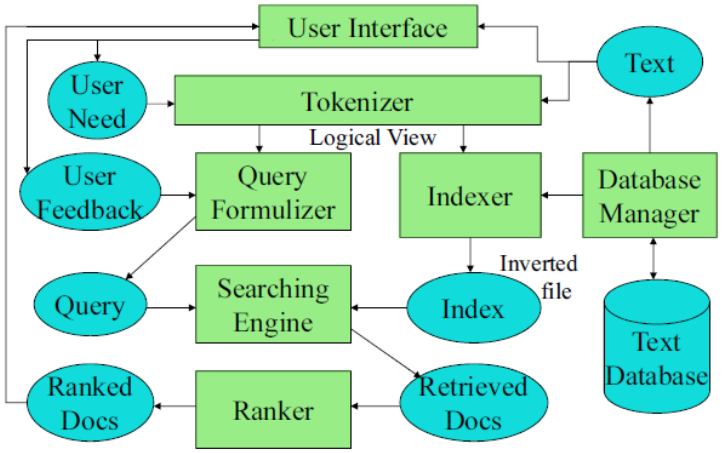
I grupa pitanja: IR sistemi i Bulov model pretraživanja

1. **Pojam pretraživanja informacija i IR sistema**

   Pretrazivanje informacija predstavlja pronalazenje materijala (uglavnom dokumenata), nestrukturne prirode (uglavnom tekst) unutar velike kolekcije (uglavnom smestene na racunaru) koji zadovoljava potrebu za informacijama.
   IR sistemi su sistemi za pronalaženje informacija
   Ulaz: Kolekcija tekstualnih dokumenata pisanih prirodnim jezikom.
   Korisnički upit – predstavljen, takodje, kao tekst
   Izlaz: Skup rangiranih dokumenata relevantnih za zadati upit

2. **Nacrtati strukturu IR sistema i objanisti ulogu svake komponente u njemu.**



- **Tokenizer** – razlaže tekst na termine, tj. Indeksne reči ili tokene
- **Indexer** – kreira takozvani invertovani indeks (veza izmedju termina i dokumenata u kojima se oni pojavljuju)
- **Searching Engine** – Korišćenjem invertovanog indeksa pronalazi dokumente relevantne za upit
- **Ranker** – rangira pronađene dokumente prema relevantnosti za upit.
- **User Interface** – omogućava interakciju sa korisnikom – prihvata korisničke upite i prikazuje rezultate pretraživanja.
- **Query Formulizer** – transformiše upit u model adekvatan modelu predtraživanja.

3. **Šta definiše model pretraživanja, kako se modeli pretraživanja dele?**
   Model pretraživanja definiše:
   - Način predstavljanja dokumenata,
   - Način predstavljanja upita,
   - Algoritam za pretraživanje,
   - Pojam relevantnosti (dokumenta za upit) - može biti definisana binarno ili kontinualnom funkcijom (rangiranje).

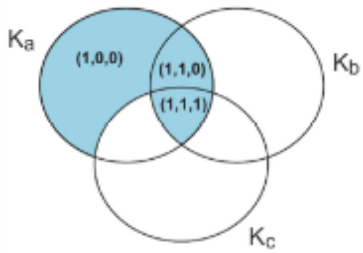
Podela modela pretraživanja prema tipu dokumenata

- Modeli za pretraživanje **nestruktrunih tekstualnih dokumenata** – Klasični modeli pretraživanja
  - Bazirani na teoriji skupova i Bulovoj algebra - Bulov model
  - Bazirani na algebri i statistici - Model vektorskog prostora
  - Bazirani na teoriji verovatnoće - Probabislistički modeli
- Modeli za pretraživanje **"semi-struktuiranih" dokumenata** (XML i sl.)
- Modeli za pretraživanje **multimedijalnih dokumenata**:
  - Pretraživanje slika
  - Pretraživanje audio zapisa
  - Pretraživanje video zapisa

4. Objasniti Bulov model pretraživanja.

Dokumenti se predstavljaju kao skupovi termina (ključnih reči ili deskriptora) $V = \{k_a, k_b, k_c\}$

Upiti se predstavljaju kao bulovi izrazi koji sadrženključne reči povezane operatorima I, ILI i NE; kao I zagrade kojima se definiše redosled izvodjenja operacija $q = k_a \wedge (k_b \vee \neg k_c)$



Algoritam za traženje se svodi na skupovne operacije

Relevantnost dokumenta je odredjena binarno (ne postoji rangiranje)

Prednosti:

- Veoma jednostavni i precizni modeli za predstavljanje dokumenata i upita
- Jednostavan algoritam traženja

Nedostaci:

- Ne postoji rangiranje dokumenata prema relevantnosti
- Rezultat pretraživanja može da sadrži ili
  - premali skup dokumenata (jer ne postoji mogućnost izdvajanja dokumnata koji delimično zadovoljavaju upit) i
  - preveliki skup dokumenata (zbog nepostojanja rangiranja prema relevantnosti kad je veliki rezultujući skup ne može se odlučiti koje dokumente prikazati korisniku, koje ne)

5. Objasniti implementaciju Bulovog modela pretraživanja korišćenjem matrice termina I dokumenata

Binarna matrica koja ima onoliko vrsta koliko se termina koristi za indeksiranje i onoliko kolona koliko je dokumenata u kolekciji.

