Szegedi Tudományegyetem
Természettudományi és Informatikai Kar

Bolyai Intézet
Sztochasztika tanszék

A PageRank és kiszámolása

Szakdolgozat

Készítette: Gáspár Tamás
Matematika BSc hallgató

Témavezető: Dr. Kevei Péter
Egyetemi docens
Sztochasztika tanszék

Szeged, 2019

Tartalomjegyzék

1. Bevezető **1**
1.1. A PageRank algoritmus története 1
1.2. PageRank program 1

2. Alapfogalmak és definíciók **2**
2.1. Linkek, web és a lógó oldalak 2
2.2. A PageRank definíciója 3
2.3. Linkmátrix 4
2.4. Webekhez rendelt Markov-láncok 5

3. A sajátértékprobléma **6**
3.1. Az 1 sajátérték 6
3.2. A legnagyobb sajátérték 7
3.3. A hatványiteráció 8

4. Konvergencia a PageRank vektorhoz **8**
4.1. Gondot okozó tényezők 8
4.2. A konvergencia garantálása 9
4.3. A Google mátrixhoz rendelt Markov-lánc 10

5. A Google mátrix és az α paraméter **11**
5.1. A Google mátrix tulajdonságai 11
5.2. Az α paraméter kérdése 11
5.3. A hatványiteráció konvergenciájának sebessége 12
5.4. α mint sajátérték 13

6. Algoritmus a PageRank meghatározására **16**
6.1. Algoritmus általános web esetén 16
6.2. Példa 17

1. Bevezető

Dolgozatom témája a PageRank, és az ennek számolásához használt algoritmus, melynek legfőbb alkalmazási területe az internetes weboldalak bizonyos szempontok szerinti rangsorolása.

Ez a fejezet röviden ismerteti a PageRank történetét és megemlít egy, a PageRank számolásához használható, általam készített programot. A következő fejezet a PageRank tárgyalásához szükséges alapvető fogalmakat és definíciókat vezet be.

1.1. A PageRank algoritmus története

A PageRank algoritmust Larry Page és Sergei Brin alkották meg 1998-ban, a Standford Egyetem hallgatóiként. Arra kerestek megoldást, hogy miként lehet az akkoriban robbanásszerű növekedésnek induló internet weboldalait rangsorolni, mivel látszott, hogy az akkor alkalmazott keresőmotorok erre már hamarosan nem lesznek képesek.

Úgy gondolták, hogy a rangsorolás alapja az internet hiperlink struktúrája kell hogy legyen. Feltették, hogy ha egy oldal linkel egy másikra, az kifejezi azt, hogy az oldal készítői megbíznak a linkelt oldalban, ezért az algoritmusukat úgy építették fel, hogy egy oldal fontossága (ezt szintén PageRanknak nevezik) attól függ, hogy mennyi és milyen fontos oldalak linkelnek rá.

Larry Page és Sergei Brin találmánya olyan jól alkalmazhatónak bizonyult, hogy Google néven saját vállalkozást alapítottak. A cég keresőmotorjának alapja máig a PageRank algoritmus, skálázhatóságát mutatja, hogy a Google ma már több mint 130 billió weboldalt indexel.

1.2. PageRank program

Dolgozatomhoz grafikus felhasználói felülettel ellátott asztali alkalmazást is készítettem, Java nyelven. Ez a program képes előre megadott oldalszámú webet különböző paraméterek alapján véletlenszerűen generálni, majd ehhez PageRankot számolni. A 150 oldalnál kisebb méretű webekhez tartozó mátrixokat meg is tudja jeleníteni.

A program és annak forráskódja is letölthető a következő oldalról:

https://github.com/Gtomika/PageRank/releases/tag/v4.1

A futtatáshoz a számítógépen telepítve kell hogy legyen a Java 8-as, vagy újabb verziója.

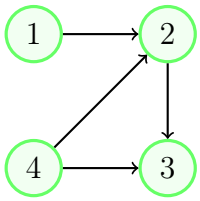
2. Alapfogalmak és definíciók

2.1. Linkek, web és a lógó oldalak

Ahhoz hogy PageRankról beszélhessünk, először a web fogalmát kell bevezetni.

1. Definíció: ***Linkhalmaz.*** Legyen V a weboldalak halmaza. Ekkor $L \subset V \times V$ **linkhalmaz**ban $(v_1, v_2) \in V$ pontosan akkor van benne ha v_1 oldalon van v_2-re mutató link.

2. Definíció: ***Web.*** Legyen V a weboldalak halmaza, L pedig az ehhez tartozó linkhalmaz. Ekkor a web egy irányított gráf, melynek csúcsai V elemei, élei pedig L elemei, ahol ha $v_1, v_2 \in V$ és $(v_1, v_2) \in L$, akkor az él v_1-ből v_2-be mutat.



1. ábra. Egy 4 oldalból álló web

A linkhalmaz definíciója ugyan megengedi azt, hogy egy oldal önmagára linkeljen, mert bár egy valós web esetén ez lehetséges, de a PageRank algoritmus esetén nem akarjuk, hogy az ilyen linkek is beleszámítsanak a rangsorolásba. Ezeket a linkeket ezért egyszerűen elhagyjuk.

Az is megfigyelhető, hogy a definíciókban a linkeknek nincsen multiplicitása, egy oldal vagy linkel egy másikra, vagy nem. Ez azért van, hogy a rangsorolást ne lehessen olyan egyszerűen manipulálni, hogy bizonyos oldalak sok linket tartalmaznak egy másikra.

Előfordulhat olyan web, ahol bizonyos oldalakról nincsenek kimenő linkek. Az ilyen oldalakat lógó oldalnak nevezzük (azokat a linkeket amelyek rájuk

mutatnak pedig lógó linkeknek). A későbbiekben ezek az oldalak problémákat okoznak, ezért megadunk egy módszert, mellyel a ezeket az oldalakat el lehet tüntetni: minden lógó oldalt helyettesítünk egy, az összes másik oldalra linkelővel (ebben a kivételes esetben megengedjük az önmagára linkelést is).

Egy lógó oldallal (4) rendelkező web, és ugyanez a web, helyettesített oldallal.

