DirichletRank algoritam i *link-spammanje* rezultata PageRank algoritma

Babić Marija
Runac Borna
Stanišić Matea
Vajdić Ivan

Siječanj 2020.

Sadržaj

1 Sažetak

Anti-spamming postao je jedan od najvažnijih izazova za web pretraživače i privlači pažnju, kako u industriji, tako i u znanosti. Kako većina današnjih pretraživača koristi algoritme koji se zasnivaju na poveznicama, *spamming* utemeljen baš na njima postaje veliki problem. U ovom radu pokazat ćemo da PageRank, popularni algoritam za rangiranje web stranica koji se temelji na poveznicama, ima manu koju nazivamo *nula-jedan praznina*. Iako se taj algoritam uspješno koristi prilikom rangiranja web stranica na Googleu, zbog nula-jedan praznina može se utjecati na rezultate rangiranja. Problem nula-jedan praznine nastaje zbog načina na koji se računaju prijelazne vjerojatnosti u modelu pretraživača. Predlažemo novi algoritam, DirichletRank algoritam, koji na bolji način računa prijelazne vjerojatnosti koje su u ovom slučaju temeljene na Bayesovskoj procjeni. DirichletRank algoritam zapravo je varijanta PageRanka, ali nema problem nula-jedan praznine, te je analitički pokazano da je bitno otporniji na *spamming* poveznica od PageRanka. Korištenjem stvarnih podataka, također je pokazano da je, zbog boljeg načina definiranja prijelaznih vjerojatnosti, DirichletRank algoritam efikasniji od PageRank algoritma.

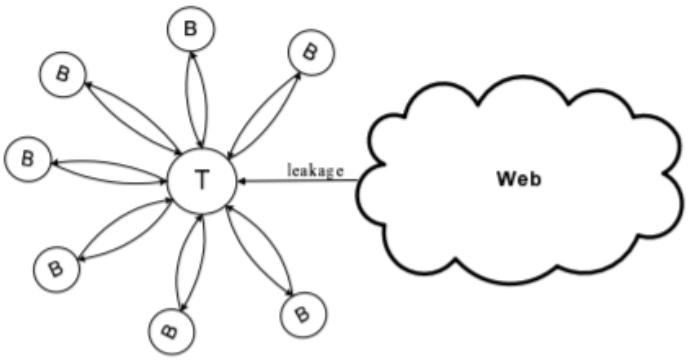
2 Uvod

Kako pretraživači postaju prevladavajući način prikupljanja informacija u svakodnevnom životu, veliku financijsku dobit ostvaruju web stranice koje se prve pojavljuju kao rezultati pretraživanja. To neizbježno dovodi do pojave *spamminga*. Web *spamming* je metoda kojom se zlonamjerno potiče pristranost pretraživača kako bi određene web stranice bile bolje rangirane. Kao posljedicu imamo pogoršanu kvalitetu rezultata pretraživanja i značajno se smanjuje korisnost web pretraživača.

Anti-spamming postaje veliki izazov za sve pretraživače. Začetci *spamminga* fokusirani su na sadržaj stranice i zasnivaju se na dodavanju velike količine ključnih riječi bez obzira na povezanost sa sadržajem stranice. Ova se vrsta *spamminga* relativno lako uoči i ne utječe previše na rezultate rangiranja. Korištenje algoritma baziranog na poveznicama među stranicama, kao što je PageRank, dovela je do razvoja novog tipa *spamminga*, koji nazivamo *link-spamming*, ili *spamming* baziran na poveznicama. Radi se o tome da se želi povećati rang neke stranice pri pretraživanju tako da se izgradi struktura poveznica koja uključuje veliki broj međusobno povezanih stranica. Za razliku od prvotnog načina *spamminga* koji je radio s ključnim riječima, *link-spamming* znatno povećava rang ciljane stranice te ga pretraživač teško može.

Kako *link-spamming* radi, objasnit ćemo na Slici 1.

Stranica T je stranica za koju želimo da je bolje rangirana. Stvaramo velik broj stranica B koje nam služe za svojevrsnu promociju stranice T. Stranice B sadrže poveznice na stranicu T te stranica T sadrži poveznicu na svaku stranicu B. PageRank algoritam dosta je osjetljiv na ovu vrstu strukture i njenom



Slika 1: Primjer *link-spamminga* uz korištenje lažnih stranica

upotrebom stranica T može biti rangirana i do 10 puta bolje. Ovo dovodi do zaključka da PageRank ima veliku manu, koju ćemo zvati problem nula-jedan praznine i koja može biti korištena za utjecaj na rezultate rangiranja.

Problem nula-jedan praznine odnosi se na nerazumno velike razlike u rezultatu PageRank algoritma među stranicama koje nemaju izlaznih poveznica i onih koje imaju samo jednu izlaznu poveznicu. Problem proizlazi iz načina računanja prijelaznih vjerojatnosti u modelu pretraživača. Kako bi riješili problem, predlaže se korištenje DirichletRank algoritma. To je poboljšana varijanta PageRanka u kojoj je riješen problem nula-jedan praznine i koja je bitno otpornija na ujecaj strukture sa slike na rezultate rangiranja pretraživača. Korištenjem stvarnih podataka, također je pokazano da je DirichletRank algoritam učinkovitiji nego PageRank algoritam.

2.1 Metode rangiranja

PageRank i HITS dva su najranije razvijena algoritma za analizu poveznica koji koriste svojstvene vektore za identifikaciju stanica putem *hyperlinkova*. Nakon njih nastalo je nekoliko metoda koje su poboljšale njihove rezultate. Neke od njih su PHITS i BHITS. Međutim, niti jedna od nastalih modifikacija nije riješila problem nula-jedan praznine, niti sturkturu sa Slike 1. Iako je PageRank stabilniji od HITS algoritma, pokazali smo da ni on nije dovoljno dobar i da se lako može manipulirati rezultatima rangiranja ukoliko pretraživač koristi taj algoritam.

Postoje i dvije varijante PageRanka kojima se pokušala poboljšati učinkovitost algoritma. To su *TrustRank* i *BadRank*. TrustRank se oslanja na činjenicu da *dobre* stranice rijetko pokazuju na *loše* stranice. Nedostatak je što treba postojati osoba koja će vršiti analizu i stranice razvrstavati u dobre i loše. S druge

strane, BadRank pretpostavlja da loša stranica uvijek pokazuje na stranice koje imaju veliku BadRank vrijedost. Naravno, ovo ne mora biti istina jer se jednostavno napravi loša stranica i na nju se postavi poveznica na dobru, čime dobroj stranici raste BadRank vrijedost, a ne bi trebala.

3 Problem nula-jedan praznine

U ovom odjeljku prvo dajemo kratak pregled PageRank algoritma, a nakon toga analiziramo njegov nedostatak — problem nula-jedan praznine i njegovu osjetljivost na *spamming* zasnovan na poveznicama.