Element matrice je definisan na sledeći način:

- $a_{ij} = 0$, ako termin t_i ne postoji u dokumentu d_j
- $a_{ij} = 1$, kada termin t_i postoji u dokumentu d_j

6. Objasniti implementaciju Bulovog modela pretraživanja korišćenjem invertovanog indeksa

Svakom dokumentu se pridružuje jedinstveni identifikacioni broj. Za svaki termin se pamti njegov **,,normalizovani zapis"** i **lista dokumenata** (tj. njihovih ID-a) u kojima se termin pojavljuje. U toj listi je onoliko elemenata koliko jedinica u vektoru učestalosti termina. U engleskoj literaturi za ovu listu se koristi termin **posting list**. Recnik se obicno pamti u operativnoj memoriji. Termini u recniku su obicno alfabetski uredjeni. Posting liste se smestaju na hard disk. Dokumenti u listi su uredjeni u rastucem redosledu prema ID-ju. Operacije se izvode nad posting listama termina, a osnovne operacije su osnovne skupovne operacije(unija, presek, razlika).

7. Rangiranje dokumenata u proširenom Bulovom modelu pretraživanja

U posting listi se uz ID dokumenta dodaje i težina termina (tj. njegova frekvenca pojavljivanja)

Izračunavanje relevantnosti rezultata

- Svaki Bulov izraz se može transformisati u PKNF.
- Relevantnost dokumenta za jedan potpuni proizvod u PKNF-u se odredjuje kao minimum frekvenci pojavljivanja termina koji u proizvodu učestvuju bez komplementa
- Relevantnost dokumenta za ceo upit (tj. za PKNF) se računa kao maksimum relevantnosti za potpune proizvode u izrazu.

8. Proširenje Bulovog modela pretraživanja dodavanjem parametara blizine

Prilikom zadavanja upita definišu se i parametri blizine izmedju termina. Parametri blizine mogu biti: susedno sa, na udaljenosti do n reči, unutar iste rečenice, unutar istog pasusa.
Načini obrade Bulovog upita sa parametrima blizine
I način: Pronaći sve dokumente koji zadovoljavaju ekvivalentan Bulov izraz bez parametara blizine. Grubim skaniranjem teksta iz izdvojenih dokumenata izbaciti one koni ne zadovoljavaju uslove blizine
Nedostatak ovog načina: sporost. Prednost: ne zahteva dodatni memorijski proctor
II način (Sa proširenjem posting liste):
U posting listi se za svaku pojavu termina u dokumentu kreira element koji pamti:

- za obradu parametara "susedno sa" i "na udaljenosti do n reči":
  - ID dokumenta,
  - redni broj reči unutar dokumenta
- za obradu svih parametara:
  - ID dokumenta,
  - redni broj pasusa kojem reč pripada,
  - redni broj rečenice unutar pasusa,
  - redni broj reči unutar rečenice.

Nedostatak ovog načina: veći utrošak memorijskog prostora. Prednost: brzina

II grupa: Invertovani indeks

1. **Šta je invertovani indeks? Objasniti osnovnu strukturu invertovanog indeksa.**

   Invertovani ideks je struktura podataka koja modelira vezu izmedju termina I dokumenta u kojima se oni pojavljuju. Za svaki termin se pamti njegov normalizovani zapis I lista dokumenata (tj njihovih ID-eva) u kojima se termin pojavljuje. Najcesce koriscene structure : uredjeni nizovi, binarna stable, B stable I Hash tabele.

2. **Koji su osnovni zadaci sistema za automatsko indeksiranje dokumenata? Kratko objasniti svaki od njih.**
   - Identifikacija dokumenata
     - Sta je dokument - 1 fajl, 1 mail, skup fajlova (slajdovi prezentacije)
     - U kom format je zapisan – pdf, word, html, txt
     - Koji jezik je korišćen u dokumentu?
     - Koji kod je korišćen?
   - Identifikacija reči (raščlanjivanje dokumenata na reči) – tokenizacija
     - Šta su reči? Nizovi karaktera medjusobno razdvojene belim simbolima i interpunkcijskim znacima.
     - Moze biti problematicno (datumi, slozenice, web ili e-mail adrese)
   - Normalizacija reči – dve faze
     - Konvertovati sva slova u velika ili sva u mala
       - Car, car i CAR zameniti sa car
     - Zameniti različite varijante iste reči jednim (korenim) oblikom
       - Ovom problematikom se bavi morfološki analizator (stemmer)
   - Odluka koje reči će se koristiti za indeksiranje dokumenata
     - Pojedinacni ili frazni termini
     - Sve reci ili izabrane reci
   - Ažuriranje invertovanog indeksa

