kann der Inverse-Iteration-Algorithmus v bereits in einer Iteration finden, jedoch müsste dafür die Inverse-Matrix $\tilde{A}^{-1}$gebildet werden, wodurch bereits ein Aufwand von $\mathcal{O}(n^3)$ entsteht [11].
Das heißt: Selbst wenn es speichertechnisch möglich wäre, eine solche Matrix aufzustellen und zu speichern, würde eine Berechnung von v entsprechend sehr lange dauern.

3.5.2 Potenzmethode (Power Method)

Man kann jedoch die Stationäre Verteilung der Markov-Kette mit der Potenzmethode approximieren, da wir wissen, dass die Markov-Kette mit Übergangsmatrix $\tilde{A}$ konvergiert. Der folgenden Algorithmus approximiert v für eine Genauigkeit $\epsilon > 0$, dabei ist $v_0 \in \mathbb{R}^n$ die Startverteilung, die wir mit $v_0 = e \cdot \frac{1}{n}$ initialisieren.

---

function v	$=$	PowerMethod(v_0,$\tilde{A}, \epsilon$){
i	$\leftarrow$	1
v_i	$\leftarrow$	v_0
repeat		
	v_i	$\leftarrow \tilde{A} \cdot v_{i-1}$
	δ	$\leftarrow \|v_i - v_{i-1}\|_1$
	i	$\leftarrow i + 1$
until	$\delta < \epsilon$	
return	v_{i-1}	
}		

Der Algorithmus bricht ab, sobald die erwünschte Genauigkeit erreicht ist. Dabei ist zu beachten, dass jeweils nur die Werte v_i und v_{i-1} gespeichert werden müssen. Der Algorithmus kann jedoch noch optimiert werden, um die speziellen Eigenschaften von $\tilde{A}$ auszunutzen, indem man $v_i = \tilde{A} \cdot v_{i-1}$ betrachtet:

$$v_i = \tilde{A} \cdot v_{i-1} = \frac{1-d}{N} \cdot e + d \cdot A \cdot v_{i-1} \tag{12}$$

Da A gegenüber $\tilde{A}$ sehr dünn besetzt ist, kann die Berechnung jeder Matrix-Vektor Multiplikation in $nnz(A)$ Schritten erfolgen (z. B. indem nur die

Nicht-Null-Einträge gespeichert werden), wobei $nnz(A)$ die Anzahl der Nicht-Null-Einträge ist. Da $nnz(A)$ im Mittel kleiner $10n$ ist [12], liegen die Kosten für eine Iteration in $\mathcal{O}(n)$.
Die Konvergenzrate der Potenzmethode wird durch $\frac{|\lambda_2|}{|\lambda_1|}$ angegeben, wobei λ_1 und λ_2 die betragsmäßig geordneten Eigenwerte von $\tilde{A}$ sind [6]. Da $\tilde{A}$ eine Markov-Matrix ist, gilt: $1 \geq |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \geq 0$ [6].
Insbesondere gilt für die spezielle Matrix $\tilde{A}$ (auch Google-Matrix): $|\lambda_2| \leq d$ bzw. sogar $\lambda_2 = d$ [7]. Also konvergiert die Potenzmethode zu einer Stationären Verteilung mit Konvergenzrate d ($\approx 0{,}85$). In [14] ist die Rede von etwa 52 nötigen Iterationen.

4 Effiziente Implementierung

Um eine PageRank basierende Suchmaschine möglichst aktuell zu halten, spielt die schnelle Ausführung der Bewertungsmethode eine große Rolle. Im Rahmen dieser Arbeit werden weitere Ansätze zur effizienten Verbesserung des PageRank-Algorithmus kurz vorgestellt und diskutiert.

4.1 Ausnutzung der Blockstrukturen im Web Graph

Empirische Untersuchungen in [9] zeigen, dass der interne Verlinkungsgrad ($\approx 80\,\%$) einer Seite, also die Anzahl der Links auf Seiten derselben Domain, sehr viel höher ist als der externe Verlinkungsgrad. Daher liegt es nahe, nicht mehr – wie zuvor – separat einzelne Seiten einer Domain zu betrachten, sondern eine Domain als Block zu behandeln. Dazu wird der Web Graph in Blöcke von Domains aufgeteilt und für die Webseiten der Domains separat ein lokaler PageRank berechnet.
Anschließend wird für einen Block ein relativer PageRank berechnet (BlockRank) und in Beziehung zu dem lokalen PageRanks der Webseiten gebracht. Man erhält somit eine bessere Approximation für den Startvektor v_0, wodurch das anschließend durchgeführte Verfahren aus 3.5.2 weniger Iterationen benötigt. Da die Matrizen für die jeweiligen Blöcke wesentlich kleiner sind als der Web Graph selbst, passen diese entweder komplett in den Hauptspeicher oder benötigen insgesamt weniger I/O-Zeit. Je nach Parametern ist dieses Verfahren etwa doppelt so schnell wie das herkömmliche Iterieren. Für Details zum BlockRank-Verfahren sei auf [9] verwiesen.

