Deeper Inside PageRank

Amy. N. Langville & Carl D. Meyer

Ausarbeitung zum Seminar

Moderne Anwendungen der Theorie der Markov-Ketten

Tim Massel

Juli 2016

Betreuer: Prof. Dr. Wolfgang König
Priv. Doz. Dr. Konstantin Fackeldey

Einleitung

Die vorliegende Arbeit wurde im Rahmen des Seminars **Moderne Anwendungen der Theorie der Markov-Ketten** im Sommersemester 2016 an der Technischen Universität Berlin verfasst. Diese Ausarbeitung ist Teil eines dreiteiligen Themenkomplexes. In einem ersten Teil wurde die mathematische Grundlage in der Perron-Frobenius-Theorie gelegt. Die daraus hergeleiteten Resultate werden hier vorausgesetzt. Der vorliegende zweite Teil stellt eine Zusammenfassung ausgewählter Themen des Papers *Deeper Inside PageRank* [4] bereit. In einem dritten Teil werden diese noch vertieft und ergänzt.

Das Originalpaper *Deeper Inside PageRank* beschäftigt sich umfassend mit der Modellierung des Internets durch Übergangsgraphen und den daraus resultierenden Übergangsmatrizen. Es knüpft an den Ideen der beiden Google-Gründer Sergey Brin und Larry Page an, die sich mit der Frage beschäftigten, wie jeder Seite des Netzes eine bestimmte Wichtigkeit zugeordnet werden kann, um diese in einer Suchmaschine verwenden zu können. Um ein solches Ranking zu erhalten, stelle man sich das Surfverhalten eines Internetnutzers zufällig vor. Es stellt sich nun die Frage: mit welcher Wahrscheinlichkeit befinde man sich nach "langem" Surfen auf einer beliebigen Seite i. Gelingt es diese Wahrscheinlichkeiten eindeutig zu ermitteln, so lassen sie sich in dem stochastischen Vektor π^{T} zusammenfassen - dem PageRank-Vektor. Ausgehend von der Frage unter welchen Voraussetzungen nun dieses π^{T} existiert und eindeutig ist, werden in dieser Arbeit zwei Lösungsstrategien zur Berechnung des gewünschten PageRank-Vektors vorgestellt. Die mathematische Theorie geht auf Perron und Frobenius zurück, die die Existenz und Eindeutigkeit der sog. stationären Verteilung gewährleistet, sobald die betrachtete Übergangsmatrix, die das zufällige Surfverhalten beschreibt, primitiv ist. Primitivität ist dadurch charakterisiert, dass Irreduzibilität und Aperiodizität vorliegen. Um Missverständnissen in der Notation vorzubeugen, sei darauf hingewiesen, dass wir für eine positive (bzw. nicht-negative) Matrix A die Symbolik $A > 0$ (bzw. $A \geq 0$) verwenden.

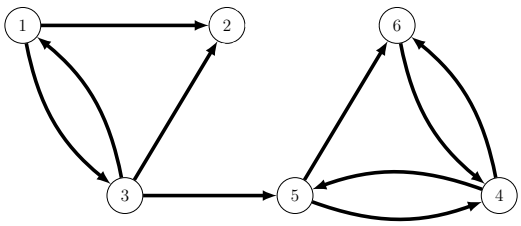
Zunächst widmen wir uns der **Modellierung** des Netzes und dem damit verbundenen zufälligen Surfen. Wir benötigen also alle Wahrscheinlichkeiten, um von einer beliebigen Seite i auf eine Seite j zu gelangen. Diese Informationen werden wir in einer Übergangsmatrix G zusammenfassen - der Google-Matrix. Genügt die Google-Matrix nun den Voraussetzungen (Primitivität) der Perron-Frobenius-Theorie, so bekommen wir mit der **Potenzmethode** automatisch ein Verfahren an die Hand, mit der wir unsere Lösung approximativ bestimmen können. Die Konvergenzgeschwindigkeit dieser Methode ist ausreichend studiert und hängt von den beiden dominanten Eigenwerten der primitiven Matrix ab. Wir werden hier einen Beweis vorstellen, der das gesamte Spektrum der Google-Matrix charakterisiert und damit Aussagen über die Konvergenzgeschwindigkeit macht. In dem letzten Abschnitt beschäftigen wir uns dann noch mit einem weiteren Lösungsverfahren zur Bestimmung von π^{T}. Wir formulieren unser Problem als ein **lineares Gleichungssystem** und wollen dieses effizienter Lösen als mit den üblichen Verfahren. Unter Ausnutzung der Linkstruktur wird es genügen ein wesentlich kleineres LGS zu lösen, sodass wir bei der Implementierung einen großen Teil an Rechenschritten einsparen können.

