Iskanje informacij po horizontalno porazdeljenih virih

Sandi Pohorec, Milan Zorman
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko
sandi.pohorec@uni-mb.si, milan.zorman@uni-mb.si

Povzetek

Uporaba spletnih iskalnikov je postala vsakodnevna stalnica. Pogosto so spletni iskalniki edini način iskanja informacij. Spletnim mestom, ki želijo svojim uporabnikom ponuditi napredne iskalne storitve, pogosto ne zadostuje uporaba storitev večjih iskalnikov (Google, Yahoo, Live Search). Razlogi za izgradnjo lastnih iskalnih storitev so predvsem v varnosti, želeni ravni poosebitve ter seveda v potrebi po prilagojenem ocenjevanju in razvrščanju rezultatov. Prispevek predstavi delovanje sodobnih iskalnikov in podrobneje opiše iskalno storitev, ki je bila razvita z vidika posameznega spletnega mesta. Iskalni zadetki se avtomatsko prilagajajo uporabniku in njegovim pravicam, ponujeno pa je tudi iskanje po sorodnih spletnih mestih ter strukturiranih in nestrukturiranih virih podatkov. Iskalna storitev ponuja prav tako poosebljanje iskalnih rezultatov glede na interesna področja uporabnika.

Ključne besede: iskalna storitev, splet, portal, porazdeljeni viri, iskanje informacij, iskalni indeksi, spletni pajki, podatkovne baze.

Abstract

SEARCHING FOR INFORMATION ON HORIZONTALLY DISTRIBUTED SOURCES

In the modern society the use of online search engines is a daily routine. More often than not it is the only way of getting the desired information instantly. For a large web portal, with the ambition to provide their users with advanced search services, the feature specifications of the search services they require far exceed the capabilities provided by large search engines (Google, Yahoo, Live Search). The main initiative for a custom search service is the need for security and authorized access to certain content; also the desired personalization level and a customized ranking of search results are both major contributors. The paper reviews the architecture of modern search engines and presents in greater detail the search service that was developed. The service provides automated use of users' credentials to limit the search results to only the content that the user has been granted access to; it provides centralized search capability across multiple web sites with similar content as well as structured and unstructured data sources. The search service also provides personalization according to the users' field of interest.

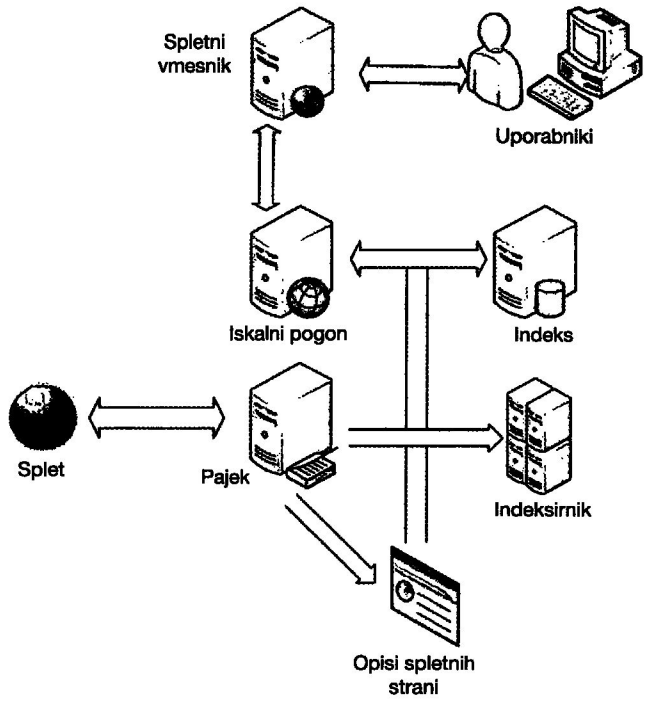
Key words: search engine, web, portal, distributed data sources, information discovery, search index, web crawlers, databases.

1 UVOD

Pod besedo iskalnik razumemo spletni iskalnik, ki išče po javno dostopnih straneh svetovnega spleta. Druge vrste iskalnikov so intranetni iskalniki, ki preiskujejo zaprto omrežje organizacije ali podjetja, osebni iskalniki, mobilni iskalniki idr. Za različna okolja in različne uporabnike se uporabljajo različni postopki in kriteriji izbire zadetkov ter njihovega urejanja po pomembnosti. Večji spletni iskalniki, kot je Google [1], delujejo po lastnem algoritmu, ki je splošen in ni prilagojen za posebnosti posameznih spletnih strani oz. aplikacij. Naš cilj je bil razviti iskalno storitev, ki se bo zavedala notranje strukture svojih (porazdeljenih) virov, omogočala različne ravni iskanja glede na pravice uporabnikov ter izboljšala najprej širino in nato še natančnost iskanja. Da lahko razvijemo lastno iskalno storitev, moramo seveda razumeti delovanje obstoječih iskalnikov, ki ga opišemo v drugem poglavju. V tretjem predstavimo lastno rešitev, ki išče po porazdeljenih spletnih virih. Prispevek zaključimo s primerjavo prednosti in slabosti našega pristopa glede na sorodne.

2 DELOVANJE SODOBNIH ISKALNIKOV

Sodobne iskalnike sestavljata dve ločeni aplikaciji. Prva je spletni pajek, ki skrbi za polnjenje in posodabljanje iskalnega indeksa s podatki indeksiranih spletnih strani, druga pa je sam iskalnik, ki uporablja iskalni indeks za izvedbo iskanj. Aplikaciji sta šibko sklopljeni, edina trdna vez med njima je iskalni indeks, ki zagotavlja, da sta medsebojno dokaj neodvisni; spremembe notranjih podatkovnih struktur, funkcij in načinov delovanja pri eni aplikaciji pa ne vplivajo na delovanje druge. Drugi vzrok za ločitev v dve aplikaciji je velika časovna zahtevnost preiskovanja vsebin spletnih strani. Splošno arhitekturo sodobnih iskalnikov prikazuje slika 1.



