Algoritam rangiranja

Faktori koji utiču na rang

- Broj pojava svake tražene reči u stranici
- Linkovi između stranica, bez uticaja broja reči
- Linkovi između stranica, sa uticajem broja reči
- Broj pronađenih reči u OR upitima

Broj pojava traženih reči

- U čvorovima trie stabla se nalaze skupovi stranica koji sadrže određenu reč, gde je uz svaku stranicu pridružen broj pojava reči
- U skupu svih stranica koji se čuva za operaciju komplementa, pridruženi broj reči uz svaku stranicu je 0
- Skup je implementiran tako da se ukupni broj traženih reči u rezultujućem skupu automatski računa tokom primene skupovnih operatora
  - Pri operaciji unije se sabiraju pomoćne vrednosti za stranice koje se nalaze u oba skupa, a čuvaju stare pomoćne vrednosti za stranice koje se nalaze u jednom skupu
  - Pri operaciji preseka se sabiraju pomoćne vrednosti
  - Pri operaciji razlike se uzima pomoćna vrednost iz levog operanda

- Neka su date stranice A, B, C, D i skupovi stranica koji sadrže određene reči:

  word1 = {A: 10, B: 20, C: 5}

  word2 = {B: 15, C: 10, D: 17}

  word3 = {A: 40}

  word4 = {C: 2, D: 50}

- Skup svih stranica je:

