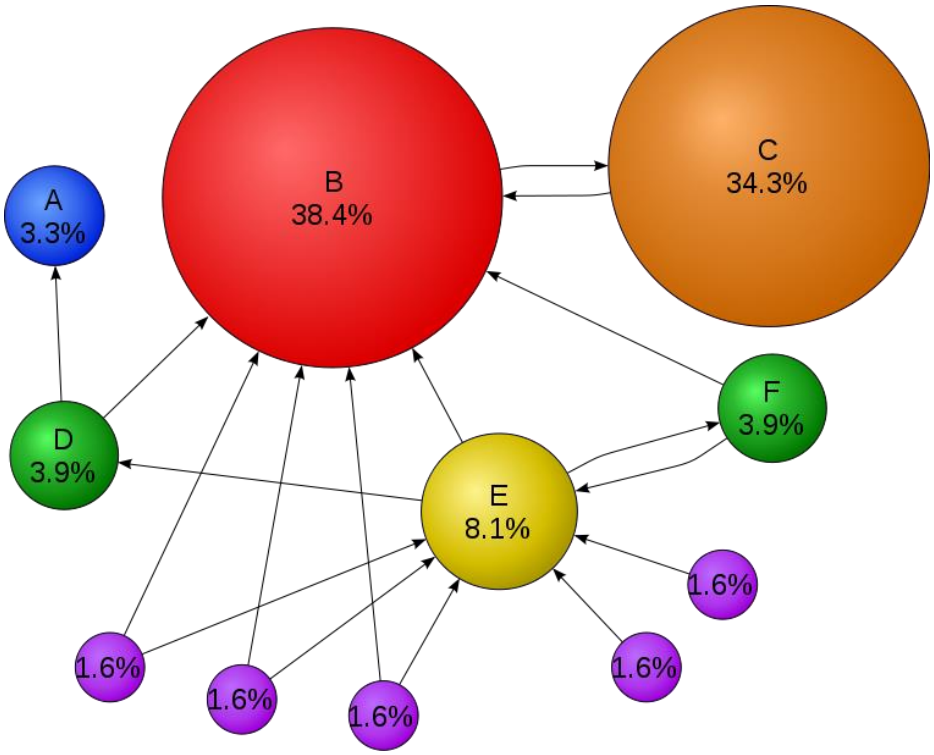
3. SLAJ, iterativne metode

PageRank

PageRank je algoritam koji *Google* koristi za potrebe rangiranja stranica. Algoritam je dobio naziv po jednom od osnivača *Google*-a, Larry Page-u, a osnovi zadatak algoritma je određivanje relevantnosti stranica.

Osnovna ideja

Svakom od elemenata iz skupa stranica se dodeljuje neka numerička vrednost koja se odredjuje iterativno, a ta vrednost zavisi od broja drugih stranica koje upućuju na nju, kao i od kvaliteta tih stranica, više od kvaliteta nego od kvantiteta.



Slika 1. Graf povezanosti stranica

Primer na slici 1 pokazuje da stranica ima C mnogo bolji *PageRank* rezultat nego stranica E, iako na stranicu C upućuje samo stranica B. Razlog tako visokog rezultata je visokva relevantnost stranice B.

Pojednostavljen algoritam

Inicijalno, sve stranice imaju jednaku verovatnoću i njena vrednost je $p \cdot r(X) = \frac{1}{N}$, gde je N ukupan broj stranica.

Za svaku od stranica se iterativno sračunava vrednost *PageRank*-a po sledećoj formuli:

$$p \cdot r(X)^{k+1} = \sum_{v \in B(x)} \frac{p \cdot r(v)^k}{L(v)}$$

, gde je $B(x)$ skup svih stranica koje pokazuju na stranicu X, a $L(v)$ je broj stranica na koje pokazuje stranica V (sumiraju se verovatnoće dolaska da svake od postojećih stranica sistema. Verovatnoća dolaska sa pojedinačne stranice sistema se određuje tako što se verovatnoća da se korisnik nalazi na datoj stranici v (sa koje treba da odabere link ka stranici X) $p \cdot r(v)$ podeli sa ukupnim brojem linkova koji postoje na stranici v. Ovde smo pretpostavili da korisnik sa istom verovatnoćom bira bilo koji od $L(v)$ linkova stranice v).

