VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis

A model for analyzing and predicting the scent of a web site

Magistro baigiamasis darbas

Atliko:	Artiom Kučaidze	(parašas)
Darbo vadovas:	Dr. Kristina Lapin	(parašas)
Recenzentas:	Doc. dr. Vytautas Čyras	(parašas)

Vilnius – 2008

Santrauka lietuvių kalba

Darbe, remiantis informacijos paieškos teorija, bandoma sukurti tinklalapio navigavimo asociacijų analizės ir prognozavimo modelį. Šio modelio tikslas – simuliuoti potencialių tinklalapio vartotojų informacijos paieškos kelius turint apibrėžtą informacinį tikslą. Modelis kuriamas apjungiant LSA, SVD algoritmus ir koreliacijos koeficientų skaičiavimus. LSA algoritmas naudojamas kuriant semantines erdves, o koreliacijos koeficientų skaičiavimai naudojami statistikoje. Kartu jie leidžia tinklalapio navigavimo asociacijų analizės ir prognozavimo modeliui analizuoti žodžių semantinį panašumą.

Darbo eigoje išskiriamos pagrindinės problemos, su kuriomis gali susidurti tinklalapio lankytojai sudarant tinklalapio navigavimo asociacijas – tai yra konkurencijos tarp nuorodų problema, klaidinančių nuorodų problema ir nesuprantamų nuorodų problema. Demonstruojama kaip sukurtas modelis atpažįsta ir analizuoja šias problemas.

Raktiniai žodžiai: Informacijos paieška, informacijos nuojauta, navigavimo asociacijos, tinklalapio panaudojamumas, tinklalapio analizė, tinklalapio navigacija, panaudojamumo problemos, LSA, Latent Semantic Analysis, SVD, Singular Value Decomposition, ilgiausiai pasikartojančios sekos, koreliacija.

Santrauka anglų kalba (*Summary*)

In this document we develop a model for analyzing and predicting the scent of a web site, which is based on information foraging theory. The goal of this model is to simulate potential web page users and their information foraging paths having specific information needs. Model is being developed combining LSA, SVD algorithms and correlation values calculations. LSA algorithm is used for creating semantic spaces and correlation values are user in statistics. Together they provide possibility to analyze word‘s semantic similarity.

Primary problems of web navigation are described in this document. These problems can occur for users while creating the scent of a web site. User can face with concurrency between links problem, wrong sense link problem and unfamiliar link problem. In this document we demonstrate how model recognizes and analyzes these problems.

Key words: Information foraging, information scent, navigation associations, web page usability, web page analysis, web page navigation, usability problems, LSA, Latent Semantic Analysis, SVD, Singular Value Decomposition, longest repeated subsequence, correlation.

Turinys

Įvadas

Užėjęs į tinklalapį, lankytojas paprastai turi tikslą – surasti kažkokią jam aktualią informaciją. Ieškodamas jos, lankytojas atlieka tam tikrus logiškus veiksmus. Akivaizdu, kad tinklalapių kūrėjai stengiasi suprasti lankytojų veiksmus ir informacijos poreikius. Taigi, iš to seka, kad egzistuoja efektyvios sistemos poreikis, kuri sugebėtų [CPP99]:

- analizuoti tinklalapio lankytojų tikslus ir veiksmus;
- analizuoti ir nuspėti (prognozuoti) tinklalapio papanaudojamumą (šiame darbe kalbama apie tinklalapio navigacijos papanaudojamumą).

Dauguma tinklalapių saugo duomenis, susijusius su vartotojų veiksmais (angl. „log files“). Egzistuoja daugybė įrankių, skirtų ištraukti įvairią informaciją iš tokių failų, tačiau nedaugelis tokių įrankių pavirto rimtais produktais. Dauguma iš jų apsiriboja log failų paprastos statistinės informacijos atvaizdavimu, tuo pačiu pateikiant mažai informacijos apie vartotojo naudojimąsi tinklalapiu [Pir00].

Darbe iškeliami tokie pagrindiniai klausimai:

1. Bendras tinklalapis – Kokie yra tinklalapio naršymo ir informacijos paieškos keliai, kokia yra tinklalapio nuorodų struktūra? Kaip vertinama tinklalapio informacijos paieška?
2. Duotas puslapis – Iš kur atėjo vartotojas (kokiais keliais jis eina?). Kur jis eina?
3. Vartotojai – Kokie yra vartotojų (tikrų, ar simuliuojamų) tikslai? Kur jie turėtų eiti žinant jų tikslą? Ar pateikiami duomenys atitinka jų poreikiams?

Paprastai Internete vartotojai ieško informacijos einant per įvairių puslapių nuorodas. Puslapių, susietų su tomis nuorodomis, turinys pateikiamas vartotojui naudojant tam tikrą tekstą, arba paveikslėlius. Vartotojas daro sprendimus remiantis šia informacija ir susidariusiomis puslapio navigavimo asociacijomis (toliau – nuojauta) (angl. the scent of a site). Informacijos nuojauta – tai puslapio pateikiamos informacijos vartotojo įvertinimas, kuriuo remiantis vartotojas daro tam tikrą sprendimą (pvz. paspaudžia nuorodą, jo manymu galinčią priartinti jį prie siekiamo tikslo) [PC99]. Taip sudaromas vartotojo informacijos paieškos kelias.

Darbe bus remiamasi informacijos paieškos teorija [PC99]. Ši teorija paaiškina žmogaus, ieškančio informacijos ir darančio logiškus veiksmus, elgesį. Terminas tinklalapio nuojauta kilo iš šios teorijos. Taip pat bus panaudota „ilgiausiai pasikartojančia veiksmų sekos“ sąvoka [PP99]. Ji bus naudojama siekiant interpretuoti simuliuojamų ir realių vartotojų paieškos kelius ieškant informacijos.

Darbo teorinė prielaida yra tokia: šiuos paieškos kelius galima simuliuoti apjungus keletą skirtingų metodų ir algoritmų – LSA (angl. Latent Semantic Analysis), SVD (angl. Singular Value Decomposition), ilgiausiai pasikartojančios sekos (IPS) sąvoką bei statistikoje naudojamus rangų koreliacijos koeficientus. Norint naudoti LSA algoritmą reikės sukurti tam tikros srities žodžių semantinę erdvę (šiuo metu nėra sukurta nei viena LSA semantinė erdvė Lietuvių kalba).

Darbo aktualumas grindžiamas tuo, kad neteisinga puslapio nuojauta gali nuvesti vartotoją neteisingu keliu, ko pasėkoje nebus surasta vartotojui reikalinga informacija. Tokiu atveju nuostolius patiria ne tik vartotojas, bet ir tinklalapio kūrėjai. Kitą kartą, ieškodamas informacijos, vartotojas tiesiog ieškos jos kitur. Šiuo metu nėra užbaigtų produktų, vertinančių būtent puslapio nuojautą, todėl darbo rezultatas – tinklalapio nuojautos analizės ir prognozavimo modelis – tikrai aktualus. Ypač geros vartotojų nuojautos sudarymu suinteresuoti tokie tinklalapiai kaip e-parduotuvės, informaciniai portalai ir t.t., kur vartotojas ateina turėdamas tikslą surasti tam tikrą jam aktualią informaciją ir ieško jos remiantis jo kelyje pasitaikančių puslapių nuojauta.

Darbo tikslai - sukurti prognozavimo ir analizės modelį, sugebantį simuliuoti potencialų tinklalapio vartotoją. Naudojant šį modelį bus siekiama sukurti priemones, skirtas puslapio nuojautos automatiniam apskaičiavimui, kurių dėka galima bus gauti modelio veikimo rezultatus ir juos išanalizuoti.

Tikslui įgyvendinti turi būti atliktos žemiau pateiktos užduotys

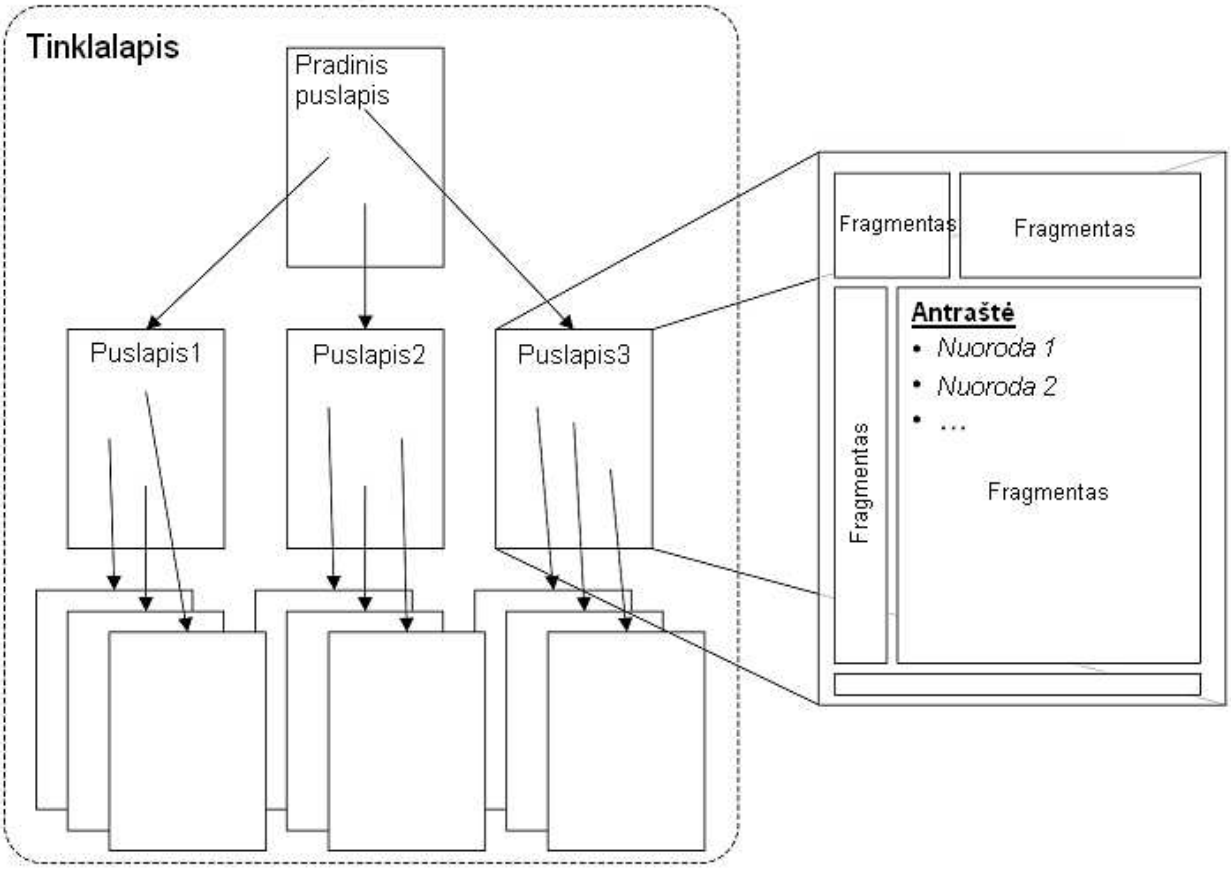
- apžvelgti egzistuojančius algoritmus, modelius ir įrankius, susijusius su darbo tikslais;
- pritaikyti informacijos paieškos teoriją ([PC99]) praktikoje ir jos pagrindu sukurti algoritmus, kurie gaus informacinį tikslą ir sieks jį įgyvendinti atlikdami logiškus veiksmus (ieškos informacijos);
- sukurti metodiką, leidžiančią prognozuoti ir analizuoti agento puslapio nuojautą;
- pritaikyti sukurtą metodiką praktikoje.

Sėkmingai atlikus užduotis pasiekti tokie rezultatai:

- prognozavimo, analizės modelis, sugebantis simuliuoti potencialų tinklalapio vartotoją;
- vartotojo nuojautos vertinimo metodika;
- vartotojo nuojautos automatinio vertinimo algoritmas, kurio pagalba demonstruojamas modelio veikimas ir atliekami bandymai;
- praktinės rekomendacijos.

Darbo tyrimo objektas: Analizuojamas vartotojo nurodytas tinklalapis. Tam naudojamas prognozavimo ir analizės modelis siekiant simuliuoti potencialių vartotojų informacijos paieškos kelius duotame tinklalapyje.

Paprastai tinklalapis turi pradinį puslapį, kuriame yra nuorodos į gilesnius tinklalapio puslapius. Kiekvienas puslapis pateikia vartotojui tam tikrą informaciją bei nuorodas. Vartotojas, ieškodamas informacijos, apžiūri šiuo metu vaizduojamą puslapį, susikoncentruoja ties jį dominančia informacija (tekstu, paragrafu, nuoroda) ir priima sprendimą – norima informacija rasta, arba ne [OC00]. Jei norima informacija nerasta, vartotojas spaudžia nuorodą, kuri jo manymu priartins jį prie informacinio tikslo. Taip elgsis ir kuriamas tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis. Tipinė tinklalapių struktūra su kuria bus dirbama pavaizduota 1 pav.



1 pav. Tipinė tinklalapių struktūra

Kuriamas modelis turės išanalizuoti duotą puslapį ir ištraukti iš jo visas korektiškas nuorodas su nuorodų tekstais. Tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis padės išrinkti nuorodą, kuri labiausiai atitiks apibrėžtą paieškos tikslą. Šis procesas pavaizduotas 2 pav.

2 pav. Informacinis tikslas – „surasti programų sistemų katedros puslapį“

Šiame pavyzdyje vartotojas, užėjęs į mif.vu.lt puslapį, turi informacinį tikslą – „surasti programų sistemų katedros puslapį“. Patekęs į pagrindinį mif.vu.lt puslapį, vartotojas apžvelgia puslapį ir susikoncentruoja ties nuoroda „Katedros“. Taip yra dėl to, kad ši nuoroda, pagal susidariusią vartotojui informacinę nuojautą, turi aukščiausią įvertinimą. Atitinkamai perėjęs į katedrų puslapį, vartotojas mato nuorodų į katedras sąrašą, kuriame, pagal tą pačią schemą, pasirenka nuorodą „Programų sistemų katedra“.

Iškeliami **pagrindiniai modelio funkciniai tikslai ir uždaviniai** – tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis turi mokėti (turint apibrėžtą informacinį tikslą ir nurodytą galutinį puslapį su ieškoma informacija):

- išanalizuoti duotą puslapį ir gauti puslapio nuorodų struktūrą (puslapio html kodo analizė);

- įvertinti kiekvieną nuorodą atsižvelgiant į informacinį tikslą (atlikti puslapio nuorodų struktūros analizę);
    - kiekvienai nuorodai priskirti svorio indeksus (reitingus). Kitaip tariant, sudaryti puslapio navigavimo asociacijas;
- pereiti prie kito puslapio pagal pasirinktą nuorodą, arba sustoti jei pasiektas puslapis su ieškoma informacija (informacijos paieškos kelio formavimas).
- pateikti informacijos paieškos tinklalapyje kelią bei analizės rezultatus.

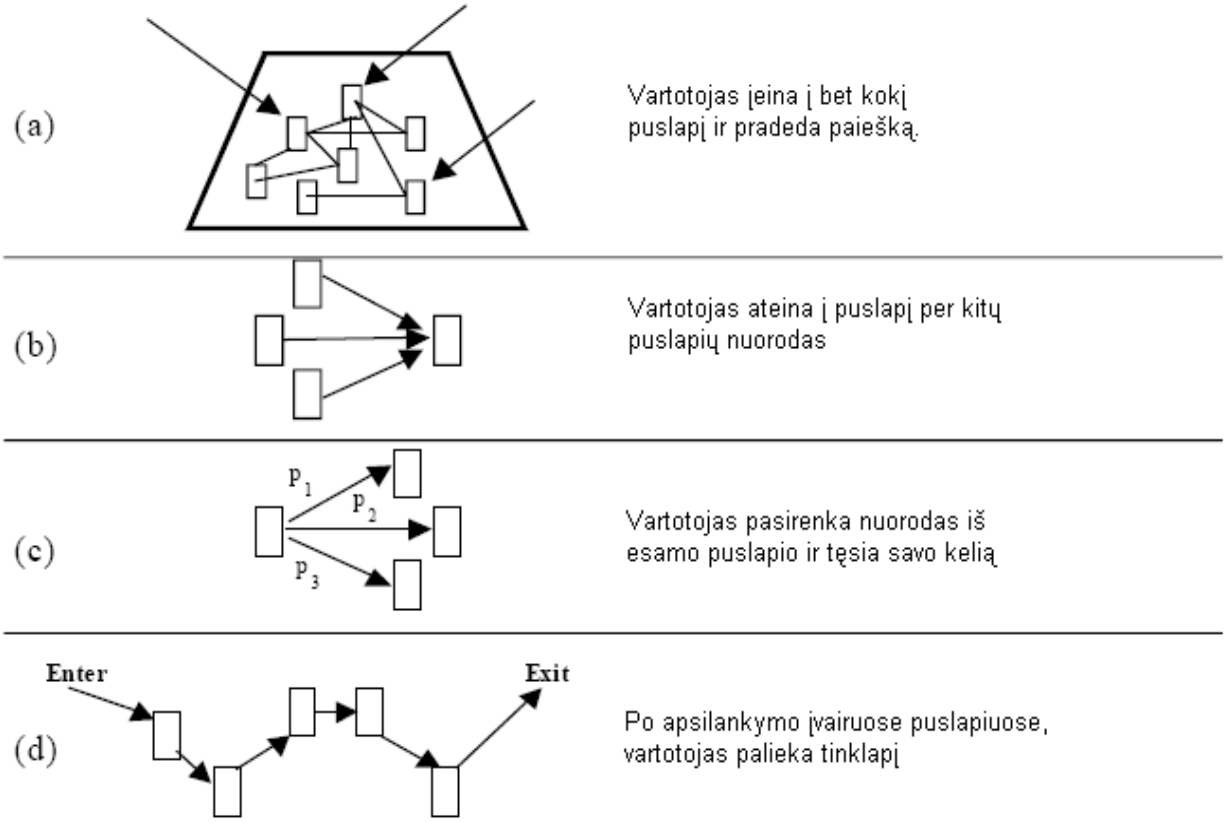
1. Egzistuojančių įrankių ir algoritmų apžvalga

1.1 WebCriteria SiteProfile

Yra tik keletas sistemų, bandančių nuspėti tinklalapio naršymo kelius, arba panaudojamumą remiantis tinklalapio dizainu. WebCriteria SiteProfile [SP] naudoja programinius agentus siekiant simuliuoti potencialių vartotojų veiklą. Agentai naršo tinklalapį ir taiko modifikuotą GOMOS modelį. Gauti duomenys integruojami į metrikas, kurios įvertina puslapių pakrovimo laiką, puslapių turinio prieinamumą ir t.t. Tačiau šie agentai vaikšto po tinklalapį atsitiktine tvarka. Kitaip sakant, jie neturi jokio informacinio tikslo, kuris remtų jų navigavimo kelių pasirinkimą. Prieinamumo metrikos remiasi tinklalapio nuorodų struktūra ir puslapių turinio dydžiu, tačiau pačio turinio analizė nėra vykdoma. Ši sistema nesimuliuoja vartotojų, kurie turi apibrėžtą informacinį tikslą, arba vartotojus, kurie gali suprasti navigavimo variantus bei priimti navigavimo sprendimus. Tuo labiau, WebCriteria kūrėjų tyrimai rodo, kad ši sistema neturi sąryšio su realių vartotojų elgesiu [Pir00]. Iš šios sistemos darbui naudingas agentų principas. Agentais darbe vadinsime autonominius algoritmus.