Slika 1: **Splošna arhitektura sodobnega iskalnika**

Iskalniki imajo podobno strukturo kot sistemi podatkovnih baz. O hranjenih dokumentih vodijo indeks. Povpraševanja se izvajajo s preiskovanjem indeksa, ki vrača rezultate, ki jih iskalnik prikaže uporabniku. Hkrati pa obstajajo številne razlike med strukturo obeh sistemov: podatkovne baze morajo podpirati kompleksna povpraševanja, medtem ko je večina povpraševanj iskalnika vezana samo na seznam izrazov, besednih zvez v naravnem jeziku. V sistemu podatkovne baze je zadetek vrstica, ki ustreza navedenemu pogoju, pri iskalniku pa je zadetek dokument, ki ustreza iskanemu nizu po statistični hevristiki in morda niti ne vsebuje vseh iskanih besed oz. fraz. Podatkovne baze vračajo vse vrstice, ki ustrezajo, iskalnik pa vrača omejeno število zadetkov, ki so urejeni glede na statistično podobnost. Sistem za upravljanje s podatkovno bazo določi univerzalni ključ za dostop (primarni ključ tabele) in omogoča iskanje po tem ključu. Preiskovanje dokumentov pomeni, da lahko obstaja nič ali mnogo dokumentov, ki imajo neničelno podobnost z iskanim nizom. Iskalni niz se lahko pojavlja v velikem številu dokumentov in vsak dokument po navadi vsebuje veliko število nizov (besed, terminov, fraz). Izzivi, ki jih predstavlja preiskovanje tekstovnih vsebin, so vodili k razvoju širokega nabora algoritmov in podatkovnih struktur. Ti vključujejo predstavitve za tekstovne indekse, tehnike za ustvarjanje indeksov in algoritme za ovrednotenje besedilnih povpraševanj.

V osnovi vsak iskalnik deluje v treh zaporednih korakih:

- preiskovanje vsebine oz. spleta (angl. Web crawling),
- indeksiranje in
- iskanje.

Poglejmo si podrobneje delovanje vsake od treh najpomembnejših komponent iskalnika.

Spletni pajek

Kot smo že omenili, metodično preiskovanje spleta (v obliki prevedenega programa ali interpretirane skripte) izvaja spletni pajek. Proces se angleško imenuje Web crawling ali tudi spidering. Spletni pajki so uporabljeni za ustvarjanje (bolj ali manj celovite) kopije vseh obiskanih strani za kasnejšo obdelavo. Obdelava pomeni shranjevanje v ustrezne podatkovne strukture iskalnika oz. v njegov indeks. Tri pomembne lastnosti spleta so vzrok za izjemno oteženo preiskovanje:

- velike količine vsebin (več terabytov),
- hitrost sprememb podatkov in
- dinamično ustvarjanje spletnih vsebin.

Čedalje več dinamičnih spletnih strani otežuje učinkovito zajemanje (indeksiranje) vsebin spletišč. Strani so zelo pogosto grajene na osnovi skriptnega jezika na strežniški strani. Obstaja namreč velika možnost, da različne kombinacije parametrov zahteve HTTP GET vrnejo enako vsebino. Ker pasovna širina, ki jo uporabljajo pajki za pregledovanje vsebin, ni niti neskončna niti brezplačna, postaja izjemno pomembno, da pajki pregledujejo vsebine razširljivo in učinkovito. Pajek mora na vsakem koraku previdno izbrati, katere strani bo obiskal v nadaljevanju.

Najpomembnejši dejavnik pri razumevanju delovanja pajka je njegova usmeritev na celotni svetovni splet, določene domene, določeno jezikovno območje ali samo na določeno spletno mesto. Strategije, ki omejujejo količino indeksiranih strani, so omejevanje sledenja povezavam, preiskovanje s spuščanjem po vsaki poti, usmerjeno preiskovanje in preiskovanje »globokega«, nevidnega dela spleta. Ažurnost iskalnega indeksa je odvisna od izbranega časovnega intervala za ponovne obiske že indeksiranih spletnih strani. Obnašanje ob samem indeksiranju določajo t. i. pravila za vljudno delovanje, ki upoštevajo predpise posameznih spletnih mest. Predpisi določajo, s

kakšno hitrostjo naj se izvaja indeksiranje, da ne pride do preobremenitve spletnega mesta. Seveda je delovanje odvisno tudi od stopnje paralelizma delovanja spletnega pajka, tj. koliko spletnih pajkov deluje vzporedno in kako si delijo breme.

Iskalni indeks

Cilj shranjevanja indeksa je optimizacija hitrosti in zmogljivosti iskanja dokumentov, ki se ujemajo s poizvedbo. Brez indeksa bi iskalnik preiskoval vse dokumente v korpusu, kar bi zahtevalo veliko časa in računske moči. Npr. za 10 000 dokumentov bi preiskovanje indeksa trajalo nekaj milisekund, zaporedno preiskovanje vsebine vsakega od njih pa bi vzelo ure. Nekatere najbolj pomembne lastnosti indeksa so: način polnjenja in posodabljanja indeksa (ali je možno hkratno delo več pajkov), tehnika fizičnega shranjevanja (ali je uporabljeno stiskanje podatkov), velikost indeksa (potrebna količina shranjevalnega medija), hitrost iskanja (čas za iskanje niza v inverznem seznamu), razmerje med časom iskanja in časom vstavljanja v indeks, kompleksnost vzdrževanja indeksa skozi daljše časovno obdobje in odpornost na napake. Podatkovne strukture, ki hranijo iskalni indeks, so lahko korenska drevesa, drevesa, sprednji indeksi (primer prikazuje tabela 1), inverzni indeksi (primer prikazuje tabela 2), indeksi citatov oz. sklicevanj, matrike termin-dokument idr.