Das Modell

Die beiden Stanford-Studenten Sergey Brin und Larry Page beschäftigten sich seit dem Jahr 1995 mit dem Problem jeder Internetseite eine eindeutige Wichtigkeit zuzuordnen, damit bei einer benutzerfreundlichen Suchabfrage die Ergebnisse nach Prioritäten vorsortiert werden können. Dabei soll eine Seite besonders wichtig sein, wenn viele Seiten auf diese verweisen. Darüber hinaus sollen Links von wichtigen Seiten höheren Einfluss haben als Links von unwichtigeren. Diese sehr simple, aber durchaus nachvollziehbare Überlegung, bildet den Grundstein für alle weiteren Modellierungen.

Die Konstruktion der Google-Matrix

Die Idee war das Netz durch einen Übergangsgraphen zu repräsentieren, bei dem jeder Knoten einer Internetseite entspricht und eine gerichtete Kante (i, j) existieren soll, wenn ein Link von der Seite i auf die Seite j verweist. Exemplarisch sei folgendes kleines Internet gegeben:



Das Surfverhalten eines jeden Nutzers wird nun durch einen *zufälligen Web-Surfer* simuliert. D.h. ausgehend von einer Startseite (z.B. Seite 1) entscheidet sich dieser *zufällige Web-Surfer* gleichwahrscheinlich für einen der ausgehenden Links. Auf der neu erreichten Seite fährt er mit diesem Verfahren gedächtnislos (also unabhängig davon wie er auf diese Seite gelangt ist) fort und entscheidet sich erneut gleichwahrscheinlich für einen der ausgehenden Links. Die Idee dahinter ist, dass auf lange Sicht jede Seite eine bestimmte Wahrscheinlichkeit erhält, mit der sich ein Internetnutzer auf ihr befindet. Die Informationen der Linkstruktur können äquivalent in der Hyperlinkmatrix H zusammengefasst werden:

$$H = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Hier wird ein Problem deutlich: bei der zweiten Zeile handelt es sich um eine Nullzeile. Diese resultiert aus der Tatsache, dass von der Seite 2 keine Links wegführen und damit der repräsentierende Knoten einen Outdegree von 0 hat. Solche Knoten werden als *dangling nodes* bezeichnet und existieren im Internet zu einem hohen Anteil. Jedes Bild, Video oder jede PDF-Datei, die keine ausgehenden Links haben, gehören zu der Klasse der *dangling nodes*. Die Matrix H ist damit nicht stochastisch, was aber eine begehrenswerte Eigenschaft darstellt, wenn im Zusammenhang mit Markov-Ketten gearbeitet werden soll. Man behilft sich hier der Tatsache, dass praktisches Suchverhalten nicht durch das Erreichen eines *dangling nodes* beendet wird. Internetnutzer haben beispielsweise die Möglichkeit in der URL-Zeile eine beliebige Internetadresse einzugeben und somit zu jeder beliebigen Seite des Netzes zu springen, wenn man an einem *dangling node* angekommen ist. Die Wahrscheinlichkeit $v_i > 0$ bezeichne die Wahrscheinlichkeit dann auf die Seite i zu springen. Der Zeilenvektor v^{T} fasst alle v_i zusammen, sodass eine neue Matrix U konstruiert werden kann:

$$U = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = H + av^{\mathrm{T}} \ .$$

Dabei stellt a einen Indikatorvektor dar, d.h.

$$a_i = \begin{cases} 1, & i \textit{ dangling node} \\ 0, & \text{sonst.} \end{cases}$$