3. **Šta su stemeri? Objasniti 2 osnovne metode u implementaciji stemera.**

   Stemeri su morfoloski analizatori. Njihova uloga je da razlicite varijante iste reci predstave jednim (korenim) oblikom

   Korišćenjem morfološkog rečnika
   - Morfološki rečnik je rečnik u kojem su upisane sve reči jezika i njihovi koreni oblici
   - Nakon izdvajanja svake reči iz teksta, pronaći u morfološkom rečniku odgovarajuću korenu reč
   - Osobine ove metode:
     - Tačna
     - Veoma spora
     - Iziskuje dodatni memorijski prostor za skladištenje morfološkog rečnika (a sami IR sistemima su već veoma zahtevni u tom pogledu)

   Heurističke metode
   - Uvode razna heuristička pravila za kreiranje osnovnih oblika reči
   - Primer – Poterov (brzi I neprecizni)

4. Navesti načine za predstavljanje rečnika u IR sistemima i prednosti i nedostatke svakog od njih.

Uredjeni nizovi
- Utrošak memorije minimalan, koliko je potrebno da se zapamti svaki termin ponaosob
- Algoritam traženja: binarno traženje (metod polovljenja intervala), Efikasnost: log2n
- Nedostaci: Algoritmi za dodavanje i brisanje termina veoma spori

Binarna stabla
- čvor sadrži jedan indeksni termin, pokazivač na podstablo koje sadrži indeksne termine koji mu prethode I pokazivac na one koji ga slede, stablo sadrži koren stabla
- Osobine: Potreban memorijski prostor se povećava, Algoritmi za traženje i dodavanje jednostavni, za brisanje čvora je nešto komplikovaniji - Izbrisani čvor se zamenjuje najmanjim potomkom iz desnog podstabla, stablo je nebalansirano – vreme traženja može da varira

B-stabla (B*,B+)
- Stablo balansirano (broj nivoa na svakoj putanji od korena do lista je isti), srednje vreme trazenja u odnosu na binarno stablo sa istim brojem termina znatno manje, manji broj nivoa, bolja iskoriscenost prostora
- čvor se satoji od maksimalno m-1 termina i m pokazivača, Termini istog čvora su uredjeni u rastućem radosledu

Hash tabele
- Termini se i dalje upisuju u nizove, ali se ne poštuje njihova leksikografska uređenost. Pozicija termina u nizu predstavlja hash kod termina. Funkcija koja termine transformiše u haš-kod, odnosno adresu, se naziva hash-funkcijom. Hash-funkcija mora da bude takva da, u najvećem broju slučajeva, različiti termini imaju različit haš kod. Kolizija – situacija kada se različitim terminima pridružuje isti hash kod.
- Vreme traženja veoma kratko, potreban memorijski prostor nešto veći, zbog narušavanja leksikografskog uredjenja termina takozvano tolerantno traženje je otežano (traženje nepotpuno definisanih termina)

5. Navesti algoritme za kreiranje invertovanog indeksa i kratko objasniti svaki od njih.

Sort-based: Kreiranje parova (termin, docID), Uredjivanje kolekcije parova (termin, docID) najpre prema terminu, prema docID-u, Na osnovu grupa parova sa istim terminom kreirati posting liste
- Svi podaci su u memoriji sve dok se ne završi ceo postupak kreiranja indeksa,
- Isti termin je zapamćeni više puta
- Neprimenljiv za velike kolekcije dokumenata

Blocked sort-based indexing (BSBI)
- Invertovani indeks se kreira u 2 prolaza:
- I prolaz: kreira se rečnik termina i svakom terminu se dodeljuje jedinstveni ID
- II prolaz:
  - Ulazna kolekcija dokumenata se deli na blokove (čija je dužina jednaka veličini jednog memorijskog bloka,
  - Kreiraju se parovi (termID, docID) za 1 blok, uredjuju se, formira se invertovani indeks bloka i smešta na disk
  - Mešaju se invertovani indeksi blokova u jedinstveni invertovani indeks
- Spor (2 porolaza kroz sve dokumente), Memorijski manje zahtevan od prethodnog, ali I dalje traži dodatni memorijski prostor jer je ceo rečnik sve vreme prisutan u opertivnoj memoriji

Single-pass in-memory indexing (SPIMI)

- Obradjuju se dokumenti i kreira se invertovani indeks dok veličina invertovanog indeksa ne bude jednaka veličini memorijskog bloka, uredjuje se i smešta na disk
- Mešanjem privremenih invertovanih indeksa dobija se invetovani indeks cele kolekcije
- Ne prave se parovi već se direktno kreira invertovani indeks,termin je zapamćen u memoriji jednom
- Posting liste se pamte kao nizovi - na početku se uvek kreiraju nizovi malih veličina a kada se posting lista napuni, njen kapacitet se duplira