  AllPages = {A: 0, B: 0, C: 0, D: 0}

Izračunati broj pojava traženih reči u pojedinim upitima za dati primer

- *word1 OR word2* u osnovnoj pretrazi, *word1 || word2* u naprednoj

  word1 | word2 = {A: 10, B: 35, C: 15, D: 17}

- *word1 AND word2* u osnovnoj pretrazi, *word1 && word2* u naprednoj

  word1 & word2 = {B: 35, C: 15}

- *!word2* u naprednoj pretrazi

  allPages – word2 = {A:0}

- *word1 NOT word2* u osnovnoj pretrazi, *word1 && !word2* u naprednoj

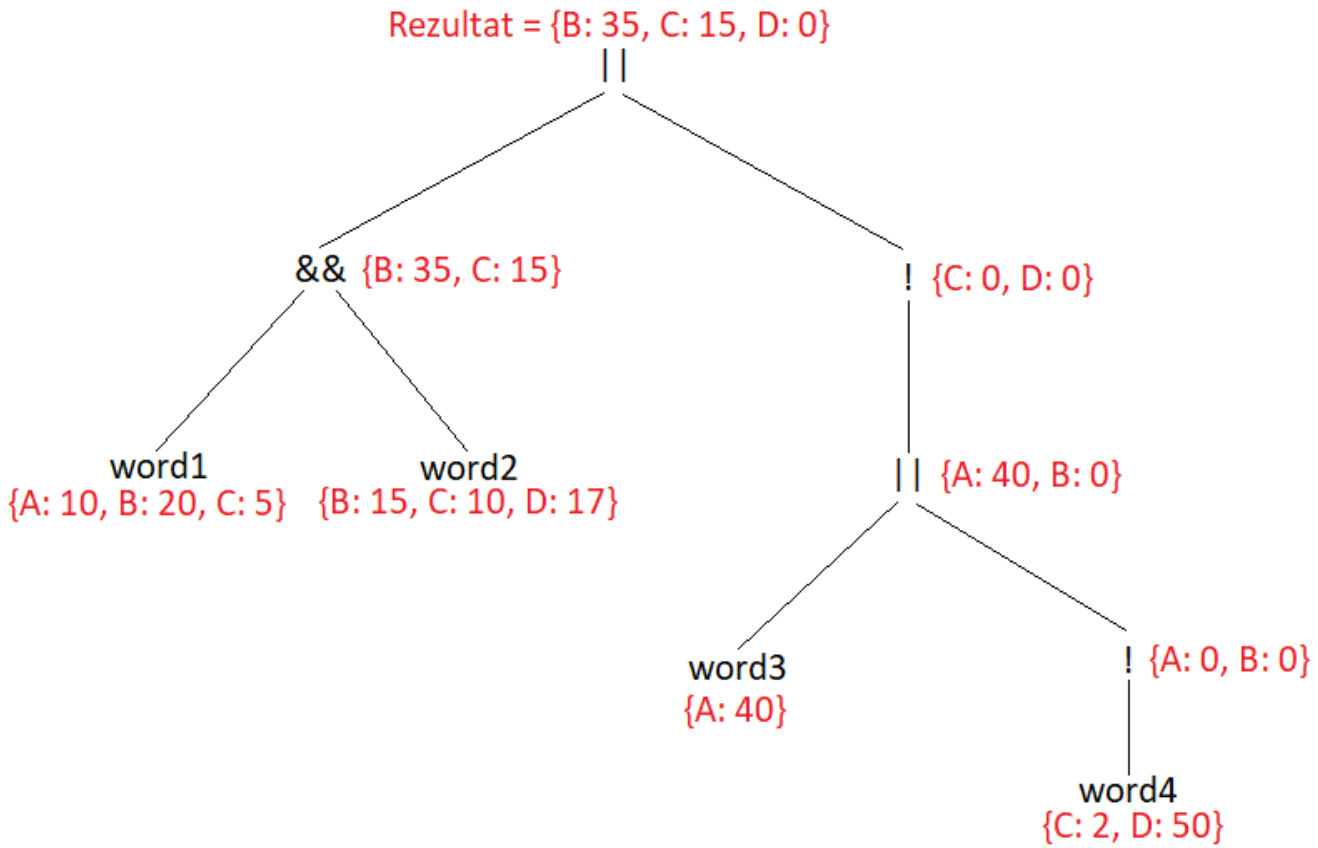
  word1 - word2 = {A: 10}

- *word1 word2 word3 word4* u osnovnoj i naprednoj pretrazi

  word1 | word2 | word3 | word4 = {A: 50, B: 35, C: 17, D: 67}

Primer računanja ukupnog broja pojava traženih reči u stablu složenog upita

- *word1 && word2 || !(word3 || !word4)*



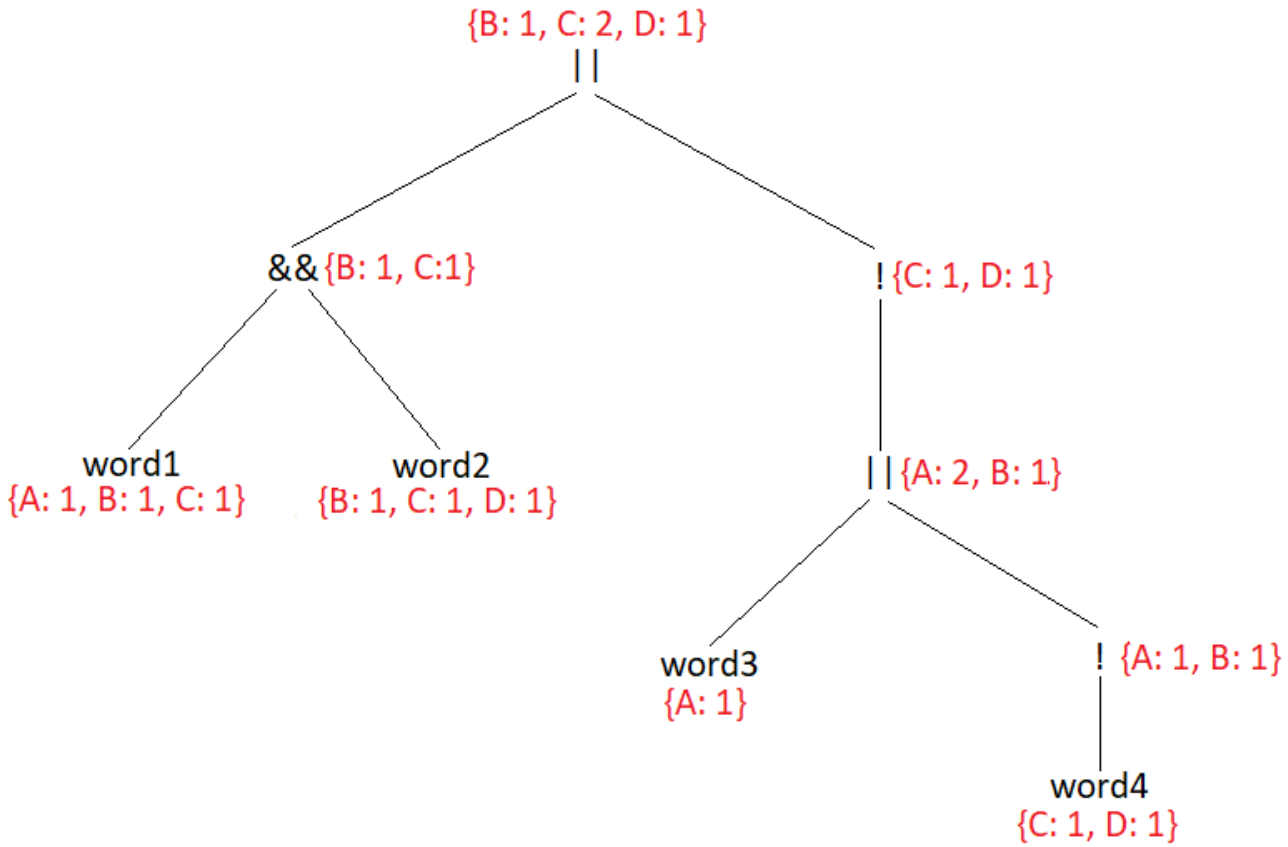
Broj pronađenih reči u OR upitima

- U upitima tipa *word1 word2 word3* ima smisla da stranice koje sadrže samo jednu od traženih reči, budu slabije rangirane od stranica koje sadrže više traženih reči
- Ideja: dodati još jednu pomoćnu vrednost koja se čuva uz stranice, koristiti je za brojanje skupova u kojima se stranica pojavila u OR upitima, te skalirati ukupni rang stranice proporcionalno izračunatoj vrednosti
- Definisati operacije nad pomoćnom vrednošću tako da upiti tipa *word1 NOT word2* i *word1 AND word2* ne izazivaju skaliranje (jer stranica može ispuniti samo čitav takav upit, nikako samo jedan njegov deo)
- Moguće tumačenje u stablima upita napredne pretrage: favorizovati stranice koje zadovoljavaju više čvorova stabla, ako je stablo takvo da čvor ne mora zadovoljavati sve čvorove