1.2 Ilgiausiai pasikartojanti seka (IPS)

Darbe bus naudojama ilgiausiai pasikartojančios sekos (IPS) sąvoką [PP99]. Yra keli būdai kaip gali būti formuojami vartotojo paieškos keliai ir, tuo pačiu, IPS (3 pav.) [PP99]

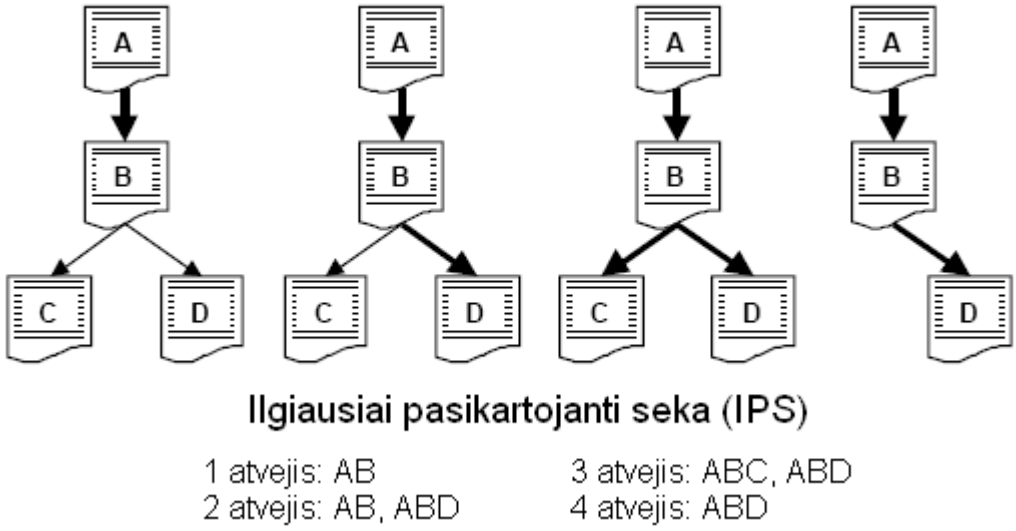


3 pav. Vartotojo paieškos kelių ir IPS formavimas

IPS – tai elementų rinkinys, kur:

- seka – tai nuoseklių elementų rinkinys;
- pasikartojančios – reiškia, kad elementas pasitaiko daugiau negu *T* kartų (kur *T* paprastai yra 1);
- ilgiausios – reiškia, kad net jei duota seka yra poaibis kitos pasikartojančios sekos, yra bent vienas atvejis, kai duota seka yra ilgiausia.

Kad būtų aiškiau, įsivaizduokime, kad turime tinklalapį, kuriame yra puslapiai A, B, C, D, kur A turi nuorodą į B, o B turi nuorodas į C ir D (4 pav.)



4 pav. Ilgiausiai pasikartojančių sekų pavyzdys

Iš 4 pav. matosi, kad vartotojai daug kartų eina iš A į B, bet tik vienas vartotojas eina iš B į C, ir vienas vartotojas eina iš B į D (1 atvejis). Šiuo atveju IPS yra AB. Tačiau, jei daugiau nei vienas vartotojas paspaudė nuorodą iš B į D (2 atvejis), tai IPS bus AB ir ABD. Šiuo atveju AB irgi bus IPS, nes egzistuoja bent vienas atvejis, kai po AB nesekė ABD (t.y. egzistuoja vartotojas, kurio kelias buvo ABC). 3 atvejis iliustruoja situaciją, kai abu – ABC ir ABD yra IPS, nes jie abu pasitaiko daugiau negu vieną kartą. 4 atvejis parodo, kad AB jau nebe IPS, nes nėra atvejo, kai AB būtų ilgiausias kelias.

IPS padės atskirti kraštutinius rezultatus ir gauti bendresnius simuliuojamus paieškos kelius.

1.3 LSA, PMI ir GLSA algoritmų analizė

Toliau trumpai apžvelgiami išvardinti algoritmai bei pasirinkamas vieną iš jų. Pasirinkto algoritmo pritaikymas bus aprašytas vėliau. Taip pat bus paminėtas TOEFL testas – tai 80 klausimų, išvestų iš testinės medžiagos, sukurtos ETS (angl. Educational Testing Service). Ši medžiaga skirta įvertinti užsienio studentų, atvykstančių į Ameriką, anglų kalbos žinias. Visi 80 klausimai yra sudaryti iš pavienių žodžių – pavyzdžiui, pasirinkti sinonimą duotam žodžiui iš pateiktų variantų.

1.3.1 Algoritmas „Latent Semantic Analysis“ (LSA)

LSA – tai metodas, kuris leidžia apskaičiuoti semantinį atstumą tarp žodžių ir kurį galima naudoti siekiant palyginti žodžių semantinį panašumą. Jis remiasi įvairių dokumentų rinkiniu. LSA atvaizduoja kiekvieną žodį kaip vektorių žodžių ir dokumentų erdvėje. Šios erdvės sukūrimas vyksta sudarant matricą *C*. Matrica sudaroma taip, kad kiekviena jos eilutė yra

unikalus žodis, o kiekvienas stulpelis yra dokumentas. Kiekvienoje celėje rašoma kiek kartų tam tikras žodis yra panaudotas tam tikrame dokumente. Po to matrica *C* yra normalizuojama ir jai pritaikoma SVD (angl. singular value decomposition). Po šių veiksmų žodžius galima interpretuoti kaip vektorius. Panašumas tarp žodžių apskaičiuojamas kaip cos tarp dviejų vektorių. Bendrai LSA algoritmą galima apibrėžti šiais žingsniais:

1. Sudaryti matricą *C*, kurioje *C[i,j]* reiškia kiek kartų žodis *i* yra naudojamas dokumente *j*.
2. Apskaičiuoti *LC* iš *C* taip, kad *LC[i,j] = log(1 + C[i,j])*.
3. Apskaičiuoti *H[i]* žodžio *i* taip, kad:
   $H[i] = \sum_j - C[i,j] \log C[i,j]$.
4. Normalizuoti reikšmes *LC: N[i,j] = LC[i,j]/H[i]*.
5. *LC* pritaikyti SVD siekiant gauti matricą *Q* k laipsnio.
6. Žodis *i* atvaizduojamas kaip vektorius *Q[i]*, o panašumas tarp žodžių *i* ir *j* apskaičiuojamas taip: *cos(Q[i] Q[j])*.

Atlikti LSA algoritmo testavimai [LFL98] parodė, kad šis algoritmas sugeba surinkti įvertinimą 72% TOEFL sinonimų teste.

1.3.2 Algoritmas „Pointwise Mutual Information“ (PMI)

PMI tarp dviejų žodžių *A* ir *B* apskaičiuoja tikimybę rasti žodį *B* tekste, jei žinoma, kad tekstas turi žodi *A* [TUR01]. Tai kartojimų matavimas, kuris matuoja tikimybę, kad du žodžiai yra susiję ir gali būti naudojami kartu. PMI tarp *A* ir *B* apskaičiuojama taip:

$$PMI(A,B) = \log \frac{p(A,B)}{p(A)p(B)} \approx \log \frac{C(A,B) \times N}{C(A)C(B)} ,$$

Kur *p(A,B)* yra tikimybė, kad A ir B pasitaiko kartu tame pačiame dokumente (dokumentu galime laikyti fiksuoto dydžio kompiuterio langą su tekstu); *p(X)* yra tikimybė, kad žodis *X* pasitaikys dokumente; *C(A,B)* yra skaičius dokumentų, kuriuose *A* ir *B* pasitaiko kartu; *C(X)* yra skaičius dokumentų, kuriuose pasitaiko žodis *X*, o *N* yra visų dokumentų skaičius. Siekiant naudoti PMI tik žodžių porų tikimybių skaičiavimui, PMI gali būti supaprastintas taip:

$$PMI(A,B) \approx \frac{C(A,B)}{C(A)C(B)}$$

Dviejų žodžių panašumas matuojamas jų PMI įvertinimu. PMI surenka įvertinimą ~62.5% - 72% TOEFL sinonimų teste.

1.3.3 Algoritmas „Generalized Latent Semantic Analysis“ (GLSA)

GLSA naudoja PMI ir LSA algoritmų idėjas [MLF+05]. Kaip ir LSA, šis algoritmas naudoja SVD siekiant išvalyti „triukšmus“ sistemoje. Tačiau, priešingai negu LSA, žodžių ir dokumentų matrica pakeičiama į PMI žodžių ir žodžių matricą.

Bendrai GLSA algoritmą galima apibrėžti šiais žingsniais (turint žodžių aibę *V*, dokumentų aibę *D* ir faktorių skaičių *k*):

1. Sudaryti matricą *C: C[i,j]* atvaizduoja kaip dažnai žodžiai *i* ir *j* žodynėlyje *V* pasitaiko kartu dokumentų aibėje *D*. Kiekvienam žodžiui i apskaičiuoti *F[i]*: kaip dažnai jis pasitaiko dokumentų aibėje.
2. Sudaryti PMI matricą:
   $PMI[i,j] = \log \frac{C[i,j]}{F(i)F(j)}$
3. Pritaikyti PMI matricai SVD, siekiant sumažinti PMI laipsnį iki *k*. Gautą matricą pažymėsim *Q*.
4. Panašumas tarp žodžių *i* ir *j* apskaičiuojamas taip: *cos(Q[i]Q[j])*

Testavimo rezultatai [MLF+05] rodo, kad GLSA surenka įvertinimą ~76% TOEFL sinonimų teste.

1.3.4 Palyginimas ir išvados

Išvados bus daromos remiantis [BRP01] atliktais tyrimais ir rekomendacijomis. Visi išvardinti algoritmai tarpusavyje yra apytiksliai lygiaverčiai. Realizacijos atžvilgiu PMI yra pats paprasčiausias algoritmas. Tačiau jo rezultatai labai svyruoja nuo pasirinktos teksto aibės. Jį sunkiau pritaikyti siekiant atvaizduoti vartotojų bazines žinias, susijusias su puslapio tematika. Palyginus su LSA, GLSA turi pranašumą naudojant mažesnes žodžių ir žodžių (LSA naudoja žodžių ir dokumentų matricą) matricas ir jis mažiau priklausomas nuo dokumentų aibės didžio. Tačiau kai tik prireikia apskaičiuoti panašumą žodžio, kurio nėra matricoje, visa žodžių matrica turi būti sudaryta ir perskaičiuota iš naujo. LSA šios problemos neturi. Bendrai šių algoritmų darbui aktualūs privalumai ir trūkumai surašyti 2 Lentelėje.

Algoritmas	TOEFL rezultatai	Privalumai	Trūkumai
LSA	72%	Įtraukiant naują žodį (dokumentą), žodžių matricą nereikia perdarynėti iš naujo.	TOEFL rezultatai ne tokie aukšti kaip GLSA algoritmo.

PMI	62.5% - 72%	Paprasčiausia realizacija palyginus su kitais algoritmais.	Rezultatai labai svyruoja nuo pasirinktos teksto aibės. Kiti algoritmai yra žymiai pastovesni.
GLSA	76%	Aukščiausi TOEFL įvertinimai palyginus su kitais algoritmais.	Įvertinant naują žodį, kurio nėra žodžių matricoje, visa žodžių matrica turi būti sudaryta ir perskaičiuota iš naujo. Tai labai sulėtina algoritmo veiklą.

1 Lentelė. LSA, PMI, GLSA privalumai ir trūkumai.

Kuriamas modelis naudos LSA algoritmą dėl šių priežasčių:

1. Kadangi reikia atvaizduoti realių vartotojų žinias – tai reiškia, kad reikia geros, su puslapio tematika susijusios semantinės erdvės. Tokios erdvės, prieinamos internete, yra sukurtos anglų kalba. Lietuvių kalbai tokių semantinių erdvių nėra. Darbe bus sudaromos naujos semantinės erdvės, todėl nuolat reikės jas pildyti naujais dokumentais, t.y. pildyti žodžių matricą, o tai lengviausiai ir greičiausiai daryti naudojant LSA algoritmą. Apie semantinės erdvės kūrimą bus kalbama vėliau.
2. Iš visų išvardintų algoritmų internete daugiausiai medžiagos yra apie LSA algoritmą.

1.4 Koreliacija

Statistinių tyrimų, paremtų skaitlingu duomenų apdorojimu, vienas iš tikslų yra padėti surasti ryšį tarp dviejų požymių – pasekmės, rezultato arba priklausomojo kintamojo ir veiksnio, priežasties, faktoriaus arba nepriklausomojo kintamojo [KG90]. Pirmasis požymis atitinka funkcijas, o antrasis – argumento sąvokas matematikoje. Kadangi racionalios veiklos pagrindas yra numatyti įvykių eigą, o tai grindžiama reiškinių ir procesų ryšių žinojimu, todėl svarbu mokėti tuos ryšius aprašyti.

Pavyzdžiui, akivaizdu, kad yra ryšys tarp gyventojų pajamų ir kokio nors maisto produkto perkamumo. Tačiau, be gyventojų pajamų (arba turimų lėšų), to produkto vartojimą lemia ir kiti ne mažiau svarbūs požymiai (poreikis, sezoniškumas, kokybė ir panašiai). Neįvertinamųjų veiksnių buvimas sąlygoja tam tikrą pasekmės neapibrėžtumą, todėl žinant tik vieną veiksnį galima apibrėžti tik vidutinę pasekmės skaitinę reikšmę.

Štai toks ryšys, kai vieną veiksnio požymio skaitinę vertę atitinka keletas pasekmės skaitinių verčių, iš kurių galima nustatyti vidutinę, vadinamas koreliaciniu [KG90]. Pati koreliacijos sąvoka reiškia savitarpio santykiavimą, tarpusavio priklausomybę. Koreliacijos

sąvoka apima dviejų ar daugiau kaip dviejų statistinių eilučių priklausomybę, kuri gali būti nagrinėjama vienu ar kitu metodu.

Statistinis koreliacinio ryšio tyrimas paprastai remiasi stebėjimų duomenimis apie tiriamų požymių verčių pasirodymą. Tačiau kartu ne visada liudija, kad vienas požymis yra priežastis, o kitas – pasekmė, nes tai gali būti kitų veiksnių išvada [KG90]. Taip pat koreliacinis ryšys tarp požymių kartais rodo ne šių požymių tarpusavio ryšį, o jų priklausomumą nuo kažkokio trečio požymio ar kelių požymių, kurie tyrime nenagrinėjami. Koreliaciniai ryšiai neparodo, kuris požymis yra priežastis, o kuris pasekmė, atvirkščiai jie liudija, kad vieno požymio kitimas sukelia kito požymio kitimą, tačiau nerodo, ar kitimo priežastis yra viename iš požymių, ar yra už tyrimo ribų. Kartais galima stebėti ar kontroliuoti kurio nors požymio kitimą ir sekti, kaip jo kitimas veikia kitus požymius. Toks požymis vadinamas nepriklausomu. Požymiai, kurie, mūsų nuomone, kinta veikiant nepriklausomiems požymiams, vadinami priklausomais. *Priklausomų ir nepriklausomų kintamųjų bendras kitimas vadinamas koreliacine priklausomybe* [KG90].

Vertinant koreliacinį ryšį, atsižvelgiama į koreliacinio ryšio koeficientą, kuris tikimybių teorijoje ir statistikoje yra statistinio ryšio tarp kintamųjų stiprumo matas. Jis visada yra skaičius iš intervalo [-1; 1]. Ryšys laikomas labai silpnu, jei yra lygus -1, o labai stiprus lygus 1. Pagal kryptį koreliaciniai ryšiai yra teigiami (tiesioginiai) ir neigiami (atvirkštiniai) [KG90].

Kiekybinių kintamųjų ryšio stiprumą galima išmatuoti Pirsono (Pearson) koreliacijos koeficientu [KG90]. Didelės šio koeficiento reikšmės, nežiūrint ar jos teigiamos, ar neigiamos, atitinka tai, ką vadiname stipria koreliacija, o mažos reikšmės – silpna koreliacija. Jei koreliacija yra nereikšminga, tai nereiškia, kad koreliacijos koeficientas tiksliai lygus nuliui, tačiau jo reikšmė yra arti nulio. Sociologai ir psichologai dažnai tiria daug kintamųjų, ir jiems svarbu nustatyti, kuris iš šių kintamųjų yra labiau susijęs su turimu kintamuoju. Kiekybinių kintamųjų tyrimui naudojami keli skirtingi koreliacijos koeficientai. Dažniausiai naudojamas Pirsono koreliacijos koeficientas **r**.

Koreliacijos stiprumo interpretacija yra tokia:

- nuo 0,9 iki 1,0 arba nuo -0,9 iki -1,0 – koreliacija laikoma labai stipria;
- nuo 0,7 iki 0,9 arba nuo -0,7 iki -0,9 – stipri koreliacija;
- nuo 0,4 iki 0,7 arba nuo -0,4 iki -0,7 – koreliacija yra vidutine;
- nuo 0,2 iki 0,4 arba nuo -0,2 iki -0,4 – koreliacija laikoma silpna ir galiausiai;
- nuo 0,2 iki -0,2 – labai silpna arba nereikšminga koreliacija.

Kartais gali tekti apskaičiuoti stiprią koreliaciją tarp kintamųjų, kurių nesieja jokie loginiai ryšiai. Tai vadinamoji *nesąmoninga koreliacija* [KG90]. Visiškai atsitiktinai kažkas gali

apskaičiuoti ryšį tarp studentų skaičiaus ir skaičiaus žuvų, sugautų vandenyje bei gauti reikšmė, pavyzdžiui, r = 0,98.

Ekonominių rodiklių sąveikos stiprumui matuoti plačiai naudojamas anglų matuoti plačiai naudojamas anglų psichologo Č. Spirmeno pasiūlytas rangų koreliacijos koeficientas [KG90].