Damit werden Nullzeilen durch v^{T} ersetzt und die resultierende Übergangsmatrix U ist nun stochastisch. Der Satz von Perron-Frobenius kann aber nicht angewendet werden, da die Matrix U i.A. nicht irreduzibel ist. Die Knoten $\{4, 5, 6\}$ bilden in diesem Beispiel eine abgeschlossene Menge, also einen reduziblen Subgraphen. Beholfen wird sich hier aber mit einem einfachen ebenfalls praktischen Trick: das Surfverhalten eines Internetnutzers hängt zwar sehr stark von der Linkstruktur (in U) ab, doch die Möglichkeit in der URL-Zeile eine beliebige Internetadresse einzugeben besteht jederzeit (man muss sich nicht an einem *dangling node* befinden). Der Lösungsansatz lautet damit wie folgt:

(i) mit Wahrscheinlichkeit $\alpha \in (0, 1)$ folge man der Linkstrukur in U,

(ii) mit Wahrscheinlichkeit $1 - \alpha$ springe man zu einer beliebigen Seite.

Mit welcher Wahrscheinlichkeit man im Fall (ii) dann auf die einzelnen Seiten springt, soll ebenfalls mit der Verteilung v^{T} beschrieben werden. Dies führt uns letztlich zur Konstruktion der Google-Matrix

$$G = \alpha U + (1 - \alpha)ev^{\mathrm{T}}.$$

Mit $e = (1, \ldots, 1)^{\mathrm{T}}$ werde der Einsvektor bezeichnet. Setzen wir α beispielsweise auf 0.9, so erhalten wir die Matrix

$$G = \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}.$$

Die Potenzmethode

Die Google-Matrix ist nach Konstruktion eine positive Matrix, sodass der Satz von Perron auf diese anwendbar ist. Konkret bedeutet dies, dass das Eigenvektorproblem

$$\pi^{\mathrm{T}} = \pi^{\mathrm{T}} G$$

eine (bis auf Skalierung) eindeutige positive Lösung hat. Durch Normierung erhalten wir einen stochastischen Vektor, der damit eindeutig die Wahrscheinlichkeiten zusammenfasst, mit der sich ein zufälliger Web-Surfer nach langem Surfen auf einer beliebigen Seite befindet. Je höher diese Wahrscheinlichkeit nun für eine Seite i ist, desto höher soll auch deren Ranking ausfallen. Daher werde π^{T} auch als PageRank-Vektor bezeichnet. Die Potenzmethode liefert uns nun zusätzlich eine Möglichkeit diese Lösung näherungsweise zu bestimmen. Für jede Startverteilung $\pi^{(0)}$ konvergiert das rekursive Schema

$$\pi^{(k)\mathrm{T}} = \pi^{(k-1)\mathrm{T}} G$$

gegen die stationäre Verteilung π^{T}. Die Konvergenzgeschwindigkeit dieser Methode hängt von den beiden dominanten Eigenwerten λ_1 und λ_2 der Google-Matrix G ab. Diese ist stochastisch und hat daher den Eigenwert $\lambda_1 = 1$. Da es sich bei G aber zusätzlich um eine positive Matrix handelt, folgt mit dem Satz von Perron, dass der subdominante Eigenwert $\lambda_2 < 1$ sein muss. Damit gilt für eine Konstante $c \in \mathbb{R}_+$

$$||\pi^{(k)\mathrm{T}} - \pi^{\mathrm{T}}||_\infty \leq c \cdot |\tfrac{\lambda_2}{\lambda_1}|^k = c \cdot |\lambda_2|^k.$$

Automatisch sind wir daran interessiert wie die Eigenwerte der Google-Matrix (insbesondere natürlich λ_2) aussehen. Das folgende Theorem liefert eine sehr umfassende Antwort.

Theorem

Sei $\sigma(U) = \{1, \mu_2, \ldots, \mu_n\}$ das Spektrum der stochastischen Matrix U, dann ist das Spektrum der Google-Matrix $G = \alpha U + (1-\alpha) e v^{\mathrm{T}}$ gegeben durch $\sigma(G) = \{1, \alpha\mu_2, \ldots, \alpha\mu_n\}$.