A helyettesítés mögötti heurisztika az, hogy ha egy böngésző egy olyan oldalra érkezik, ahonnan a linkeken keresztül nem tud továbbmenni, akkor véletlenszerűen, egyenletes eloszlás szerint választ az összes oldal közül.

2.2. A PageRank definíciója

Egy oldal fontossága azon múlik, hogy mennyi oldal linkel rá, és hogy ezek milyen fontosak. A linkelő oldalak fontosságára azért van szükség, mert enélkül egy oldaltulajdonos tudná úgy növelni a weboldalának fontosságát, hogy rengeteg oldalt hoz létre, melyek mind linkelnek egymásra és a saját oldalára (ezt link farmnak nevezik [8]).

Egy webet el lehet úgy is képzelni, mint az oldalak demokráciáját ahol minden oldalnak egy szavazata van és ezt a szavazatot (és még a kapott szavazatokat is) továbbosztja úgy, hogy linkel a többi oldalra.

3. Definíció: ***Weboldal PageRankja.*** Legyen adott egy web, V az oldalak, L a linkek halmaza. Legyen $v_i \in V$ oldal PageRankja $r(v_i)$, az oldalról kimenő linkek száma pedig $|v_i|$.

Jelölje $B_i \subset V$ azon oldalak halmazát amelyen linkelnek v_i-re, azaz

$$B_i = \{v_j \in V : \exists l \in L, \quad l = (v_j, v_i)\}.$$

Ekkor v_i oldal PageRank-ja definíció szerint:

$$r(v_i) = \sum_{v_j \in B_i} \frac{r(v_j)}{|v_j|}.$$

A definícióban megjelenik az is, hogy egy oldalnak mennyi kimenő linkje van. Minél több oldalra linkel, annál kevesebbet fog számítani az ő linkjének értéke. Ez ellensúlyozza a már említett link farmokat.

A szummában szereplő oldalak egyikénél sem lehet a kimenő linkek száma nulla, mert mindegyik oldal eleme a B_i halmaznak, azaz legalább v_i-re linkelnek. Előfordulhat, hogy egy oldalra nincs egyetlen link sem. A definícióban ilyenkor egy üres összeg szerepel, azaz az oldal rangja 0.

A PageRank definíciója tehát rekurzív. Egy oldal rangjának meghatározásához minden rá linkelő oldal rangját ismernünk kell. Ha a webünk n db oldalt tartalmaz, akkor a definíció meghatároz n db lineáris egyenletet.

A cél az, hogy olyan algoritmust adjunk meg, amely egyrészt pontosan meghatározza minden oldal PageRankját, másrészt nagyon nagy n-re is hatékonyan működik, mind idő-, mind tárigény szempontjából.

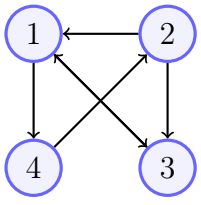
2.3. Linkmátrix

Egy web linkmátrixa az oldalak közötti kapcsolatokat reprezentálja mátrixos formában.

4. Definíció: ***Linkmátrix.*** Legyen adott egy web ahol az oldalak halmaza V. Ennek linkmátrixa legyen $A = (a_{i,j}) \quad i, j = 1, \ldots, |V|$, ahol

$$a_{i,j} = \begin{cases} \frac{1}{|v_i|}, & \text{ha } (v_i, v_j) \in L, \\ 0, & \text{egyébként.} \end{cases}$$

Tehát a linkmátrix sorai kifejezik azt, hogy egy oldal mennyi és melyik másik oldalakra linkel. Megfigyelhető, hogy amennyiben a web nem tartalmaz lógó oldalt, úgy a linkmátrix sztochasztikus lesz, azaz sorainak összege mindig 1, amennyiben pedig tartalmaz ilyen oldalt, a 2.1 alfejezetben megadott helyettesítéssel kapott mátrix már sztochasztikus lesz.



$$\begin{pmatrix} 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Példa: egy 4 oldalból álló web és linkmátrixa

2.4. Webekhez rendelt Markov-láncok

Vegyünk egy lógó oldalt nem tartalmazó webet. Legyen adott egy úgynevezett véletlen szörföző, aki minden lépésben egyenletes eloszlás szerint választ a jelenlegi oldalon lévő linkek közül és továbbhalad a választott link által mutatott oldalra. A kezdeti eloszlás szintén legyen egyenletes.

Ehhez a modellhez rendelhetünk egy Markov-láncot, melynek állapotai a V oldalhalmaz elemei, az átmenetvalószínűségeket pedig a linkmátrix adja meg:

$$p_{i,j} = (a_{i,j}) \quad i, j = 1, \ldots, |V|,$$

ahol $a_{i,j}$ az A linkmátrix $i.$ sorának $j.$ eleme.

A Markov-lánchoz tartozó átmenetmátrix tehát éppen a linkmátrix lesz. A következő fejezetben igazoljuk, hogy a keresett PageRank vektor megoldása az $\mathbf{x} = \mathbf{x}A$ lineáris egyenletrendszernek. Látszik, hogy ez éppen a Markov-lánc invariáns eloszlása.

Ez logikus, hiszen az invariáns eloszlás mutatja, hogy hosszú távon az idő mekkora részét tölti a véletlen szörföző az állapotokban, tehát azok az oldalak lesznek a legfontosabbak, ahol várhatóan a legtöbb időt töltjük.

Megjegyzés: Nem minden webhez rendeletünk egy ilyen Markov-láncot. A lógó oldalakat tartalmazó webek linkmátrixában vannak csupa nulla sorok, melyek elrontják a mátrix sztochasztikusságát, de a 2.1 alfejezetben megadott helyettesítéssel kapott webek már megfelelőek.

A linkmátrix 1 sajátértékének multiplicitása és a webhez rendelt Markov-lánc között kapcsolat van. Itt kimondunk egy tételt [5], amelynek segítségével meghatározhatjuk az 1 sajátérték multiplicitását a linkmátrix (mint átmenetmátrix) bizonyos tulajdonságaiból. Ez a későbbiekben (5.4. fejezet) hasznos lesz.