Rangai – tai visumos vienetų eilės numeriai rangų eilutėje. Rangavimo pavyzdys gali būti tokių duomenų išdėstymas:

Įmonių eil. Nr.	Jų pelnas (mln.Lt)	Rangai
1	10	6,5
2	12	4
3	10	6,5
4	12	4
5	12	4
6	15	2
7	17	1
8	9	8

Septinta įmonė gauna didžiausią pelną. Jai suteikiamas pirmas rangas, šeštai – antras. Antra, ketvirta ir penkta įmonės gauną vienodą pelno dydį. Jos pagal pelną turėtų trečią, ketvirtą ir penktą vietas. Vidutinė jų vieta: (3 + 4 + 5) : 3 = 4. Toks jų rangas. Pirma ir trečia taip pat gauna vienodą pelną. Jų rangas: (6 + 7) : 2 = 6,5. Aštuntoji gauna mažiausią pelną, o jos rangas aštuntas.

Darbui aktuali yra geometrinė koreliacijos interpretacija. Auksčiau aprašyto pavyzdžio duomenis galima interpretuoti kaip vektorius, tada koreliacijos koeficientas gali būti skaičiuojamas kaip kampo tarp dviejų vektorių kosinusas [KG90]:

$$\cos\theta = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \, \|\mathbf{y}\|}$$

Koreliacija, o tiksliau rangų koreliacijos koeficientai, bus naudojami modelyje ir padės įvertinti ryšį tarp dviejų žodžių matricos eilučių. Taip galima sužinoti kaip stipriai du žodžiai yra susiję tarpusavyje (įvertinsim jų ryšį skaičiais iš intervalo [-1; 1]).

2 Principiniai darbo sprendimai

Kaip jau buvo minėta, pagrindiniai modeliui iškelti funkciniai tikslai ir uždaviniai (turint apibrėžtą informacinį tikslą ir nurodytą galutinį puslapį su ieškoma informacija) yra tokie:

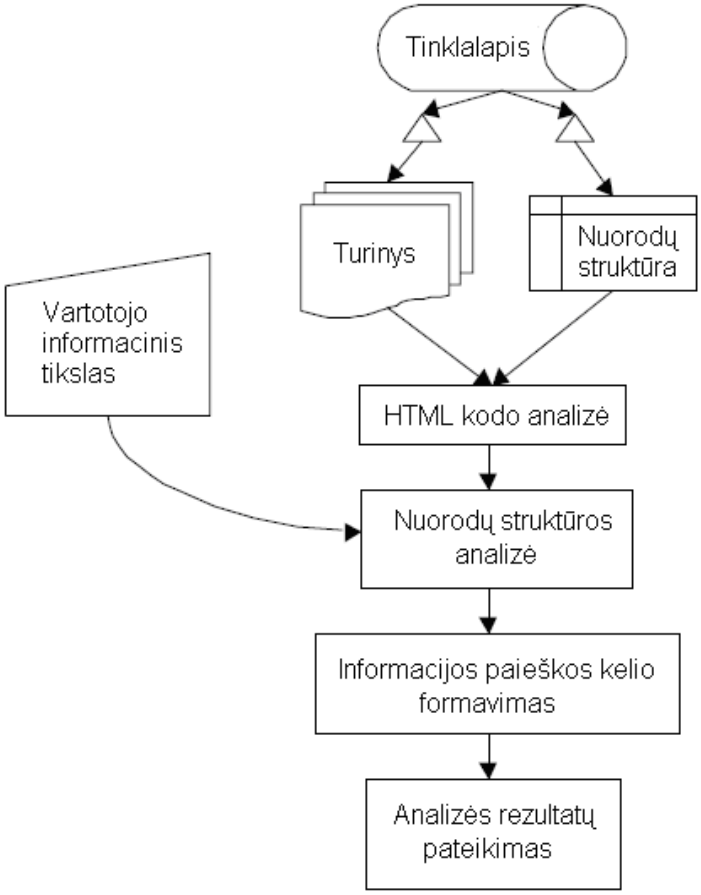
- išanalizuoti duotą puslapį ir gauti puslapio nuorodų struktūrą (puslapio html kodo analizė);

- įvertinti kiekvieną nuorodą atsižvelgiant į informacinį tikslą (atlikti puslapio nuorodų struktūros analizę);
    - kiekvienai nuorodai priskirti svorio indeksus (reitingus). Kitaip tariant, sudaryti puslapio navigavimo asociacijas;
- pereiti prie kito puslapio pagal pasirinktą nuorodą, arba sustoti jei pasiektas puslapis su ieškoma informacija (informacijos paieškos kelio formavimas).
- pateikti informacijos paieškos tinklalapyje kelią bei rekomendacijas.

Atsižvelgiant į šiuos uždavinius, modelis bus suskaidytas į komponentus:

- puslapio HTML kodo analizė;
- puslapio nuorodų struktūros analizė;
- informacijos paieškos kelio formavimas;
- analizės rezultatų pateikimas.

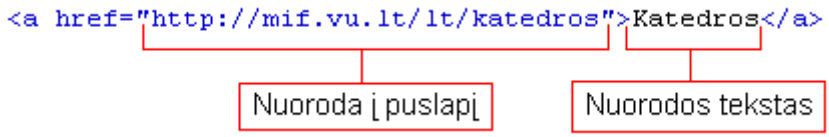
Šie komponentai ir bendra modelio schema pavaizduoti 5 pav.



5 pav. Tinklalapio navigavimo asociacijų analizės ir prognozavimo modelio schema

2.1 Tinklalapio puslapio HTML kodo analizės principai

Pirmas žingsnis, kurį turi atlikti kuriamas modelis kiekviename puslapyje – tai gauti to puslapio nuorodų struktūrą. Nuorodos HTML kalboje žymimos taip:



Kuriamas modelis išanalizuos duoto tinklalapio puslapio HTML kodą ir gaus to puslapio nuorodų struktūrą. Domins tik tos nuorodos, kurios turės nuorodos tekstą, kadangi pagal tą tekstą nuorodoms bus priskirti reitingai. Laikysime, kad vartotojai naudoja informacinę nuojautą, susidariusią iš nuorodos tekstų [Kle99]. Realiame gyvenime tai nebūtinai yra pilna sąlyga vartotojų informacijos nuojautai suformuoti, tačiau darbe bus apsiribota būtent nuorodų tekstų analize. Tai argumentuojama tuo, kad WEB puslapio HTML kodą yra sunku interpretuoti ir suskaidyti į fragmentus dėl skirtingų programuotojų kodavimo stilių [SDR+98]. Vieni gali fragmentus kurti lentelių pavidalu, naudojant antraštėms <b> tipo žymes, kiti antraštes gali žymėti <h1> ir fragmentus skaidyt žymėjimais t.t. Dėl to tampa beveik neįmanoma suskaidyti bet kokį puslapį į fragmentus taip, kaip tai daro gyvas žmogus [Kle99].

Apibendrinus, pateiksime šios analizės schemą:

Įeities duomenys	Išeities duomenys
Puslapio adresas. Pavyzdžiui: *http://mif.vu.lt/lt/*	Nuorodų masyvas. Pavyzdžiui: *Naujienos;* *Studijos;* *Katedros;* ...

2 Lentelė. Puslapio HTML kodo analizės įeities ir išeities duomenys.

2.2 Tinklalapio puslapio nuorodų struktūros analizės principai

Informacijos nuojauta yra pagrindinė priemonė, kuri padeda vartotojams ieškoti reikiamos informacijos tam tikrame tinklalapyje. Vartotojas sprendžia kokią nuorodą pasirinkt remdamasis jo informaciniu tikslu bei intuicija (informacijos nuojauta), kad pasirinktas nuorodos tekstas gali būti susijęs su jo ieškoma informacija. Teorinės analizės [Pir05] ir empiriniai papanaudojamumo tyrimai [SPB04] teigia, kad informacijos nuojauta (tinklalapio navigavimo asociacijos) yra pagrindinis faktorius naršant tinklalapius ir ieškant informacijos.

Taigi, iškyla natūralus klausimas – kaip galima išmatuoti informacijos nuojautą, kuri susidaro vartotojui žiūrint į tam tikrą puslapį? Tai padaryti yra labai sunku: jei vartotojas ieško žodžio „Kiaušinienė“, kokią nuorodą jis pasirinks – „Kiaušiniai“, „Receptai“, arba, galbūt, „Viengungių maistas“?

Dažniausiai vartotojams, užėjusiems į puslapi su tam tikru informaciniu tikslu iškyla tokios problemos analizuojant nuorodų tekstus bei sudarant informacijos nuojautą [BRP01]:

1. *Bazinių žinių, susijusių su puslapio tematika, trūkumas*. Pavyzdžiui, vartotojas užeina į ligoninės tinklalapį turint minimalias medicinos žinias. Jo informacinis tikslas yra surasti tam tikros ligos simptomus. Akivaizdu, kad jam bus labai sunku sudaryti informacijos nuojautą ir teisingai pasirinkti nuorodas, susijusias su jo informacijos tikslu. Tokiu atveju tinklalapio kūrėjų parinkti nuorodų tekstai gali būti vartotojui tiesiog nesuprantami. Taigi, iškyla ***nesuprantamų nuorodų problema***.
2. *Nuorodų tekstų semantinis panašumas*. Pavyzdžiui, vartotojas mato nuorodas, susijusias su jo informaciniu tikslu, tačiau nežino kurią pasirinkti. Pavadinkim tai nuorodų konkurencija. Įsivaizduokim tokią situaciją – kompiuterių parduotuvės tinklalapyje vartotojas ieško garso plokštės. Tinklalapio meniu yra tokios nuorodos: „Kompiuterių komponentai“, „Multimedia“ ir t.t. Kompiuterio garso plokštės suvokimas gali būti susijęs ir su kompiuterių komponentais, ir su multimedia. Kokią nuorodą turėtų pasirinkti vartotojas? Taigi, iškyla semantinės ***konkurencijos tarp nuorodų problema***.
3. *Klaidinantis nuorodų tekstas*. Tai toks nuorodų tekstas, kuris vartotojo manymu yra susijęs su jo informaciniu tikslu, bet iš tikrųjų tokios nuorodos pasirinkimas nuves vartotoją neteisingu informacijos paieškos keliu. Tai problema, kai teisingos nuorodos (teisinga nuoroda laikoma nuoroda, priartinanti vartotoją prie jo informacinio tikslo) tekstas mažiau asocijuojasi vartotojui su informaciniu tikslu, negu kitos nuorodos tekstas. Tai yra ***klaidinančių nuorodų problema***.

Galima padaryti išvadą, kad kuriamas modelis, analizuojant nuorodų tekstus, turi gebėti analizuoti puslapio nuorodų struktūrą (puslapio nuorodų tekstus), o tam jis ***turi turėti galimybę susieti nuorodų tekstus su turimu informaciniu tikslu***. Siekiant maksimaliai priartinti simuliavimo rezultatus prie realių vartotojų veiksmų, modelis turi turėti tam tikros tematikos „bazinių žinių“. Kitaip sakant, kaip ir tikri vartotojai, ***modelis turi turėti savo semantinę žodžių erdvę***. Pavyzdžiui, turint informacinį tikslą „Apple“ ir kompiuterių semantinę erdvę (t.y. semantinę erdvę, kurioje surinkti dokumentai susieti su kompiuterių tematika), modelis turi suprasti, kad žodžiai „Apple“ ir „Kompiuteris“ yra tarpusavyje labai susiję – tai reiškia, kad jei

vartotojas, ieškodamas termino „Apple“, pamato nuorodą „kompiuteriai“, tai jis greičiausiai ją paspaus. Tačiau kitoje semantinėje erdvėje, žodžiai „Apple“ ir „Kompiuteris“ gali būti tarpusavyje mažai susiję. Šis aprašytas reikalavimas bus kuriamo modelio pagrindu.

Egzistuoja keli algoritmai, kurie galėtų padėti įvykdyti apibrėžtą reikalavimą: Latent Semantic Analysis (LSA) [LFL98], Pointwise Mutual Information (PMI) [Tur01], ir Generalized Latent Semantic Analysis (GLSA) [MLF+05]. Iš jų buvo išrinktas vienas, kuris kartu su kitais algoritmais sudarys kuriamo modulio branduolį.

2.3 LSA ir SVD algoritmų pritaikymas

LSA – tai metodas, kuris naudojamas siekiant suformuoti semantinę erdvę iš tekstinių dokumentų [LFL98]. Jis remiasi faktu, kad tam tikri žodžiai pasitaiko panašiame kontekste ir taip galima nustatyti semantinius ryšius tarp tų žodžių. LSA leidžia palyginti tekstus tarpusavyje patikimesniu būdu, negu paprasčiausiai skaičiuojant žodžio panaudojimų skaičių kiekviename dokumente. Net jei keli žodžiai nebuvo naudojami kartu, jie gali būti semantiškai palyginti tarpusavyje.

LSA tikslas yra sukurti semantinę erdvę, kurioje žodžiai ir dokumentai gali būti palyginami tarpusavyje. Darbe LSA algoritmas pritaikomas apibrėžtiems tikslams. Naudojantis LSA gautais įvertinimais galima bus pasakyti ant kiek du žodžiai yra susiję tarpusavyje. Žodžiai – tai nuorodų tekstai. Nuorodų tekstai bus lyginami su turimu informaciniu tikslu. Taip kiekvienai nuorodai galima bus priskirti reitingą ir pasirinkti aukščiausią reitingą turinčią nuorodą. Kalbant apie realų vartotoją – gausis efektas, kai simuliuojamas nuorodos pasirinkimas, priklausantis nuo vartotojo turimų žinių (semantinės erdvės) bei informacinio tikslo.

Toliau apžvelgiamas SVD skaičiavimo algoritmas bei semantinės erdvės sukūrimas. Kartu tai leis tinklalapio navigavimo asociacijų analizės ir prognozavimo modelyje pritaikyti ir naudoti LSA algoritmą.

2.3.1 Algoritmas „Singular value decomposition“ (SVD)

Kadangi LSA naudoja SVD metodą, toliau pateikiamas jo aprašymas ir skaičiavimo algoritmas. SVD algoritmas išskaido originalią matricą į trijų matricų sandaugą $A=USV^T$ [LFL98]. Matrica U atvaizduoja pradinės matricos stulpelių vektorius (dokumentus), o matrica V – eilučių vektorius (žodžius). Matrica S – tai diagonali matrica, kuri naudojama siekiant gauti pradinę matricą sudauginus gautas tris matricas. Egzistuoja matematinis įrodymas, kad bet kokią matricą galima išskaidyti į tokių trijų matricų sandaugą [LFL98].

Naudojant LSA algoritmą pradinės matricos eilė mažinama siekiant gauti labiau apibendrintą semantinę erdvę, iš kuriuos galima būtų spręsti apie žodžių semantinį panašumą net

jei tie žodžiai nebuvo naudojami viename dokumente (pavyzdys 2.3.2 skyrelyje). Pradinės matricos eilę galima sumažinti paprasčiausiai palikus *n* pirmų eilučių matricoje S, o likusias pašalinti. Sudauginus U, V ir S matricas pagal aukščiau pateiktą formulę gaunama *n-tos* eilės pradinė matrica.

Išnagrinėkime uždavinį iš [LFL98], išskaidydami sprendimą į žingsnius, kuriuos naudosime kompiuteriniame algoritme. Šio uždavinio sprendime demonstruojamas pradinės matricos skaidymas į trijų matricų sandaugą. Pradinės matricos erdvė šiame pavyzdyje nėra mažinama.

Uždavinys: Pritaikyti SVD duotai matricai:

$$A = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$$

Sprendimas:

Žingsnis 1. Transponuoti matricą ir gauti A^T bei A^TA.

Kadangi $A^T = \begin{bmatrix} 4 & 3 \\ 0 & -5 \end{bmatrix}$ tai, $A^TA = \begin{bmatrix} 4 & 3 \\ 0 & -5 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$

$$A^TA = \begin{bmatrix} 25 & -15 \\ -15 & 25 \end{bmatrix}$$

Žingsnis 2. Gauti A^TA tikrines reikšmes ir surūšiuoti juos mažėjimo tvarka. Gauti vienetines reikšmes iš gautų tikrinių reikšmių (apskaičiuojant tikrinių reikšmių šaknis).

$$A^TA - cI = \begin{bmatrix} 25 - c & -15 \\ -15 & 25 - c \end{bmatrix}$$

$| A^TA - cI | = (25 – c)(25 – c) – (-15)(-15) = 0;$

$c^2 – 50c + 400 = 0;$

Ši kvadratinė lygtis turi du sprendimus, juos reikia surūšiuoti mažėjimo tvarka: $|40| > |10|$;

Tikrinės reikšmės yra tokios: $\mathbf{c_1} = 40$, $\mathbf{c_2} = 10$;

Vienetinės reikšmės: $\mathbf{s_1} = \sqrt{40} = 6.3245 > \mathbf{s_2} = \sqrt{10} = 3.1622$.

Žingsnis 3. Sukonstruoti matricą S. Tam reikia išrikiuoti vienetines reikšmes tos matricos diagonalėje mažėjimo tvarka. Sudaryti atvirkštinę matricą S^{-1}.

$$S = \begin{bmatrix} 6.3245 & 0 \\ 0 & 3.1622 \end{bmatrix} \qquad S^{-1} = \begin{bmatrix} 0.1581 & 0 \\ 0 & 0.3162 \end{bmatrix}$$

Žingsnis 4. Naudojant surūšiuotas tikrines reikšmes gautas antrame žingsnyje, apskaičiuoti A^TA tikrinius vektorius. Iš gautų reikšmių sudaryti matricą V ir apskaičiuoti V^T.

