

Web-Informationssysteme, WS 2009/10

Übungsblatt 5

Besprechung am Di 01.12.2009

Aufgabe 5-1 HITS-Algorithmus

$\bar{\mathbf{h}}^k$  sei der nichtnormalisierte *Hub*-Vektor, der im  $k$ -ten HITS-Update-Schritt entsteht.

$\bar{\mathbf{a}}^k$  sei der nichtnormalisierte *Authority*-Vektor, der im  $k$ -ten HITS-Update-Schritt entsteht.

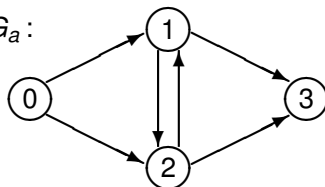
$\mathbf{h}^k$  sei der Vektor, der durch Normalisierung von  $\bar{\mathbf{h}}^k$  entsteht.

$\mathbf{a}^k$  sei der Vektor, der durch Normalisierung von  $\bar{\mathbf{a}}^k$  entsteht.

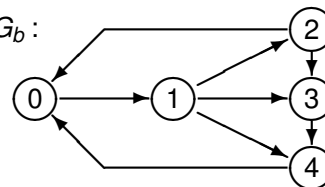
Sei  $\bar{\mathbf{h}}^0 = \bar{\mathbf{a}}^0 = (\underbrace{1, 1, \dots, 1}_{n \text{ Mal}})^T$  der  $n$ -dimensionale Startvektor, also  $\mathbf{h}^0 = \mathbf{a}^0 = \frac{1}{\sqrt{n}}(1, 1, \dots, 1)^T = \begin{pmatrix} \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \vdots \\ \frac{1}{\sqrt{n}} \end{pmatrix}$ .

Berechnen Sie mit dem HITS-Algorithmus alle Vektoren bis  $\mathbf{h}^3$  und  $\mathbf{a}^3$  für:

a)  $G_a$ :



b)  $G_b$ :



Aufgabe 5-2 HITS-Algorithmus

Das Gesamtgewicht eines gerichteten Graphen  $G = (V, E)$  sei  $I(G) = \sum_{\exists(i,j) \in E} h_i + \sum_{\exists(j,i) \in E} a_i$

Dabei sei  $h_i$  der Hubwert und  $a_i$  der Autoritätswert von Knoten  $i$  nach dem HITS-Algorithmus.

Der nebenstehende Graph  $G$  hat drei bipartite Komponenten  $G_1, G_2, G_3$ .

a) Berechnen Sie mit dem HITS-Algorithmus, aber **ohne Normalisierung**, die Vektoren bis  $\bar{\mathbf{h}}^3$  und  $\bar{\mathbf{a}}^3$  für  $G$ .

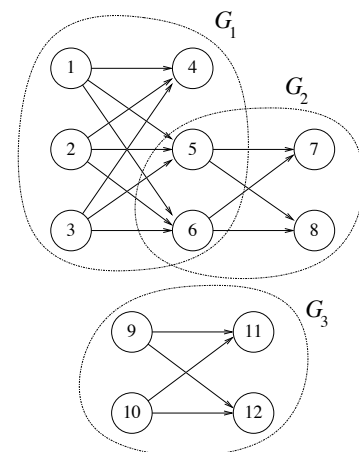
b) Geben Sie zu jedem Schritt  $k$  die Gesamtgewichte  $I^k(G_1), I^k(G_2), I^k(G_3)$  der drei Komponenten an.

c) Welche der drei bipartiten Komponenten  $G_1, G_2, G_3$  wird asymptotisch das höchste Gesamtgewicht erhalten? Woran liegt das?

d) In  $G$  werde die Kante  $(3, 9)$  eingefügt. Welche *qualitativen* Auswirkungen hat das auf die Gesamtgewichte der drei Komponenten?

e) Zusätzlich werde die Kante  $(10, 3)$  eingefügt. Welche *qualitativen* Auswirkungen hat das auf die Gesamtgewichte der drei Komponenten?