3.1 Osnovni PageRank

Osnovni PageRank algoritam modelira dio interneta kao usmjereni graf $G(V, E)$ sa skupom vrhona V, koji se sastoji od N stranica, i skupom usmjerenih bridova E. Skup E reprezentiramo $N \times N$ matricom A koja je definirana kao

$$A = [a_{ij}]_{N\times N} \text{ gdje je } a_{ij} = \begin{cases} 1, & \text{ako } (i,j) \in E \\ 0, & \text{inače} \end{cases}$$

Pretpostavljamo da ako stranica u pokazuje na stranicu v da tada tvorac stranice u potvrđuje važnost stranice v. Štoviše, želimo da važna stranica više doprinosi važnosti stranica na koje pokazuje nego nevažna stranica. Ideja PageRanka je da važnost svake stranice računa uzimajući u obzir važnosti stranica koje na nju pokazuju. Za svaku stranicu v, s N_v označavamo skup stranica na koje ona pokazuje, s B_v skup stranica koje pokazuju na nju te s $r(v)$ PageRank *score* (važnost stranice) od v. Tako dolazimo do formule

$$r(v) = \sum_{u\in B_v} \frac{r(u)}{|N_u|}. \tag{1}$$

Ako normiramo matricu A po retcima i to označimo s M, PageRank iteracije možemo zapisati kao

$$r = M^T r. \tag{2}$$

Ako u svakom retku od A postoji barem jedna vrijednost različita od 0, matrica M je (stohastička) matrica prijelaza. Iteriranjem, zbog eventualne konvergencije, dolazimo do svojstvenog vektora r matrice M. Konvergencija je garantirana jedino ako je M ireducibilna i aperiodična. Kako M ne mora biti ireducibilna, definiramo uniformnu matricu U ($U_{ij} = \frac{1}{N}$) i interpoliramo je s matricom M uz faktor $1 - \lambda$

$$\tilde{M} = (1 - \lambda)M + \lambda U \tag{3}$$

gdje je λ vjerojatnost da će pretraživač s trenutne stranice otići na neku nasumičnu stranicu. Poboljšani PageRank *score* dobijemo kao

$$\begin{aligned} r &= \tilde{M}^T r \\ &= (1-\lambda)M^T r + \frac{\lambda}{N} e_N \end{aligned}$$

gdje je $e_N = (1, ..., 1)^T$ vektor stupac koji se sastoji od N jedinica. Treba još napomenuti da se obično vrijednost od λ postavlja na 0.15. Ako malo bolje proučimo ove formule, one nam kažu da će pretraživač slijediti izlazne poveznice neke stranice s vjerojatnošću $1-\lambda$, a s vjerojatnošću λ odlazi na neku slučajnu stranicu, pri čemu skup stranica na koje može otići ima uniformnu distribuciju. PageRank *score* jedne stranice možemo tumačiti kao vjerojatnost da će pretraživač opet posjetiti tu stranicu ako dovoljno dugo pretražuje.

3.2 Problem stranica bez izlaznih poveznica

Osnovni PageRank pretpostavlja da svaki redak matrice M ima barem jednu vrijednost različitu od 0, odnosno da sve stranice imaju barem jednu poveznicu. Nažalost, jako puno web stranica uopće ne sadrži poveznice. Štoviše, često se dogodi da se promatra samo podgraf cijelog interneta. U tim slučajevima, čak i da stranica ima izlazne poveznice, oni neće biti prikazani na podgrafu. Uklanjanje stranica koje nemaju poveznice nije rješenje, jer to može dovesti do toga da novi podgraf opet sadrži redak u kojem su sve vrijednosti 0. Zbog toga se sva poznata rješenja svode na rješavanje

$$\tilde{M} = (1-\lambda)M + \tilde{\lambda}U \tag{4}$$

gdje je

$$\tilde{\lambda} = \begin{cases} \lambda & \text{ako je } \sum_j M_{ij} = 1 \\ 1 & \text{inače} \end{cases}.$$

PageRank *scoreovi* dani su s

$$r = (1-\lambda)M^T r + \frac{\tau}{N} e_N$$

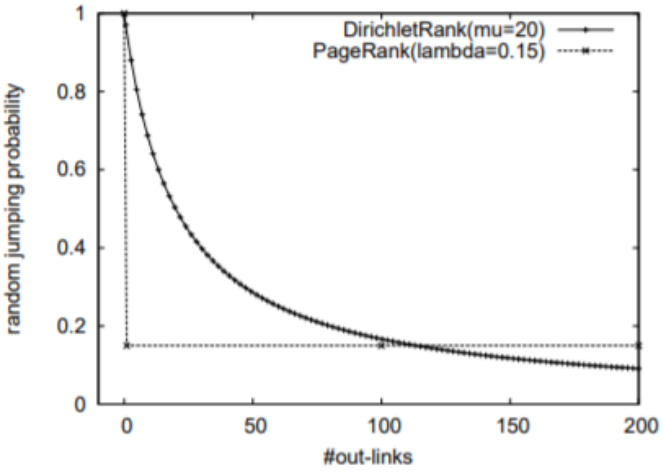
gdje je τ dobiven kao

$$\tau = \sum_{i=1}^{N} r(i) \times \tilde{\lambda}(i).$$

Vrijedi da je $\tau = \lambda$ ukoliko je $\tilde{\lambda}(i) = \lambda$ za svaki $i \in 1, ..., N$. Dokazano je da su (3) i (4) ekvivalentne u smislu konačnog rangiranja, čak i ako $\tilde{M}$ iz (3) nije Markovljeva matrica. Nadalje koristimo (4) kao formulu za PageRank.

3.3 Rješenje problema nula-jedan praznine

Iako jednadžba (4) rješava problem sa stranicama bez izlaznih poveznica, trebamo riješiti još jedan problem: vjerojatnost odlaska na slučajnu stranicu je 1 kod stranica koje nemaju izlaznih poveznica, a ta vjerojatnost padne na λ kod stranica koje imaju jednu izlaznu poveznicu. Usporedba vjerojatnosti skoka na slučajnu stranicu kod PageRank i Dirichletovog algoritma prikazana je na Slici 2.



Slika 2: Grafički prikaz vjerojatnosti skakanja u ovisnosti o broju poveznica

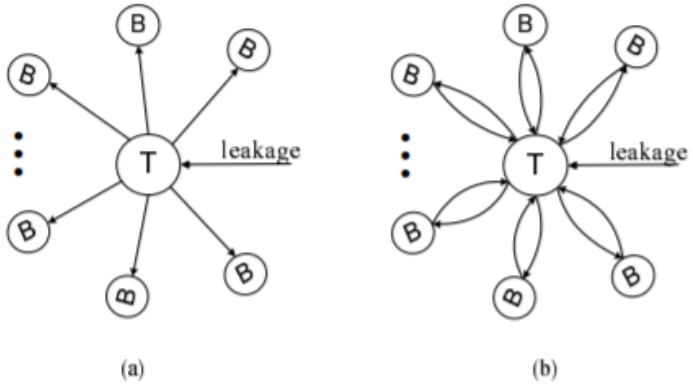
Možemo primijetiti kako kod PageRanka postoji velika razlika između stranice koja nema izlaznih poveznica i one koja ima samo jednu.

Dodatno, možemo pogledati i razlike između stranice koja nema izlaznih poveznica i one koja ih ima više. Taj slučaj prikazan je na Slici 3. Dok (a) prikazuje slučaj bez *spamminga*, u slučaju (b) prikazana je klasična struktura *link spamminga*. S $r_o(\cdot)$ označavamo PageRank *score* u prvom slučaju, a s $r_s(\cdot)$ PageRank *score* u drugom slučaju.

Teorem 1. *Uz k lažnih stranica, σ curenja i τ kao težinsku sumu vjerojatnosti skakanja, vrijedi*

$$r_o(T) = \sigma + \frac{\tau}{N}$$
$$r_s(T) = \frac{1}{2\lambda - \lambda^2}\left[\sigma + \frac{\tau(k(1-\lambda)+1)}{N}\right]$$

i vrijedi da je $r_s(T) \geq \frac{1}{2\lambda-\lambda^2} r_o(T)$.



Slika 3: Dvije međusobno suprotne strukture

Koristeći Teorem 1, pokaže se da vrijedi da je $r_s(T)$ veći ili jednak od $r_o(T)$ za svaki pozitivan k. Također se pokaže da to isto vrijedi i za bilo koji λ. Budući da $\frac{1}{2\lambda-\lambda^2} \to \infty$ ako $\lambda \to 0$, možemo zaključiti da mala vrijednost od λ, kakva se preferira u PageRanku, rezultira time da je $r_s(T)$ puno veći od $r_o(T)$. Na primjer, za $\lambda = 0.15$, $r_s(T)$ je tri puta veći od $r_o(T)$. To znači da će se kod stranica koje imaju samo jednu stranicu B pretraživač vratiti na prvotnu stranicu s vjerojatnošću 0.85. Ovim smo pokazali da problem nula-jedan praznine zaista predstavlja velik nedostatak PageRank algoritma i čini ga osjetljivim na *link spamming*.