Sprendimas, kai $c_1 = 40$

$$A^TA - cI = \begin{bmatrix} 25-40 & -15 \\ -15 & 25-40 \end{bmatrix} = \begin{bmatrix} -15 & -15 \\ -15 & -15 \end{bmatrix}$$

$$(A^TA - cI)\,x_1 = 0$$

$$\begin{bmatrix} -15 & -15 \\ -15 & -15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-15x_1 + -15x_2 = 0$$
$$-15x_1 + -15x_2 = 0$$

Gaunam, kad $x_2 = -x_1$

$$x_1 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ -x_1 \end{bmatrix}$$

Dalinam iš jo ilgio,

$$L = \sqrt{x_1^2 + x_2^2} = x_1\sqrt{2}$$

$$x_1 = \begin{bmatrix} x_1 / L \\ -x_1 / L \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0.7071 \\ -0.7071 \end{bmatrix}$$

Sprendimas, kai $c_2 = 10$

$$A^TA - cI = \begin{bmatrix} 25-10 & -15 \\ -15 & 25-10 \end{bmatrix} = \begin{bmatrix} 15 & -15 \\ -15 & 15 \end{bmatrix}$$

$$(A^TA - cI)\,x_2 = 0$$

$$\begin{bmatrix} 15 & -15 \\ -15 & 15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$15x_1 + -15x_2 = 0$$
$$-15x_1 + 15x_2 = 0$$

Gaunam, kad $x_2 = x_1$

$$x_2 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_1 \end{bmatrix}$$

Dalinam iš jo ilgio,

$$L = \sqrt{x_1^2 + x_2^2} = x_1\sqrt{2}$$

$$x_2 = \begin{bmatrix} x_1 / L \\ x_1 / L \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}$$

Galiausiai gaunam, kad

$$V = \begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

Žingsnis 5. Sudaryti U: $U = AVS^{-1}$. Galiausiai SVD rezultatu bus matrica A, sudaryta iš $A=USV^T$.

$$U = AVS^{-1} = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix} \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \begin{bmatrix} 0.1581 & 0 \\ 0 & 0.3162 \end{bmatrix}$$

$$U = AVS^{-1} = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix} \begin{bmatrix} 0.1118 & 0.2236 \\ -0.1118 & 0.2236 \end{bmatrix}$$

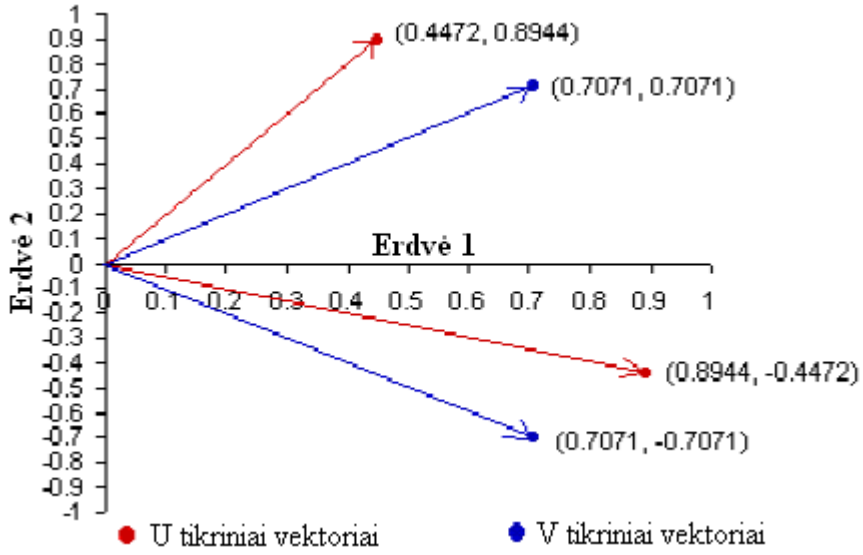
$$U = AVS^{-1} = \begin{bmatrix} 0.4472 & 0.8944 \\ 0.8944 & -0.4472 \end{bmatrix}$$

$$A = USV^{T} = \begin{bmatrix} 0.4472 & 0.8944 \\ 0.8944 & -0.4472 \end{bmatrix} \begin{bmatrix} 6.3245 & 0 \\ 0 & 3.1622 \end{bmatrix} \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

$$A = USV^{T} = \begin{bmatrix} 0.4472 & 0.8944 \\ 0.8944 & -0.4472 \end{bmatrix} \begin{bmatrix} 4.4721 & -4.4721 \\ 2.2360 & 2.2360 \end{bmatrix}$$

$$A = USV^{T} = \begin{bmatrix} 3.9998 & 0 \\ 2.9999 & -4.9997 \end{bmatrix} \approx \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$$

Taigi, išskaidėme pradinę matricą A į tris matricas (U, S, V). Matricos U ir V susideda iš tikrinių vektorių, o jų ortogonalumą galima pamatyti 6 pav. Šios matricos bus naudojamos siekiant gauti kelių žodžių panašumo įvertinimą. Apie tai bus kalbama vėliau.



6 pav. Matricų U ir V tikriniai vektoriai

2.3.2 Semantinės erdvės sukūrimo principai, LSA naudojimas

Įsivaizduokime, kad turime tokį dokumentų (tekstų) rinkinį:

- „The man walked the dog“;
- „The man took the dog to the park“;
- „The dog went to the park“.

Kaip jau buvo minėta, sudaroma žodžių ir dokumentų matricą (3 Lentelė):

	Dokumentas 1	Dokumentas 2	Dokumentas 3
the	2	3	2
man	1	1	0
walked	1	0	0
dog	1	1	1
took	0	1	0
to	0	1	1
park	0	1	1
went	0	0	1

3 lentelė. Žodžių ir dokumentų matrica.

Priminsime, kad žodžių ir dokumentų matricos susideda iš dokumentų (stulpeliai) bei unikalių žodžių (eilutės), naudojamų šiuose dokumentuose. Šiame pavyzdyje sudarytos matricos stulpeliai yra aprašyti dokumentai, o eilutės – unikalūs žodžiai. Kiekviena celė yra tam tikro žodžio pasikartojimų skaičius tam tikrame dokumente. Pavyzdžiui, žodis „dog“ yra naudojamas po vieną kartą visuose dokumentuose.

Sudarytai matricai pritaikomas SVD siekiant gauti dvi tikrinių vektorių aibes:

$$\begin{bmatrix} 0.46, 0.73, 0.51 \\ -0.77, 0.04, 0.64 \\ 0.45, -0.68, 0.58 \end{bmatrix} \begin{bmatrix} 5.03 \\ 1.57 \\ 1.09 \end{bmatrix} \begin{bmatrix} 0.82, 0.24, 0.09, 0.34, 0.14, 0.25, 0.25, 0.10 \\ -0.10, -0.47, -0.49, -0.06, 0.02, 0.43, 0.43, 0.40 \\ -0.01, -0.22, 0.41, 0.31, -0.63, -0.10, -0.10, 0.53 \end{bmatrix}$$

Sumažinama pradinės matricos eilė:

$$\mathbf{U} = \begin{bmatrix} 0.46, 0.73, 0.51 \\ -0.77, 0.04, 0.64 \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} 5.03 \\ 1.57 \end{bmatrix}$$

$$\mathbf{V} = \begin{bmatrix} 0.82, 0.24, 0.09, 0.34, 0.14, 0.25, 0.25, 0.10 \\ -0.10, -0.47, -0.49, -0.06, 0.02, 0.43, 0.43, 0.40 \end{bmatrix}$$

$$A = \begin{bmatrix} 2.0030 & 2.9955 & 2.0038 \\ 1.1064 & 0.8366 & 0.1380 \\ 0.8017 & 0.3047 & -0.2572 \\ 0.8491 & 1.2318 & 0.8043 \\ 0.3047 & 0.5319 & 0.3952 \\ 0.0474 & 0.9271 & 1.0615 \\ 0.0474 & 0.9271 & 1.0615 \\ -0.2572 & 0.3952 & 0.6664 \end{bmatrix}$$

Matrica *A* – tai sumažintos eilės pradinė matrica. Sumažinus matricos eilę pasikeitė ir kiekvienos celės reikšmės, kurios dabar reiškia kiek kartų tam tikras žodis *galėtų* būti naudojamas tam tikrame dokumente. Semantinė erdvė tapo bendresnė – celės su nulinėmis reikšmėmis dabar įgijo nenulines reikšmes. Pavyzdžiui, žodis „went“ antrame dokumente nebuvo naudojamas, tačiau bendresnėje semantinėje erdvėje tikimąsi, kad šis žodis antrame dokumente galėtų būti naudojamas 0,39 kartų – ir pagal semantinę antro dokumento prasmę tai iš tikrųjų yra tikėtina. Nėra konkrečių taisyklių iki kiek reikia mažinti pradinės matricos eilę – kiekvienai semantinei erdvei reikia eksperimentuojant ieškoti optimalų eilės skaičių.

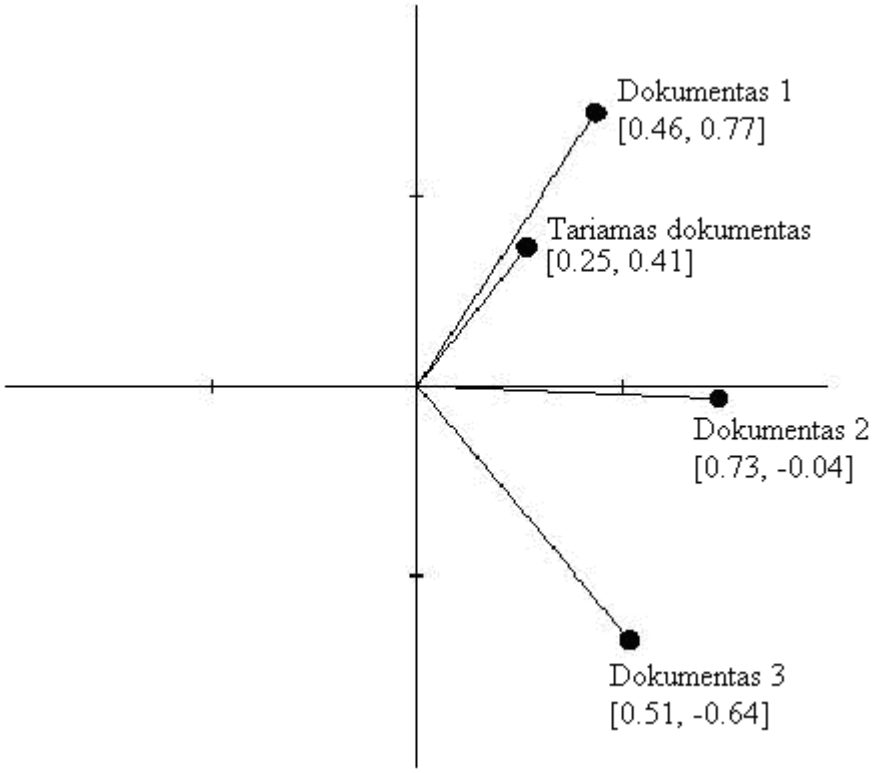
Matrica *U* atvaizduoja dokumentų aibę, o *V* atvaizduoja žodžių aibę. Jos bus naudojamos siekiant apskaičiuoti žodžių, arba dokumentų semantinį panašumą. Pavyzdžiui, dokumentų semantinį panašumą galima apskaičiuoti įkeliant naują dokumentą į semantinę erdvę ir apskaičiuojant pagal formulę: $p' = S^{-1} U^T p$. Taip gauname naujo vektoriaus koordinates sukurtoje semantinėje erdvėje. Kad būtų aiškiau pateiksime pavyzdį – palyginkime naują dokumentą „The dog walked“. Šis dokumentas tampa tariamu dokumentu sukurtoje semantinėje erdvėje:

p = [1, 0, 1, 1, 0, 0, 0, 0]

Jis įkeliamas į sukurtą semantinę erdvę ir jo panašumas skaičiuojamas padauginus tariamą dokumentą iš *S* atvirkštinės matricos ir matricos *U* (kadangi lyginame dokumentų panašumą):

$$p' = S^{-1} U^T p = \begin{bmatrix} 0.25 \\ 0.41 \end{bmatrix}$$

7 pav. demonstruoja, kad tariamas dokumentas pagal semantinę prasmę yra arčiausiai pirmo dokumento „The man walked the dog“. Reikia pastebėti, kad šiame pavyzdyje pavyko pavaizduoti dokumentų vektorius dvimatėje erdvėje, tačiau tai tik atskiras atvejis. Jei mes nemažintume matricos U eilės – vektoriai būtų vaizduojami trimatėje erdvėje, o bendru atveju gali būti naudojama *n-matė* erdvė.



7 pav. Dokumentų matricos tikrinių vektorių išdėstymas dvimatėje erdvėje.

Analogiškai galima skaičiuoti ir žodžių panašumą, tačiau dauginti reikia iš matricos *V*. Tačiau jei nuorodų tekstas, arba informacijos tikslas bus sudarytas tik iš vieno žodžio, aprašytas metodas nėra visiškai tinkamas. Taip yra dėl to, kad sudarius tariamą dokumentą jis neš labai mažai informacijos. Geriausiu atveju kažkoks sudaryto vektoriaus elementas bus didesnis, arba lygus 1. Padarysime prielaidą, kad koreliacijos koeficientų naudojimas galės tiksliau įvertinti žodžių panašumą. Darbe bus palyginti šie du aprašyti metodai ir bus parodyta, kad lyginant pavienius žodžius tikslesni rezultatai gaunami naudojant koreliacijos koeficientus.

Siekiant gauti geresnius rezultatus ir sumažinti skaičiavimus, sudarinėdami pradinę žodžių matricą galima ignoruoti tokius angliškus žodžius kaip „the“, „a“, „and“ ir t.t [WG03]. Taip pat galima nenaudoti ir panašių lietuviškų žodžių („ir“, „bet“ ir t.t.). Juos galima filtruoti įvedant minimalaus žodžio ilgio parametrą. Taip pat interpretuosime sakinius kaip atskirus dokumentus, kurie sudarys žodžių ir dokumentų matricą. Šiam tikslui įvesime papildomus parametrus – minimalaus sakinio ilgio parametrą ir parametrą, kuris nurodys kiek sakinių bus apjungta siekiant gauti vieną dokumentą.

2.3.3 Apibendrinimas

LSA algoritmo naudojimas leis kuriamam modeliui atlikti duoto puslapio nuorodų struktūros analizę. Šio sprendimo naujumas yra tame, kad žodžių semantinės erdvės analizė naudojama ir interpretuojama siekiant atsakyti į iškeltas problemas (nesuprantamų nuorodų

problema; konkurencijos tarp nuorodų problema; klaidinančių nuorodų problema). Pasirinkant semantinę erdvę bus galima simuliuoti vartotojų, turinčių atitinkamų žinių, informacijos paieškos kelius. Siekiant testuoti lietuviškus tinklalapius bus bandoma sukurti smulkią lietuvišką semantinę erdvę iš įvairių dokumentų.

<table>
<tr><th>Įeities duomenys</th><th>Išeities duomenys</th></tr>
<tr><td>Informacinis tikslas. Pavyzdžiui:
Safari;
Nuorodų tekstų masyvas.
Pavyzdžiui:
Africa
Asia
Europe
North America
South America</td><td>Nuorodų tekstų masyvas su priskirtais svorio reitingais.
Pavyzdžiui:<table><tr><td>Africa</td><td>0.41</td></tr><tr><td>Asia</td><td>0.05</td></tr><tr><td>Europe</td><td>0.02</td></tr><tr><td>North America</td><td>0.10</td></tr><tr><td>South America</td><td>0.12</td></tr></table></td></tr>
</table>

4 Lentelė. Puslapio nuorodų struktūros analizės įeities ir išeities duomenys.

2.4 Informacijos paieškos kelio formavimas

Šioje stadijoje jau yra vartotojo informacinis tikslas, galutinio puslapio (puslapio su ieškoma informacija) adresas bei duoto (t.y. šiuo metu analizuojamo) puslapio nuorodų struktūra su priskirtais reitingais. Kaip jau buvo minėta aprašant WebCriteria SiteProfile [SP], kuriamas modelis naudos programinius agentus (arba algoritmus) siekiant simuliuoti potencialių vartotojų veiklą Galutinis puslapio adresas bus lyginamas su analizuojamo puslapio adresu. Sutapimas reikštų, kad agentas pasiekė savo informacinį tikslą.

Anksčiau buvo minėta, kad vartotojai gali susidurti puslapyje su tam tikromis problemomis ir išskyrėme trijų tipų problemas:

- nesuprantamų nuorodų problema;
- konkurencijos tarp nuorodų problema;
- klaidinančių nuorodų problema.

Todėl kuriamas tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis naudos trijų tipų programinius agentus:

1. Pirmo tipo agentai padės nustatyti problemą, kai vartotojai nesupranta nuorodų tekstų. Turint informacinį tikslą ir nuorodų svorio reitingus šie agentai tikrins, ar teisingos nuorodos reitingas pagal LSA algoritmą yra mažesnis nei **n**, kas reikštų, kad

pasirinktoje semantinėje erdvėje toks nuorodos tekstas yra nesuprantamas. Modelyje įvesime parametrą (pirmas parametras), kuris leis apibrėžti (ir koreguoti) minimalaus svorio reitingo reikšmę. Taip pat galima bus nustatyti ar agentai, suradę nežinomas nuorodas, baigs savo darbą (tai atvaizduos vartotojus, kurie dėl žinių trūkumo palieka tinklalapį), arba naršys toliau, remdamiesi labai silpna informacine nuojauta.

2. Antro tipo agentai padės nustatyti konkurencijos tarp nuorodų problemą. Jei egzistuoja kelios nuorodos su beveik vienodais reitingais (iš vienos koreliacijos interpretacijos grupės), tarp kurių yra teisinga nuoroda, šie agentai atsitiktiniu būdu pasirinks vieną iš šių nuorodų. Vėliau, pritaikius IPS (ilgiausiai pasikartojanti seka) [PP99], bus parodytas paieškos kelias, ir perspėta apie konkurencijos tarp nuorodų problemą.
3. Trečio tipo agentai padės nustatyti problemą, kai teisingos nuorodos svorio reitingas yra mažas (antras parametras), bet egzistuoja kita nuoroda, kurios svorio reitingas yra pakankamai didelis (trečias parametras). Kitaip sakant, vartotojui susidaro klaidinga informacijos nuojauta ir jis pasirenka neteisingą nuorodą.

Taip pat bus įvesti papildomas parametras, leidžiantis apriboti agentų žingsnių skaičių (ketvirtas parametras). Tai leis išvengti problemos, kai agentai naršys ir analizuos tinklalapio puslapius nesurasdami savo tikslo, arba patekus į uždarą ciklą.

2.5 Analizės rezultatų pateikimas

Analizės rezultatas – agentų informacijos paieškos keliai. Kiekvieno tipo agentas pateiks informaciją, susijusią su jo analizuojamos problemos tipu. Bendru atveju analizės rezultatų pagrindas yra puslapių, kuriuos analizavo agentai, nuorodų struktūros įvertinimas (svorio reitingai). Taip pat bus žinomi visi informacijos paieškos žingsniai (kokios nuorodos ir kokiuose puslapiuose buvo pasirinktos). Tai sudarys pagrindą analizės ataskaitų pateikimui.

Vertintojas matys agentų informacijos paieškos kelią medžio pavidalu. Pasirinkus bet kokį puslapį bus pateikiama informacija apie to puslapio nuorodų struktūrą, jos svorio reitingai. Modelis praneš vertintojui kokios nuorodos yra nesuprantamos agentams, kokios yra konkuruojančios, kokia nuoroda buvo pasirinkta ir t.t. Remiantis šia informacija vertintojas darys sprendimus kaip galima pagerinti (pakeisti) nuorodų tekstus (arba nuorodų struktūrą), kad jos taptų vienareikšmiškai suprantamos vartotojams ir vartotojų informacijos paieškos kelias būtų kuo galima trumpesnis.