Tabela 1: **Primer sprednjega indeksa**

Dokument	Besede
Dokument 1	poimenujejo, informatike, in
Dokument 2	in
Dokument 3	pojave, in
Dokument 4	informatike
Dokument 5	pojave

Tabela 2: **Primer inverznega indeksa**

Beseda	Dokumenti
poimenujejo	Dokument 1
pojave	Dokument 3, Dokument 5
informatike	Dokument 1, Dokument 4
in	Dokument 1, Dokument 2, Dokument 3

Iskanje

Iskalna poizvedba oz. iskalni niz je izraz, ki ga uporabnik vnese v spletni iskalnik z namenom zadovoljitve svojih potreb po informacijah. Značilno je, da poizvedbe niso strukturirane in imajo lahko več pomenov. S tem se močno razlikujejo od klasičnih povpraševalnih jezikov (SQL), ki temeljijo na strogih sintaktičnih pravilih. Trije najpogostejši tipi povpraševanj so informacijska povpraševanja (ustreza na tisoče zadetkov, npr. »avtomobili«), navigacijska povpraševanja (iskano je določeno spletno mesto, npr. »youtube«) in transakcijska povpraševanja (za pomoč pri izvedbi določene »transakcije«, npr. nakup avtomobila).

2.1 Semantični splet in iskalniki

Semantični splet in semantične spletne tehnologije nam ponujajo nov način upravljanja s podatki in procesi, ki temelji na ustvarjanju in uporabi semantičnih metapodatkov. Metapodatki obstajajo na dveh ravneh. Na prvi ravni opisujejo dokument, kot je spletna stran ali posamezni del tega dokumenta (npr. odstavek, povezava, tabela ...), na drugi pa opisujejo entitete znotraj dokumenta, kot so osebe ali ustanove. V vsakem primeru je pomembno, da metapodatki dodatno opišejo pomen dokumenta ali entitete znotraj dokumenta, tj. da nam dajejo informacije o vsebini dokumenta (tema dokumenta ali relacije z drugimi dokumenti) ali o entitetah znotraj dokumenta. Metapodatki v hipertekstovnih dokumentih, ki jih danes najdemo na svetovnem spletu, so zakodirani znotraj HTML-ja in opisujejo samo predstavitveni format (obliko, angl. design). Z uporabo HTML-ja lahko oblikujemo vsebino, ne moremo pa zapisati, da niz predstavlja ceno nekega produkta, avtorjevo ime in podobne lastnosti. Ko bodo znotraj hipertekstovnih dokumentov na voljo semantični metapodatki, bo mogoče vsebine uporabiti v novih storitvah, kot so:

- **Organizacija in iskanje informacij, osnovanih na pomenu.** Z uporabo semantike je mogoče razpoznati, katere besede ali fraze imajo enak pomen. Prav tako je mogoče ugotoviti, če imajo enake besede različen pomen. Torej, z iskanjem besede »Jaguar« (avtomobilska znamka in ne žival) lahko izločijo vse zadetke, ki vsebujejo informacije o velikih mačkah, in vrnejo samo tiste o avtomobilski znamki.
- **Z uporabo semantike lahko močno izboljšamo način predstavitve informacij.** Najosnovnejša izboljšava so rezultati, ki so razdeljeni v skupine glede na pomen, kar nam nadomesti linearni seznam zadetkov.
- **Uporaba semantičnih metapodatkov je ključna pri integraciji informacij znotraj ene organiza-**

cije med več organizacijami. Po navadi se uporabljajo različne sheme za opis in klasifikacijo informacij ter različne terminologije znotraj organizacij. Z ustvarjanjem preslikav med njimi je mogoča medopravilnost med procesi, ki uporabljajo informacije.

Glavno prednost semantičnega spleta za področje iskanja informacij vidimo predvsem v odpravi slabosti trenutnih iskalnikov. Iskalni nizi imajo lahko več pomenov. Ker konvencionalni iskalniki ne zmorejo interpretacije pomena uporabnikovega iskanja, vodi večpomenskost iskalnega niza do vračanja nepomembnih informacij. Čeprav se večpomenskost lahko razreši s skrbno izbranimi dodatnimi iskalnimi besedami, večina uporabnikov tega ne stori.

Večpomenskost predstavlja težavo, ker konvencionalni iskalniki primerjajo iskalni niz z indeksom, temelječem na ključnih besedah. Besede iskalnega niza so namreč lahko izrazito različne od besed v indeksu, čeprav imajo enak pomen (sopomenke). Običajni iskalniki se poleg nezmožnosti obravnave sopomenk in večpomenskih besed ne zavedajo tudi vseh drugih semantičnih povezav med koncepti. Če se osredinjimo na naslednji primer poizvedbe »telekomunikacijska družba Evropa Janez Novak direktor«, je očitno, da uporabnik išče vse dokumente, ki so povezani z direktorjem evropske telekomunikacijske družbe, ki se imenuje Janez Novak. Vendar iskalnik ne bi vrnil npr. dokumenta z naslednjim stavkom: »Imenovan je novi izvršni vodja ljubljanske izpostave Vodafone, J. Novak.« Da bi iskalnik vrnil navedeni dokument, bi se moral zavedati naslednjih semantičnih povezav: »Vodafone Live« je operater mobilne telefonije, kar je vrsta telekomunikacijske družbe; Ljubljana je v Sloveniji, ki je del Evrope; izvršni vodja je vrsta direktorja.