4 DirichletRank algoritam

U ovom odjeljku opisat ćemo algoritam pod nazivom *DirichletRank algoritam*. Radi se o unapređenju već postojećeg PageRank algoritma u smislu poboljšanja alokacije i procjene prijelaznih vjerojatnosti. Osim što ovaj algoritam rješava prethodno opisani problem nula-jedan praznine, također na vrlo jednostavan i intuitivan način rješava problem u situacijama u kojima stranice nemaju izlaznih poveznica na druge stranice. Pokazat ćemo prednosti našeg algoritma naspram originalnog PageRank algoritma te analitički dokazati da je DirichletRank mnogo otporniji i stabilniji nego PageRank u slučajevima perturbacije poveznica ili većeg broja lažnih stranica.

4.1 Bayesovska procjena prijelaznih vjerojatnosti

U prethodnim odjeljcima uočili smo da je problem nula-jedan praznine uzrokovan nepravilnom alokacijom prijelaznih vjerojatnosti u modelu slučajnog pretraživanja. Potrebno je prilagoditi te prijelazne vjerojatnosti.

Neka je s v označena neka stranica u našem modelu slučajnog pretraživanja, a s L_v skup svih izlaznih poveznica stranice v. Tada skup L_v možemo smatrati slučajnim uzorkom (iz našeg modela). Prirodno je na ovakvom modelu prilagoditi multinomijalnu razdiobu stranici v − označimo ju s Θ_v. Dakle, možemo pisati

$$\Theta_v = (\theta_1, \theta_2, \ldots, \theta_N)\,,$$

gdje θ_i označava vjerojatnost da pretraživač prijeđe sa stranice v na neku drugu stranicu i (iz skupa L_v).

Klasično rješenje frekvencijske statistike bilo bi procijeniti parametre razdiobe Θ_v korištenjem metode maksimalne vjerodostojnosti. U tom su slučaju odgovarajuće procjene naprosto uzoračke proporcije. Te procjene, za svaku moguću stranicu, daju nam procjenu matrice M. Međutim, problem ovog pristupa je da će mnoge ovakve procjene biti jednake nuli, budući da u će velikom broju slučajeva stranice imati izlazne poveznice tek na nekoliko preostalih stranica.

S druge strane, Bayesovska statistika predlaže da parametru Θ_v zadamo neku priornu razdiobu (zaista, u Bayesovskoj statistici parametar Θ_v smatramo slučajnim vektorom koji ima neku razdiobu). Prirodno se nameće korištenje Dirichletove razdiobe čija je funkcija gustoće dana s

$$f(\Theta_v) = \frac{1}{\mathbf{B}(\alpha_1, \ldots, \alpha_N)} \cdot \prod_{i=1}^{N} \theta_i^{\alpha_i - 1}\,,$$

za $\theta_1, \ldots, \theta_N \geq 0$ takve da je $\displaystyle\sum_{i=1}^{N} \theta_i = 1$ te za neke parametre $\alpha_1, \ldots, \alpha_N > 0$. Funkcija $\mathbf{B}(\alpha_1, \ldots, \alpha_N)$ označava proširenu *beta*-funkciju definiranu s

$$\mathbf{B}(\alpha_1, \ldots, \alpha_N) = \frac{\prod_{i=1}^{N} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{N} \alpha_i\right)}\,.$$

Kako Dirichletova razdioba pripada eksponencijalnoj familiji, ona ima konjugiranu posteriornu razdiobu, tj. pokazuje se da je razdioba od $\Theta_v \mid L_v$ (razdioba našeg parametra nakon opažanja slučajnog uzorka) ponovno Dirichletova, ali s nekim drugim parametrima. Formalno, koristeći Bayesovo pravilo, imamo da je funkcija gustoće posteriorne razdiobe

$$f_{\Theta_v \mid L_v}(\theta_v, l) = \frac{f_{\Theta_v, L_v}(\theta_v, l)}{f_{L_v}(l)} = \frac{f_{L_v \mid \Theta_v}(l \mid \theta_v) \cdot f_{\Theta_v}(\theta_v)}{f_{L_v}(l)}\,,$$

što se klasično u Bayesovskoj statistici zapisuje kao

$$f_{\Theta_v|L_v}(\theta_v,\, l) \propto f_{L_v|\Theta_v}(l \mid \theta_v) \cdot f_{\Theta_v}(\theta_v)\,,$$

gdje znak $\propto$ znači *proporcionalno*.

Primijetimo da znamo razdiobe sve tri gornje slučajne varijable. Zaista, $L_v \mid \Theta_v$ ima multinomijalnu razdiobu s parametrima Θ_v, za razdiobu od Θ_v uzeli smo Dirichletovu razdiobu, a funkcija gustoće od L_v dobije se integriranjem brojnika po svim mogućim vrijednostima θ_i, tj.

$$f_{L_v}(l) = \int\limits_{\substack{(\theta_1,\ldots,\theta_N);\\ \theta_i \geq 0;\\ \sum_{i=1}^{N} \theta_i = 1}} f_{L_v|\Theta_v}(l \mid \theta_v) \cdot f_{\Theta_v}(\theta_v)\, d\theta_1\, d\theta_2 \cdots d\theta_N\,.$$

(Ponekad je teško ili nemoguće izračunati ovakav integral pa se u tom slučaju koriste procjene pomoću varijacijskih Bayesovih metoda. U našem slučaju integral se može egzaktno izračunati.)

Dakle, možemo odrediti i posteriornu razdiobu, tj. razdiobu od $\Theta_v \mid L_v$. Pokazuje se da je Dirichletova razdioba konjugirana priorna razdioba za multinomijalnu razdiobu te da je u tom slučaju

$$\boxed{\Theta_v \mid L_v \sim \mathrm{Dir}\left(\alpha_1 + c(l_1, L_v), \ldots, \alpha_N + c(l_N, L_v)\right)},$$

gdje su $c(l_i, L_v)$ opažene vrijednosti u uzorku (to jest, koliko se puta poveznica l_i pojavila u L_v).

Riječima, prvo zadamo neku priornu razdiobu na naše (nepoznate) parametre Θ_v (neko prethodno znanje ili mišljenje o našim parametrima) te opazimo slučajni uzorak uz te pretpostavljene parametre (tj. $L_v \mid \Theta_v$). Bayesov teorem tada kaže da je posteriorna razdioba (tj. razdioba našeg parametra nakon opažanja tog urozka) proporcionalna vjerodostojnosti našeg uzorka i priorne distribucije, što se često zapisuje kao

$$\text{Posterior} \propto \text{Prior} \times \text{Likelihood}\,.$$

U slučaju konjugirane priorne distribucije (kao ovdje), jednostavno „ažuriramo" naše prvotno znanje ili mišljenje o nepoznatom parametru Θ_v.

Procjena nepoznatih parametara se tada može izvršiti na nekoliko načina. Najčešće se za procjenu uzima očekivanje (ili vektor očekivanja) posteriorne distribucije (to je slučaj ovdje). Ukoliko želimo nešto otporniju procjenu, možemo uzeti medijan posteriorne distribucije. Neke od preostalih mogućnosti su uzimanje moda posteriorne distribucije (tzv. *maximum a posteriori*), tj. vrijednosti koja maksimizira posteriornu distribuciju, ukoliko taj maksimum postoji, zatim uzimanje one vrijednosti koja minimizira neku zadanu funkciju gubitka (tzv.

loss function), a najčešće se uzima varijanca posteriorne razdiobe), itd. Konkretno, neka su parametri α_i dani s

$$\alpha_i = \frac{\mu}{N}\,, \quad \forall i \in \{1,\, 2,\, \ldots,\, N\}\,,$$

gdje je μ neka vrijednost koju ćemo kasnije specificirati. Napomenimo još da, ako je $X \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_N)$, tada je očekivanje (tj. vektor očekivanja) od X dano s

$$\mathbb{E}(X) = \left(\frac{\alpha_1}{\sum_{i=1}^N \alpha_i},\ \frac{\alpha_2}{\sum_{i=1}^N \alpha_i},\ \ldots,\ \frac{\alpha_N}{\sum_{i=1}^N \alpha_i} \right),$$

što se lako vidi iz činjenice da su marginalne razdiobe od X ustvari beta-razdiobe.