3 Modelio realizacija ir analizė

Kadangi vienas iš darbo tikslų yra sukurti vartotojo nuojautos automatinio vertinimo algoritmą – tai reiškia, kad visus aukščiau aprašytus darbo principinius sprendimus reikia realizuoti ir patikrinti jų korektiškumą ir veikimą praktikoje. Tam buvo sukurta windows forms tipo programa, kurioje realizuotas tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis.

Darbo įrankiai:

- Microsoft Visual Studio 2005 [MVS]
- RegexBuddy [RB].

Programavimo kalba:

- C#.

Pagalbiniai komponentai:

- BlueBit.MatrixLibrary [BBML] (30 day Trial).

Modelis suskaidomas į skirtingus komponentus (arba galima vadinti juos žingsniais):

- puslapio HTML kodo analizė;
- puslapio nuorodų struktūros analizė;
- informacijos paieškos kelio formavimas ir analizės rezultatų pateikimas.

Toliau aprašoma kiekvieno komponento realizacija, modelio taikymas ir atlikta gautų rezultatų analizė.

3.1 HTML kodo analizė

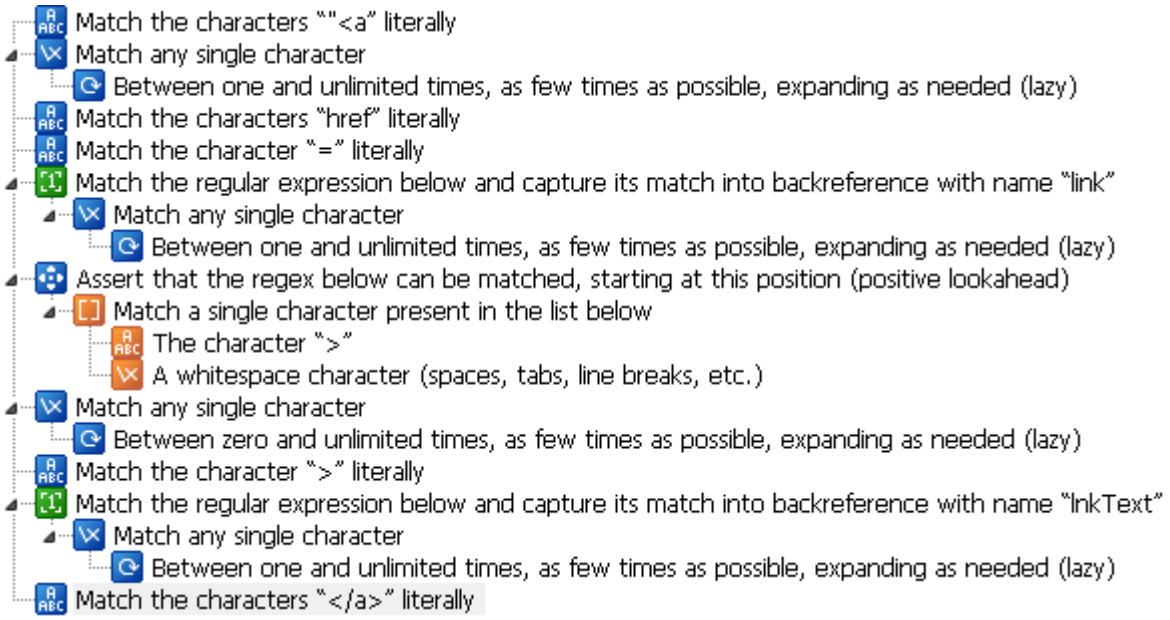
Tikslas: išanalizuoti duotą tinklalapio puslapį ir gauti visas duoto puslapio nuorodas, atitinkančias iškeltus reikalavimus:

- nuoroda turi prasidėti „http://...“;
- nuorodos tekstas negali būti tuščias.

Šiam uždaviniui labiausiai tinka Regular Expressions (Regex) naudojimas. Viena eilute galima sukurti šabloną (filtrą), pagal kurį bus atrenkami tinkami teksto fragmentai. Darbe Regex paieška pritaikoma duoto tinklalapio puslapio išeities tekste – taip gaunamos visos puslapyje aprašytos nuorodos:

```
    MatchCollection matchColl = Regex.Matches(strWebPageContent,
"<a.+?href\\=(?<link>.+?)(?=[>\\s]).*?>(?<lnkText>.+?)</a>",
RegexOptions.Singleline | RegexOptions.IgnoreCase);
```

Čia „strWebPageContent“ – kintamasis, kuriame saugomas puslapio išeities tekstas. Šio kodo rezultatas – visos nuorodos ir nuorodų tekstai bus patalpinti į matchColl kintamąjį. Pritaikytą regex filtą galima paaiškinti RegexBuddy [RB] programos pagalba, kuri aprašo bet kokio regex filtro taisykles (8 pav.):



8 pav. Regex filtro, skirto gauti nuorodų struktūrą, taisyklės

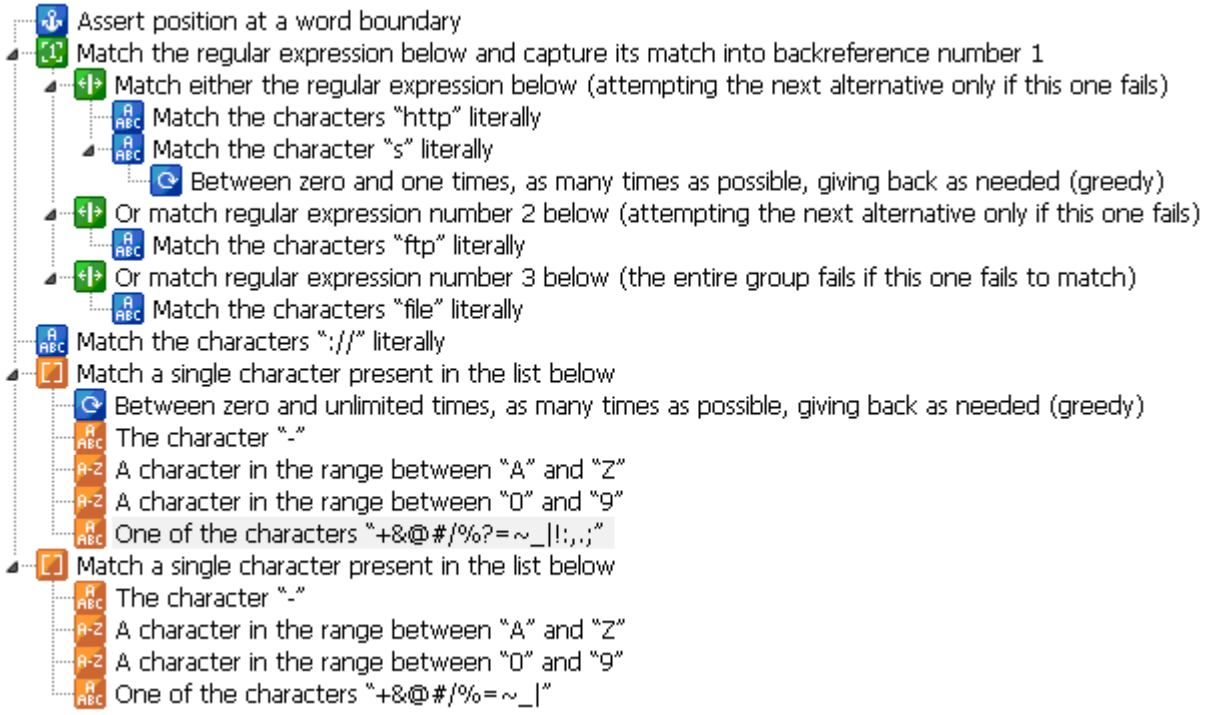
Trumpai tariant, aprašyto filtro dėka galima išgauti visas puslapio nuorodas nepraleidžiant net sudėtingai aprašytų nuorodų, tokių kaip, pavyzdžiui, *„Apie“*. Šitą nuorodą aprašytas regex filtras išskaidys į dvi dalis:

1. Nuorodą – *„http://mif.vu.lt/lt/apie“*.
2. Nuorodos tekstą – *„Apie“*.

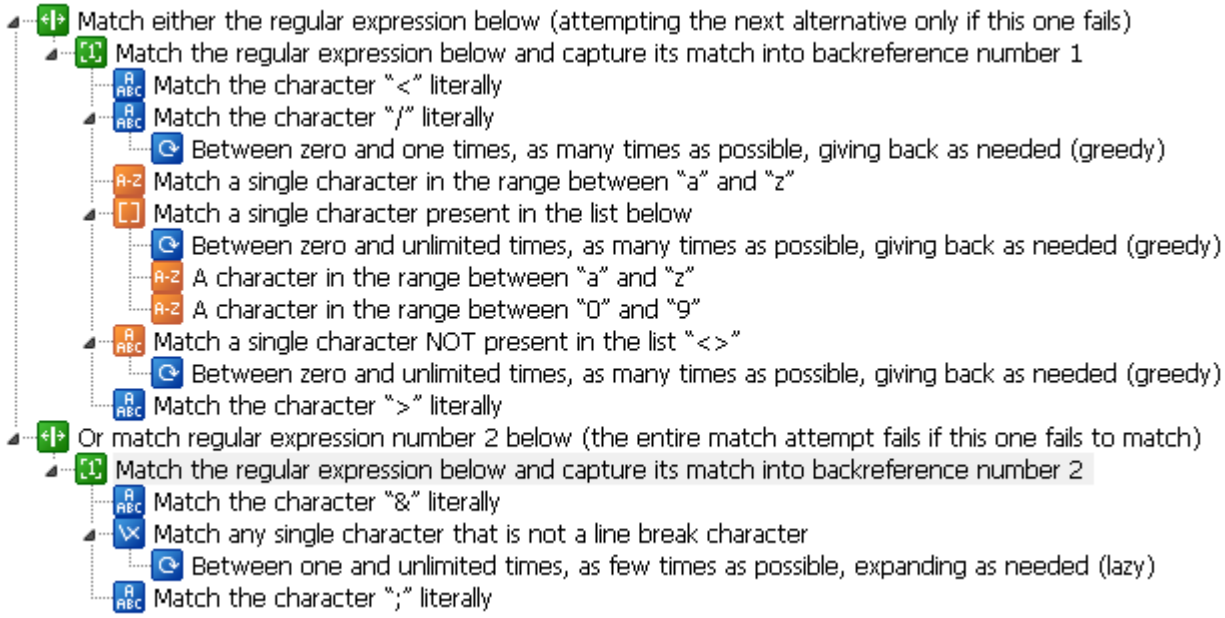
Tačiau siekiant gauti korektiškas nuorodas ir nuorodų tekstus reikia atlikti dar du žingsnius:

1. Įsitikinti, kad nuoroda atitinka reikalavimą – prasideda „http://...“. Šiam tikslui naudojamas kitas regex filtras:
   "\\b(https?|ftp|file)://[-A-Z0-9+&@#/%?=~_|!:,.;][-A-Z0-9+&@#/%=~_|]"*
   Šio filtro taisyklės aprašytos 9 pav.
2. Išimti visas HTML žymas ir specialius simbolius iš nuorodos teksto. Šis žingsnis reikalingas, nes nuorodos tekste gali būti naudojamos HTML žymos, pavyzdžiui, „“, „“ ir t.t. bei specialūs simboliai, pavyzdžiui, „ “ ir panašiai. Šiam tikslui naudojamas toks regex filtras:
   "(</?[a-z][a-z0-9][^<>]*>)|(&.+?;)"*

Šio filtro taisyklės aprašytos 10 pav.



9 pav. Regex filtro, skirto patikrinti nuorodų korektiškumą, taisyklės



10 pav. Regex filtro, skirto pašalinti HTML žymas ir spec. simbolius, taisyklės

Puslapio HTML kodo analizės žingsnio rezultatas – korektiškų nuorodų bei nuorodų tekstų sąrašas. Šis sąrašas paduodamas sekančiam žingsniui – puslapio nuorodų struktūros analizės komponentui, kuris kiekvienam nuorodos tekstui, atsižvelgiant į informacinį tikslą, priskirs reitingą ir surūšiuos gautą sąrašą pagal reitingus mažėjimo tvarka.

3.2 Nuorodų struktūros analizė

Šiame žingsnyje jau yra duoto puslapio korektiškas nuorodų ir nuorodų tekstų sąrašas.

Tikslas: Kiekvienam nuorodos tekstui sudarytį reitingą, atsižvelgiant į informacinį tikslą. Reitingas – tai skaičius nuo [-1; 1], kuris nusako kaip nuorodos tekstas ir informacinis tikslas yra susiję (semantine prasme). Kitaip sakant, reitingas – tai koreliacijos koeficientas tarp nuorodos teksto ir informacinio tikslo. Reitingo stiprumo interpretacija irgi yra tokia pat:

- nuo 0,9 iki 1,0 arba nuo -0,9 iki -1,0 – koreliacija laikoma labai stipria;
- nuo 0,7 iki 0,9 arba nuo -0,7 iki -0,9 – stipri koreliacija;
- nuo 0,4 iki 0,7 arba nuo -0,4 iki -0,7 – koreliacija yra vidutine;
- nuo 0,2 iki 0,4 arba nuo -0,2 iki -0,4 – koreliacija laikoma silpna ir galiausiai;
- nuo 0,2 iki -0,2 – labai silpna arba nereikšminga koreliacija.

Puslapio nuorodų struktūros analizės komponentas susideda iš tokių dalių:

1. Semantinės erdvės sukūrimas – formuojama žodžių ir dokumentų matrica, kurioje kiekviena eilutė – tai unikalus žodis, o stulpelis - unikalus dokumentas. Kaip buvo minėta, darbe dokumentu laikoma bet kokia žodžių seka (sakinys, pastraipa, tekstinis dokumentas ir t.t.).
2. Reitingų priskyrimas nuorodoms – priklausomai nuo nuorodos teksto ir informacinio tikslo (ar tai pavieniai žodžiai, ar sakiniai) reitingai skaičiuojami pagal vieną iš šių algoritmų:
    a. Žodis su žodžiu tipo palyginimas.
    b. Dokumentas su dokumentu tipo palyginimas.

3.2.1 Semantinės erdvės sukūrimas

Šiame žingsnyje reikia iš duoto tekstinio dokumento (arba kelių tekstinių dokumentų) suformuoti žodžių ir dokumentų matricą. Įvairių tematikų semantinių erdvių kūrimui naudosime po vieną word tipo dokumentą. Semantinės erdvės matricoje stulpeliai (tariami dokumentai) bus sudaromi apjungiant nurodytą skaičių sakinių iš nurodyto tekstinio failo.

Žingsnis 1. Visi unikalūs žodžiai ir apjungti sakiniai talpinami į `Hashtable` tipo kintamuosius:

```
Hashtable htDocs = new Hashtable(); // dokumentų sąrašas
Hashtable htWords = new Hashtable(); // žodžių sąrašas

foreach (string sentence in fileContent.Split('.', '?', '!'))
{
```

```
        if (sentence.Trim().Split(' ').Length >= minSentenceLength)
        {
           ...
           foreach (string word in sentence.Trim().Split(' ','-'))
           {
             //iš sakinių imami žodžiai ir talpinami į htWords kintamąjį
             ...
             string wordToAdd = (word.Length < 8) ?
                       word.Remove(word.Length – 2).ToLower() :
                       word.Remove(word.Length - 3).ToLower();
             ...
           }
           ...
        }
        ...//sakiniai talpinami į htDocs kintamąjį
    }
```

Kadangi vienas žodis gali būti naudojamas su skirtingomis galūnėmis (linksniuojamas), reikia pasistengi neatsižvelgti į žodžių galūnes, kadangi priešingu atveju galima gauti labai apytikslius rezultatus (tas pats žodis, bet skirtingai sulinksniuotas, bus traktuojamas kaip keli skirtingi unikalūs žodžiai). Todėl, įtraukiant žodžius į htWords kintamąjį, ištrinam paskutinius du (jei žodis yra trumpesnis arba lygus *n*), arba tris (jei jis yra ilgesnis už *n*) simbolius. Darbo metu atlikti bandymai rodo, kad geriausi rezultatai gaunami kai *n* yra lygus 8

Žingsnis 2. Sukuriama ir užpildoma semantinės erdvės matrica:

```
//Sukuriame semantinės erdvės matricą
int m = htWords.Count;
int n = htDocs.Count;
Matrix a = new Matrix(m, n);
//Užpildome semantinės erdvės matricą
for (int i = 0; i < htDocs.Count; i++)
{
    for (int j = 0; j < htWords.Count; j++)
    {
        a[j, i] = CountWordsInDoc(htWords[j].ToString(),
                                  tDocs[i].ToString());
    }
}
```

Šiame žingsnyje svarbiausia yra apskaičiuoti kiek kartų duotas žodis pasitaiko duotame dokumente. Tam naudojama `CountWordsInDoc` funkcija:

```
private int CountWordsInDoc(string word, string doc)
{
    int count = 0;
    foreach (string split in doc.Replace(".", "").Split(' '))
    {
        if (split.ToLower().IndexOf(word.ToLower()) == 0 &&
                                split.Length – word.Length <= 3) count++;
    }
    return count;
}
```

Čia žiūrima, kad ieškomas žodis ir sakinio žodis sutaptų taip, kad sutapimas prasidėtų nuo pirmo simbolio ir lyginamų žodžių ilgiai skirtųsi ne daugiau kaip 3 simboliais. Taip yra dėl to, kad talpinant žodžius į sąrašą yra ištrinamos jų galūnės (daugiausiai 3 simboliai), todėl jei duotas žodis ir sakinio žodis yra tas pats žodis – ilgių skirtumas bus ne didesnis negu 3 simboliai. Pavyzdžiui, ieškomas žodis yra „žmogus“, o sakinys yra „žmogui, žmogiškas, žmogėnas“. Ieškomas žodis „žmogus“ žodžių sąraše bus išsaugotas kaip „žmog“, todėl, jei neskaičiuotumę žodžių ilgių skirtumų, funkcija grąžintų, kad duotame sakinyje yra 3 sutapimai, tačiau iš tikrųjų yra tik vienas – „žmogui“.