Rešenje korišteno za tretiranje pomoćne vrednosti

- Inicijalno se uz svaku stranicu u skupovima trie stabla čuva vrednost 1 – to naznačava da je uzimanjem tog skupa, stranica ispunila 1 uslov upita
- U skupu svih stranica takođe se svakoj stranici dodeljuje vrednost 1
- Tretiranje pri skupovnim operacijama:
  - Pri operaciji unije, sabiraju se vrednosti stranica koje se nalaze u oba skupa, a čuvaju stare vrednosti stranica koje se nalaze u jednom skupu
  - Pri operaciji preseka, između vrednosti stranice u levom i desnom operandu bira se veća, i ona se uzima – ovako se osigurava da će stranica imati pomoćnu vrednost 1 u osnovnim upitima tipa *word1 AND word2*, i da će se relativno jednako tretirti podstabla kod složenih upita
  - Pri operaciji razlike uzima se vrednost iz levog operanda

Primer računanja OR rezultata u stablu kompleksnog upita



Problemi sa pomoćnim vrednostima u stablima kompleksnih upita

- Računanje podataka za rangiranje upotrebom pomoćnih vrednosti tokom izvršavanja skupovnih operacija može izazvati problem prilikom evaluacije stabla međukoda kompleksnog upita
- Moguće je napisati upite koji su logički ekvivalentni, ali sadrže različite operacije
- Zbog osobina operacije komplementa i načina računanja podataka za rangiranje, može doći do gubitka podataka za rangiranje pri evaluaciji stabala takvih upita
- Posledica: rezultujući skup sadrži iste stranice, ali nisu isto rangirane
- Dve vrste problema:
  - Upiti sa uzastopnim operatorima komplementa
  - Upiti koji se mogu transformisati De Morganovim zakonima