Beweis. Da U eine stochastische Matrix ist, hat sie $(1, e)$ als Eigenpaar. Wir möchten eine Ähnlichkeitstransformation durchführen, benötigen also eine invertierbare Matrix

$$Q = \begin{pmatrix} e & X \end{pmatrix} \in \mathbb{R}^{n,n}.$$

Ihre Inverse sei gegeben durch

$$Q^{-1} = \begin{pmatrix} y^{\mathrm{T}} \\ Y^{\mathrm{T}} \end{pmatrix}.$$

Wir erhalten aus der Identität

$$I = Q^{-1}Q = \begin{pmatrix} y^{\mathrm{T}}e & \mathbf{0} \\ Y^{\mathrm{T}}e & Y^{\mathrm{T}}X \end{pmatrix}$$

die beiden für uns nützlichen Gleichungen

(i) $y^{\mathrm{T}}e = 1$ und

(ii) $Y^{\mathrm{T}}e = \mathbf{0}$.

Wenden wir nun die Transformation auf die stochastische Matrix U an, so erhalten wir

$$Q^{-1}UQ = \begin{pmatrix} y^{\mathrm{T}}Ue & * \\ Y^{\mathrm{T}}Ue & Y^{\mathrm{T}}UX \end{pmatrix} = \begin{pmatrix} y^{\mathrm{T}}e & * \\ Y^{\mathrm{T}}e & Y^{\mathrm{T}}UX \end{pmatrix} = \begin{pmatrix} 1 & * \\ \mathbf{0} & Y^{\mathrm{T}}UX \end{pmatrix},$$

wobei uns die mit $*$ gekennzeichneten Blöcke nicht interessieren. Da die resultierende Matrix obere Blockdreiecksgestalt hat und die Eigenwerte unverändert bleiben, folgt

$$\sigma(Y^{\mathrm{T}}UX) = \{\mu_2, \ldots, \mu_n\}.$$

Wir sind aber an den Eigenwerten der Google-Matrix interessiert und wenden deshalb die gleiche Transformation auf G an. Mit der Eigenschaft $v^{\mathrm{T}}e = 1$ des stochastischen Vektors v^{T} ergibt sich

$$\begin{aligned} Q^{-1}GQ &= Q^{-1}(\alpha U + (1-\alpha)ev^{\mathrm{T}})Q \\ &= \alpha Q^{-1}UQ + (1-\alpha)Q^{-1}ev^{\mathrm{T}}Q \\ &= \begin{pmatrix} \alpha & * \\ \mathbf{0} & \alpha Y^{\mathrm{T}}UX \end{pmatrix} + (1-\alpha)\begin{pmatrix} y^{\mathrm{T}}e \\ Y^{\mathrm{T}}e \end{pmatrix}\begin{pmatrix} v^{\mathrm{T}}e & * \end{pmatrix} \\ &= \begin{pmatrix} \alpha & * \\ \mathbf{0} & \alpha Y^{\mathrm{T}}UX \end{pmatrix} + (1-\alpha)\begin{pmatrix} 1 & * \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} 1 & * \\ \mathbf{0} & \alpha Y^{\mathrm{T}}UX \end{pmatrix}. \end{aligned}$$

Dies bewirkt, dass sich die Eigenwerte von G zusammensetzen aus

$$\begin{aligned} \sigma(G) &= \{1\} \cup \sigma(\alpha Y^{\mathrm{T}}UX) \\ &= \{1\} \cup \alpha\sigma(Y^{\mathrm{T}}UX) \\ &= \{1, \alpha\mu_2, \ldots, \alpha\mu_n\}. \square \end{aligned}$$

Mit diesem Resultat stellen wir fest, dass die Konvergenzgeschwindigkeit der Potenzmethode maßgeblich von der Wahl des Parameters α abhängt, denn es gilt

$$|\tfrac{\lambda_2}{\lambda_1}|^k = |\tfrac{\alpha\mu_2}{1}|^k = \alpha^k|\mu_2|^k \leq \alpha^k \to 0.$$

Die Wahl des Parameters α ist also von entscheidender Bedeutung. Eine schnelle Konvergenz (α nahe bei 0) spiegelt das tatsächliche Surfverhalten nicht gut wieder, da keine Relevanz auf die eigentliche Linkstruktur mehr gelegt wird. Wählt man dagegen α nahe bei 1, so wird ein befriedigendes Resultat mit der Potenzmethode erst nach unzähligen Iterationen zu erwarten sein. Gerade bei riesigen Datenmengen (der Google-Index von 2016: 60,000,000,000,000; d.h. umfasst über 60 Billionen Seiten) ist dies nicht wünschenswert. Zur geschickten Implementierung der Potenzmethode und zu weiteren Problemen, die bei der Wahl des α auftreten (Sensitivität) siehe den dritten Teil des Themenkomplexes.