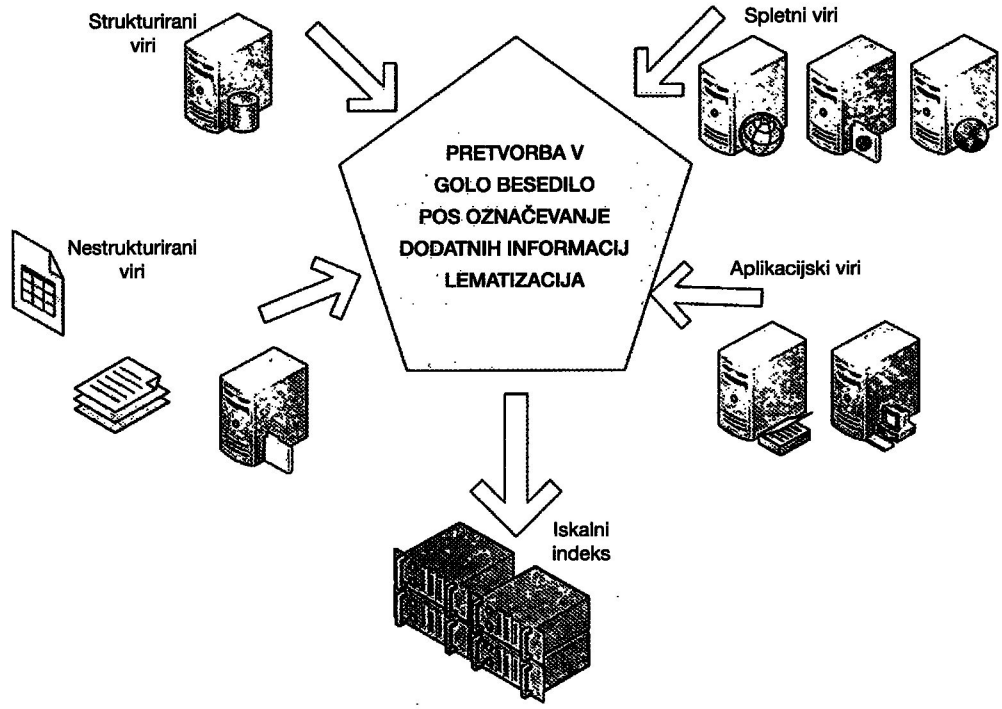
3 ISKANJE PO HORIZONTALNO PORAZDELJENIH SPLETNIH VIRIH

Naša rešitev ponuja celovito iskalno storitev za določeno spletno mesto. Iskanje poteka tako po vsebini samega mesta kot tudi po vsebini aplikacij, ki so vključene v spletno mesto (vendar niso neposredno njegov del), ter po vsebini drugih spletnih mest, ki so po vsebini sorodna našemu ali pa bi njihove vsebine lahko zanimale uporabnike našega spletnega mesta. Horizontalno porazdeljeni viri so medsebojno enakovredni viri, ki se razlikujejo glede na dostop (lokalni, oddaljeni), fizično obliko (spletna stran, dokument, podatkovna baza) in način predstavitve (v naravnem jeziku oz. nestruktrirani ali strukturirani). Pri klasičnih spletnih iskalnikih, kot je Google [1], so iskalni viri kar vse spletne strani. Mi pa ločimo med privzetim virom (samo spletno mesto) in dodatnimi iskalnimi viri, ki lahko predstavljajo druga spletna mesta, aplikacije, strukturirane in nestrukturirane vire ali dodatne statične spletne strani. Naš indeksirnik (aplikacija, ki prenaša indeksirane vsebine na naš strežnik in polni iskalne indekse) združuje zmogljivosti indeksiranja spletnih mest (pajek), indeksiranje podatkovnih baz in indeksiranje drugih strukturiranih (npr. XML datoteke) ter nestrukturiranih virov podatkov. Slika 2 prikazuje njegovo arhitekturo.

Indeksiranje spletnih mest upošteva navodila datoteke Robots.txt [2], če jo spletna mesta seveda imajo. Ker datoteka ponuja možnost prilagoditve za posamezne spletne pajke, nam tako spletna mesta, s katerimi imamo dogovor o sodelovanju, lahko omogočijo globlje indeksiranje njihovih vsebin. Za dodeljevanje nalog in razporejanje del indeksirnika skrbi posebna komponenta, ki ob vsakem času priskrbi seznam URL-jev, ki čakajo na indeksiranje. Za določanje, kdaj so potrebni ponovni obiski, uporabljamo splošno poznani dvojiški funkciji za oceno ponovnega obiska, imenovani *svežina* in *starost*. Definiciji obeh funkcij sta:

$Svežina_{stranx}(t)$ = {1; če je stran x enaka lokalni kopiji v času t | 0; sicer}
$Starost_{stranx}(t)$ = {0; če stran x v času t ni spremenjena | čas spremembe; sicer}

Indeksirnik ima po navadi odprtih več hkratnih povezav na različne strani, s čimer se doseže večja hitrost indeksiranja, obenem pa ne prihaja do večjih obremenitev indeksiranih spletnih strani. Pajek namreč ne sme preplavljati posameznih strani z velikimi količinami zahtev v kratkem času, zato med zahtevami na isto stran počaka določen časovni interval. Ta čas porabi za pošiljanje zahtev in obdelavo drugih odprtih povezav. Ker je na svetovnem spletu potrebno razpoznavanje različnih vsebin, mora pajek obravnavati različne možne napake v vsebini. Indeksiranje podatkovnih baz lahko izvajamo na dva načina: indeksiranje glede na opis podatkovne baze in popolnoma avtomatsko indeksiranje. V prvem načinu nam administrator podatkovne baze v obliki datoteke XML predpiše, katere tabele je treba inde-



Slika 2: **Arhitektura našega indeksirnika**

ksirati in (če želi) ročno označi prioriteto posameznih stolpcev v podatkovni bazi. Indeksirajo se samo naštete tabele in njihovi stolpci. Administrator lahko namesto sheme poda tudi stavekSQL, ki vrne želene stolpce ene ali več tabel. Tako lahko zagotovi lastno oblikovanje ali ponudi združene podatke iz različnih tabel kot eno samo. Kot bomo videli kasneje, so zadetki ob iskanju združeni v obliki datoteke XML. Ob opisu tabel lahko administrator podatkovne baze določi, kateri stolpec (stolpci) bo prikazan kot naslov, na katerega naj kaže povezava, in kako naj bo sestavljen opis zadetka (koliko besed levo in desno od iskalnega niza bo prikazanih, iz katerih stolpcev je sestavljen itn.).