Žingsnis 3. Sukurtai semantinės erdvės matricai pritaikomas SVD algoritmas; gaunamos U, S, V matricos; gautos matricos sumažinamos iki duotos erdvės; paskaičiuojama nauja semantinės erdvės matrica (pavadinkime ją USV) pagal formulę: $USV = U*S*V^T$.

```
//Naudojant Bluebit.MatrixLibrary klasę apskaičiuojama SVD pagal (2.3.1)
//skyriuje aprašytą algoritmą.
SVD svd = new SVD(a); //a – apskaičiuota anksčiau semantinės erdvės
                      //matrica
//Gaunam U, S, V matricas ir sumažinam jas iki duotos erdvės (dimension)
Matrix U = svd.U; U.Resize(U.Rows, dimention); U.Resize(svd.U.Rows,
                                                        svd.U.Cols);
Matrix S = svd.S; S.Resize(dimention, dimention); S.Resize(svd.S.Rows,
                                                        svd.S.Cols);
Matrix V = svd.V; V.Resize(V.Rows, dimention); V.Resize(svd.V.Rows,
                                                        svd.V.Cols);
//Apskaičiuojam naują semantinės erdvės matricą (pavadinkim ją USV)
Matrix USV = U * S * V.Transpose();
```

3.2.2 Reitingų priskyrimas nuorodoms

Turint sumažintos erdvės semantinę matricą, galima kiekvienam nuorodos tekstui apskaičiuoti reitingą. Kaip buvo minėta, siekiant gauti tikslesnius rezultatus, bus naudojami du algoritmai: pirmas – kai lyginami žodžiai (CompareTerms funkcija), antras – kai lyginami tariami dokumentai (CompareDocs funkcija).

Lyginant žodžius bus naudojami koreliacijos koeficientai – **r**, o lyginant dokumentus iš pradžių jie bus įtraukiami į sukurtą semantinę erdvę, o po to pagal formulę apskaičiuojam reitingus.

Žodis su žodžiu tipo palyginimas

Semantinės erdvės matricos eilutės – tai unikalūs žodžiai, todėl, siekiant palyginti tam tiktus žodžius, reikia apskaičiuoti **r** koreliacijos koeficientą tarp atitinkamų matricos eilučių.

Šiame žingsnyje jau yra galutinė semantinės erdvės matrica – USV, todėl uždavinys yra surasti joje dominančių žodžių eilučių indeksus. Suradę eilučių indeksus apskaičiuosime koreliacijos koeficientus tarp jų.

```
//Ieškom norimų žodžių indeksus matricoje (paieška vykdoma pagal tą
//patį principą, kaip ir CountWordsInDoc funkcija).
foreach (int key in htWords.Keys)
{
    if (search1.IndexOf(htWords[key].ToString()) == 0 &&
             search1.Length –  htWords[key].ToString().Length <= 3)
    { i1 = key;}
    if (search2.IndexOf(htWords[key].ToString()) == 0 &&
             search2.Length – htWords[key].ToString().Length <= 3)
    { i2 = key; }
}

//Naudojant Bluebit.MatrixLibrary klasę apskaičiuojama koreliacija pagal
//(1.4) skyriuje aprašytą algoritmą.
if (i1 >= 0 && i2 >= 0)
{
    Int rating = USV.ColCorrelation(i1, i2).ToString();
}
```

Dokumentas su dokumentu tipo palyginimas

Siekiant palyginti du dokumentus, visų pirmą juos reikia įtraukti į sukurtą semantinę erdvę. Neužtenka tiesiog paskaičiuoti kiek kartų tam tikri žodžiai pasitaiko šiuose dokumentuose, reikia

gauti šių dokumentų vektorines išraiškas sumažintos erdvės semantinės erdvės matricoje. Tai atliekama tam, kad būtų galima palyginti du dokumentus semantinės erdvės pagrindu (taip pat kaip ir lyginant žodžius). Programoje tai atliekama trimis žingsniais:

1. Pirmas žingsnis – sudaromas pradinis dokumento vektorius. Jis sudaromas lygiai taip pat kaip ir pradinė semantinės erdvės matrica, t.y. skaičiuojama kiek kartų tam tikras unikalus žodis pasitaiko dokumente:

```
Vector q1 = new Vector(htWords.Count);
foreach (int key in htWords.Keys)
{
q1[key] = CountWordInDoc(htWords[key].ToString(),
textBox2.Text);
}
```

2. Antras žingsnis – sudarytas vektorius įtraukiamas į sumažintos erdvės semantinę matricą. Tai atliekama pagal formulę $q1' = S^{-1} * U^{T} * q1$ [LFL98].

```
Vector q1_reduced = ((Matrix)(S.Inverse() * U.Transpose() *
q1)).ColVector(0);
```

3. Trečias žingsnis, kai jau yra apskaičiuoti dokumentų vektoriai, galima juos palyginti pagal formulę [KG90]:

$$\mathbf{sim(q,\ d) = \frac{q \bullet d}{|\,q\,|\,|\,d\,|}}$$

Pirmas dokumentas visada bus apibrėžtas informacinis tikslas, todėl jo vektorinė išraiška nesikeis; antras dokumentas – tai analizuojama nuoroda, kuriai priskiriamas reitingas.

```
//pageAnalyse.extractedLinkList – tai gautas puslapio nuorodų struktūros
//sąrašas;
//ExtractedLink – tai klasė, atvaizduojanti vieną nuorodą (nuoroda, nuorodos
//tektas, reitingas);
//q1 – informacinio tikslo vektorius, q2 – nuorodos vektorius.
foreach (ExtractedLink extractedLink in pageAnalyse.extractedLinkList)
{
      Vector q2 = new Vector(htWords.Count);
      foreach (int key in htWords.Keys)
      {
            q2[key] = CountWordInDoc(htWords[key].ToString(),
                                                extractedLink.URLText);
      }
      Vector q2_reduced = ((Matrix)(S.Inverse() * U.Transpose() *
                                                      q2)).ColVector(0);
```

```
        double result = Vector.DotProduct(q1_reduced, q2_reduced) /
                                        (q1_reduced.Norm() * q2_reduced.Norm());
        extractedLink.Rating = result;
}
```

Puslapio nuorodų struktūros analizės rezultatas – pilnai paruoštas `ExtractedLinkList` tipo sąrašas, kuris sudaromas iš `ExtractedLink` tipo objektų. `ExtractedLink` klasė – tai klasė, atvaizduojanti išgautą nuorodą su priskirtu jau reitingu (nuoroda, nuorodos tekstas, reitingas):

```
public class ExtractedLink
{
    private string m_URL = "";
    public string URL
    {
        get { return m_URL; }
        set { m_URL = value; }
    }

    private string m_URLText = "";
    public string URLText
    {
        get { return m_URLText; }
        set { m_URLText = value; }
    }

    private double m_Rating = -1;
    public double Rating
    {
        get { return m_Rating; }
        set { m_Rating = value; }
    }

    public ExtractedLink()
    {
    }

    public ExtractedLink(string url, string urlText)
    {
        URL = url;
        URLText = urlText;
    }
```

```
}

public class ExtractedLinkList : List<ExtractedLink>
{ }
```

Galiausiai surūšiuojam nuorodų sąrašą pagal reitingus mažėjimo tvarka:

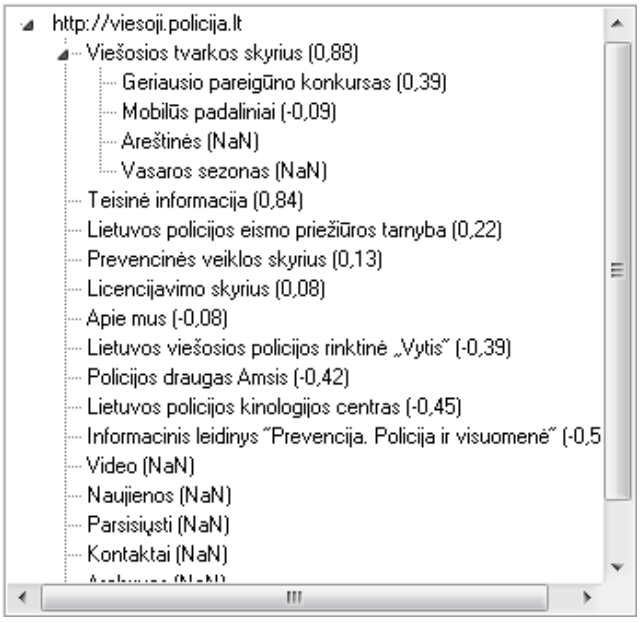
```
pageAnalyse.extractedLinkList.Sort(delegate(ExtractedLink link1,
                                            ExtractedLink link2)
                { return link2.Rating.CompareTo(link1.Rating); }
                          );
```

3.3 Informacijos paieškos kelio formavimas ir analizės rezultatų pateikimas

Šiame žingsnyje jau yra duoto puslapio korektiškas ir surūšiuotas nuorodų ir nuorodų tekstų sąrašas su priskirtais reitingais.

Tikslas: Suformuoti galutinį informacijos paieškos kelią ir pateikti analizės rezultatus.

Visų pirmą, einamojo puslapio nuorodos įtraukiamos į paieškos kelio medį. Šis medis pateiks vertintojui hierarchinį paieškos kelią. Medyje kiekviena šaka – tai nuorodos tekstas ir nuorodos reitingas. Kadangi praeitame žingsnyje gautas nuorodų sąrašas yra surūšiuotas reitingų mažėjimo tvarka – tai bus atvaizduota medyje (11 pav.):



11 pav. Galutinio medžio, atvaizduojančio paieškos kelią, pavyzdys

11 pav. pavaizduotame medyje matosi, kad iš pradinio puslapio nuorodų buvo pasirinkta nuoroda, turinti aukščiausią reitingą – „Viešosios tvarkos skyrius“. Po to algoritmas kartojasi, tačiau pradinio puslapio adresu jau laikoma pasirinkta nuoroda.

Anksčiau buvo minėta, kad vartotojai gali susidurti puslapyje su tam tikromis problemomis ir buvo išskirtos trijų tipų problemos:

- nesuprantamų nuorodų problema;
- konkurencijos tarp nuorodų problema;
- klaidinančių nuorodų problema

Todėl, formuojant paieškos kelią, reikia analizuoti šias problemas ir pranešti apie jas vertintojui. Trečio tipo problemų automatinis vertinimas neturi prasmės, nes vertintojas ir pats žino teisingą paieškos kelią, o klaidinančias nuorodas jis mato pagal reitingų dydžius. Tačiau reikia pateikti vertintojui galimybę rankiniu būdu pasirinkti norimą nuorodą (nepriklausomai nuo reitingų) ir atlikti tolimesnį paieškos kelio formavimą bei klaidinančių nuorodų problemų analizę. Modelio realizacijoje ši galimybė yra numatyta dvigubai paspaudžius ant norimo nuorodos teksto paieškos kelio medyje.

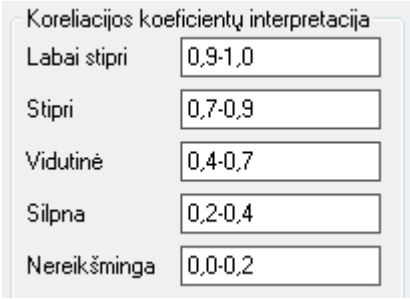
Kitų tipų problemos automatiškai analizuojamos formuojant paieškos kelią (jos gali pasireikšti vienu metu):

Konkurencijos tarp nuorodų problema

Aprašant koreliaciją (1.4 skyrius) buvo minėta, kad koreliacija gali būti labai stipri, stipri, vidutinė, silpna ir nereikšminga (pavadinkim tai koreliacijos interpretacijomis):

```
public enum CorrelationMeanings
        {
            VeryStrong,
            Strong,
            Medium,
            Weak,
            VeryWeak,
            None
        }
```

Modelio realizacijoje numatyta galimybė koreguoti koreliacijos koeficientų rėžius, taip apibrėžiant šias koreliacijos interpretacijas:



12 pav. Koreliacijos koeficientų rėžiai, apibrėžiantys koreliacijos interpretacijas

Darbe laikoma, kad konkurencijos tarp nuorodų problema yra tada, kai aukščiausius reitingus turinčios nuorodos priklauso vienodai koreliacijos interpretacijai. Kitaip sakant, šių nuorodų reitingai yra vienodame intervale:

```
//Funkcija gražina duoto skaičiaus koreliacijos interpretaciją
public CorrelationMeanings GetCorrelationMeaning(double number)
{
   ...
}

if (GetCorrelationMeaning(pageAnalyse.extractedLinkList[0].Rating) ==
       GetCorrelationMeaning(pageAnalyse.extractedLinkList[1].Rating))
{
      tbOutput.Text += "Konkurencijos tarp nuorodų problema." +
                                                  Environment.NewLine;
}
```

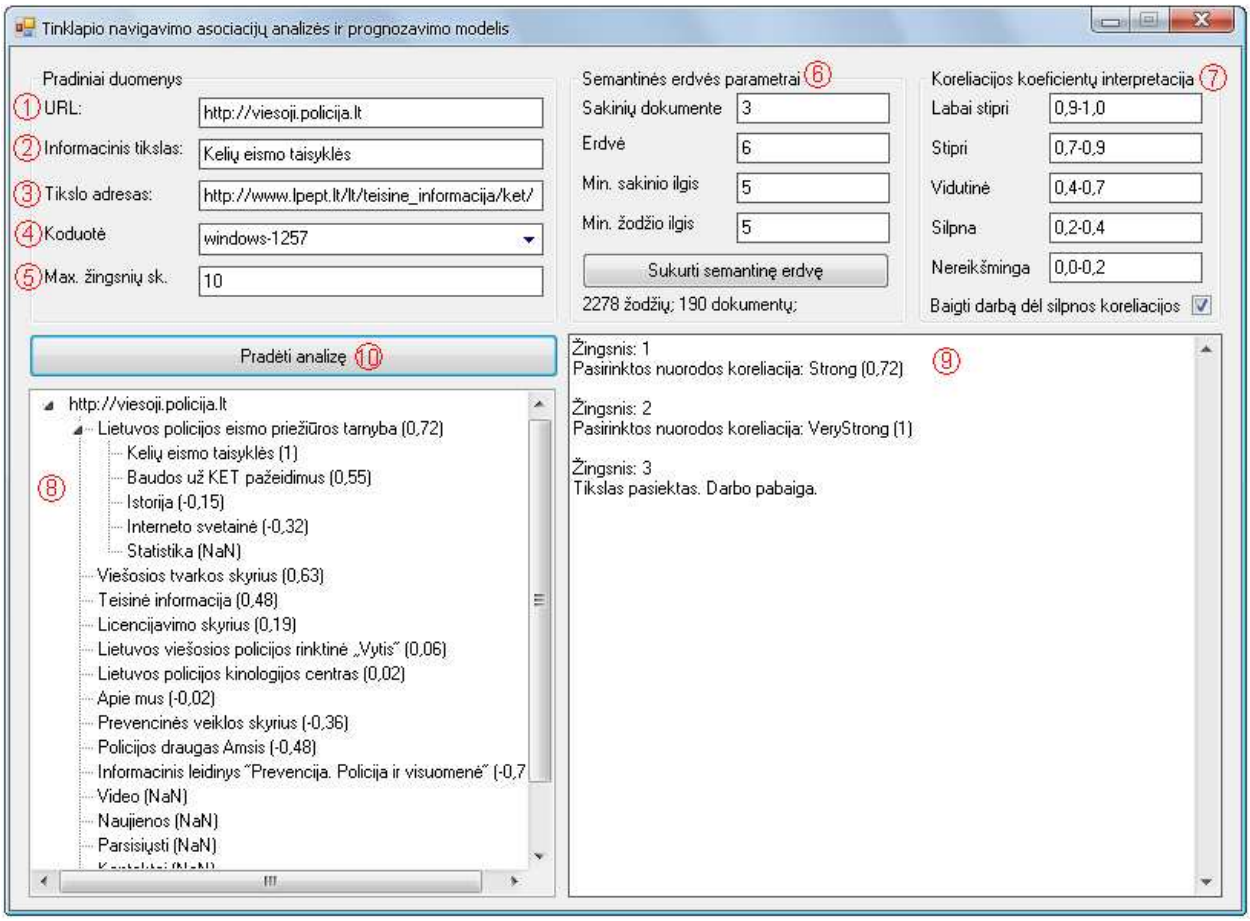
Nesuprantamų nuorodų problema

Šio tipo problema pasitaiko kai didžiausias nuorodų reitingas priklauso silpnai, arba žemesnei koreliacijos interpretacijai:

```
double rating = Math.Round(pageAnalyse.extractedLinkList[0].Rating, 2);
if (GetCorrelationMeaning(rating) == CorrelationMeanings.Weak ||
       GetCorrelationMeaning(rating) == CorrelationMeanings.VeryWeak ||
       GetCorrelationMeaning(rating) == CorrelationMeanings.None)
{
      tbOutput.Text += "Nesuprantamų nuorodų problema (arba nuorodos
      nesusijusios su tikslu). Darbo pabaiga." + Environment.NewLine;
      //Ar baigti darbą dėl silpnos koreliacijos?
      if (cbExitOnWeakCorrelation.Checked) return;
}
```

Kai pasireiškia nesuprantamų nuorodų problema vartotojas arba palieka tinklalapį (visi nuorodos tekstai jam nesuprantami), arba jis bando sudaryti paieškos kelią remiantis silpna puslapio nuojauta. Todėl modelio realizacijoje yra galimybė simuliuoti abu šiuos atvejus – šios galimybės valdomos pažymėjus varnelę „Baigti darbą dėl silpnos koreliacijos“.

Galutinis modelio realizacijos langas pavaizduotas 13 pav.:



13 pav. Galutinis modelio realizacijos langas

13 pav. paaiškinimai:

1. Pradinis tinklalapio adresas.
2. Informacinis tikslas. Nusakomas vienu ar daugiau žodžių arba sakinių.
3. Informacinio tikslo adresas.
4. Tinklalapio koduotė.
5. Maksimalus agentų žingsnių skaičius.
6. Semantinės erdvės parametrai. Į šiuos parametrus įeina sakinių dokumente skaičius, semantinės erdvės matricos erdvė, minimalus sakinių bei žodžių ilgiai. Apačioje vaizduojama informacija apie sukurtą semantinę erdvę.

7. Koreliacijos koeficientų interpretacijos nustatymai. Nustatomi kiekvienos interpretacijos rėžiai.
8. Analizės rezultatų pateikimas – informacijos paieškos kelias medžio pavidalu. Medyje vaizduojama tinklalapio nuorodos struktūra, kuri buvo analizuojama formuojant paieškos kelią.
9. Analizės rezultatų pateikimas – analizės žingsnių ir galimų problemų aprašymai.
10. Analizės pradžios mygtukas.

Galutinis analizės rezultatas – tai informacijos paieškos kelias, pavaizduotas medžio pavidalu bei žingsnių ir potencialių problemų aprašymas. Remiantis paieškos kelio medžiu, nuorodų reitingais ir aprašymu vertintojas gali daryti išvadas apie tinklalapio nuorodų struktūros patogumą ir suprantamumą.