5. Definíció: ***Zárt részhalmaz.*** Egy átmenetmátrix elemeinek $H \subseteq V$ kommunikációs osztálya zárt, ha minden $i \in H$ és $j \notin H$ állapotok esetén a $p_{i,j}$ átmenetvalószínűség nulla.

6. Definíció: ***Irreducibilis zárt részhalmaz.*** Egy átmenetmátrix elemeinek $H \in V$ részhalmaza irreducibilis zárt, hogyha zárt és egyetlen valódi részhalmaza sem zárt.

1. Tétel. *Egy átmenetmátrix esetén az 1 sajátérték multiplicitása megegyezik a mátrixban lévő irreducibilis zárt részhalmazok számával.*

3. A sajátértékprobléma

Kapcsolat van az oldalak PageRankjának definíciójából kapott egyenletrendszer és a linkmátrix között. Ennek segítségével a PageRank definíciójából kapott egyenletrendszer felírható a következő mátrixegyenletként:

$$\mathbf{x} = \mathbf{x}A,$$

ahol $\mathbf{x}$ az oldalak rangjait tartalmazó sorvektor. Ezt átalakítva a következőt kapjuk:

$$\mathbf{x}^T = A^T\mathbf{x}^T.$$

Ebből mátrixegyenletből látszik, hogy a keresett vektor az A^T mátrix 1 sajátértékéhez tartozó sajátvektor. Mivel egy sajátvektor bármilyen nem nulla skalárszorosa is sajátvektor ezért az egyértelműség kedvéért tegyük fel, hogy az általunk keresett vektor az, ahol a komponensek összege 1. Ez azért hasznos, mert az előző fejezetben kiderült, hogy a PageRank vektor invariáns eloszlás is, ahol a komponensek összege éppen 1.

Megjegyzés: Ha webnek az internet egy sok oldalból álló részhalmazát választjuk (márpedig a PageRankokat ilyen esetre akarjuk kiszámolni), akkor a linkmátrix nagyon ritka lesz, azaz szinte az összes eleme 0. Ennek oka, hogy az egy oldal által linkelt további oldalak száma elenyésző az összes oldal számához képest. A linkmátrix ritkasága egy számolási szempontból nagyon kedvező tulajdonság.

3.1. Az 1 sajátérték

Felmerül a kérdés, hogy minden A^T mátrixnak sajátértéke-e az 1. Ehhez elég belátni, hogy az A linkmátrixnak az 1 mindig sajátértéke, mivel egy mátrixnak és transzponáltjának a sajátértékei megegyeznek.

2. Tétel. *Az 1 sajátértéke minden lógó oldalt nem tartalmazó web linkmátrixának.*

Bizonyítás. Legyen A egy tetszőleges, lógó oldalt nem tartalmazó, n oldalból álló web linkmátrixa. Ekkor az i. sorhoz ($i = 1, 2, \ldots, n$) tartozó sorösszeg $|v_i|\frac{1}{|v_i|} = 1$, azaz minden ilyen mátrix sztochasztikus.

Legyen $\mathbf{e}$ az n komponensű oszlopvektor, melynek minden komponense 1. Ekkor az

$$A\mathbf{e} = \mathbf{e}$$

egyenlet teljesül, mert az eredményvektor i. komponense ($i = 1, 2, \ldots, n$)

$$\sum_{j=1}^{n} 1 a_{i,j} = 1,$$

a mátrix sztochasztikus tulajdonsága miatt. Tehát bármely linkmátrixnak az $\mathbf{e}$ vektor sajátvektora, 1 sajátértékkel. □

Ugyan bármely mátrixnak és transzponáltjának sajátértékei megegyeznek, de az nem igaz, hogy ezen sajátértékekhez tartozó sajátvektorok is azonosak, ezért a fenti tételben bevezetett $\mathbf{e}$ vektor nem biztos, hogy megoldása lesz a PageRank definíciója által meghatározott egyenletrendszernek. Ezzel a megoldás egyébként is használhatatlan az oldalak rangsorolásában, mivel minden oldalnak azonos rangot ad.

A lógó oldalakat tartalmazó webek esetén az 1 nem lesz mindig sajátértéke az A^T mátrixnak (vannak 0 összegű sorok, amik a fenti bizonyítást elrontják), azonban figyeljük meg, hogy ha elvégezzük a 2.1 alfejezetben megadott helyettesítést akkor a kapott mátrix már sztochasztikus lesz, és erre már érvényes az előbbi tétel.

Tehát beláttuk, hogy minden ilyen mátrixnak vagy sajátértéke az 1, vagy egy egyszerű helyettesítéssel el lehet érni, hogy az legyen.

3.2. A legnagyobb sajátérték

3. Tétel. *Minden lógó oldalt nem tartalmazó linkmátrix bármely λ sajátértékre $|\lambda| \leq 1$.*

Bizonyítás. A sajátértékek abszolút értékének felülről becsléséhez alkalmazzuk a Gershgorin körtételt. Eszerint a mátrix összes sajátértékének benne kell lennie legalább egy Gershgorin körben. A Gershgorin körtétel ugyan komplex mátrixokra is érvényes, de a linkmátrix csak valós elemeket tartalmazhat. A köröket meghatározó halmazok:

$$K_i = \{r \in \mathbb{R} : |r - a_{i,i}| \leq R_i\} \quad i = 1, 2, \ldots, n,$$

ahol R_i az i. sorösszeg és $a_{i,i}$ az i. sorban lévő főátlóbeli elem.

Már korábban beláttuk, hogy a feltételeknek eleget tévő linkmátrixok sztochasztikusak, azaz a sorösszeg minden sornál egy. A linkhalmaz 2.1 alfejezetben szereplő definíciójában nem engedjük meg azt, hogy egy oldal önmagára linkeljen, ezért $a_{i,i}$ minden sorra 0.

Ezen állítások miatt a köröket meghatározó halmazok a következőre egyszerűsödnek minden sor esetén:

$$K_i = \{r \in \mathbb{R} : |r| \leq 1\} \quad i = 1, 2, \ldots, n$$

Ebből következik, hogy bármely λ sajátértékre $|\lambda| \leq 1$-nek teljesülnie kell. □

Beláttuk, hogy az egy minden esetben sajátérték, és azt is hogy ennél (abszolút értékben) nem is lehet nagyobb sajátérték. Ebből azonban nem következik az, hogy domináns is lenne. Előfordulhat olyan eset is, hogy a -1 is sajátérték, vagy az 1 többszörös multiplicitással szerepel.