Iz gornje diskusije zaključujemo da su Bayesovske procjene odgovarajućih parametara (prijelaznih vjerojatnosti) dane s

$$\begin{aligned}
\hat{\mathbb{P}}(l \mid L_v) &= \frac{\frac{\mu}{N} + c(l, L_v)}{|L_v| + \mu} \\
&= \left(1 - \frac{\mu}{|L_v| + \mu}\right) \cdot \underbrace{\frac{c(l, L_v)}{|L_v|}}_{= P_{\mathrm{ml}}} + \underbrace{\frac{\mu}{|L_v| + \mu}}_{= \omega_v} \cdot \underbrace{\frac{\mu}{N}}_{= P_{\mathrm{rand}}} \\
&= (1 - \omega_v) \cdot P_{\mathrm{ml}} + \omega_v \cdot P_{\mathrm{rand}}\,,
\end{aligned}$$

gdje prva jednakost slijedi iz

$$\sum_{i=1}^N (\alpha_i + c(l_i, L_v)) = \sum_{i=1}^N \frac{\mu}{N} + \sum_{i=1}^N c(l_i, L_v) = \mu + |L_v|\,.$$

Vrijednost P_{ml} predstavlja procjenu metodom maksimalne vjerodostojnosti (uzoračke proporcije), a P_{rand} uniformne prijelazne vjerojatnosti. Napravivši ovakvu procjenu za sve stranice v dobivamo relaciju

$$\tilde{M} = \mathrm{diag}\{1 - \omega_1,\, \ldots,\, 1 - \omega_N\} \cdot M + \mathrm{diag}\{\omega_1,\, \ldots,\, \omega_N\} \cdot U\,,$$

gdje su oznake za $\tilde{M}$, M i U zadržane iz prethodnih odjeljaka.

Ranking scores tada dobivamo rješavajući matričnu jednadžbu

$$\mathbf{r} = \tilde{M}^{\mathrm{T}} \cdot \mathbf{r}$$

te tada i-ta vrijednost u vektoru $\mathbf{r}$ odgovara *ranking scoreu* i-te web-stranice.

Iz definicije ω_v lako vidimo da je vrijednost $1 - \omega_v$ velika kad je i $|L_v|$ velika, to jest, ako stranica v ima mnogo izlaznih poveznica, pretraživač će s većom vjerojatnošću odabrati neku poveznicu sa stranice v.

Napomenimo još na kraju da smo ovaj algoritam nazvali *DirichletRank* algoritam upravo zbog postavljanja Dirichletove distribucije za priornu distribuciju nepoznatog parametra Θ_v.

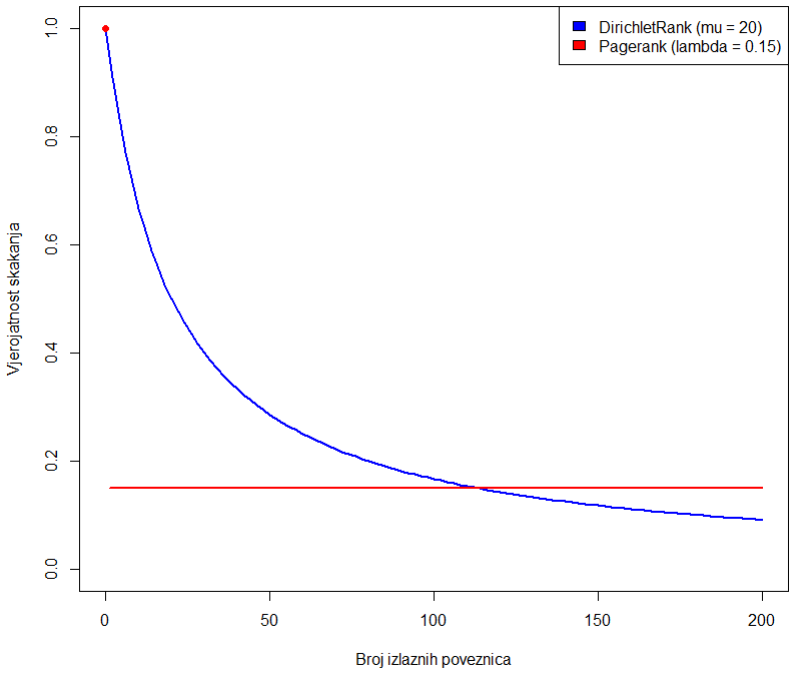
4.2 Usporedba s PageRank algoritmom

Metode Bayesovskih procjena daju nam elegantan način za određivanje prijelaznih vjerojatnosti. U ovome pododjeljku pokazat ćemo da ovakva alokacija prijelaznih vjerojatnosti dobro postupa u slučaju kad neka stranica nema izlaznih poveznica te da rješava problem nula-jedan praznine.

Vjerojatnost prelaska pretraživača na drugu stranicu (vjerojatnost skakanja) u DirichletRank algoritmu dana je s

$$\omega(n) = \frac{\mu}{n+\mu}, \qquad \text{za } 0 \leq n \leq +\infty,$$

gdje je $n \in \mathbb{N}$ broj izlaznih poveznica, a μ parametar priorne Dirichletove razdiobe. Postavimo $\mu = 20$ i nacrtajmo graf funkcije ω u ovisnosti o n. Graf je prikazan na slici ispod. Vidimo da u DirichletRank algoritmu graf izgleda glatko te nema praznine između nule i jedinice kao što je to slučaj kod PageRanka.



Slika 4: Grafički prikaz vjerojatnosti skakanja u ovisnosti o broju poveznica

Definirajmo vrijednosti $d_o(\cdot)$ i $d_s(\cdot)$ za DirichletRank algoritam slično kao i r_o i r_s definirane ranije. Vrijedi sljedeći rezultat.

Teorem 2. *Uz k lažnih stranica, σ curenja i τ kao težinsku sumu vjerojatnosti skakanja, vrijedi*

$$d_o(T) = \sigma + \frac{\tau}{N}$$
$$d_s(T) = \left(1 + \frac{k}{\mu^2 + (k+1)\mu}\right) \cdot \left(\sigma + \frac{k+\mu+1}{\mu+1} \cdot \frac{\tau}{N}\right).$$

Nadalje, vrijedi $d_s(T) \geq d_o(T)$, za svaki prirodni broj k.

Na prvi pogled zaključili bismo da nemamo neko značajno poboljšanje u odnosu na *scoreove* iz PageRank algoritma, budući da je ponovno vrijednost $d_s(T)$ uvijek veća ili jednaka vrijednosti $d_o(T)$. Međutim, lako se vidi da je $d_s(T)$, iako uvijek veće ili jednako $d_o(T)$, dosta blizu vrijednosti $d_o(T)$. Zaista, ako, na primjer, stavimo $\mu = 20$ i $k = 1$, imamo

$$d_s(T) = \left(1 + \frac{1}{20^2 + 2 \cdot 20}\right) \cdot d_o(T) = 1.0023 \cdot d_o(T) \approx d_o(T).$$

To pokazuje da ne postoji značajna razlika u *scoreu* od T prije i nakon *spamanja*, odnosno da je DirichletRank algoritam stabilniji od PageRank algoritma i ne toliko osjetljiv na promjenu lokalne strukture podataka.