Distribuirano indeksiranje

- Pomocu vise racunara
- Uloge u distrubuiranom indeksiranju
  - Master komjuter – dobija dokumente od spidera I dodeljuje ih parsermina
  - Parseri – mašine koje parsiraju dokumente I kreiraju parove (term,docID) koje upisuju u particije
  - Invertori – mašine koje uzimauju po jednu particiju iz svih parsera i kreiraju 1 particiju invertovanog indeksa za koju su oni zaduženi

Dinamičko indeksiranje

- Sve prethodne metode podrazumevaju da je korpus dokumenata koji se pretražuje nepromenljiv
- U praksi se vrlo često dodaju novi dokumenti i Brišu ili modifikuju postojeći dokumenti
- Metode za prevazilaženje ovog problema - povremeno pregenerisati ceo invertovani indeks I podržati permanentno ažuriranje invertovanog indeksa – dinamičko indeksiranje
- Održavati “veliki” glavni invertovani indeks
- Kreirati pomoćni invrtovani indeks za nove dokumente
- Traženje vršiti u oba i dobijene rezultate “smešati”
- Održavati i jedan bit-vektor u koji za svaki indeksirani dokument čuva indikator da li je dokument obrisan ili ne
- Rezultat pretraživanja na kraju filtritati pomoću ovog vektora. Povremeno ažurirati glavni invertovani indeks

6. Navesti metode za kompresiju rečnika termina i kratko objasniti svaku od njih.

Predstavljanje rečnika u obliku jedinstvenog stringa

Osim stringa u kojem su zapamćeni svi termini, deo rečnika je i pomoćna struktura u kojoj se pamti: Frekfenca pojavljivanja termina, pokazivač na posting listu, pokazivač na početak termina. Za traženje termina u ovakvoj strukturi podataka koristi se algoritam binarnog traženja

Predstavljanje rečnika u obliku stringa podeljenog na blokove

Jedinstveni string sa terminima se deli na blokove fiksne dužine (obično 4). Pre svakog termina u stringu pamti se njegova dužina U pomoćnoj strukturi se pamti: Pokazivač na početak bloka i za svaki termin frekfenca pojavljivanja termina i pokazivač na posting listu,

Kompresija termina sa istim prefiksom

Termini sa itim prefiksom: parada, paradajz, paradox
Prethodni način: pamti se svaki termin i njegova dužina
Kompresovani način: Prvi termin sa zajedničkim prefiksom se pamti kompletan pri čemu se jasno obeleži kraj prefiksa a za sve ostale termine sa istim prefiksom pamti se dužina sufiksa i sufiks

7. **Navesti metode za kompresiju posting listi i kratko objasniti svaku od njih.**

Umesto ID-a svakog dokumenta, pamte se razlike izmedju 2 susedna elementa u listi

Na ovaj nači se smanjuje opseg brojeva koje treba pamtiti u posting listi pa samim tim i memorijski prostor potreban za njihovo pamćenje. U praksi se pokazalo da je za pamćenje razlike dovoljno koristiti 20 bitova, tj. da njena vrednost ne prelazi 2^{20}

Umesto binarnog pbrojnog sitema koriste se posebni kodovi za predstavljanje brojeva

Pošto dužine brojeva koji se pamte u posting listi drastično variraju, koriste se kodovi promenljive dužine

- Promenljivi bajt-kod (VB – valiable byte code) - U svakom bajtu prvi bit se koristi kao indikator da li je tekući bajt (početak novog podatka ili nastavak prethodnog). Za pamćenje broja se u svakom bajtu koristi 7 bitova
- Unarni kod - Prirodni broj n se predstavlja nizom od n jedinica iza kojeg sledi jedna 0
    - 3: 1110
    - 30: 111111111111111111111111111110
- Gama kod (kod promenljive duzine) - Broj se predstavlja u obliku para: length, offset. length – broj bitova pomoću kojih je predstavljen offset, offset – dobija se tako što se iz binarne reprezentacije broja izbaci jedinica sa pozicije najveće težine. Dužina gama koda - broj bitova potrebnih za predstavljanje broja B je $2 \log_2 B + 1$ bitova

Kreiramo γ kod broja 30

$(30)_{10} = (11110)_2$

$offset = 1110$

$length = 4 \rightarrow 11110$

γ kod = 111101110

Kreiramo γ kod broja 3

$(3)_{10} = (11)_2$

$offset = 1$

$length = 1 \rightarrow 10$

γ kod = 101

III grupa: Vektorski model pretraživanja i evaluacija IR sistema

1. **Objasniti vektorski model pretraživanja**

U modelu vektorskog prostora za svaki termin se definiše težina sa kojom on učestvuje u opisu nekog dokumenta/upita. Dokumenti i upiti se definišu kao vektori težina u prostoru svih termina iz rečnika.