3.3. A hatványiteráció

Ha sok oldalból álló webek esetén akarjuk meghatározni a rangokat, akkor a pontos megoldást adó módszerek nagyon lassúak, ezért közelítő, iteratív módszerre van szükség. Ideális választás a hatványiteráció alkalmazása az A^T mátrixra, mert ebben az esetben elég tárolni mindössze a linkmátrix transzponáltját, és a PageRank vektor éppen aktuális iterációját a számoláshoz. A hatványiteráció általános alakja

$$x^{i+1} = Mx^i,$$

ahol $M \in \mathbb{R}^{n \times n}$ és $x^0 \in \mathbb{R}^n$.

A hatványiteráció azonban a domináns sajátértékhez tartozó sajátvektorhoz konvergál. Ha ilyen nincsen, akkor nem tudjuk garantálni a konvergenciát. Ahhoz, hogy alkalmazhassuk az egyhez tartozó sajátvektor megtalálására, vagy be kell látni, hogy az 1 mindig domináns sajátérték, vagy úgy kell módosítani a linkmátrixon, hogy az legyen.

4. Konvergencia a PageRank vektorhoz

4.1. Gondot okozó tényezők

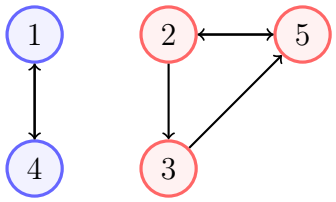
Amikor egy nagy webhez tartozó PageRank vektort szeretnénk közelíteni, fontos tudni, hogy az iteráció úgy fog viselkedni ahogyan várjuk. A következő lehetőségeket kell ellenőrizni:

1. Biztosan konvergálni fog-e a hatványiteráció?

2. Ha igen, akkor az indítási helytől függetlenül mindig ugyanabba a vektorba?

Egyértelmű, hogy az indulási helytől független, biztos konvergenciát a 3.2 alfejezet végén említett szélsőséges esetek ronthatják el [10].

Ezek az eseten olyan webek esetén fordulnak elő, amelyek valójában több egymástól független (azaz nincsenek közöttük linkek) webből állnak össze [2].



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Két egymással nem érintkező "szubwebből" álló web és linkmátrixa.

A fenti ábrán látható web linkmátrixának sajátértékei között szerepel a -1 és az 1 is, kétszeres multiplicitással.

4.2. A konvergencia garantálása

A domináns sajátérték (és ezzel együtt a konvergencia) létezésének garantálásához használjuk fel a **Perron–Frobenius tételt**:

4. Tétel. *Ha egy* $A = (a_{i,j}) \in \mathbb{R}^{n\times n}$ *mátrix minden eleme pozitív, akkor biztosan létezik domináns sajátérték, azaz ha a sajátértékek* $\lambda_1, \lambda_2, \ldots, \lambda_n$, *akkor*

$$\exists r \in \{1, 2, \ldots, n\} : \quad \forall i \neq r \quad |\lambda_i| < \lambda_r.$$

A tétel értelmében tehát az egyértelmű PageRank vektor létezéséhez annyit kell tenni, hogy a linkmátrixot úgy módosítjuk, hogy csak pozitív elemei legyenek.

7. Definíció: ***Google mátrix.*** Legyen $A \in \mathbb{R}^{n\times n}$ egy lógó oldalt nem tartalmazó web linkmátrixa, S pedig egy egy olyan $n \times n$-es mátrix, melynek minden eleme $\frac{1}{n}$. Ekkor a Google mátrix definíció szerint

$$G := \alpha A + (1 - \alpha)S, \qquad \alpha \in [0, 1]$$

Látható, hogy a Google mátrix értéke függ az α paramétertől, de majdnem minden esetben (az $\alpha = 1$-et kivéve) a kapott mátrix minden eleme biztosan pozitív, azaz létezik domináns sajátérték és a hatványiteráció indulási helytől függetlenül az ehhez tartozó sajátvektorhoz fog konvergálni.

$$A = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 1 & 0 \end{pmatrix}, S = \begin{pmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}, G = \begin{pmatrix} 0{,}038 & 0{,}463 & 0{,}463 & 0{,}038 \\ 0{,}321 & 0{,}038 & 0{,}321 & 0{,}321 \\ 0{,}321 & 0{,}321 & 0{,}38 & 0{,}321 \\ 0{,}038 & 0{,}038 & 0{,}888 & 0{,}038 \end{pmatrix}$$

Egy web linkmátrixa, az S mátrix és az $\alpha = 0,7$ paraméterrel kapott Google mátrix.

4.3. A Google mátrixhoz rendelt Markov-lánc

A linkmátrixhoz hasonlóan a Google mátrixhoz is rendelhető egy Markov-lánc ez is sztochasztikus mátrix (ennek rövid igazolása a következő fejezet elején található). Az ehhez a mátrixhoz tartozó Markov-lánc néhány tulajdonságát a következő tétel mondja ki.

5. Tétel. *A Google mátrixhoz tartozó Markov-lánc irreducibilis és aperiodikus.*

Bizonyítás. Mindkét tulajdonság következik abból, hogy a Google mátrixot úgy konstruáltuk, hogy egy elem se lehessen benne nulla. Az irreducibilitás azt jelenti, hogy bármely állapotból bármely másikba el lehessen jutni, és az, hogy nincs nulla az átmenetmátrixban éppen ezt jelenti.

Az aperiodikusságot az garantálja, hogy a fentiek miatt a főátlóbeli elemek sem nullák, ezért a Markov-láncnak nem lehet semmilyen egytől különböző periódusa. □

Megjegyzés: *(A helyettesítés mögötti heurisztika.)*
Tekintsük a webhez rendelt Markov-láncot. Ezt úgy módosítjuk, hogy a véletlen szörföző most már nem csak a linkeken keresztül juthat el a következő oldalra, hanem $1 - \alpha$ valószínűséggel egy egyenletes eloszlás szerint választott véletlen oldalra ugrik. Ebben a modellben már nem lehetnek teljesen különálló oldalhalmazok, a véletlen szörföző bárhonnan eljuthat bárhova.