2. Lösungsansatz: Lineares Gleichungssystem

Wir interessieren uns neben der Potenzmethode für weitere Strategien den PageRank-Vektor π^{T} zu bestimmen. Aus der ursprünglichen Gleichung

$$\pi^{\mathrm{T}} = \pi^{\mathrm{T}} G = \pi^{\mathrm{T}}(\alpha U + (1-\alpha) e v^{\mathrm{T}})$$

erhalten wir nach Umformung das lineare Gleichungssystem

$$\pi^{\mathrm{T}}(I - \alpha U) = (1-\alpha) v^{\mathrm{T}}.$$

Da das Lösen eines linearen Gleichungssystems kubischer Komplexität ($\mathcal{O}(n^3)$) und unser $n = 60 \cdot 10^{12}$ gigantisch ist, sind wir in diesem Abschnitt daran interessiert, wie wir eine effizientere Lösungsstrategie formulieren können.

Zunächst untersuchen wir die Matrix $(I - \alpha U)$ auf nützliche Eigenschaften und benötigen dafür das folgende Hilfsresultat.

Lemma

Für $B \geq 0$ und $s > \rho(B)$ sind Matrizen von der Form

$$sI - B$$

invertierbar mit $(sI - B)^{-1} \geq 0$.

Beweis. Der Beweis dieser Aussage folgt aus [5, S. 626 f.]. □

Satz

Es gelten folgende Eigenschaften:

(i) $(I - \alpha U)$ ist invertierbar,

(ii) $(I - \alpha U)^{-1} \geq 0$.

Beweis. Die Matrix $(I - \alpha U)$ genügt den Voraussetzungen des Lemmas, denn

- $\alpha U \geq 0$ und
- $\rho(\alpha U) = \alpha\rho(U) < \rho(U) \leq 1$. $\square$

Die Übergangsmatrix $U = H + av^{\mathrm{T}}$ setzte sich zusammen aus der Hyperlinkmatrix H und einer Matrix vom Rang 1, in der der stochastische Vektor v^{T} in den zu *dangling nodes* gehörigen Zeilen steht. Deshalb lässt sich das Gleichungssystem

$$\pi^{\mathrm{T}}(I - \alpha U) = (1 - \alpha)v^{\mathrm{T}}$$

schreiben als

$$\pi^{\mathrm{T}}(I - \alpha(H + av^{\mathrm{T}})) = (1 - \alpha)v^{\mathrm{T}}$$

und schließlich umformulieren zu

$$\pi^{\mathrm{T}}(I - \alpha H) = (1 - \alpha + \alpha\pi^{\mathrm{T}}a)v^{\mathrm{T}}.$$

Der Wert $\gamma := \pi^{\mathrm{T}}a$ beschreibt dabei die Summe der π_i, wobei i ein *dangling node* ist. Uns interessiert an dieser Stelle gar nicht, welchen konkreten Wert γ annimmt. Unsere Lösung π^{T} ist eindeutig bestimmt (bis auf Skalierung). Da wir aber gerade einen stochastischen Vektor erhalten wollen (also mit $\pi^{\mathrm{T}}e = 1$), normieren wir die Lösung noch entsprechend. Daher kann γ für unsere Zwecke einen beliebigen positiven Wert annehmen - das Endresultat bleibt wegen der Normierung am Ende unverändert. Unser Schema wird also nur durch einen Skalierungsfaktor ergänzt. Wir setzen der Einfachheit halber $\gamma := 1$, was zu dem Gleichungssystem

$$\pi^{\mathrm{T}}(I - \alpha H) = v^{\mathrm{T}}$$

führt.

Mit analogen Argumenten wird deutlich, dass auch $(I - \alpha H)$ die Voraussetzungen des Lemmas erfüllt und damit gilt

(i) $(I - \alpha U)$ ist invertierbar und

(ii) $(I - \alpha U)^{-1} \geq 0$.