Popolnoma avtomatsko indeksiranje pomeni, da naš indeksirnik preprosto indeksira vse tabele, v katerih se nahajajo stolpci s tekstovnimi podatki. Podatke poskuša združevati glede na shemo podatkovne baze. Shema relacijske podatkovne baze podrobno opiše strukturo podatkov in implementira konceptualne modele, kot sta ER (entitetno-relacijski model) in razredni diagram. Tako shema baze v bistvu opiše domenske koncepte (in povezave med njimi), ki so zajeti v podatkih podatkovne baze. Za avtomatsko indeksiranje podatkovnih baz so pomembni predvsem naslednji elementi, ki so del sheme: povezave med tabelami, osnovni podatkovni tipi atributov, omejitve atributov (npr. enoličnost) in primarni ter tuji ključi tabel. Ne glede na to, ali se podatkovne baze indeksirajo avtomatsko, ali glede na podani opis, jih najprej prenesemo na indeksirni strežnik in jih indeksiramo lokalno. Prenos podatkov je izveden prek spletne storitve, ki je optimizirana za te naloge in omogoča preverjanje integritete prenesenih datotek, primerjavo oddaljene in lokalne datoteke (če so od zadnjega obiska nastale kakšne spremembe v podatkih) in nadaljevanje prekinjenih prenosov.

Za spletna mesta, na katerih so potrebne prilagoditve indeksiranja, ki jih ni mogoče zapisati v shemo ali pa ne zadoščajo pravila v datoteki Robots.txt, ponujamo možnost vtičnikov (angl. plugins). Arhitektura indeksirnika je zasnovana tako, da je mogoče dinamično nalaganje programske kode. Posledično lahko za vsako spletno mesto ustvarimo popolnoma prilagojen indeksirnik. Možna je seveda vključitev dodatnih vtičnikov tudi v primeru, ko je iskalna storitev že nameščena na strežniku. Vsi iskalni viri so popolnoma nastavljivi prek administratorskega vmesnika in jih je mogoče v času izvajanja vključiti, izključiti ali nadomestiti. Komponenta indeksirnika, ki skrbi za pretvorbo iz različnih formatov v golo

besedilo, podpira veliko število različnih datotečnih formatov, med njimi izpostavimo:

- Microsoft Office Word (.doc),
- Microsoft Office Powerpoint (.ppt),
- Microsoft Office Excel (.xls),
- spletne strani in vsebina v označevalnem jeziku HTML (.htm, .html),
- Adobe Portable Document Format (.pdf) in
- Rich Text Format (.rtf).

Za pretvorbo uporabljamo enako tehnologijo kot indeksirne storitve operacijskega sistema Windows (»Desktop Search«), in sicer pretvorbo prek vmesnika IFilter. IFilter je standard, ki omogoča dostop do besedila dokumenta, če za določeni format zapisa obstaja implementacija »filtra«. Za implementacijo je takšna pretvorba dokaj zahtevna naloga, vendar ponuja univerzalno arhitekturo. Dodajanje novih formatov za pretvorbo namreč ne zahteva nikakršnih sprememb v indeksirniku. Vse kar je potrebno, je namestitev novega filtra na operacijski sistem strežnika, ki poganja indeksirnik.

Naš iskalni indeks je v osnovi sestavljen iz treh vrst indeksov:

- sprednji indeks (poda informacijo o tem, katere besede so v dokumentu),
- inverzni indeks (poda informacijo o tem, v katerih dokumentih je beseda) in
- slovar indeksiranih besed (poda informacijo o tem, ali se beseda sploh pojavi v indeksirani vsebini).

Podatkovna struktura osnovnih indeksov je razširjena oblika klasične matrike »termin-dokument«, ki se po navadi uporablja pri analizi skritih pomenov in hrani pojavitve besed v dokumentih v dvodimenzionalni matriki.

Poleg osnovnih uporabljamo še dokumentne in besedilne indekse. Oboji uporabljajo podatkovno strukturo, ki ni odvisna od posameznih virov vsebin.

Dokumentni indeksi hranijo:

- podatke o viru, kateremu dokument (spletna stran) pripada,
- v katerem datotečnem formatu je vsebina,
- spletno mesto, na katerem je vsebina dostopna,
- informacije o potrebni ravni dostopa za obisk vsebine in
- druge podatke.

Besedilni indeks hrani informacije o pomembnosti posameznih delov besedil v dokumentih. Večinoma lahko pomembnost posameznih delov razberemo že iz same strukture besedila. Implicitno je naslov pomembnejši kot besedilo iz vsebine, saj nudi več informacij o temi celotnega dokumenta. Ko gre za vir, ki je opisan s shemo, pa lahko iz sheme pridobimo pomembnost posameznih stolpcev. Ločimo absolutno (izračunani oceni se prišteje določeno število točk) in relativno pomembnost (prišteje se določen odstotek točk).