Za definisanje algoritma pretraživanja i relevantnosti dokumenata za upit uvodi se pojam sličnosti izmedju dokumenata i upita. Sličnost izmedju dokumenta i upita se definiše kao kosinus ugla koji zahvataju vektor dokumenta i vektor upita. U pitanju je ranked retrival. Prednosti: model jednostavan I precizno definisan. Nedostaci: kompleksnost predstavljanja dokumenta, upita I algoritma za trazenje.

2. **Navesti mere koje se u vektorskom modelu pretraživanja koriste kao težina termina u dokumentu ili upitu i objasniti logaritamsku frekvencu pojavljivanja termina**

Načini definisanja težine termina :

- Korišćenjem frekvence temina u dokumentu/upitu
- Korišćenjem normalizovane frekvence termina
- Korišćenjem logaritma frekvence termina
- Korišćenjem tf-idf mere

Frekvenca termina predstavlja broj pojavljivanja termina u dokumentu ili upitu.
Posledica: Dokumenti sa većim brojem pojavljivanja datog termina će se smatrati relevantinijim
Nedostatak: Učestali termini lošije opisuju sadržaj dokumenta nego retko korišćeni termini.
Normalizovana frekvenca termina: Količnik frekvence pojavljivanja termina u posmatranom dokumentu i maksimalne frekvence pojavljivanja tog termina u bilo kom dokumentu korpusa.
Logaritam frekvence termina (logaritamska težina): umesto frekvence termina kao tezina se koristi velicina:

$$w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d}, & \text{ako je } tf_{t,d} > 0 \\ 0, & \text{inace} \end{cases}$$

Korišćenje mešovitih mera: Često se u pretraživačima koriste različite težine u vektorima koji predstavlaju i upite.
Za dokumente se najčešće kotisti tf-idf težina. Za upite se obično koristi logaritamska frekfenca termina bez normalizacije (bez deljenja intenzitetom vektora)

3. **Navesti mere koje se u vektorskom modelu pretraživanja koriste kao težina termina u dokumentu ili upitu i objasniti tf-idf meru**

Tf-idf mera je proizvod logaritma frekvence termina i inverzne frekvence dokumenata za posmatrani termin. Frekvenca dokumenata za dati termin t(fdt) je broj dokumenata u korpusu koji sadrže posmatrani termin. Inverzna frekvenca dokumenata za posmatrani termin t se definiše kao: $idf_t = \log_{10}(N/df_t)$ gde je N ukupan broj dokumenata u korpusu. Inverzna frekvenca dokumenata je veća za termine koji se retko pojavljuju u korpusu.

$$w_{t,d} = (1 + \log_{10} tf_{t,d}) \times \log_{10}(N / df_t)$$

4. **Objasniti algoritam traženja u vektorskom modelu pretraživanja**

- Predstaviti upit kao vektor težina u vektorskom modelu pretrazivanja
- Predstaviti svaki dokument kao vektor tf-idf težina
- Izračunati sličnost upita i svakog dokumenta u korpusu kao kosinus ugla izmedju odgovarajućih vektora
- Rangirati dokumente prema sličnosti sa upitom
- Vratiti korisniku prvih K dokumenata

```
CosineScore(q)
1  float Scores[N] = 0
2  float Length[N]
3  for each query term t
4  do calculate w_{t,q} and fetch postings list for t
5     for each pair(d, tf_{t,d}) in postings list
6     do Scores[d] += w_{t,d} × w_{t,q}
7  Read the array Length
8  for each d
9  do Scores[d] = Scores[d]/Length[d]
10 return Top K components of Scores[]
```

5. **Objasniti neheurističke metode za optimizaciju algoritma traženja u vektorskom modelu pretraživanja**
   - **Zanemariti težine termina u upitu**

Najčešće se u upitu svaki termin pojavljuje samo jednom . U tom slučaju sve nenulte komponente u vektoru težina za upit imaju istu vrednost (w), tj. vektor težina upita se može predstaviti kao $\vec{v}(q) = w \times \vec{V}(q)$ gde je V(q) vektor koji ima vrednost 1 na pozicijama koji odgovaraju terminima koji se pojavljuju u upitu. Dokument d1 je relevantniji za upit od dokumenta d2 ukoliko je zadovoljen uslov

$$\frac{\vec{v}(d_1)\cdot\vec{v}(q)}{|\vec{v}(d_1)|\cdot|\vec{v}(q)|} > \frac{\vec{v}(d_2)\cdot\vec{v}(q)}{|\vec{v}(d_2)|\cdot|\vec{v}(q)|} \Leftrightarrow \frac{\vec{v}(d_1)\cdot\vec{V}(q)}{|\vec{v}(d_1)|} > \frac{\vec{v}(d_2)\cdot\vec{V}(q)}{|\vec{v}(d_2)|}$$