Analizirajmo sada utjecaj broja k (broja lažnih stranica) na PageRank i DirichletRank. Ranije smo zaključili da za PageRank vrijedi

$$r_s(T) = \frac{1}{2\lambda - \lambda^2} \cdot \left(\sigma + \frac{\tau \cdot (k \cdot (1-\lambda) + 1)}{N}\right)$$

$$= \left(\frac{1-\lambda}{2\lambda - \lambda^2} \cdot \frac{\tau}{N}\right) \cdot k + \frac{1-\lambda}{2\lambda - \lambda^2} \cdot \left(\sigma + \frac{\tau}{N}\right).$$

Budući da je $1 + \frac{k}{\mu^2+(k+1)\mu} < 1 + \frac{1}{\mu}$, $\forall k \in \mathbb{N}$ (jer je ova nejednakost ekvivalentna s $\frac{-\mu-1}{\mu^2+(k+1)\mu} < 0$, što je istina), imamo

$$d_s(T) = \left(1 + \frac{k}{\mu^2 + (k+1)\mu}\right) \cdot \left(\sigma + \frac{k+\mu+1}{\mu+1} \cdot \frac{\tau}{N}\right)$$

$$< \left(1 + \frac{1}{\mu}\right) \cdot \left(\sigma + \frac{k+\mu+1}{\mu+1} \cdot \frac{\tau}{N}\right)$$

$$= \left(\frac{\tau}{\mu N}\right) \cdot k + \frac{\mu+1}{\mu} \cdot \left(\sigma + \frac{\tau}{N}\right).$$

Dakle, za *page score* u PageRanku vrijedi da on raste linearno u ovisnosti o k, s koeficijentom rasta $c_r = \frac{1-\lambda}{2\lambda-\lambda^2} \cdot \frac{\tau}{N}$. S druge strane, *page scoreovi* u DirichletRanku su odozgo ograničeni linearnom funkcijom s koeficijentom $c_d = \frac{\tau}{\mu N}$.

Obično se za vrijednost λ uzima $\lambda = 0.15$ iz čega slijedi da je $c_r = 3.06 \cdot \frac{\tau}{N}$. Međutim, kako je c_d opadajuća funkcija po μ, a za najmanju moguću vrijednost $\mu = 1$ vrijedi $c_d = \frac{\tau}{1 \cdot N} = \frac{\tau}{N}$, to zaključujemo da je za svaku moguću vrijednost μ koeficijent c_d manji od c_r. Ranije smo već postavili da je $\mu = 20$ pa za tu vrijednost vrijedi da je

$$c_d \leq \frac{\tau}{20 \cdot N} = 0.05 \cdot \frac{\tau}{N} \ll 3.06 \cdot \frac{\tau}{N} = c_r \, .$$

Zaključujemo da, ako *spammer* kreira $k \in \mathbb{N}$ lažnih stranica, njihov utjecaj na rangiranje kod DirichletRanka je značajno manji nego kod PageRanka.

Već smo ranije napomenuli da će u DirichletRanku slučajni pretraživač pratiti poveznice neke stranice v s velikom vjerojatnošću samo ako ta stranica v ima mnogo izlaznih poveznica. To nam je svojstvo ustvari i poželjno budući da nije nerazumno pretpostaviti da je takva stranica (s mnogo izlaznih poveznica) *hub* koja preusmjerava pretraživača na dobre *authority* stranice.

Komputacijski, ovaj je algoritam jednako složen kao i PageRank, to jest, ne izvršava se ništa dulje nego PageRank. Preostali poznati heuristički algoritmi za *anti-spamming* nemaju to svojstvo, to jest, vrijeme izvršavanja im je značajno dulje. Stoga ovaj algoritam predstavlja napredak u *anti-spammingu* te se zbog svoje brzine može efikasno primjenjivati na web-tražilice (npr. Google) i slično.

4.3 Utjecaj na *anti-spamming*

U ovome pododjeljku pretpostavljat ćemo *spam* poveznicu kao na Slici 3 (b) te koristeći takvu strukturu pokazati izuzetnu robusnost DirichletRank algoritma naspram klasičnog PageRank algoritma.

Primijetili smo da će, ukoliko stranica v ima malo izlaznih linkova, pretraživač prijeći na neku slučajnu stranicu s većom vjerojatnošću. To znači da pretpostavljena situacija (sa Slike 3 (b)) ne pogoduje *spammeru* budući da, ukoliko se pretraživač nađe na stranici T on će s velikom vjerojatnošću izaći iz cijele te strukture i prijeći na neku slučajnu stranicu. Naravno, *spammer* može pokušati stvoriti još poveznica na svakoj lažnoj stranici koje sve opet pokazuju na preostale lažne stranice. Takav čin izgleda opasno za naš algoritam budući da bi on trebao smanjiti vjerojatnosti skakanja i „zarobiti" pretraživača u *spammerovoj* strukturi lažnih stranica. Međutim, idući rezultat pokazuje da to nije tako.

Teorem 3. *Pretpostavimo da vrijede sljedeća tri uvjeta:*

1.) *Ciljna stranica T ima poveznicu na svaku lažnu stranicu B;*
2.) *Svaka lažna stranica B ima poveznicu na T;*
3.) *Stranice T i B nemaju izlaznih poveznica prema ostalim stranicama.*

Tada međusobne veze između lažnih stranica B nemaju utjecaj na DirichletRank algoritam.

Teorem nam kaže da dodavanje bilo kakvih veza između lažnih stranica B ne može promijeniti *page scoreove* u DirichletRank algoritmu. Sljedeći teorem pokazuje da je upravo situacija pretpostavljena na početku ovog pododjeljka (tj. situacija sa Slike 3 (b)) optimalna za *spammera* (tj. to je najbolje što *spammer* može učiniti). Ipak, potrebno je pretpostaviti da nema curenja na lažne stranice B, no to je opravdana pretpostavka jer su takve stranice kreirane upravo radi povećanja ranga ciljne stranice T.

Teorem 4. *Neka je $k \in \mathbb{N}$ broj lažnih stranica i pretpostavimo da ne postoji curenje ni prema kojoj lažnoj stranici. Tada je najbolji DirichletRank score za proizvoljnu spamming strukturu*

$$d_s(T) = \left(1 + \frac{k}{\mu^2 + (k+1)\mu}\right) \cdot \left(\sigma + \frac{k+\mu+1}{\mu+1} \cdot \frac{\tau}{N}\right).$$

Dakle, dodavanje bilo kakve izlazne poveznice koja vodi izvan *spamming* strukture može samo pogoršati tu strukturu. Iz prethodna dva teorema zaključujemo da je postavljanje *spamming* strukture kao na Slici 3 (b) najbolje što *spammer* može učiniti. Dakle, i s najboljom mogućom *spam* strukturom, utjecaj *spamminga* na DirichletRank algoritam je zanemarivo malen, to jest, DirichletRank *score* s optimalnom *spamming* strukturom je vrlo blizu *scoreu* bez ikakvog *spamminga*.

Ono što ipak može narušiti DirichletRank algoritam je kreiranje i postavljanje velikog broja lažnih stranica. Međutim, u prethodnom pododjeljku pokazali smo da je utjecaj broja lažnih stranica na DirichletRank *score* (tj. koeficijent c_d uz k) vrlo malen, što znači da bi *spammer* trebao kreirati velik broj lažnih stranica, što, osim što je zahtjevnije, omogućuje i njihovu lakšu detekciju. Kakav je točno utjecaj broja lažnih stranica k istražit ćemo u idućem poglavlju gdje se bavimo testiranjem našeg algoritma.