Samo zasnovo in zaporedne korake iskalnega algoritma smo povzeli po iskalniku Google. Ovrednotenje iskalnega niza in postopek iskanja, ki ga uporablja Google [4], se izvaja po naslednjem (poenostavljeno zapisanem) postopku:

- razpoznava iskalnega niza;
- pretvorba besed iz niza v »wordID-je«;
- pomik na začetek seznama dokumentov v »kratkem sodčku« za vsako besedo;
- preiskovanje po seznamu dokumentov, dokler ni najden dokument, ki se ujema z vsemi besedami iskalnega niza;
- izračun ocene tega dokumenta glede na iskalni niz po algoritmu »PageRank« (1);
- če smo v »kratkih sodčkih« in na koncu katerega koli seznama dokumentov, preskočimo na začetek seznama dokumentov v »dolgem sodčku« za vsako besedo in potem gremo na korak 4;
- če nismo na koncu katerega koli seznama dokumentov, gremo na korak 4. Dokumente, ki se ujemajo, uredimo po oceni in vrnemo prvih k dokumentov.

Očitno je, da je ključni del sistema ocenitvena funkcija. Google hrani več informacij o spletnih dokumentih kot drugi iskalniki. Vsak seznam zadetkov vključuje položaj, pisavo in informacijo o velikosti črk. Dodatno se upoštevajo zadetki v besedilu povezav in vrednost »*PageRank*« dokumenta (1). V enačbi *PR(A)* pomeni vrednost »*PageRank*« za spletno stran *A*, T_1 do T_n so spletne strani, ki imajo povezave na stran *A*, *C(A)* pa je število povezav, ki kažejo iz strani *A*. *d* je koeficient, ki ima lahko vrednost med 0 in 1, po navadi je nastavljen na 0.75.

$$PR(A) = (1-d) + d(PR(T1)/C(T1) + ... + PR(Tn)/C(Tn)) \quad (1)$$

Naše iskanje poteka po naslednjem algoritmu:

- Obdelava iskalnega niza.
- Za vsako besedo (ali celotni niz, če gre za frazno iskanje) se preveri ali obstaja v iskalnem indeksu in se pridobi njen univerzalni identifikator.

- Vrne se seznam dokumentov, ki vsebujejo iskalni niz (in ustrezajo uporabnikovi stopnji pravic).
- Za vsak dokument se izračuna ocena pomembnosti, ki služi kot kriterij za urejanje seznama (uporabi se metrika TFIDF, pomembnost vira, starost dokumenta, stopnje pomembnosti iz shematskih opisov, metrika medsebojne podobnosti dokumentov itd.).
- Za vsak dokument se poišče povzetek (del besedila, v katerem se nahaja iskalni niz).
- Seznam, dopolnjen s povzetki, prikažemo uporabniku.

Metrika TFIDF določa utež posamezne besede v določenem dokumentu, definirana je z naslednjo enačbo:

$$d(i) = TF(Wi,d)IDF(Wi), \quad (2)$$

kjer je $IDF(W_i) = \log \frac{D}{DF(W_i)}$,

kjer je
D – število dokumentov,
DF(W) – število dokumentov, v katerih se je beseda W pojavila vsaj enkrat, in
TF(W,d) – število ponovitev besede W v dokumentu d.

Dodatno k osnovnemu iskanju ponujamo tudi možnost naprednega iskanja. Uporabniki imajo možnost izbire med posameznimi viri iskanja in omejevanje iskanja glede na starost zadetkov. Izbirajo lahko tudi med fraznim (popolno ujemanje iskalnega niza) ali lematiziranim (uporabijo se oblikoslovne osnove posameznih besed iskalnega niza; leme) iskanjem.

Pogosti so primeri, ko imajo posamezna spletna mesta več virov. Zato smo uvedli koncept metaiskalnih virov. Za razlago vzemimo primer iskanja po digitalni knjižnici. Recimo, da ta ponuja dva iskalna vira: revije in knjige. Metaiskalni viri skrijejo arhitekturno zasnovo iskalnika (porazdeljeni viri) in uporabniku predstavijo logično zasnovano iskanje po virih, kar pomeni, da so določeni iskalni viri že vnaprej združeni in predstavljeni kot en sam. Poleg združevanja več iskalnih virov istega spletnega mesta nam metaiskalni viri ponujajo tudi druge možnosti. Združimo lahko vse vire, ki imajo enako vsebino. Tako lahko izvedemo iskanje po vseh virih, ki kot rezultat vrnejo slike, in dobimo iskalnik slik. Metaviri omogočajo statično razvrščanje svojih virov glede na medsebojno pomembnost.

V iskalno storitev smo vključili tudi poosebljanje. Poosebitev sloni na zgodovini iskanj in morebitnih informacijah o osebnih interesih uporabnikov iz njihovih profilov. Iz zgodovine iskanj in seznama ogledanih dokumentov med zadetki lahko izpeljemo uporabnikova interesna področja. Na sam proces poosebljanja moramo gledati kot na ciklično operacijo z neskončnim številom ponovitev. Ključne faze so zbiranje podatkov o uporabniku, klasifikacija uporabnikov v skupine (glede na skupna interesna področja), klasifikacija dokumentov, uporaba podatkov za poosebitev iskalnih rezultatov in meritve uspešnosti.

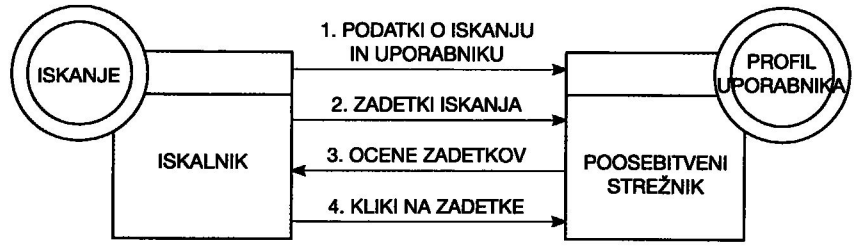
Zbiranje podatkov o uporabnikih poteka na dva načina: z vnosom (uporabniki sami zgradijo svoj profil) in posredno, s spremljanjem njihovih akcij. Prednost prvega načina je, da lahko poosebitev izvajamo že od začetka uporabe storitev, saj profila ni treba zgraditi na podlagi dolgotrajne uporabe. Prednost drugega načina pa je, da avtomatsko spremlja in beleži spremembe pri uporabnikovih interesih. Podatke za prvi način uporabniki vnesejo prek spletnega vprašalnika, drugi način pa beleži uporabniške akcije in sledi predpisu W3C konzorcija za razširjeni format logiranja [4].