- **Umesto svih dokumenata, obraditi samo dokumente koji sadrže bar jedan termin iz upita**

Umesto svih dokumenata (N), obraditi samo dokumente koji se pojavljuju u posting listama termina koji učestvuju u upitu (J). Efekat: Ušteda memorije I smanjenje vremena postrebnog za sortitanje rezultata

- **Izdvajanje k najrelevantnijih dokumenata pomoću heapa**

Kreiranje Max-Heapa od J elemenata = dodati J elemenata u Max-Heap

Dodavanje: Upisati element na prvo slobodno mesto u heapu. Ukoliko je dodati element manji od svog roditelja, zameniti im mesta. Zamenu nastaviti sve dok dodati element ne bude manji od svog roditelja, ili dok ne bude postavljen u koren heapa

Izdvajanje K najrelevantnijih dokumenata = K puta izbaciti najveći element iz heapa q

Izbacivanje: Izbaciti element iz korena heapa. U koren heapa prebaciti poslednji element heapa. Ukoliko je neki potomak veći od elementa u korenu, zameniti im mesta. Postupak zamene nastaviti posmatrani element ne bude veći od oba svoja potomka, ili dok ne bude postavljen u list stable

6. **Objasniti heurističke metode za optimizaciju algoritma traženja u vektorskom modelu pretraživanja**

Generalna ideja je da iz skupa svih dokumenata izdvojiti podskup A takav da je $k \leq |A| \leq N$ i A sadrži većinu dokumenata koji spadaju u k najsličnijih sa upitom. Procesirati samo dokumente podskupa A, tj. izdvojiti k dokumenata iz podskupa najsličnijih upit

Kriterijumi za izbor podskupa dokumenata koji će se procesirati: Procesirati samo dokumente sa najvećom učestanošću termina iz upita, ,,ocenom kvaliteta", one koji sadrže m od ukupno n termina iz upita ili podeliti ulaznog skupa dokumenata na klastere

- **Izdvajanje dokumenata sa najvećom frekvencom termina iz upita**

Elementi posting liste se uredjuju po frekvenci termina u dokumentu, procesira se samo prvih r dokumenata (r>>k). Nedostatak: paralelna obrada posting listi nije moguca

- **Izdvajanje dokumenata sa najvećom ocenom kvaliteta**

Uvodi se statička ocena kvaliteta dokumenta (nezavisna od upita). Elementi u posting listama se uredjuju u opadajućem redosledu prema kvalitetu dokumenta. Procesira se, ponovo, samo prvih r dokumenata svake postng liste,moguća je paralelna obrada

- **Metoda sa podelom ulaznog skupa dokumenata na klastere**
  - Faza pretprocesiranja - Iz ulaznog skupa dokumenata slučajno se izabere ÖN dokumenata i proglase se liderima. Oko svakog lidera se formira klaster tako što se za sve ostale dokumente iz korpusa izračuna najbliži (najsličniji) lider
  - Faza traženja: Za zadati upit se odredi najbliži (najsličniji) lider, iz korpusa najbližeg lidera odredi se k dokumenata najsličnijih upitu

7. **Navesti mere koje se koriste za evaluaciju IR sistema**

Kao parametri koji se koriste za evaluaciju IR sistema javljaju se odziv i preciznost. Odziv predstavlja odnos broja izdvojenih relevantnih dokumenata i ukupnog broja relevantnih dokumenata. Preciznost predstavlja odnos broja izdvojenih relevantnih dokumenata i ukupnog broja izdvojenih dokumenata. Za uske upite preciznost je velika, a odziv mali, dok je kod sirih upita ovo obrnuto. Sto nas dovodi do zakljucka da su ova dva parametra reciprocna. Da li povecati odziv ili preciznost zavisi od potreba korisnika.

Težinska harmonijska sredina odziva i preciznosti:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)P \cdot R}{\beta^2 \cdot P + R}$$

gde je $\beta^2 = \frac{1-\alpha}{\alpha}$, $\alpha \in [0,1]$ pa je prema tome $\beta \in [0,\infty]$.

Za $\alpha = \frac{1}{2}$ je $\beta = 1$ i dobija se balansirana F-mera:

$$F_{\beta=1} = \frac{2 \cdot P \cdot R}{P+R}$$

8. **Objasniti postupak testiranja IR sistema. Šta standardna kolekcija za testiranje treba da sadrži?**
   Standardna kolekcija za testiranje treba da sadrzi dovoljno veliki i dovoljno raznovrstan korpus dokumenata, dovoljno veliki i dovoljno reprezentativan skup upita, za svaki par(dokument, upit) procena relevantnosti(obicno binarna). Treba da sadrzi i listu relevantnih dokumenata za svaki upit.