Zasad smo dokazali da je DirichletRank otporan samo na ovu specifičnu *spamming* strukturu. Naravno, *spamming* struktura koju smo ovdje opisali tek je jedna od velikog broja mogućih struktura. Međutim, većina *spamming* struktura i dosad proučavanih *anti-spamming* algoritama oslanja se na PageRank, čiji je glavni nedostatak problem nula-jedan praznine (a to je ono što omogućuje uspješan *spamming*). DirichletRank rješava problem nula-jedan praznine i svojom otpornošću nadmašuje PageRank.

5 Testiranje

Način određivanja ranga opisan u prethodnim poglavlljima reprezentirali smo algoritmom u programskom jeziku `Python`. Stoga smo u ovom poglavlju, koristeći podatke iz [1] i [2], analizirali dobivene rezultate *PageRank* i *DirichletRank* algoritma s ciljem usporedbe na osnovu točnosti ali i na temelju otpornosti na *nula-jedan praznina* problem.

5.1 Podaci

Podaci dobiveni iz [2] sadrže 10879 čvorova koji predstavljaju *hostove* u *Gnutella peer-to-peer* mreži u travnju 2002. Čvorovi su povezani sa drugim *hostovima* i ta veza predstavlja usmjerenu povezanost između njih. Prosječni koeficijent grupiranja ove mreže je 0.0062.

Podaci dobiveni iz [1] sadrže 9915 čvorova koji reprezentiraju stranice na domeni *Computer science Stanford* prikupljeni u 2001. Usmjerene veze između čvorova predstavljaju linkovi među stranicama. Stranice su *snažno* povezane.

5.2 Točnost algoritma

U ovom ćemo odjeljku usporediti točnost određivanja ranga stranice *DirichletRank* algoritmom u odnosu na *PageRank*. Točnost smo odredili koristeći formulu:

$$acc \;=\; \frac{\sum_{i=1}^{n} \mid pagerank(node_i) - dirichletrank(node_i) \mid}{\sum_{i=1}^{n} pagerank(node_i)} \tag{5}$$

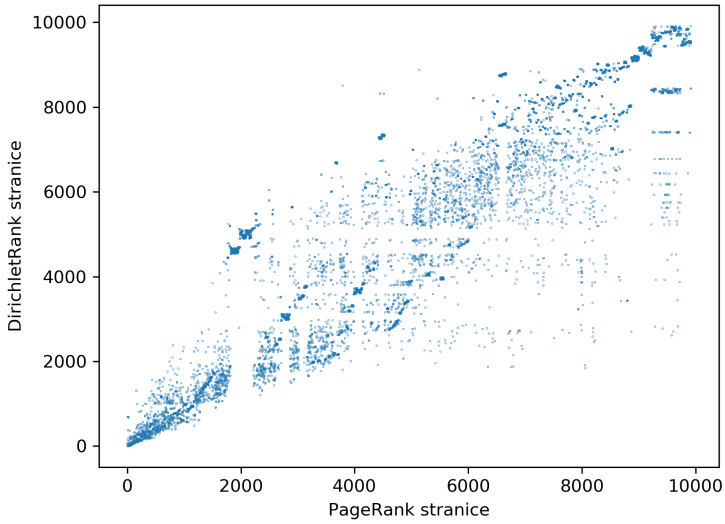
gdje je n broj čvorova u grafu, $node_i$ i-ti čvor u grafu, $pagerank(node_i)$ i $dirichletrank(node_i)$ rang čvora i dobiven *PageRankom*, odnosno *DirichletRank-om*. Uočimo da je sasvim svejedno koristimo li u nazivniku $pagerank(node_i)$ ili $dirichletrank(node_i)$ jer vrijedi:

$$\sum_{i=1}^{n} pagerank(node_i) = \sum_{i=1}^{n} dirichletrank(node_i).$$

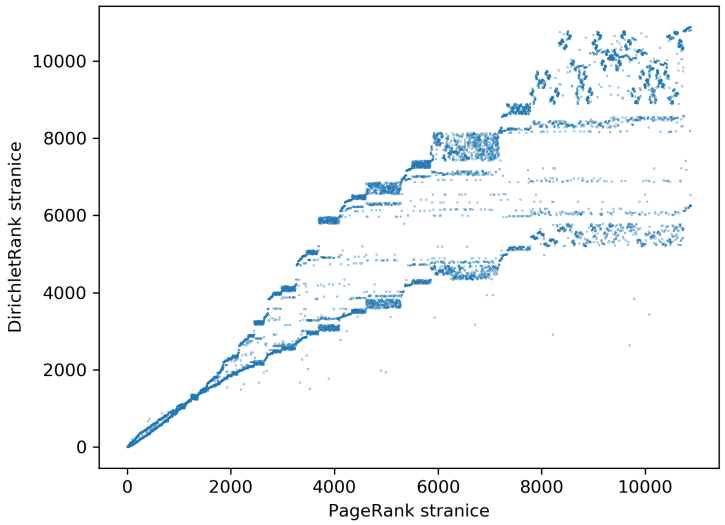
Što je vrijednost acc bliža 0, to je određivanje ranga *DirichletRank-om* točnije u odnosu na *PageRank*.

Pogledajmo rezultate za podatke [1] na Slici 5. Vidimo da rangovi stranica dobiveni *DirichletRank-om* nisu identični onima dobivenih *PageRankom*. Za to najvjerojatnije možemo okriviti veliki skup podataka od gotovo 10000 čvorova. No, iako rangovi nisu sasvim identični, iz Slike 5 se također može vidjeti da rezultati *prate dijagonalu*, što bi značilo da su poprilično slični. To se može vidjeti i iz izračunate vrijednosti 5 koja iznosi 0.1741180099435071.

Pogledajmo sada rezultate za podatke [2] na Slici 6. I ovdje rangovi stranica dobiveni *DirichletRank-om* nisu identični onima dobivenih *PageRankom*.



Slika 5: Rangovi stranica na *cs stanford* domeni koristeći *PageRank* i *DirichletRank*



Slika 6: Rangovi *hostova* na *peer-to-peer Gnutella* mreži koristeći *PageRank* i *DirichletRank*

Kako se i ovdje radi o podacima s više od 10000 čvorova, oni su vjerojatno krivac lošije točnosti ovih rezultata. No, isto tako moramo uzeti u obzir da su ovo podaci o slabo prosječno grupiranim *hostovima* u mreži, a ne o stranicama na internetu koje su najčešće gušće povezane. Sa slike također vidimo da rezultati jednim djelom *prate dijagonalu*, a drugim liniju iznad dijagonale. Vrijednost 5 za ove rezultate iznosi 0.1955298248812621, što nije preloša točnost, ali je gora od one za podatke iz [1].

5.3 Otpornosti na *nula-jedan praznina* problem

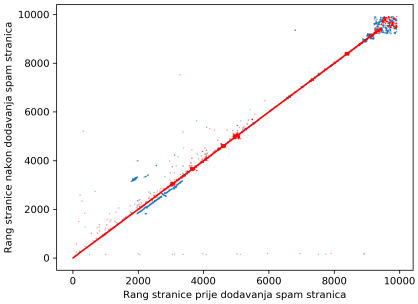
U ovom odlomku prikazati ćemo kako se *PageRank* i *DirichletRank* nose sa *link spamming* problemom. Otpornost na problem prikazali smo grafički projicirajući na isti graf rang stranica prije i poslije reproduciranja problema.

Prvo smo izračunali rang čvorova određenim algoritmom na preuzetim podacima. Problem smo dalje imitirali tako da smo preuzete podatke izmijenili na sljedeći način:

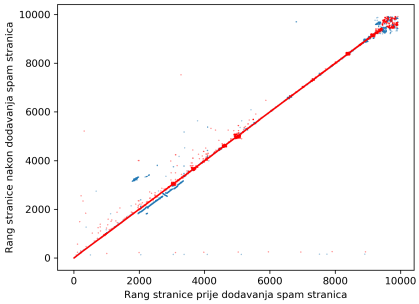
- Odabrali smo 10 ciljanih čvorova (npr. za podatke iz [2] odabrali smo čvorove sa sljedećim rangovima: 1000., 2000., ..., 10000.) .
- Maknuli smo sve izlazne veze iz ciljanih čvorova.
- Dodali smo k naivnih čvorova tako da je svaki imao jednu ulaznu vezu od ciljanog čvora i jednu izlaznu vezu prema ciljanom čvoru.