Upiti sa uzastopnim operatorima komplementa

- Primer:

  allPages = {A: (0, 1), B: (0, 1), C: (0, 1), D: (0, 1)}, word = {A: (10, 1)}

  word: rezultat = {A: (10, 1)}

  !(!word): rezultat = allPages - (allPages - word) = {A: (0, 1)}

- Rešenje: transformacija stabla međukoda uklanjanjem suvišnih NOT čvorova
- Radi se depth-first prolaz kroz stablo, i traže se NOT čvorovi čiji je operand takođe NOT čvor
- Ako se pronađe, oba NOT čvora se uklanjaju iz stabla

Upiti koji se mogu transformisati De Morganovim zakonima

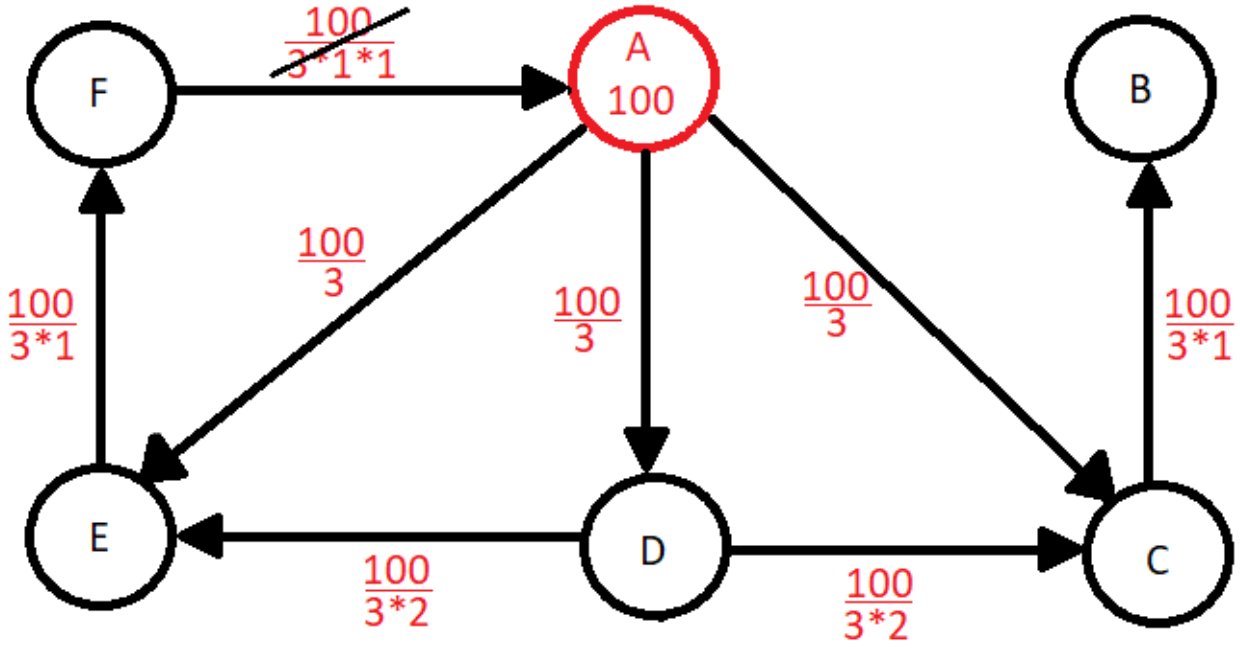
- Upit tipa *!(left && right)*, ekvivalentan sa *!left || !right*
- Izaziva različito tretiranje pomoćne vrednosti za favorizovanje stranica zbog OR upita
- Može sakriti uzastopne operatore komplementa - npr. *!(!left && right)*
- Upit tipa *!(left || right)*, ekvivalentan sa *!left && !right*
- Može sakriti uzastopne operatore komplementa
- Rešenje: transformacija stabla međukoda primenom De Morganovih zakona
- Pronalaze se čvorovi koji odgovaraju ovim upitima, i zamjenjuju ekvivalentima, što izaziva postepeno pomeranje operatora komplementa prema listovima stabla