Az így kapott Markov-lánc átmenetmátrixa éppen a Google mátrix lesz, ahol az α paraméter dönt arról, hogy mekkora valószínűséggel követjük a linkeket és mekkorával ugrunk véletlen oldalra.

5. A Google mátrix és az α paraméter

5.1. A Google mátrix tulajdonságai

A Google mátrix szintén sztochasztikus lesz, bármely α paraméter esetén. A linkmátrixról (A) és a csupa $1/n$ komponensből álló (S) mátrixokról tudjuk, hogy sztochasztikusak, ezért αA és $(1-\alpha)S$ mátrixok sorösszegei rendre α és $(1-\alpha)$. A Google mátrix esetén így a sorok összege $\alpha + (1-\alpha) = 1$ lesz.

A hatványiterációt nem a Google mátrixra, hanem a transzponáltjára kell alkalmaznunk, ami a linkmátrix és az S mátrix segítségével a

$$G^T = (\alpha A + (1-\alpha)S)^T = \alpha A^T + (1-\alpha)S^T$$

alakban írható fel.

A 2.3 alfejezetben említettük, hogy a linkmátrix általában egy ritka mátrix és hogy ez számítási szempontból nagyon előnyös. A Google mátrix ennek éppen az ellentéte, nemhogy sűrű, de biztosan nem tartalmaz egyetlen 0 elemet sem, ugyanis ez szükséges ha a hatványiteráció konvergenciáját a Perron–Frobenius tétellel akarjuk garantálni. Sűrű mátrixok esetében azonban a mátrixszorzás sokkal lassabb mint ritkáknál.

A Google mátrix sűrűségéből adódó számítási problémát ki lehet küszöbölni. Ha hatványiteráció i.-ik lépése

$$\mathbf{x}^{i+1} = G^T\mathbf{x}^i,$$

ahol $\mathbf{x}$ a PageRank oszlopvektor, akkor ez ekvivalens a következő alakkal:

$$\mathbf{x}^{i+1} = \alpha A^T\mathbf{x}^i + \frac{(1-\alpha)}{n}\mathbf{x}^i.$$

Először tehát A^T skalár szorosával szorozzuk a PageRank vektor jelenlegi iterációját, majd ezután adjuk hozzá a csupa $\frac{1-\alpha}{n}$ elemekből álló vektort. Ekkor az elvégzett mátrixszorzásban már nem szerepel a sűrű Google mátrix, de az iteráció eredménye ugyanaz, mintha azzal szoroztunk volna.

5.2. Az α paraméter kérdése

Az előző fejezetben láttuk, hogy szinte bármilyen $\alpha \in [0,1]$ választás esetén (az $\alpha = 1$ esetet kivéve) garantáljuk a hatványiteráció konvergenciáját, de

felmerül a kérdés, hogy van-e egyéb jelentősége ennek a paraméternek, és ha igen, akkor hogyan válasszuk meg.

A 4.3 alfejezetben megadott heurisztikából kiderül, hogy α nem lesz más, mint annak a valószínűsége, hogy a véletlen szörföző követi a linkeket, ahelyett, hogy tetszőleges oldalra ugrana. Mivel az oldalak fontosságát a linkek alapján szeretnénk meghatározni (és a PageRankot is így definiáltuk), ezért adódik, hogy α-t válasszuk az egyhez minél közelebbi értéknek, mert így a web struktúrája, a linkek nagy hangsúlyt kapnak a Google mátrixban is. Az egyhez közeli α azt jelenti, hogy a véletlen szörföző szinte mindig a linkeken keresztül halad tovább.

Tudjuk azonban, hogy a Google, saját PageRank számolásánál $\alpha \approx 0,85$ [7] értéket használ. A fentiek ismeretében miért nem 0,9 vagy akár 0,999 ez az érték?

A válasz az, hogy α-tól nagyban függ a konvergencia sebessége, amit a következő fejezetekben indoklunk.

5.3. A hatványiteráció konvergenciájának sebessége

Legyenek a mátrixnak, melyre a hatványiterációt szeretnénk alkalmazni a következők a sajátértékei:

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|.$$

Látható, hogy λ_1 domináns sajátérték, azaz a hatványiteráció biztosan konvergálni fog. A valódi sajátvektor és az iteráció eredménye közti különbség minden lépésben aszimptotikusan csökken $|\lambda_1|/|\lambda_2|$ valamilyen többszörösével [1].

λ_1 domináns sajátérték, ezért a vizsgált hányadosról biztosan tudjuk, hogy

$$1 < |\lambda_1|/|\lambda_2|.$$

Látható, hogy minél kisebb ez a hányados, annál kisebb mértékben fog csökkenni a hiba mértéke az iterációs lépések között, azaz a konvergencia sebessége akkor a legrosszabb amikor $|\lambda_1|/|\lambda_2| \approx 1$, azaz $|\lambda_1| \approx |\lambda_2|$. A Google mátrix estében a sajátértékek:

$$1 > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|,$$

ezért a fentebb említett legrosszabb eset a következőképpen módosul:

$$1/|\lambda_2| \approx 1.$$

Minél közelebb van tehát a második legnagyobb sajátérték egyhez (a domináns sajátértékhez), annál lassabb lesz a konvergencia sebessége. A Google mátrixra alkalmazott hatványiteráció konvergenciájának pontos sebességét a 5.4 alfejezet végén adjuk meg, mivel ehhez a következő alfejezetben bizonyított ismeretek kellenek.

5.4. α mint sajátérték

A konvergencia sebessége úgy kapcsolódik az α paraméterhez, hogy α maga is sajátérték (a legtöbb esetben). Ezt az alábbi két tétel igazolja [4].