Podobnost med dvema dokumentoma merimo s kosinusno podobnostjo med besednima vektorjema, ki predstavljata dokumenta. Algoritem združevanja v skupine združuje dokumente glede na medsebojno podobnost. Kosinusna podobnost je splošno uporabljena tudi pri nadzorovanih (angl. supervised) algoritmih učenja za kategorizacijo dokumentov. Kosinusno podobnost med vsemi dokumenti in novim dokumentom uporabljamo torej za iskanje k najbolj podobnih dokumentov, katerih kategorije (teme) se nato uporabijo za dodelitev kategorij novemu dokumentu. Za dokumenta d_i in d_j se podobnost izračuna po naslednji enačbi (3):

$$\cos(d_i, d_j) = \frac{\sum_k d_{ik} d_{jk}}{\sum_l d_{il}^2 \sum_m d_{jm}^2} \quad (3)$$

Povezava med iskalnikom in poosebitvenim strežnikom v času sprožitve iskalne poizvedbe je izvedena v treh korakih. Prvi korak se izvede ob sprožitvi iskanja. Takrat iskalnik pošlje poosebitvenemu strežniku podatke o uporabniku, iskalnem

nizu, izvoru iskanja (spletni strani, na kateri je uporabnik sprožil iskanje) in o vseh zadetkih iskanja. Drugi korak se izvede pri samem iskanju, takrat poosebitveni strežnik posreduje iskalniku dodatni ocenitveni faktor za razvrščanje zadetkov. Ta faktor se prišteje k običajni oceni in tako pomaga razvrščati zadetke glede na pretekla iskanja in interese uporabnika. Tretji korak se izvede ob uporabnikovem kliku na posamezni zadetek, takrat iskalnik sporoči, kateri zadetek je kliknil uporabnik. Če je uporabnik z zadetkom zadovoljen ali ne, se oceni glede na razmerje med dolžino besedila zadetka in časom ogleda. Ta ocena temelji na predpostavki, da zadetkov, ki mu ne ustrezajo, uporabnik ne pregleduje dalj kot nekaj sekund. Interakcijo med iskalnikom in poosebitvenim strežnikom prikazuje slika 3.



Slika 3: **Interakcija iskalnik – poosebitveni strežnik**

Ključni izziv pri poosebljanju je konsistenca. S časom se namreč uporabniški profili izboljšajo, tako je lahko iskanje bolje prilagojeno, ob enakem iskalnem nizu pa se močno spremeni število zadetkov, njihov položaj v seznamu in njihova vsebina. Za uporabnika to včasih predstavlja neprijetno presenečenje, posebno v primeru, ko po dolgem času ponovno išče določene podatke in se spomni, kako jih je našel v preteklosti. Večinoma uporabniki pričakujejo enake rezultate za enaka povpraševanja. Dodaten izziv je tudi časovna komponenta iskanja, ki lahko močno oslabi uporabnost poosebljanja. Za primer vzemimo uporabnika, ki naše iskalne storitve uporablja večinoma na delovnem mestu, kjer je zaposlen kot razvijalec. Poosebitveni strežnik ima na voljo kar dolgo zgodovino uporabe in ima dokaj natančno klasificirana njegova interesna področja. Tako ob uporabnikovem vnosu niza »AJAX« in ob upoštevanju njegovega profila iskalnik uvrsti višje vsebine, pri čemer se AJAX nanaša na tehnike spletnega razvoja. Potencialne zadetke, ki se nanašajo na čistilo z imenom AJAX, pa uvrsti na dno seznama. Takšno razvrščanje bi uporabniku na delovnem mestu skoraj zagotovo olajšalo in pospešilo iskanje, saj ga, kot razvijalca, ne zanimajo čistila, temveč spletne tehnologije. Vendar lahko isti uporabnik v prostem času sproži iskanje z nizom AJAX in ga dejansko zanimajo podatki o čistilu. V tem primeru bi poosebljeno razvrščanje rezultatov zelo otežilo dostop do podatkov o čistilu. Zato moramo ob ponujenih rezultatih, ki so poosebljeni, vedno ponuditi tudi možnost iskanja brez upoštevanja uporabnikovega profila.

4 SKLEP

Izdelali smo iskalno storitev, ki združuje različne iskalne vire in ponuja možnost poosebljanja. Naš indeksirnik lahko indeksira različne vrste virov in ponuja možnost popolne prilagoditve z vtičniki. Uporabniku ponujamo iskanje glede na njegove pravice. Tako imajo prijavljeni uporabniki možnost iskanja tudi za mejo javno dostopnih vsebin (angl. publicly indexable web; PIW) [5]. Ponujamo tudi združevanje iskalnih virov glede na poljubne kriterije. Iskanje se lahko izvaja s popolnim ujemanjem ali ujemanjem glede na oblikoslovne osnove besed iskalnega niza.