4.2 Filter-Based-Adaptive PageRank

Die Idee des Adaptiven PageRanks resultiert aus der Beobachtung, dass manche Webseiten langsamer (insbesondere solche mit hohem PageRank) als andere konvergieren. Der zugrunde liegende Web Graph G lässt sich also in einem Iterationsschritt k in folgende Mengen unterteilen: $C_k \subseteq G$ die Menge der Webseiten, die zum Iterationsschritt k bereits konvergiert sind,

$N_k \subseteq G$ die Menge der Webseiten, die noch nicht konvergiert sind. $\tilde{A}$ ist die Übergangsmatrix. Durch Permutation der Zeilen von $\tilde{A}$ lässt sich eine Matrix $\tilde{A}'$ bilden, sodass $\tilde{A}' = \begin{pmatrix} \tilde{A}_N \\ \tilde{A}_C \end{pmatrix}$, $\tilde{A}_N \in \mathbb{R}^{m \times n}$, $\tilde{A}_C \in \mathbb{R}^{(n-m) \times n}$. $\tilde{A}_N$ und $\tilde{A}_C$ seien genau so gewählt, dass sie die zugehörigen Zeilen zu N_k bzw. C_k beinhalten. Analog sei $v(k) = \begin{pmatrix} v_N \\ v_C \end{pmatrix}$ als PageRank-Approximation definiert. Da nach Annahme v_C bereits konvergiert ist, gilt somit:

$$v(k+1) = \begin{pmatrix} \tilde{A}_N \\ \tilde{A}_C \end{pmatrix} \cdot \begin{pmatrix} v_N \\ v_C \end{pmatrix} = \begin{pmatrix} \tilde{A}_N \cdot v_N \\ v_C \end{pmatrix} \tag{13}$$

v_C muss also nicht mehr neu berechnet werden. Jedoch ist das Permutieren $\tilde{A}$ nach $\tilde{A}'$ in der Praxis sehr teuer und somit unrealistisch. Wir umgehen daher die Permutation, indem wir die entsprechenden Stellen von $\tilde{A}$ sukzessive nullieren. Dazu definieren wir folgende "Filter" $v'(k) \in \mathbb{R}^n$,$\tilde{A}'' \in \mathbb{R}^{n \times n}$ mit:

$$v'(k)_j = \begin{cases} v(k)_j & \text{falls } j \in C_k \\ 0 & \text{sonst} \end{cases} \text{ und } \tilde{A}''_{ij} = \begin{cases} 0 & \text{falls } i \in C_k \\ \tilde{A}_{ij} & \text{sonst} \end{cases}$$

daraus folgt:

$$v(k+1) = \tilde{A}''_{ij} \cdot v(k) + v'(k)$$

Somit ist kein teures Permutieren von $\tilde{A}$ notwendig. Man könnte nun denken, dass die Kosten pro Iterationsschritt gleich geblieben sind, da sich die $dim(\tilde{A}'') = dim(\tilde{A})$. Da aber – wie bereits in 3.5.2 erläutert – die Kosten der Matrix-Vektor Multiplikation vor allem von $nnz(\tilde{A}'')$ abhängen und per Definition von $\tilde{A}''$ gilt: $nnz(\tilde{A}'') \leq nnz(\tilde{A})$, stellt dies offensichtlich eine Verbesserung da.
Ein Problem dieses Verfahrens sind Webseiten, die kurzzeitig langsam konvergieren und von denen fälschlicherweise angenommen wird, dass diese bereits konvergiert sind. Dies kann behoben werden, indem die Filter nicht bei jeder Iteration, sondern nach einer bestimmten Anzahl von Iterationen gebildet werden. Nach [11] stellt der Algorithmus eine Verbesserung von $18-28\,\%$ dar.

4.3 Quadratische Extrapolation

Die Methode der Quadratischen Extrapolation beschleunigt die Konvergenz der Potenzmethode, wobei die Grundidee darin besteht, den Vektor $v(k)$ durch seine Vorgänger $v(k-1)$, $v(k-2)$, $v(k-3)$ abzuschätzen.
Es wird die Annahme getroffen, dass $\tilde{A}$ lediglich nur drei Eigenvektoren u_1, u_2 und u_3 besitzt, deren zugehörige Eigenwerte λ_1, λ_2 und λ_3 ungleich

Null sind. Weiter wird angenommen, die Stationäre Verteilung des Markov-Prozesses sei eine Linearkombination aus diesen Eigenvektoren. Der PageRank wird letztendlich durch Schätzungen für die drei Eigenvektoren bestimmt.
Natürlich nimmt das Verfahren an, $\tilde{A}$ hätte nur drei Eigenvektoren, was lediglich zu einer Approximation von $v(k)$ führt. Allerdings erzielt das Verfahren je nach Wahl des Dämpfungsfaktors d eine Verbesserung von $23 - 69\,\%$. Das Verfahren zeigt insbesondere gute Ergebnisse bei einem hohen Dämpfungsfaktor $d \approx 1$. Der Overhead, der durch die Quadratische Extrapolation entsteht, lässt bei geeigneter Wahl der Verfahren sich auf $\mathcal{O}(n)$ reduzieren. Des Weiteren stellte man fest, dass es nicht notwendig ist, die Quadratische Extrapolation so oft wie möglich anzuwenden, da nach etwa fünf Anwendungen nahezu der maximale Nutzen erreicht ist. Für weitere Details sei an dieser Stelle auf [10] verwiesen.