6. Tétel. *Ha a Google mátrix* $G = \alpha A + (1-\alpha)S$ *(*$\alpha \in [0,1)$*,* $G \in \mathbb{R}^{n\times n}$*) sajátértékei* $1 > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|$*, akkor* $\lambda_2 \leq \alpha$*.*

Bizonyítás. Először igazoljuk a tételt az $\alpha = 0$ esetre. Ekkor $G = S$, ahol $S = (\frac{1}{n})_{i,j}$ $(i,j = 1,2,\ldots,n)$. Ekkor $S = \mathbf{e}\mathbf{e}_p^T$, ahol $\mathbf{e}$ a csupa egyeseket, $\mathbf{e}_p$ pedig a csupa $\frac{1}{n}$ komponenseket tartalmazó oszlopvektorok. Mivel S felírható ilyen alakban, ezért $Rank(S) = 1$, ami miatt csak két sajátérték lehetséges: 1, mivel $(\mathbf{e}^T\mathbf{e}_p = 1)$ és 0 [9]. A második legnagyobb sajátérték tehát 0 és az egyenlőtlenség teljesül.

Az általános $0 < \alpha < 1$ esetet a következő lemmák segítségével igazoljuk.

6.1. Lemma. *A* G^T *mátrix* λ_2 *sajátértékhez tartozó sajátvektor (legyen* $\mathbf{x}_2$*) ortogonális az* $\mathbf{e}$ *vektorra, vagyis* $\mathbf{e}^T\mathbf{x}_2 = 0$*.*

Bizonyítás. Tudjuk, hogy egy mátrixhoz és transzponáltjához tartozó sajátvektorok ortogonálisak, ha nem ugyanahhoz a sajátértékhez tartoznak [6]. Mivel $\mathbf{x}_2$ a λ_2-höz tartozik G^T mátrixban, ezért ortogonális lesz G legnagyobb sajátértékéhez tartozó sajátvektorra.

Ez a sajátvektor éppen $\mathbf{e}$, mivel $G\mathbf{e} = \mathbf{e}$, amiből

$$(\alpha A + (1-\alpha)S)\mathbf{e} = \mathbf{e}$$

$$\alpha A\mathbf{e} + (1-\alpha)S\mathbf{e} = \mathbf{e}$$

A 3.1 fejezetben láttuk, hogy $A\mathbf{e} = \mathbf{e}$ és az is igaz, hogy $S\mathbf{e} = \mathbf{e}$, ezért az egyenlőség teljesül, $\mathbf{e}$ valóban sajátvektora G-nek, és ortogonális $\mathbf{x}_2$-re. □

6.2. Lemma. *Legyen* $\mathbf{x}_2$ *a* λ_2 *sajátértékhez tartozó sajátvektor a* G^T *mátrixban. Ekkor* $S^T\mathbf{x}_2 = 0$*.*

Bizonyítás. Tudjuk, hogy $S = \mathbf{e}\mathbf{e}_p^T$, vagyis $S^T = \mathbf{e}_p\mathbf{e}^T$. Ekkor $S^T\mathbf{x}_2 = \mathbf{e}_p\mathbf{e}^T\mathbf{x}_2$. Az előző lemmából tudjuk, hogy $\mathbf{e}^T\mathbf{x}_2 = 0$, ezért $S\mathbf{x}_2$ is 0 lesz. □

6.3. Lemma. $\mathbf{x}_2$ *vektor sajátvektora lesz A-nak is, és ha a hozzá tartozó sajátérték* μ*, akkor* $\mu = \lambda_2/\alpha$*.*

Bizonyítás. Az $\mathbf{x}_2$ vektor a G^T mátrix λ_2 sajátértékéhez tartozó sajátvektor, azaz $G^T\mathbf{x}_2 = \lambda_2\mathbf{x}_2$. Ha behelyettesítünk a $G = \alpha A+(1-\alpha)S$ helyettesítéssel, akkor

$$(\alpha A^T + (1-\alpha)S^T)\mathbf{x}_2 = \lambda_2\mathbf{x}_2.$$

Az előző lemmából tudjuk, hogy $S^T\mathbf{x}_2 = 0$, ezért

$$\alpha A^T\mathbf{x}_2 = \lambda_2\mathbf{x}_2.$$

Mivel az $\alpha = 0$ esetet külön bizonyítottuk, feltehetjük, hogy $\alpha \neq 0$ és oszthatunk vele, amiből

$$A^T\mathbf{x}_2 = \frac{\lambda_2}{\alpha}\mathbf{x}_2$$

adódik. Legyen $\lambda_2/\alpha = \mu$, ekkor

$$A^T\mathbf{x}_2 = \mu\mathbf{x}_2.$$

Látható, hogy $\mathbf{x}_2$ valóban sajátvektora A^T-nek is, és a hozzátartozó sajátérték éppen $\mu = \lambda_2/\alpha$. □

Az előző lemmából tudjuk, hogy $\lambda_2 = \mu\alpha$, továbbá A, a linkmátrix sztochasztikus, ezért $|\mu| \leq 1$. Ezekből következik, hogy $|\lambda_2| < \alpha$ (mivel egynél kisebb számmal kellett szorozni, hogy egyenlőek legyenek). □

A bizonyítás során azért nem foglalkoztunk az $\alpha = 1$ esettel, mert ekkor a Google mátrix a linkmátrix lesz, és még a konvergenciát sem tudjuk garantálni.

7. Tétel. *Ha a Google mátrix* $G = \alpha A + (1-\alpha)S$ *(*$\alpha \in [0,1)$*,* $G \in \mathbb{R}^{n\times n}$*) sajátértékei* $1 > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|$ *és A-ban van legalább kettő irreducibilis zárt részhalmaz, akkor* $\lambda_2 = \alpha$*.*

Bizonyítás. Az $\alpha = 0$ esetre a tételt már az előző bizonyítás elején beláttuk. A $0 < \alpha < 1$ esetet a következő két lemma segítségével igazoljuk.

Jelölések: Legyen $\mathbf{y}_i$ az A^T transzponált linkmátrix i. sajátvektora, μ_i pedig az ehhez tartozó sajátérték. Legyen $\mathbf{x}_i$ a G^T transzponált Google mátrix i. sajátvektora. Az előző bizonyításhoz hasonlóan $S = \mathbf{e}\mathbf{e}_p^T$, ahol $\mathbf{e}$ a csupa egyeseket, $\mathbf{e}_p$ pedig a csupa $\frac{1}{n}$ komponenseket tartalmazó oszlopvektorok.