$G$ :



**Aufgabe 5-3 MapReduce: Problemdarstellung**

Gegeben sei das Zugriffs-Log eines Web-Servers, also eine Liste von Zeitpunkt-URL-Paaren. Sein Typ ist  $(TimePoint * URL) list$  in SML-Notation bzw.  $[(TimePoint, URL)]$  in Haskell-Notation. Jede *input*-Liste der folgenden MapReduce-Aufgaben entsteht durch Extraktion gewisser Paare aus diesem Zugriffs-Log.

- a) Aus dem Zugriffs-Log werden alle Paare mit URL  $u$  für ein gegebenes  $u$  extrahiert. Gesucht ist die Anzahl der Zugriffe von  $u$  pro Zeitintervall  $i$ . Für eine Lösung mit dem MapReduce-Modell sei definiert:

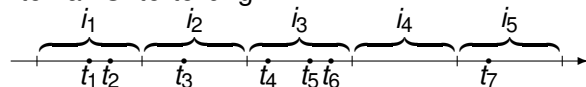
$$\begin{aligned} \text{map} : (TimePoint, URL) &\rightarrow [(TimeInterval, Int)] & \text{reduce} : (TimeInterval, [Int]) &\rightarrow [Int] \\ (t, u) &\mapsto [(i, 1)] \text{ für das } i \text{ mit } t \in i & (i, \ell) &\mapsto [length(\ell)] \end{aligned}$$

Geben Sie *input*-Liste, *intermediate*-Liste und *output*-Liste des MapReduce-Prozesses für folgende Beispieldaten an:

Extrahierte Paare:

$[(t_1, u), (t_2, u), (t_3, u), (t_4, u), (t_5, u), (t_6, u), (t_7, u)]$ .

Zeitintervall-Unterteilung:



- b) Das gesamte Zugriffs-Log wird betrachtet. Gesucht sind die URLs, die mindestens 3 Mal darin vorkommen.

Definieren Sie *map* und *reduce* dafür geeignet. Geben Sie *input*-Liste, *intermediate*-Liste und *output*-Liste des MapReduce-Prozesses für folgende Beispieldaten an:

Extrahierte Paare:  $[(t_1, u_1), (t_2, u_2), (t_3, u_1), (t_4, u_2), (t_5, u_3), (t_6, u_1), (t_7, u_3), (t_8, u_3)]$ .

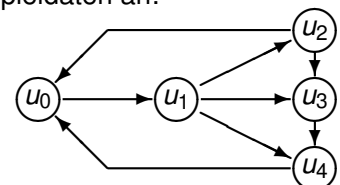
- c) Aus dem Zugriffs-Log werden alle Paare mit dem ersten Vorkommen pro URL extrahiert. Gesucht ist für jede Seite, deren URL im Zugriffs-Log vorkommt, von welchen Seiten auf die gegebene verwiesen wird.

Definieren Sie *map* und *reduce* dafür geeignet. Geben Sie *input*-Liste, *intermediate*-Liste und *output*-Liste des MapReduce-Prozesses für folgende Beispieldaten an:

Extrahierte Paare:

$[(t_0, u_0), (t_1, u_1), (t_2, u_2), (t_3, u_3), (t_4, u_4)]$ .

Verweise:

**Aufgabe 5-4 MapReduce: Berechnung von TF\_IDF**

Berechnen Sie mit dem MapReduce-Modell den Wert  $TF\_IDF(D_i, T_j)$  für jedes Dokument  $D_i$  und jeden Term  $T_j$  in einer gegebenen Dokumentensammlung.

Die Anzahl  $N$  der Dokumente sei fix und gegeben, alle anderen Werte müssen berechnet werden. Jedes Dokument  $D_i$  habe einen Namen  $id_i$  und sei gegeben als Liste  $d_i$  von Termen.

Die *input*-Liste für den MapReduce-Prozess ist also  $[(id_1, d_1), \dots, (id_N, d_N)]$ . Seine *output*-Liste ist  $[\dots, ((id_i, t_j), tfidf_{ij}), \dots]$  mit  $tfidf_{ij} = TF\_IDF(D_i, T_j)$  für jedes Dokument  $D_i$  und jeden Term  $T_j$ .

*Hinweis:* es werden mehrere hintereinandergeschaltete MapReduce-Schritte benötigt.