IV grupa: Pretrazivanje weba

1. **Sta cini pretrazivanje weba komplikovanijim u odnosu na pretrazivanje poznatog korpusa dokumenata?**
   Web je jedna ogromna zbirka nestruktuiranih i nepoznatih elemenata, razlicitih formata, pisanih na razlicitim jezicima i distribuirana na ogromnom broju servera. Sve to cini pretrazivanje web-a znatno tezim u odnosu na struktuirane, poznate i lokalizovane dokumente. Velika distribuiranost, heterogenost, permanentna promena sadrzaja, relevantnost informacija na web-u, kolicina relevantnih rezultata u odnosu na zadati upit.

2. **Sta je SEO? Navesti i kratko objasniti tehnike koje SEO koristi.**

   Spam ili SEO obuhvata skup metoda za manipulisanje sadrzajem Web stranica kako bi se nasle na vrhu rang liste algoritamskog trazenja po zadatim kljucnim recima. Spam tehnike: Cloacking-vracanje razlicitog sadrzaja zavisno od toga da li stranici pristupa web crawler ili obican korisnik, Doorway stranica-stranica sa kratkim sadrzajem(obicno sa samo nekoliko kljucnih reci) koje sadrze link do prave stranice, Link spamming- ubacivanje vidljivih i nevidljivih linkova na svojim i tudjim web stranicama do stranica ciji se rejting podize.

3. Kako se vrsi procena kvaliteta indeksiranja weba?

Fizicka velicina indeksa nije dovoljan pokazatelj kvaliteta Web pretrazivaca. Za pretrazivace Web-a cemo
pod velicinom indeksa podrazumevati procenat celokupnog Weba koji je indeksiran. To je nemoguce precizno odrediti jer tacna velicinaWeba nije nepoznata i postoji beskonacno mnogo dinamickih stranica. Jedino mozemo uporediti velicine delova Weba koji su indeksirani razlicitim pretrazivacima(capture-recapture metod). – neka je E1 indeks prvog pretrazivaca, a E2 indeks drugog, odabiramo slucajnu stranicu iz E1 i ispitujemo da li se nalazi i u indeksu pretrazivaca E2, istovremeno ispitujuci da li se slucajno odabrana stranica iz E2 nalazi u E1. Na osnovu navedene procedure procenjujemo koliki se deo indeksa E1(x) nalazi u indeksu E2 i koliki se deo indeksa E2 nalazi u indeksu E1(y). Odnos x/y predstavlja odnos velicina indeksa ova dva pretrazivaca.

4. Navesti i kratko objasniti metode za prepoznavanje duplikata na Webu. $J(S(d_1), S(d_2)) = \frac{|S(d_1) \cap S(d_2)|}{|S(d_1) \cup S(d_2)|}$

Metoda otiska prsta - za svaki dokument se kreira hash kljuc(otisak) obicno 64-bitni, duplikati se otkrivaju poredjenjem otisaka. Nedostatak ovog metoda je to sto su otisci isti samo ako su dokumenti identicni.
Shingling metoda – k-shingles dokumenta d (skup svih uzastopnih nizova k termina u d). S(dj) – skup shingl-ova dokumenata dj. Jaccardov koeficijent – meri stepen poklapanja izmedju dva dokumenta. 2 dokumenta se smatraju duplikatima ako je on veci od nekog praga, npr 0.9.

5. Sta je crawler?

Crawler(spider) je program koji prikuplja stranice sa interneta radi njihovog indeksiranja u okviru Web pretrazivaca. **Obavezne osobine crawlera:**
Robusnost – crawleri moraju biti dizajnirani tako da budu otporni na zlonamerno generisanje web stranice koje imaju za cilj onemogucavanje rada crawler-a
Uctivost – web server imaju implicitne I eksplicitne propise koji regulisu koliko cesto crawleri mogu da ih posecuju
Pozeljne osobine:
Distribuiranost - treba da ima mogućnost izvršavanja na distribuiran način
Skalabilnost – arhitektura treba da omoguci povecanje kapaciteta dodabanjem dodatnih masina
Performanse i efikasnost – maksimalno iskoriscavanje sistemskih resursa
Kvalitet – trebalo bi da daje prioritet prikupljanju korisnih web stranica
Azurnost – ponovno posecivanje vec pribavljene stranice radi dobijanja svezih kopija
Prosirljivost – razresavanje pitanja novih formata podataka, protokola za prikupljanje podataka…