7.1. Lemma. *A^T bármely $\mathbf{y}_i$ sajátvektora, amelyik ortogonális az $\mathbf{e}$ vektorra, a G^T mátrixnak is sajátvektora lesz. Továbbá ekkor $\lambda_i = \alpha\mu_i$.*
Bizonyítás. Tudjuk, hogy $\mathbf{e}^T\mathbf{y}_i = 0$, ezért

$$S^T\mathbf{y}_i = (\mathbf{e}\mathbf{e}_p^T)^T\mathbf{y}_i = \mathbf{e}_p\mathbf{e}^T\mathbf{y}_i = 0.$$

Ha kiírjuk a $G^T\mathbf{y}_i$ szorzatot, akkor

$$(\alpha A^T + (1-\alpha)S^T)\mathbf{y}_i = \alpha A^T\mathbf{y}_i + (1-\alpha)S^T\mathbf{y}_i = \alpha A^T\mathbf{y}_i$$

adódik. A sajátérték definíciója alapján $A^T\mathbf{y}_i = \mu_i\mathbf{y}_i$, ezért a fenti kifejezés megegyezik $\alpha\mu_i\mathbf{y}_i$-vel. Tehát

$$G^T\mathbf{y}_i = (\alpha\mu_i)\mathbf{y}_i,$$

vagyis $\mathbf{y}_i$ valóban sajátvektora lesz G^T-nek a $\alpha\mu_i$ sajátértékkel. □

7.2. Lemma. *$\exists i \in \{1, 2, \ldots, n\}$, hogy $\lambda_i = \alpha$.*
Bizonyítás. Keressük az $\mathbf{x}$ vektort, mely sajátvektora P-nek és ortogonális az $\mathbf{e}$ vektorra. A 2.4. fejezetben lévő tétel alapján az 1 sajátérték multiplicitása legalább kettő, mivel legalább ennyi irreducibilis zárt részhalmaz van A^T mátrixban. Ekkor biztosan létezik két lineárisan független sajátvektor, melyek az egy sajátértékhez tartoznak, legyenek ezek $\mathbf{y}_1, \mathbf{y}_2$. Ekkor

$$k_1 = \mathbf{y}_1^T\mathbf{e},$$
$$k_2 = \mathbf{y}_2^T\mathbf{e}.$$

Definiáljuk a keresett $\mathbf{x}$ vektort a következőképpen:

$$\mathbf{x} = \begin{cases} \frac{\mathbf{y}_1}{k_1} - \frac{\mathbf{y}_2}{k_2}, & \text{ha } k_1, k_2 \neq 0, \\ \mathbf{y}_1, & \text{ha } k_1 = 0, \\ \mathbf{y}_2, & \text{ha } k_2 = 0. \end{cases}$$

Ekkor $\mathbf{x}$ sajátvektora A^T-nek, az 1 sajátértékhez tartozik és ortogonális $\mathbf{e}$-re. A második és harmadik esetben ez egyértelműen látszik, az első esetben azért igaz, mert

$$A^T(\frac{\mathbf{y}_1}{k_1} - \frac{\mathbf{y}_2}{k_2}) = \frac{\mathbf{y}_1}{k_1} - \frac{\mathbf{y}_2}{k_2}$$
$$\frac{A^T\mathbf{y}_1}{k_1} - \frac{A^T\mathbf{y}_2}{k_2} = \frac{\mathbf{y}_1}{k_1} - \frac{\mathbf{y}_2}{k_2},$$

és $\mathbf{y}_1, \mathbf{y}_2$-ről tudjuk, hogy sajátvektorai A^T-nek az 1 sajátértékkel ezért $A^T\mathbf{y}_1 = \mathbf{y}_1$, $A^T\mathbf{y}_2 = \mathbf{y}_2$ és az egyenlőség teljesül.

Az előző lemma miatt ekkor $\mathbf{x}$ G^T-nek is sajátvektora lesz, méghozzá az α sajátértékkel. □

Ekkor tehát G^T-nek biztosan sajtértéke α és $\lambda_2 \geq \alpha$, mivel λ_2 a második legnagyobb sajátérték, α pedig biztosan nem lehet a legnagyobb, mert az egy. Az előző tétel viszont garantálja, hogy $\lambda_2 \leq \alpha$. A két egyenlőtlenség csak akkor teljesülhet egyszerre, ha $\lambda_2 = \alpha$. □

A tétel feltétele, vagyis az, hogy a linkmátrixban van legalább kettő irreducibilis zárt részhalmaz ugyan nem teljesül minden web linkmátrixára, de a sok oldalból álló webek estén szinte bisztosan igaz lesz, a linkmátrix 2.3. fejezet megjegyzésében említett ritkasága miatt.

Láttuk tehát, hogy α a vizsgált mátrix második legnagyobb sajátértéke, ezért az 5.3 alfejezet értelmében a hatványiteráció konvergenciájának sebessége függ α-tól. Ezt a sebességet a következő tétel adja meg pontosan [3]:

8. Tétel. *Legyen π a PageRank vektor, $\mathbf{x}^i$ pedig a Google mátrixra alkalmazott hatványiteráció i. lépésének erdeménye. Ekkor $\exists C \in \mathbb{R}$:*

$$\|\mathbf{x}^i - \pi\| \leq C\alpha^i, \quad i = 1, 2, \ldots$$

Megjegyzés: Ha a második tétel feltétele nem is teljesül, az első tétel akkor is érvényes. Ez azt jelenti, hogy nem ismerjük ugyan pontosan a második legnagyobb sajátértéket (és ezzel együtt a konvergencia sebességét), de felülről tudjuk becsülni α-val.

6. Algoritmus a PageRank meghatározására

Megadunk egy algoritmust a PageRank kiszámolására és a weboldalak fontosság szerinti sorba állítására, ami az előző fejezetekben leírtakon alapul. Az 1.2 alfejezetben említett program is ezt a módszert használja.

6.1. Algoritmus általános web esetén

Legyen adott egy tetszőleges, n oldalból álló web.