Damping faktor

Malo realnija varijanta algoritma u kojoj se korisniku daje mogućnost nasumičnog kretanja: korisnik može pristupiti određenoj stranici klikom na neki od linkova koji vodi ka datoj stranici, ali može i da direktno ukuca URL koji vodi do date stranice. Najčešće se pretpostavlja se da će korisnik u 85% slučajeva doći na stranicu putem linkova, da da će u svega 15% slučajeva sam da unese URL. Ovo predstavljamo vrednošću damping faktora $d = 0.85$.
Inicijalne verovatnoće posete svih stranica su i u ovom slučaju međusobno jednake, ali se formula za iterativno sračunavanje *PageRank*-a menja:

$$p \cdot r(X)^{k+1} = \frac{1-d}{N} + d \sum_{v \in B(x)} \frac{p \cdot r(v)^k}{L(v)}$$

, gde su:

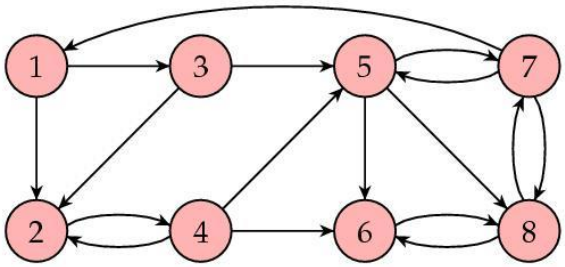
$\frac{1-d}{N}$: verovatnoća da će korisnik na nasumičan način (unosom URL-a) doći na našu stranicu. 15% verovatnoće da korisnik ukucava URL se uniformno raspodeljuje svim stranicama.

$d \sum_{v \in B(x)} \frac{p \cdot r(v)^k}{L(v)}$: verovatnoća da će korisnik doći na našu stranicu preko ostalih stranica koje pokazuju na našu (vrednosti iz prethodne formule bez damping faktora se množe sa verovatnoćom od 0.85 da korisnik na stranicu dolazi putem linkova).

PageRank i iterativne metode za rešavanje SLAJ

Problem određivanja ranga stranica se može svesti na problem rešavanja SLAJ. Definišimo matricu povezanosti H na sledeći način:

$$H_{ij} = \begin{cases} \frac{1}{l_j}, ako\ stranica\ j\ ima\ link\ ka\ stranici\ i, pri\ čemu\ je\ l_j\ ukupan\ broj\ linkova\ stranije\ j \\ 0, u\ suprotnom \end{cases}$$



stranica 2 pokazuje samo na stranicu 4

na stranicu 1 pokazuje samo stranica 7

$$H = \begin{array}{c} \\ 1 \\ \\ \\ 4 \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccccc} & 2 & & & & & 7 & \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 1/2 & 0 & 1/2 & 1/3 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1 & 1/3 & 0 \end{bmatrix} \end{array}$$

Slika 2. Matrica povezanosti

Vrednosti *PageRank*-a su elementi sopstvenog vektora r za modifikovanu matricu povezanosti H:

$$r = p + dHR$$
$$r - dHr = p$$
$$(I - dH)r = p$$

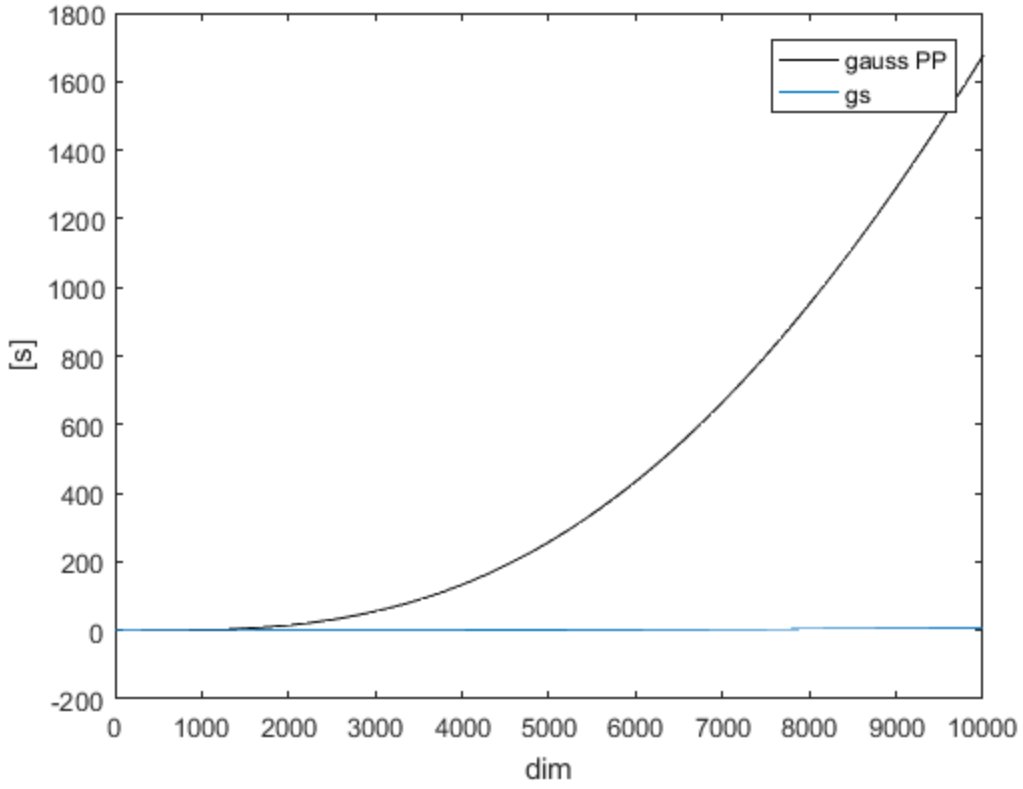
, gde su p vektor personalizacije, a I jedinična matrica. Najčešće su mu sve vrednosti jednake $\frac{1-d}{N}$ (u skladu sa pretpostavkom da će surfer sa jednakom verovatnoćom uneti URL ka bilo kojoj od N stranica sistema), mada je moguće da bude i drugačiji, tj. može da se podesi za svakog korisnika posebno (dati surfer ima preferencije ka određenim