Propagacija značaja stranice kroz linkove

- Svakoj stranici skupa koji se rangira dodeljen je neki značaj, koji se kroz linkove propagira na ostale stranice
- Za svaku stranicu vrši se breadth-first obilazak grafa linkova
- Ako stranica od koje se počinje obilazak ima značaj R i L linkova, rangu svake stranice koju ona linkuje dodaje se vrednost R/L
- Linkovane stranice na isti način propagiraju dalje vrednost R/L na stranice koje one linkuju
- Svaka veza u grafu se obilazi maksimalno jednom – to znači da iako jednoj stranici može biti dodato više vrednosti na rang u toku jednog obilaska, ona će dalje na stranice koje linkuje propagirati samo prvu (zbog breadth-first obilaska, to će biti vrednost koju je dobila najkraćom putanjom od početnog čvora)

Primer propagacije značaja jedne stranice kroz graf

- Obilazak grafa od čvora A
- Na granama je napisana vrednost koja se dodaje rangu linkovane stranice
- Stranice E i C dobijaju vrednost i od A i od D, ali dalje propagiraju samo vrednost koju su dobile kraćom putanjom
- Vrednost koju propagira stranica F se poništava, jer se smatra da stanica A ne može uticati na vlastiti rang



Performanse propagacije značaja kroz graf

- Obilazak čitavog grafa je kompleksnosti **O(v + e)**, gdje je *v* broj čvorova, a *e* broj grana grafa
- Za računanje uticaja linkova na rang skupa stranica od *n* elemenata potrebno je napraviti *n* obilazaka grafa, pa je ukupna kompleksnost **O(n*(v + e))**
- Ovakva kompleksnost nije prihvatljiva za potrebe brzog rangiranja rezultata pretrage, pa se uvodi dodatni parametar *d* koji ograničava dubinu obilaska grafa
- Za *d = 1*, za svaku stranicu se obilaze samo njeni direktni linkovi, pa je ukupna kompleksnost **O(n + e)**
- Za *d*->∞, kompleksnost teži ka **O(n*(v + e))**
- Za trenutnu implementaciju i test skup, utvrđeno je da je prihvatljiva vrednost parametra *d* maksimalno 2

Uticaj linkova stranica relevantnih za pretragu

- Vrši se propagacija značaja stranica kroz linkove
- Kao značaj stranice uzima se prethodno izračunati ukupni broj pojava traženih reči
- Samo stranice koje se nalaze u rezultujućem skupu imaju značaj, pa se samo za njih pokreću obilasci grafa
- Rezultat propagacije predstavlja jednu od komponenti konačnog ranga

Uticaj linkova svih stranica

- Vrši se propagacija značaja stranica kroz linkove
- Svakoj stranici iz skupa svih stranica dodeljuje se jednak značaj, koji se zatim propagira
- Ova komponenta ranga je konstantna, pa se računa samo jednom, prilikom učitavanja direktorijuma
- Značaj ove komponenete je prvenstveno za rangiranje stranica koje su rezultat upita tipa *!word*, jer su za takve upite broj reči i propagacija broja reči 0, a uticaj OR upita 1 za sve stranice

Konačni rang

- Neka su izračunate komponente ranga stranice označene sa:
  - Ukupni broj pojava reči - *WordScore*
  - Vrednost koju je stranica dobila propagacijom broja pojava reči kroz graf – *RelevantLinkScore*
  - Vrednost koju je stranica dobila propagacijom konstanti kroz graf – *GeneralLinkScore*
  - Vrednost koju je stranica dobila od OR upita – *OrScore*

- Prve tri komponente se skaliraju tako da nose određeni procenat od ukupnog ranga
- Ako je oznaka komponente *Score*, a procenat koji ona nosi *ScoreInfluence*, skaliranje se vrši po formuli:

$$ScoreScaled = ScoreInfluence * Score / Score_{max}$$

- U trenutnoj implementaciji, podrazumevani procenat koji komponente nose je:
  - WordInfluence = 50%
  - RelevantLinkInfluence = 40%
  - GeneralLinkInfluence = 10%
- Korisniku je omogućeno da menja ove parametre

- Konačni rang se računa po formuli (za komponente se podrazumeva da su skalirane):

$$\text{BaseRank} = \text{WordScore} + \text{RelevantLinkScore} + \text{GeneralLinkScore}$$

$$\text{Rank} = \text{BaseRank} * (1 + (\text{OrScore} - 1) * \text{OrWeight})$$

- *OrWeight* je takođe parametar koji korisnik može da izabere, uz podrazumevanu vrednost 0.5