Če povzamemo, so glede na sorodne pristope glavne prednosti naše rešitve naslednje: uporaba oblikoslovnega slovarja, možnost poosebljenega iskanja, aktualnost iskalnih zadetkov (pri večjih iskalnikih se iskalni indeksi ne posodabljajo tako pogosto) in prilagojenost iskalnih indeksov posebnostim slovenskega jezika. Uporaba oblikoslovnega slovarja je dobrodošla predvsem za razširjanje iskalnega niza. V primeru, da iskanje ne vrne zadetkov, razširimo iskalni niz. Če uporabnik išče niz »mariborskega župana« in se tak niz ne pojavi v celotni indeksirani vsebini, lahko iskanje avtomatsko ponovimo za osnovno obliko niza »mariborski župan«. Tako uporabniku ni treba popravljati iskalnih nizov in ponovno sprožiti iskanja. Slovar uporabljamo tudi za pre-

verjanje napak ob vnosu iskalnega niza (uporabnik se lahko zatipka in vnese besedo »žpan« namesto »župan«) in ponujanje pravilnega iskalnega niza. Za iskanje pravilnega niza uporabljamo Levenshteinovo razdaljo [6]. Poosebitev pri iskanju nam omogoča, da so zadetki ocenjeni glede na posameznega uporabnika, posledično so zadetki s področij, ki so uporabnika zanimala v preteklosti, ocenjeni bolje.

Kljub prednostim imamo še veliko možnosti za izboljšave oz. odpravo slabosti naše rešitve. Indeksiranje avtorskih formatov bi bilo treba izboljšati tako, da vsebina, ki je v naslovih poglavij ali je kako drugače označena kot bolj pomembna, dobi boljše ocene. Ker so iskalni indeksi prilagojeni slovenskemu jeziku, imamo slabšo podporo za tuje jezike. Podpora različnim jezikom pri večjih iskalnikih ni vprašljiva, saj je njihova zasnova jezikovno neodvisna. Manj pozornosti podpori večjezičnosti smo namenili, ker je bila naša rešitev usmerjena v področje slovenskega spleta. Vsekakor bo to ena od usmeritev našega nadaljnjega dela. Druge usmeritve bodo predvsem v označevanju vsebine, posledično se bo iskalna storitev zavedala pomena in konteksta besed. To bo v prihodnosti naša prednost pred večjimi iskalniki. Le-ti, če upoštevamo velikost njihovih indeksov, ne morejo ponuditi označevanja vsebin, ki bi bilo učinkovito (koristno za končne uporabnike) in hkrati cenovno upravičeno. Z uporabo konteksta bomo tudi izboljšali delovanje poosebitvenega strežnika.

Eden izmed ciljev za prihodnost je tudi izkoristiti domensko specifično znanje. Običajni iskalniki nimajo modela, ki bi predpisoval, kako semantično povezati koncepte, ki se skrivajo v terminih iskalnega niza. Če npr. iščemo določene publikacije določenega avtorja, bi nam bilo v veliko pomoč, če bi nam iskalnik vrnil dodatne informacije o tem avtorju (druge publikacije, življenjepis, avtorjeve kontaktne podatke). Torej, cilj je iskalna arhitektura, ki omogoča izkoriščanje področno specifičnega znanja. Postopek iskanja bi bil po naslednjem zaporedju: uporabnik kot običajno poda svojo iskalno poizvedbo v obliki seznama ključnih besed. Poizvedba se posreduje običajnemu iskalniku, ki dodeli ocene vsakemu vrnjenemu dokumentu na običajen način. Dodatno sistem vsebuje področno specifično bazo znanja. Ta je sestavljena iz modela področja, ki vsebuje vozlišča. Vozlišča kažejo na posamezne vire in imajo dodatne podatke v obliki povezanih lastnosti, kot jih določa področni model (ontologija). Uteži določajo moč vsake povezave v ontologiji. Iskalni proces se nadaljuje na drugi stopnji (prva stopnja je klasično iskanje po besedilu dokumentov). Vrnjeni dokumenti so povezani s primerki v ontologiji, ki imajo obtežene povezave do drugih vozlišč ontologije. Ta vozlišča so vhod v razširitveno-aktivacijski algoritem, v katerem so začetne aktivacijske vrednosti vsakega vozlišča enake položaju dokumenta v vrnjenem seznamu. Končni seznam rezultatov iskanja bo tako urejen glede na izhod razširitveno-aktivacijskeg a algoritma in bo ponujal več zadetkov, ki bodo bliže pričakovanjem uporabnika.

VIRI IN LITERATURA

[1] Barroso, Luiz, Dean, Jeffrey, Hoelzle, Urs: Web Search for a Planet: The Google Cluster Architecture, Micro, IEEE, 2003, Vol. 23, št. 2, str. 22–28.

[2] About robots.txt; spletni vir: http://www.robotstxt.org/robotstxt.html, zadnji obisk 15. 5. 2009.

[3] Brin, Sergey, Page, Lawrence: The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Networks and ISDN Systems, 1998, Vol 30, št. 1–7, str. 107–117.

[4] W3C: Extended Log File Format, http://www.w3.org/TR/WD-logfile, zadnji obisk 15. 5. 2009.

[5] Lawrence, Steve, Giles, C. Lee: Searching the world wide web, Science, 1998, Vol. 280, št. 5360, str. 98–100.

[6] Li, Yujian, Liu, Bo: A Normalized Levenshtein Distance Metric, IEEE, 2007, Vol. 29, št. 6, str. 1091–1095.

Sandi Pohorec je raziskovalec na Inštitutu za računalništvo na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegova raziskovalna področja zajemajo predvsem inteligentne sisteme in njihovo aplikacijo iskanje informacij, obdelavo naravnih besedil, podatkovno rudarjenje in odkrivanje novega znanja iz obstoječih podatkovnih zbirk.

Milan Zorman je izredni profesor na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegovo področje raziskav so klasični in hibridni inteligentni sistemi ter njihova aplikacija na različnih področjih s poudarkom na medicini in zdravstvu. Je direktor CIMRS – Centra za interdisciplinarne in multidisciplinarne raziskave in študije Univerze v Mariboru in vodja projekta in pisarne Evropske podjetniške mreže na CIMRS.