stranicama sistema). Za matricu H važi $\sum_{i=1}^{n} H_{ij} = 1,\ j = 1,2,\ldots,n$ (kada se saberu sve verovatnoće dolaska putem linkova na svaku od stranica – ukupna verovatnoća treba da bude 1).

Rangiranje stranica se dobija rešavanjem sistema jednačina:

$$r = (I - dH) \backslash p$$

Rešenje ovog sistema je moguće dobiti i upotrebom Gausove metode. Gausova metoda je tačna, ali je relativno spora. Što je matrica H veća, duže se čeka na rešenje (slika 3) i zbog toga se traži manje tačno rešenje (jer apsolutna greška u ovom slučaju nije relevantna) dobijeno **iterativnim metodama**, ali do koga se dolazi daleko brže.



Slika 3. Ubrzanje dobijeno upotrebom iterativne metode, gde dim predstavlja dimenziju matrice

Zadatak 1

Data je matrica C koja definiše *link*-ove između 10 stranica dobijenih pretragom po nekoj ključnoj reči:

```
C =
     0  1  0  0  0  1  0  1  1  1
     0  0  0  1  0  0  1  0  1  1
     0  1  0  0  1  0  0  0  1  1
     1  0  0  0  1  1  1  1  0  1
     0  1  1  1  0  0  0  0  1  1
     1  1  0  1  0  0  0  0  1  1
     1  0  1  0  0  0  0  0  1  1
     0  1  1  1  1  0  1  0  0  1
     0  1  0  1  1  0  1  1  0  1
     0  1  1  1  1  1  1  1  1  0
```

a) Napisati funkciju `H = makeH(C)`, koja pretvara matricu C u matricu povezanosti H. Za dati broj *link*-ova l u koloni i matrice C pomnožiti kolonu C_i vrednošću $\frac{1}{l}$ da bi se dobila kolona H_i matrice H:

```
H =
         0    0.1429         0         0         0    0.3333         0    0.2500    0.1429    0.1111
         0         0         0    0.1667         0         0    0.2000         0    0.1429    0.1111
         0    0.1429         0         0    0.2000         0         0         0    0.1429    0.1111
    0.3333         0         0         0    0.2000    0.3333    0.2000    0.2500         0    0.1111
         0    0.1429    0.2500    0.1667         0         0         0         0    0.1429    0.1111
    0.3333    0.1429         0    0.1667         0         0         0         0    0.1429    0.1111
    0.3333         0    0.2500         0         0         0         0         0    0.1429    0.1111
         0    0.1429    0.2500    0.1667    0.2000         0    0.2000         0         0    0.1111
         0    0.1429         0    0.1667    0.2000         0    0.2000    0.2500         0    0.1111
         0    0.1429    0.2500    0.1667    0.2000    0.3333    0.2000    0.2500    0.1429         0
```

b) Sa pretpostavkom da korisnik do svake stranice dolazi praćenjem linka sa verovatnoćom od 85%, umesto unosom URL-a ($d = 0.85$), i da korisnik ima jednaku preferenciju ka svim stranicama ($p_{ij} = \frac{1-d}{10} = \frac{0.15}{10}, i \in [1, 10], j = 1$) **iterativnom metodom** naći vektor rangiranja stranica r:

```
r =
    0.1004
    0.0755
    0.0655
    0.1354
    0.0843
    0.0988
    0.0844
    0.1000
    0.1073
    0.1482
```

c) Napisati funkciju `pages = sortPages(pages, r)` koja na osnovu vektora rangiranja `r` sortira vektor stranica `pages` u opadajućem redosledu. Vektor stranica definisati na sledeći način:

```
pages = [
    {'page 1'}
    {'page 2'}
    {'page 3'}
    {'page 4'}
    {'page 5'}
    {'page 6'}
    {'page 7'}
    {'page 8'}
    {'page 9'}
    {'page 10'}]
```

Rezultat izvršavanja funkcije `sortPages`:

```
sortedPages =
    'page 10'
    'page 4'
    'page 9'
    'page 1'
    'page 8'
    'page 6'
    'page 7'
    'page 5'
    'page 2'
    'page 3'
```