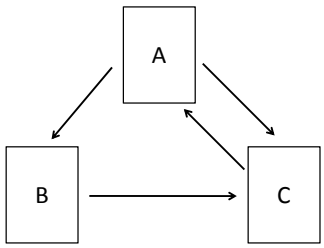
PageRank

Die Hauptidee hinter Google's PageRank Paradigma [1] ist die Propagation der Wichtigkeit von Webseiten durch Hyperlinks. Jeder Seite $p \in P$ (P = Menge aller betrachteten Seiten) ist eine *Rangzahl* $r(p)$ zugeordnet. Der aus diesen Rangzahlen gebildete Vektor r heisst Rangvektor. Weiter sei $l(p)$ die Menge aller Seiten, welche von p referenziert werden, und $|l(p)|$ sei die Anzahl Elemente in dieser Menge. Um nun den Page Rank für alle Seiten in P zu berechnen, wird die folgende Iteration solange fortgeführt, bis ein Fixpunkt erreicht ist. Dabei bezeichnet der hochgestellte Index (^{i}) den Iterationsschritt Nr. i.

$$\forall p \in P : r^{(i)}(p) = (1-\alpha) \cdot \tau(p) + \alpha \cdot \sum_{\forall q \in P : p \in l(q)} \frac{r^{(i-1)}(q)}{|l(q)|} \tag{1}$$

Die obige Formel (1) besteht aus zwei Teilen, der *Jump-Komponente* (links des Pluszeichens) und der *Walk-Komponente* (rechts des Pluszeichens), gewichtet mit $(1-\alpha)$ bzw. α (gewöhnlich $\alpha = 0.85$). $\frac{r^{(i-1)}(q)}{|l(q)|}$ ist die durch die Seite q im i-ten Iterationsschritt an die durch sie referenzierten Seiten propagierte Wichtigkeit. Intuitiv heisst dies, dass ein ,,Random"-Surfer mit Wahrscheinlichkeit α einem von der aktuellen Seite ausgehenden Link folgt und mit Wahrscheinlichkeit 1 - α zu eine Zufallsseite springen wird. Der Hauptzweck von α ist Konvergenzgarantie und Vermeidung von ,,Rangsenken". $\tau(p)$ ist der Vektor, der zur Anfangsverteilung von Rängen gehört. Sie können in dieser Uebung annehmen, dass $\tau(p) = \frac{1}{|P|}$ gilt.

Das folgende kleine Beispiel illustriert den PageRank Algorithmus:



Betrachten Sie ein kleines Web bestehend aus nur drei Seiten A, B und C, wobei die Seite A die Seiten B und C referenziert, die Seite B die Seite C und die Seite C die Seite A. Dann erhalten wir die folgenden Gleichungen für die PageRank Berechnung:

$$r(A) = 0.15/3 + 0.85r(C)$$
$$r(B) = 0.15/3 + 0.85(r(A)/2)$$
$$r(C) = 0.15/3 + 0.85(r(A)/2 + r(B))$$

Dieses System hat eine eindeutige Lösung

$$r(A) = 686/1769 = 0.387789712$$
$$r(B) = 380/1769 = 0.214810627$$
$$r(C) = 703/1769 = 0.397399661$$

In dieser Uebung sollen Sie einen auf dem Map-Reduce Muster basierenden Algorithmus angewandt auf das PageRank Problem [1] konzipieren und implementieren. Stützen Sie Ihre Implementation auf die folgende Schnittstelle ab:

```
static IEnumerable<KeyValuePair<Key, Value>>
  MapReduce<Source, Key, Value>(
    IEnumerable<Source> source,
    Func<Source, IEnumerable<KeyValuePair<Key, Value>>> map,
    Func<Key, IEnumerable<Value>, Value> reduce
  )
```

Hinweis: Für die Erzeugung des Beispiel Webs steht ein Template zur Verfügung. Kopieren Sie es von

```
https://svn.inf.ethz.ch/svn/gutknecht/pp/shared/hw04/template
```

in Ihren Übungsfolder und lesen Sie die Anweisungen in `Program.cs`.

Referenzen

[1] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Stanford InfoLab. 1999.

Abgabe

Bitte speichern Sie ihre Lösung als Visual Studio Projekt in folgendem SVN Verzeichnis ab. Achten Sie darauf, dass Ihr Assistent das Projekt öffnen und kompilieren kann.

```
https://svn.inf.ethz.ch/svn/gutknecht/pp/<Assistent>/<nethz id>
```

Die Namen der Assistenten lauten `carnecky`, `cebulla`, `ernst`, `gehr`, `gruebel`, `kao`, `kuendig`, `leuenberger`, `roth` und `widmer`. Falls Sie Probleme beim Zugriff auf das Verzeichnis haben, wenden Sie sich bitte an Florian Negele (`negelef@inf.ethz.ch`). Geben Sie Erklärungen und Abschätzungen in separaten Files oder direkt als Kommentar im Quellcode ab. Ihr Assistent wird dann bei der Korrektur ein File namens `feedback.txt` erstellen. **Bitte checken Sie keine Binärdateien (.dll, .exe, .obj, .pdb usw.) ein.**