Nakon što smo izmjenili podatke, ponovno smo izračunali rang čvorova te ih u konačnici grafički usporedili s početnim rangom.

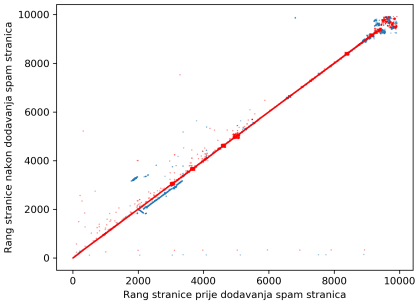
5.3.1 Rezultati za podatke iz [1]



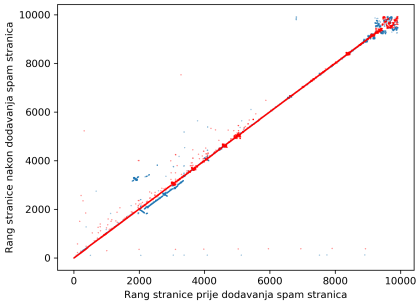
Slika 7: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 1 naivnog čvorova na podacima iz [1]



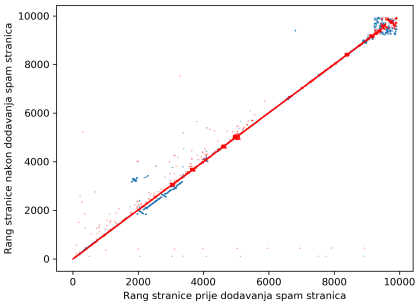
Slika 8: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 5 naivnih čvorova na podacima iz [1]



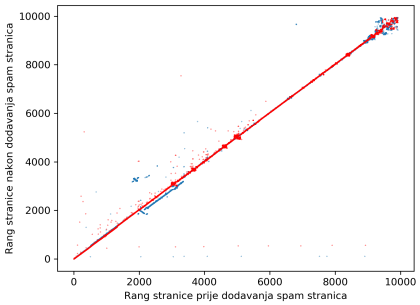
Slika 9: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 10 naivnih čvorova na podacima iz [1]



Slika 10: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 15 naivnih čvorova na podacima iz [1]



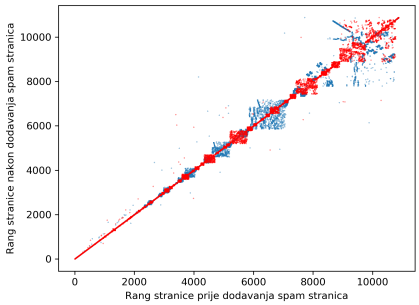
Slika 11: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 20 naivnih čvorova na podacima iz [1]



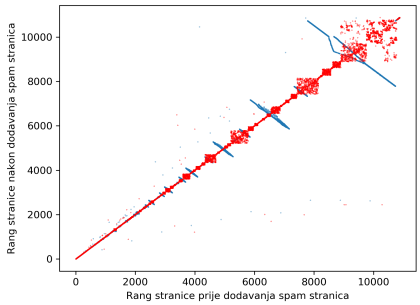
Slika 12: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 30 naivnih čvorova na podacima iz [1]

Podaci plavom bojom prikazuju rang čvorova dobiven *PageRankom* prije i poslije reproduciranja *link spamming* problema, dok su crvenom bojom prikazani čvorevi dobiveni *DirichletRank-om*. Sa slika se jasno vidi da je *DirichletRank* stabilniji za svaki $k = 1,\ 5,\ 10,\ 15,\ 20,\ 30$.

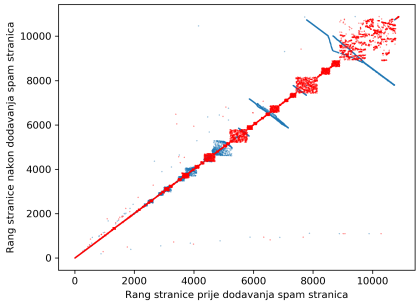
5.3.2 Rezultati za podatke iz [2]



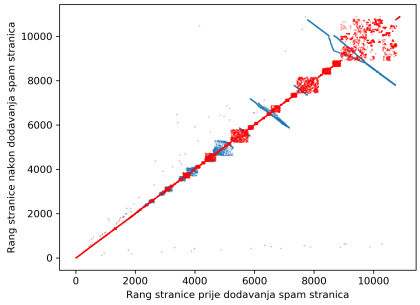
Slika 13: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 1 naivnog čvorova na podacima iz [2]



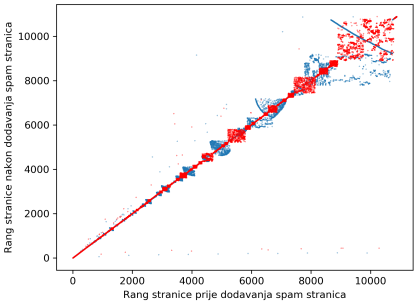
Slika 14: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 5 naivnih čvorova na podacima iz [2]



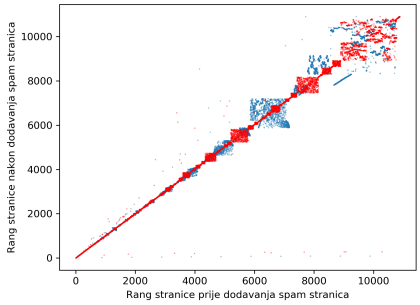
Slika 15: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 10 naivnih čvorova na podacima iz [2]



Slika 16: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 15 naivnih čvorova na podacima iz [2]



Slika 17: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 20 naivnih čvorova na podacima iz [2]



Slika 18: Otpornosti *PageRanka* i *DirichletRank-a* na *nula-jedan praznina* problem dodavanjem 30 naivnih čvorova na podacima iz [2]

Podaci plavom bojom prikazuju rang čvorova dobiven *PageRankom* prije i poslije reproduciranja *link spamming* problema, dok su crvenom bojom prikazani čvorevi dobiveni *DirichletRank-om*. Iako su ovdje rezultati puno nestabilniji nego za podatke iz [1], *DirichletRank* je stabilniji za svaki $k = 1, 5, 10, 15, 20, 30$. Odnosno u svim slučajevima, crveni rezultati puno bolje *prate dijagonalu*, nego plavi. Također, na ovim podacima može se lakše uočiti kako su algoritmi otporniji na problem na najbolje rangiranim stranicama, dok se na onim lošijim jasno vide odstupanja, posebice *PageRank* algoritma.

6 Rangiranje profesionalnih tenisača PageRank algoritmom

U ovom poglavlju demonstrirat ćemo kako se PageRank može koristiti za rangiranje sportaša. *ATP lista* je ljestvica *Udruženja teniskih profesionalaca* (engl. *Association of Tennis Professionals - ATP*) kojom se rangiraju tenisači na temelju njihovih rezultata na međunarodnim turnirima. Sastavlja se od 1973. godine te ima svoja pravila bodovanja. Svakom tenisaču na ATP listi se pribrajaju bodovi za uspjeh na turnirima u protekla 52 tjedna. Broj osvojenih bodova ovisi o veličini turnira te koliko dugo je igrač ostao u turniru. Na primjer, za osvajanje *Grand Slam* turnira pobjednik dobiva 2000 bodova, dok za osvajanje *ATP 500* turnira pobjednik dobiva 500 bodova [6]. 13. siječnja 2020. prvi na ATP listi je bio Španjolac Rafael Nadal [4].