4.4 Topic-Sensitive-PageRank

Bei der ursprünglichen Berechnung des PageRanks wird ein einzelner PageRank-Vektor über den gesamten Web Graphen ermittelt, um die relative Wichtigkeit von Webseiten zu bestimmen. Dies geschieht jedoch vollkommen unabhängig von einer bestimmten Suchanfrage. Die Berechnung von unterschiedlichen PageRank-Werten für Seiten, welche jeweils auf bestimmte Themen abgestimmt sind, könnte jedoch präzisere Suchresultate hervorbringen. Das Topic-Sensitive-PageRank-Schema [8] berechnet eine $N \times k$ Approximation für $\tilde{A}$ mit k Themen. Die Zugehörigkeit zwischen Seiten und Themen werden durch Menschen z. B. im Open Directory Project [1] festgelegt. Dabei erhalten Seiten, die in einer bestimmten Kategorie auftauchen, bereits initial einen höheren PageRank für diese Kategorie. Bei einer Suchanfrage wird letztendlich der Themenkontext der Anfrage bestimmt und die jeweiligen PageRank-Werte werden passend kombiniert.

5 Zusammenfassung

Wir haben das Konzept des PageRanks im Zusammenhang eines Kriteriums für die Wichtigkeit einer Seite im Web durch das Verhalten eines Zufallssurfers modelliert. Darüberhinaus haben wir die Existenz und Eindeutigkeit des PageRanks im Web Graphen erläutert und festgestellt, dass Rechenoperationen auf dem Web Graph in der Praxis nicht immer durchführbar sind und klassische Lösungsansätze hier versagen. Aus diesem Grund wurde ein iteratives Berechnungsverfahren vorgestellt, für welches wir anschließend weitere Optimierungsvarianten in Ausblick gestellt haben.
Das Thema PageRank-Algorithmus und die Interpretation des Webs als Graph ist ein weites Forschungsgebiet, wobei in dieser Arbeit nur ein kleiner Einblick in die Grundlagen gegeben werden kann. Insbesondere stellt

das PageRank-Verfahren und die dahinter liegenden Suchverfahren viele zusätzliche Anforderungen an Implementierung, Datenstrukturen, Effizienz, Hardware- und Serverdesign, Sicherheit, Data-Mining, Personalisierung und vieles mehr.

Literatur

[1] *Open Directory Project*; http://www.dmoz.org.

[2] Jesse Alpert und Nissan Hajaj: *We knew the web was big...*; *Google Blog*; July 2008; http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html Letzte Sichtung: 31. Nov. 2011.

[3] Amihood Amir, Moshe Lewenstein und Noa Lewenstein: *Pattern Matching in Hypertext*; *Journal of Algorithms*; 35(1), S. 82 – 99; 2000.

[4] Sergey Brin und Lawrence Page: *The anatomy of a large-scale hypertextual Web search engine*; in *Proceedings of the seventh international conference on World Wide Web 7*; WWW7; S. 107–117; 1998.

[5] Andrei Broder, Ravi Kumar et al.: *Graph structure in the Web*; *Computer Networks*; 33(1-6); 2000.

[6] Lars Eldén: *A Note on the Eigenvalues of the Google Matrix*; 2004.

[7] Taher Haveliwala und Sepandar Kamvar: *The Second Eigenvalue of the Google Matrix*; Technical Report 2003-20; Stanford InfoLab; 2003.

[8] Taher H. Haveliwala: *Topic-sensitive PageRank*; in *WWW '02: Proceedings of the 11th international conference on World Wide Web*; S. 517–526; 2002.

[9] Sepandar Kamvar, Taher Haveliwala et al.: *Exploiting the Block Structure of the Web for Computing PageRank*; Technical Report 2003-17; Stanford InfoLab; 2003.

[10] Sepandar Kamvar, Taher Haveliwala et al.: *Extrapolation methods for accelerating PageRank computations*; S. 261–270; 2003.

[11] Sepandar Kamvar, Taher Haveliwala und Gene Golub: *Adaptive Methods for the Computation of PageRank*; Technical Report 2003-26; Stanford InfoLab; April 2003.

[12] Amy N. Langville und Carl D. Meyer: *Google's PageRank and beyond: the science of search engine rankings*; Princeton University Press; 2006.

[13] Danil Nemirovsky: *Web Graph and PageRank algorithm*; *JASS Conference*; 2005.

[14] Lawrence Page, Sergey Brin et al.: *The PageRank Citation Ranking: Bringing Order to the Web.*; Technical Report 1999-66; Stanford InfoLab; November 1999.

[15] Gopal Pandurangan, Prabhakar Raghavan und Eli Upfal: *Using PageRank to characterize web structure*; *Internet Math.*; 3(1), S. 1–20; 2006.

[16] E. Seneta: *Non-negative matrices and Markov chains*; Springer series in statistics; Springer; 2006.