# Übungsblatt 5

## Aufgabe 1 – PageRank

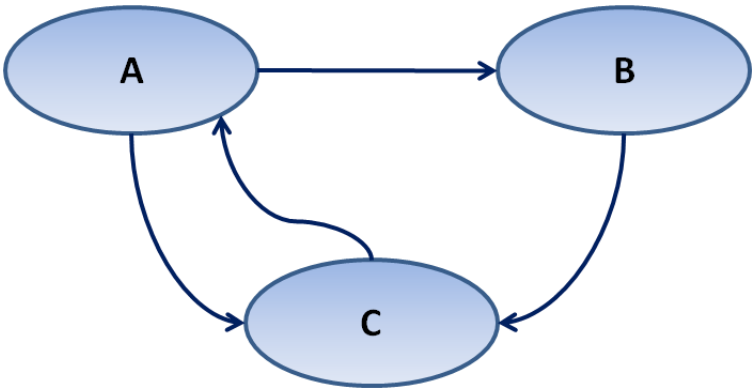
1) Eine vereinfachte Version des PageRankings einer Website ist gegeben durch:

$$PR(u) = c \sum \frac{PR(v)}{N_v}$$

Hierbei entspricht $N_v$ der Anzahl der ausgehenden Links der Website $v$. In der Matrix-Schreibweise:

$$\vec{R} = cA\vec{R}$$

beinhaltet $\vec{R}$ (Eigenvektor zur Matrix A), die PageRankings aller Websites während **c** ein Normierungsfaktor ist.
Berechnen Sie mit dieser vereinfachten Annahme den konvergierten Zustand des folgenden Diagramms. Die Start-PageRankings **PR($v$)** für die Websites **A**,**B** und **C** und der Normierungsfaktor sind jeweils **1**. Die Pfeile stellen die Forward-Links dar.




2) Überlegen Sie sich Diagramme bei denen die oben eingeführte vereinfachte Version der PageRank Berechnung zu Problemen führt. Beschreiben Sie die Problematik.

3) Die endgültige Version sieht wie folgt aus:

$$PR(u) = (1 - c) + c \sum \frac{PR(v)}{N_v}$$

bzw. in der Matrix Schreibweise:

$$\vec{R} = cA\vec{R} + \vec{E}$$

wobei der Vektor $\vec{E}$ aus den Elementen **(1-c)** besteht.
Beschreiben Sie in welchem Zusammenhang die Normierungsgröße **c** bzw. **(1-c)** das Surfverhalten simuliert.
Wie lässt sich das Surfverhalten eines User für $c \to 0$ bzw. $c \to 1$ beschreiben?

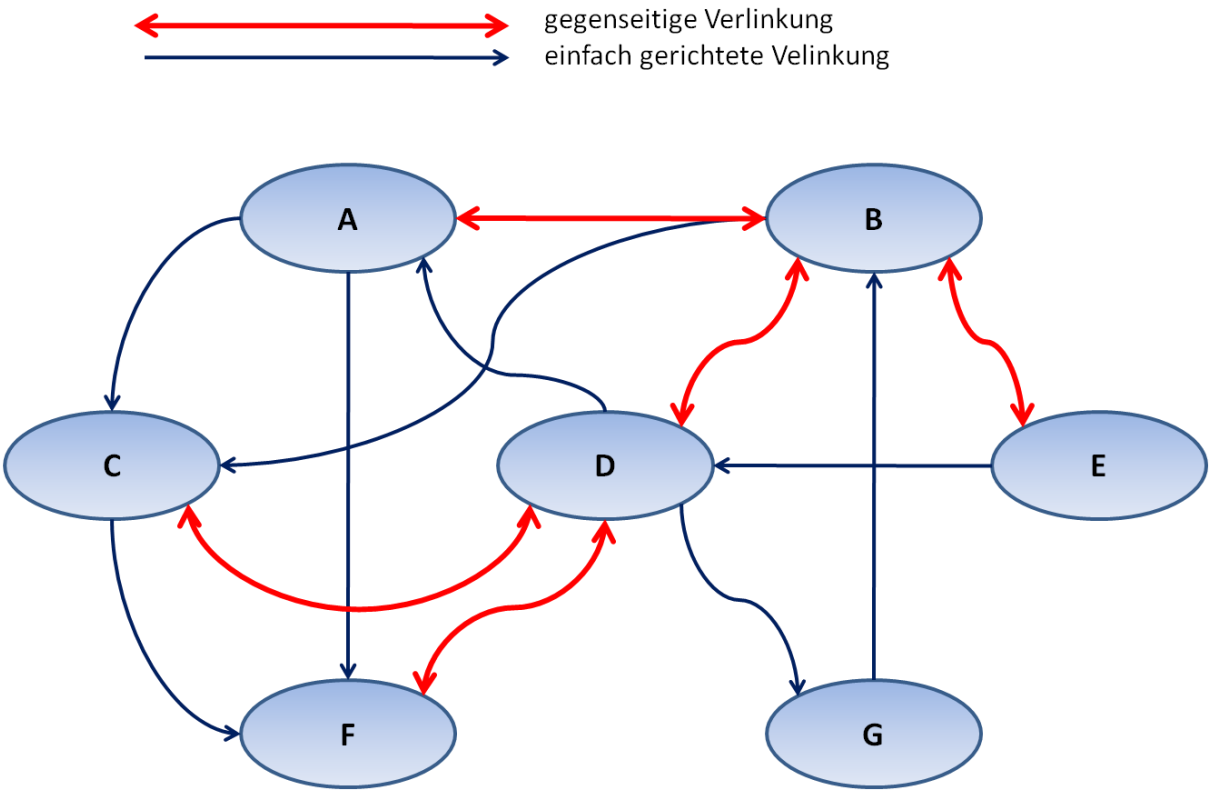
Berechnen Sie erneut den konvergierten PageRank Zustand des ersten Diagramms mit dem modifizierten Algorithmus. Wählen Sie einen Wert für **c = 0.85**.