6.1 Podaci

Koristili smo podatke o svim profesionalnim mečevima odigranih 2017. godine koji se mogu preuzeti sa servisa *DataHub* [3]. Kako bismo rangirali tenisače PageRank algoritmom, rezultate tenisača modeliramo usmjerenim grafom. Igrač

A pokazuje na igrača B ako ga je pobijedio bar jedanput u kalendarskoj godini. Bridovima dodjeljujemo težine koje su zapravo broj pobjeda igrača A nad igračem B. Na ovaj način dolazimo do $N \times N$ matrice A analogno matrici u poglavlju 3.1.

$$A = [a_{ij}]_{N\times N}, \quad a_{ij} = \text{broj pobjeda igrača } i \text{ nad igračem } j.$$

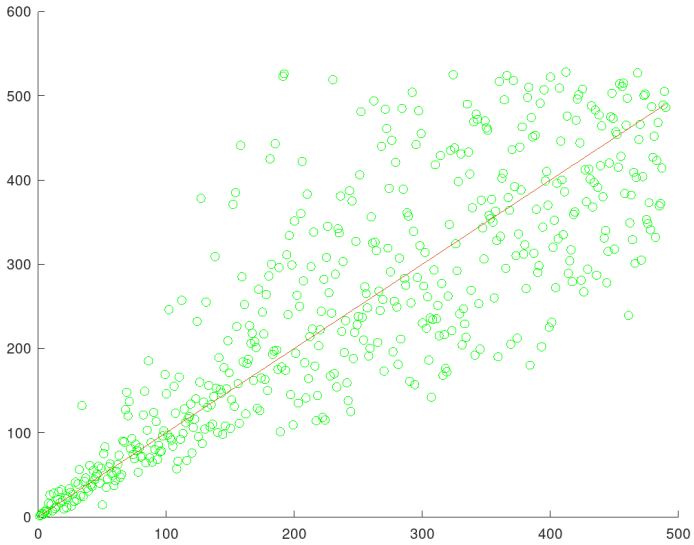
Neka je M matrica A normirana po retcima te neka je $\overline{M}$ matrica definirana kao u poglavlju 3.1. Uzimamo vrijednost $\lambda = 0.15$. Sada smo spremni provesti PageRank algoritam.

Alternativno, matricu A možemo definirati tako da umjesto da a_{ij} definiramo kao broj pobjeda igrača i nad igračem j, a_{ij} definiramo kao broj osvojenih setova ili broj osvojenih gemova igrača i nad igračem j. Na taj način utječemo na rezoluciju podataka te dobivamo različite rezultate rangiranje PageRankom.

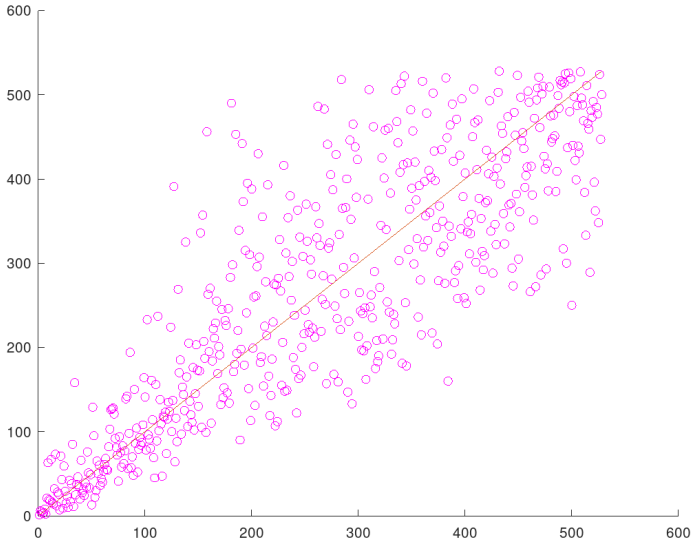
6.2 Usporedba s ATP ljestvicom

Rezultat PageRank algoritma prikazan je na sljedećim grafovima. Na horizontalnoj osi je pozicija igrača na ATP ljestvici na kraju 2017. godine, a na vertikalnoj osi je redni broj igrača dobiven PageRank algoritmom. Grafovi predstavljaju rezultate PageRank algoritma temeljene redom na broju pobjeda, broju osvojenih setova i broju osvojenih gemova. Crvena linija predstavlja dijagonalu.

Slika 19: Odnos ATP rankinga i PageRanka na temelju broja pobjeda



Slika 20: Odnos ATP rankinga i PageRanka na temelju broja osvojenih setova



Slika 21: Odnos ATP rankinga i PageRanka na temelju broja osvojenih gemova

ranking	ATP ljestvica	PageRank (broj pobjeda)	PageRank (broj setova)	PageRank (broj gemova)
1	Nadal	Federer	Nadal	Nadal
2	Federer	Nadal	Federer	Goffin
3	Dimitrov	Zverev	Zverev	Zverev
4	Zverev	Goffin	Dimitrov	Thiem
5	Thiem	Dimitrov	Goffin	Dimitrov
6	Cilic	Del Potro	Thiem	Federer
7	Goffin	Thiem	Del Potro	Bautista Agut
8	Sock	Kyrgios	Cilic	Cilic
9	Wawrinka	Cilic	Bautista Agut	Ramos Vinolas
10	Carreno Busta	Sock	Querrey	Mannarino

Neki osnovni podaci za svaki od tri PageRank rezultata prikazani su u tablici ispod, dok je u tablici iznad navedeno TOP 10 tenisača po ATP ljestvici i PageRanku. Za svaki PageRank računamo koeficijent korelacije u odnosu na ATP ranking te broj iteracija za konvergenciju do preciznosti $\varepsilon = 10^{-10}$. Također računamo koeficijent korelacije u odnosu na ATP ranking za TOP100 tenisača po ATP rankingu.

vrsta PageRanka	korelacija TOP100	korelacija svi tenisači	broj iteracija
broj pobjeda	0.83528	0.82306	25
broj osvojenih setova	0.79927	0.83382	30
broj osvojenih gemova	0.68226	0.84499	34

Zanimljivo je vidjeti da PageRank bolje prati ATP ranking ako koristimo podatke s većom rezolucijom (broj osvojenih gemova). Nasuprot tome, vidimo da PageRank bolje prati ATP ranking za TOP 100 tenisača na podacima s manjom rezolucijom (broj pobjeda). Također vidimo da je konvergencija sporija na podacima s većom rezolucijom. Možemo zaključiti da algoritmu treba više iteracija da bi obradio detaljnije podatke.

Vidjeli smo primjenu PageRanka za rangiranje profesionalnih tenisača te možemo zaključiti da se rangiranje PageRankom može koristiti za rangiranje rezultata sportaša. Predlažemo daljne primjene PageRanka u ostalim sportovima. U momčadskim sportovima poput nogometa ili košarke, PageRank se može koristiti za otkrivanje igrača koji najviše doprinose igri momčadi, na primjer mjerenjem broja dodavanja među igračima.

Literatura

[1] *Gleich/wb-cs-stanford*. https://sparse.tamu.edu/Gleich/wb-cs-stanford. Dohvaćeno: 2020-01-11.

[2] *Gnutella peer-to-peer network, August 4 2002*. http://snap.stanford.edu/data/p2p-Gnutella04.html. Dohvaćeno: 2020-01-11.

[3] *Rezultati tenisača s ATP liste*. https://datahub.io/sports-data/atp-world-tour-tennis-data. Dohvaćeno: 2020-01-11.

[4] *Službena stranica ATP toura*. https://www.atptour.com/en/rankings/singles. Dohvaćeno: 2020-01-13.

[5] Xuanhui Wang i dr. „DirichletRank: Solving the zero-one gap problem of PageRank." *ACM Trans. Inf. Syst.* 26 (siječanj 2008).

[6] *Wikipedia - ATP rankings*. https://en.wikipedia.org/wiki/ATP_Rankings. Dohvaćeno: 2020-01-13.