(I) Adjuk meg a webhez tartozó $A = (a_{i,j})$ linkmátrixot, a 2.3 fejezetben megadott képlet alapján:

$$a_{i,j} = \begin{cases} \frac{1}{|v_i|}, & \text{ha } (v_i, v_j) \in L. \\ 0, & \text{egyébként.} \end{cases}$$

A használt jelölések magyarázata a fent említett fejezetben található.

(II) Végezzük el a helyettesítést a lógó oldalak kiküszöbölésére, azaz ha az $n \times n$-es linkmátrixban van csupa nulla sor, azt cseréljük le $1/n$ elemekből álló sorra.

(III) Válasszunk egy $\alpha \in (0, 1)$ paramétert, figyelembe véve azt, hogy a túl kicsi érték elnyomja a linkek szerkezetét, az egyhez túl közeli pedig nagyon lelassítja a hatványiteráció konvergenciáját. Konstruáljuk meg a Google mátrixot ezzel a paraméterrel:

$$G = \alpha A + (1 - \alpha)S,$$

ahol S egy csupa $1/n$ elemekből álló $n \times n$-es mátrix.

(IV) Végezzük a hatványiterációt a G^T mátrixon, amíg a két iterációs lépés közötti különbség nem csökken valami kicsi ϵ érték alá, vagy adjunk meg konkrét iterációs lépésszámot. Az x_0 kiindulási pont legyen a csupa $1/n$ komponensekből álló vektor.

Megjegyzés: Az iterációt ne közvetlenül a sűrű G^T mátrixon, hanem az 5.1 alfejezetben említett

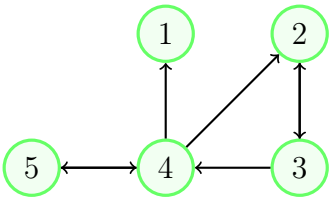
$$x^{i+1} = \alpha A^T x^i + (1 - \alpha)S^T$$

felbontás szerint végezzük, mert így csak a ritka linkmátrixal kell szorozni.

(V) Az iteráció eredménye a PageRank vektor közelítése, ahol az i. komponens az i. oldal fontosságát adja meg. Rendezzük a komponenseket csökkenő sorrendbe, így kapjuk az oldalak fontossági rangsorát.

6.2. Példa

Végrehajtjuk az előző alfejezetben leírt algoritmust egy konkrét webre, ez legyen a lenti ábrán látható irányított gráf.



(I) A gráfról leolvasható linkmátrix a következő lesz:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

(II) Látható, hogy az első oldal egy lógó oldal, ugyanis nincs belőle kimenő link, ezért a linkmátrixban helyettesítést kell alkalmaznunk az első sorban. A módosított linkmátrix ezért

$$\begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

(III) Válasszuk az $\alpha = 0,85$ értéket, a konvergencia sebessége és a linkek szerkezete közötti kompromisszumként. A Google mátrixot a

$$0,85 \begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} + 0,15 \begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix}$$

egyenletből kapjuk:

$$G = \begin{pmatrix} 0,2 & 0,2 & 0,2 & 0,2 & 0,2 \\ 0,03 & 0,03 & 0,88 & 0,03 & 0,03 \\ 0,03 & 0,455 & 0,03 & 0,455 & 0,03 \\ 0,313 & 0,313 & 0,03 & 0,03 & 0,313 \\ 0,03 & 0,03 & 0,03 & 0,88 & 0,03 \end{pmatrix}$$

(IV) Végezzük el a hatványiteráció első 5 lépését G^T mátrixra. A kapott eredményvektorok sorvektor formában:

$$\begin{aligned}
x_0 &= (0{,}2,\ \ 0{,}2,\ \ 0{,}2,\ \ 0{,}2,\ \ 0{,}2)\\
x_1 &= (0{,}121,\ \ 0{,}206,\ \ 0{,}234,\ \ 0{,}319,\ \ 0{,}121)\\
x_2 &= (0{,}141,\ \ 0{,}24,\ \ 0{,}225,\ \ 0{,}253,\ \ 0{,}141)\\
x_3 &= (0{,}126,\ \ 0{,}221,\ \ 0{,}258,\ \ 0{,}269,\ \ 0{,}126)\\
x_4 &= (0{,}128,\ \ 0{,}237,\ \ 0{,}239,\ \ 0{,}268,\ \ 0{,}128)\\
x_5 &= (0{,}128,\ \ 0{,}229,\ \ 0{,}253,\ \ 0{,}262,\ \ 0{,}128).
\end{aligned}$$

Ilyen kis méretű mátrix esetén nem jelent nehézséget a valódi sajátérték vektor meghatározása. A PageRank vektor ebben az esetben

$$\begin{pmatrix} 0,128 & 0,235 & 0,252 & 0,268 & 0,128 \end{pmatrix},$$

ami jól láthatóan közel van az iteráció eredményéhez, már az ötödik lépés után.

(V) Rendezzük sorba és rendeljük hozzá az oldalakhoz a kapott fontossági pontszámokat. Az eredmény leolvasható a táblázatból.

4. oldal	0,268
3. oldal	0,252
2. oldal	0,235
5. oldal	0,128
1. oldal	0,128

A legfontosabb oldal tehát a negyedik. A rangsorolást nehezíti, hogy a első és az ötödik oldal pontszáma annyira közel van egymáshoz, hogy 3 tizedesjegy pontossággal nem is lehet őket megkülönböztetni.

Hivatkozások

[1] David Bindel. *Power iteration.* 2016.

[2] Kurt Bryan and Tanya Leise. *The linear algebra behind Google.* 2016.

[3] Stoyan Gisbert and Takó Galina. *Numerikus módszerek.*

[4] Taher H. Haveliwala and Sepandar D. Kamvar. *The Second Eigenvalue of the Google Matrix*. 2003.

[5] D. L. Isaacson and R. W. Madsen. *Markov Chains: Theory and Applications*. 1976.

[6] E. L. Lady. *Eigenvectors and Diagonalizing Matrices*.

[7] Amy N. Langville and Carl D. Meyer. *Deeper inside PageRank, Internet Math*. 2005.

[8] Amy N. Langville and Carl D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. 2012.

[9] Silvia Monica Osnaga. *On rank one matrices and invariant subspaces*.

[10] Christiane Rousseau. *How Google works: Markov chains and eigenvalues*. 2015.