4) Eine weitere Modifikation:

$$PR(u) = \frac{(1-c)}{n} + c \sum \frac{PR(v)}{N_v}$$

führt dazu, dass die Summe des PageRanks aller Websites genau **1** ist, wobei nun **n** der Anzahl aller Websites entspricht.

Berechnen Sie damit iterativ, basierend auf dem folgenden Diagramm, die Wahrscheinlich eines Surfers eine bestimmte Seite (**A..G**)zu besuchen. Verwenden Sie wieder einen Normierungsfaktor von **c = 0.85** und finden Sie heraus mit welchen PageRank-Startwerten der Algorithmus am schnellsten konvergiert. Wie viele Iterationen sind minimal nötig bis sich alle PageRank-Werte bis auf ein ‰ nicht mehr verändern?




PS: 1998 wurde die iterative PageRank Berechnung mit 75 Millionen URLs in ungefähr fünf Stunden durchgeführt. Eine Konvergenz bei einer annehmbaren Toleranz wurde dabei bei 52 Iterationen erreicht, wobei anzunehmen ist, dass der Algorithmus bei ungefähr 100 Iterationen vollständig konvergiert.
Quelle: *L. Page; S. Brin; R. Motwani; T. Winograd: „The PageRank Citation Ranking: Bringing Order to the Web“*

## Aufgabe 2 – Hadoop

In dieser Aufgabe werden Sie mit Hilfe des Hadoop/MapReduce Frameworks einen größeren Datensatz untersuchen. Es handelt sich hierbei um anonymisierte Daten des Auktionshauses eBay. Detailliert beschrieben sind 8000 Auktionen diverser iPod-Fabrikate:
http://dl.dropbox.com/u/4497643/hadoop/ebay.txt

Die Struktur des Datensatzes ist hier zu finden:
http://dl.dropbox.com/u/4497643/hadoop/struktur.pdf

Finden Sie heraus inwieweit das Zielmerkmal **gms_greater_avg** jeweils von den Variablen **gallery_fee_flag** bzw. **bold_fee_flag** abhängig ist. Gehen Sie dabei wie folgt vor:

1) Die Untersuchung soll auf einer Hadoop Single Node (pseudo-distributed hadoop installation) durchgeführt werden. Hierbei soll „Cloudera Distribution for Hadoop“ (http://www.cloudera.com/hadoop/) verwendet werden. Sie können hierfür ein vorgefertiges VMWare Image (http://www.cloudera.com/downloads/) benutzen oder auf einer gängigen Linux Distribution die erforderlichen Pakete selber einspielen (z.B. in der Amazon EC2 Cloud). Auf den Übungsfolien finden sie eine Anleitung für eine Ubuntu 10.04 Lucid Distribution (http://dl.dropbox.com/u/4497643/hadoop/hadoop.pdf). Eine ausführliche Anleitung ist hier zu finden: https://ccp.cloudera.com/display/CDHDOC/CDH3+Quick+Start+Guide

2) Machen Sie sich mit dem Hadoop Framework und der Umgebung vertraut:
http://www.cloudera.com/wp-content/uploads/2010/01/GettingFamiliar.pdf

3) Schreiben Sie Ihr eigenes MapReduce Programm in Python um die Aufgabenstellung zu lösen. Hierbei könnten Ihnen das folgende Tutorial helfen:
http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/