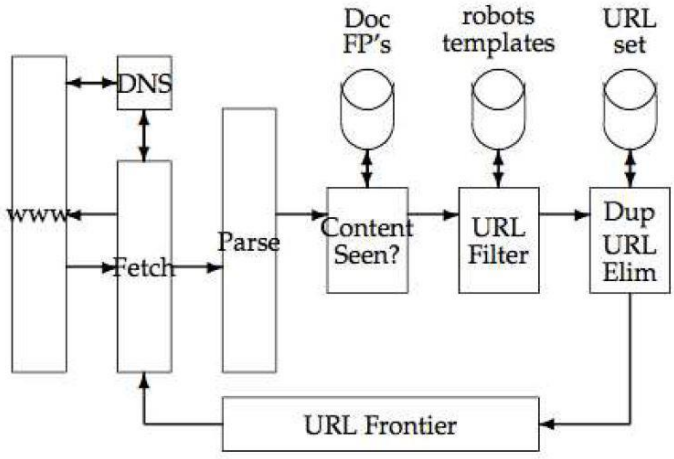
6. Sta defnise fajl robot.txt?
Web serveri imaju i implicitne i eksplicitne propise koji regulisu koliko cesto crawleri mogu da ih posecuju. Sto se tice eksplicitnih propisa, u root direktorijumu web sajta se nalazi fajl "robots.txt" u koje se definise koji crawleri mogu pristupati kojim delovima sajta.
User-agent: *
Disallow: /yoursite/temp/
User-agent: searchengine
Disallow:

7. Objasniti algoritam crawl-ovanja.

Inicijalni skup URL-ova upisati u frontier (frontier-struktura podataka koja cuva URL adrese neposecenih stranica i koja radi na principu redova cekanja), zatim treba ponavljati sledece korake:

- uzeti sledeci URL iz frontiera
- pribaviti stranicu sa date URL adrese
- proveriti da li je ta stranica vec pribavljena
- parsirati pribavljenu stranicu, tj. izdvojiti tekst i hiperlinkove
- tekst proslediti indekseru
- hiperlinkove upisati u frontier

8. Objasniti strukturu web crawlera.



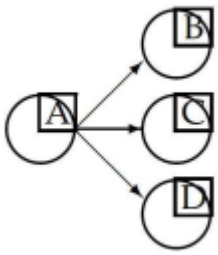
- Modul za DNS razresenje-odredjuje adresu web servera na kome se nalazi trazeni URL
- Modul za pribavljanje-koristi http protokol za pribavljanje web stranica sa URL-a
- Modul za parsiranje-izdvaja tekst i skup linkova iz pribavljene stranice(vrsi i normalizaciju linkova, linkovi koji vode do stranica sa istog sajta su obicno dati kao relevantni)
- Model za eliminisanje duplikata-odredjuje da li se izolovani link vec nalazi u URL frontieru ili tek treba da bude dodat

9. Objasniti PageRank algoritam za rangiranje stranica na osnovu linkova.

Bazira se na izracunavanju verovatnoce da odredjena stranica bude posecena. Stranica moze da bude posecena na dva nacina: koriscenjem nekog od linkova koji vode do nje ili "teleport" akcijom odnosno direktnim kucanjem njene adrese u pretrazivac bez obzira na to koja je stranica prethodno bila posecena.

Korišćenjem linkova:

Posmatramo deo Web grafa
Ako je trenutno posećena stranica A, verovatnoća da u sledećem trenutku bude posećena neka od stranica B, C ili D praćenjem linkova je 0.33.



Korišćenjem “teleporta”: Verovatnoća posete svih stranica je ista (1/N)

10. Objasniti HITs algoritam za rangiranje stranica na osnovu linkova

Za svaku stranicu se uvode dve mere kvaliteta: **autoritativnost** (autoritativni rejting)-odredjena brojem linkova koji vode do nje, **povezanost** (vezni rejting)-odredjena brojem i kvalitetom stranica do kojih vode njeni linkovi. Sve stranice na webu se dele u dve kategorije: **autoritete** (stranice sa velikim autoritativnim rejtingom)-smatra se da pruzaju kvalitetne informacije, **stranice za vezu** (stranice sa velikim veznim rejtingom)-smatra se da sadrze linkove do dobrih autoriteta.
Autoritativni rejting se izračunava kao suma veznih rejtinga stranica koje sadrže linkove do posmatrane strane $a(x) = \sum_{y \to x} h(y)$. Vezni rejting se izračunava kao suma autoritativnih rejtinga stranica do kojih vode linkovi sa posmatrane strane $h(x) = \sum_{x \to y} a(y)$. Iterativni postupak za njihovo izracunavanje:
Obelezimo sa A matricu povezanosti Web grafa. Tada je:

$$\vec{h} = A \cdot \vec{a} \qquad \overrightarrow{h_k} = A \cdot A^T \cdot \overrightarrow{h_{k-1}}$$

$$\vec{a} = A^T \cdot \vec{h} \quad \text{i} \quad \overrightarrow{a_k} = A^T \cdot A \cdot \overrightarrow{a_{k-1}}$$

Za pocetne vrednosti vektora a i h uzima se $\overrightarrow{a_0} = \overrightarrow{h_0} = (1\,1\,1 \ldots 1)$