Es ergibt sich unmittelbar

$$\pi^{\mathrm{T}} = v^{\mathrm{T}}(I - \alpha H)^{-1}$$

Das Berechnen der Inversen $(I - \alpha H)^{-1}$ wird nie explizit durchgeführt, doch liefert uns dies andere nützliche Eigenschaften. Wir stellen zunächst fest, dass die Hyperlinkmatrix H nach Konstruktion eine sehr dünn besetzte Matrix ist, was vorteilhaft beim Lösen eines linearen Gleichungssystems sein kann. Wir wissen aber ferner, dass es sich bei den Zeilen der

dangling nodes um Nullzeilen handelt. Permutieren wir die Zeilen jetzt derart, dass Zeilen zu *non-dangling nodes* oben und die zu *dangling nodes* darunter stehen, so erhält H die Form

$$\begin{array}{c} \\ ND \\ D \end{array} \begin{array}{c} \begin{array}{cc} ND & D \end{array} \\ \begin{pmatrix} H_{11} & H_{12} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \end{array}.$$

Insbesondere wird die Koeffizientenmatrix des Gleichungssystems zu

$$(I - \alpha H) = \begin{pmatrix} I - \alpha H_{11} & -\alpha H_{12} \\ \mathbf{0} & I \end{pmatrix}$$

und ihre Inverse zu

$$(I - \alpha H)^{-1} = \begin{pmatrix} (I - \alpha H_{11})^{-1} & \alpha (I - \alpha H_{11})^{-1} H_{12} \\ \mathbf{0} & I \end{pmatrix}.$$

Damit kann der PageRank-Vektor $\pi^{\mathrm{T}} = v^{\mathrm{T}}(I - \alpha H)^{-1}$ geschrieben werden als

$$\pi^{\mathrm{T}} = (v_1^{\mathrm{T}}(I - \alpha H_{11})^{-1} \mid \alpha v_1^{\mathrm{T}}(I - \alpha H_{11})^{-1} H_{12} + v_2^{\mathrm{T}}).$$

Da der Term $v_1^{\mathrm{T}}(I - \alpha H_{11})^{-1}$ auf beiden Seiten auftaucht, reduziert sich das Lösen des Gleichungssystems $\pi^{\mathrm{T}}(I - \alpha H) = v^{\mathrm{T}}$ auf den ersten Teil der *non-dangling nodes*. Konkret liefert der folgende Algorithmus das gewünschte Resultat.

Algorithmus

1. Löse $\pi_1^{\mathrm{T}}(I - \alpha H_{11}) = v_1^{\mathrm{T}}$.
2. Berechne $\pi_2^{\mathrm{T}} = \alpha \pi_1^{\mathrm{T}} H_{12} + v_2^{\mathrm{T}}$.
3. Normiere π^{T}.

Zusammenfassend muss nur ein Gleichungssystem für *non-dangling nodes* gelöst werden. Mit dieser Lösung lässt sich der zweite Teil π_2^{T} für die *dangling nodes* explizit ausrechnen. Für die Eindeutigkeit normieren wir am Ende. Da der Anteil der *dangling nodes* im Internet sehr groß ist und immer mehr an Bedeutung gewinnt (Fotos, Videos, wissenschaftliche Arbeiten, etc. werden online gestellt) ist eine Methode, die sich dieser Eigenschaft zu Nutze macht, von großem Interesse.

Literaturverzeichnis

[1] A. Berman and R. J. Plemmons. Nonnegative Matrices in the Mathematical Sciences. New York: Academic Press, Inc., 1979.

[2] S. Brin, L. Page, R. Motwami, and T. Winograd. “The PageRank Citation Ranking: Bringing Order to the Web.” Technical report, Computer Science Department, Stanford University, 1998.

[3] T. H. Haveliwala and S. D. Kamvar. “The Second Eigenvalue of the Google Matrix.” Technical report. Stanford University, 2003.

[4] A. N. Langville, C. D. Meyer. "Deeper Inside PageRank". Internet Mathematics Journal, 1(3): 335−80, 2005.

[5] C. D. Meyer. Matrix Analysis and Applied Linear Algebra. Philadelphia, PA: SIAM, 2000.

[6] C. Moler. “The World’s Largest Matrix Computation.” In Matlab News and Notes, October 2002, pp. 12−13.