3.4 Modelio taikymas ir gautų rezultatų analizė

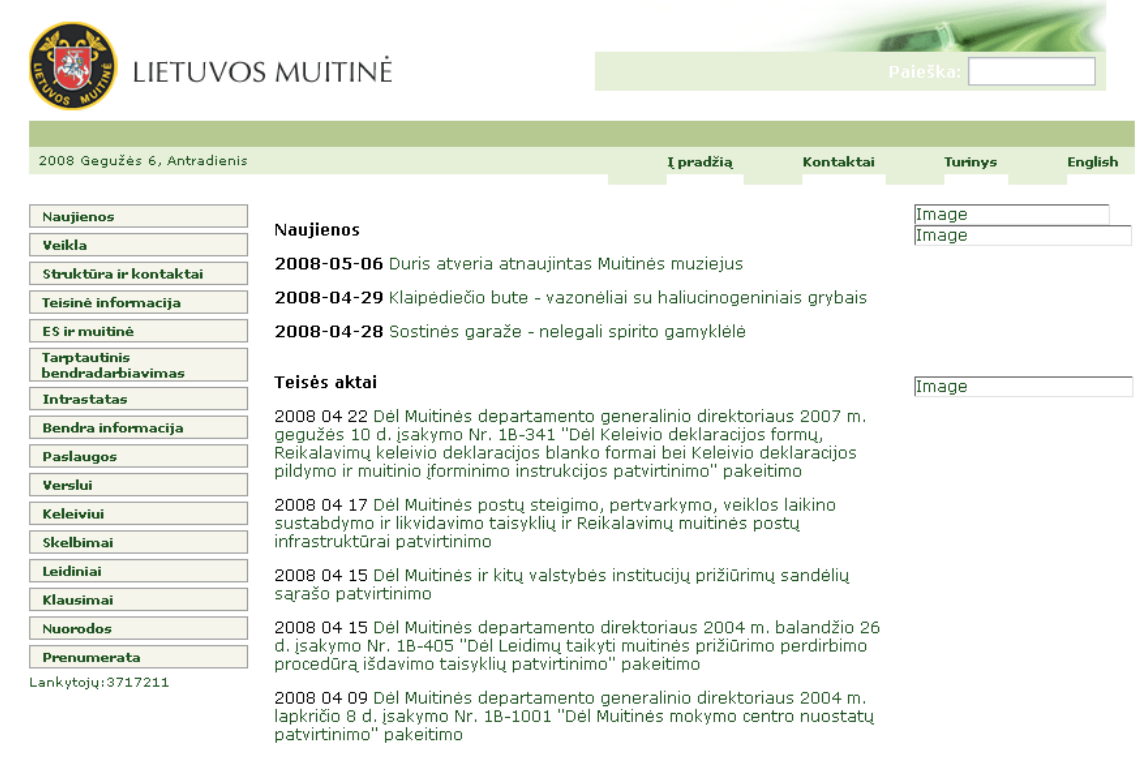
Toliau sukurta modelio realizacija pritaikoma praktikoje ir gauti rezultati palyginami su realių vartotojų rezultatais.

Atliktuose bandymuose dalyvavo 20 žmonių grupė. Kiekvieno bandymo metu grupei buvo duodamas tinklalapis bei informaciniai tikslai. Gauti realių vartotojų paieškos keliai buvo lyginami su modelio gautais rezultatais, buvo stebima ar modelio prognozuotos problemos pasitaikė realybėje.

Kiekvienam analizuojamam tinklalapiui buvo sukurta jo tematiką atitinkanti semantinė erdvė. Semantinės erdvės buvo kuriamos iš referatų apie policiją ir Lietuvos muitinės veiklą (referatus pateikė ir leido naudoti jų autoriai). Analizuojant buvo naudojami pradiniai modelio nustatymai (semantinės erdvės parametrai ir koreliacijos interpretacijos parametrai). Šie nustatymai buvo gauti modelio realizacijos kūrimo metu atlikus daugybę bandymų su įvairiais skaičiais. Buvo pastebėta, kad didžiausią įtaka rezultatams daro semantinės erdvės matricos eilės skaičius.

Analizuojami tinklalapiai – tai http://viesoji.policija.lt (14 pav.) ir http://www.cust.lt (15 pav.). Šių dviejų tinklalapių pilnai pakanka modelio testavimui, kadangi jie yra skirtingų tematikų ir jiems analizuoti reikalingos skirtingos semantinės erdvės. Taip pat šie tinklalapiai turi patogią hierarchinę nuorodų struktūrą, kurią patogu analizuoti.

14 pav. viesoji.policija.lt tinklalapio pradinis puslapis

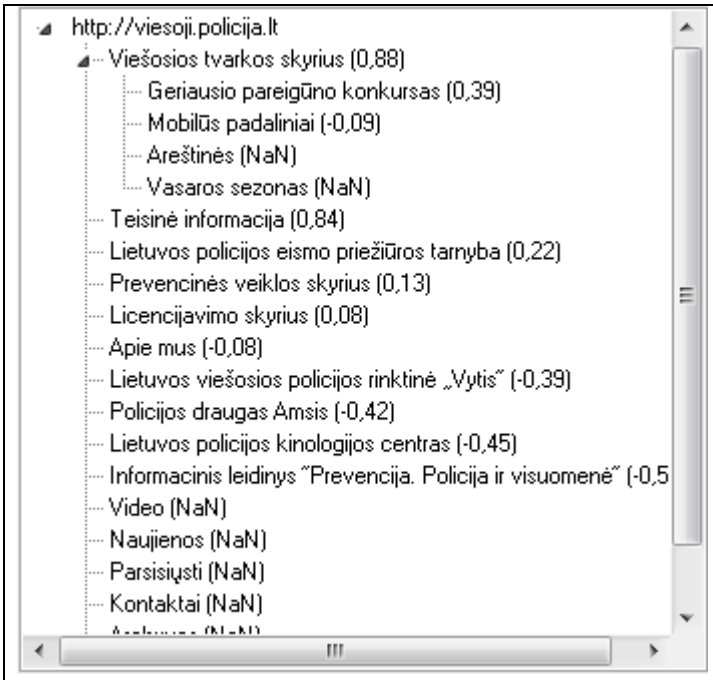


15 pav. cust.lt tinklalapio pradinis puslapis

Bandymas 1

Analizuojamas tinklalapis	Bandyme dalyvavo	Semantinės erdvės dydis (žod./ dok.); Eilė	Informacinis tikslas	Teisingas informacijos paieškos kelias
http://viesoji.policija.lt	20 dalyvių	2278/190; 6	„Vaikų nusikalstamumas“	Prevencinės veiklos skyrius → Vaikų nusikalstamumas

Automatinė tinklalapio analizė su duotu informaciniu tikslu buvo baigta per du žingsnius (1 lent.). Pirmame žingsnyje buvo pasirinkta aukščiausią reitingą gavusi nuorodą – „Viešosios tvarkos skyrius“. Taip pat šiame žingsnyje buvo pastebėta konkurencijos tarp nuorodų problema – nuorodų „Viešosios tvarkos skyrius“ ir „Teisinė informacija“ koreliacija buvo įvertinta kaip „stipri“. Antrame žingsnyje buvo pastebėta nesuprantamų nuorodų problema (visų nuorodų koreliacija buvo įvertinta kaip „silpna“ arba „nereikšminga“), todėl sistema baigė savo darbą. Darant išvadas galima pasakyti, kad „Viešosios tvarkos skyrius“ yra potencialiai klaidinanti nuoroda.



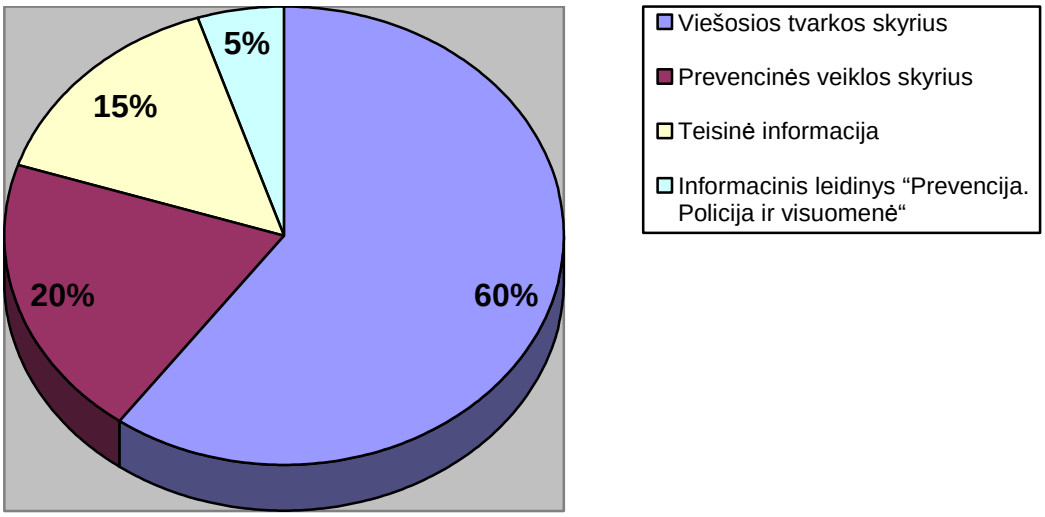
Žingsnis: 1
Konkurencijos tarp nuorodų problema.
Pasirinktos nuorodos koreliacija: Strong (0,88)

Žingsnis: 2
Nesuprantamų nuorodų problema (arba nuorodos nesusijusios su tikslu). Darbo pabaiga.

1 lent. Bandymas 1 – automatinės analizės rezultatai

Realių vartotojų paieškos rezultatai pavaizduoti 16 pav. 60% vartotojų pasirinko tokią pat pradinę nuorodą („Viešosios tvarkos skyrius“) kaip ir automatinė analizė. Pasirinkus šią nuorodą visi vartotojai baigė paiešką dėl nieko bendro su informaciniu tikslu neturinčių nuorodų („Geriausio pareigūno konkursas“, „Vasaros sezonas“, „Mobilūs padaliniai“, „Areštinės“,

„Archyvas“). Taip pat automatinė analizė pastebėjo konkurencijos tarp nuorodų problemą (nuorodos „Viešosios tvarkos skyrius“ ir „Teisinė informacija“), o 15% vartotojų pasirinko nuoroda „Teisinė informacija“, kas leidžia įsitikinti, kad problema iš tiesų egzistuoja. Tačiau automatinė analizė nepastebėjo kitos, labiau konkuruojančios nuorodos – „Prevencinės veiklos skyrius“.



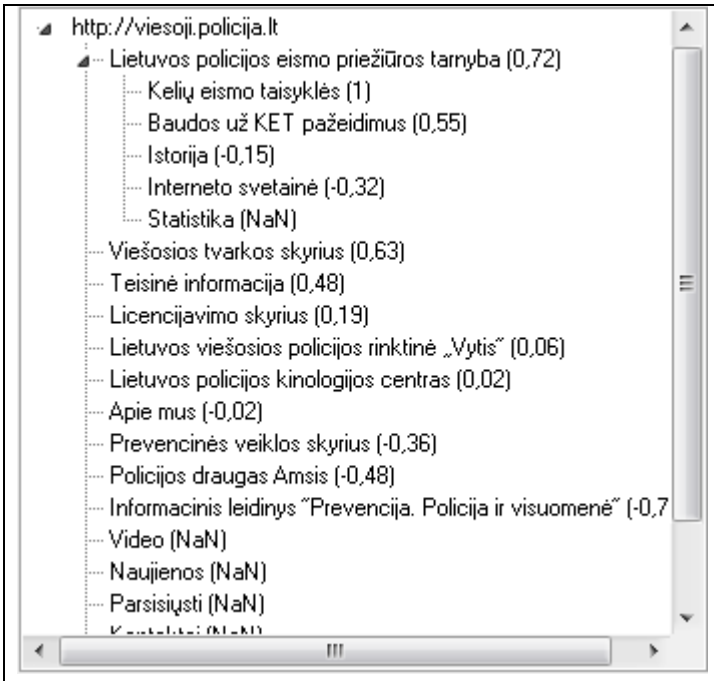
16 pav. Bandymas 1 – realių vartotojų paieškos rezultatai

Išvados: Automatinės analizės rezultatai su duotu informaciniu tikslu atitinka realių vartotojų paieškos rezultatus. Pastebėtos potencialios problemos dėl nuorodų iškilo realybėje. Tačiau automatinė analizė pastebėjo ne visas konkuruojančias nuorodas. Nuoroda „Prevencinės veiklos skyrius“ gavo reitingą 0.13 – nereikšminga koreliacija, tačiau realybėje ją pasirinko 20% vartotojų. Taip gavosi dėl to, kad semantinei erdvei sukurti panaudotame dokumente mažai rašoma apie prevencinės veiklos skyrių ir jo užduotis.

Bandymas 2

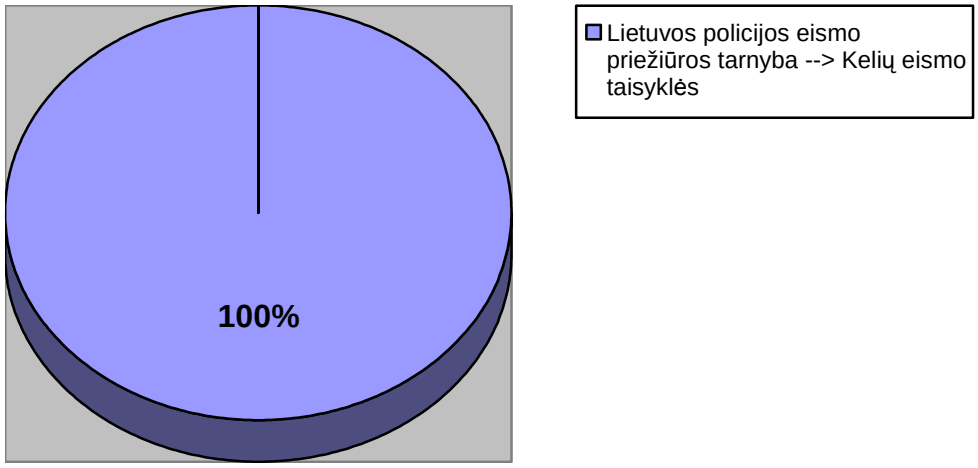
Analizuojamas tinklalapis	Bandyme dalyvavo	Semantinės erdvės dydis (žod./ dok.); Eilė	Informacinis tikslas	Teisingas informacijos paieškos kelias
http://viesoji.policija.lt	20 dalyvių	2278/190; 6	„Kelių eismo taisyklės“	Lietuvos policijos eismo priežiūros tarnyba → Kelių eismo taisyklės

Automatinė analizė turint nurodytą informacinį tikslą buvo baigta per tris žingsnius. Nebuvo pastebėta jokių potencialių problemų, galinčių iškilti pas tinklalapio lankytojus. Visuose žingsniuose pasirinktos nuorodos turi stiprią, arba labai stiprią koreliaciją. Analizės rezultatas pavaizduotas 2 lent.

	Žingsnis: 1 Pasirinktos nuorodos koreliacija: Strong (0,72) Žingsnis: 2 Pasirinktos nuorodos koreliacija: VeryStrong (1) Žingsnis: 3 Tikslas pasiektas. Darbo pabaiga.

2 lent. Bandymas 2 – automatinės analizės rezultatai

Realių vartotojų paieškos rezultatai pavaizduoti 17 pav. Visi 20 vartotojų (100%) pasirinko tokį pat informacijos paieškos kelią kaip ir buvo prognozuota atlikus automatinę analizę.



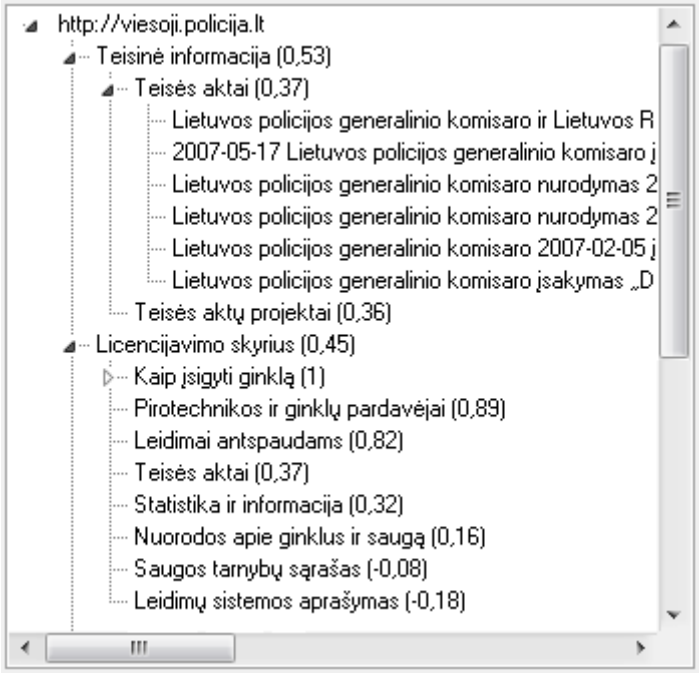
17 pav. Bandymas 2 – realių vartotojų paieškos rezultatai

Išvados: Automatinės analizės rezultatai su duotu informaciniu tikslu visiškai atitinka realių vartotojų paieškos rezultatus. Jokių potencialių problemų nebuvo pastebėta. Visi bandyme dalyvavę žmonės neturėjo dvejonių ieškant informacijos (formuojant paieškos kelią).

Bandymas 3

Analizuojamas tinklalapis	Bandyme dalyvavo	Semantinės erdvės dydis (žod./ dok.); Eilė	Informacinis tikslas	Teisingas informacijos paieškos kelias
http://viesoji.policija.lt	20 dalyvių	2278/190; 6	„Kaip įsigyti ginklą“	Licencijavimo skyrius → Kaip įsigyti ginklą

Atliekant automatinę analizę buvo pastebėta gan stipri konkurencijos tarp nuorodų problema. Šios konkuruojančios nuorodos, kaip matosi iš 3 lent., yra „Teisinė informacija“ bei „Licencijavimo skyrius“. Abiejų nuorodų reitingai yra panašūs. Automatinės analizės metu buvo pasirinkta nuoroda, turinti aukščiausią reitingą – „Teisinė informacija“ (0.53), tačiau tai nėra pradžioje apibrėžtas teisingas informacijos paieškos kelias. Antrame žingsnyje vėl buvo pastebėtos konkuruojančios nuorodos – „Teisės aktai“ ir „Teisės aktų projektai“. Po automatinės analizės pabaigos buvo rankiniu būdu pasirinkta nuorodos „Licencijavimo skyrius“ analizė, kurios rezultatai irgi matosi 3 lent. – tikslas pasiekiamas pasirinkus nuorodą „Kaip įsigyti ginklą“, turinčią labai stiprią koreliaciją.



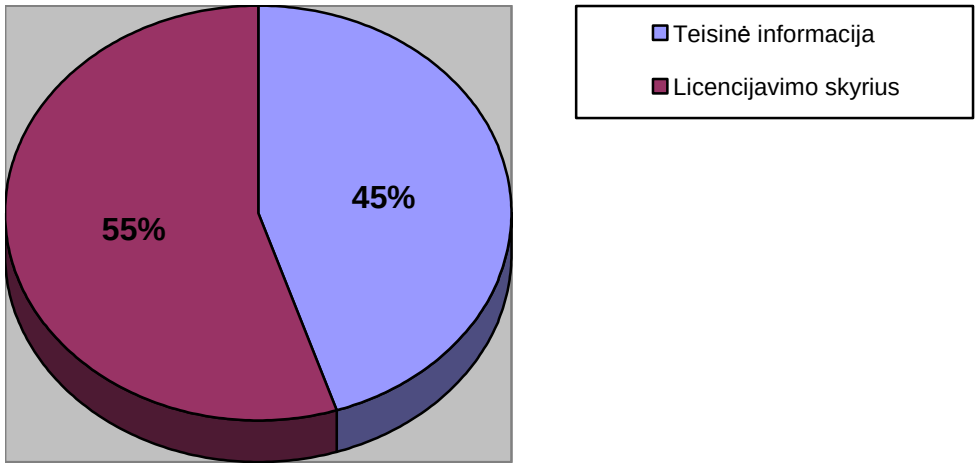
Žingsnis: 1
Konkurencijos tarp nuorodų problema. Pasirinktos nuorodos koreliacija: Medium (0,53)

Žingsnis: 2
Konkurencijos tarp nuorodų problema. Pasirinktos nuorodos koreliacija: Weak (0,37)

Žingsnis: 3
Nesuprantamų nuorodų problema (arba nuorodos nesusijusios su tikslu). Darbo pabaiga.

3 lent. Bandymas 3 – automatinės analizės rezultatai

Realių vartotojų paieškos rezultatai pavaizduoti 18 pav. Konkurencijos tarp nuorodų problema pasitvirtino – rezultatai pasiskirstė beveik per pusę. 11-a vartotojų pasirinko teisingą informacijos paieškos kelią, o 9-ių vartotojų paieškos kelias sutapo su automatinės analizės prognozuotu keliu.



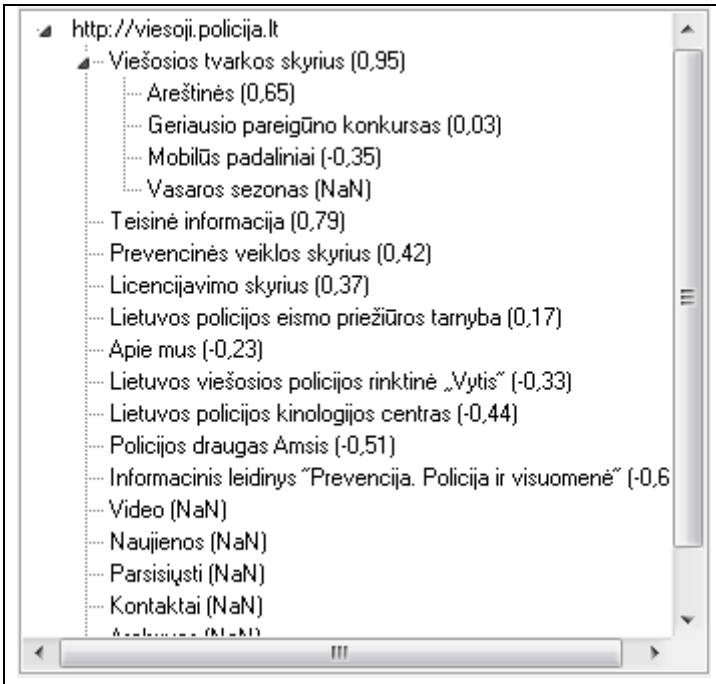
18 pav. Bandymas 3 – realių vartotojų paieškos rezultatai

Išvados: Automatinė analizė parodė, kad turint duotą informacinį tikslą vartotojai turės spėlioti kokią nuorodą pasirinkti. Tai pasitvirtino realybėje. Nors dauguma (55%) vartotojų pasirinko teisingą paieškos kelią ir pasiekė tikslą – tai yra nepakankamas procentas ir nuostoliai yra per dideli. Automatinė analizė teisingai pastebėjo potencialias problemas, susijusias su šiuo informaciniu tikslu.

Bandymas 4

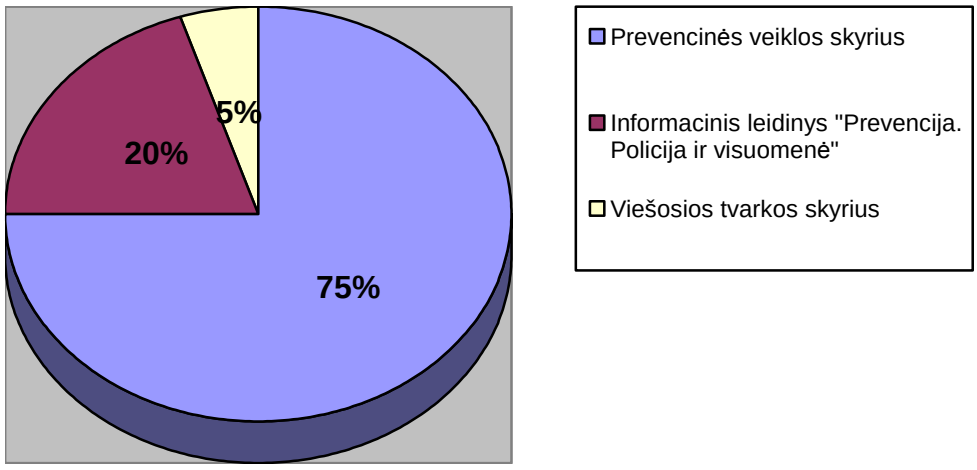
Analizuojamas tinklalapis	Bandyme dalyvavo	Semantinės erdvės dydis (žod./ dok.); Eilė	Informacinis tikslas	Teisingas informacijos paieškos kelias
http://viesoji.policija.lt	20 dalyvių	2278/190; 6	„Narkomanijos prevencija“	Prevencinės veiklos skyrius → Narkomanijos prevencija

Šis bandymas iliustruoja situaciją, kai tinklalapio automatinė analizė nepasiteisino. Kaip matosi iš 4 lent., nuoroda „Viešosios tvarkos skyrius“ gavo 0.95 reitingą, kas yra labai stipri koreliacija. Antrame žingsnyje buvo pasirinkta nuoroda „Areštinės“, turinti vidutinę koreliaciją. Taip yra dėl tekstinio dokumento, naudojamo semantinei erdvei sudaryti, konteksto. Šiame dokumente yra aprašyta apie narkomanija, tačiau nėra semantinio ryšio su prevencinės veiklos skyriumi. Todėl, remiantis turima semantine erdve, automatinė analizė pasirinko būtent toki informacijos paieškos kelią. Tačiau akivaizdu, kad žmogus, analizuojant nuorodas, turėtų pasirinkti nuorodą „Prevencinės veiklos skyrius“ dėl bendro žodžio „prevencija“.

	Žingsnis: 1 Pasirinktos nuorodos koreliacija: VeryStrong (0,95) Žingsnis: 2 Pasirinktos nuorodos koreliacija: Medium (0,65) Žingsnis: 3 Nuorodų nerasta. Darbo pabaiga.

4 lent. Bandymas 4 – automatinės analizės rezultatai

Realių vartotojų paieškos rezultatai pavaizduoti 19 pav. Realių vartotojų paieškos rezultatai visiškai nesutampa su automatinės analizės rezultatais. 95% (19 iš 20) dalyvių pasirinko nuorodas, kuriuose yra žodis „prevencija“. 75% iš jų pasiekė informacinį tikslą.



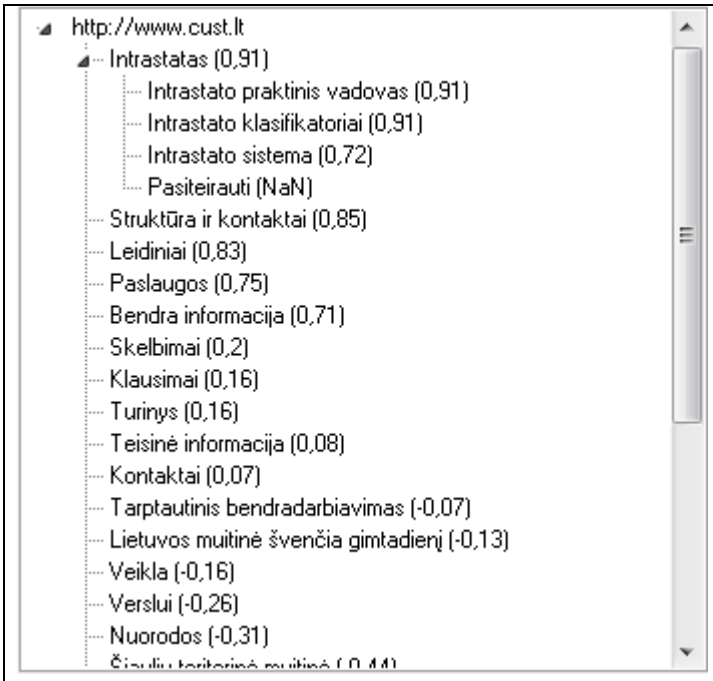
19 pav. Bandymas 4 – realių vartotojų paieškos rezultatai

Išvados: Automatinė analizė su duotu informaciniu tikslu nepasiteisino. Jos rezultatai neatitinka realių vartotojų paieškos rezultatų. Bandymai koreguoti semantinės erdvės parametrus nepadėjo, kas reiškia, kad semantinė erdvė turi per mažai informacijos apie policijos prevencinės veiklos skyrių.

Bandymas 5

Analizuojamas tinklalapis	Bandyme dalyvavo	Semantinės erdvės dydis (žod./ dok.); Eilė	Informacinis tikslas	Teisingas informacijos paieškos kelias
http://www.cust.lt	20 dalyvių	2137/44; 6	„Duomenų apie Lietuvos prekybą su Europos Sąjungos valstybėmis surinkimo sistemos naudojimas“	Intrastatas → Intrastato praktinis vadovas

Šis bandymas iliustruoja situaciją, kai informacinis tikslas visiškai nesusijęs su tinklalapio nuorodomis (žodine prasme). Šiame bandyme pasirinktas informacinis tikslas yra vienas iš Intrastato apibrėžimų, todėl, kaip matosi iš 5 lent., automatinės analizės rezultatas – tai sėkmingai pasiektas informacinis tikslas. Nuorodos „Intrastatas“ koreliacija buvo įvertinta kaip labai stipri (0.91).



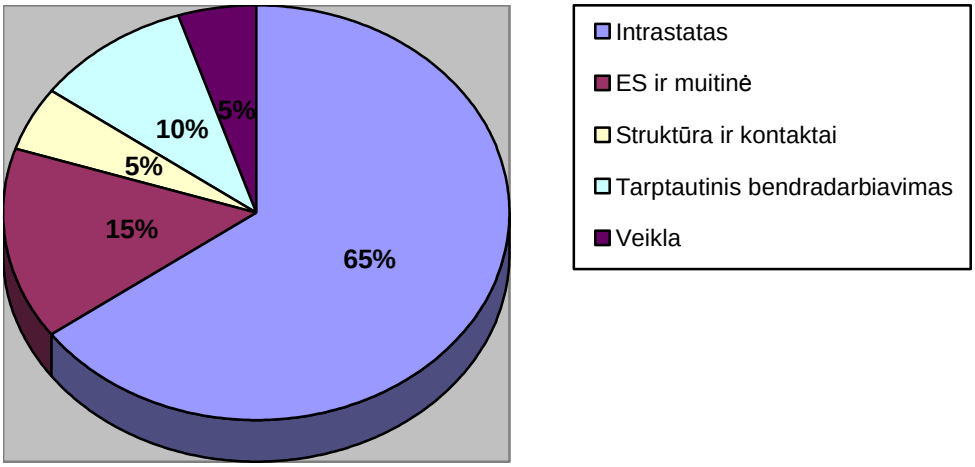
Žingsnis: 1
Pasirinktos nuorodos koreliacija: VeryStrong (0,91)

Žingsnis: 2
Konkurencijos tarp nuorodų problema.
Pasirinktos nuorodos koreliacija: VeryStrong (0,91)

Žingsnis: 3
Tikslas pasiektas. Darbo pabaiga.

5 lent. Bandymas 5 – automatinės analizės rezultatai

Realių vartotojų paieškos rezultatai pavaizduoti 20 pav. 65% dalyvių pasiekė informacinį tikslą, kiti 35% pasirinko skirtingas nuorodas. Po bandymo atlikimo paaiškėjo, kad visiems iš 35% dalyvių, pasirinkusių neteisingas nuorodas, trūko bazinių su muitine susijusių žinių, todėl jie nežinojo kas yra Intrastatas. Intrastatas buvo aprašytas tekstiniame dokumente, kuris buvo naudojamas semantinei erdvei sukurti, todėl automatinė analizė „turėjo bazinių žinių“ apie šį žodį.



20 pav. Bandymas 5 – realių vartotojų paieškos rezultatai

Išvados: Šis bandymas parodė, kad informacinis tikslas gali būti visiškai nesusijęs su tinklalapio nuorodomis žodžių prasme. Kitaip sakant, jis gali visiškai neturėti bendrų žodžių su tinklalapio nuorodomis. Duotas informacinis tikslas, susidedantis iš 11 žodžių, buvo puikiai atpažintas, todėl automatinė analizė pasirinko teisingą informacijos paieškos kelią ir pasiekė tikslą.

4 Rezultatai ir išvados

Darbo pradžioje apibrėžti laukiami rezultatai pasiekti - tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis yra sukurtas, automatinio vertinimo algoritmas realizuotas ir pritaikytas praktikoje kartu su vertinimo metodika. Informacijos paieškos teorija [PC99] buvo pritaikyta praktikoje.

Galima padaryti pagrindines darbo išvadas:

1. Semantinės erdvės sudarymui skirtų algoritmų analizė parodė, kad LSA algoritmas yra labiausiai tinkamas naudoti tinklalapio navigavimo asociacijų analizės ir prognozavimo modelyje atsižvelgiant į tikslumo ir naujų elementų įtraukimo į semantinę erdvę sudėtingumo aspektus. Prie LSA algoritmo rezultatų pritaikius koreliacijos koeficientų skaičiavimus atsiranda galimybė semantiškai lyginti pasirinktus žodžius ir dokumentus.
2. Modelio naudojimas rekomenduojamas, kai tinklalapio vertintojai turi galimybę pilnai išreikšti tinklalapio dalykinę sritį tekstinio dokumento pavidalu. Tada automatinės analizės rezultatai būna tikslūs ir sugeba pastebėti darbe iškeltas potencialias tinklalapio problemas, su kuriomis gali susidurti vartotojai - konkurencijos tarp nuorodų problemas, klaidinančių nuorodų problemas ir nesuprantamų nuorodų problemas.
3. Darbo pradžioje buvo tikimasi, kad tinklalapio navigavimo asociacijų analizės ir prognozavimo modelis galės būti taikomas bet kokiam tinklalapiui. Tačiau bandymai parodė, kad tikslios, tinklalapio dalykinę sritį atvaizduojančios semantinės erdvės kūrimas yra sudėtingas darbas. Užduotis sunkėja jei tinklalapio dalykinė sritis yra abstrakti (pavyzdžiui, bendro pobūdžio informacinis portalas). Tokiais atvejais vertintojams gali neapsimokėti investuoti laiką ir resursus į specifinės semantinės erdvės kūrimą, o gera semantinė erdvė yra būtina norint gauti korektiškus automatinės analizės rezultatus. Kitaip sakant, ne visada analizės rezultatai padengia pasiruošimo analizei pastangas.

5 Šaltinių sąrašas

[BBML] BlueBit .NET Matrix Library 4.0. [žiūrėta 2008-04-25]. Prieiga per Internetą: http://www.bluebit.gr/

[BRP01] R. Budiu, C. Royer, P. Pirolli. Modeling Information Scent: A Comparison of LSA, PMI and GLSA Similarity Measures on Common Tests and Corpora. Palo Alto Research Center, Palo Alto. 2001.

[CPP99] Chi, E. H., Pirolli, P., Pitkow, J. Using Information Scent to Model User Information Needs and Actions on the Web. Xerox Palo Alto Research Center. 1999.

[KG90] M. Kendall, J. D. Gibbons. Rank Correlation Methods. Charles Griffin Book Series. 5 edition. 1990.

[Kle99] J. Kleinberg. Authoritative sources in a hyperlinked environment. Journal of ACM (JASM). 46(1). 1999. pp. 53-75.

[LFL98] T. K Landauer, P. W. Foltz, D. Laham. An Introduction to Latent Semantic Analysis. Discourse Processes. 25. 1998. pp. 259-284.

[MLF+05] I. Matveeva, G. Levow, A. Farahat, C. Royer. Terms representation with generalized latent semantic analysis. Palo Alto Research Center, Palo Alto. 2005.

[MVS] Microsoft Visual Studio 2005. [žiūrėta 2008-04-20]. Prieiga per Internetą: http://msdn.microsoft.com/vstudio/

[OC00] C. Olston, E. H. Chi. ScentTrails: Integrating Browsing and Searching on the World Wide Web. ACM Transactions on Computer-Human Interaction. 10(3). 2003. pp. 1–21.

[PC99] P. Pirolli, S. K. Card. Information foraging. Psychological Review. 106(4). 1999. pp. 643-675.

[Pir00] P. Pirolli. A Web site user model should at least predict something about users. Internetworking. 3(1). 2000. pp 78-89.

[Pir05] P. Pirolli. Rational analyses of information foraging on the web. Cognitive science. 29(3). 2005. pp. 343-373.

[PP99] J. Pitkow, P. Pirolli. Mining longest repeated subsequences to predict World Wide Web surfing. Proceedings of the USENIX Conference on Internet. 1999.

[RB] JG Soft RegexBuddy. [žiūrėta 2008-05-01]. Prieiga per Internetą: http://www.regexbuddy.com/

[SDR+98] S. Chakrabarti, B. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, J. Kleinberg Automatic resource compilation by analyzing hyperlink structure and associated text. In Proc. of the 7th International World Wide Web Conference (WWW7). Brisbane, Australia. 1998. pp. 65-74.

[SP] WebCriteria SiteProfile. [žiūrėta 2007-12-20]. Prieiga per Internetą: http://www.webcriteria.com

[SPB04] J. Spool, C. Perfetti, D. Brittan. Designing for the scent of information. UI Engineering. 2004.

[Tur01] P. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. Proceedings of the Twelfth European Conference on Machine Learning. Freiburg, Germany. 2001. pp. 491-502.

[WG03] M. B. W. Wolfe and S. R. Goldman. Use of Latent Semantic Analysis for Predicting Psychological Phenomena: Two Issues and Proposed Solutions. Behavior research methods, instruments & computers. 35(1). 2003. pp. 22-31.